

ADA511: Data science and data-driven engineering

Steffen Mæland
PierGianLuca Porta Mana

2023-06-06

Table of contents

Preface	5
1 Introduction	6
2 Framework and first building blocks	7
2.1 The framework: Decision Theory	7
2.2 First basic notions	9
2.2.1 Agents	9
2.2.2 Sentences: data, information, hypotheses	10
2.3 Well-posed and ill-posed sentences	11
Reading list	12
3 Data: use and communication	13
3.1 Sentences – or, what is “data”?	13
3.2 Well-posed and ill-posed sentences	14
Reading list	15
4 Inference	16
4.1 What is inference?	16
4.2 Certain and uncertain inference	17
5 Truth inference	19
5.1 Building blocks	19
5.1.1 Basic sentences	19
5.1.2 Connectives	20
5.1.3 Data or axioms	21
5.2 Truth-inference rules	22
5.3 Logical AI agents and their limitations	23
6 Probability inference	25
6.1 When truth isn’t known: probability	25
6.2 No new building blocks	28
6.3 Probability-inference rules	28
6.4 How the inference rules are used	30
6.4.1 Derived rules	31

6.5	Law of total probability or “extension of discourse”	32
6.5.1	Bayes’s theorem	32
6.6	Common points of certain and uncertain inference	32
7	Data and information	33
7.1	Kinds of data	33
7.1.1	Binary	33
7.1.2	Nominal	33
7.1.3	Ordinal	33
7.1.4	Continuous	33
7.1.5	Complex data	33
7.1.6	“Soft” data	33
7.2	Data transformations	33
8	Allocation of uncertainty among possible data values: probability distributions	34
8.1	The difference between Statistics and Probability Theory	34
8.2	What’s “distributed”?	34
8.3	Distributions of probability	34
8.3.1	Representations	34
8.4	Summaries of distributions of probability	35
8.4.1	Location	35
8.4.2	Dispersion or range	35
8.4.3	Resolution	35
8.4.4	Behaviour of summaries under transformations of data and errors in data	35
8.5	Outliers and out-of-population data	35
8.6	Marginal and conditional distributions of probability	35
8.7	Collecting and sampling data	35
8.7.1	“Representative” samples	35
8.7.2	Unavoidable sampling biases	36
8.8	Quirks and warnings about high-dimensional data	36
9	Making decisions	37
9.1	Decisions, possible situations, and consequences	37
9.2	Gains and losses: utilities	37
9.2.1	Factors that enter utility quantification	37
9.3	Making decisions under uncertainty: maximization of expected utility	37

Preface

****WARNING: THIS IS A WORKING DRAFT. TEXT WILL CHANGE A LOT. MANY PASSAGES ARE JUST TEMPORARY, INCOHERENT, AND DISJOINTED.**

To be written.

1 Introduction

To be written: motivation and structure of this course.

2 Framework and first building blocks

Every data-driven engineering problem is unique. But there also are similarities among all engineering problems. We shall now learn a framework, some notions, and a set of principles that allow us to frame and face any data-driven engineering problem, and any sub-problems into which a problem can be broken down. The set of principles is important because it mathematically guarantees an optimal solution to the problem – within the goals, means, and data into which we framed the problem.

2.1 The framework: Decision Theory

There is always a goal underlying any engineering problem. The problem itself is how to reach that goal. Typically there are several possible courses of actions available; the question is which one to choose. Choice of a specific action will lead to a set of consequences; this could be the goal we want to reach, but it could also be something else, possibly undesirable. The decision upon a specific action is often difficult because its consequences are not known with certainty. This uncertainty comes in turn from a more general uncertainty about the whole context of the problem: past or present unknown details, unknown future events and responses, and so on.

@@ example

This is what we call a **decision problem**.

A specific course of action may in turn be realized in several ways that are equivalent in regard to the outcome, but inequivalent in regard to costs, time, resources. We thus face a choice within a choice. In general, a decision problem may

involve several decision sub-problems, in turn involving sub-sub-problems, and so on.

The main engineering goal itself could be to design and build a device that chooses an optimal course of action in a specific kind of uncertain situation, in an automated way. Think for instance of an aeronautic engineer who is designing an autopilot system.

To analyse and face this kind of situations, we would therefore like to have a framework that takes into consideration choices, consequences, costs and gains, and uncertainties. It must also be suited to recursive application if needed. And it must be suited to being used not only for human engineers, but also for automated or artificial-intelligence devices.

Such a framework turns out to exist: it is called **Decision Theory**.

