

Lecture Notes for Machine Learning and Data Science Courses

THIS WEBSITE IS CURRENTLY NOT UPDATED. Check
<https://faculty.washington.edu/otoomet/machineLearning.pdf> for the
most recent version

Ott Toomet

June 15, 2021

Preface

This is a collection of notes made for INFO370, INFO371, IMT573 and IMT574 courses, taught at the Information School, University of Washington. It is largely a collection of such topics where I haven't found an alternative material at suitable level and suitable coverage. However, later I have also added some material where good coverage exists. Check also out the [python companion](#) (very much preliminary) and [R companion](#) (even more preliminary).

The source of the notes is available at it's [bitbucket repo](#), feel free to leave feedback in it's issue tracker.

Contents

0.1 Notation	vii
1 Introduction to Statistics	1
1.1 Different Kind of Values	2
1.1.1 Measures: Possible Mathematical Operations	2
1.1.2 Values: Which Values Are the Possible	5
1.2 Doing Descriptive Statistics	5
1.2.1 Data Quality	5
1.2.2 Describing Data	6
1.2.3 Sampling	11
1.2.4 Misusing Statistics: Lies, Damned Lies, and Statistics . .	14
1.3 Probability	18
1.3.1 Events and Sample Space	18
1.3.2 Conditional Probability and Bayes Theorem	19
1.3.3 Random Variable	22
1.3.4 Expected Value and Variance	24
1.3.5 Distributions	28
1.3.6 Information and Entropy	31
1.4 Statistical Inference	34
1.4.1 Statistical Hypotheses and Hypothesis Testing	34
1.4.2 Doing Statistical Inference	38
1.4.3 Comparing Distributions	43
2 Linear Algebra	49
2.1 Why Linear Algebra in Machine Learning	49
2.2 Vectors and Vector Spaces	50
2.2.1 Vectors	50
2.2.2 Norm and Distance	57
2.3 Matrices	60
2.3.1 What are matrices	60
2.3.2 Matrix operations	64
2.3.3 Inverse Matrix	70
2.3.4 Eigenvalues	73
2.3.5 Applications: Images in Matrix Form	74

3 General Machine Learning Strategies	83
3.1 Numeric Data	83
3.2 Metric Distance: A Revisit	84
3.2.1 Data-Driven Metrics	84
3.2.2 Cosine similarity and angular distance	91
4 Regression Models	95
4.1 Linear Regression	95
4.1.1 The Problem: Why We Want Linear Regression	95
4.1.2 Simple Regression	96
4.1.3 Model evaluation: MSE, RMSE, R^2	107
4.1.4 Multiple Regression	110
4.1.5 Categorical Variables	113
4.1.6 Interactions Effects	115
4.1.7 Feature Transformation	120
4.1.8 Assumptions in OLS Models	124
4.1.9 Linear Regression: The matrix approach	124
4.1.10 Solving the Linear Regression Model	127
4.2 Logistic Regression	130
4.2.1 What Is Logistic Regression And What Is It Good For? .	130
5 Causality	135
5.1 What is cause?	136
5.1.1 Sufficiency and Necessity	136
5.1.2 Measuring the Amount of Cause and Effect	137
5.2 Causality with data: three explanations	137
5.3 Strategies for Causal Inference	141
5.3.1 We Know Which Model is Feasible	141
5.3.2 Randomized Controlled Trials	142
5.3.3 Natural Experiments	144
5.3.4 Case-Control Study	145
5.3.5 Controlling for Confounding Factors	146
5.3.6 Explicit Modeling of Selection Process	146
5.4 Causal inference in linear regression framework	146
5.4.1 Counterfactual and Identifying Assumption	147
5.4.2 A Few Popular Estimators	149
5.5 Cognitive Illusions in Causal Inference	162
5.6 Causality and complex social problems	164
5.6.1 Effect of bike helmet laws	164
6 Assessing model goodness	167
6.1 Categorization	167
6.1.1 Confusion matrix and related concepts	167
6.2 Overfitting and Validation	176
6.2.1 What is overfitting	176

7 Machine Learning Models	181
7.1 <i>k</i> -Nearest Neighbors	181
7.1.1 Introductory Example	181
7.1.2 What is Distance	182
7.2 Trees and tree-based methods	184
7.2.1 Decision Trees	184
7.2.2 Ensemble Methods	192
7.3 Naïve Bayes	195
7.3.1 Before We Begin: A Single Word-Based Bayesian Classifier	195
7.3.2 Naive Bayes Classifier	198
7.4 Neural Networks	209
7.4.1 Feed-Forward Networks	209
7.4.2 Convolutional Neural Networks	216
8 Different Types of Data	221
8.1 Text as Data	221
8.1.1 Preprocessing: Tokenization, Stemming, and Lemmatization	221
8.1.2 <i>n</i> -grams	223
8.1.3 Bag of Words and Document-Term-Matrix	223
8.1.4 TF-IDF	225
9 Machine Learning Techniques	229
9.1 Loss Function and Non-Linear Optimization	229
9.1.1 Loss Function	229
9.1.2 Maximum Likelihood (ML)	231
9.2 Gradient Ascent	234
9.3 OLS Example	238
9.4 Gradient Descent	240
9.4.1 Gradient	241
9.4.2 Optimization	242
9.4.3 Problems	243
9.5 Key Concepts	244
9.5.1 Resources	244
9.6 Feature Selection and Regularization	245
9.6.1 Feature Selection	245
9.6.2 Regularization	247
10 Unsupervised Learning	249
10.1 Introduction	249
10.2 Cluster Analysis	250
10.2.1 Idea	250
10.2.2 Cluster Analysis More Generally	251
10.2.3 <i>k</i> -Means Clustering	252
10.3 Principal Component Analysis	255
10.3.1 Motivation	255

10.3.2 Principal Component Regression	261
10.4 Comparison of Clustering and PCA	265
11 Applications	269
11.1 Recommender Systems	269
11.1.1 Collaborative Filtering	269
12 Responsible Data Science	273
12.1 Explainable AI	273
12.2 Ethical Questions	274
12.2.1 Who Are Represented in Big Data?	274
12.2.2 Big Data, Big Inequality?	275
12.2.3 Fairness: Different Measures are Incompatible	275
12.3 Human Versus Algorithmic Decision-Making	277
A Mathematics	279
A.1 High-School Mathematics	279
A.1.1 Logarithm	279
A.1.2 Differentiation	280
A.2 Matrix calculus	280
A.2.1 Gradient	283
B Exercise Solutions	289
References	299
Index	301

0.1 Notation

These notes contain a lot of mathematical notation. Here is a list conventions and more common notation:

Greek alphabet Mathematical notation uses Greek alphabet extensively. Table 1 shows a complete list of it, both in upper and lower case form. Note that several upper case letters are identical with the corresponding Latin letters, and there are two ways to write certain lower case letters.

Table 1: Greek alphabet

Letter	Lower case	Upper case
Alpha	α	A
Beta	β	B
Gamma	γ	Γ
Delta	δ	Δ
Epsilon	ϵ, ε	E
Zeta	ζ	Z
Eta	η	E
Theta	θ, ϑ	Θ
Iota	ι	I
Kappa	κ	K
Lambda	λ	Λ
Mu	μ	M
Nu	ν	N
Omicron	o	O
Pi	π	Π
Rho	ρ, ϱ	R
Sigma	σ	Σ
Tau	τ	T
Upsilon	υ	Y
Phi	ϕ, φ	Φ
Chi	χ	X
Psi	ψ	Ψ
Omega	ω	Ω

Numbers We generally follow the following notation:

- number of observations (cases) is denoted by N .
- number of variables in a model is denoted by K .
- predicted or estimated values are denoted by “hat” $\hat{\cdot}$, such as \hat{y} for predicted y and $\hat{\beta}$ for estimated value of β .

Sets

- general sets are denoted by calligraphic letters like \mathcal{S} , \mathcal{A} , \mathcal{Q} .
- number of elements in a set is denoted with the same symbol as norm, $\|\mathcal{S}\|$.
- Set of integers is denoted by \mathbb{Z} , set of pairs of integers is \mathbb{Z}^2 , and so on.
- Set of real numbers is denoted by \mathbb{R} , pairs of real numbers are \mathbb{R}^2 , etc.
- Intervals are denoted by $[a,b]$ for a closed interval from a to b , (a,b) for an open interval, and $[a,b)$ and $(b,a]$ for semi-open intervals.

Scalars, vectors, matrices

- scalar values (just numbers) are denoted with ordinary latin and Greek letters, such as a and v . Upper case letters denote integer constants (such as number of observations), lower case letters are both continuous values and integer indices. For instance, in case of x_i , x may be a continuous variable while i is an integer index.
- vectors are denoted in bold lower-case letters, for instance \mathbf{x} and $\boldsymbol{\epsilon}$.
- Matrices are written in sans-serif capital letters. For instance, \mathbf{A} and \mathbf{I} are matrices.
- unit matrix (identity matrix) is denoted by \mathbf{I} , or \mathbf{I}_n in case we want to stress it is $n \times n$ identity matrix.
- vectors are just $n \times 1$ or $1 \times n$ matrices. We denote by \mathbf{x} an $n \times 1$ column vector and \mathbf{x}^\top an $1 \times n$ row vector. When defining a column vector, we often use notation like $(1 \ 2 \ 3)^\top$, a row vector transposed, do denote the column vector $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$.
- We use dot, \cdot , to denote multiplication where it helps to understand the formulas. This applies to both scalar and matrix multiplication. So

$$\lambda \mathbf{x}^\top \mathbf{y} \quad \lambda \mathbf{x}^\top \cdot \mathbf{y} \quad \lambda \cdot \mathbf{x}^\top \cdot \mathbf{y} \quad (0.1.1)$$

are all equivalent and denote a product of three factors.

- We use \odot to denote elementwise product of matrices and vectors. For instance,

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \odot \begin{pmatrix} 10 & 20 \\ 30 & 40 \end{pmatrix} = \begin{pmatrix} 10 & 40 \\ 90 & 160 \end{pmatrix} \quad (0.1.2)$$

- we use large dot, \bullet , to denote “all indices at this dimension”. For instance $\mathbf{A}_{\bullet 2}$ means second column of matrix \mathbf{A} while $\mathbf{A}_{2\bullet}$ is its second row.

Functions We use notation $f : A \rightarrow B$ to denote a function that maps every element from set A to an element of set B . For instance, $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a function that maps elements from \mathbb{R}^n to \mathbb{R}^m , i.e. assigns a m -dimensional real vector to every n -dimensional real vector. $g : \mathbb{R} \rightarrow \mathbb{R}$ is the “traditional” function that computes a new real number from every other real number.

We use the following special functions in the text:

- indicator function $\mathbb{1}(A)$:

$$\mathbb{1}(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise.} \end{cases} \quad (0.1.3)$$

For instance, $\mathbb{1}(x > 0)$ is 1 if x is positive, and zero otherwise.

Acronyms Here is a list of common acronyms used in the text:

- CI : confidence interval
- CLT : Central Limit Theorem
- $i.i.d$: independently identically distributed
- RV : random variable

Chapter 1

Introduction to Statistics

Descriptive statistics is widely used to explore and summarize data. This includes computing means and variances, comparing certain obvious groups in data, analyzing data quality, and creating plots and tables. These methods are somewhat distinct from a formal and precise mathematical theory, *mathematical statistics*. Both branches of statistics are very important in data science. Much of “know your data”, and data visualization and presentation can be counted as descriptive statistics; and machine learning is largely based on formal statistical models. Here we talk about descriptive statistics and consider certain concepts of mathematical statistics below in Sections 1.3 and 1.4.

There are broadly three reasons we use statistics in data science and machine learning:

- Descriptive statistics a good way to summarize data. For instance, GDP per capita (2017, nominal) in Madagascar is \$450 and in Canada it is \$44,841 ([world-o-meter data](#)). Despite all problems with reducing the complexity of an economy into a single index, in practice it is a very good proxy to describe life quality in various aspects in these places.
- Data is imprecise and we describe errors as random variables. This may include missings, measurement errors, computation errors, validity and reliability issues. This is one of the major motivations to use mathematical statistics for data analysis.
- The world is complex and unpredictable, and we model uncertain factors as random variables. This is the other reason mathematical statistics has done such strong inroads into econometrics and machine learning. It is also related to the previous problem, that of incomplete data—if we had better data, we would be able to predict the world better. But we have to live in this world using the data we can have access to.

1.1 Different Kind of Values

Data contains values of different types. Here we discuss two potential ways to categorize those, first according to what kind of mathematical operations (comparison, addition, ...) the particular data type permits, and second according to the possible values data can take.

