ANNAMACHARYA
INSTITUTE OF TECHNOLOGY AND SCIENCES
(AUTONOMOUS)

Approved by AICTE, New Delhi & Permanent Affiliation to JNTUA, Anantapur.

Three B. Tech Programmes (CSE , ECE & CE) are accredited by NBA, New Delhi,Accredited by NAAC with 'A' Grade , Bangalore.

A-grade awarded by AP Knowledge Mission. Recognized under sections 2(f) & 12(B) of UGC Act 1956.

Venkatapuram Village, Renigunta Mandal, Tirupati, Andhra Pradesh-517520.

Department of Computer Science and Engineering

Academic Year 2023-24

IV. B.Tech I Semster

Principles of Data Science

(Common to CSE, CIC)

(20APE0519/20APE3618)

Prepared By

Mr. V SambaSiva., M.Tech(Ph.D).
Mrs. C Malathi., M.Tech(Ph.D).
Assistant Professor
Department of CSE, AITS

UNIT-1

Introduction to Data Science

What is data science?

Data science is the art and science of acquiring knowledge through data. Whenever we use the word "data", we refer to a collection of information in either an organized or unorganized format:

- **Organized data:** This refers to data that is sorted into a row/column structure, where every row represents a single observation and the columns represent the characteristics of that observation.
- **Unorganized data:** This is the type of data that is in the free form, usually text or raw audio/signals that must be parsed further to become organized. Whenever you open Excel (or any other spreadsheet program), you are looking at a blank row/column structure waiting for organized data. These programs don't do well with unorganized data. For the most part, we will deal with organized data as it is the easiest to glean insight from, but we will not shy away from looking at raw text and methods of processing unorganized forms of data.

Data science is all about how we take data, use it to acquire knowledge, and then use that knowledge to do the following:

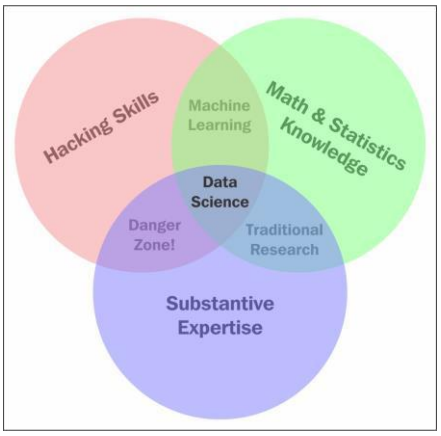
- Make decisions
- Predict the future
- Understand the past/present
- Create new industries/products

The data science Venn diagram

It is a common misconception that only those with a PhD or geniuses can understand the math/programming behind data science. This is absolutely false. Understanding data science begins with three basic areas:

- **Math/statistics:** This is the use of equations and formulas to perform analysis
- **Computer programming:** This is the ability to use code to create outcomes on the computer
- **Domain knowledge:** This refers to understanding the problem domain (medicine, finance, social science, and so on)

The following Venn diagram provides a visual representation of how the three areas of data science intersect:



Those with hacking skills can conceptualize and program complicated algorithms using computer languages. Having a *Math & Statistics Knowledge* base allows you to theorize and evaluate algorithms and tweak the existing procedures to fit specific situations. Having *Substantive Expertise* (domain expertise) allows you to apply concepts and results in a meaningful and effective way.

Structured versus unstructured data

The distinction between structured and unstructured data is usually the first question you want to ask yourself about the *entire* dataset. The answer to this question can mean the difference between needing three days or three weeks of time to perform a proper analysis.

The basic breakdown is as follows (this is a rehashed definition of organized and unorganized data in the first chapter):

- **Structured (organized) data**: This is data that can be thought of as observations and characteristics. It is usually organized using a table method (rows and columns).
- **Unstructured (unorganized) data**: This data exists as a free entity and does not follow any standard organization hierarchy

Here are a few examples that could help you differentiate between the two:

- Most data that exists in text form, including server logs and Facebook posts, is *unstructured*
- Scientific observations, as recorded by careful scientists, are kept in a very neat and organized (*structured*) format
- A genetic sequence of chemical nucleotides (for example, ACGTATTGCA) is *unstructured* even if the order of the nucleotides matters as we cannot form descriptors of the sequence using a row/column format without taking a further look

Structured data is generally thought of as being much easier to work with and analyze. Most statistical and machine learning models were built with structured data in mind and cannot work on the loose interpretation of unstructured data. The natural row and column structure is easy to digest for human and machine eyes. So why even talk about unstructured data? Because it is so common! Most estimates place unstructured data as 80-90% of the world's data. This data exists in many

forms and for the most part, goes unnoticed by humans as a potential source of data. Tweets, e-mails, literature, and server logs are generally unstructured forms of data.

Example of data preprocessing

When looking at text data (which is almost always considered unstructured), we have many options to transform the set into a structured format. We may do this by applying new characteristics that describe the data. A few such characteristics are as follows:

- Word/phrase count
- The existence of certain special characters
- The relative length of text
- Picking out topics

I will use the following tweet as a quick example of unstructured data, but you may use any unstructured free-form text that you like, including tweets and Facebook posts.

"This Wednesday morn, are you early to rise? Then look East. The Crescent Moon joins Venus & Saturn. Afloat in the dawn skies."

Word/phrase counts

We may break down a tweet into its word/phrase count. The word this appears in the tweet once, as does every other word. We can represent this tweet in a structured format, as follows, thereby converting the unstructured set of words into a row/column format:

	this	wednesday	morn	are	this wednesday
Word Count	1	1	1	1	1

Presence of certain special characters

We may also look at the presence of special characters, such as the question mark and exclamation mark. The appearance of these characters might imply certain ideas about the data that are otherwise difficult to know. For example, the fact that this tweet contains a question mark might strongly imply that this tweet contains a question for the reader. We might append the preceding table with a new column, as shown:

	this	wednesday	morn	are	this wednesday	?
Word Count	1	1	1	1	1	1

Relative length of text

This tweet is 121 characters long.

len("This Wednesday morn, are you early to rise? Then look East. The Crescent Moon joins Venus & Saturn. Afloat in the dawn skies.") # get the length of this text (number of characters for a string)
# 121
The average tweet, as discovered by analysts, is about 30 characters in length. So, we might impose a new characteristic, called **relative length**, (which is the length of the tweet divided by the average length), telling us the length of this tweet as compared to the average tweet. This tweet is actually 4.03 times longer than the average tweet, as shown:

$$\frac{121}{30} = 4.03$$

We can add yet another column to our table using this method:

	this	wednesday	morn	are	this wednesday	?	Relative length
Word Count	1	1	1	1	1	1	4.03

Picking out topics

We can pick out some topics of the tweet to add as columns. This tweet is about astronomy, so we can add another column, as illustrated:

	this	wednesday	morn	are	this wednesday	?	Relative length	Topic
Word Count	1	1	1	1	1	1	4.03	astronomy

Quantitative versus qualitative data

When you ask a data scientist, "what type of data is this?", they will usually assume that you are asking them whether or not it is mostly quantitative or qualitative. It is likely the most common way of describing the *specific* characteristics of a dataset.
For the most part, when talking about quantitative data, you are *usually* (not always) talking about a structured dataset with a strict row/column structure (because we don't assume unstructured data even *has* any characteristics). All the more reason why the preprocessing step is so important.
These two data types can be defined as follows:
• **Quantitative data**: This data can be described using numbers, and basic mathematical procedures, including addition, are possible on the set.
• **Qualitative data**: This data cannot be described using numbers and basic mathematics. This data is generally thought of as being described using "natural" categories and language.
Example – coffee shop data
Say that we were processing observations of coffee shops in a major city using the following five descriptors (characteristics):

Data: Coffee Shop

- Name of coffee shop
- Revenue (in thousands of dollars)
- Zip code
- Average monthly customers
- Country of coffee origin

- **Name of coffee shop – Qualitative**

The name of a coffee shop is not expressed as a number and we cannot perform math on the name of the shop.

- **Revenue – Quantitative**

How much money a cafe brings in can definitely be described using a number. Also, we can do basic operations such as adding up the revenue for 12 months to get a year's worth of revenue.

- **Zip code – Qualitative**

This one is tricky. A zip code is always represented using numbers, but what makes it qualitative is that it does not fit the second part of the definition of quantitative—we cannot perform basic mathematical operations on a zip code. If we add together two zip codes, it is a nonsensical measurement. We don't necessarily get a new zip code and we definitely don't get "double the zip code".

- **Average monthly customers – Quantitative**

Again, describing this factor using numbers and addition makes sense. Add up all of your monthly customers and you get your yearly customers.

- **Country of coffee origin – Qualitative**

We will assume this is a very small café with coffee from a single origin. This country is described using a name (Ethiopian, Colombian), and not numbers.

The four levels of data

It is generally understood that a specific characteristic (feature/column) of structured data can be broken down into one of four levels of data. The levels are:

- The nominal level
- The ordinal level
- The interval level
- The ratio level

As we move down the list, we gain more structure and, therefore, more returns from our analysis. Each level comes with its own accepted practice in measuring the center of the data. We usually think of the mean/average as being an acceptable form of center, however, this is only true for a specific type of data.

The nominal level

The first level of data, the *nominal* level, (which also sounds like the word name) consists of data that is described purely by name or category. Basic examples include gender, nationality, species, or yeast strain in a beer. They are not described by numbers and are therefore qualitative. The following are some examples:

- A type of animal is on the nominal level of data. We may also say that if you are a chimpanzee, then you belong to the mammalian class as well.

• A part of speech is also considered on the nominal level of data. The word *she* is a pronoun, and it is also a *noun*.

Of course, being qualitative, we cannot perform any quantitative mathematical operations, such as addition or division. These would not make any sense.

Mathematical operations allowed

We cannot perform mathematics on the nominal level of data except the basic equality and set membership functions, as shown in the following two examples:

• Being a tech entrepreneur is the same as being in the tech industry, but not vice versa

• A figure described as a square falls under the description of being a rectangle, but not vice versa

Measures of center

A **measure of center** is a number that describes what the data *tends to*. It is sometimes referred to as the *balance point* of the data. Common examples include the mean, median, and mode.

In order to find the *center* of nominal data, we generally turn to the *mode* (the most common element) of the dataset. For example, look back at the WHO alcohol consumption data. The most common continent surveyed was Africa, making that a possible choice for the *center* of the continent column.

Measures of center such as the mean and median do not make sense at this level as we cannot order the observations or even add them together.

What data is like at the nominal level

Data at the nominal level is mostly categorical in nature. Because we generally can only use words to describe the data, it can be lost in translation among countries, or can even be misspelled.

While data at this level can certainly be useful, we must be careful about what insights we may draw from them. With only the mode as a basic measure of center, we are unable to draw conclusions about an *average* observation. This concept does not exist at this level. It is only at the next level that we may begin to perform true mathematics on our observations.

The ordinal level

The nominal level did not provide us with much flexibility in terms of mathematical operations due to one seemingly unimportant fact—we could not order the observations in any natural way. Data in the *ordinal* level provides us with a rank order, or the means to place one observation before the other; however, it does not provide us with relative differences between observations, meaning that while we may order the observations from first to last, we cannot add or subtract them to get any real meaning.

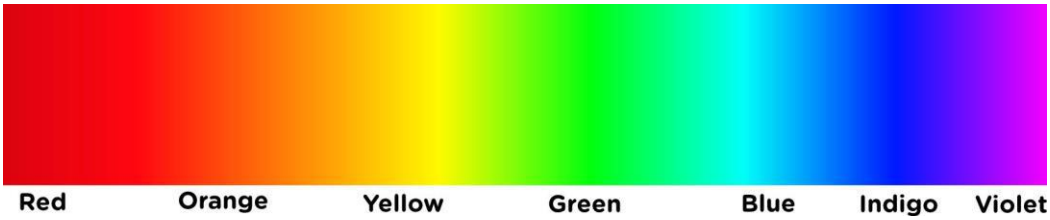
Mathematical operations allowed

We are allowed much more freedom on this level in mathematical operations. We inherit all mathematics from the ordinal level (equality and set membership) and we can also add the following to the list of operations allowed in the nominal level:

• Ordering

• Comparison

Ordering refers to the natural order provided to us by the data; however, this can be tricky to figure out sometimes. When speaking about the spectrum of visible light, we can refer to the names of colors—red, orange, yellow, green, blue, indigo, and violet. Naturally, as we move from left to right, the light is gaining energy and other properties. We may refer to this as a natural order.



Comparisons are another new operation allowed at this level. At the ordinal level, it would not make sense to say that one country was naturally better than another or that one part of speech is worse than another. At the ordinal level, we can make these comparisons

Measures of center

At the ordinal level, the **median** is usually an appropriate way of defining the center of the data. The mean, however, would be impossible because division is not allowed at this level. We can also use the mode like we could at the nominal level.

We will now look at an example of using the median:

Imagine you have conducted a survey among your employees asking "how happy are you to be working here on a scale from 1-5", and your results are as follows:

5, 4, 3, 4, 5, 3, 2, 5, 3, 2, 1, 4, 5, 3, 4, 4, 5, 4, 2, 1, 4, 5, 4, 3, 2, 4, 4, 5, 4, 3, 2, 1

Let's use Python to find the median of this data. It is worth noting that most people would argue that the mean of these scores would work just fine. The reason that the mean would not be as mathematically viable is because if we subtract/add two scores, say a score of four minus a score of two, the difference of two does not really mean anything. If addition/subtraction among the scores doesn't make sense, the mean won't make sense either.

```
import numpy
results = [5, 4, 3, 4, 5, 3, 2, 5, 3, 2, 1, 4, 5, 3, 4, 4, 5, 4, 2, 1, 4, 5, 4, 3, 2, 4, 4, 5, 4, 3, 2, 1]
sorted_results = sorted(results)
print sorted_results
'''
[1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5]
'''
print numpy.mean(results) # == 3.4375
print numpy.median(results) # == 4.0
```

The interval level

Now we are getting somewhere interesting. At the interval level, we are beginning to look at data that can be expressed through very quantifiable means, and where much

more complicated mathematical formulas are allowed. The basic difference between the ordinal level and the interval level is, well, just that—difference.

Data at the interval level allows meaningful subtraction between data points.

Mathematical operations allowed

We can use all the operations allowed on the lower levels (ordering, comparisons, and so on), alongwith two other notable operations:

• Addition

• Subtraction

The allowance of these two operations allows us to talk about data at this level in a whole new way

Measures of center

At this level, we can use the median and mode to describe this data; however, usually the most accurate description of the center of data would be the **arithmetic mean**, more commonly referred to as, simply, "the mean". Recall that the definition of the mean requires us to add together all the measurements. At the previous levels, addition was meaningless; therefore, the mean would have lost extreme value. It is only at the interval level and above that the arithmetic mean makes sense.

We will now look at an example of using the mean.

Suppose we look at the temperature of a fridge containing a pharmaceutical company's new vaccine. We measure the temperate every hour with the following data points (in Fahrenheit):

31, 32, 32, 31, 28, 29, 31, 38, 32, 31, 30, 29, 30, 31, 26

Using Python again, let's find the mean and median of the data:

```
import numpy
temps = [31, 32, 32, 31, 28, 29, 31, 38, 32, 31, 30, 29, 30, 31, 26]
print numpy.mean(temps) # == 30.73 print numpy.median(temps) # == 31.0
```

Measures of variation

This is something new that we have not yet discussed. It is one thing to talk about the center of the data but, in data science, it is also very important to mention how "spread out" the data is. The measures that describe this phenomenon are called **measures of variation**.

Standard deviation

Arguably, standard deviation is the most common measure of variation of data at the interval level and beyond. The standard deviation can be thought of as the "average distance a data point is at from the mean". While this description is technically and mathematically incorrect, it is a good way to think about it. The formula for standard deviation can be broken down into the following steps:

1. Find the mean of the data.
2. For each number in the dataset, subtract it from the mean and then square it.
3. Find the average of each square difference.
4. Take the square root of the number obtained in step three. This is the

standard deviation.
Notice how, in the steps, we do actually take an arithmetic mean as one of the steps.
For example, look back at the temperature dataset. Let's find the standard deviation of the dataset using Python:

```
import numpy
temps = [31, 32, 32, 31, 28, 29, 31, 38, 32, 31, 30, 29, 30, 31, 26]
mean = numpy.mean(temps) # == 30.73 squared_differences = []
# empty list o squared differences
for temperature in temps:
difference = temperature - mean # how far is the point from the mean
squared_difference = difference**2 # square the difference
squared_differences.append(squared_difference) # add it to our list
average_squared_difference = numpy.mean(squared_differences) # This number is also called the "Variance"
standard_deviation = numpy.sqrt(average_squared_difference) # We did it!
print standard_deviation # == 2.5157
```

The ratio level

Finally, we will take a look at the ratio level. After moving through three different levels with differing levels of allowed mathematical operations, the ratio level proves to be the strongest of the four.
Not only can we define order and difference, the ratio level allows us to multiply and divide as well. This might seem like not much to make a fuss over but it changes almost everything about the way we view data at this level.

Measures of center

The arithmetic mean still holds meaning at this level, as does a new type of mean called the geometric mean. This measure is generally not used as much even at the ratio level, but is worth mentioning. It is the square root of the product of all the values.
For example, in our fridge temperature data, we can calculate the geometric mean as shown here:

```
import numpy
temps = [31, 32, 32, 31, 28, 29, 31, 38, 32, 31, 30, 29, 30, 31, 26]
num_items = len(temps)
product = 1.
for temperature in temps:
product *= temperature
geometric_mean = product**(1./num_items)
print geometric_mean # == 30.634
```

The five steps

The five essential steps to perform data science are as follows:

1. Asking an interesting question
2. Obtaining the data
3. Exploring the data
4. Modeling the data
5. Communicating and visualizing the results

First, let's look at the five steps with reference to the big picture.

Ask an interesting question

This is probably my favorite step. As an entrepreneur, I ask myself (and others) interesting questions every day. I would treat this step as you would treat a brainstorming session. Start writing down questions regardless of whether or not you think the data to answer these questions even exists. The reason for this is twofold. First off, you don't want to start biasing yourself even before searching for data. Secondly, obtaining data might involve searching in both public and private locations and, therefore, might not be very straightforward. You might ask a question and immediately tell yourself "Oh, but I bet there's no data out there that can help me," and cross it off your list. Don't do that! Leave it on your list.

Obtain the data

Once you have selected the question you want to focus on, it is time to scour the world for the data that might be able to answer that question. As mentioned before, the data can come from a variety of sources; so, this step can be very creative!

Explore the data

Once we have the data, we use the lessons learned in *Chapter 2*, *Types of Data*, of this book and begin to break down the types of data that we are dealing with. This is a pivotal step in the process. Once this step is completed, the analyst generally has spent several hours learning about the domain, using code or other tools to manipulate and explore the data, and has a very good sense of what the data might be trying to tell them.

Model the data

This step involves the use of statistical and machine learning models. In this step, we are not only fitting and choosing models, we are implanting mathematical validation metrics in order to quantify the models and their effectiveness.

Communicate and visualize the results

This is arguably the most important step. While it might seem obvious and simple, the ability to conclude your results in a digestible format is much more difficult than it seems. We will look at different examples of cases when results were communicated poorly and when they were displayed very well.

Explore the data

The process of exploring data is not defined simply. It involves the ability to recognize the different types of data, transform data types, and use code to systemically improve the quality of the entire dataset to prepare it for the modeling

stage. In order to best represent and teach the art of exploration, I will present several different datasets and use the python package pandas to explore the data. Along the way, we will run into different tips and tricks for how to handle data.

Basic questions for data exploration

When looking at a new dataset, whether it is familiar to you or not, it is important to use the following questions as guidelines for your preliminary analysis:

- **Is the data organized or not?**

We are checking for whether or not the data is presented in a row/column structure. For the most part, data will be presented in an organized fashion. In this book, over 90% of our examples will begin with organized data. Nevertheless, this is the most basic question that we can answer before diving any deeper into our analysis.

A general rule of thumb is that if we have unorganized data, we want to transform it into a row/column structure. For example, earlier in this book, we looked at ways to transform text into a row/column structure by counting the number of words/phrases.

- **What does each row represent?**

Once we have an answer to how the data is organized and are now looking at a nice row/column based dataset, we should identify what each row actually represents. This step is usually very quick, and can help put things in perspective much more quickly.

- **What does each column represent?**

We should identify each column by the level of data and whether or not it is quantitative/qualitative, and so on. This categorization might change as our analysis progresses, but it is important to begin this step as early as possible.

- **Are there any missing data points?**

Data isn't perfect. Sometimes we might be missing data because of human or mechanical error. When this happens, we, as data scientists, must make decisions about how to deal with these discrepancies.

- **Do we need to perform any transformations on the columns?**

Depending on what level/type of data each column is at, we might need to perform certain types of transformations. For example, generally speaking, for the sake of statistical modeling and machine learning, we would like each column to be numerical. Of course, we will use Python to make any and all transformations.

All the while, we are asking ourselves the overall question, *what can we infer from the preliminary inferential statistics?* We want to be able to understand our data a bit more than when we first found it.

Dataset 1 – Yelp

The first dataset we will look at is a public dataset made available by the restaurant review site, Yelp. All personally identifiable information has been removed. Let's read in the data first, as shown here:

```
import pandas as pd
yelp_raw_data = pd.read_csv("yelp.csv") yelp_raw_data.head()
```

A quick recap of what the preceding code does:

- Import the pandas package and nickname it as pd.
- Read in the .csv from the Web; call is yelp_raw_data.