Decision theory has a long history, going back to Leibniz in the 1600s or maybe even to Aristotle in the –300s, and appearing in its present form around 1920–1960. What’s remarkable about it is that it is not only *a* framework, but *the* framework we must use. A logico-mathematical theorem shows that any framework that does not break basic optimality and rationality criteria has to be equivalent to Decision Theory (in other words, it can use different technical terminology and rewrite mathematical operations in a different way, but it boils down to the same notions and operations of Decision Theory). So if you wanted to invent and use another framework, then it either (a) would lead to some irrational or illogical consequences, or (b) would lead to results identical to Decision Theory’s. Many frameworks that you are probably familiar with, such as optimization theory, are just specific applications or particular cases of Decision Theory.

Decision theory consists of two main parts: **Probability Theory**, which deals with data, information, uncertainty, inference; and **Utility Theory**, which deals with actions, consequences, gain and loss, choice.

We shall get acquainted with Decision Theory step by step, introducing its main ideas and notions as they become necessary. Let us start with the first building blocks.



Remember: the main goal is to be able to identify the notions in any data-driven engineering problem. What matters is not their technical names or formal definitions, but their roles and actual use in a concrete engineering problem.

2.2 First basic notions

2.2.1 Agents

The agent is the person or device that has to make a choice between different courses of action. An agent has a specific set of data and background information available, a specific set of choices, and can incur specific gains or losses dependent on the consequences of the available choices.

It is important to identify the agent or agents involved in a problem, because each one will generally have different data, or different available choices, or different gains and losses. A person buying an insurance policy from an insurance company is an example of two agents that have roughly the same data and a common course of action (buy-sell) that is optimal for both. The optimality comes from the fact that the two agents have very different gains and losses for their various courses of action.

Reading

§ 1.1.4 in *Artificial Intelligence*

Notation

When necessary, agents are typically denoted by capital letters: A, B, \dots . But we'll rarely need symbols for them.

2.2.2 Sentences: data, information, hypotheses

What is “data”? “Data” (from Latin “given”) is used more or less in the same sense as “information”, and in these notes we’ll use the two words as synonyms.

Data is often presented as numbers; but it’s obviously more than that. I give you this number: “8”. Is it “data”? what is it about? what should you do with it? We can hardly call this number a piece of information, since we have no clue what we could do with it. Instead, if I tell you: “*The number of official planets in the solar system is 8*”, then we can say that I’ve given you data. So “data” is not just numbers: a number is not “data” unless there’s some verbal, non-numeric context associated with it – even if this context is only implicitly understood.

Data can also be completely non-numeric. A clinician saying “*The patient has fully recovered from the disease*” (we imagine to know who’s the patient and what was the disease) is giving us a piece of information that we could further use, for instance, to make prognoses about other, similar patients. The clinician’s statement surely is “data”. It is essentially non-numeric data, even if in some situations we can represent it as “1”, say, while “0” would represent “not recovered”.

From these two examples, and with some further thought, we realize that “data” – and in general any piece of information – can universally be represented and communicated by **sentences**, also called *propositions* or *statements*¹. In some cases we can summarize a sentence by a number, as a shorthand, when the full meaning of the sentence is understood.

Sentences also represent and convey *hypotheses*.

Recognizing that data, information, hypotheses are ultimately represented by sentences has important practical consequences:

- **Clarity, analysis, goal-orientation.** A data engineer must acquire information and convey information. Acquiring information is not simply making some measurement or counting something: the engineer must understand *what* is being measured and *why*. If data is

¹These three terms are not always equivalent in Formal Logic, but here we’ll use them as synonyms.

gathered from third parties, the engineer must ask what exactly the data mean and how they were acquired. In designing and engineering a solution, it is important to understand what information or outcomes the end user exactly wants. As a data engineer it will often happen that you ask “wait, what do you mean by that?”; this question is not just an unofficial parenthesis in the official data-transfer workflow between you and someone else. It is an integral part of that workflow; it means that the data has not been completely transferred yet.

- **Artificial Intelligence.** Sentences are the central components of knowledge representation and inference in artificial-intelligence agents.

Reading

§ 7.1 in *Artificial Intelligence*

Notation

We'll denote sentences by sans-serif italic letters: A, B, a, b, \dots
For example,

$$O := \text{'The power output is 100 W'}$$

means that the symbol O stands for the sentence above. In the next chapters we'll see how more complex sentences are built from simpler ones. No matter whether complex or simple, any sentence can be represented by symbols like the ones above.

A set of data, being a collection of sentences, will also be denoted by this kind of symbols, very often with D .

2.3 Well-posed and ill-posed sentences

We face problems when the sentences that should convey information and data are not clear. Suppose that an electric-car model consumes 150 Wh/km and has a range of 200 km; a second car model consumes 250 Wh/km and has a range of 600 km. Someone says “I think the second model is better;

what do you think?”. It isn’t clear how we should answer; what does “better” mean? If it refers to consumption, then the first car model is “better”. If it refers to range, then the second model is “better”. If it refers to a combination of these two characteristics, or to something else, then we simply can’t answer. Here we have a problem with querying and giving data, because the sentence underlying such query is not clear.