TBD: mode, median, mean

1.1.1 Measures: Possible Mathematical Operations

The values are often categorized to according to their *measure*, namely *nominal* (no comparison possible), *ordinal* (comparison possible, but cannot compute difference), *interval* (can compute difference but not ratio), and *ratio* (can compute ratio too). Below we discuss these (and a few other) measure in a more detail.

Before nominal Nominal measures are usually exemplified with unique labels. However, there are important classes of data where such labeling does not make much sense. This includes text and images. Labeling all texts or images uniquely means not to label category but the text or image itself, so a single different letter or a single different pixel will result in a different label. While we can do this, such labeling is usually not helpful. Unless we are looking for very short and a kind of standard texts (say, a review text is “wonderful”), almost all texts and images are unique and labeling will not help us to do any useful analysis. We have to use different tools for, e.g. categorizing images or deducing the sentiment of the texts.

Nominal measures In many cases we have a limited number of different categories (and we can always lump too small ones together into “other”). Such categories often do not follow any inherent order, and hence are not comparable (in a sense as smaller/larger, better/worse, ...). Examples include gender, college attended and membership of political parties. In such cases there are limited number of mathematical operations possible. These include

- Comparison: we can tell if two cases are equal
- Mode: we can tell which category is most common.

But usually we cannot tell which case “preceeds” another in any meaningful sense.

Ordinal measures Another large set of values are categorical with an inherent order, it is always possible to tell if one case is “larger” or “smaller” of another case (or equal to it). Examples include various opinion questions like “do you support the president” with the answers ranging from 1 (not at all) to 5 (very much support); continuous values measured in brackets like income categories (0-10k, 10k-30k, 30k-60k, ...).

One can use ordinal measures for all operations as nominal measures, but now we can also compare the cases: which case is “smaller” or “larger”. This, in turn, allows us to order the observations, and compute the median (the middle value), and other sample quantiles.

However, the difference of such values carries little meaning. Sometimes the categories carry numeric label (like the opinion about president’s performance) but the difference between these numbers may be hard to interpret. There is no guarantee that the difference in the feeling about the president between strong and weak opponents (values 0 and 1) is similar as between weak and strong supporters (values 4 and 5).¹

Interval measures These are numeric values that are comparable like ordinal measures, but where also the differences are meaningful. The examples include temperature (in both degrees of C or F) and GPA.

In case of temperature we can, for instance, say that 2019 global temperature was 0.98C above the temperature of 1951–1980 base period, and that of 2001 was 0.54C above the same baseline. Even more, these two figures, 0.98 and 0.54 are directly comparable, so we can say that the temperature anomaly in 2020 was 1.81 times larger.² However, the temperature values in this sense are not comparable in the same way. The baseline temperature over this period was approximately 14C, and hence the corresponding values were 14.54 and 14.98C. Now it carries little meaning to say that the temperature in 2019 was 1.03 times larger than that of 2001. Degrees of Celsius are based on the temperature of the freezing point of water, and from the climate perspective, it is an arbitrary reference point.

Interval measures allow all the operations as ordinal measures, and one can also subtract and add values. This allows for a number of common statistical measures, including mean, standard deviation, and variance.

Ratio measures These are numerical quantities that have well-defined zero. This includes various physical measures like height or area, age, income (in money, not in income categories) and the like. In case of ratio measures one can claim that one house is “twice as large” as the other house, or one tree is “three times older than me”. Note that while ratio measures require the existence of a well-defined zero, they do not require any objects actually to be of measure 0. For instance, in case of human height, 0 is very well defined despite of no human ever being of height 0.

A special ratio-related measure is *percent*. By definition, this is a proportion and hence requires a ratio-type measure. For instance, if elevation of Mount Adams is 3,743 m, and that of Mount Hood 3,429 m, then Adams is $(3,743 - 3,429)/3,429 = 1.092$ times higher than Hood. This is 9.2% difference. Note

¹Although, strictly speaking, one cannot compute the differences, it is fairly common in practice when comparing different samples. For instance, one may find that the average support is 4.0 among those without college degree and 3.5 among the college graduates. Such averages serve for quick and easy comparison of groups.

²NASA data

that we have used Hood's elevation as the base here, related to the expression "...higher than Hood". Alternatively, we can also use Adams as the baseline: Hood is $3,429/3,743 = 0.9161$ times higher, i.e. 8.389% shorter than Adams.

A measure closely related to percents is *percentage points*. This is difference between two values, expressed in percentages. For instance, ECB deposit interest rate at the end of 2008 was 2.00 percent and refinancing rate was 2.50 percent. The difference between these rates was 0.5 *percentage point*. One can also say that refinancing rate was 25% higher than deposit rate as percent measures are ratios. However, such percent-of-percent figures are rarely used as this is a perfect source of confusion.

Table 1.1 summarizes the measures and some of the related descriptive statistics.

Table 1.1: Quantitative measures and associated statistical operations

measure	operations	plots	examples
nominal	equality, count categories	(unordered) histogram	bicycle brands
ordinal	+ greater/smaller	(ordered) histogram, median	income categories
interval	+ add/subtract: mean, variance, standard deviation		temperature, IQ, GPA
ratio	+ divide		length, height, income

One should also be aware that the boundaries between the measure types may not be quite clear cut. As soon as one uses numeric labels for a variable, one can do all mathematical operations with these data. The question is more about whether the results have any applications. For instance, imagine financial data that contains a student status variable with two potential values *student* and *not student*. These are clearly nominal variables. But we can mark "student" as 1 and "not student" as 2. These two numeric labels are as arbitrary as any other labels, but as these are numeric, we can now perform mathematical operations with these, e.g. compute the mean. The result, most likely a number between 1 and 2, will not carry much meaning if applied to a particular person. However, it is a good descriptor of "studentness" of datasets, i.e. what is percentage of students or non-students in any particular dataset. In a similar fashion, one can assign sequential numeric codes to ordinal measures, such as language skills, to compute an average or standard deviation. This average by itself does not carry much meaning but is useful for comparing samples.

Are these nominal, ordinal, interval, or ratio measures?

- talent show result (e.g. first place, second place, 10th place...)
- height in cm

- height in feet, in
- colors by name
- temperature in C
- IMDB movie ratings (on scale 1-10)

1.1.2 Values: Which Values Are the Possible

While measures described the nature of data from the mathematical operations' point of view, it does not explain what kind of values are possible. Here we describe a common types of values. This is in no way an exhaustive list.

Discrete labels Quite often the values must belong to a pre-determined set of discrete labels. These are often nominal measures but they do not have to be nominal.

Example: your major must belong to a set of all majors offered in the college.

Counts Counts must be non-negative integers. Any other value is clearly erroneous.

Example: number of children in a family.

Continuous positive measures Certain values can only be non-negative. For instance, length or light intensity can take any fractional value, however, both must be non-negative.

Other limited values There are a plethora of other possible limitations. Some figures must fall in a certain range, for instance unemployment rate must be between 0 and 1 by construction.

1.2 Doing Descriptive Statistics

Descriptive statistics is largely data description. It serves several purposes, including to familiarize the analyst and the reader with the data, and to provide an easy overview of the traits in the data that are central for the current analysis. Descriptive statistics is also a useful way to summarize a huge amount of data into a few manageable figures.

1.2.1 Data Quality

In many cases it may not be necessary to inform the reader or to provide a data summary. However, it is almost always extremely important for the analyst to familiarize herself with the data. There are multiple important steps:

- How is data collected? This is mainly for understanding the external validity of the results. Ideally, the data represents the population under study well.

See more in [Section 1.2.3 Sampling](#).

- What variables/information are in data and how these are encoded? A good starting point may be to start from reading the documentation. There are well-documented or easy-to-understand datasets but too often one has to rely on just eyeballing the numbers and doing guesswork based on the variable names. As good documentation requires a lot of work, it is often skipped over, and you as an analyst will suffer because of that.
- Does it contain information you need? If we are interested in relationship between income and education, it is not just enough to have a dataset that contains “income” and “education”. Both of the variables may coded in a way that is not informative for our purpose. Imagine the case where we want to say something about how income is related to college degree, but the dataset only tells if someone has graduated from high school.
- Missings and implausible entries. The variables we are interested may also suffer from many missing values, and from implausible entries. For instance, what should one do with negative income? Or negative age?
- How are values coded? Sometimes it may be obvious, e.g. if *age* value fall between 18 and 81 it is probably age in years. But if variable *sex* has values 1 and 2? Or maybe 1, 2, 9? Unless there is suitable documentation, it may be not possible to deduce the meaning of these values with certainty.

Understanding all such nuances is a substantial work, and in case of large datasets researchers usually try to stay within the realm of the data they know.

1.2.2 Describing Data

When we have satisfactory answers to the questions above, we may want to get a quick overview of the content of data itself. This mainly serves the purpose of getting a quick overview of the processes we are interested in, it may also be useful for assessing the informational content of data (we don't need much data—we need much information!).

Here we describe three traits in data: central tendency, such as mean; variation, such as range and variance; and distributions in the form of histograms.

Central Tendency

One of the crucial bit of information about data is where are the data points located. And we may want to summarize the location with just a single number. Mean and median are the most popular such numbers.

Mean Mean is the most popular way to describe the location (the “center”) of the data. For N observations x_1, \dots, x_N it is defined as

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i. \quad (1.2.1)$$

Mean is very intuitive measure and humans have good innate abilities to estimate mean value by just looking at the sample.

Computing mean requires the data to be of interval measure, otherwise addition is not defined. However, also often sloppily applied to ordered measures when comparing distributions. In that case mean is, strictly speaking, not a central tendency measure but just a test statistic we are comparing across distributions.

The main disadvantage of mean is that it is sensitive to outliers and missing values. For instance, consider data 2.1, 2.2, 2.3. Its mean is $\bar{x} = 2.2$. However, if in case of a data entry error we have 2.1, 2.2, 23 instead, the mean will be 9.1. If any data point is missing, mean cannot be computed at all. Mean is not *robust statistic*.

Mean may correspond to a non-existing or even impossible case. For instance, we may find that an average family has 0.5 children. One cannot conclude from this number that industry should supply more kids “half-beds”. But in order to evaluate the demand for daycare or school places, this number is very much applicable.

Mean is a good description for data that is fairly concentrated. For instance, if all employees have income between 40,000 and 60,000, the mean would describe all these salaries fairly well. But if our sample contains 100 people in poverty (income 10,000) and one billionaire (income 1,000,000,000), the average (9,900,000) does not describe the incomes well. One may wrongly assume that everyone in this sample has income around 10M, and hence poverty is not an issue.

Mean is the sample analog of [expected value](#) (see Section 1.3.4).

Median Median is another popular measure of data location. Median is the “middle value”, a value where there is an equal number larger and smaller values in the data. For instance, in a dataset 1, 2, 3, 4, 10, median is 3 as there are two smaller and two larger values. If there is no such datapoint, e.g. in a sample 1, 2, 3, 4, the median can be defined in different ways, one encounters values 2, 2.5 and 3.

Median is much less sensitive to outliers than mean. If we take the example above where instead of 2.1, 2.2, 2.3 we observe 2.1, 2.2, 23 due to a data entry error, we can see that the error leaves median, 2.2, unchanged. Median is a robust statistic. Median is also less affected of missing values. For instance, consider the same data but now assume the last observation is not wrong but missing: 2.1, 2.2, *NA*. While we cannot say anything about the mean, we can still stay that $2.1 \leq \text{median} \leq 2.2$: if the missing value is larger than 2.2 then median is 2.2, if it is smaller than 2.1 then median is 2.1, and if it is somewhere

in between, then the unknown value is also the median. We are not quite sure about the median value but in this example we can give fairly narrow bounds. Computing median involves just comparison and no addition as in case of mean. So it can be computed on ordered measures, interval properties are not needed.

Median describes well “typical” values in data but fails to capture information about “non-typical” values. For instance, in case of the poverty-billionaire example, the median income will be 10,000. The median person is in poverty. However, median does not provide any hint about the fact that we also have a billionaire in the sample. In a similar fashion, if we find that median household does not have any children, we cannot conclude that no household have any kids. In order to design family policies we have to incorporate other values than median.

Mode The third popular measure of data location is mode. Mode is just the most common data value. For instance, if data looks like 1, 2, 2, 3, 3, 3, the mode is 3. Computing mode only requires comparing equality, so mode is well defined even for nominal measures. However, mode may not work well for continuous values. In case of discrete outcomes, there is only a limited set of possible values, but in continuous case it is unlikely that many data points have exactly the same value. Computing mode for continuous variables typically includes some sort of smoothing, and thereafter finding the maxima of the smoothed values.