• Look at the head of the data (just the first few rows).

	business_id	date	review_id	stars	text	type	user_id	cool	useful	funny
0	9yKzy9PApeiPPOUJEtnvkg	2011-01-26	fWKvX83p0-ka4JS3dc6E5A	5	My wife took me here on my birthday for breakf...	review	rLtl8ZkDX5vH5nAx9C3q5Q	2	5	0
1	ZRJwVLyzEJq1VAihDhYiow	2011-07-27	IjZ33sJrzXqU-0X6U8NwyA	5	I have no idea why some people give bad review...	review	0a2KyEL0d3Yb1V6aivbIuQ	0	0	0
2	6oRAC4uyJCsJl1X0WZpVSA	2012-06-14	IESLBzqUCLdSzSqm0eCSxQ	4	love the gyro plate. Rice is so good and I als...	review	0hT2KtfLiobPvh6cDC8JQg	0	1	0
3	_1QQZuf4zZOyFCvXc0o6Vg	2010-05-27	G-WvGaISbqqaMHlNnByodA	5	Rosie, Dakota, and I LOVE Chaparral Dog Park!!...	review	uZetl9T0NcROGOyFfughhg	1	2	0
4	6ozycU1RpktNG2-1BroVtw	2012-01-05	1uJFq2r5QfJG_6ExMRCaGw	5	General Manager Scott Petello is a good egg!!!...	review	vYmM4KTsC8ZfQBg-j5MWkw	0	0	0

Is the data organized or not?

• Because we have a nice row/column structure, we can conclude that this data seems pretty organized.

What does each row represent?

• It seems pretty obvious that each row represents a user giving a review of a business. The next thing we should do is to examine each row and label it by the type of data it contains. At this point, we can also use python to figure out just how big our dataset is. We can use the shape quality of a Dataframe to find this out, as shown:

yelp_raw_data.shape # (10000,10)

• It tells us that this dataset has 10000 rows and 10 columns. Another way to say this is that this dataset has 10,000 observations and 10 characteristics.

What does each column represent?

Note that we have 10 columns:

• **business_id:** This is likely a unique identifier for the business the review is for. This would be at the **nominal level** because there is no natural order to this identifier.

• **date:** This is probably the date at which the review was posted. Note that it seems to be only specific to the day, month, and year. Even though time is usually considered continuous, this column would likely be considered discrete and at the **ordinal level** because of the natural order that dates have.

• **review_id:** This is likely a unique identifier for the review that each post represents. This would be at the **nominal level** because, again, there is no natural order to this identifier.

• **stars:** From a quick look (don't worry; we will perform some further analysis soon), we can see that this is an ordered column that represents what the reviewer gave the restaurant as a final score. This is ordered and qualitative; so, this is at the **ordinal level**.

• **text:** This is likely the raw text that each reviewer wrote. As with most text, we place this at the **nominal level**.

• **type:** In the first five columns, all we see is the word *review*. This might be a column that identifies that each row is a review, implying that there might be another type of row other than a review. We will take a look at this later. We place this at the **nominal level**.

• **user_id:** This is likely a unique identifier for the user who is writing the review. Just like the other unique IDs, we place this data at the **nominal level**.

Are there any missing data points?

• Perform an isnull operation. For example if your dataframe is called awesome_dataframe then try the python command awesome_dataframe. isnull().sum()which will show the number of missing values in each column.

Do we need to perform any transformations on the columns?

• At this point, we are looking for a few things. For example, will we need to change the scale of some of the quantitative data, or do we need to create dummy variables for the qualitative variables? As this dataset has only qualitative columns, we can only focus on transformations at the ordinal and nominal scale.

Before starting, let's go over some quick terminology for pandas, the python data exploration module.

UNIT-2

Basic Mathematics

Basic symbols and terminology

First, let's take a look at the most basic symbols that are used in the mathematical process as well as some more subtle notations used by data scientists.

Vectors and matrices

A vector is defined as an object with both magnitude and direction. This definition, however, is a bit complicated for our use. For our purpose, a vector is simply a 1-dimensional array representing a series of numbers. Put in another way, a vector is a list of numbers.

It is generally represented using an arrow or bold font, as shown:

$$\vec{x} \quad or \quad x$$

Vectors are broken into components, which are individual members of the vector. We use index notations to denote the element that we are referring to, as illustrated:

$$\vec{x} = \begin{pmatrix} 3 \\ 6 \\ 8 \end{pmatrix}$$

If $x_1 = 3$ then

In Python, we can represent arrays in many ways. We could simply use a Python list to represent the preceding array:

```
x = [3, 6, 8]
```

it is better to use the numpy array type to represent arrays, as shown,

because it gives us much more utility when performing vector operations:

```
import numpy as np
x = np.array([3, 6, 8])
```

Regardless of the Python representation, vectors give us a simple way of storing multiple dimensions of a single data point/observation.

Consider that we measure the average satisfaction rating (0-100) of employees for three departments of a company as being 57 for HR, 89 for engineering, and 94 for management. We can represent this as a vector with the following formula:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 57 \\ 89 \\ 94 \end{pmatrix}$$

This vector holds three different bits of information about our data. This is the perfect use of a vector in data science.

The matrix is our generalization of the Pandas Dataframe. It is arguably one of the most important mathematical objects in our toolkit. It is used to hold organized information, in our case, data.

Revisiting our previous example, let's say we have three offices in different locations, each with the same three departments: HR, engineering, and management. We could make three different vectors, each holding a different office's satisfaction scores, as shown:

$$x = \begin{pmatrix} 57 \\ 89 \\ 94 \end{pmatrix}, y = \begin{pmatrix} 67 \\ 87 \\ 84 \end{pmatrix}, z = \begin{pmatrix} 65 \\ 98 \\ 60 \end{pmatrix}$$

Let's make a matrix where each row represents a different department and each column represents a different office, as shown:

	Office 1	Office 2	Office 3
HR	57	67	65
Engineering	89	87	98
Management	94	84	60

This is much more natural. Now, let's strip away the labels, and we are left with a matrix!

$$X = \begin{pmatrix} 57 & 67 & 65 \\ 89 & 87 & 98 \\ 94 & 84 & 60 \end{pmatrix}$$

ARITHMETIC SYMBOLS

Summation

The uppercase sigma symbol is a universal symbol for addition. Whatever is to the right of the sigma symbol is usually something iterable, meaning that we can go over it one by one (for example, a vector).

For example, let's create the representation of a vector:

X = [1, 2, 3, 4, 5]

To find the sum of the content, we can use the following formula:

$$\sum x_i = 15$$

In Python, we can use the following formula:

```
sum(x) # == 15
```

For example, the formula for calculating the mean of a series of numbers is quite common. If we have a vector (x) of length n, the mean of the vector can be calculated as follows:

$$mean = \frac{1}{n}\sum x_i$$

This means that we will add up each element of *x*, denoted by x_i , and then multiply the sum by *1/n*, otherwise known as dividing by *n*, the length of the vector.

Proportional

The lowercase alpha symbol α represents values that are proportional to each other. This means that as one value changes, so does the other. The direction in which the values move depends on how the values are proportional. Values can either vary directly or indirectly. If values vary directly, they both move in the same direction.

Consider the following examples:

- The sales of a company vary directly with the number of customers. This can be written as $Sales \alpha Customers$.
- Gas prices vary (usually) indirectly with oil availability, meaning that as the availability of oil goes down (it's more scarce), gas prices will go up. This can be denoted as $Gas \alpha Oil\ Availability$.

Dot product

The dot product is an operator like addition and multiplication. It is used to combine two vectors, as shown:

$$\begin{pmatrix}3\\7\end{pmatrix}\cdot\begin{pmatrix}9\\5\end{pmatrix} = 3*9+7*5 = 62$$

Consider that, on a scale of 1-5, a customer loves comedies, hates romantic movies, and is alright with action movies. We might represent this as follows:

$$\begin{pmatrix}5\\1\\3\end{pmatrix}$$

Here:

- 5 denotes the love for comedies,
- 1 is the hatred for romantic
- 3 is the indifference of action

Now, let's assume that we have two new movies, one of which is a romantic comedy and the other is a funny action movie. The movies would have their own vector of qualities, as shown:

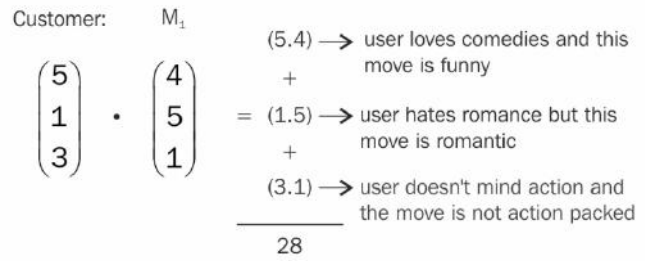
$$m_1 = \begin{pmatrix}4\\5\\1\end{pmatrix} \text{ and } m_2 = \begin{pmatrix}5\\1\\5\end{pmatrix}$$

Here, is our romantic comedy and is our funny action movie.

In order to make a recommendation, we will apply the dot product between the customer's preferences for each movie. The higher value will win and, therefore, will be recommended to the user.
Let's compute the recommendation score for each movie. For movie 1, we want to compute:

$$\begin{pmatrix}5\\1\\3\end{pmatrix}.\begin{pmatrix}4\\5\\1\end{pmatrix}$$

We can think of this problem as such:



The answer we obtain is 28, but what does this number mean? On what scale is it?
Well, the best score anyone can ever get is when all values are 5, making the outcome as follows:

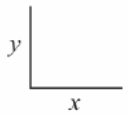
$$\begin{pmatrix}5\\5\\5\end{pmatrix}.\begin{pmatrix}5\\5\\5\end{pmatrix}=5^2+5^2+5^2=75$$

The lowest possible score is when all values are 1, as shown:

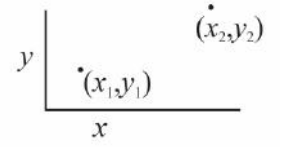
$$\begin{pmatrix}1\\1\\1\end{pmatrix}.\begin{pmatrix}1\\1\\1\end{pmatrix}=1^2+1^2+1^2=3$$

Graphs

conventions with regard to graphs and notations.



This is a basic Cartesian graph (x and y coordinate). The x and y notation are very standard but sometimes do not entirely explain the big picture. We sometimes refer to the x variable as being the independent variable and the y as the dependent variable. This is because when we write functions, we tend to speak about them as being y is a function of x, meaning that the value of y is dependent on the value of x. This is what a graph is trying to show.
Suppose we have two points on a graph, as shown:



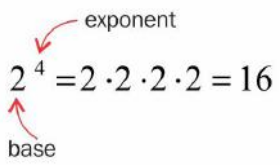
We refer to the points as (x_1, y_1) and (x_2, y_2)

The slope between these two points is defined as follows:

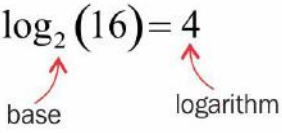
$$slope = m = \frac{y_2 - y_1}{x_2 - x_1}$$

Logarithms/exponents

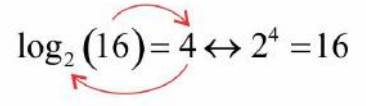
An exponent tells you how many times you have to multiply a number to itself, as illustrated:



A logarithm is the number that answers the question: "what exponent gets me from the base to this other number?" This can be denoted as follows:



If these two concepts seem similar, then you are correct! Exponents and logarithms are heavily related. In fact, the words exponent and logarithm actually mean the same thing! A logarithm is an exponent. The preceding two equations are actually two versions of the same thing. The basic idea is that 2 times 2 times 2 times 2 is 16



Consider the following examples:

- $\log_3 81 = 4$ because $3^4 = 81$
- $\log_5 125 = 3$ because $5^3 = 125$

Set theory

The set theory involves mathematical operations at a set level. It is sometimes thought of as a basic fundamental group of theorems that governs the rest of mathematics. For our purpose, we use the set theory in order to manipulate groups of elements.

A set is a collection of distinct objects.

That's it! A set can be thought of as a list in Python, but with no repeat objects. In fact, there even exists a set of objects in Python:

```
s = set()
s = set([1, 2, 2, 3, 2, 1, 2, 2, 3, 2])
# will remove duplicates from a list
s == {1, 2, 3}
dict = {"dog": "human's best friend", "cat": "destroyer of world"}
dict["dog"]# == "human's best friend"
len(dict["cat"]) # == 18
```

```
dict["dog"] = "Arf"
dict
{"dog": "Arf", "cat": "destroyer of world"}
```

The magnitude of a set is the number of elements in the set and is represented as follows:

$$|A| = magnitude\ of\ A$$

If we wish to denote that an element is within a set, we use the epsilon notation, as shown:

$$2 \in \{1,2,3\}$$

This means that the element 2 exists in the set of 1, 2, and 3. If one set is entirely inside another set, we say that it is a subset of its larger counterpart.

$$A = \{1,5,6\}, B = \{1,5,6,7,8\}$$

$$A \subseteq B$$

So, A is a subset of B and B is called the superset of A. If A is a subset of B but A does not equal B (meaning that there is at least one element in B that is not in A), then A is called a proper subset of B.

Consider the following examples:

- A set of even numbers is a subset of all integers
- Every set is a subset, but not a proper subset, of itself
- A set of all tweets is a superset of English tweets

Consider that we are a marketing firm trying to predict where a person wants to shop for clothes. We are given a set of clothing brands the user has previously visited, and our goal is to predict a new store that they would also enjoy. Suppose a specific user has previously shopped at the following stores:

```
user1 = {"Target","Banana Republic","Old Navy"}
# note that we use {} notation to create a set
# compare that to using [] to make a list
```

So, user1 has previously shopped at Target, Banana Republic, and Old Navy. Let's also look at a different user, called user2, as shown:

```
user2 = {"Banana Republic","Gap","Kohl's"}
```

Suppose we are wondering how similar these users are. With the limited information we have, one way to define similarity is to see how many stores there are that they both shop at. This is called an intersection.

The intersection of two sets is a set whose elements appear in both the sets. It is denoted using the symbol , as shown:

$$user1 \cap user2 = \{Banana\ Republic\}$$

$$|user1 \cap user2| = 1$$

The intersection of the two users is just one store. So, right away that doesn't seem great. However, each user only has three elements in their set, so having 1/3 does

not seem as bad. Suppose we are curious about how many stores are represented between the two of them; this is called a union.

The union of two sets is a set whose elements appear in either set. It is denoted using the symbol U, as shown:

$$user1 \cup user2 = \{Banana\ Republic, Target, Old\ Navy, Gap, Kohl's\}$$

$$|user1 \cup user2| = 5$$

Linear algebra

Remember the movie recommendation engine we looked at earlier? What if we had 10,000 movies to recommend and we had to choose only 10 to give to the user? We'd have to take a dot product between the user profile and each of the 10,000 movies. Linear algebra provides the tools to make these calculations much more efficient. It is an area of mathematics that deals with the math of matrices and vectors. It has the aim of breaking down these objects and reconstructing them in order to provide practical applications. Let's look at a few linear algebra rules before proceeding.

Matrix multiplication

Like numbers, we can multiple matrices together. Multiplying matrices is, in essence, a mass produced way of taking several dot products at once. Let's, for example, try to multiple the following matrices:

$$\begin{pmatrix} 1 & 5 \\ 5 & 8 \\ 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 3 & 4 \\ 2 & 5 \end{pmatrix}$$

A couple of things:

• Unlike numbers, multiplication is not commutative, meaning that the order in which you multiply matrices matters a great deal.

• In order to multiply matrices, their dimensions must match up. This means that the first matrix must have the same number of columns as the second matrix has rows.

To remember this, write out the dimensions of the matrices. In this case, we have a 3x2 times a 2 x 2 matrix. You can multiple matrices together if the second number in the first dimension pair is the same as the first number in the second dimension p

$$3 \times \boxed{2 \cdot 2} \times 2$$

The resulting matrix will always have dimensions equal to the outer numbers in the dimension pairs (the ones you did not circle in the second point). In this case, the resulting matrix will have a dimension of 3 x 2.

How to multiply matrices

To multiply matrices, there is actually a quite simple procedure. Essentially, we are performing a bunch of dot products.

Recall our earlier sample problem, which was as follows:

$$\begin{pmatrix} 1 & 5 \\ 5 & 8 \\ 7 & 8 \end{pmatrix} \cdot \begin{pmatrix} 3 & 4 \\ 2 & 5 \end{pmatrix}$$

We know that our resulting matrix will have a dimension of 3 x 2. So, we know it will look something like the following:

$$\begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \\ m_{31} & m_{32} \end{pmatrix}$$

The m_{xy} element is the result of the dot product of the x^{th} row of the first matrix and the y^{th} column of the second matrix. Let's solve a few:

$$m_{11} = \begin{pmatrix} 1 \\ 5 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 2 \end{pmatrix} = 13$$

$$m_{12} = \begin{pmatrix} 1 \\ 5 \end{pmatrix} \cdot \begin{pmatrix} 4 \\ 5 \end{pmatrix} = 29$$

Moving on, we will eventually get a resulting matrix as follows:

$$\begin{pmatrix} 13 & 29 \\ 31 & 60 \\ 37 & 68 \end{pmatrix}$$

Way to go! Let's come back to the movie recommendation example. Recall the user's movie genre preferences of comedy, romance, and action, which are illustrated as follows:

$$U = user\ prefs = \begin{pmatrix} 5 \\ 1 \\ 3 \end{pmatrix}$$

Probability

Basic definitions

One of the most basic concepts of probability is the concept of a procedure. A procedure is an act that leads to a result. For example, throwing a dice or visiting a website.

An event is a collection of the outcomes of a procedure, such as getting a heads on a coin flip or leaving a website after only 4 seconds. A simple event is an outcome/ event of a

procedure that cannot be broken down further. For example, rolling two dice can be broken down into two simple events: rolling die 1 and rolling die 2.

The sample space of a procedure is the set of all possible simple events. For example, an experiment is performed, in which a coin is flipped three times in succession. What is the size of the sample space for this experiment?

The answer is eight, because the results could be any one of the possibilities in the following sample space—{HHH, HHT, HTT, HTH, TTT, TTH, THH, or THT}.

Probability

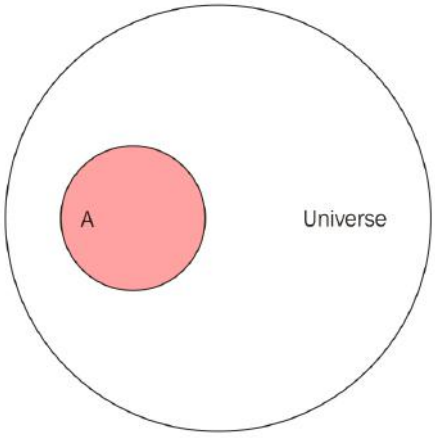
The probability of an event represents the frequency, or chance, that the event will happen.

For notation, if *A* is an event, P(*A*) is the probability of the occurrence of the event.

We can define the actual probability of an event, *A*, as follows:

$$P(A) = \frac{\text{number of ways A occur}}{\text{size of sample space}}$$

Here, *A* is the event in question. Think of an entire universe of events where anything is possible, and let's represent it as a circle. We can think of a single event, *A*, as being a smaller circle within that larger universe, as shown in the following diagram:



Let's now pretend that our universe involves a research study on humans, and the ***A*** event is people in that study who have cancer.

If our study has 100 people and *A* has 25 people, the probability of *A* or *P(A)* is 25/100.

The maximum probability of any event is 1. This can be understood as the red circle grows so large that it is the size of the universe (the larger circle).

The most basic examples (I promise they will get more interesting) are coin flips. Let's say we have two coins and we want the probability that we will roll two heads. We can very easily count the number of ways two coins could end up being two heads. There's only one! Both coins have to be heads. But how many options are there? It could either be two heads, two tails, or a heads/tails combination.

First, let's define *A*. It is the event in which two heads occur. The number of ways that *A* can occur is 1.

The sample space of the experiment is {HH, HT, TH, TT}, where each two letter *word* indicates the outcome of the first and second coin simultaneously. The size of the sample space is four. So, *P(getting two heads) = 1/4*.

Let's refer to a quick visual table to prove it. The following table denotes the options for coin 1 as the columns and the options for coin 2 as the rows. In each cell, there is either a True or a False. A True value indicates that it satisfies the condition (both heads) and False indicates otherwise.

	Coin 1 is Heads	**Coin 1 is Tails**
Coin 2 is Heads	True	False
Coin 2 is Tails	False	False

So, we have one out of a total of four possible outcomes.

Bayesian versus Frequentist

The preceding example was almost too easy. In practice, we can hardly ever truly count the number of ways something can happen. For example, let's say that we want to know the probability of a random person smoking cigarettes at least once a day. If we wanted to approach this problem using the classical way (the previous formula), we would need to figure out how many different ways a person is a smoker—someone who smokes at least once a day—which is not possible!

When faced with such a problem, two main schools of thought are considered when it comes to calculating probabilities in practice: the Frequentist approach and the Bayesian approach. This chapter will focus heavily on the Frequentist approach while the subsequent chapter will dive into the Bayesian analysis.

Frequentist approach

In a Frequentist approach, the probability of an event is calculated through experimentation. It uses the past in order to predict the future chance of an event.

The basic formula is as follows:

$$P(A) = \frac{\text{number of times A occurred}}{\text{number of times the procedure was repeated}}$$

Basically, we observe several instances of the event and count the number of times *A* was satisfied. The division of these numbers is an approximation of the probability.

The Bayesian approach differs by dictating that probabilities must be discerned using theoretical means. Using the Bayes approach, we would have to think a bit more critically about events and why they occur. Neither methodology is 100% the correct answer all the time. Usually, it comes down to the problem and the difficulty of using either approach.

The crux of the Frequentist approach is the relative frequency.

The **relative frequency** of an event is how often an event occurs divided by the total number of observations.

Example – marketing stats

Let's say that you are interested in ascertaining how often a person who visits your website is likely to return on a later date. This is sometimes called the rate of repeat visitors. In the previous definition, we would define our *A* event as being a visitor coming back to the site. We would then have to calculate the number of ways a person can come back, which doesn't really make sense at all! In this case, many people would turn to a Bayesian approach; however, we can calculate what is known as relative frequency.