We say that such sentences are **not well-posed**, or that they are **ill-posed**.

This may seem an obvious discussion to you. Yet you’d be surprised by how often unclear sentences appear in scientific papers about data engineering! Not seldom we find discussions and disagreements that actually come from unclear underlying sentences, that two parties interpret in different ways.

As a data engineer, you’ll often have the upper hand if you are on the lookout for ill-posed sentences. Whenever you face an important question, or you’re given an important piece of information, or you must provide an important piece of information, *always take a little time to examine whether the question or information is actually well-posed*.

@@ [TODO] Exercise: give actual paper to analyse__

Reading list

3 Data: use and communication

3.1 Sentences – or, what is “data”?

What is “data”?

“Data” (from Latin “given”) is used more or less in the same sense as “information”, and in these notes we’ll use the two words as synonyms.

“Data” is often presented as numbers; but it’s obviously more than that. I give you this number: “8”. Is it “data”? what is it about? what should you do with it? We can hardly call this number a piece of information, since we have no clue what we could do with it. Instead, if I tell you: “*The number of official planets in the solar system is 8*”, then we can say that I’ve given you data. So “data” is not just numbers. A number is not “data” unless there’s some verbal, non-numeric context associated with it – even if this context is only implicitly understood.

Data can also be completely non-numeric. A clinician saying “*The patient has fully recovered from the disease*” (we imagine to know who’s the patient and what was the disease) is giving us a piece of information that we could further use, for instance, to make prognoses about other, similar patients. The clinician’s statement surely is “data”. It is essentially non-numeric data, even if in some situations we can represent it as “1”, say, while “0” would represent “not recovered”.

From these two examples, and with some further thought, we realize that “data” – and in general any piece of information or hypothesis – can universally be represented and communicated by *sentences*, also called *statements* or *propositions*¹. In some cases we can summarize or represent such sentences

¹These terms are not equivalent in Logic, but sometimes we’ll use them as synonyms.

as numbers. But the numbers alone, by themselves, are not data.

So our conclusion is that *information* or *data* is represented by *sentences*.

Recognizing that data and information are ultimately sentences has important practical consequences:

Clarity and goal-orientation. As a data engineer you'll have to acquire information and convey information. Acquiring information is not simply making some measurement or counting something: you must understand *what* you are measuring and *why*. If you gather data from third parties, you have to ask what exactly the data mean and how they were acquired. In designing and engineering a solution, you'll have to understand what information or outcomes the end user exactly wants. It will often happen that you ask "wait, what do you mean by that?"; this question is not just an unofficial parenthesis in the official data-transfer workflow between you and someone else: it is an integral part of that workflow, it means that the data has not been completely transferred yet.

Artificial Intelligence Sentences are the central components of knowledge representation and inference in artificial-intelligence agents.

Reading

§ 7.1 in *Artificial Intelligence*

3.2 Well-posed and ill-posed sentences

We face problems when the sentences that should convey information and data are not clear. Suppose that an electric-car model *consumes 150 Wh/km* and *has a range of 200 km*; a second car model consumes 250 Wh/km and has a range of 600 km. Someone says "I think the second model is better; what do you think?". It isn't clear how we should answer; what does "better" mean? If it refers to consumption, then the first car model is "better". If it refers to range, then the second model is "better". If it refers to a combination

of these two characteristics, or to something else, then we simply can't answer. Here we have a problem with querying and giving data, because the sentence underlying such query is not clear.

We say that such sentences are **not well-posed**, or that they are **ill-posed**.

This may seem an obvious discussion to you. Yet you'd be surprised by how often unclear sentences appear in scientific papers about data engineering! Not seldom we find discussions and disagreements that actually come from unclear underlying sentences, that two parties interpret in different ways.

As a data engineer, you'll often have the upper hand if you are on the lookout for ill-posed sentences. Whenever you face an important question, or you're given an important piece of information, or you must provide an important piece of information, *always take a little time to examine whether the question or information is actually well-posed*.