Many types of data are *unimodal*, i.e. they have a single mode, often around the middle of the values (given the data is ordered). Other types of data are *bimodal* or *multimodal*, i.e. they have two or more different values that are most common. Normally one talks about bimodal distribution even if the two modes do not have the exact same frequency, but are clearly separated with less common values (see Distributions below).

Exercise 1.2.1: Mean, median, mode

Consider data $\mathbf{x} = (1, 2, 3, 3, 3, 5, 5, 10)$. Compute

1. mean
2. median
3. mode

Now assume the first observation is missing: $\mathbf{x} = (NA, 2, 3, 3, 3, 5, 5, 10)$. What can you tell about mean, median and mode?

Variability

While humans have very good intuitive idea of mean, our understanding of variability is not as good. We can still well understand range but variance and standard deviation are much harder to grasp.

Range TBD: ends must be dense

Sample variance Variance and standard deviation are measures of variability that are extremely important from the theoretical point of view. Many common statistical tests, including *t*-test, are based on these values.

Sample variance is defined as average squared deviation from sample mean:

$$s^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (1.2.2)$$

where \bar{x} is the sample mean. This is not an intuitive measure but we can think of variance as a “typical” squared deviation from the mean. Sample variance does not have standard notation but s^2 is a fairly common notation.

Variance has a number of advantages over just data range:

- Variance is much less sensitive to lone outliers. Although it is still sensitive—the definition involves the deviation squared, so large deviations have overly large influence—it also includes an average over all other data points. So variance is “made” of all data, not just of two extreme observations.
- Variance, in particular its [analogue for random variables](#) (see Section 1.3.4), is an extremely important theoretical concept.

There are two main disadvantages of variance: it is not an intuitive concept despite of its theoretical importance, and second, it is measured in squared units. For instance, if we are working with human age data, variance is measured in years squared. This is not an unit that we understand. Fortunately, this problem is easy to ameliorate. We can just take square root of variance, measured in the same units as data. This is called *standard deviation* or *standard error*. The difference between these two concepts is mostly semantic, they are equivalent descriptions of variability. The notion “standard deviation” is used primarily in the context where we talk about variability in data, “standard error” in a context where the variability describes uncertainty in our results. Standard deviation is denoted in various ways, in formulas it is often s (as square root of sample variance s^2), in text and tables it is often written as *std.dev* and *std.err*.

Variance can be computed using definition (1.2.2) above. Let us compute variance for data (1, 2, 3). We do this by constructing a table for the auxiliary results (Table 1.2). The first column in the table just displays the data, average of which, \bar{x} is in the last row. In the second column we compute the deviation from the average, and the last column displays the deviation squared. The average of the latter is variance, in this example $s^2 = 2/3$. Note also that the middle column, $x - \bar{x}$, averages to 0. This is always true by the definition of mean, and explains why we square the deviations in the definition of variance.

This approach, based on the definition (1.2.2) is easy when coding, when computing variance manually, it is easier to use the shortcut formula

$$s^2 = \bar{x^2} - (\bar{x})^2, \quad (1.2.3)$$

Table 1.2: Computing variance. The last row displays the averages, the of those is just the sample average $\bar{x} = 2$, and the last one is variance s^2 . Note that the average of the middle column is 0. This is always true through the definition of mean.

x	$x - \bar{x}$	$(x - \bar{x})^2$
1	-1	1
2	0	0
3	1	1
average	2	0
		2/3

the difference between mean of x^2 and the square of mean x . For the example in Table 1.2, we see immediately that $(\bar{x})^2 = 4$, and we can easily find that $\bar{x}^2 = (1 + 4 + 9)/3 = 4\frac{2}{3}$, and hence their difference is $s^2 = 2/3$, the same number we found when using the definition (1.2.2).

Proof 1.2.1: Where the shortcut formula is coming from

The shortcut formula can be derived from the definition of variance (1.2.2). We start by opening the parenthesis and re-arranging the terms:

$$\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 = \frac{1}{N} \sum_{i=1}^N (x_i^2 - 2x_i \bar{x} + \bar{x}^2) = \quad (1.2.4)$$

$$= \frac{1}{N} \sum_{i=1}^N x_i^2 + \frac{1}{N} \sum_{i=1}^N (-2\bar{x} x_i + \bar{x}^2) = \dots \quad (1.2.5)$$

Here we already have the first term, \bar{x}^2 . Now we use the fact that \bar{x} does not depend on i can be take out of the sum:

$$\dots = \bar{x}^2 - 2\bar{x} \frac{1}{N} \sum_{i=1}^N x_i + \frac{1}{N} N \bar{x}^2 = \dots \quad (1.2.6)$$

Here we have used the fact that as $\sum_{i=1}^N \bar{x} = N \bar{x}$. And finally, using the definition of mean we have

$$\dots = \bar{x}^2 - 2\bar{x} \bar{x} + \bar{x}^2 = \quad (1.2.7)$$

$$= \bar{x}^2 - \bar{x}^2. \quad (1.2.8)$$

This is the shortcut formula.

TBD: Exercise, properties: multiply by a constant

It also appears that the results computed from (1.2.2) on small sample tend to underestimate the variance on a larger sample of the same data. This can be easily understood when looking at a sample of a single observation only.

Obviously, in this case $\bar{x} = x_1$ and hence $(x_1 - \bar{x})^2 = 0$ and we have the sample variance $s^2 = 0$. This is definitely an underestimate for anything besides constant values. The solution is to compute the corrected variance, often called *population variance*

$$s^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2. \quad (1.2.9)$$

The difference is the value $N - 1$ instead of N in the denominator. This inflates the sample variance estimator, and now the estimates are not too small even on small data. For instance, in our single observation example, this formula would give $0/0$, an undefined value.

The bias correction, using $N - 1$ instead of N when computing the average, is related to the fact that we typically compute the average \bar{x} on the same data. This removes one degree of freedom from data, and hence we use $N - 1$, not N nor $N - 2$ when computing variance. One can intuitively see that data where the mean is “extracted” is “poorer” than the original data. Its average must now be 0, and hence if you know the values of $N - 1$ data points, you can immediately compute the value of the N -th data point. Only $N - 1$ data points are “free”, hence the degrees of freedom is $N - 1$. But if we compute \bar{x} through other means, not from the same data, the unbiased variance should be computed using N , not $N - 1$. See more in Section 1.3.4.

The two variance concepts, sample variance and population variance, are a source of a lot of confusion. Fortunately the difference in anything resembling of a dataset of a reasonable size is minimal.

An additional source of confusion is caused by the theoretical concept corresponding to population variance that is also called [variance](#) (see Section 1.3.4). These concepts are related in a similar way as mean and expected value, but unfortunately they share the same name.

Distribution TBD: Histograms

TBD: Density plots

TBD: Sample quantiles

1.2.3 Sampling

Sample and Population

In statistics-related tasks one typically wants to make conclusions about the whole “population” based on a small “sample”. Intuitively, we need a large enough sample, that covers all the important subgroups. Here we call *population* the whole set of cases we want to apply our results to. In contrast, *sample* is the set of observations we have access to. For instance, based on a sample of 1000 voters, a consultant may make claims about the election outcome, decided by the population of all voters.

Why do we need to consider sampling and sampling designs?

- It is rarely possible to collect data on the complete population. Even if possible, sometimes it is cheaper to collect and analyze a sample, and generalize the results to the population.
- Sometimes it is inherently infeasible to collect data about the whole population. This includes cases where we are concerned about future or the past.
- Other times we can collect the data about “everything” but we still want to generalize our results to even larger populations. For instance, we can easily collect grades of *all* students of a particular course. Do we still need to generalize? This depends on the question:
 - If we are only interested in students of that course then we have the full population. We do not have to consider generalization issues. In this case the sample and the population are the same. Say, if the instructor is concerned that the grades are too low and considers curving those up, there is no need to think about generalization of any kind.
 - If we are interested in “all” students in “similar” courses then our data only represents a sample. For instance, we may be concerned that the course may be hard for students with certain background. This concern is a generic one, also applying to the future students in similar courses. In this case we need to generalize from the sample (students of this course) to the population.

Sampling Process

Getting a sample of data typically involves many steps, some of these deliberate choices, and some caused by external factors, such as data or funding availability.

Sampling contains broadly such steps:

1. Theoretical population. Who (or what) do we want our results to generalize to?
2. Study population. What part of the theoretical population can we access?
3. Sampling frame. What part of the population will be studied? This is the part of the population that is accessible from the practical point of view, i.e. we have information about their presence and location, and can realistically access them. It is often based on proxy information, for instance phone directory when surveying humans, or geographic location when counting wildlife.
4. Sample. What part of the population do you end up getting data for?

Example 1.2.1: Predicting election results

Look at the sampling process when predicting election results.

1. The population of interest is clearly all actual voters (i.e. those who will cast their vote, not all registered voters), and we are interesting in finding their favored candidates.
2. Study population is a list of voters we have records about, either their address, phone, or just the fact that they exist. If the analysts have access to all registered voters records, then the study population will almost overlap with the theoretical population. However, the overlap may still not be perfect, as between now and the election day, more people may register as voters, and some in our records will die (or otherwise leave the records).
3. Sampling frame is a (potential) voter register with contact information. In the best case this is the actual voter register, but it may also be any other accessible proxy, e.g. phone directory, street maps, or lists of public places to visit.
4. Sample. This is taken from the sampling frame, the voters we were able to access and who did answer the questions about their (prospective) voting behavior.

Each step in the sampling process can introduce a corresponding error.

- *External validity* concerns the generalizability of study population to the theoretical population. For instance, are the results collected for current students also valid for future students?
- *Coverage error* are errors, resulting from non-perfect overlap between study population and sampling frame. If many voters do not have phone, we miss those voters.
- *Sampling error* arises from the fact that instead of the whole population, we only work with a small population. Sampling errors can be addressed by increasing the sample size, if feasible. If the sampling process is well known, the errors can also be quantified, and taken into account through confidence intervals .

Sampling bias is a combination of coverage error and external validity problems. Sampling bias can sometimes be accounted for if we know the sources of coverage error and validity problems. But full extent of these problems is rarely known and hence the sampling bias is a common issue, and the results may lack external validity.

Sample without replacement:

- Everyone is sampled either 0 or 1 times

- If sampled, you are removed from the “at risk” population

Example: urn with 2 white and 2 black balls. What is the probability to sample 2 black balls?

- The probability to sample 1 black ball is $1/2$ (2 out of 4)
- The probability to sample 2nd black ball is $1/3$ (now 1 out of 3 is left black)
- Hence the answer $1/2 \cdot 1/3 = 1/6$
- Everyone can be sample 0 or more times
- If sampled, you are put back to the “at risk” population

Example: urn with 2 white and 2 black balls. What is the probability to sample 2 black balls?

- The probability to sample 1 black ball is $1/2$
- The probability to sample 2nd black ball is $1/2$ (still 2 out of 4 is left black).
- Hence the answer $1/2 \cdot 1/2 = 1/4$

1.2.4 Misusing Statistics: Lies, Damned Lies, and Statistics

Statistics is often colloquially accused of being unreliable, and sometime one can hear claims that “anything can be proven with statistics”. There are obviously many reasons for such unfavorable image for statistics, we discuss here just a few.

Uncertainty Statistics works with uncertainty. Even more, most of the statistical results are uncertain. A typical result of statistical analysis reads like “it is more than 95% certain that the difference between the two groups is at least 1”. Understanding such claims requires quite a bit statistical literacy and willingness to think a second, both of which are often in short supply. Compare this with some other results, e.g. India-Iran soccer match ended with 2:1. In the former case statistics *cannot* predict precise results, while in the latter case it is trivial. As a result, statistical claims are often simplified into “everyday” language in a wrong way. For instance, the “layman’s version” of the claim above may be “the difference is 1”. This may be incorrect.

Misleading Presentation Another common issue is to present correct results in a misleading manner. This often includes mixing everyday language where words typically have more vague meaning, and statistical or logical language where the words have slightly different meaning. As an example, one may claim that certain food makes it “twice more likely” to get cancer, and the difference

is “highly significant”. However, even if both claims are true, this alone does not give enough information for policymakers. We need somewhat different information: given someone eats this food, how much more likely it will be, in *absolute terms* (percentage points), to get cancer? For instance, if the cancer rate grows by 1 percentage point, from 1% to 2%, this will amount to 1% of those who eat the food. But twice as large may also mean an increase from 0.0001% to 0.0002%, and increase of 0.0001 pct points, or one in million. In the latter case the problem is much less urgent, and there are probably much easier ways to improve public health. The problem here is that the words “twice as likely” and “significant” in everyday language suggest the presence of an important effect. But this may not be true if we interpret these words in the strict statistical sense.