So, in this case, we can take the visitor logs and calculate the relative frequency of event *A* (repeat visitors). Let's say, of the *1,458* unique visitors in the past week, *452* were repeat visitors. We can calculate this as follows:

$$P(A)\ RF(A) = \frac{452}{1458} = .31$$

So, about 31% of your visitors are repeat visitors.

The law of large numbers

The reason that even the Frequentist approach can do this is because of the law of large numbers, which states that if we repeat a procedure over and over, the relative frequency probability will approach the actual probability. Let's try to demonstrate this using Python.

If I were to ask you the average of the numbers 1 and 10, you would very quickly answer around 5. This question is identical to asking you to pick the *average* number between 1 and 10. Let's design the experiment to be as follows:

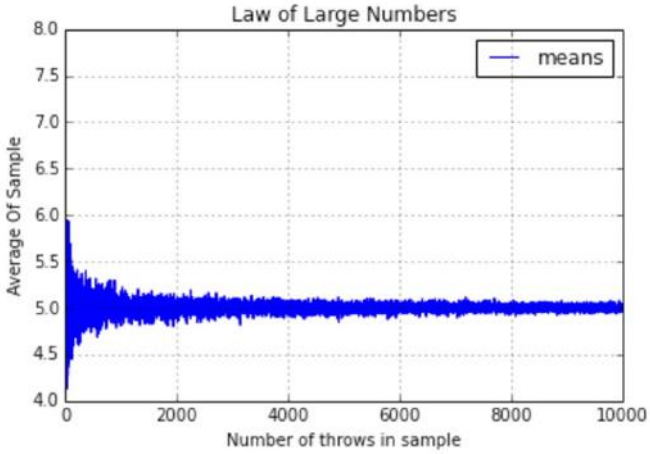
Python will choose *n* random numbers between 1 and 10 and find their average.

We will repeat this experiment several times using a larger *n* each time, and then we will graph the outcome. The steps are as follows:

1. Pick a random number between 1 and 10 and find the average.
2. Pick two random numbers between 1 and 10 and find their average.
3. Pick three random numbers between 1 and 10 and find their average.
4. Pick 10,000 random numbers between 1 and 10 and find their average.
5. Graph the results.

Let's take a look at the code:

```
import numpy as np import pandas as pd
from matplotlib import pyplot as plt %matplotlib inline
results = []
for n in range(1,10000):
nums = np.random.randint(low=1,high=10, size=n) # choose n numbers between 1 and 10
mean = nums.mean() # find the average
of these numbers
results.append(mean) # add the average
to a running list
# POP QUIZ: How large is the list results?
len(results) # 9999
# This was tricky because I took the range from 1 to 10000 and usually we do from 0 to
10000
df = pd.DataFrame({ 'means' : results})
print df.head() # the averages in the beginning are all over the place!
# means # 9.0 # 5.0 # 6.0 # 4.5 # 4.0
print df.tail() # as n, our size of the sample size, increases, the averages get closer to 5!
# means # 4.998799 # 5.060924 # 4.990597 # 5.008802 # 4.979198
df.plot(title='Law of Large Numbers') plt.xlabel("Number of throws in sample")
plt.ylabel("Average Of Sample")
```



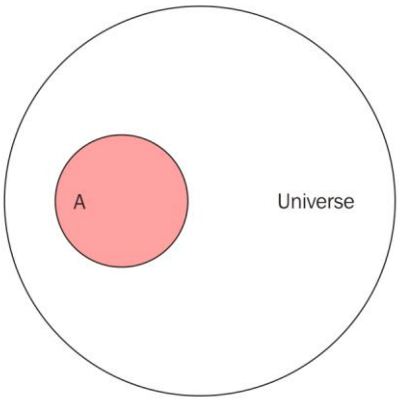
Compound events

Sometimes, we need to deal with two or more events. These are called **compound events**. A compound event is any event that combines two or more simple events. When this happens, we need some special notation.

Given events *A* and *B*:

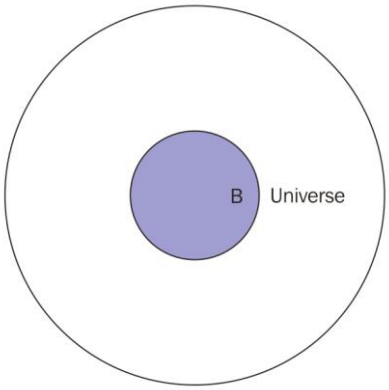
- The probability that *A* and *B* occur is $P(A \cap B) = P(A \text{ and } B)$
- The probability that either *A* or *B* occurs is $P(A \cup B) = P(A \text{ or } B)$

Understanding why we use set notation for these compound events is very important. Remember how we represented events in a universe using circles earlier? Let's say that our Universe is 100 people who showed up for an experiment, in which a new test for cancer is being developed:

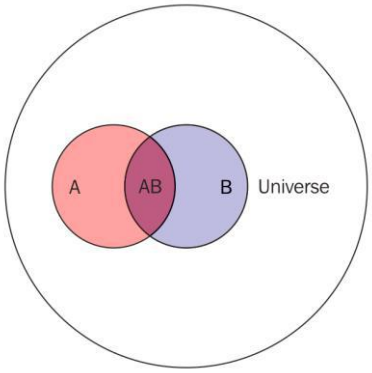


In the preceding diagram, the red circle, **A**, represents 25 people who actually have cancer. Using the relative frequency approach, we can say that *P(A) = number of people with cancer/number of people in study*, that is, *25/100 = $\frac{1}{4}$ = .25*. This means that there is a 25% chance that someone has cancer.

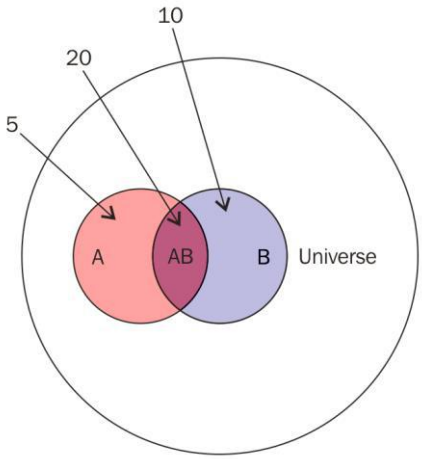
Let's introduce a second event, called **B**, as shown, which contains people for whom the test was positive (it claimed that they had cancer). Let's say that this is for 30 people. So, *P(B) = 30/100 = 3/10 = .3*. This means that there is a 30% chance that the test said positive for any given person:



These are two separate events, but they interact with each other. Namely, they might intersect or have people in common, as shown here:



Anyone in the space that both **A** and **B** occupy, otherwise known as *A intersect B* or *A ∩ B*, are people for whom the test claimed they were positive for cancer (**A**) and they actually do have cancer. Let's say that's 20 people. The test said positive for 20 people, that is, they have cancer, as shown here:



This means that *P(A and B) = 20/100 = 1/5 = .2 = 20%*.

If we want to say that someone has cancer *or* the test came back positive. This would be the total sum (or union) of the two events, namely, the sum of 5, 20, and 10, which is 35. So, 35/100 people either have cancer or had a positive test outcome. That means, *P(A or B) = 35/100 = .35 = 35%*.

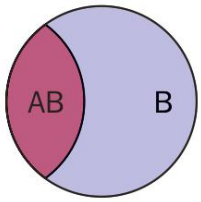
All in all, we have people in the following four different classes:

• **Pink**: This refers to the people who have cancer and had a negative test

outcome
• **Purple (A intersect B)**: These people have cancer and had a positive test outcome
• **Blue**: This refers to the people with no cancer and a positive test outcome
• **White**: This refers to the people with no cancer and a negative test outcome
So, effectively, the only times the test was *accurate* was in the white and purple regions. In the blue and pink regions, the test was incorrect.

Conditional probability

Let's pick an arbitrary person from this study of 100 people. Let's also assume that you are told that their test result was positive. What is the probability of them actually having cancer? So, we are told that event *B* has already taken place, and that their test came back positive. The question now is: what is the probability that they have cancer, that is *P(A)*? This is called a **conditional probability of A given B** or **P(A|B)**. Effectively, it is asking you to calculate the probability of an event given that another event has already happened.
You can think of conditional probability as changing the relevant universe. *P(A|B)* (called the probability of *A* given *B*) is a way of saying, given that my entire universe is now *B*, what is the probability of *A*? This is also known as transforming the sample space.



Zooming in on our previous diagram, our universe is now B, and we are concerned with AB (A and B) inside of B
The formula can be given as follows:
P(A|B) = P(A and B) / P(B) = (20/100) / (30/100) = 20/30 = .66 = 66%
There is a 66% chance that if a test result came back positive, that person had cancer. In reality, this is the main probability that the experimenters want. They want to know how good the test is at predicting cancer.

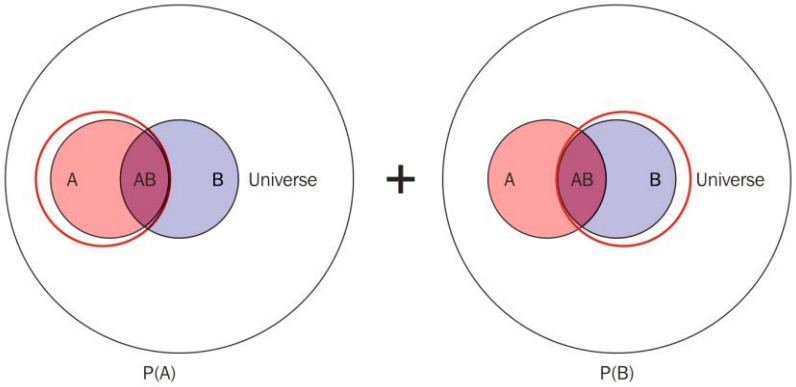
The rules of probability

In probability, we have some rules that become very useful when visualization gets too cumbersome. These rules help us calculate compound probabilities with ease.

The addition rule

The addition rule is used to calculate the probability of *either or* events. To calculate *P(A ∪ B) = P(A or B)*, we use the following formula:
P(A∪ B) = P(A) + P(B) - P(A ∩ B)

The first part of the formula *(P(A) + P(B))* makes complete sense. To get the union of the two events, we have to add together the area of the circles in the universe. But why the subtraction of *P(A and B)*? This is because when we add the two circles, we are adding the area of intersection twice, as shown in the following diagram:



See how both the red circles include the intersection of **A** and **B**? So, when we add them, we need to subtract just one of them to account for this, leaving us with our formula. Recall that we wanted the number of people who either had cancer or had a positive test result? If *A* is the event that someone has cancer, and *B* is that the test result was positive, we have:

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B) = .25 + .30 - .2 = .35$$

Mutual exclusivity

We say that two events are mutually exclusive if they cannot occur at the same time. This means that $A \cap B = \phi$ *or just that the intersection of the events is the empty set.*

When this happens, $P(A \cap B) = P(A \text{ and } B) = 0$.

If two events are mutually exclusive, then:

$$P(A \cup B) = P(A \text{ or } B) = P(A) + P(B) - P(A \cap B) = P(A) + P(B)$$

This makes the addition rule much easier. Some examples of mutually exclusive events include the following:

- A customer seeing your site for the first time on both Twitter and Facebook
- Today is Saturday and today is Wednesday
- I failed Econ 101 and I passed Econ 101

None of these events can occur simultaneously

The multiplication rule

The multiplication rule is used to calculate the probability of *and* events. To calculate $P(A \cap B) = P(A \text{ and } B)$, we use the following formula:

$$P(A \cap B) = P(A \text{ and } B) = P(A) \cdot P(B|A)$$

Why do we use $B|A$ instead of B? This is because it is possible that B depends on A. If this is the case, then just multiplying $P(A)$ and $P(B)$ does not give us the whole picture. In our cancer trial example, let's find $P(A \text{ and } B)$. To do this, let's redefine A to be the event that the trial is positive and B to be the person having cancer (because it doesn't matter what we call the events). The equation will be as follows:

$$P(A \cap B) = P(A \text{ and } B) = P(A) \cdot P(B|A) = .3 * .6666 = .2 = 20\%$$

This was calculated before visually.

It's difficult to see the true necessity of using the conditional probability, so, let's try another, more difficult problem.

For example, of a randomly selected set of 10 people, 6 have iPhones and 4 have Androids. What is the probability that if I randomly select two people, they both will have iPhones? This example can be retold using event spaces, as follows:
I have the following two events:
• *A*: This event shows the probability that I choose a person with an iPhone first
• *B*: This event shows the probability that I choose a person with an iPhone second
So, basically, I want the following:
• *P(A and B)*: *P(I choose a person with an iPhone and a person with an iPhone)* So, I can use my *P(A and B) = P(A) · P(B|A)* formula.
P(A) is simple, right? People with iPhones are 6 out of 10, so, I have a *6/10 = 3/5 = 0.6* chance of A. This means *P(A) = 0.6*.
So, if I have a 0.6 chance of choosing someone with an iPhone, the probability of choosing two should just be *0.6 * 0.6*, right?
But wait! We only have 9 people left to choose our second person from, because one was taken away. So in our new transformed sample space, we have 9 people in total, 5 with iPhones and 4 with droids, making *P(B) = 5/9 = .555*.
So, the probability of choosing two people with iPhones is *0.6 * 0.555 = 0.333 = 33%*.
I have a 1/3 chance of choosing two people with iPhones out of 10. The conditional probability is very important in the multiplication rule as it can drastically alter your answer.

Independence
Two events are independent if one event does not affect the outcome of the other, that is *P(B|A) = P(B)* and *P(A|B) = P(A)*.
If two events are independent, then:
P(A ∩ B) = P(A) · P(B|A) = P(A) · P(B)
Some examples of independent events are as follows:
• It was raining in San Francisco, and a puppy was born in India
• Flip a coin and get heads and flip another coin and get tails None of these pairs of events affect each other.

Complementary events

The complement of A is the opposite or negation of A. If A is an event, $\bar{A}$ represents the complement of A. For example, if A is the event where someone has cancer, $\bar{A}$ is the event where someone is cancer free.

To calculate the probability of $\bar{A}$, use the following formula:

$P(\bar{A}) = 1 - P(A)$

For example, when you throw two dice, what is the probability that you rolled higher than a 3?

Let A represent rolling higher than a 3.

$\bar{A}$ represents rolling a 3 or less.

$\bar{A}$

P(A) = 1 – P( ) P(A) =

l – (P(2)+P(3))

= 1 – (2/36 + 2/36)

= 1 – (4/36)

= 32/36 = 8 / 9

= .89

For example, a start-up team has three investor meetings coming up. We will have the following probabilities:

• 60% chance of getting money from the first meeting
• 15% chance of getting money from the second
• 45% chance of getting money from the third

What is the probability of them getting money from at least one meeting?

Let A be the team getting money from at least one investor $\bar{A}$, *and be the team not getting any money.* P(A) *can be calculated as follows:*

P(A) = 1 – P($\bar{A}$)

To calculate P($\bar{A}$)*, we need to calculate the following:*

P($\bar{A}$) = P(no money from investor 1 AND no money from investor 2 AND no money from investor 3)

If we assume that these events are independent (they don't talk to each other), then:

P($\bar{A}$) = P(no money from investor 1) * P(no money from investor 2) * P(no money from investor 3) =

0.4 * 0.85 * 0.55 = 0.187

P(A) = 1 - 0.187 = 0.813 = 81%

So, the startup has an 81% chance of getting money from at least one meeting!

Collectively exhaustive events

When given a set of two or more events, if at least one of the events must occur, then such a set of events is said to be collectively exhaustive.

Consider the following examples:

• Given a set of events {temperature < 60, temperature > 90}, these events are not collectively exhaustive because there is a third option that is not given in this set of events: The temperature could be between 60 and 90.

However, they are mutually exhaustive because both cannot happen at the same time.

• In a dice roll, the set of events of rolling a {1, 2, 3, 4, 5, or 6} are collectively exhaustive because these are the only possible events, and at least one of them must happen.

Bayesian ideas revisited

In the last chapter, we talked, very briefly, about Bayesian ways of thinking. In short, when speaking about Bayes, you are speaking about the following three things and how they all interact with each other:

• A prior distribution
• A posterior distribution
• A likelihood

Basically, we are concerned with finding the posterior. That's the thing we want to know.

Another way to phrase the Bayesian way of thinking is that data shapes and updates our belief. We have a prior probability, or what we naively think about a hypothesis, and then we have a posterior probability, which is what we think about a hypothesis, given some data.

Bayes theorem

Bayes theorem is the big result of Bayesian inference. Let's see how it even comes about. Recall that we previously defined the following:

• *P(A) = The probability that event A occurs*
• *P(A|B) = The probability that A occurs, given that B occurred*
• *P(A, B) = The probability that A and B occurs*
• *P(A, B) = P(A) * P(B|A)*

That last bullet can be read as *the probability that A and B occur is the probability that A occurs times the probability that B occurred, given that A already occurred*.

It's from that last bullet point that Bayes theorem takes its shape.

We know that:

P(A, B) = P(A) * P(B|A)

P(B, A) = P(B) * P(A|B)

P(A, B) = P(B, A)

So:

P(B) * P(A|B) = P(A) * P(B|A)

Dividing both sides by P(B) gives us Bayes theorem, as shown:

$$P(A|B) = \frac{P(A) * P(B|A)}{P(B)}$$

You can think of Bayes theorem as follows:

• It is a way to get from P(A|B) to P(B|A) (if you only have one)

• It is a way to get P(A|B) if you already know P(A) (without knowing B)

Let's try thinking about Bayes using the terms hypothesis and data. Suppose H = your

hypothesis about the given data and D = the data that you are given.

Bayes can be interpreted as trying to figure out P(H|D) (the probability that our hypothesis is correct, given the data at hand).

To use our terminology from before:

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)}$$

• P(H) is the probability of the hypothesis before we observe the data, called the prior probability or just prior

• P(H|D) is what we want to compute, the probability of the hypothesis after we observe the data, called the posterior

• P(D|H) is the probability of the data under the given hypothesis, called the likelihood

• P(D) is the probability of the data under any hypothesis, called the normalizing constant.

This concept is not far off from the idea of machine learning and predictive analytics. In many cases, when considering predictive analytics, we use the given data to predict an outcome. Using the current terminology, H (our hypothesis) can be considered our outcome and P(H|D) (the probability that our hypothesis is true, given our data) is another way of saying: what is the chance that my hypothesis is correct,

given the data in front of me?.

Let's take a look at an example of how we can use Bayes formula at the workplace.

Consider that you have two people in charge of writing blog posts for your company—Lucy and Avinash. From past performances, you have liked 80% of Lucy's work and only 50% of Avinash's work. A new blog post comes to your desk in the morning, but the author isn't mentioned. You love the article. A+. What is the probability that it came from Avinash? Each blogger blogs at a very similar rate.

Before we freak out, let's do what any experienced mathematician (and now you) would do. Let's write out all of our information, as shown:

• H = hypothesis = the blog came from Avinash

• D = data = you loved the blog post

P(H|D) = the chance that it came from Avinash, given that you loved it

P(D|H) = the chance that you loved it, given that it came from Avinash

P(H) = the chance that an article came from Avinash

P(D) = the chance that you love an article

Note that some of these variables make almost no sense without context. P(D), the probability that you would love any given article put on your desk is a weird concept, but trust me, in the context of Bayes formula, it will be relevant very soon. Also, note that in the last two items, they assume nothing else. P(D) does not assume the origin of the blog post; think of P(D) as if an article was plopped on your desk from
some unknown source, what is the chance that you'd like it? (again, I know it sounds weird out of context).

So, we want to know P(H|D). Let's try to use Bayes theorem, as shown, here:

$$P(H \mid D) = \frac{P(D \mid H)P(H)}{P(D)}$$

But do we know the numbers on the right-hand side of this equation? I claim we do! Let's see here:

- P(H) is the probability that any given blog post comes from Avinash. As bloggers write at a very similar rate, we can assume this is .5 because we have a 50/50 chance that it came from either blogger (note how I did not assume D, the data, for this).
- P(D|H) is the probability that you love a post from Avinash, which we previously said was 50%, so, .5.
- P(D) is interesting. This is the chance that you love an article in general. It means that we must take into account the scenario if the post came from Lucy or Avinash. Now, if the hypothesis forms a suite, then we can use our laws of probability, as mentioned in the previous chapter. A suite is formed when a set of hypotheses is both collectively exhaustive and mutually exclusive. In laymen's terms, in a suite of events, exactly one and only one hypothesis can occur. In our case, the two hypotheses are that the article came from Lucy, or that the article came from Avinash. This is definitely a suite because of the following reasons:

° At least one of them wrote it
° At most one of them wrote it
° Therefore, exactly one of them wrote it

When we have a suite, we can use our multiplication and addition rules, as follows:

$$D = (From\ Avinash\ AND\ loved\ it)\quad OR\quad (From\ Lucy\ AND\ loved\ it)$$

$$P(D) = P(Loved\ AND\ from\ Avinash)\quad OR\quad P(Loved\ AND\ from\ Lucy)$$

$$P(D) = P(From\ Avinash)\,P(Loved \mid form\ Avinash) + P(from\ Lucy)\,P(Loved \mid from\ Lucy)$$

$$P(D) = .5(.5) + .5(.8) = \mathbf{.65}$$

Now we can finish our equation, as shown:

$$P(H \mid D) = \frac{P(D \mid H)P(H)}{P(D)}$$

$$P(H \mid D) = \frac{.5 * .5}{.65} = .38$$

This means that there is a 38% chance that this article comes from Avinash. What is interesting is that P(H) = .5 and P(H|D) = .38. It means that without any data, the chance that a blog post came from Avinash was a coin flip, or 50/50. Given some data (your thoughts on the article), we updated our beliefs about the hypothesis and it actually lowered the chance.