- *[TODO] Exercise: give actual paper to analyse*

Reading list

4 Inference

4.1 What is inference?

The first core problem in all data-driven engineering applications – and in daily life too – is to *draw inferences*, that is, acquire information. We may wish to acquire information out of simple curiosity, or for some specific engineering reason or goal, as we'll discuss later. Examples:

1. We'd like to know whether it'll rain today, so we can decide whether to get an umbrella or rain clothes.
2. A clinician would like to know which disease affects a patient, so as to decide for the optimal treatment.
3. The X-player of this game of Xs & Os:  needs to know where put the next **X** in order to win.
4. The computer of a self-driving car needs to know whether a particular patch of colours in the visual field is a person, so as to slow down the car and stop.
5. In order to launch a rocket to the Moon, a rocket engineer needs to know, within two significant digits, [how much is the velocity](#) $\sqrt{2GM/r}$, where $G = 6.67 \cdot 10^{-11} \text{ m}^3 \text{ s}^{-2} \text{ kg}^{-1}$, and $M = 5.97 \cdot 10^{24} \text{ kg}$ and $r = 6.37 \cdot 10^6 \text{ m}$ are the mass and radius of the Earth.
6. We'd like to know whether the rolled die will show , so we can win a bet.
7. An [aircraft's autopilot system](#) needs to predict how much the [aircraft's roll](#) will change by increasing the right wing's [angle of attack](#) by 0.1 rad.
8. An archaeologist would like to know whether the fossil bone just dug out belonged to a Tyrannosaurus rex.

9. An automated system in an assembly line needs to predict whether an electric component of a widget will fail within the next two years.

Note how each of these inferences boils down to determining whether some sentences are true or false. In example 1. we want to know whether the sentence 'It rains today' is true or not. In example 2. the clinician wants to know which of the sentences 'The patient has pneumonia', 'The patient has asthma', 'The patient has bronchitis', and so on, are true (several can be true at the same time). In example 5. the rocket engineer wants to know which among the sentences 'The velocity is 0.010 m/s', 'The velocity is 0.011 m/s', ..., 'The velocity is 130 m/s', and so on, is true. The sentences that underlie an inference can be extremely many and complex, and yet we must have an idea of what they are (otherwise, do we really know what our inference is about?).

Exercise

Try to identify which sentences underlie the other example inferences above.

4.2 Certain and uncertain inference

The example inferences above present very different levels of difficulty.

Inferences 3. and 5. are special because they can actually be drawn *exactly*, that is, we really find out which of their underlying sentences are true and false. In example 3. it is trivial that putting the next **X** in the mid-right slot makes the X-player win. In example 5. a couple of mathematical operations show that the sentence 'The velocity is 11 km/s' is true. When we can obtain the data we want from the data we have by using "only"¹ logic and mathematical operations, our inference is *certain*, also called a "deduction"; in these notes we shall call it a *truth inference*. But every deduction can be basically drawn by repeatedly applying the rules of logic.

¹"Only" in quotation marks because the logical analysis and operations leading to the answer can still be computationally very expensive.

The other example inferences cannot be drawn exactly, in the sense that we cannot know for sure whether all their underlying sentences are true or false. But this doesn't mean that we cannot say anything whatsoever. In example 6. we consider the sentence 'The die shows six pips' to be more likely false than true. In example 2. the clinician might be quite sure about the disease, after observing the symptoms. On the other hand, in example 1. we might really have no clue whether 'It rains today' will turn out to be true or false. These inferences are *uncertain*. Certain inferences can be considered as a limit case of uncertain ones, in which the uncertainty vanishes or is extremely small.

To draw certain inferences, we follow the rules of Logic. What rules do we follow to draw uncertain inferences?

5 Truth inference

5.1 Building blocks

Consider the following trivial but certain inference:

$$\frac{\text{'My umbrella is either blue or red'} \quad \text{'My umbrella is not red'}}{\text{'My umbrella is blue'}}$$

Above the line we write the sentences representing the data we have. Below the line we infer the information that supposedly interests us.

How could we draw this obvious inference? Which rules did we follow?

Logic is a huge field that formalizes and makes rigorous the rules that a rational person or an artificial intelligence should use in drawing certain inferences. We'll get a glimpse of it here, as a trampoline for jumping towards our data-driven engineering problems.

5.1.1 Basic sentences

We start by writing down the *basic*¹ sentences that constitute our data and that underlie the inferences we want to draw. “Basic” in the sense that we will not analyse these sentences into further sub-sentences. In the trivial example above we identify two such sentences: ‘My umbrella is blue’, and ‘My umbrella is pink’. Let's represent them by symbols:

$$\begin{aligned} b &:= \text{'My umbrella is blue'} \\ r &:= \text{'My umbrella is red'} \end{aligned}$$

¹A more technical term is “atomic”

⚠ Note a subtlety in our data – and again why we need to make their underlying sentences as clear as possible: it is understood here that my umbrella is all of one colour.