This kind of misleading presentation is sometimes related to lack of statistical literacy, but sometimes it is also a deliberate strategy to advance a different agenda.

Exercise 1.2.2: How to multiply wealth of all Icelanders

Misleading presentation is sometimes deliberately used in politics. Imagine a politician running for an office in Iceland with a slogan *Vote for me! I'll multiply the wealth of all Icelanders!* How can she fulfill her campaign promise if elected? What exactly is misleading in this claim? For reference, the population of Iceland is 360,000 and its total wealth is \$38 billion.

Incomplete/Missing Data We seldom enjoy working with high-quality data that describes the problem we are interested very well. Data is often collected for another purpose, omits or under-represents certain population groups (non-homogeneous or unknown sampling), and only contains proxies for what we want to know (low or unknown validity). It is easy to come to wrong conclusions when ignoring these issues.

Sometimes the analysts forget to include in the model the features that are *not* in data. For instance, in order to analyze the efficacy of tourniquet to save life in case of severe injuries, a study looked at its usage in Iraq war. The data, collected by hospitals, contained information about injuries, treatment (using tourniquet or not), and whether the soldiers survived. The authors did not find any difference related to tourniquet usage. But what was missing in the analysis? When using hospital data, the could only include information on soldiers who reached hospitals. Those who died before reaching hospital were not included in data. Hence, even if tourniquet has no effect on those who reach hospital, we cannot conclude this is true for the first stage between battlefield and hospital.

Exercise 1.2.3: Damage in Allied bombers

You are a statistician, attached to the Allied air force during the WW2. Your task is analyze the damage of the bombers returning from missions over the continent, and based on the damage pattern to make recommendations about where to place armor. As armor is heavy, one cannot just armor the whole airplane, but it is feasible to put armor on certain vulnerable places. Your analysis reveals that the planes tend to have a lot of damage in wings and in the fuselage. You don't see many damaged engines and cockpits.

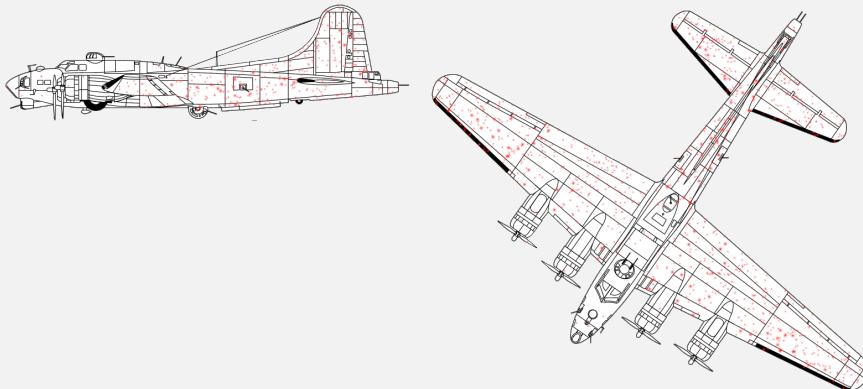


Figure 1.1: Hypothetical damage in allied bombers. Red dots denote damage in any of the thousands of bombers, marked here on a single figure. Original image: Emoscopes / CC BY-SA (<https://creativecommons.org/licenses/by-sa/2.5>)

What is your recommendation: which parts of the airplanes should be armored?

Wrong Assumptions/Wrong Models As in any other analysis, not just our data but also our models must be correct to produce correct results.

A common fallacy is to forget about confounding factors and claim causality when just observing correlation. For instance, when observing that taller children are smarter, one might conclude that height somehow causes cognitive skills. However, the reason may just be that taller kids are older (and we know that skills develop rapidly over time).

Just Errors Finally, it is also possible that statistical analysis contain just simple computation errors or other similar errors. If the methods are feasible and well-known, these are easy to correct.

Too imprecise questions Sometimes a bad result starts with a vague question. Both our language and our thoughts tend to be imprecise, and we may not realize that a question cannot be answered, or can be answered in many different ways, before actually attempt to answer this in a logical fashion.

Consider a question: which country is more dangerous in terms of shark attacks on people—China or Indonesia? What would be the answer to this question? Just count of shark attacks in those countries? But populations differ, so perhaps number of attacks per person? Or perhaps the number of attacks per swimmer, as the inland population may not matter much in terms of sharks? But both of these countries contain a large number of different beaches, some which virtually never see any sharks? So do we want to compute a probability to be attacked on an “average” beach if you swim there? Is this number useful for anything?

Note that if policymakers want to consider installing shark nets or banning swimming on certain beaches, they need information about particular beaches, not country averages.



Figure 1.2: Raja Ampat islands in Indonesia have some of the most beautiful beaches on this planet, but some of those are also frequently visited by sharks. Does this make Indonesia dangerous? Rolandandika, CC BY-SA 4.0, via Wikimedia Commons.

1.3 Probability

This section discusses probability theory, in particular the concepts of *random variable*, *expected value* and *variance*. We use these concepts extensively later in statistics.

1.3.1 Events and Sample Space

Probability theory is concerned about probability of *events* that may or may not occur in the task, in the stochastic phenomenon or experiment we are analyzing. Obviously, what kind of events are out there depends on the phenomenon. Some example events include

- Flip a coin. An event is *get heads*;
- Roll two dice. An event is *get at least one six*;
- Flight arrival. An event is *the plane arrives in time*.

It is often useful to distinguish between *simple events* and *compound events*. Simple events are such events that cannot be partitioned into anything simpler, while compound events can. As an example, event of *heads in coin flip* cannot be divided into anything more basic. But *get a six when rolling two dice* can be any of $(1, 6)$, $(2, 6)$, $(6, 6)$ or a number of other possibilities. In a similar fashion, *plane arrives in time* may mean it arrived *exactly* in time, or also that it arrived (exactly) 2 minutes early.

All possible events together form *sample space* \mathcal{S} . So sample space is a set of all kind of events that can occur in the phenomenon we are considering. Although the concept may feel trivial, it is extremely helpful when thinking about the random outcomes. Here are a few examples:

- Toss a coin. There are only two options, heads and tails, so $\mathcal{S} = \{H, T\}$
- Roll two dice. Each die can come up with sides 1 to 6, so we the sample space is a set of tuples (ordered pairs)

$$\mathcal{S} = \left\{ (1,1), (1,2), \dots, (1,6), (2,1), (2,2), \dots, (2,6), \dots, (6,1), (6,2), \dots, (6,6) \right\}.$$

Note that we distinguish $(1, 6)$ and $(6, 1)$, i.e. we distinguish between the first and second die. These two simple events make the compound event *one and six*. If the dies are so similar that we cannot distinguish these, we may consider these two events to be a simple event instead.

- Flight delay. This can be any number, and we cannot really put a lower or upper limit on it in general, so we can consider the sample space to be

$$\mathcal{S} = (-\infty, \infty)$$

The first two of these examples are finite discrete sample spaces. The third one is a continuous one. Note also that the first two are not numeric: when tossing coins, we receive heads and tails, not numbers. When rolling two dice, we receive pairs of numbers, not numbers. Or, to be even more precise, we receive pairs of sides with a certain dot patterns on them. Finally, the third example, the flight delay, is numeric, but not just numeric but it also has a unit (say, delay in minutes).

TBD: mutually exclusive events

TBD: Probability

1.3.2 Conditional Probability and Bayes Theorem

Prerequisites: events, sample space

TBD: history

Bayes theorem is a rule about computing conditional probabilities—probabilities that something happens, given something else happened. Conditional probabilities play a very large role in statistics and machine learning, in a sense all supervised learning is about computing conditional probabilities. For instance, if you are predicting house prices based on house size, you can ask questions like “what is the probability that this house costs over \$500,000 given it is 200 m² big?

Conditional probability of event A happening given event B happens is denoted by $\text{Pr}(A|B)$. For instance, in the house price/house size example, the question can be written as $\text{Pr}(\text{price} > 500,000 | \text{size} = 200)$.

Venn Diagram

The rules to compute conditional probabilities can be explained using *Venn diagram* (Fig 1.3). The figure depicts the sample space S (the white rectangle) and two events, A and B . (It is best to understand these as compound events, consisting of single points as simple events.) The events overlap to a certain extent, i.e. it is possible that both A and B occur. But they overlap only to a certain extent, so if A occurs, it is not certain that B will occur and the way around. Finally, as the events do not occupy the full sample space, it is possible that neither will happen.

An example of such two events may be $A = \text{it is raining}$, and $B = \text{the class is canceled}$. It is obviously possible that neither of these two events happens (it is not raining and the class is not canceled), so these two events do not make a complete sample space. Alternatively, it is also possible that only A happens (it is raining but the class takes place), only B happens (it is not raining but the class is canceled), and finally both of these may happen too. However, in other type of examples not all four options may be possible. For instance, in case of coin toss, heads H and tails T are mutually exclusive events and hence it is not possible that both of these occur simultaneously. Even more, there are no more possible events in the sample space and hence either H or T occurs for sure.

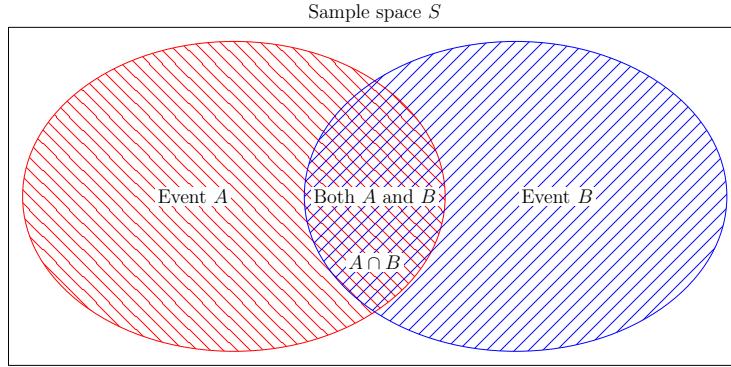


Figure 1.3: Venn diagram. The events A and B overlap partially, so it is possible that only A happens, only B happens, and both A and B happen. As A and B do not sum to the whole sample space (there is a white “leftover area”), it is also possible that neither happens.

Formally we denote the corresponding probabilities by $\Pr(A)$ for the probability of event A and $\Pr(B)$ for the event B . Further, we denote by $\Pr(A, B) \equiv \Pr(A \cap B)$ the probability that both A and B occurred. Normally we prefer the shorter notation $\Pr(A, B)$ to denote joint events (this is also common in literature), but sometimes we want to stress that this is an overlap of A and B and we write $\Pr(A \cap B)$. We use these concepts to define conditional probability below.

Conditional probability is the probability that the other event occurs, given the first event takes place, for instance the probability that B happens, given A happens. Note that the conditioning is not necessarily related to timing or causality. There is nothing wrong to discuss conditional probabilities like “what is the probability that the weather is dry today, given it will rain tomorrow?” Also, conditional probability is not the same as causal relationship. While house size is definitely part of factors that determine the house price, it is not always the case. For instance, when computing probability that an email is spam given it contains the word “viagra”, we cannot say that the word “causes” email to be spam. Email is either spam or not, and spam emails are more likely to contain certain words than non-spam emails.

Formally, we denote the probability that B occurs conditional on A occurring as $\Pr(B|A)$ and the opposite probability, that A happens given that B happens, as $\Pr(A|B)$. In machine learning context, a major application of conditional probability is predictive modeling. Using statistical tools we are trying to answer the question: what is the probability of the outcome Y , given the data X ?

Let us now discuss $\Pr(A|B)$. Intuitively, computing conditional probability involves conditioning, focusing on event B only. Essentially we analyze now a smaller sample space, $S_B = S \cap B$, the blue oval in Figure 1.3. In this new smaller sample space, the event A transforms to $A_B = A \cap B$, the red and blue

overlap area in the Figure. In the new, conditioned-on- B world, the probability of B (or more precisely, $\Pr(B|B)$) is one. We just ignore all events that do not involve B . One can intuitively see that $\Pr(A|B)$ depends on the “size” of $A \cap B$ relative to B . If all points inside of B and A are equally likely, we can find the conditional probability just by dividing the size of $A \cap B$ by the size of B . Now the events A and B do not have anything like “size”³ but they have well-defined probability. Hence we compute the conditional probability as

$$\Pr(A|B) = \frac{\Pr(A \cap B)}{\Pr(B)} \equiv \frac{\Pr(A, B)}{\Pr(B)} \quad (1.3.1)$$

Example 1.3.1: Gender and Titanic Survival

Consider sinking of RMS Titanic in 1912. She had 1309 passengers, 843 male and 466 female, out of whom 161 male and 339 female survived. Intuitively, it is easy to compute conditional probability that a passenger survived, conditional of being female:

$$\Pr(\text{survived}|\text{female}) = \frac{339}{466} = 0.727.$$

This calculation is essentially an application of Bayes theorem. Consider the figure below.