Random variables

A **random variable** uses real numerical values to describe a probabilistic event. In our previous work with variables (both in math and programming), we were used to the fact that a variable takes on a certain value. For example, we might have a triangle in which we are given a variable *h* for the hypotenuse, and we must figure out the length of the hypotenuse. We also might have, in Python:

x = 5

Both of these variables are equal to one value at a time. In a random variable, we are subject to randomness, which means that our variables' values are, well just that, variable! They might take on multiple values depending on the environment.

The main distinction

between variables as we have seen them and a random variable is the fact that a random variable's value may change depending on the situation.

However, if a random variable can have many values, how do we keep track of them all? Each value that a random variable might take on is associated with a percentage. For every value that a random variable might take on, there is a single probability that the variable will be this value.

With a random variable, we can also obtain our probability distribution of a random variable, which gives the variable's possible values and their probabilities.

Written out, we generally use single capital letters (mostly the specific letter X) to denote random variables. For example, we might have:

- X = the outcome of a dice roll
- Y = the revenue earned by a company this year
- Z = the score of an applicant on an interview coding quiz (0-100%)

Effectively, a random variable is a function that maps values from the sample space of an event (the set of all possible outcomes) to a probability value (between 0 and 1). Think about the event as being expressed as the following:

$$f(event) = probability$$

It will assign a probability to each individual option. There are two main types of random variables: discrete and continuous.

Discrete random variables

A discrete random variable only takes on a countable number of possible values. For example, the outcome of a dice roll, as shown here:

X = the outcome of a single dice roll

Value	$X=1$	$X=2$	$X=3$	$X=4$	$X=5$	$X=6$
Probability	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

Random variables have many properties, two of which are their *expected value* and the *variance*.

We will use a **probability mass function** (**PMF**) to describe a discrete random variable. They take on the appearance of the following: P(X = x) = PMF

So, for a dice roll, *P(X = 1) = 1/6* and *P(X = 5) = 1/6*. Consider the following examples of discrete variables:

- The likely result of a survey question (for example, on a scale of 1-10)
- Whether the CEO will resign within the year (either true or false)

The expected value of a random variable defines the mean value of a long run of repeated samples of the random variable. This is sometimes called the mean of the variable.

For example, refer to the following Python code that defines the random variable of a dice roll:

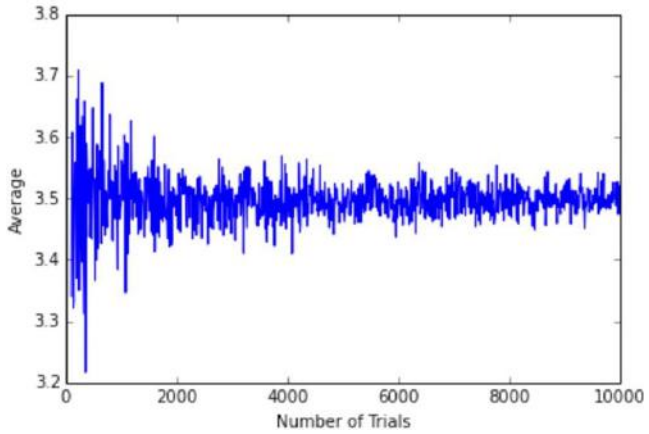
```
import random
def random_variable_of_dice_roll():
return random.randint(1, 7)
# a range of (1,7)
 # includes 1, 2, 3,4, 5, 6, but NOT 7
```

This function will invoke a random variable and come out with a response. Let's roll 100 dice and average the result, as follows:

```
trials = []
num_trials = 100
for trial in range(num_trials):
trials.append( random_variable_of_dice_roll() )
print sum(trials)/float(num_trials) # == 3.77
```

So, taking 100 dice rolls and averaging them gives us a value of 3.77! Let's try this with a wide variety of trial numbers, as illustrated here:

```
num_trials = range(100,10000, 10)
avgs = []
for num_trial in num_trials:
trials = []
for trial in range(1,num_trial):
trials.append( random_variable_of_dice_roll() )
avgs.append(sum(trials)/float(num_trial))
plt.plot(num_trials, avgs)
plt.xlabel('Number of Trials')
plt.ylabel("Average")
```



The preceding graph represents the average dice roll as we look at more and more dice rolls. We can see that the average dice roll is rapidly approaching 3.5. If we look towards the left of the graph, we see that if we only roll a die about 100 times, then we are not guaranteed to get an average dice roll of 3.5. However, if we roll 10,000 dice one after another, we see that we would very likely expect the average dice roll to be about 3.5.

For a discrete random variable, we can also use a simple formula, shown as follows, to calculate the exp

$$\text{Expected value} = E[X] = \mu_X = \sum x_i p_i$$

Where x_i *is the* i^{th} *outcome and* p_i *is the* i^{th} *probability.*

So, for our dice roll, we can find the exact expected value as being as follows:

The preceding result shows us that for any given dice roll, we can "expect" a dice roll of 3.5. Now, obviously, that doesn't make sense because we can't get a 3.5 on a dice roll, but it does make sense when put in the context of many dice rolls. If you roll 10,000 dice, your average dice roll should approach 3.5, as shown in the graph and

code previously.

The average of the expected value of a random variable is generally not enough to grasp the full idea behind the variable. For this reason, we introduce a new concept, called variance.

The variance of a random variable represents the spread of the variable. It quantifies the variability of the expected value.

$$\frac{1}{6}(1)+\frac{1}{6}(2)+\frac{1}{6}(3)+\frac{1}{6}(4)+\frac{1}{6}(5)+\frac{1}{6}(6)=3.5$$

The formula for the variance of a discrete random variable is expressed as follows:

$$\text{Variance} = V[X] = \sigma_X^2 = \sum (x_i - \mu_X)^2 p_i$$

Where x_i *and* p_i *represent the same values as before and* μ_x *represents the expected value of the variable. In this formula, I also mentioned sigma of X. Sigma,*

in this case, is the standard deviation, which is defined simply as the square root of the variance. Let's look at a more complicated example of a discrete random variable.

UNIT-3
STATISTICS

BASICS

What are statistics?

This might seem like an odd question to ask, but I am frequently surprised by the number of people who cannot answer this simple and yet powerful question: what are statistics? Statistics are the numbers you always see on the news and in the paper. Statistics are useful when trying to prove a point or trying to scare you, but what
are they?

Now, consider that you want to ask a question about your population, for example, if your population is all of your employees (assume that you have over 1,000 employees), perhaps you want to know what percentage of them use illicit drugs. The question is called a **parameter**.

We can define a parameter as a numerical measurement describing a characteristic of a population.

For example, if you ask all 1,000 employees and 100 of them are using drugs, the rate of drug use is 10%. The parameter here is 10%.

First, we will take a sample of the population.

We can define a sample of a population as a subset (random not required) of the population.

So, we perhaps ask 200 of the 1,000 employees you have. Of these 200, suppose 26 use drugs, making the drug use rate 13%. Here, 13% is not a parameter because we didn't get a chance to ask everyone. This 13% is an estimate of a parameter. Do you know what that's called?

That's right, a statistic!

OBTAINING DATA

There are two main ways of collecting data for our analysis: **observational** and **experimentation**. Both these ways have their pros and cons, of course. They each produce different types of behavior and, therefore, warrant different types of analysis.

Observational

We might obtain data through observational means, which consists of measuring specific characteristics but not attempting to modify the subjects being studied. For example, you have a tracking software on your website that observes users' behavior on the website, such as length of time spent on certain pages and the rate of clicking on ads, all the while not affecting the user's experience, then that would be an observational study.

This is one of the most common ways to get data because it's just plain easy. All you have to do is observe and collect data. Observational studies are also limited in the types of data you may collect. This is because the observer (you) is not in control of the environment. You may only watch and collect natural behavior.

Experimental

An experiment consists of a treatment and the observation of its effect on the subjects. Subjects in an experiment are called experimental units. This is usually how most scientific labs collect data. They will put people into two or more groups (usually just two) and call them the control and the experimental group.

The control group is exposed to a certain environment and then observed. The experimental group is then exposed to a different environment and then observed. The experimenter then aggregates data from both the groups and makes a decision about which environment was more favorable (favorable is a quality that the experimenter gets to decide)

In a marketing example, consider that we expose half of our users to a certain landing page with certain images and a certain style (website A), and we measure whether or not they sign up for the service. Then, we expose the other half to a different landing page, different images, and different styles (website B) and again measure whether or not they sign up. We can then decide which of the two sites performed better and should be used going further. This, specifically, is called an A/B test. Let's see an example in Python! Let's suppose we run the preceding test and obtain the following results as a list of lists:

results = [['A', 1], ['B', 1], ['A', 0], ['A', 0] …]

Here, each object in the list result represents a subject (person). Each person then has the following two attributes:

- Which website they were exposed to, represented by a single character
- Whether or not they converted (0 for no and 1 for yes)

We can then aggregate and come up with the following results table:

```
users_exposed_to_A = []
users_exposed_to_B = []
# create two lists to hold the results of each individual website
```

Once we create these two lists that will eventually hold each individual conversion Boolean (0 or 1), we will iterate all of our results of the test and add them to the appropriate list, as shown:

```
for website, converted in results: # iterate through the results
# will look something like website == 'A' and converted == 0
if website == 'A':
users_exposed_to_A.append(converted)
elif website == 'B':
users_exposed_to_B.append(converted)
```

Now, each list contains a series of 1s and 0s.

To get the total number of people exposed to website A, we can use the len() feature in Python, as illustrated:

```
len(users_exposed_to_A) == 188 #number of people exposed to website A
len(users_exposed_to_B) == 158 #number of people exposed to website B
```

To count the number of people who converted, we can use the sum() of the list, as shown:

```
sum(users_exposed_to_A) == 54 # people converted from website A
sum(users_exposed_to_B) == 48 # people converted from website B
```

If we subtract the length of the lists and the sum of the list, we are left with the number of people who did not convert for each site, as illustrated:

```
len(users_exposed_to_A) - sum(users_exposed_to_A) == 134
# did not convert from website A
len(users_exposed_to_B) - sum(users_exposed_to_B) == 110
# did not convert from website B
```

We can aggregate and summarize our results in the following table that represents our experiment of website conversion testing:

	Did not sign up	Signed up
Website A	134	54
Website B	110	48

We can quickly drum up some descriptive statistics. We can say that the website conversion rates for the two websites are as follows:

- Conversion for website A: $\frac{54}{134+54} = .288$
- Conversion for website B: $\frac{48}{110+48} = .3$

Not much difference, but different nonetheless. Even though B has the higher conversion rate, can we really say that the version B significantly converts better?

Not yet. To test the statistical significance of such a result, a hypothesis test should be used. These tests will be covered in depth in the next chapter, where we will revisit this exact same example and finish it using the proper statistical test.

Sampling data

Remember how statistics are the result of measuring a sample of a population. Well, we should talk about two very common ways to decide who gets the honor of being in the sample that we measure. We will discuss the main type of sampling, called random sampling, which is the most common way to decide our sample sizes and our sample members.

Probability sampling

Probability sampling is a way of sampling from a population, in which every person has a known probability of being chosen but that number might be a different probability than another user. The simplest (and probably the most common) probability sampling method is random sampling

Random sampling

Suppose that we are running an A/B test and we need to figure out who will be in group A and who will be in group B. There are the following three suggestions from your data team:

- **Separate users based on location:** Users on the west coast are placed in group A, while users on the east coast are placed in group B
- **Separate users based on the time of day they visit the site:** Users who visit between 7 p.m. and 4 a.m. get site A, while the rest are placed in group B
- **Make it completely random:** Every new user has a 50/50 chance of being placed in either group

The first two are valid options for choosing samples and are fairly simple to implement, but they both have one fundamental flaw: they are both at risk of introducing a sampling bias.

A sampling bias occurs when the way the sample is obtained systemically favors some outcome over the target outcome.

It is not difficult to see why choosing option 1 or option 2 might introduce bias. If we chose our groups based on where they live or what time they log in, we are priming our experiment incorrectly and, now, we have much less control over the results. Specifically, we are at risk of introducing a confounding factor into our analysis, which is bad news.

A confounding factor is a variable that we are not directly measuring but connects the variables that are being measured.

Unequal probability sampling

Recall that I previously said that a probability sampling might have different probabilities for different potential sample members. But what if this actually

introduced problems? Suppose we are interested in measuring the happiness level of our employees. We already know that we can't ask every single person on the staff because that would be silly and exhausting. So, we need to take a sample. Our data team suggests random sampling and at first everyone high fives because they feel very smart and statistical.

How do we measure statistics?

Measures of center

Measures of center are how we define the middle, or center, of a dataset.

The arithmetic mean of a dataset is found by adding up all of the values and then dividing it by the number of data values

Suppose we wish to find the mean of the following numbers:

import numpy as np

np.mean([11, 15, 17, 14]) == 14.25

Simple enough, our average is 14.25 and all of our values are fairly close to it.

But what if we introduce a new value: 31?

np.mean([11, 15, 17, 14, 31]) == 17.6

The median is the number found in the middle of the dataset when it is sorted in order, as shown:

np.median([11, 15, 17, 14]) == 14.5

np.median([11, 15, 17, 14, 31]) == 15

Measures of variation

Consider that we take a random sample of 24 of our friends on Facebook and wrote down how many friends that they had on Facebook. Here's the list:

friends = [109, 1017, 1127, 418, 625, 957, 89, 950, 946, 797, 981, 125, 455, 731, 1640, 485, 1309, 472, 1132, 1773, 906, 531, 742, 621]

np.mean(friends) == 789.1

It's basically a way to see how spread out the data is. There is a general formula to calculate the standard deviation, which is as follows:

$$s = \sqrt{\frac{\sum (x - \bar{x})^2}{n}}$$

Here:

- s is our sample standard deviation
- is each individual data point.
- is the mean of the data
- is the number of data points

We start by taking the difference between each data value and the mean, squaring it, adding them all up, dividing it by one less than the number of values, and then taking its square root. This would look as follows:

$$s = \sqrt{\frac{(109-789)^2 + (1017-789)^2 + \cdots + (621-789)^2}{24}}$$

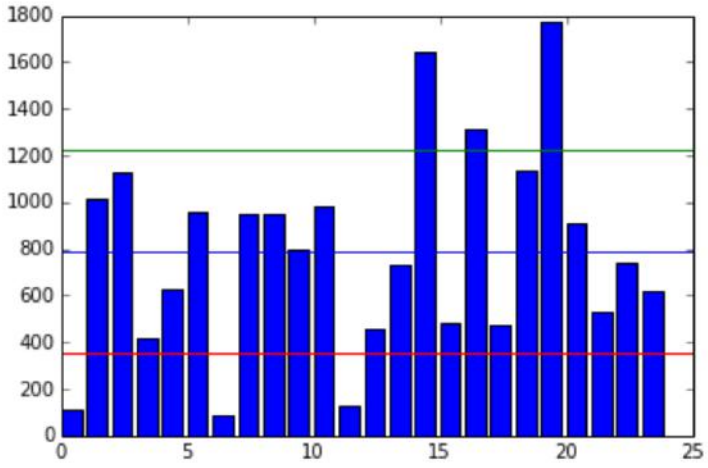
On the other hand, we can take the Python approach and do all this programmatically (which is usually preferred).

np.std(friends) # == 425.2

In the following plot, every

person will be represented by a single bar in the bar chart, and the height of the bars represent the number of friends that the individuals have:

```
import matplotlib.pyplot as plt
%matplotlib inline
y_pos = range(len(friends))
plt.bar(y_pos, friends)
plt.plot((0, 25), (789, 789), 'b-')
plt.plot((0, 25), (789+425, 789+425), 'g-')
plt.plot((0, 25), (789-425, 789-425), 'r-')
```



The blue line in the center is drawn at the mean (789), the red line on the bottom is drawn at the mean minus the standard deviation (789-425 = 364), and, finally, the green line towards the top is drawn at the mean plus the standard deviation

(789+425 = 1,214).

Measures of relative standing

We can combine both the measures of centers and variations to create measures of relative standings. Measures of variation measure where particular data values are positioned, relative to the entire dataset.

Let's begin by learning a very important value in statistics, the z-score.

The z-score is a way of telling us how far away a single data value is from the mean.

The z-score of a x data value is as follows:

$$z = \frac{x - \overline{x}}{s}$$

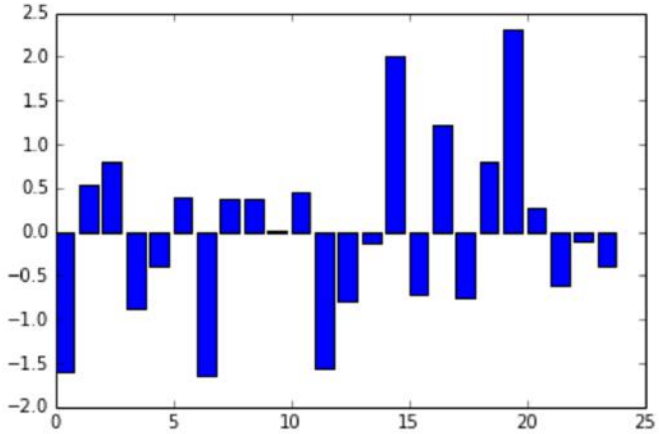
Where:

- x is the data point
- $\overline{x}$ is the mean
- s is the standard deviation.

Remember that the standard deviation was (sort of) an average distance that the data is from the mean, and, now, the z-score is an individualized value for each particular data point. We can find the z-score of a data value by subtracting it from the mean and dividing it by the standard deviation. The output will be the standardized distance a value is from a mean. We use the z-score all over statistics. It is a very effective way of normalizing data that exists on very different scales, and also to put data in context of their mean.

```
z_scores = []
m = np.mean(friends)    # average friends on Facebook
s = np.std(friends)     # standard deviation friends on Facebook
for friend in friends:
z = (friend - m)/s      # z-score
z_scores.append(z)   # make a list of the scores for plotting
```

```
plt.bar(y_pos, z_scores)
```

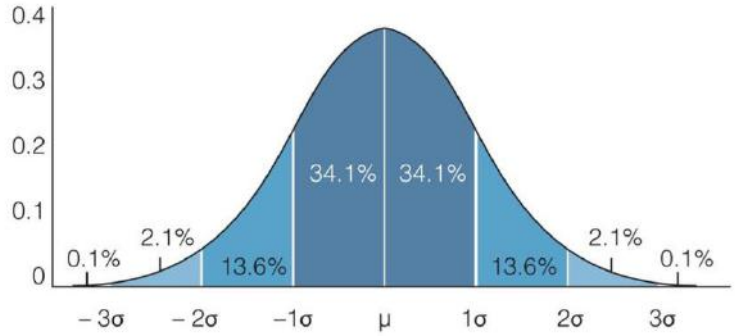


• We have negative values (meaning that the data point is below the mean)

• The bars' lengths no longer represent the raw number of friends, but the degree to which that friend count differs from the mean.

The Empirical rule

Recall that a normal distribution is defined as having a specific probability distribution that resembles a bell curve. In statistics, we love it when our data behaves normally. For example, if we have data that resembles a normal distribution, like so:



The Empirical rule states that we can expect a certain amount of data to live between sets of standard deviations. Specifically, the Empirical rule states for data that is distributed normally:

- about 68% of the data fall within 1 standard deviation
- about 95% of the data fall within 2 standard deviations
- about 99.7% of the data fall within 3 standard deviations

For example, let's see if our Facebook friends' data holds up to this. Let's use our Dataframe to find the percentage of people that fall within 1, 2, and 3 standard deviations of the mean, as shown:

```
# finding the percentage of people within one standard deviation of the mean
within_1_std = df_scaled[(df_scaled['friends_scaled'] <= 1) & (df_
scaled['friends_scaled'] >= -1)].shape[0]
within_1_std / float(df_scaled.shape[0])
# 0.75
# finding the percentage of people within two standard deviations of the mean
within_2_std = df_scaled[(df_scaled['friends_scaled'] <= 2) & (df_
scaled['friends_scaled'] >= -2)].shape[0]
within_2_std / float(df_scaled.shape[0])
# 0.916
# finding the percentage of people within three standard deviations of the mean
within_3_std = df_scaled[(df_scaled['friends_scaled'] <= 3) & (df_
scaled['friends_scaled'] >= -3)].shape[0]
within_3_std / float(df_scaled.shape[0])
# 1.0
```

We can see that our data does seem to follow the Empirical rule. About 75% of the people are within a single standard deviation of the mean. About 92% of the people are within two standard deviations, and all of them are within three standard deviations.

Example – exam scores

Let's say that we're measuring the scores of an exam and the scores generally have a bell-shaped normal distribution. The average of the exam was 84% and the standard deviation was 6%. We can say, with approximate certainty, that:

• About 68% of the class scored between 78% and 90% because 78 is 6 units below 84, and 90 is 6 units above 84

• If we were asked what percentage of the class scored between 72 and 96%, we would notice that 72 is 2 standard deviations below the mean, and 96 is 2 standard deviations above the mean, so, the Empirical rule tells us that about 95% of the class scored in that range.

However, not all data is normally distributed, so, we can't always use the Empirical rule. We have another theorem that helps us analyze any kind of distribution.

Point estimates

Recall that, in the previous chapter, we mentioned how difficult it was to obtain a population parameter; so, we had to use sample data to calculate a statistic that was an estimate of a parameter. When we make these estimates, we call them point estimates.

A point estimate is an estimate of a population parameter based on sample data. We use point estimates to estimate population means, variances, and other statistics. To obtain these estimates, we simply apply the function that we wish to measure for our population to a sample of the data.

For example, suppose there is a company of 9,000 employees and we are interested in ascertaining the average length of breaks taken by employees in a single day. As we probably cannot ask every single person, we will take a sample of the 9,000 people and take a mean of the sample. This sample mean will be our point estimate.