5.1.2 Connectives

You notice that we didn't consider 'My umbrella is either blue or red' and 'My umbrella is not red' as basic sentences. These sentences can indeed be expressed in terms of the basic sentences b and r . We consider one way or operation to change a sentence into another related to it, and two ways or operations to combine two or more sentences together. These operations are called "connectives". Our natural language offer many more operations to combine sentences, but these three turn out to be all we need in virtually all engineering problems. The three connectives are:

Not: \neg for example,

$$\neg r = \text{'My umbrella is not red'}$$

And: \wedge for example,

$$b \wedge r = \text{'My umbrella is blue, and it is red'}$$

Or: \vee for example,

$$b \vee r = \text{'My umbrella is blue, or red, or both'}$$

⚠ Note some subtleties of the connectives:

- "Not" doesn't mean some kind of complementary quality, but only the negation. For instance, \neg 'The chair is black' does not mean 'The chair is white'.
- $b \vee r$ does not exclude, a priori, that my umbrella cannot be both blue and black (there is a connective for that: "exclusive-or", but it can be constructed out of the three we already have.)
- It is important to distinguish between *logical* im-

possibility and *physical* impossibility. It is physically impossible that an umbrella be fully white and fully black, but formal logic does allow us to consider the sentence “the umbrella is fully white and fully black”, without setting it true or false a priori. We express its physical impossibility by assigning it the value **false** in any inference we want to make. This generality of formal logic is a feature: it allows us to entertain and study hypotheses when we still don’t know whether they are physically impossible. Scientific research would be impossible without this feature.

From this last remark we see that the sentence ‘My umbrella is either blue or red’ does not correspond to $b \vee r$. The sentence also means implicitly that my umbrella cannot be both blue or red. We could rewrite it as ‘My umbrella is either blue or red, and it is not both blue and red’. Convince yourself that in symbols we can write it like this:

$$(b \vee r) \wedge \neg(b \wedge r) = \text{‘My umbrella is either blue or red’}$$

5.1.3 Data or axioms

Now we have the sentences to represent our data, and even symbols to represent it in a compact way. But what do our data actually say? They say that the sentences ‘My umbrella is either blue or red’ and ‘My umbrella is not red’ are true. Here’s how we express this in symbols.

We represent our data by the symbol D and use the notation²

$$\begin{aligned} &\neg r \mid D \\ &(b \vee r) \wedge \neg(b \wedge r) \mid D \end{aligned}$$

to mean that ‘My umbrella is not red’ and ‘My umbrella is either blue or red’ are **true** according to our data.

With this notation we can also augment our data with additional assumptions or hypotheses, even if just temporarily. For example,

²Current notation in logic writes $D \models \neg r$. We use a different notation for an easier transition to probability logic.

$$\neg r \mid b \wedge D$$

means that ‘My umbrella is not red’ is **true** according to data D *together with* the additional assumption that ‘My umbrella is blue’ is **true**.

5.2 Truth-inference rules

Deduction systems in formal logic give us a set of rules for making correct inferences. These rules can be represented in a wide variety of ways. For instance as lines: above, we write what the data say; below, what inference we can draw. An example is this:

$$\frac{a \wedge b \mid D}{a \mid D} \quad (5.1)$$

In total there are a dozen or so rules of this kind.

But we can compactly encode all these rules in the following way. First, represent **true** with the number 1, and **false** with 0. Second, express the fact that a sentence a – which can be made of subsentences combined by connectives – is **true** according to data D by writing

$$T(a \mid D) = 1$$

and that it is **false** according to D by writing

$$T(a \mid D) = 0$$

Then the rules of truth inference are summarized by the following equations, which must always hold:

Let’s see how the inference rule (5.1), for example, is encoded in these equations. The rule starts with saying that $a \wedge b$ is **true** according to D . This means that $T(a \wedge b \mid D) = 1$. But, by rule (5.3), we must then have $T(b \mid a \wedge D) \cdot T(a \mid D) = 1$.

Rule for “not”:

$$T(\neg a \mid D) + T(a \mid D) = 1 \quad (5.2)$$

Rule for “and”:

$$T(a \wedge b \mid D) = T(a \mid b \wedge D) \cdot T(b \mid D) = T(b \mid a \wedge D) \cdot T(a \mid D) \quad (5.3)$$

Rule for “or”:

$$T(a \vee b \mid D) = T(a \mid D) + T(b \mid D) - T(a \wedge b \mid D) \quad (5.4)$$

Rule of self-consistency:

$$T(a \mid a \wedge D) = 1 \quad (5.5)$$

This can only happen if both $T(b \mid a \wedge D)$ and $T(a \mid D)$ are equal to 1. So we can conclude that $T(a \mid D) = 1$, which is exactly the conclusion under the line in rule (5.1).

Exercise

Try to prove our initial inference

$$\frac{(b \vee r) \wedge \neg(b \wedge r) \mid D \quad \neg r \mid D}{b \mid D}$$

using the basic rules (5.2, 5.3, 5.4, 5.5). Remember that you can use each rule as many times as you like, and that there is not only one way of constructing a proof.