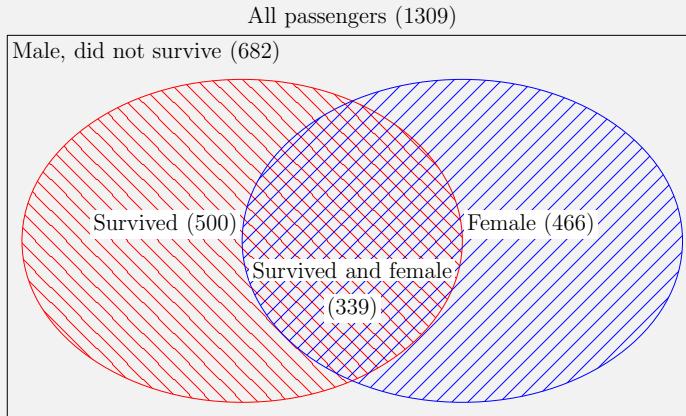


Figure 1.4: Gender distribution among Titanic passengers, displayed as a Venn diagram. Out of 1309 passengers, 466 were female, 500 survived, and 339 were both female and survived.

Our sample space (the box) is “made of” all 1309 passengers. Out of these passengers, 466 were female, i.e. $\Pr(\text{female}) = 466/1309 = 0.356$ (blue on the figure). 500 passengers survived, so $\Pr(\text{survived}) = 500/1309 = 0.382$ (red on the figure). There is also overlap—339 females who survived. We can compute this probability (out of all passengers) as $\Pr(\text{survived}, \text{female}) = 339/1309 = 0.259$ (red/blue cross shaded in the figure). However, we are

³What corresponds to the intuitive concept of “size” is called *measure* in probability theory.

not interested in $\Pr(\text{survived}, \text{female})$, probability that a random passenger was female and survived, but in the conditional probability $\Pr(\text{survived}|\text{female})$, probability that a random female passenger survived. So we should divide the cross-shaded overlap area with the blue female area:

$$\Pr(\text{survived}|\text{female}) = \frac{339/1309}{466/1309} = \frac{339}{466} = 0.727.$$

This is the same result we got above.

Independent Events

Events are *independent* if their joint probability can be factored into a product of two individual event probability. In case of two events:

$$\Pr(A, B) = \Pr(A) \cdot \Pr(B) \quad (1.3.2)$$

In case of K events:

$$\Pr(A_1, A_2, \dots, A_K) = \prod_{k=1}^K \Pr(A_k) \quad (1.3.3)$$

TBD: Examples, Bayes formula, etc

1.3.3 Random Variable

Random variable (RV) is a central concept that connects probability theory to statistics. In particular it makes numbers out of events so that one can use mathematical apparatus to analyze the random processes. The concept of RV is somewhat complex, so we start with the easy part. Namely, it is easy to remember what RV is not–first, random variable is not random, and second, random variable is not a variable.

But then what *is* RV? It is basically a function that connects the events in our stochastic phenomenon with their numeric measured outcomes. The *numeric* is crucial here, the outcome of a RV must be numeric. So RV is a function that assigns a numeric value to each event in the sample space. Formally, a RV X is $X : S \rightarrow \mathbb{R}$. When we talk about RV-s, we talk about random phenomena or experiments.

For instance, if our experiment is flipping a coin, then its sample space is $\{H, T\}$, just heads or tails. We can assign zero to tails and one to heads, and define the RV X as

$$X(E) = \begin{cases} 0 & \text{if } E = T \\ 1 & \text{if } E = H. \end{cases} \quad (1.3.4)$$

Obviously, one can also define it in the opposite way $X(H) = 0$ and $X(T) = 1$; and in a myriad of other ways. For instance, if we want the expected value to be 0, (see [Section 1.3.4](#) for explanation) we can define $X(T) = -1$ and $X(H) = 1$.

Now when we actually conduct the experiment and toss the coin, we will receive either H or T . We use RV X to convert the realized outcome into a number and hence we get either 1 or 0. These are two possible *observed values* or *realizations* of the RV. RV-s are traditionally denoted by upper case Latin letters, such as X or Y and their observed values (realizations) with the corresponding lower case letters, such as x and y . If we observe many realizations (e.g. toss the coin multiple times), we usually denote those using a subscript like x_1, x_2, \dots, x_N .

Remember 1.3.1: Random variable and realization

- **Random variable** (RV) is a way to assign numbers to outcomes in a random experiment.
- **Realizations** are numbers, resulting in a random experiment when converting the outcomes using a RV.
- A number of realizations together form a **sample**.

The different ways to define RV-s is related to questions about the physical world we are interested in. Take example of rolling two dice (see Section 1.3.1). For instance, if we are just interested in different outcomes, we can enumerate the combinations by defining

$$Y(E) = \begin{cases} 1 : \text{if } E = (1,1) \\ 2 : \text{if } E = (1,2) \\ \dots \\ 36 : \text{if } E = (6,6). \end{cases} \quad (1.3.5)$$

Now the RV will tell us if we got (1,5), (5,1) or (2,4). All these combinations correspond to the different values. However, we may not be interested in the different combinations but instead in the sum of the points, whichever sides come up. Now we can define

$$Z(E) = \begin{cases} 2 : \text{if } E = (1,1) \\ 3 : \text{if } E = (1,2) \text{ or } E = (2,1) \\ 4 : \text{if } E = (1,3) \text{ or } E = (2,2) \text{ or } E = (3,1) \\ \dots \\ 12 : \text{if } E = (6,6). \end{cases} \quad (1.3.6)$$

Exercise 1.3.1: Rolling two dice

Take the example of two dice. Construct a random variable that answers the question: did we get any 6-s?

The RV outcomes have, in general, different probabilities as they correspond to different events in the sample space. In case of the coin-toss RV X , the value

0 corresponds only to event T and the value 1 to the event H . As both of these events have probability 0.5 (assume it is a fair coin), the values 0 and 1 will also have equal probability. However, this is not the case for RV Z that counts the points on two dice. Although all the atomic events are equiprobable, the RV values are not because those correspond to different compound events. Value 2 corresponds only to a single atomic event (1,1) and hence has probability $1/36$. Value 3 corresponds to two atomic events, (1,2) and (2,1) and hence has probability $2/36$. Probability of 4 is $3/36$ and so on.

Exercise 1.3.2: Find $\Pr(Z = 6)$

Consider the RV Z as defined above. Find $\Pr(Z = 6)$, the probability that rolling two dice will give you sum 6.

Hint: consider drawing a 6×6 table of faces and marking the sum of the dots in each table cell.

A few more words about the notation. Typically a capital letter, such as Z , denotes the random variable, the process of rolling dice and adding points. Individual realizations, the actual number of points $2, \dots, 12$ we get when we roll the dice, are typically denoted with lower case letters z . As we usually consider many outcomes, we denote those by subscripts z_1, z_2 , and so on. But in different situations the subscripts may mean different things. Occasionally we may roll a pair of dice many times and hence z_1 denotes the sum of the first roll, z_2 the second roll etc. In other times we may want to enumerate the different possible outcomes. Now $z_1 = 2, z_2 = 3, \dots, z_{11} = 12$. In this case $\Pr(Z = z_{11})$ means $\Pr(Z = 12)$, “the probability to get sum 12 when rolling dice”. The notation can be quite confusing, and one has to understand what exactly z_i mean in each case.

TBD: RV as probability table

There are also different ways to define RV-s in case of continuous sample space like flight delay. The first and most obvious case is just to use the length of the delay in, say, minutes. Alternatively one may construct a RV that answers the question: what was the delay, given the flight was delayed? So we are not interested in early arrivals, only delays:

$$X = \max(0, d) \tag{1.3.7}$$

where d is the delay in minutes.

TBD: Functions of random variables

TBD: independent random variables

1.3.4 Expected Value and Variance

Expected Value

The section 1.2.2 above discussed mean as a way to characterize the central tendency of data samples. Intuitively, one can easily understand that as the sample grows, its mean will converge to the “true mean”. When thinking about

“true mean” we intuitively have in mind a more general sample, the “population”, or perhaps a stochastic process where the current data is sampled. This population or stochastic process is essentially a RV and the “true mean” is a certain property of this RV. The property is called *expected value* or *expectation*. It is usually denoted by capital “ \mathbb{E} ”, e.g. $\mathbb{E}X$ means the expected value of random variable X . Its numeric value is often denoted by μ . Unfortunately it is also common to refer to the expected value as “mean”, e.g. when talking about distributions. So “mean” can refer to either sample mean or to the expected value of a RV. However, “average” is never used to denote the expected value.

For discrete RVs, expectation can be computed as the weighted average of possible outcome values where the weights are the corresponding probabilities:

$$\mathbb{E} X = \sum_i p_i \cdot x_i \quad (1.3.8)$$

where i counts over all possible outcomes of X , denoted by x_i .⁴ Consider the coin toss example where we assigned 1 to heads and 0 to tails. The expected value of X is

$$\mathbb{E} X = 0.5 \cdot 0 + 0.5 \cdot 1 = 0.5. \quad (1.3.9)$$

This is intuitively obvious: in average, we get heads half of the times.

Example 1.3.2: Expecation of a 3-valued RV

Consider another, a more complex example. Take a RV

$$Y = \begin{cases} 0 & \text{with probability } 0.5 \\ 1 & 0.25 \\ 2 & 0.25 \end{cases}. \quad (1.3.10)$$

Its expectation is $\mathbb{E} Y = 0.5 \cdot 0 + 0.25 \cdot 1 + 0.25 \cdot 2 = 0.75$

Exercise 1.3.3: Expected value of die

Consider rolling a die as a RV D . Denote its values by $1, 2, \dots, 6$. What is its expected value $\mathbb{E} D$?

The weighted sum (1.3.8) to compute expected value transforms to an integral in case of continuous value:

$$\mathbb{E} X = \int f(x) x \, dx \quad (1.3.11)$$

where $f(x)$ is the probability density function (see Section 1.3.5 below). The integral is taken over the complete support of X .

⁴Note: here i enumerates the possible outcomes, not consecutive experiments. See page 24 for comments on notation.

As an example, we calculate the expected value for standard uniform RV. Standard uniform is a continuous distribution where all values between 0 and 1 are equally likely, and other values are impossible. Its density function is just $f(x) = \mathbb{1}(x \in [0,1])$,⁵ so we just integrate over the $[0,1]$ interval:

$$\mathbb{E} X = \int_0^1 1 x \, dx = \frac{1}{2} x^2 \Big|_0^1 = \frac{1}{2}. \quad (1.3.12)$$

This is an intuitive result: if all values between 0 and 1 are equally likely, we get the average between these numbers.

TBD: Expected value of constant; $\mathbb{E} \mathbb{E} X$

TBD: A more complex example

TBD: An exercise

It is important to keep in mind that expectation is not sample mean and the way around. Expectation is a property of random variable, a precisely defined stochastic process. Sample mean is a property of sample. Even if expectation is sometimes called “mean”, it is important to realize that sample and RV have different properties. For instance, sample mean is random and its mean fluctuates depending on which observations were sampled. But expectation is constant and does not change. Say, tossing coin a few times may result in different means, but the expected number of heads is always 0.5. One can also compute mean for every sample but not every RV has expectation (see, e.g. [Pareto distribution](#) below).

TBD: Conditional expectations

Variance

While expectation is built on the concept of sample mean, variance is built on the concept of sample variance. Like sample variance is much less intuitive than sample mean, so is variance much less intuitive than expectation. The naming convention is not helpful either: unlike expectation versus mean, both of these concepts are called “variance”. We can stress the difference though by writing “variance of the RV” or “theoretical variance” when we talk about random variables, and “sample variance” when we talk about data.

Variance is typically denoted by Var , e.g. $\text{Var } X$ is variance of the random variable X . Its numerical values are often denoted by σ^2 , stressing that its definition is related to squared deviations. This also let’s one to denote the standard deviation by just σ .

Variance is one of the most important statistical concepts, most of the statistical inference is in fact based on variance. It is defined in the same way as sample variance while replacing means with expectations. So variance of RV X is defined as

$$\text{Var } X = \mathbb{E}(X - \mathbb{E} X)^2. \quad (1.3.13)$$

⁵Remember, $\mathbb{1}(\cdot)$ is indicator function (see Section 0.1 on page [ix](#)). It is just a shorthand to write $f(x) = 1$ if $x \in [0,1]$, otherwise $f(x) = 0$.