The following code is broken into three parts:

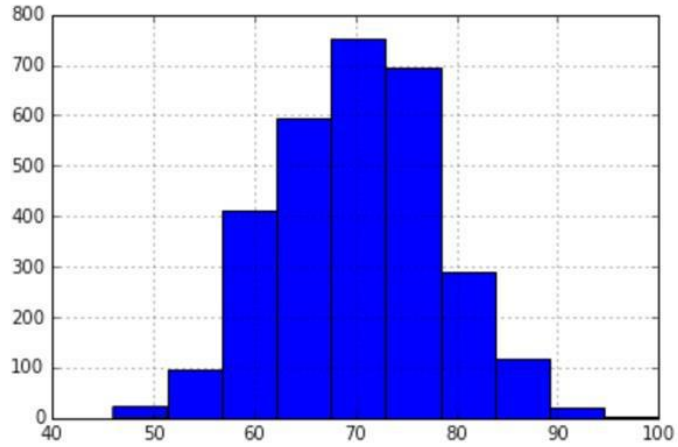
• We will use the probability distribution, known as the Poisson distribution, to randomly generate 9,000 answers to the question: for how many minutes in a day do you usually take breaks? This will represent our "population". Remember, from Chapter 6, Advanced Probability, that the Poisson random variable is used when we know the average value of an event and wish to model a distribution around it.

Let's take a look at the following code:

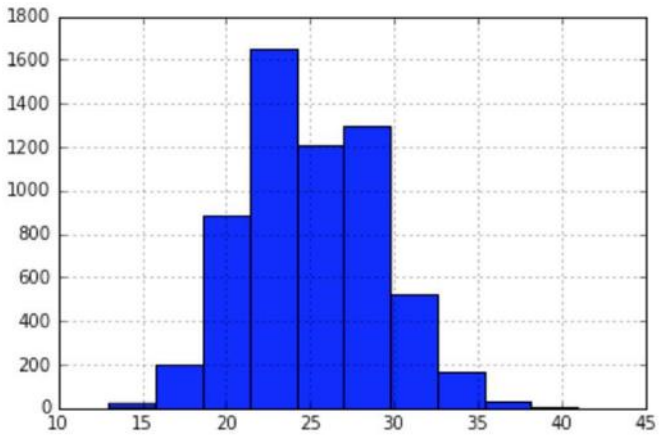
```
np.random.seed(1234)
```

long_breaks = stats.poisson.rvs(loc=10, mu=60, size=3000)
# represents 3000 people who take about a 60 minute break

The long_breaks variable represents 3000 answers to the question: how many minutes on an average do you take breaks for?, and these answers will be on the longer side. Let's see a visualization of this distribution, shown as follows:
pd.Series(long_breaks).hist()

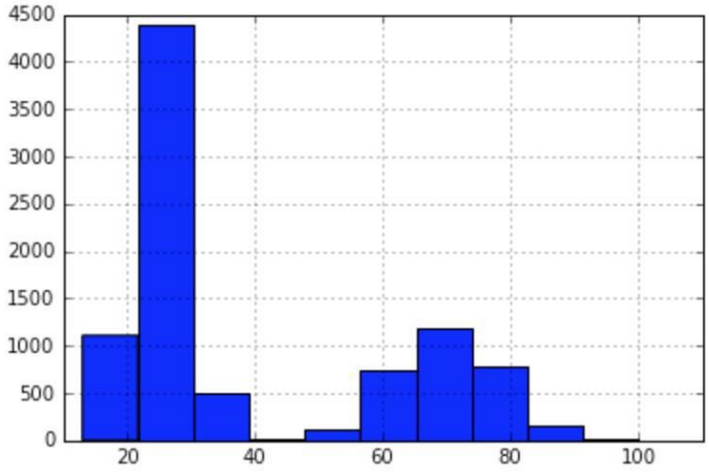


We see that our average of 60 minutes is to the left of the distribution. Also, because we only sampled 3000 people, our bins are at their highest around 700-800 people. Now, let's model 6000 people who take, on an average, about 15 minutes' worth of breaks. Let's again use the Poisson distribution to simulate 6000 people, as shown:
short_breaks = stats.poisson.rvs(loc=10, mu=15, size=6000)
# represents 6000 people who take about a 15 minute break
pd.Series(short_breaks).hist()



Again, note how our average
break length of 15 minutes falls to the left-hand side of the distribution, and note that the tallest bar is about 1600 people.
breaks = np.concatenate((long_breaks, short_breaks))
# put the two arrays together to get our "population" of 9000 people

pd.Series(breaks).hist()



We can find the total average break length by running the following code:

breaks.mean()

39.99 minutes is our parameter

So, to make our point, we want to simulate a world where we ask 100 random people about the length of their breaks. To do this, let's take a random sample of 100 employees out of the 9,000 employees we simulated, as shown:

sample_breaks = np.random.choice(a = breaks, size=100)

taking a sample of 100 employees

Now, let's take the mean of the sample and subtract it from the population mean and see how far off we were:

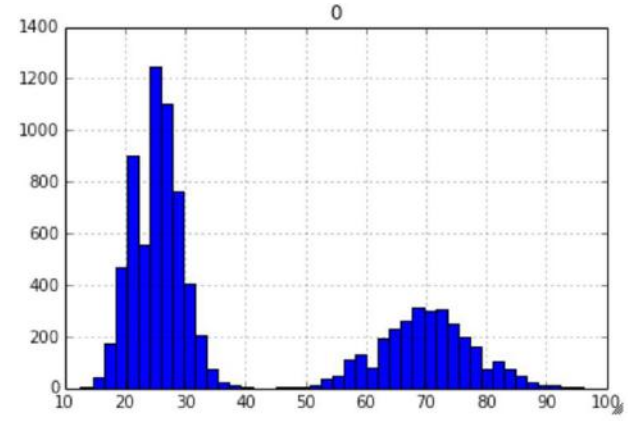
breaks.mean() - sample_breaks.mean()

difference between means is 4.09 minutes, not bad!

Sampling distributions

One of the reasons for this is that many statistical tests (including the ones we will use in this chapter) rely on data that follows a normal pattern, and for the most part, a lot of real-world data is not normal (surprised?)

pd.DataFrame(breaks).hist(bins=50,range=(5,100))



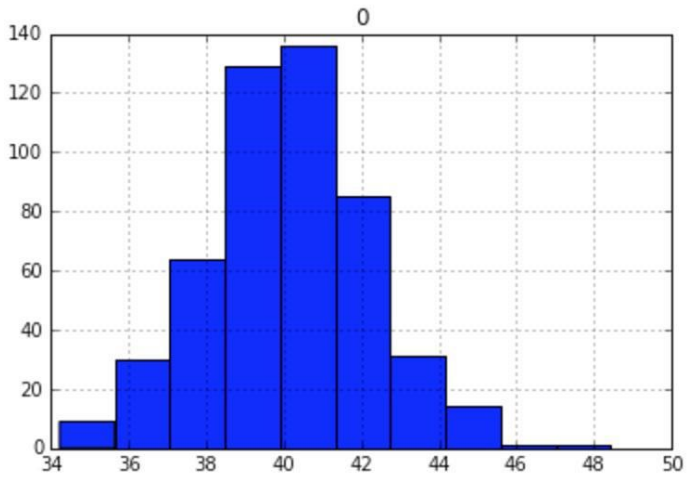
As you can see, our data is definitely not following a normal distribution, it appears to be bi-modal, which means that there are two peaks of break times, at around 25 and 70 minutes.

First off, we will need to utilize what is known as a sampling distribution, which is a distribution of point estimates of several samples of the same size. Our procedure for creating a sampling distribution will be the following:

1. Take 500 different samples of the break times of size 100 each.
2. Take a histogram of these 500 different point estimates (revealing their distribution).

The number of elements in the sample (100) was arbitrary, but large enough to be a representative sample of the population. The number of samples I took (500) was also arbitrary, but large enough to ensure that our data would converge to a normal distribution:

```
point_estimates = []
for x in range(500): # Generate 500 samples
sample = np.random.choice(a= breaks, size=100)
#take a sample of 100 points
point_estimates.append( sample.mean() )
# add the sample mean to our list of point estimates
pd.DataFrame(point_estimates).hist()
# look at the distribution of our sample means
```



Our data converged to a normal distribution because of something called the central limit theorem, which states that the sampling distribution (the distribution of point estimates) will approach a normal distribution as we increase the number of samples taken.

What's more, as we take more and more samples, the mean of the sampling

distribution will approach the true population mean, as shown:

```
breaks.mean() - np.array(point_estimates).mean()
# .047 minutes difference
```

Confidence intervals

While point estimates are okay estimates of a population parameter and sampling distributions are even better, there are the following two main issues with these approaches:

• Single point estimates are very prone to error (due to sampling bias among other things)

• Taking multiple samples of a certain size for sampling distributions might not be feasible, and may sometimes be even more infeasible than actually finding the population parameter For these reasons and more, we may turn to a concept, known as confidence interval, to find statistics.

A confidence interval is a range of values based on a point estimate that contains the true population parameter at some confidence level.

Confidence is an important concept in advanced statistics. Its meaning is sometimes misconstrued. Informally, a confidence level does not represent a "probability of being correct"; instead, it represents the frequency that the obtained answer will be accurate. For example, if you want to have a 95% chance of capturing the true population parameter using only a single point estimate, we would have to set our confidence level to 95%.

Calculating a confidence interval involves finding a point estimate, and then, incorporating a margin of error to create a range. The margin of error is a value that represents our certainty that our point estimate is accurate and is based on our desired confidence level, the variance of the data, and how big your sample is. There are many ways to calculate confidence intervals; for the purpose of brevity and simplicity, we will look at a single way of taking the confidence interval of a population mean. For this confidence interval, we need the following:

• A point estimate. For this, we will take our sample mean of break lengths from our previous example.

• An estimate of the population standard deviation, which represents the variance in the data.

° This is calculated by taking the sample standard deviation (the standard deviation of the sample data) and dividing that number by the square root of the population size.

• The degrees of freedom (which is the -1 sample size).

Obtaining these numbers might seem arbitrary but, trust me, there is a reason for all of them. However, again for simplicity, I will use prebuilt Python modules, as shown, to calculate our confidence interval and, then, demonstrate its value:

```
sample_size = 100
# the size of the sample we wish to take
```

```
sample = np.random.choice(a= breaks, size = sample_size)
# a sample of sample_size taken from the 9,000 breaks population from
before
sample_mean = sample.mean()
# the sample mean of the break lengths sample
sample_stdev = sample.std()
# sample standard deviation
sigma = sample_stdev/math.sqrt(sample_size)
# population standard deviation estimate
stats.t.interval(alpha = 0.95, # Confidence level 95%
df= sample_size - 1, # Degrees of freedom
loc = sample_mean, # Sample mean
scale = sigma) # Standard deviation
# (36.36, 45.44)
```

To reiterate, this range of values (from 36.36 to 45.44) represents a confidence interval for the average break time with a 95% confidence.
We already know that our population parameter is 39,99, and note that the interval includes the population mean of 39.99.
I mentioned earlier that the confidence level was not a percentage of accuracy of our interval but the percent chance that the interval would even contain the population parameter at all.
To better understand the confidence level, let's take 10,000 confidence intervals and see how often our population mean falls in the interval. First, let's make a function, as illustrated, that makes a single confidence interval from our breaks data:

```
# function to make confidence interval
def makeConfidenceInterval():
sample_size = 100
sample = np.random.choice(a= breaks, size = sample_size)
sample_mean = sample.mean()
# sample mean
sample_stdev = sample.std()
# sample standard deviation
sigma = sample_stdev/math.sqrt(sample_size)
# population Standard deviation estimate
return stats.t.interval(alpha = 0.95, df= sample_size - 1, loc =
sample_mean, scale = sigma)
```

Now that we have a function that will create a single confidence interval, let's create a procedure that will test the probability that a single confidence interval will contain the true population parameter, 39.99:

1. Take 10,000 confidence intervals of the sample mean.
2. Count the number of times that the population parameter falls into our confidence intervals.
3. Output the ratio of the number of times the parameter fell into the interval by 10,000:

```
times_in_interval = 0.
for i in range(10000):
interval = makeConfidenceInterval()
if 39.99 >= interval[0] and 39.99 <= interval[1]:
# if 39.99 falls in the interval
times_in_interval += 1
print times_in_interval / 10000
# 0.9455
```

Success! We see that about 95% of our confidence intervals contained our actual population mean. Estimating population parameters through point estimates and confidence intervals is a relatively simple and powerful form of statistical inference.

Hypothesis tests

Hypothesis tests are one of the most widely used tests in statistics. They come in many forms; however, all of them have the same basic purpose.

A hypothesis test is a statistical test that is used to ascertain whether we are allowed to assume that a certain condition is true for the entire population, given a data sample. Basically, a hypothesis test is a test for a certain hypothesis that we have about an entire population. The result of the test then tells us whether we should believe the hypothesis or reject it for an alternative one.

You can think of the hypothesis tests' framework to determine whether the observed sample data deviates from what was to be expected from the population itself. Now this sounds like a difficult task but, luckily, Python comes to the rescue and includes built-in libraries to conduct these tests easily.

A hypothesis test generally looks at two opposing hypotheses about a population.

We call them the null hypothesis and the alternative hypothesis. The null hypothesis is the statement being tested and is the default correct answer; it is our starting point and our original hypothesis. The alternative hypothesis is the statement that opposes the null hypothesis. Our test will tell us which hypothesis we should trust and which we should reject.

Based on sample data from a population, a hypothesis test determines whether or not to reject the null hypothesis. We usually use a p-value (which is based on our significance level) to make this conclusion.

The following are some examples of questions you can answer with a hypothesis test:
• Does the mean break time of employees differ from 40 minutes?
• Is there a difference between people who interacted with website A and people who interacted with website B (A/B testing)?
• Does a sample of coffee beans vary significantly in taste from the entire population of beans?

Conducting a hypothesis test

There are multiple types of hypothesis tests out there, and among them are dozens of different procedures and metrics. Nonetheless, there are five basic steps that most hypothesis tests follow, which are as follows:

1. Specify the hypotheses:
° Here, we formulate our two hypotheses: the null and the alternative
° We usually use the notation of H_0 to represent the null hypothesis and H_a to represent our alternative hypothesis

2. Determine the sample size for the test sample:
° This calculation depends on the chosen test. Usually, we have to determine a proper sample size in order to utilize theorems, such as the central limit theorem, and assume the normality of data.

3. Choose a significance level (usually called alpha or a):
° A significance level of 0.05 is common

4. Collect the data:
° They collect a sample of data to conduct the test

5. Decide whether to reject or fail to reject the null hypothesis:
° This step changes slightly based on the type of test being used. The final result will either yield in rejecting the null hypothesis in favor of the alternative or failing to reject the null hypothesis.

UNIT-4

IDENTIFYING EFFECTIVE AND INEFFECTIVE VISUALIZATIONS

SCOTTER PLOTS

A scatter plot is probably one of the simplest graphs to create. It is made by creating two quantitative axes and using data points to represent observations. The main goal of a scatter plot is to highlight relationships between two variables and, if possible,

reveal a correlation.

For example, we can look at two variables: average hours of TV watched in a day and a 0-100 scale of work performance (0 being very poor performance and 100 being excellent performance). The goal here is to find a relationship (if it exists) between watching TV and average work performance.

The following code simulates a survey of a few people, in which they revealed the

amount of television they watched, on an average, in a day against a companystandard work performance metric:

```
import pandas as pd

hours_tv_watched = [0, 0, 0, 1, 1.3, 1.4, 2, 2.1, 2.6, 3.2, 4.1, 4.4,4.4,5]

work_performance = [87, 89, 92, 90, 82, 80, 77, 80, 76, 85, 80, 75,73, 72]
```

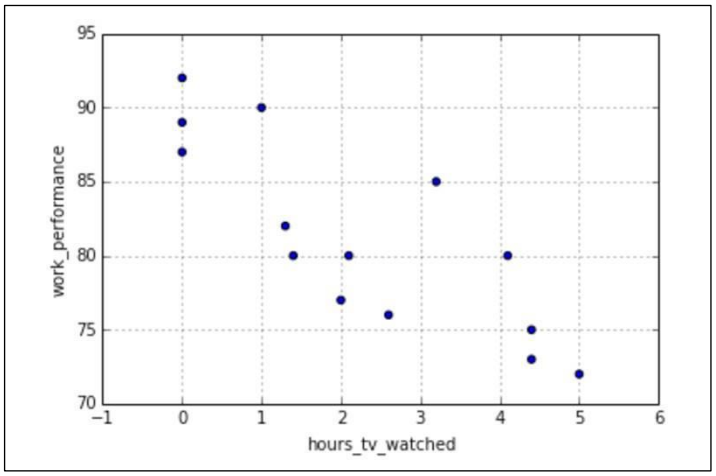
This line of code is creating 14 new sample survey results of the same people being rated on their work performance on a scale from 0 to 100.

```
df = pd.DataFrame({'hours_tv_watched':hours_tv_watched,
'work_performance':work_performance})
```

Here, we are creating a Dataframe in order to ease our exploratory data analysis and

make it easier to make a scatter plot:

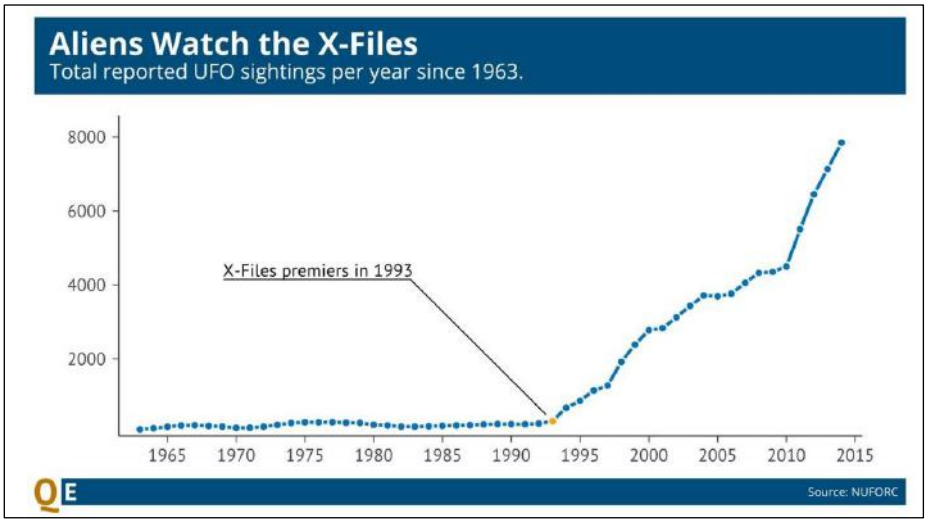
```
df.plot(x='hours_tv_watched', y='work_performance', kind='scatter')
```



Line graphs

Line graphs are, perhaps, one of the most widely used graphs in data communication. A line graph simply uses lines to connect data points and usually represents time on the x axis. Line graphs are a popular way to show changes in variables over time. The line graph, like the scatter plot, is used to plot quantitative variables.

As a great example, many of us wonder about the possible links between what we see on TV and our behavior in the world. A friend of mine once took this thought to an extreme—he wondered if he could find a relationship between the TV show, The X-Files, and the amount of UFO sightings in the U.S.. He then found the number of sightings of UFOs per year and plotted them over time. He then added a quick
graphic to ensure that readers would be able to identify the point in time when the X-files were released:



This graphic, albeit light-hearted, is an excellent example of a simple line graph. We are told what each axis measures, we can quickly see a general trend in the data, and we can identify with the author's intent, which is to show a relationship between the number of UFO sightings and the X-files premier.

On the other hand, the following is a less impressive line chart:

This line graph attempts to highlight the change in the price of gas by plotting three points in time. At first glance, it is not much different than the previous graph—we have time on the bottom x axis and a quantitative value on the vertical y axis. The (not so) subtle difference here is that the three points are equally spaced out on the x axis; however, if we read their actual time indications, they are not equally spaced out in time. A year separates the first two points whereas a mere 7 days separates the last two points.

Bar charts

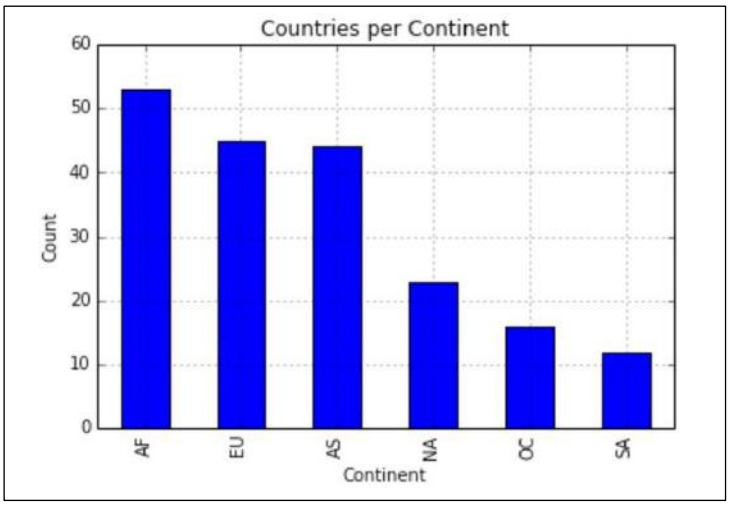
We generally turn to bar charts when trying to compare variables across different groups. For example, we can plot the number of countries per continent using a bar chart. Note how the x axis does not represent a quantitative variable, in fact, when using a bar chart, the x axis is generally a categorical variable, while the y axis is quantitative.