5.3 Logical AI agents and their limitations

The basic rules above are also the rules that a logical artificial-intelligent agent should follow.

Reading

[Ch. 7 in *Artificial Intelligence*](#)

Many – if not most – inference problems that a data engineer must face are, however, of the *uncertain* kind: it is not possible to surely infer the truth of some data, and the truth of some initial data may not be known either. In the next chapter we shall see how to generalize the logic rules to uncertain situations.

🔑 For the extra curious

Our cursory visit of formal logic only showed a microscopic part of this vast field. The study of logic rules continues still today, with many exciting developments and applications. Feel free take a look at *Logic in Computer Science*, *Mathematical Logic for Computer Science*, *Natural Deduction Systems in Logic*

6 Probability inference

6.1 When truth isn't known: probability

In most real-life and engineering situations we don't know the truth or falsity of sentences and hypotheses that interest us. But this doesn't mean that nothing can be said or done in such situations.

When we cross a busy city street we look left and right to check whether any cars are approaching. We typically don't look up to check whether something is falling from the sky. Yet, couldn't it be **false** that cars are approaching? and couldn't it be **true** that *some object is falling from the sky*? Of course both events are possible. Then why do we look left and right, but not up?

The main reason¹ is that we *believe strongly* that cars might be approaching, *believe very weakly* that some object might be falling from the sky. In other words, we consider the first occurrence to be very *probable*; the second, extremely improbable.

We shall take the notion of **probability** as intuitively understood (just as we did with the notion of truth). Terms equivalent for “probability” are *degree of belief*, *plausibility*, *credibility*.

❗ In technical discourse, *likelihood* means something different and is *not* a synonym of “probability”, as we'll explain later.

Probabilities are quantified between 0 and 1, or equivalently between 0% and 100%. Assigning to a sentence a probability 1 is the same as saying that it is **true**; and a probability

¹We shall see later that one more factor enters the explanation.

0, that it is **false**. A probability of 0.5 represents a belief completely symmetric with respect to truth and falsity.

It is important to emphasize and agree on some facts about probabilities:

- **Probabilities are assigned to *sentences*.** Consider an engineer working on a problem of electric-power distribution in a specific geographical region. At a given moment the engineer may believe with 75% probability that the measured average power output in the next hour will be 100 MW. The 75% probability is assigned not to the quantity “100 MW”, but to the *sentence*

‘The measured average power output in the next hour will be 100 MW’

This difference is extremely important. Consider the alternative sentence

‘The average power output in the next hour will be *set* to 100 MW’

the quantity is the same, but the meaning is very different. The probability can therefore be very different (if the engineer is the person deciding the output, the probability is 100%). The probability depends not only on a number, but on what it’s being done with that number – measuring, setting, third-party reporting, and so on. Often we still write simply ‘ $O = 100\text{ W}$ ’ or even just ‘100 W’, provided that the full sentence behind the shorthand is understood.

- **Probabilities are agent- and context-dependent.** A coin is tossed, comes down heads, and is quickly hidden from view. Alice sees that it landed heads-up. Bob instead doesn’t manage to see the outcome and has no clue. Alice considers the sentence ‘Coin came down heads’ to be **true**, that is, to have 100% probability. Bob considers the same sentence to have 50% probability.

Note how Alice and Bob assign two different probabilities to the same sentence; yet both assignments are completely rational. If Bob assigned 100% to ‘heads’, we would suspect that he had seen the outcome after all; if he assigned 0% to ‘heads’, we would consider that groundless and silly. We would be baffled if Alice assigned 50% to ‘heads’, because she saw the outcome was

actually heads; we would hypothesize that she feels unsure about what she saw.

An omniscient agent would know the truth or falsity of every sentence, and assign only probabilities 0 or 1. Some authors speak of “*actual* (but unknown) probabilities”; if there were “actual” probabilities, they would be all 0 or 1, and it would be pointless to speak about probabilities at all – every inference would be a truth inference.

- **Probabilities are not frequencies.** The fraction of defective mechanical components to total components produced per year in some factory is a quantity that can be physically measured and would be agreed upon by every agent. It is a *frequency*, not a degree of belief or probability. It is important to understand the difference between them, to avoid making sub-optimal decisions; we shall say more about this difference later. Frequencies can be unknown to some agents, probabilities cannot be unknown (but can be difficult to calculate). Be careful when you read authors speaking of an “unknown probability”; either they actually mean “unknown frequency”, or a probability that has to be calculated (it’s “unknown” in the same sense that the value of $1 - 0.7 \cdot 0.2 / (1 - 0.3)$ is unknown to you right now).
- **Probabilities are not physical properties.** Whether a tossed coin lands heads up or tails up is fully determined by the initial conditions (position, orientation, momentum, rotational momentum) of the toss and the boundary conditions (air velocity and pressure) during the flight. The same is true for all macroscopic engineering phenomena (even quantum phenomena have never been proved to be non-deterministic, and there are [deterministic and experimentally consistent](#) mathematical representations of quantum theory). So we cannot measure a probability using some physical apparatus; and the mechanisms underlying any engineering problem boil down to physical laws, not to probabilities.