Let us explain what this means. First, $\mathbb{E} X$ is the expected value of X . It is just a number, a constant. Next, $X - \mathbb{E} X$ is the deviation of X from its expectation. It is just X value minus a number. As X is a RV, so is $X - \mathbb{E} X$. Third, $(X - \mathbb{E} X)^2$ is just a squared value of the deviation. As the deviation was a RV, so is its square. And finally, $\mathbb{E}(X - \mathbb{E} X)^2$ is the expected value of that RV. So variance can be computed in a similar fashion as expectations. Let us consider an example. Take the RV from Example 1.3.2:

y	$\Pr(Y = y)$
0	0.5
1	0.25
2	0.25

Above we computed $\mathbb{E} Y = 0.75$. Let us now compute its variance using the definition. The most straightforward is to extend the table by the auxiliary RV-s (Table 1.3). The first two columns represent the RV realizations y and the corresponding probabilities $\Pr(Y = y)$. The third column is the deviation from the expected value, $Y - \mathbb{E} Y$. The fourth column is the deviation squared. The variance is just the expected value of the fourth column. The probability values are not affected by the other operations, computing the deviation and squaring it. Hence the variance is $\mathbb{E}(Y - \mathbb{E} Y)^2 = 0.5 \cdot 0.5625 + 0.25 \cdot 0.0625 + 0.25 \cdot 1.5625 = 0.6875$.

Table 1.3: Computing variance of the random variable

y	$\Pr(Y = y)$	$Y - \mathbb{E} Y$	$(Y - \mathbb{E} Y)^2$
0	0.50	-0.75	0.5625
1	0.25	0.25	0.0625
2	0.25	1.25	1.5625

The easiest way to compute the variance of a RV by using the definition is to add columns for $Y - \mathbb{E} Y$ and $(Y - \mathbb{E} Y)^2$ in the table of RV values. Variance is simply the expected value of the last column, here 0.6875. See explanations in text.

In practice it is somewhat easier to use another formula

$$\text{Var } X = \mathbb{E}(X^2) - (\mathbb{E} X)^2. \quad (1.3.14)$$

This involves computing the expected value of X^2 . It is easy to show that this formula is equivalent to the definition of variance (1.3.13). Let us re-compute the variance we did above using this formula. First, we have to find $\mathbb{E} X^2$. This is $\mathbb{E} X^2 = 0.5 \cdot 0^2 + 0.25 \cdot 1^2 + 0.25 \cdot 2^2 = 0.25 + 1 = 1.25$. Hence the variance is $\text{Var } X = \mathbb{E} X^2 - (\mathbb{E} X)^2 = 1.25 - 0.75^2 = 1.25 - 0.5625 = 0.6875$. This is the same number we found above.

Exercise 1.3.4: Compute variance of a RV

Consider a RV

x	$\Pr(X = x)$
-1	0.25
0	0.50
1	0.25

Compute its variance using a) the definition formula (1.3.13); and b) the shortcut formula 1.3.14.

TBD: Variance of mean

1.3.5 Distributions

Distribution in case of RV describes the “frequency” of different values. It is a rather intuitive concept and closely related to histograms. We start with discrete RV-s where the corresponding function, *probability mass function* (p.m.f), corresponds to the observed frequency, and move to continuous RV-s thereafter where *probability density function* (p.d.f) has a slightly different interpretation. Another popular measure, *cumulative distribution function* (c.d.f) gives the probability that the observed values is *less* than a given one.

Discrete Case

Let’s start with a simple example: we toss two coins and count the number of heads (assume both are fair coins). We can get the following outcomes:

x	$\Pr(X = x)$
0	0.25
1	0.5
2	0.25

This table essentially describes what is known as *probability mass function* (p.m.f) in case of discrete RV-s. P.m.f is a function that for each possible value of x assigns the corresponding probability. (It can also be trivially extended by assigning the probability 0 to every impossible value.):

$$f(x) = \Pr(X = x) \quad (1.3.15)$$

So we can, somewhat trivially, restate the table as p.m.f:

$$f(x) = \begin{cases} 0.25 & \text{if } x = 0 \\ 0.5 & \text{if } x = 1 \\ 0.25 & \text{if } x = 2 \end{cases} \quad (1.3.16)$$

In a general discrete case, p.m.f must be described as the table above, or a corresponding graph. However, there are numerous processes that generate p.m.f-s with a given structure, for instance the example case of tossing two coins results in a binomial distribution, more precisely in $\text{Binom}(2, 0.5)$.

Another widely used function is *cumulative distribution function* (c.d.f). It answer the question “what is the probability that the outcome is no larger than a given number”:

$$F(x) = \Pr(X \leq x). \quad (1.3.17)$$

If f is the p.m.f, one can easily compute c.d.f as

$$F(x) = \sum_{x' \leq x} f(x'). \quad (1.3.18)$$

In the example case above we have

$$F(x) = \begin{cases} 0 & \text{if } x < 0 \\ 0.25 & \text{if } 0 \leq x < 1 \\ 0.75 & \text{if } 1 \leq x < 2 \\ 1 & \text{if } x \geq 2 \end{cases} \quad (1.3.19)$$

Exercise 1.3.5: Measure for c.d.f

What kind of measure—nominal, ordinal, difference, or ratio—must X be for its c.d.f to be well defined?

Continuous case

TBD: Distribution vs process

Popular Distributions

Here we discuss and give examples of a few commonly used distributions.

Bernoulli Distribution TBD: Bernoulli distribution

Discrete Uniform Distribution This is a discrete distribution where all possible outcomes have equal probability. The examples include toss of fair coin, roll of fair die, or suite of a random card, drawn from the complete deck. Discrete uniform distribution over elements of set \mathcal{S} assigns equal probability on each element of \mathcal{S} . Its p.m.f is simply

$$f(x) = \begin{cases} \frac{1}{|\mathcal{S}|} & \text{if } x \in \mathcal{S} \\ 0 & \text{otherwise} \end{cases} \quad (1.3.20)$$

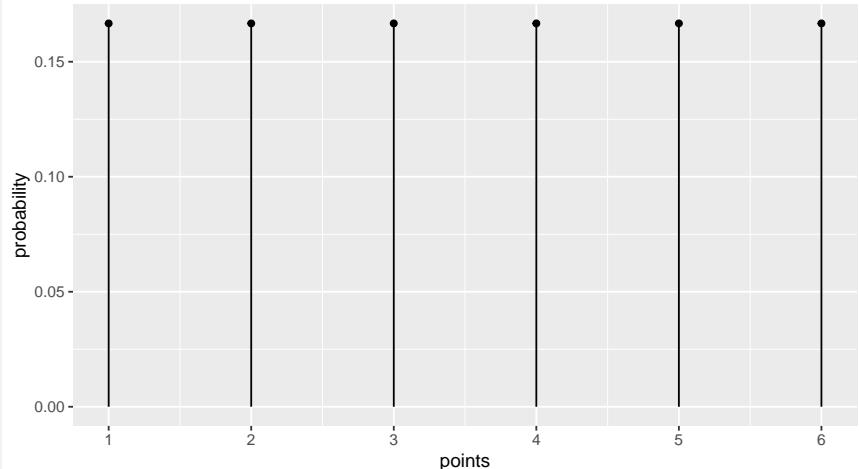
When we talk about “random sample” or “selecting at random”, we usually mean a discrete uniform selection where all possible events have equal probability to end up being selected. Strictly speaking, it does not have to be so—random sample is still a random sample even if the subjects are not picked with the equal probability, but such usage of the word is much less common (a related concept is *stratified sample*).

Example 1.3.3: Rolling a die

On a fair die, all size sides are equally likely and hence the p.m.f is

$$f(x) = \begin{cases} 1/6 & \text{if } x = 1 \\ 1/6 & \text{if } x = 2 \\ 1/6 & \text{if } x = 3 \\ 1/6 & \text{if } x = 4 \\ 1/6 & \text{if } x = 5 \\ 1/6 & \text{if } x = 6 \end{cases} \quad (1.3.21)$$

The function can be depicted graphically as follows:

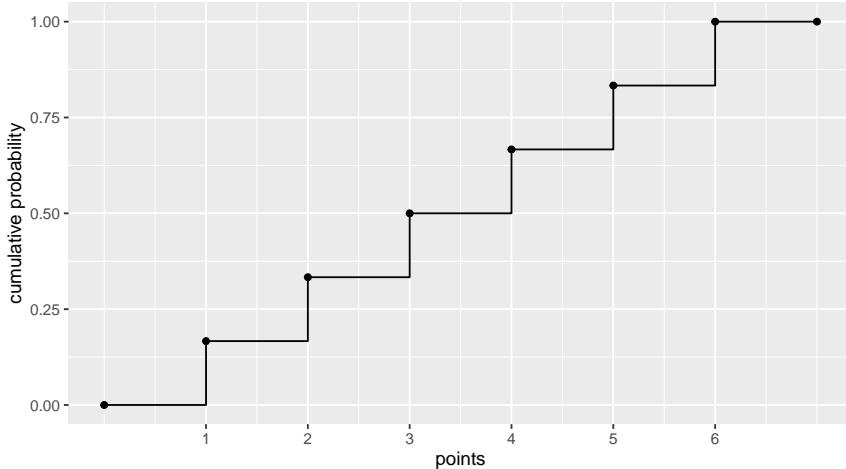


Each bar corresponds to the probability to get the corresponding number on a die, in case of uniform distribution these are all of equal length.

The respective c.d.f is given by

$$F(x) = \begin{cases} 0 & \text{if } x < 1 \\ 1/6 & \text{if } 1 \leq x < 2 \\ 2/6 & \text{if } 2 \leq x < 3 \\ 3/6 & \text{if } 3 \leq x < 4 \\ 4/6 & \text{if } 4 \leq x < 5 \\ 5/6 & \text{if } 5 \leq x < 6 \\ 1 & \text{if } x \geq 6 \end{cases} \quad (1.3.22)$$

and looks like a staircase on graph:



The dots stress that the function has achieved the “upper” level at these points, for instance $F(1) = 1/6$, and not zero.

TBD: Uniform

Exercise 1.3.6: Quantiles of standard uniform distribution

Consider standard uniform distribution

$$f(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \quad (1.3.23)$$

Find the theoretical quantiles $q_{0.025}$ and $q_{0.975}$.

TBD: normal

TBD: t

TBD: log-normal

Pareto distribution TBD: fat-tailed distributions, Pareto

TBD: CLT

Conditional Expectation

TBD: notation, perhaps conditional variance

1.3.6 Information and Entropy

Entropy is a measure of information in a random variable. If a random variable has entropy 0, then we do not gain any information when we learn about the outcome. Everything was already known in advance. In case of positive entropy the result is uncertain, and learning about the outcome helps to clarify this. The

larger the entropy, the more uncertain is the outcome and the more information we gain when we learn about it.

In discrete case where the random variable X can take values $k = 1, 2, \dots, K$, entropy is defined as

$$\mathbb{H}(X) = - \sum_{k=1}^K \Pr(X = k) \cdot \log_2 \Pr(X = k). \quad (1.3.24)$$

When using binary logarithm, the entropy is measured in *bits*. When using natural logarithms, the units are called *nats*. It is easy to see that $1 \text{ nat} = \log_2 e = 1.443 \text{ bits}$. We use binary logarithms in this text but some authors prefer natural logarithms.

Note that for zero-probability states, $0 \log 0$ is undefined. However, as the corresponding $\lim_{x \rightarrow 0} x \log x = 0$ (see (A.1.3) in Section A.1.1) the contribution to entropy is 0. This means that as we expand our sample space with more zero-probability events, the entropy value is not affected. Knowing more types of events that almost never happen will not affect the amount of information we can gain. Only events with positive probability matter in terms of information.

Example 1.3.4: Entropy of uniform distribution

In order to have an intuitive understanding about what entropy measures, let us analyze entropy of discrete uniform distribution. In case of K potential outcome states with equal probability $1/K$, (1.3.24) gives:

$$\mathbb{H}(X) = -K \left(\frac{1}{K} \cdot \log_2 \frac{1}{K} \right) = \log_2 K. \quad (1.3.25)$$

So entropy is just logarithm of the number of states. Figure 1.5 displays such an example. We have 8 possible states, $A-H$, some of which have probability 0 (white on figure) while the others are equally likely. The first row depicts the case where all the probability mass is concentrated in the state A with $\Pr(A) = 1$ while every other state S has $\Pr(S) = 0$. Here we cannot gain any information as we know in advance that A happens for sure. This is reflected by the corresponding $\mathbb{H} = 0$. However, the next state already contains some uncertainty: we don't know if A or B will happen, both are equally likely. When we learn about which event happened, we gain $\log_2 2 = 1$ bit of information. Further down in the table, there are more possible states and hence more uncertainty, and we gain more information when we learn about the outcome.