Note that, for this code, I am using the World Health Organization's report on

alcohol consumption around the world by country:

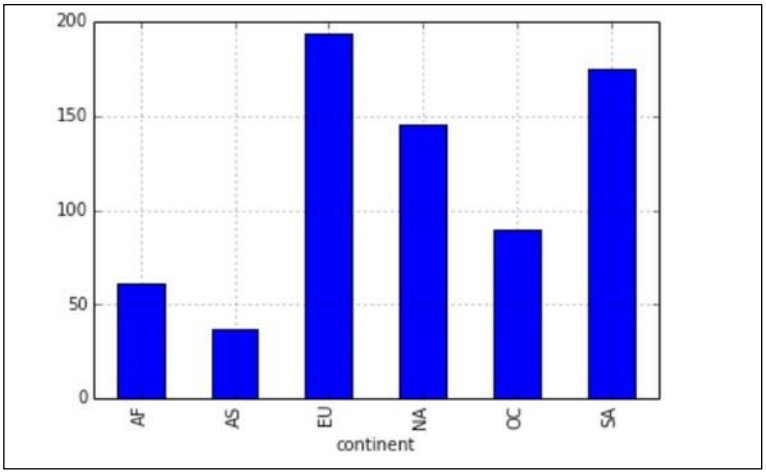
```
drinks = pd.read_csv('data/drinks.csv')

drinks.continent.value_counts().plot(kind='bar', title='Countries per Continent')
plt.xlabel('Continent')
plt.ylabel('Count')
```

The following graph shows us a count of the number of countries in each continent



In addition to the count of countries, we can also plot the average beer servings per continent using a bar chart, as shown:

drinks.groupby('continent').beer_servings.mean().plot(kind='bar')



Note how a scatter plot or a line graph would not be able to support this data because they can only handle quantitative variables; bar graphs have the ability to demonstrate categ

Histograms

Histograms show the frequency distribution of a single quantitative variable by splitting up the data, by range, into equidistant bins and plotting the raw count of observations in each bin. A histogram is effectively a bar chart where the x axis is a bin (subrange) of values and the y axis is a count. As an example, I will import a store's daily number of unique customers, as shown:

```
rossmann_sales = pd.read_csv('data/rossmann.csv')
rossmann_sales.head()
```

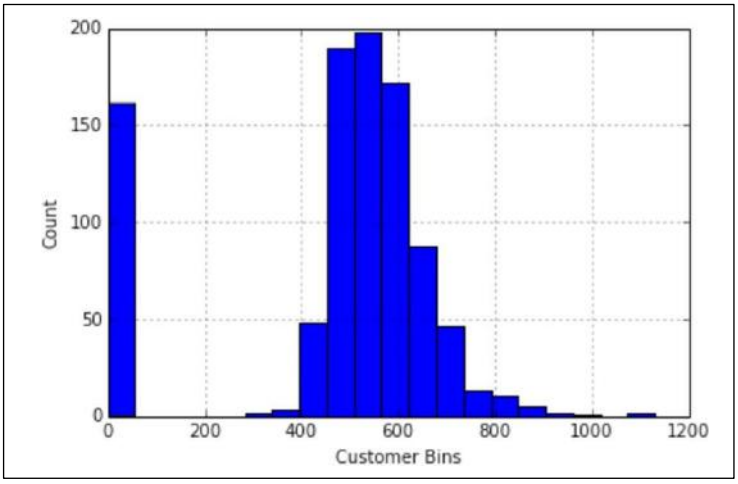
	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	5	2015-07-31	5263	555	1	1	0	1
1	2	5	2015-07-31	6064	625	1	1	0	1
2	3	5	2015-07-31	8314	821	1	1	0	1
3	4	5	2015-07-31	13995	1498	1	1	0	1
4	5	5	2015-07-31	4822	559	1	1	0	1

Note how we have multiple store data (by the first Store column). Let's subset this data for only the first store, as shown:

```
first_rossmann_sales = rossmann_sales[rossmann_sales['Store']==1]
```

Now, let's plot a histogram of the first store's customer count:

```
first_rossmann_sales['Customers'].hist(bins=20)
plt.xlabel('Customer Bins')
plt.ylabel('Count')
```



The x axis is now categorical in that each category is a selected range of values, for example, 600-620 customers would potentially be a category. The y axis, like a bar chart, is plotting the number of observations in each category. In this graph, for example, one

might take away the fact that most of the time, the number of customers on any given day will fall between 500 and 700.

Box plots

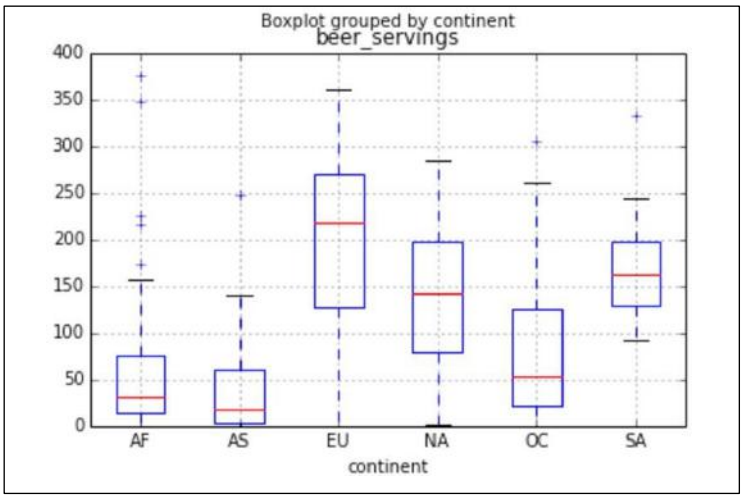
Box plots are also used to show a distribution of values. They are created by plotting the five number summary, as follows:

- The minimum value
- The first quartile (the number that separates the 25% lowest values from the rest)
- The median
- The third quartile (the number that separates the 25% highest values from the rest)
- The maximum value

In Pandas, when we create box plots, the red line denotes the median, the top of the box (or the right if it is horizontal) is the third quartile, and the bottom (left) part of the box is the first quartile.

The following is a series of box plots showing the distribution of beer consumption according to continents:

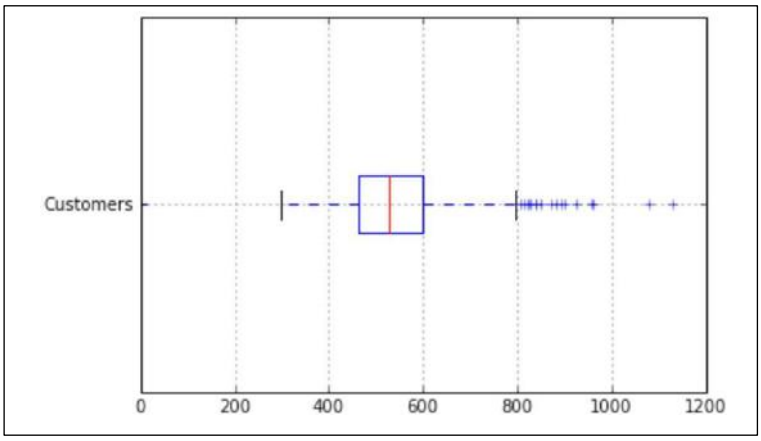
drinks.boxplot(column='beer_servings', by='continent')



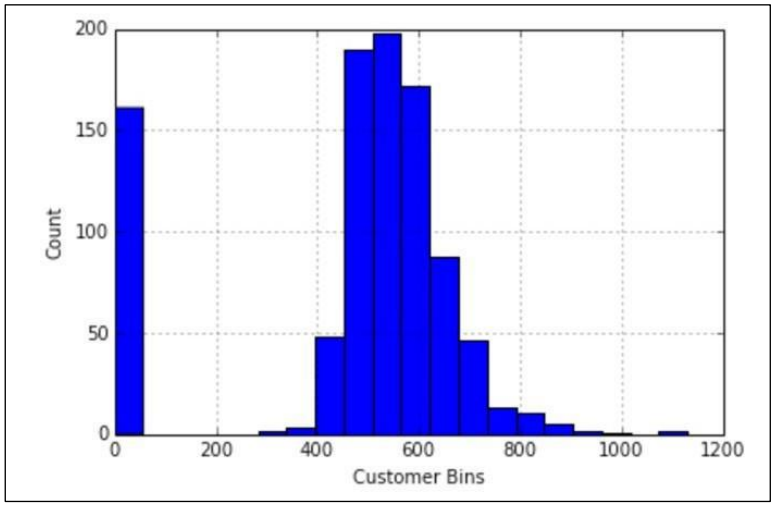
Now, we can clearly see the distribution of beer consumption across the seven continents and how they differ. Africa and Asia have a much lower median of beer consumption than Europe or North America.

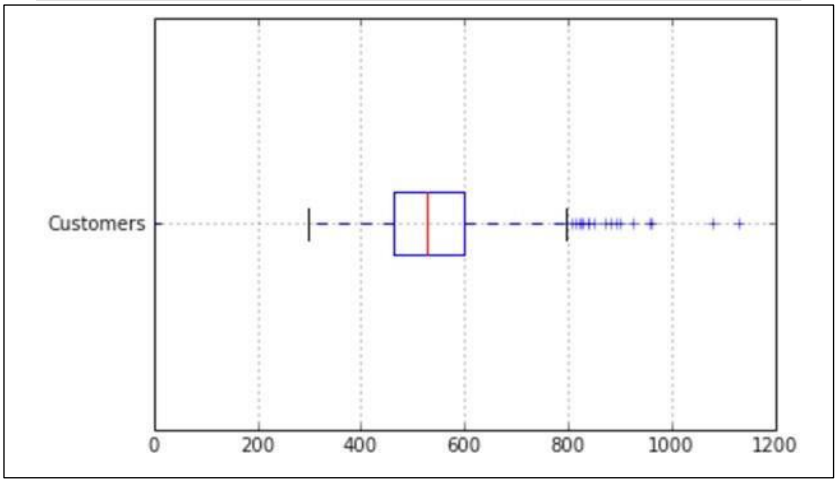
Getting back to the customer data, let's look at the same store customer numbers, but using a box plot:

first_rossmann_sales.boxplot(column='Customers', vert=False)



This is the exact same data as plotted earlier in the histogram; however, now it is shown as a box plot. For the purpose of comparison, I will show you both the graphs one after the other:





Note how the x axis for each graph are the same, ranging from 0 to 1,200. The box plot is much quicker at giving us a center of the data, the red line is the median, while the histogram works much better in showing us how spread out the data is and where

people's biggest bins are. For example, the histogram reveals that there is a very large bin of zero people. This means that for a little over 150 days of data, there were zero customers.

GRAPHS AND STATISTICS LIE

CORRELATION VERSUS CAUSATION

Correlation is a quantitative metric between -1 and 1 that measures how two variables move with each other. If two variables have a correlation close to -1, it means that as one variable increases, the other decreases, and if two variables have a correlation close to +1, it means that those variables move together in the same direction—as one increases, so does the other, and vice versa.

Causation is the idea that one variable affects another.

For example, we can look at two variables: the average hours of TV watched in a day and a 0-100 scale of work performance (0 being very poor performance and 100 being excellent performance). One might expect that these two factors are negatively correlated, which means that as the number of hours of TV watched increases in a 24 hour day, your overall work performance goes down. Recall the code from earlier, which is as follows:

```
import pandas as pd
hours_tv_watched = [0, 0, 0, 1, 1.3, 1.4, 2, 2.1, 2.6, 3.2, 4.1, 4.4,4.4, 5]
```

Here, I am looking at the same sample of 14 people as before and their answers to the question, how many hours of TV do you watch on average per night:
work_performance = [87, 89, 92, 90, 82, 80, 77, 80, 76, 85, 80, 75, 73, 72]
we have their work performance as

graded by the company or a third-party system:

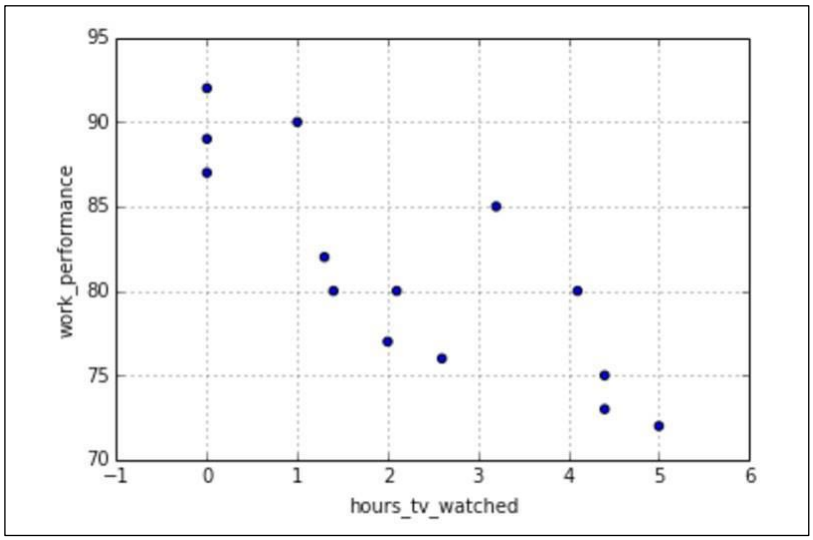
```
df = pd.DataFrame({'hours_tv_watched':hours_tv_watched,
'work_performance':work_performance})
```

Now we can introduce a new line of code that shows us the correlation between

these two variables:

```
df.corr() # -0.824
```

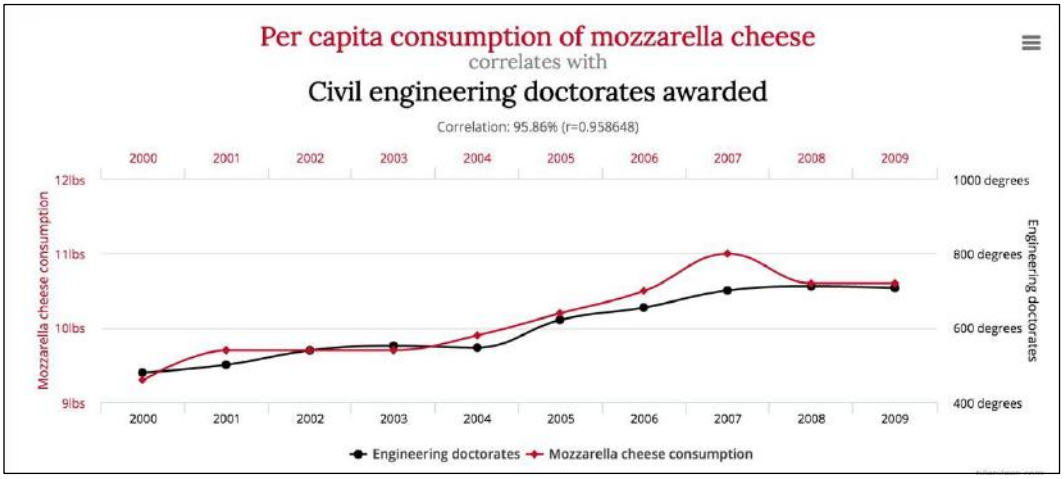
If your visuals and your numbers are off, people are less likely to take your analysis seriously:



What is more terrifying is that no one might know that the analysis is incomplete and you may make actionable decisions based on simple correlational work.

This can be for a variety of reasons, some of which are as follows:

• There might be a confounding factor between them. This means that there is a third lurking variable that is not being factored and that acts as a bridge between the two variables. For example, previously, we showed that you might find that the amount of TV you watch is negatively correlated with work performance, that is, as the number of hours of TV you watch increases, your overall work performance may decrease. That is a correlation. It doesn't seem quite right to suggest that watching TV is the actual cause of a decrease in the quality of work performance. It might seem more plausible to suggest that there is a third factor, perhaps hours of sleep every night, that might answer this question. Perhaps, watching more TV decreases the amount of time you have for sleep, which in turn limits your work performance.



It is much more likely that these two variables only happen to correlate (more strongly than our previous example, I may add) that cheese consumption determines the number of civil engineering doctorates in the world.

Simpson's paradox

Simpson's paradox is a formal reason for why we need to take confounding variables seriously. The paradox states that a correlation between two variables can be completely reversed when we take different factors into account. This means that even if a graph might show a positive correlation, these variables can become anti-correlated when another factor (most likely a confounding one) is taken into consideration. This can be very troublesome to statisticians.

Suppose we run a preliminary test and find the following conversion results:

Page A	Page B
75% (263/350)	83% (248/300)

This means that page B has almost a 10% higher conversion rate than page A. So, right off the bat, it seems like page B is the better choice because it has a higher rate of conversion. If we were going to communicate this data to our colleagues, it would seem that we are in the clear!

However, let's see what happens when we also take into account the coast that the user was closer to, as shown:

	Page A	Page B
West Coast	95% (76 / 80)	93% (231/250)
East Coast	72% (193/270)	34% (17/50)
Both	75% (263/350)	83% (248/300)

Thus the paradox! When we break the sample down by location, it seems that Page A was better in both categories but was worse overall. That's the beauty and, also, the horrifying nature of the paradox. This happens because of the unbalanced classes between the four groups.

The Page A / East Coast group as well as the Page B / West Coast group are providing most of the people in the sample, therefore skewing the results to be nonexpected. The confounding variable here might be the fact that the pages were given at different hours of the day and the west coast people were more likely to see page B, while the East coast people were more likely to see page A

The main takeaway from Simpson's paradox is that we should not unduly give causational power to correlated variables. There might be confounding variables that have to be examined. Therefore, if you are able to reveal a correlation between two variables (such as website category and conversation rate or TV consumption and work performance),

then you should absolutely try to isolate as many variables as possible that might be the reason for the correlation or can at least help explain your case further.

If correlation doesn't imply causation, then what does?

As a data scientist, it is often quite frustrating to work with correlations and not be able to draw conclusive causality. The best way to confidently obtain causality is, usually, through randomized experiments, such as the ones we saw in Chapter 8, Advanced Statistics.

VERBAL COMMUNICATION

Apart from visual demonstrations of data, verbal communication is just as important when presenting results. If you are not merely uploading results or publishing, you are usually presenting data to a room of data scientists, executives, or to a conference hall In any case, there are key areas to focus on when giving a verbal presentation, especially when the presentation is regarding findings about data.

There are generally two styles of oral presentations: one meant for more professional settings, including corporate offices where the problem at hand is usually tied directly to company performance or some other KPI (key performance indicator), and another meant more for a room of your peers where the key idea is to motivate the audience to care about your work.

It's about telling a story

Whether it is a formal or casual presentation, people like to hear stories. When you are presenting results, you are not just spitting out facts and metrics, you are attempting to frame the minds of your audience to believe in and care about what you have to say.

On the more formal side of things

When presenting data findings to a more formal audience, I like to stick to the following six steps:

1. Outline the state of the problem.

In this step, we go over the current state of the problem, including what the problem is and how the problem came to the attention of the team of data scientists.

2. Define the nature of the data.

Here, we go more in depth about who this problem affects, how the solution would change the situation, and previous work done on the problem, if any.

3. Divulge an initial hypothesis.

Here, we state what we believed to be the solution before doing any work. This might seem like a more novice approach to presentations; however, this can be a good time to outline not just your initial hypothesis, but, perhaps, the hypothesis of the entire company. For example, "we took a poll and 61% of the company believes there is no correlation between hours of TV watched and work performance".

4. Describe the solution and, possibly, the tools that led to the solution.

Get into how you solved the problem, any statistical tests used, and any assumptions that were made during the course of the problem.

5. Share the impact that your solution will have on the problem.

Talk about whether your solution was different from the initial hypothesis. What will this mean for the future? How can we take action on this solution to improve ourselves and our company?

6. Future steps.

Share what future steps can be taken with the problem, such as how to implement the said solution and what further work this research sparked.

The why/how/what strategy of presenting

When speaking on a less formal level, the why/how/what strategy is a quick and easy way to create a presentation worthy of praise. It is quite simple, as shown:

1. Tell your audience why this question is important without really getting into what you are actually doing.
2. Then, get into how you tackled this problem, using data mining, data cleaning, hypothesis testing, and so on.
3. Finally, tell them what your outcomes mean for the audience.

This model is borrowed from famous advertisements. The kind where they would not even tell you what the product was until 3 seconds left. They want to catch your attention and then, finally, reveal what it was that was so exciting. Consider the following example:

"Hello everyone, I am here to tell you about why we seem to have a hard time focusing on our job when the Olympics are being aired. After mining survey results and merging this data with company-standard work performance data, I was able to find a correlation between the number of hours of TV watched per day and average work performance. Knowing this, we can all be a bit more aware of our TV watching habits and make sure we don't let it affect our work. Thank you."

UNIT-5

APPLICATIONS OF DATA SCIENCE

TECHNOLOGIES FOR VISUALIZATION

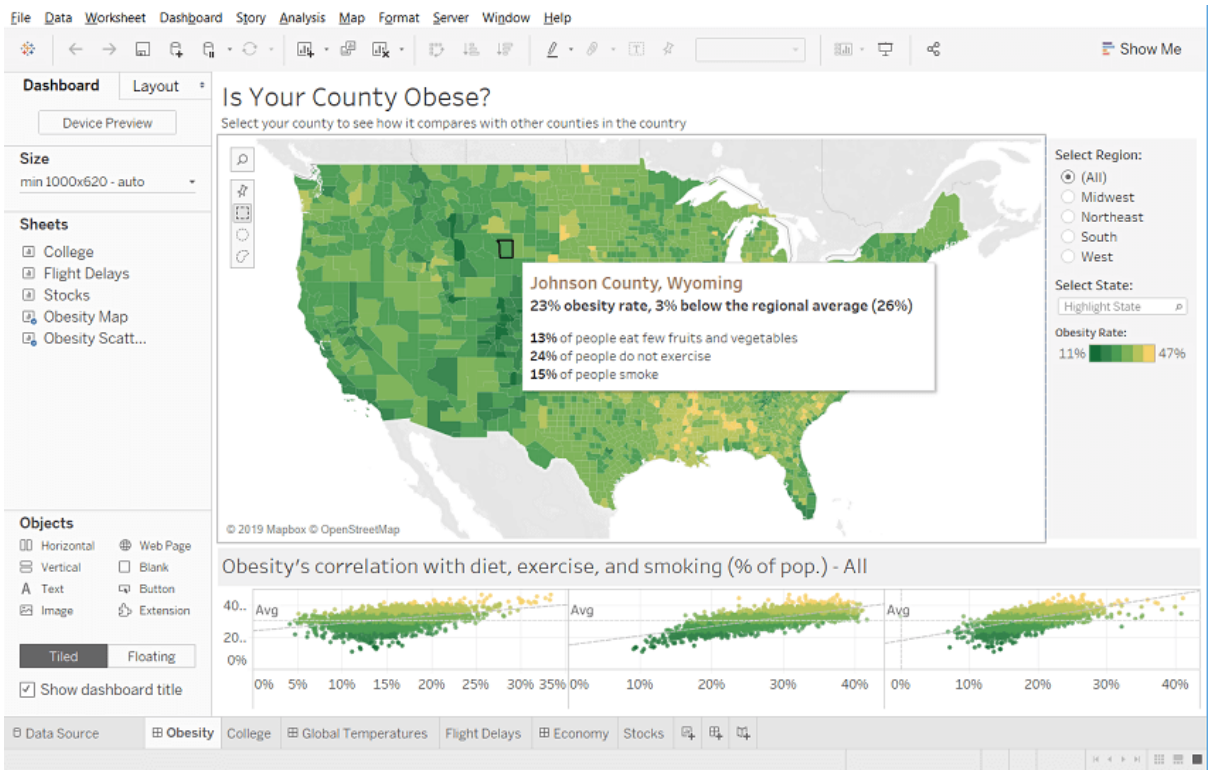
here are tools which help you to visualize all your data in a few minutes. They are already there; only you need to do is to pick the right data visualization tool as per your requirements.