Reading

Dynamical Bias in the Coin Toss

These facts are not just a matter of principle. They have important practical consequences. A data engineer who is not attentive about the source of the data (measured? set? reported, and so maybe less trustworthy?), or who does not carefully assess the context of a probability, or who mixes it up with something else, or who does not take advantage (when possible) of the physics involved in the engineering problem, will design a system with sub-optimal performance² – or even cause deaths.

6.2 No new building blocks

In discussing [truth-inference](#) we introduced notations such as $T(a \mid b \wedge D)$, which stands for the truth-value 0 or 1 of sentence a in the context of data D and supposing (even if only hypothetically) sentence b to be true. We can simply extend this notation to probability-values, using a P instead of T :

$$P(a \mid b \wedge D) \in [0, 1]$$

represents the probability or degree of belief in sentence a in the context of data D and supposing also sentence b to be true. Keep in mind that both a and b could be complex sentences (for instance $a = (\neg c \vee d) \wedge e$). Note that truth-values are included as the special cases 1 or 0:

$$P(a \mid b \wedge D) = 0 \text{ or } 1 \iff T(a \mid b \wedge D) = 0 \text{ or } 1$$

6.3 Probability-inference rules

Extending our truth-inference notation to probability-inference notation has been straightforward. But how do we draw inferences when probabilities are involved?

Consider the inference about my umbrella in a more uncertain situation:

²This fact can be mathematically proven.

$$\frac{P(\text{'My umbrella is either blue or red'} \mid D) = 1 \quad P(\text{'My umbrella is not red'} \mid D) = 0.5}{P(\text{'My umbrella is blue'} \mid D) = ?}$$

or more compactly, using the symbols we introduced earlier,

$$\frac{P[(b \vee r) \wedge \neg(b \wedge r) \mid D] = 1 \quad P(\neg r \mid D) = 0.5}{P(b \mid D) = ?}$$

This says, above the line, that: according to our data D my umbrella is either blue or red (and can't be both), with full certainty; and according to our data we have no preferential beliefs on whether my umbrella is not red. What should then be the probability of my umbrella being blue, according to our data?

Intuitively that probability should be 50%: $P(b \mid D) = 0.5$. But which rules did we follow in arriving at this probability? More generally, which rules should we follow in assigning new probabilities from given ones?

The amazing result is that *the rules for truth-inference, formulae (5.2, 5.4, 5.3, 5.5), extend also to probability-inference*. The only difference is that they now hold for all values in the range $[0, 1]$, rather than only values 0 and 1.

This important result was taken more or less for granted at least since Laplace in the 1700s. But was formally proven for the first time in the 1940s by R. T. Cox; the proof has been refined since then. What kind of proof is it? It shows that if we don't follow the rules we arrive at illogical conclusions; we'll show some examples later.

So here are the fundamental rules of probability inference; in these rules, all probabilities can have values in the range $P() \in [0, 1]$, and the symbols a, b, D represent sentences of any complexity:

It is amazing that **ALL** inference is just a repeated application of these four rules – billions of times or more in some inferences. All machine-learning algorithms are just applications or approximations of these rules. Methods that you may have heard about in statistics are just specific applications of these rules. Truth inferences are also special applications of these rules. Most of this course is, at bottom, just a study of how to apply these rules in particular kinds of problems.

**“Not” \neg rule**

$$P(\neg a \mid D) + P(a \mid D) = 1$$

“And” \wedge rule

$$P(a \wedge b \mid D) = P(a \mid b \wedge D) \cdot P(b \mid D) = P(b \mid a \wedge D) \cdot P(a \mid D)$$

“Or” \vee rule

$$P(a \vee b \mid D) = P(a \mid D) + P(b \mid D) - P(a \wedge b \mid D)$$

Self-consistency rule

$$P(a \mid a \wedge D) = 1$$



Reading

- *Probability, Frequency and Reasonable Expectation*
- Ch. 2 of *Bayesian Logical Data Analysis for the Physical Sciences*
- §§ 1.0–1.2 of *Data Analysis*
- Feel free to skim through §§ 2.0–2.4 of *Probability Theory*