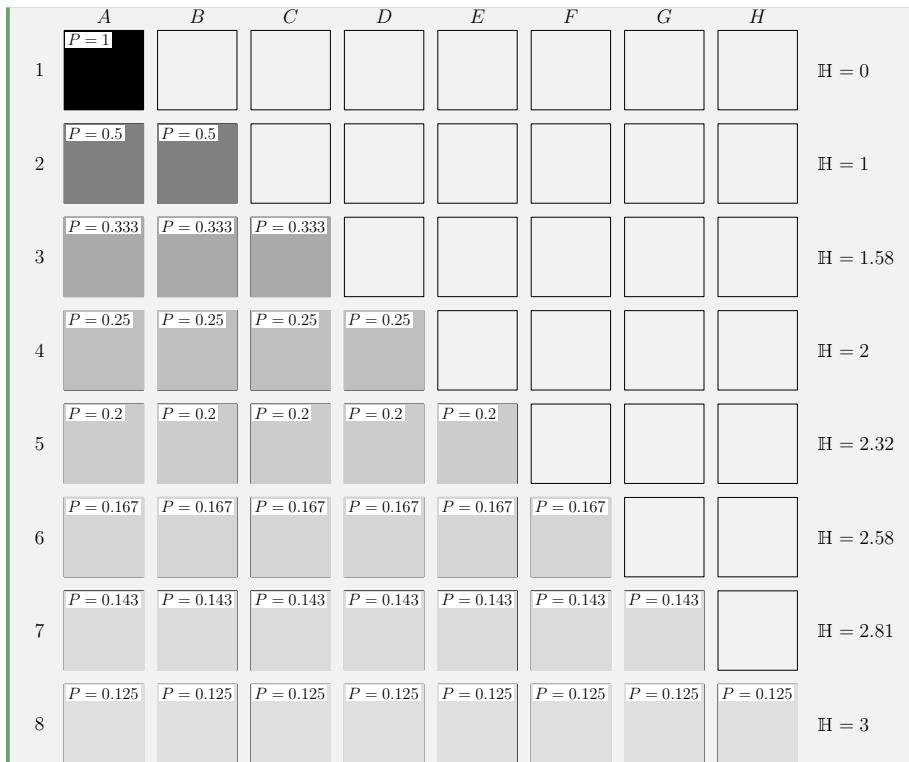


Figure 1.5: Entropy in case of different probability over states. Different shades of gray denote different probability, uniformly spread over 8 states A–H. The rightmost column is the corresponding entropy.

Exercise 1.3.7: Entropy of Bernoulli random variable

Look at a Bernoulli process with parameter p . Compute it's entropy as a function of p . Explain the intuition behind how the result depends on p .

Hint: consider making a plot on computer.

1.4 Statistical Inference

Statistical inference refers to statistically sound conclusions based on data that *can be generalized* to the whole population. The statistical methods we discussed in Section 1.2 are sound, but do not allow generalizations. Descriptive statistics describes data using statistical tools. For instance, we may find that in our sample, mothers who smoke give birth to smaller babies. If this is all we are interested in, we can stop here. But can our data tell something about another sample of mothers and babies? Or about *all* mothers and babies? Yes, the current sample can tell something about other samples, and about the whole population. Inferential statistics does exactly that.

1.4.1 Statistical Hypotheses and Hypothesis Testing

In this section we introduce a lot of concepts: confidence intervals, confidence levels, significance levels, statistical hypothesis and hypothesis testing, and different types of errors. This section just introduces the concepts, how to actually compute the confidence intervals is discussed in Sections 1.4.2 and 1.4.3.

Hypothesis testing and confidence level

Statistical inference is typically done through statistical hypotheses and hypothesis testing. Statistical hypotheses are claims about the world, claims that may or may not be compatible with data. If data contradict the claim, we *reject the hypothesis*, if data are compatible with the claim, we do not reject the hypothesis. Unlike many other fields, statistics normally does not give definite answers. For instance, while economists typically want to say that unemployment rate is “11.2 pct”, a statistics-based answer will include a measure of uncertainty. A statistically correct answer may be “with 95% confidence we can say that unemployment rate is between 10.8 and 11.6 pct”.⁶ Such answers are pretty much the only type of results that statistics can offer.

What does such a claim mean? There are two components here:

- We don’t know what exactly is the unemployment rate, but we are “reasonably certain” it belongs to the interval [10.8, 11.6].
- The “reasonably certain” means we are 95% certain.

As one can see, understanding such somewhat fuzzy claims requires a bit of statistical literacy.

A *statistical hypothesis*, often denoted as H or H_0 , is a claim, usually stated in a definitive manner. In the example above, a hypothesis might be H_0 : “unemployment rate is 11.2%”, or perhaps instead H_1 : “unemployment rate is more than 10%”. The hypotheses can either be *rejected* (it means it is incompatible with data) or not rejected (if it is compatible with data). Note that hypotheses

⁶You may have noticed that such interval-based claims are also quite common in sciences. Often they originate from statistics-based methods.

cannot be confirmed! Hypotheses can only be rejected or not rejected at certain *confidence level*. Confidence level means the probability that the rejection is the correct decision. It is based on data quality, sample size and other factors. In applications we typically look at confidence levels 95% or 99%. If a hypothesis is rejected, we can consider it “wrong”—given our data and methods was correct. If a hypothesis is not rejected, it is compatible with the data. It may be correct, or it may still be wrong if our dataset is too small or too noisy.

Hypotheses are often presented in pairs, where one is called *null-hypothesis* H_0 and the other *alternative hypothesis*, often denoted by H_1 or H_A . The null hypothesis is the original claim we are testing and potentially rejecting, H_A is the alternative that must be true when H_0 is false. For instance, when analyzing mothers’ smoking habits and babies birth weight, H_0 might be “smoking and non-smoking mothers give birth to babies of equal weight in average”. The alternative H_A in this case will be “Birth weight of babies, born to smoking and non-smoking mothers, differs in average”. Note that H_A does not claim that babies of either one or another group weigh more. If the weight is not equal, it must differ. If one wants to test if the weight differs in a certain way, that is a separate hypothesis. While certain sources always state H_0 and H_A explicitly, other studies either only discuss H_0 or leave the hypotheses implicit. In that case it is often obvious from question that is analyzed.

TBD: confidence interval

Example 1.4.1: Rejecting and not rejecting a statistical hypothesis

Assume we analyze unemployment data and conclude that with 95% confidence the true unemployment rate is between 10.8 and 11.6 pct with the best estimate being 11.2.

Now consider the government, that always prefers to paint a bit more rosy picture, claims that the rate is just 8.9%. We can treat this as a statistical hypothesis $H_0 : u = 8.9$ (where u means unemployment rate). The government’s claim is clearly incompatible with our data (and model), after all, according to our analysis, we are 95% confident that the rate is at least 10.8%. So we can reject H_0 and accept the alternative $H_A : u \neq 8.9$. But note that we were just 95% certain that $u \in [10.8, 11.6]$ and hence we cannot reject it definitively, but only with 95% confidence.

However, the politically independent Central Bank has no incentive to make the figures any better than they are and publishes its own analysis according to which $u = 11.4\%$. This can also be written as a statistical hypothesis $H_1 : u = 11.4$.^a What can we say about this? The number 11.4 fits squarely inside our confidence interval and hence the result is compatible with our data. So we cannot reject H_1 . But neither can we tell that it is correct—it is just compatible with data, maybe correct, maybe not correct.

^aHere we use H_0 and H_1 to denote different hypotheses. H_1 here is not the alternative to H_0 , the alternative to the latter is H_A above.

Which hypotheses can be rejected depends on data quality—how much rele-

vant information there is in data, and on the analysis—how well can we extract that information. The lower the data quality, the less can we tell, the fewer hypotheses can we reject. In the extreme case where the data contains no information (or we do not have any data), we cannot reject anything.

Example 1.4.2: Unemployment example with bad data

Now imagine we only have access to inferior data and our results are much less precise. We are only able to conclude that unemployment must be in a range of $[7, 14]\%$. Can we now reject $H_0 : u = 8.9$ (the Government's claim) and $H_1 : u = 11.4$ (the Central Bank's claim)? As both of these fit into our interval, we cannot reject either of them. Both claims are compatible with our low quality data. But we can still say that a scaremongerer who claims $u = 30$ is wrong.

However, if we do not have any data, the only thing we can say is that $u \in [0, 100]$ (this must be true by definition of the unemployment rate). Now even the scaremongerer can go unopposed, we just have no way to evaluate that claim.

Confidence level, significance level, and p -value

Significance level is the mirror image of confidence level. It is the probability that we reject H_0 even if it is correct (type-I error, see below). We normally want this number to be small. Significance level is frequently denoted by α and often chosen to be $\alpha = 0.05$.

Significance level is *not* something computed from data. It is a choice that should be done before beginning the analysis: when are we willing to say that the hypothesis is not compatible with data and reject it? If our data and model suggest that H_0 is only 10 percent likely, are we willing to reject it? What if it is only 1% likely?

Hypothesis testing is typically done by computing a *test statistic*, such as t -value or F -statistic. If H_0 is correct, the test statistic tends to have certain kind of values, often small values near zero; while if H_0 is not correct, the test statistic will have other, “more extreme” values. But as we are working with random processes, such extreme values may also occur if H_0 is correct, just it is that likely. This is the idea of p -value. p -value is probability that at least as extreme test statistic value is observed under H_0 . If we want to reject H_0 , we must have p -value smaller than the significance level, $p < \alpha$.

Example 1.4.3: Significance and p -value

Imagine we are analyzing whether smoking is related to birth weight. We collect data about mothers' smoking behavior and their babies' birth weight. We choose H_0 : “mother's smoking is not associated with birth weight”. We also have to decide a significance level, here $\alpha = 0.05$ is an appropriate choice.

A suitable test statistic is t -statistic (see [Section 1.4.3](#)). t -statistic be-

haves in the way as described above—if H_0 is correct, we expect to see mostly small values, and only rarely large values. Say, we find $t = 2.8$. From the t -value table we can find that the corresponding p -value is 0.006. This means if H_0 —there is no relationship—is correct, the probability to see a t -value 2.8 or larger is just 0.006. As this probability is less than our chosen significance level $\alpha = 0.05$, $p < \alpha$, we reject H_0 and conclude that smoking and birth weight are related.

However, if we find $t = 1.8$, the table suggests $p = 0.075$ instead. This means we have 7.5% probability to observe this large number even if H_0 is correct. As now $\alpha < p$, we cannot reject H_0 , so we cannot conclude that smoking and birth weight are related.

It is important to chose significance level before the analysis. Otherwise one may inadvertently adjust the level up if p -value turns out to be too large.

Type-I and Type-II errors

Our hypothesis testing can go wrong in two ways:

1. We reject H_0 even if it is correct. These errors are called *type-I errors* or *false positives*.
2. We fail to reject H_0 even if it is incorrect. Such errors are called *type-II errors* or *false negatives*.

The words “positive” and “negative” are easy to understand if you think about diagnosing a medical condition, e.g. cancer. Test being “positive” means it indicates the patient has cancer. Negative test means no sign of cancer. But no test is perfect and sometimes a person with no cancer will receive a positive result. This is type-I error or false positive. In an analogous fashion, if the test fails to discover cancer (it comes back negative despite the patient having cancer) then we made a type-II, or false negative error.

There is always a trade-off between type-I and type-II errors. If we pick very low confidence level, we immediately reject H_0 as soon as it does not look quite right, even if the reason is just random noise in our data. We do a lot of type-I errors but few type-II errors. In contrary, if we pick a very high confidence level, we often fail to reject H_0 even if it is wrong. We do many type-II errors but very few type-I errors. In the extreme case where we pick confidence level 0, we reject all H_0 -s, and if we pick confidence level 1, we never reject anything. The optimal choice, obviously, is somewhere in between. How we want to balance between false positives and false negatives depends on the associated costs. If false positives are cheap but false negatives expensive, we want to use a low confidence level to avoid false negatives. If the opposite is the case, we set confidence level very high.

TBD: Example

TBD: some sort of literature example where one chooses different confidence levels for different error costs.

Remember 1.4.1: Summary of the concepts

The statistical inference section above introduced a large number of concepts. Here is a summary of the most important ones.

Null hypothesis a claim about the world we want to test, usually by trying to “reject” it based on data. We reject it if it is incompatible with data. Often denoted by H_0 .

Confidence level is the probability that you do not falsely reject H_0 , often chosen to be 95%.