Data visualization allows you to interact with data. Google, Apple, Facebook, and Twitter all ask better a better question of their data and make a better business decision by using data visualization.

Here are the top 10 data visualization tools that help you to visualize the data:

1. Tableau

Tableau is a data visualization tool. You can create graphs, charts, maps, and many other graphics.



A tableau desktop app is available for visual analytics. If you don't want to install tableau software on your desktop, then a server solution allows you to visualize your reports online and on mobile.

A cloud-hosted service also is an option for those who want the server solution but don't want to set up manually. The customers of Tableau include Barclays, Pandora, and Citrix.

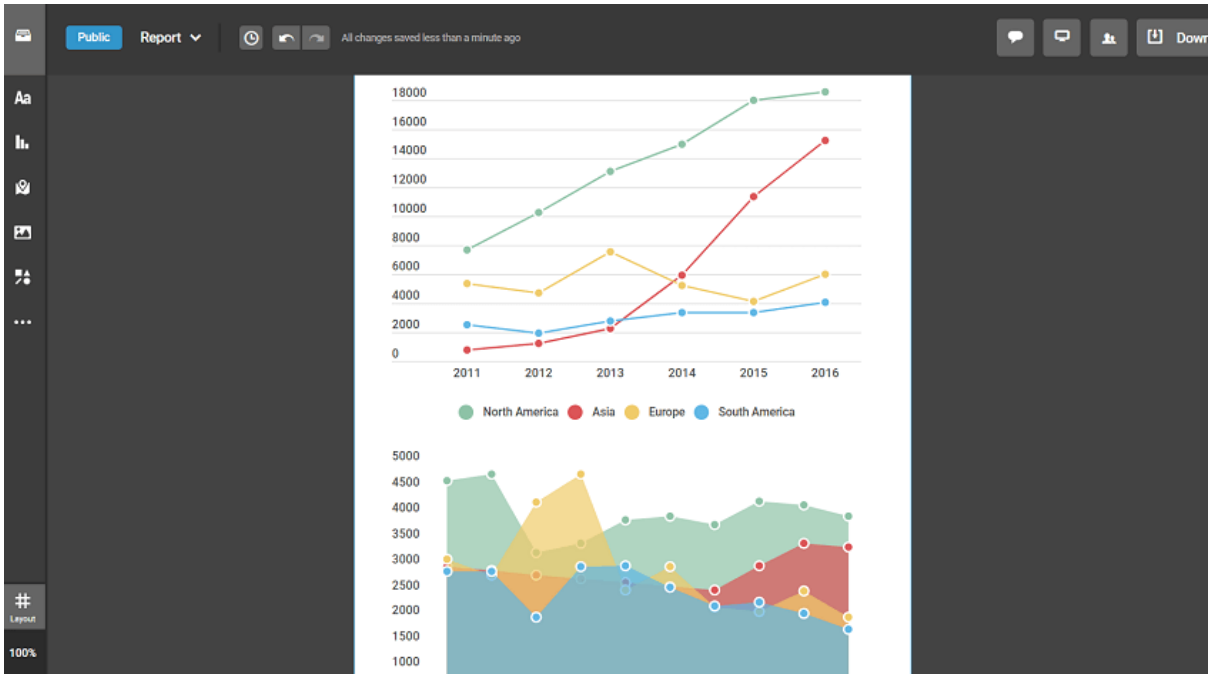
2. Infogram

Infogram is also a data visualization tool. It has some simple steps to process that:

First, you choose among many templates, personalize them with additional visualizations like maps, charts, videos, and images.

Then you are ready to share your visualization.

Infogram supports team accounts for journalists and media publishers, branded designs of classroom accounts for educational projects, companies, and enterprises.

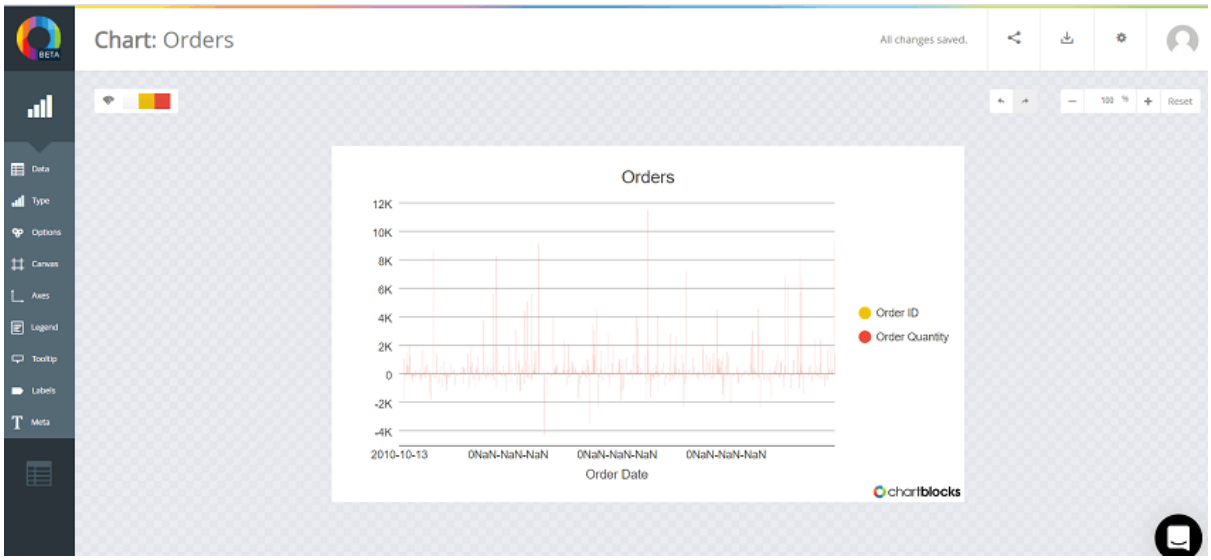


An infogram is a representation of information in a graphic format designed to make the data easily understandable in a view. Infogram is used to quickly communicate a message, to simplify the presentation of large amounts of the dataset, to see data patterns and relationships, and to monitor changes in variables over time.

Infogram abounds in almost any public environment such as traffic signs, subway maps, tag clouds, musical scores, and weather charts, among a huge number of possibilities.

3. Chartblocks

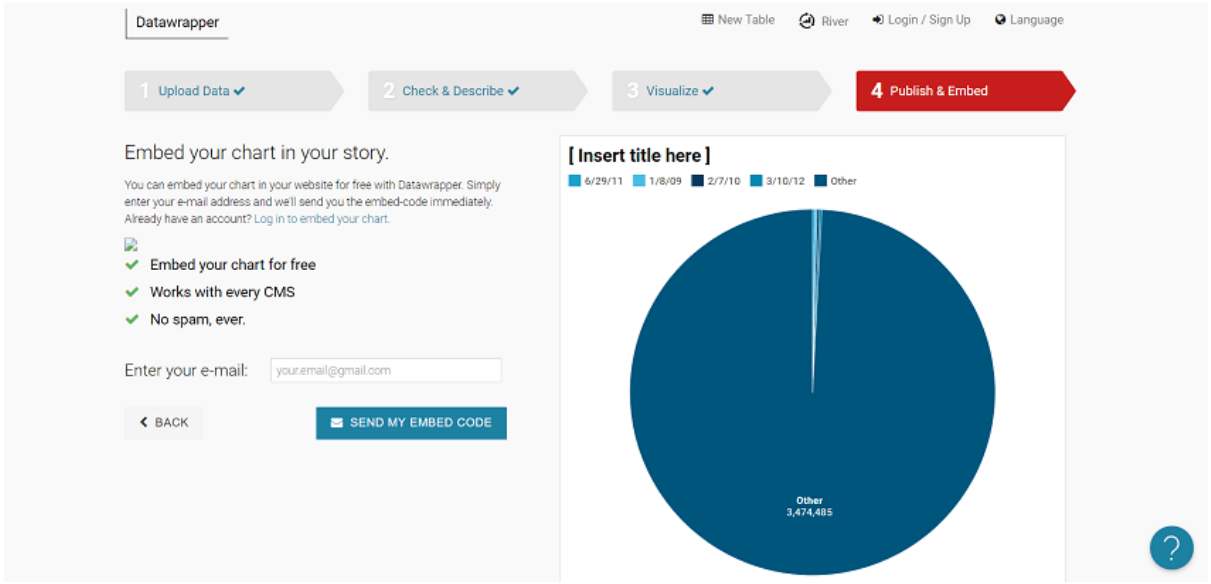
Chartblocks is an easy way to use online tool which required no coding and builds visualization from databases, spreadsheets, and live feeds.



Your chart is created under the hood in html5 by using the powerful JavaScript library D3.js. Your visualizations is responsive and compatible with any screen size and device. Also, you will be able to embed your charts on any web page, and you can share it on Facebook and Twitter.

4. Datawrapper

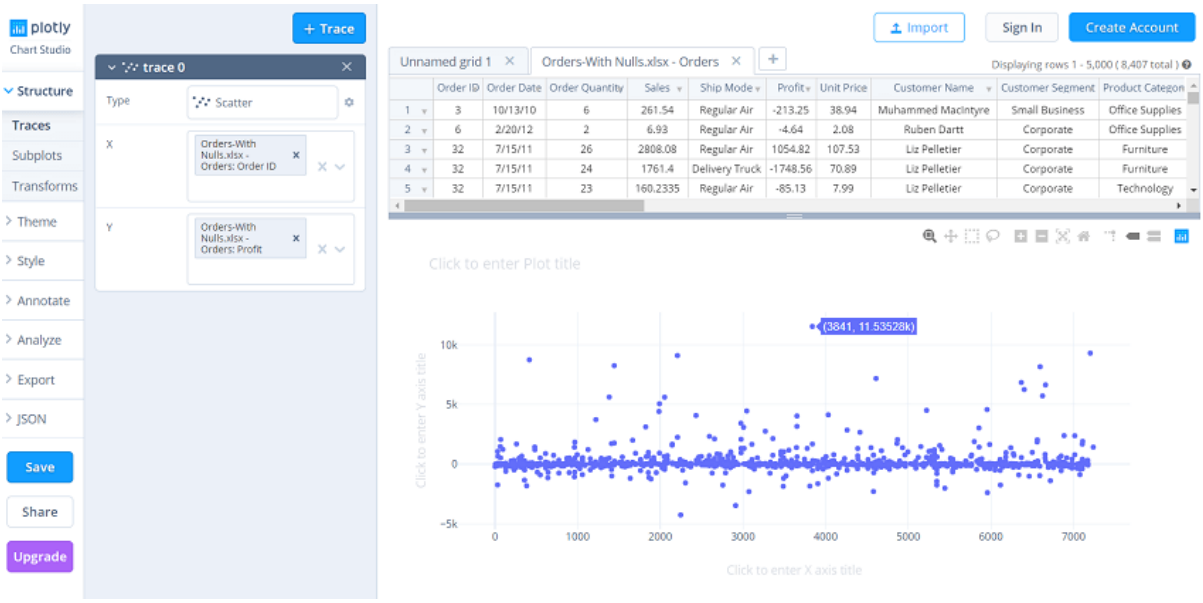
Datawrapper is an aimed squarely at publisher and journalist. The Washington Post, VOX, The Guardian, BuzzFeed, The Wall Street Journal and Twitter adopts it.



Datawrapper is easy visualization tool, and it requires zero codings. You can upload your data and easily create and publish a map or a chart. The custom layouts to integrate your visualizations perfectly on your site and access to local area maps are also available.

5. Plotly

Plotly will help you to create a slick and sharp chart in just a few minutes or in a very short time. It also starts from a simple spreadsheet.

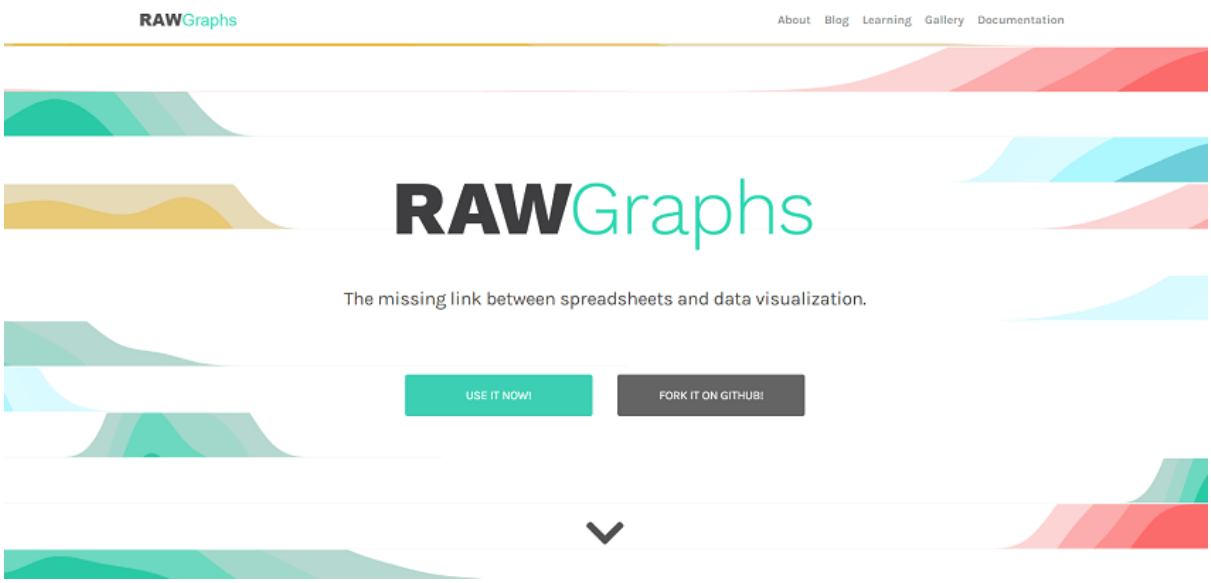


The guys use Plotly at Google and also by the US Air Force, Goji and The New York University.

Plotly is very user-friendly visualization tool which is quickly started within a few minutes. If you are a part of a team of developers that wants to have a crack, an API is available for JavaScript and Python languages.

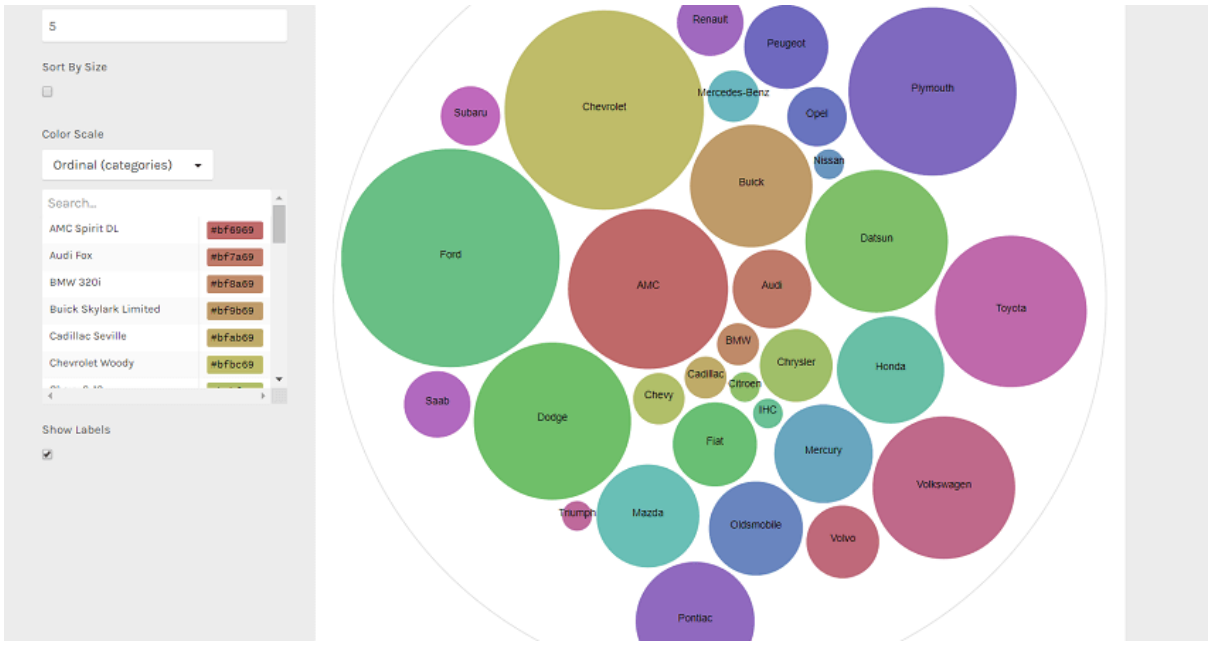
6. RAW

RAW creates the missing link between spreadsheets and vector graphics on its home page.



Your Data can come from Google Docs, Microsoft Excel, Apple Numbers, or a simple comma-separated list.

Here the kicker is that you can export your visualization easily and have a designer to make it look sharp. RAW is compatible with Inkscape, Adobe Illustrator, and Sketch. RAW is very easy to use and get quick results.



7. Visual.ly

Visual.ly is a visual content service. It has a dedicated data visualization service and their impressive portfolio that includes work for Nike, VISA, Twitter, Ford, The Huffington post, and the national geographic.

By a streamlined online process, you can find entire outsource your visualizations to a third-party where you describe your project and connected with a creative team that will stay with you for the entire duration of the project.

Visual.ly sends you an email notification for all the event you are hitting, and also it will give you constant feedback to your creative team. Visual.ly offer their distribution network for showcasing your project after it's completed.

BOKEH(PYTHON)

Bokeh is a data visualization library in Python that provides high-performance interactive charts and plots. Bokeh output can be obtained in various mediums like notebook, html and server. It is possible to embed bokeh plots in Django and flask apps.

Bokeh provides two visualization interfaces to users:

bokeh.models : A low level interface that provides high flexibility to application developers.

bokeh.plotting : A high level interface for creating visual glyphs.

To install bokeh package, run the following command in the terminal:

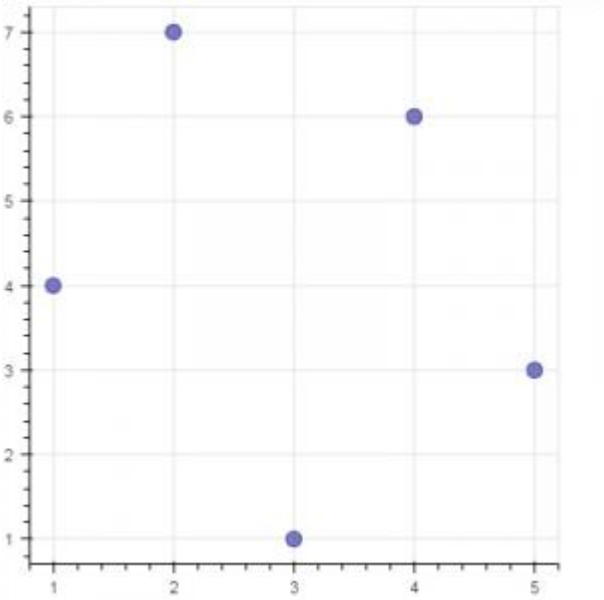
pip install bokeh

The dataset used for generating bokeh graphs is collected from Kaggle.