6.4 How the inference rules are used

The fundamental rules represent, first of all, constraints of logical consistency among probabilities. If we have probabilities $P(a \mid D) = 0.7$, $P(b \mid a \wedge D) = 0.1$, $P(a \wedge b \mid D) = 0.2$, then there’s an inconsistency somewhere, because these values violate the and-rule: $0.2 \neq 0.1 \cdot 0.7$. In this case we must find

the inconsistency and solve it. The rules also imply more general constraints. For example it is easy to see that we must *always* have

$$\begin{aligned} P(a \wedge b \mid D) &\leq \min\{P(a \mid D), P(b \mid D)\} \\ P(a \vee b \mid D) &\geq \max\{P(a \mid D), P(b \mid D)\} \end{aligned}$$

The rules also allow us to calculate new probabilities from given ones; the calculated probabilities will be automatically consistent. This is the main use of the rules in data-driven engineering problems. For each equation shown in the rules we can calculate one probability given the remaining ones in the equation, with some special cases when values of 0 or 1 appear.

For example, if we have $P(a \wedge b \mid D) = 0.2$ and $P(a \mid D) = 0.7$, from the and-rule we find $P(b \mid a \wedge D)$:

$$P(b \mid a \wedge D) = \frac{P(a \wedge b \mid D)}{P(a \mid D)} = \frac{0.2}{0.7} \approx 0.2857$$

Since probabilities are quantified by real numbers, it's possible and acceptable to have slight discrepancies owing to numerical round-off errors.

@@ umbrella example

6.4.1 Derived rules

The rules above are in principle all we need to use. But from them it is possible to derive some additional shortcut rules that are automatically consistent with the fundamental ones.

First, it is possible to show that all rules you may know from Boolean algebra are a consequence of the fundamental rules. For example, we can always make the following convenient

replacements anywhere in a probability expression:

$$\begin{aligned}A \wedge A &= A \vee A = A & \neg\neg A &= A \\A \wedge B &= B \wedge A & A \vee B &= B \vee A \\ \neg(A \wedge B) &= \neg A \vee \neg B & \neg(A \vee B) &= \neg A \wedge \neg B \\A \wedge (B \vee C) &= (A \wedge B) \vee (A \wedge C) \\A \vee (B \wedge C) &= (A \vee B) \wedge (A \vee C)\end{aligned}$$

Two other derived rules are used extremely often, so we treat them separately.

6.5 Law of total probability or “extension of discourse”

6.5.1 Bayes’s theorem

No premises? No conclusions!

@@ consequences of not following the rules, §12.2.3 of AI

- *Exercise: Monty-Hall problem & variations*
- *Exercise: clinical test & diagnosis*

6.6 Common points of certain and uncertain inference

7 Data and information

7.1 Kinds of data

7.1.1 Binary

7.1.2 Nominal

7.1.3 Ordinal

7.1.4 Continuous

- unbounded
- bounded
- censored

7.1.5 Complex data

2D, 3D, images, graphs, etc.

7.1.6 “Soft” data

- orders of magnitude
- physical bounds

7.2 Data transformations

- log
- probit
- logit

8 Allocation of uncertainty among possible data values: probability distributions

8.1 The difference between Statistics and Probability Theory

Statistics is the study of collective properties of collections of data. It does not imply that there is any uncertainty.

Probability theory is the quantification and propagation of uncertainty. It does not imply that we have collections of data.

8.2 What's “distributed”?

Difference between distribution of probability and distribution of (a collection of) data.

8.3 Distributions of probability

8.3.1 Representations

- Density function
- Histogram
- Scatter plot

Behaviour of representations under transformations of data.

8.4 Summaries of distributions of probability

8.4.1 Location

Median, mean

8.4.2 Dispersion or range

Quantiles & quartiles, interquartile range, median absolute deviation, standard deviation, half-range

8.4.3 Resolution

Differential entropy

8.4.4 Behaviour of summaries under transformations of data and errors in data

8.5 Outliers and out-of-population data

(Warnings against tail-cutting and similar nonsense-practices)

8.6 Marginal and conditional distributions of probability

8.7 Collecting and sampling data

8.7.1 “Representative” samples

Size of minimal representative sample = $(2^{\text{entropy}})/\text{precision}$

- *Exercise: data with 14 binary variates, 10000 samples*

8.7.2 Unavoidable sampling biases

In high dimensions, all datasets are outliers.

Data splits and cross-validation cannot correct sampling biases

8.8 Quirks and warnings about high-dimensional data

9 Making decisions

9.1 Decisions, possible situations, and consequences

9.2 Gains and losses: utilities

9.2.1 Factors that enter utility quantification

Utilities can rarely be assigned a priori.

9.3 Making decisions under uncertainty: maximization of expected utility

10 The most general inference problem