Code #1: Scatter Markers

To create scatter circle markers, circle() method is used.

```
# import modules
from bokeh.plotting import figure, output_notebook, show

# output to notebook
output_notebook()

# create figure
p = figure(plot_width = 400, plot_height = 400)

# add a circle renderer with
# size, color and alpha
p.circle([1, 2, 3, 4, 5], [4, 7, 1, 6, 3],
                size = 10, color = "navy", alpha = 0.5)

# show the results
show(p)
```

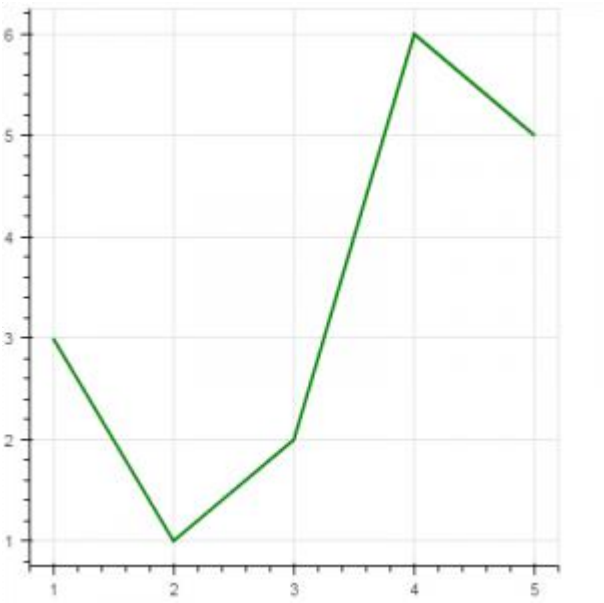


Code #2: Single line

To create a single line, line() method is used.

```
# import modules
from bokeh.plotting import figure, output_notebook, show

# output to notebook
output_notebook()

# create figure
p = figure(plot_width = 400, plot_height = 400)
```

```
# add a line renderer
p.line([1, 2, 3, 4, 5], [3, 1, 2, 6, 5],
                line_width = 2, color = "green")

# show the results
show(p)
```

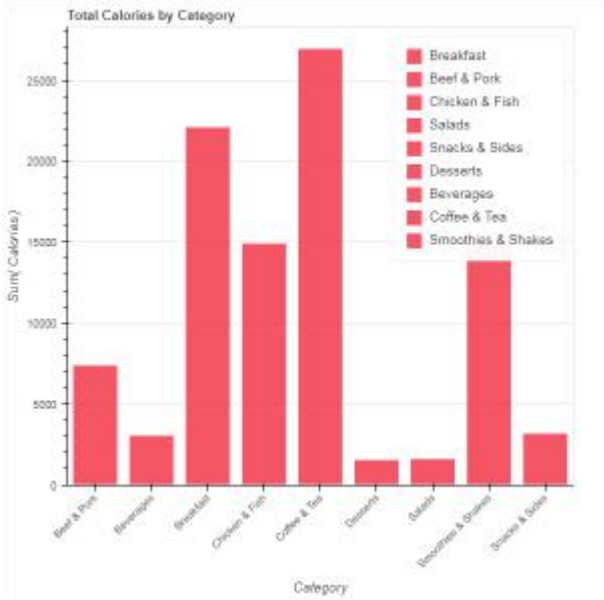


Code #3: Bar Chart

Bar chart presents categorical data with rectangular bars. The length of the bar is proportional to the values that are represented.

```
# import necessary modules
import pandas as pd
from bokeh.charts import Bar, output_notebook, show

# output to notebook
output_notebook()

# read data in dataframe
df = pd.read_csv(r"D:/kaggle/mcdonald/menu.csv")

# create bar
p = Bar(df, "Category", values = "Calories",
                title = "Total Calories by Category",
                                        legend = "top_right")
```

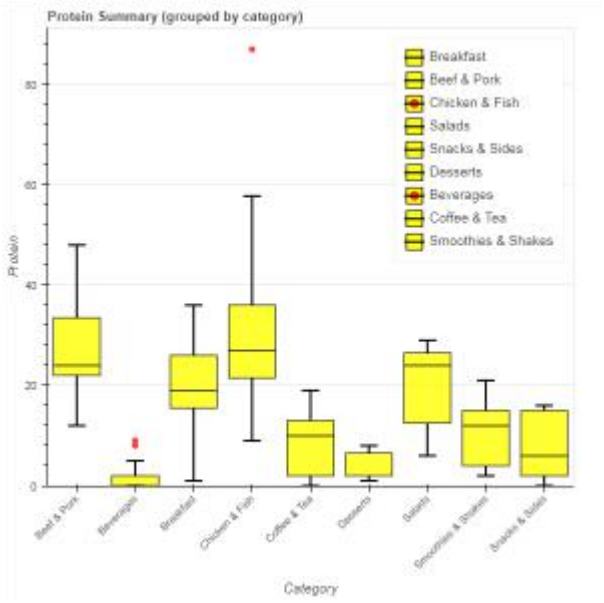
```
# show the results
show(p)
```



Code #4: Box Plot

Box plot is used to represent statistical data on a plot. It helps to summarize statistical properties of various data groups present in the data.

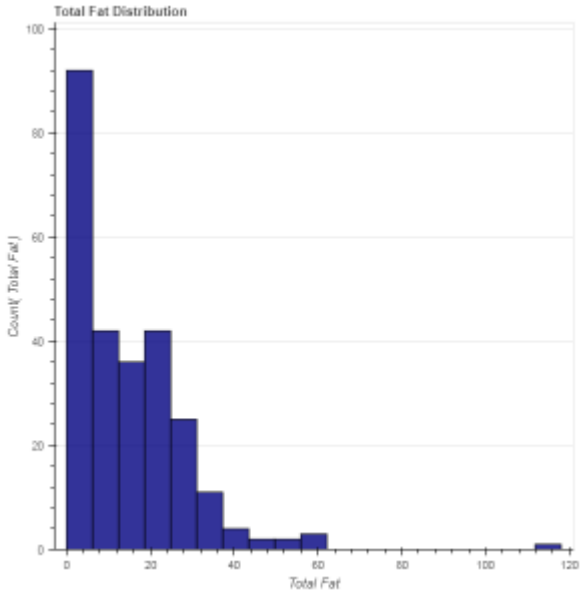
```
# import necessary modules
from bokeh.charts import BoxPlot, output_notebook, show
import pandas as pd

# output to notebook
output_notebook()

# read data in dataframe
df = pd.read_csv(r"D:/kaggle / mcdonald / menu.csv")

# create bar
p = BoxPlot(df, values = "Protein", label = "Category",
                        color = "yellow", title = "Protein Summary (grouped by category)",
                        legend = "top_right")

# show the results
show(p)
```



Code #5: Histogram

Histogram is used to represent distribution of numerical data. The height of a rectangle in a histogram is proportional to the frequency of values in a class interval.

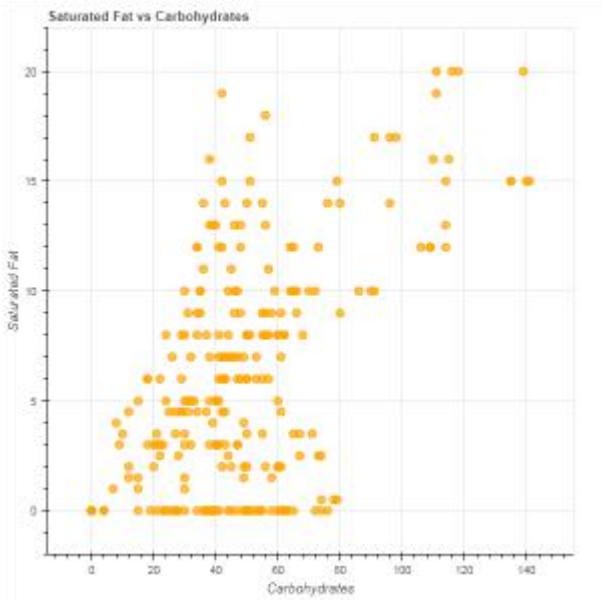
```
# import necessary modules
from bokeh.charts import Histogram, output_notebook, show
import pandas as pd

# output to notebook
output_notebook()

# read data in dataframe
df = pd.read_csv(r"D:/kaggle / mcdonald / menu.csv")

# create histogram
p = Histogram(df, values = "Total Fat",
                        title = "Total Fat Distribution",
                        color = "navy")

# show the results
show(p)
```



Code #6: Scatter plot

Scatter plot is used to plot values of two variables in a dataset. It helps to find correlation among the two variables that are selected.

```
# import necessary modules
from bokeh.charts import Scatter, output_notebook, show
import pandas as pd

# output to notebook
output_notebook()

# read data in dataframe
df = pd.read_csv(r"D:/kaggle / mcdonald / menu.csv")

# create scatter plot
p = Scatter(df, x = "Carbohydrates", y = "Saturated Fat",
                        title = "Saturated Fat vs Carbohydrates",
                        xlabel = "Carbohydrates", ylabel = "Saturated Fat",
                        color = "orange")

# show the results
show(p)
```



RECENT TRENDS IN VARIOUS DATA COLLECTION AND ANALYSIS TECHNIQUES

In today's current market trend, data is driving any organization in a countless number of ways. Data Science, Big Data Analytics, and Artificial Intelligence are the key trends in today's accelerating market. As more organizations are adopting data-driven models to streamline their business processes, the data analytics industry is seeing humongous growth. From fueling fact-based decision-making to adopting data-driven models to expanding data-focused product offerings, organizations are inclining more towards data analytics.

These progressing data analytics trends can help organizations deal with many changes and uncertainties. So, let's take a look at a few of these Data Analytics trends that are becoming an inherent part of the industry.

Trend 1: Smarter and Scalable Artificial Intelligence

COVID-19 has changed the business landscape in myriad ways and historical data is no more relevant. So, in place of traditional AI techniques, arriving in the market are some scalable and smarter Artificial Intelligence and Machine Learning techniques that can work with small data sets. These systems are highly adaptive, protect privacy, are much faster, and also provide a faster return on investment. The combination of AI and Big data can automate and reduce most of the manual tasks.

Trend 2: Agile and Composed Data & Analytics

Agile data and analytics models are capable of digital innovation, differentiation, and growth. The goal of edge and composable data analytics is to provide a user-friendly,

flexible, and smooth experience using multiple data analytics, AI, and ML solutions. This will not only enable leaders to connect business insights and actions but also, encourage collaboration, promote productivity, agility and evolve the analytics capabilities of the organization.

Trend 3: Hybrid Cloud Solutions and Cloud Computing

One of the biggest data trends for 2022 is the increase in the use of hybrid cloud services and cloud computation. Public clouds are cost-effective but do not provide high security whereas a private cloud is secure but more expensive. Hence, a hybrid cloud is a balance of both a public cloud and a private cloud where cost and security are balanced to offer more agility. This is achieved by using artificial intelligence and machine learning. Hybrid clouds are bringing change to organizations by offering a centralized database, data security, scalability of data, and much more at such a cheaper cost.

Trend 4: Data Fabric

A data fabric is a powerful architectural framework and set of data services that standardize data management practices and consistent capabilities across hybrid multi-cloud environments. With the current accelerating business trend as data becomes more complex, more organizations will rely on this framework since this technology can reuse and combine different integration styles, data hub skills, and technologies. It also reduces design, deployment, and maintenance time by 30%, 30%, and 70%, respectively, thereby reducing the complexity of the whole system. By 2026, it will be highly adopted as a re-architect solution in the form of an IaaS (Infrastructure as a Service) platform.

Trend 5: Edge Computing For Faster Analysis

There are many big data analytic tools available in the market but still persists the problems of enormous data processing capabilities. This has led to the development of the concept of quantum computing. By applying laws of quantum mechanics, computation has speeded up the processing capabilities of the enormous amount of data by using less bandwidth while also offering better security and data privacy. This is much better than classical computing as the decisions here are taken using quantum bits of a processor called Sycamore, which can solve a problem in just 200 seconds.

However, Edge Computing will need a lot of fine-tuning before it can be significantly adopted by organizations. Nevertheless, with the accelerating market trend, it will soon make its presence felt and will become an integral part of business processes.

Trend 6: Augmented Analytics

Augmented Analytics is another leading business analytics trend in today's corporate world. This is a concept of data analytics that uses Natural Language Processing, Machine Learning, and Artificial Intelligence to automate and enhance data analytics, data sharing, business intelligence, and insight discovery.

From assisting with data preparation to automating and processing data and deriving insights from it, Augmented Analytics is now doing the work of a Data Scientist. Data within the enterprise and outside the enterprise can also be combined with the help of augmented analytics and it makes the business processes relatively easier.

Trend 7: The Death of Predefined Dashboards

Earlier businesses were restricted to predefined static dashboards and manual data exploration restricted to data analysts or citizen data scientists. But it seems dashboards have outlived their utility due to the lack of their interactivity and user-friendliness. Questions are being raised about the utility and ROI of dashboards, leading organizations and business users to look for solutions that will enable them to explore data on their own and reduce maintenance costs.

It seems slowly business will be replaced by modern automated and dynamic BI tools that will present insights customized according to a user's needs and delivered to their point of consumption.

Trend 8: XOps

XOps has become a crucial part of business transformation processes with the adoption of Artificial Intelligence and Data Analytics across any organization. XOps started with DevOps that is a combination of development and operations and its goal is to improve business operations, efficiencies, and customer experiences by using the best practices of DevOps. It aims in ensuring reliability, re-usability, and repeatability and also ensure a reduction in the duplication of technology and processes. Overall, the primary aim of XOps is to enable economies of scale and help organizations to drive business values by delivering a flexible design and agile orchestration in affiliation with other software disciplines.

Trend 9: Engineered Decision Intelligence

Decision intelligence is gaining a lot of attention in today's market. It includes a wide range of decision-making and enables organizations to more quickly gain insights needed to drive actions for the business. It also includes conventional analytics, AI, and complex adaptive system applications. When combined with composability and common data fabric, engineering decision intelligence has great potential to help organizations rethink how

they optimize decision-making. In other words, engineered decision analytics is not made to replace humans, rather it can help to augment decisions taken by humans.

VARIOUS VISUALIZATION TECHNIQUES

Data visualization is a graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. This blog on data visualization techniques will help you understand detailed techniques and benefits.

In the world of Big Data, data visualization in Python tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.

Data Visualization Techniques

- Box plots
- Histograms
- Heat maps
- Charts
- Tree maps
- Word Cloud/Network diagram

Box Plots

The image above is a box plot. A boxplot is a standardized way of displaying the distribution of data based on a five-number summary ("minimum", first quartile (Q1), median, third quartile (Q3), and "maximum"). It can tell you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed.
A box plot is a graph that gives you a good indication of how the values in the data are spread out. Although box plots may seem primitive in comparison to a histogram or density plot, they have the advantage of taking up less space, which is useful when comparing distributions between many groups or datasets. For some distributions/datasets, you will find that you need more information than the measures of central tendency (median, mean, and mode). You need to have information on the variability or dispersion of the data.

List of Methods to Visualize Data

- **Column Chart:** It is also called a vertical bar chart where each category is represented by a rectangle. The height of the rectangle is proportional to the values that are plotted.

- **Bar Graph:** It has rectangular bars in which the lengths are proportional to the values which are represented.
- **Stacked Bar Graph:** It is a bar style graph that has various components stacked together so that apart from the bar, the components can also be compared to each other.
- **Stacked Column Chart:** It is similar to a stacked bar; however, the data is stacked horizontally.
- **Area Chart:** It combines the line chart and bar chart to show how the numeric values of one or more groups change over the progress of a viable area.
- **Dual Axis Chart:** It combines a column chart and a line chart and then compares the two variables.
- **Line Graph:** The data points are connected through a straight line; therefore, creating a representation of the changing trend.
- **Mekko Chart:** It can be called a two-dimensional stacked chart with varying column widths.
- **Pie Chart:** It is a chart where various components of a data set are presented in the form of a pie which represents their proportion in the entire data set.
- **Waterfall Chart:** With the help of this chart, the increasing effect of sequentially introduced positive or negative values can be understood.
- **Bubble Chart:** It is a multi-variable graph that is a hybrid of Scatter Plot and a Proportional Area Chart.
- **Scatter Plot Chart:** It is also called a scatter chart or scatter graph. Dots are used to denote values for two different numeric variables.
- **Bullet Graph:** It is a variation of a bar graph. A bullet graph is used to swap dashboard gauges and meters.
- **Funnel Chart:** The chart determines the flow of users with the help of a business or sales process.
- **Heat Map:** It is a technique of data visualization that shows the level of instances as color in two dimensions.

Histograms

A histogram is a graphical display of data using bars of different heights. In a histogram, each bar groups numbers into ranges. Taller bars show that more data falls in that range. A histogram displays the shape and spread of continuous sample data.

It is a plot that lets you discover, and show, the underlying frequency distribution (shape) of a set of continuous data. This allows the inspection of the data for its underlying distribution (e.g., normal distribution), outliers, skewness, etc. It is an accurate

representation of the distribution of numerical data, it relates only one variable. Includes bin or bucket- the range of values that divide the entire range of values into a series of intervals and then count how many values fall into each interval.

Bins are consecutive, non- overlapping intervals of a variable. As the adjacent bins leave no gaps, the rectangles of histogram touch each other to indicate that the original value is continuous.

Heat Maps

A heat map is data analysis software that uses colour the way a bar graph uses height and width: as a data visualization tool.

If you're looking at a web page and you want to know which areas get the most attention, a heat map shows you in a visual way that's easy to assimilate and make decisions from. It is a graphical representation of data where the individual values contained in a matrix are represented as colours. Useful for two purposes: for visualizing correlation tables and for visualizing missing values in the data. In both cases, the information is conveyed in a two-dimensional table.

Note that heat maps are useful when examining a large number of values, but they are not a replacement for more precise graphical displays, such as bar charts, because colour differences cannot be perceived accurately.

Charts

Line Chart

The simplest technique, a line plot is used to plot the relationship or dependence of one variable on another. To plot the relationship between the two variables, we can simply call the plot function.

Bar Charts

Bar charts are used for comparing the quantities of different categories or groups. Values of a category are represented with the help of bars and they can be configured with vertical or horizontal bars, with the length or height of each bar representing the value.

Pie Chart

It is a circular statistical graph which decides slices to illustrate numerical proportion. Here the arc length of each slide is proportional to the quantity it represents. As a rule, they are used to compare the parts of a whole and are most effective when there are limited components and when text and percentages are included to describe the content.

However, they can be difficult to interpret because the human eye has a hard time estimating areas and comparing visual angles.

Scatter Charts

Another common visualization technique is a scatter plot that is a two-dimensional plot representing the joint variation of two data items. Each marker (symbols such as dots, squares and plus signs) represents an observation. The marker position indicates the value for each observation. When you assign more than two measures, a scatter plot matrix is produced that is a series scatter plot displaying every possible pairing of the measures that are assigned to the visualization. Scatter plots are used for examining the relationship, or correlations, between X and Y variables.

Bubble Charts

It is a variation of scatter chart in which the data points are replaced with bubbles, and an additional dimension of data is represented in the size of the bubbles.

Timeline Charts

Timeline charts illustrate events, in chronological order — for example the progress of a project, advertising campaign, acquisition process — in whatever unit of time the data was recorded — for example week, month, year, quarter. It shows the chronological sequence of past or future events on a timescale.

Tree Maps

A treemap is a visualization that displays hierarchically organized data as a set of nested rectangles, parent elements being tiled with their child elements. The sizes and colours of rectangles are proportional to the values of the data points they represent. A leaf node rectangle has an area proportional to the specified dimension of the data. Depending on the choice, the leaf node is coloured, sized or both according to chosen attributes. They make efficient use of space, thus display thousands of items on the screen simultaneously.

APPLICATIONN DEVELOPMENT METHODS USED IN DATA SCIENCE

The field of data science has matured greatly in the past decade. And yet, teams often struggle to apply an appropriate data science methodology and team-based collaboration framework. Consider the following three issues:

Production Models: A one-off model often does not provide sustained value. Rather organizations often need a sustainable productized system that delivers model results

over a longer period of time. This necessitates a solid data science methodology that expands beyond just model development and into machine learning operations.

Team Approach: Data science is increasingly becoming a team sport. The concept of a back office unicorn data scientist working in isolation is not the norm. But rather, many projects have a diverse team consisting of multiple team roles. Thus, teams need to leverage a modern team-based approach to coordinate their work.

Agile Approach: Data science is a highly iterative process — especially once you extend beyond the classroom and into the real-world with real-time changing market conditions, technological shifts, and ever-evolving business needs. Long and inflexible upfront planning processes won't work. Rather, this gives rise to agile approaches. But how can you effectively apply Agility to data science?

Ad Hoc Approaches

Ad hoc processes focus on delivering a specific implementation without concern for broader impact or repeatable processes. In short, you can just "wing it".
This approach may work well for one-off, smaller, and low-impact projects. Think of a toy side project or an academic exercise.
Yet, the appropriate use cases for ad hoc in the real world are becoming less frequent. Unfortunately, many people still just result to Ad Hoc.

Data Science Methodologies

A data science life cycle (also known as a data science methodology) describes the step-by-step approach you take to deliver a project. Data scientists (even if they have not explicitly studied various methodologies) intuitively understand these steps. Documenting them can help increase repeatably and prevent you from forgetting a step. This is increasingly important in the world of distributed teams that extend beyond data science to areas such as legal or business.

There are dozens of different defined data science methodologies. This guide explores the most well-known.

Approach	Description	Strengths	Challenges	Best For…
Waterfall	Plan your work. Work your plan	• Easily understood • Matches traditional corporate culture	• Inflexible • Delays testing • Documentation heavy	• Avoid for data science

Approach	Description	Strengths	Challenges	Best For…
KDD	5 Phases from Selection to Evaluation	• Decent explanation of core data mining technical project	• Outdated • Ignores teams • Many same shortcomings as Waterfall • Ignores biz understanding & deployment	• "Toy" projects with a well-defined scope that don't need productized
SEMMA	5 Phases from Sample to Assess	• Decent explanation of core data mining technical project	• Outdated • Ignores teams • Many same shortcomings as Waterfall • Ignores biz understanding & deployment	• "Toy" projects with a well-defined scope that don't need productized
CRISP-DM	6 Phases from Business Understanding to Deployment	• Well-known • More comprehensive than KDD, SEMMA • Defined guide	• Outdated • Ignores teams • Many same shortcomings as Waterfall	• Teams looking for an established practice
TDSP	Combines CRISP-DM and Scrum practices	• Comprehensive open-source documentation	• Includes Agile concepts • Strong team focus	• Teams looking to "modernize" CRISP-DM
Domino	Combines CRISP-DM and Agile practices	• Visual roadmap with clear flow and decision points • Includes practical tips	• More of a concept as opposed to a fully vetted approach	• Teams looking to "modernize" CRISP-DM
Others	Lesser-known life cycles	• Each includes a novel viewpoint	• Not well-known or vetted	• Good "food for thought"