Significance level is the probability that you do falsely reject H_0 . It is often chosen to be 5% and denoted by α .

Test statistic a number computed from data. If H_0 is correct test statistic values are typically small and it is unlikely to find large values.

p-value probability to find test statistic at least as extreme (as large) if H_0 is correct. This means if p-value is small, H_0 can be rejected.

Confidence interval A region where the parameter of interest lies with a certain confidence level. For instance, “with 95% confidence, the population mean is between 7 and 8”.

Type-I errors (false positives) erroneously rejecting H_0 .

Type-II errors (false negatives) erroneously not rejecting H_0 .

1.4.2 Doing Statistical Inference

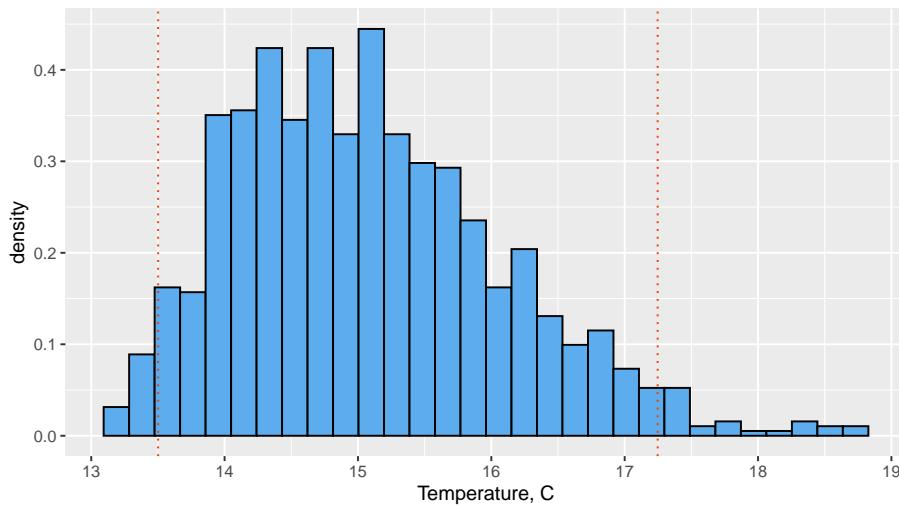
In the previous section we talked a lot about confidence intervals, confidence level, and p -value. But we did not discuss how can we actually compute those figures. The central task in this section is to do exactly that, namely to devise methods to compute the confidence intervals. We start with a somewhat trivial task, namely making statistical claims about individual random variable realizations. This helps us to devise the necessary machinery for more complex problems later.

Consider a RV X . What would be a statistically sound claim about its realization x ? Imagine X is the sequence of daily temperatures, and x will be the temperature tomorrow. Obviously, we don’t know the correct value. But we can still say something like “with 95% confidence the temperature tomorrow will be between 20 and 24 degrees”. How can we find the boundaries 20 and 24, and where is the 95% coming from?

The Simulation Approach

Let's start with the “95%” first. This is the confidence level, the mirror image of significance level which we have to decide before the analysis. In the current case the significance level means the probability that our prediction is wrong, and hence in this case our prediction will be correct in 95% of cases. So we can imagine an implicit H_0 : our prediction is correct. However, prediction here is not just a single number but a possible range.

Figure 1.6: Temperature distribution from a hypothetical weather model



All these values are uniquely determined from the distribution of X . Let's assume we are doing weather forecast. In this case the distribution is originating from the weather model we are using. Assume we run the model 1000 times and receive 1000 temperature predictions as in Figure 1.6. The distribution is somewhat right skewed, and predicts the temperature between 13 and 19 degrees with the expected value being 15.1. But other values than 15.1, such as 14 and 16 also look perfectly feasible. But which values do we consider feasible? For instance, 18 degrees seems to be unlikely, and even warmer weather seems even less plausible.

A common answer to this question is to look at the central 95% of the predictions. Central 95% of the observations means that we consider the leftmost (the coldest) 2.5% and the rightmost (the warmest) 2.5% predictions to be unlikely. Given the 1000 predictions, we can just remove the 25 coldest and 25 warmest temperatures, and we are left with the central 95% of the predictions. This amounts to computing the 0.025-th and 0.975-th sample quantiles. In case of this sample the corresponding quantiles are 13.5 and 17.25 degrees (dotted

vertical lines on the figure). This interval, [13.5, 17.25] is called *confidence interval* (CI), more precisely 95% confidence interval for the predicted temperature. So 95% CI is the interval that contains the actual value with 95% probability, given that our weather model is correct. The values in this range are considered likely, and those outside this range are considered unlikely. So we may say that +16 degrees will be quite likely but +18 will be unlikely. We can reject H_0 : “temperature will be over 18C” at 5% significance level.

As the 95% CI are only correct 95% of time, one may be tempted to improve on the type-I error and report 99% or 99.9% CI instead. This is fair, but unfortunately the result will be less informative. If I say that temperature tomorrow will be between -100 and +100C then I am correct 100% of time (well, on Earth at least). However, such a prediction does not help us to make any decisions, e.g. what to wear tomorrow. There is a trade-off between providing precise estimates and avoiding errors, and sometimes wrong information is often better than always-correct claim devoid of any information. Sometimes one can compute the optimal confidence level by considering the cost of type-I (false positives) and type-II (false negatives) errors, and choosing a confidence level that leads to the smallest overall loss.

But why did we select the central 95% interval as our confidence region? Why not leave out the largest 5% and pick the smallest value till 0.95-th quantile as the confidence region? Or the other way around? And what about picking both extremes and leaving a narrow 5% gap in the middle? There are a few reasons for this, some of those more theoretical, some more practical.

- First, we are usually interested in a confidence region that is as concentrated as possible: we want the 95% of possible outcomes to have a small error margin. The narrower my temperature forecast, the better idea you have what to wear tomorrow. The obvious choice is to pick the region on the histogram with highest chances and not to leave a gap in the middle where the values are most likely.
- Second, in typical applications the distribution is symmetric and unimodal (usually close to normal). Hence both tails are thin and similar, and it makes sense to cut the 2.5% of observations in both tails if we want to get the most concentrated confidence region. But if it is not symmetric, we may actually choose a different percentages in different tails.

TBD: Example of asymmetric CI

If the distribution is bimodal, we may actually want to leave a gap in the middle.

TBD: Example of bimodal CI

TBD: Example to compute the loss and CI based on the loss

Theoretical Confidence Intervals

If we know the stochastic process that generates our data then we can often compute the distribution of the corresponding RV and find the quantiles from theoretical considerations. Sometimes this can be calculated easily (e.g. for

uniform distribution), sometimes one has to consult tables (e.g. for normal and t -distribution). For instance, for standard normal distribution, the lower 2.5% quantile $q_{0.025} = -1.96$ and the upper 2.5% quantile is $q_{0.975} = 1.96$ ($q_{1-\alpha} = -q_\alpha$ because standard normal is symmetric). For a general normal with expectation equal to μ and variance σ^2 , one has $q_{0.025} = \mu - 1.96\sigma$ and $q_{0.975} = \mu + 1.96\sigma$.

Example 1.4.4: Confidence intervals for human height

Look at the fathers' and sons' height data, in particular sons' heights. The distribution as a histogram is shown on the figure below, overlapped with approximated normal density curve (red).

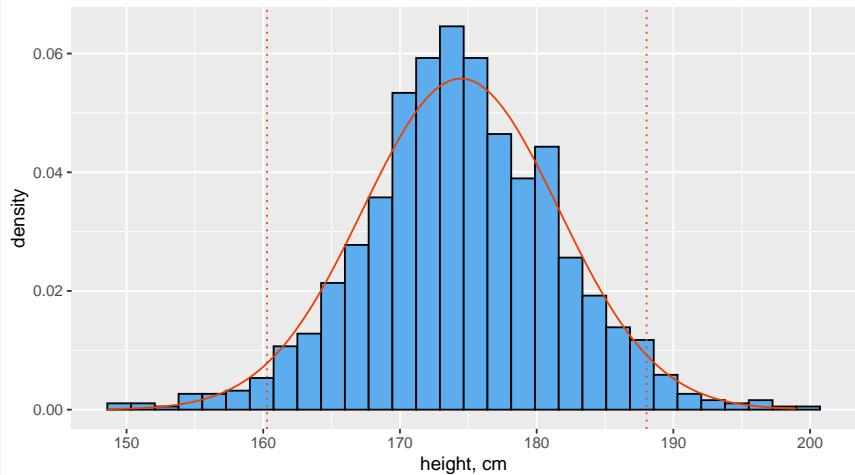


Figure 1.7: Distribution of son's height in fathers-sons data. It is well approximated by normal distribution (red line) with mean $\mu = 174.5$ and standard deviation $\sigma = 7.2$, this is common for measures of adult animals. Dotted vertical lines are the boundaries of the 95% confidence intervals.

As evident from the figure, most of observations are concentrated around the mean 174.5 cm, roughly in the interval of 160–190. More precisely, the lower 2.5% observations are shorter than 160.3 and the upper 2.5% of observations are taller than 188 cm. This means that the middle 95% of the observations fall into the interval [160.3, 188]. This is the 95%-confidence interval (CI) for sons' height. If data were exactly normally distributed with a similar mean and standard deviation, the corresponding theoretical quantiles were 160.4 and 188.5. As one can see, the deviations from the theoretical values are rather small. Normal distribution is a good approximation for human height.

Now we move to a more important task, namely to find a confidence interval

of sample mean. Remember that the variance of mean

$$\text{Var } \bar{X} = \frac{1}{N} \text{Var } X \quad (1.4.1)$$

where \bar{X} denotes the mean of a sample of size N . It can also be shown that sum of independent normals is a normal too: if $X \sim N(\mu, \sigma^2)$, we have that

$$\bar{X} \sim N\left(\mu, \frac{\sigma^2}{N}\right). \quad (1.4.2)$$

Accordingly, the 95% confidence intervals of a mean of normals is

$$\left[\mu - 1.96 \frac{\sigma}{\sqrt{N}}, \mu + 1.96 \frac{\sigma}{\sqrt{N}} \right]. \quad (1.4.3)$$

Example 1.4.5: Sample mean of sons' height

Let's look again at the sons' height data. As a reminder, the mean sons' height is 174.5 and it's standard deviation is 7.2. Now we take 1000 times a random sample of 4 sons and calculate their mean height. In this way we get 1000 sample means, and we plot the results on a similar histogram as in Example 1.4.4. Note that we have to take many samples, in order to compute many sample means, and be able to plot their distributions.

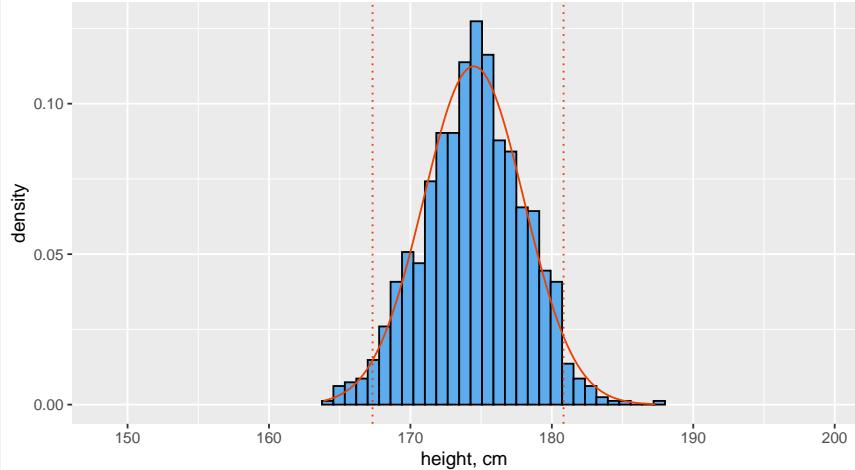


Figure 1.8: Distribution of mean height of four sons. As individual heights, the distribution of means are well approximated by normal (red line) with mean $\mu = 174.5$ and standard deviation $\sigma = 3.5$. Dotted vertical lines are the boundaries of the 95% confidence intervals.

The figure is plotted in a similar scale as Figure 1.7. It reveals that the result is well approximated with a normal density, however this time the spread

is narrower. The sample of means has mean $\mu = 174.5$, almost exactly the same as the sample of individual heights $\mu = 174.5$. Its standard deviation $\sigma = 3.5$ is only half of that for the heights, $\sigma = 7.2$. This is to be expected as the standard deviation of a sample of four should be $1/\sqrt{4} = 1/2$ of the standard deviation of individual values. Accordingly, both the empirical confidence intervals [167.3, 180.8] and theoretical confidence intervals are only half as wide, [167.5, 181.4].

TBD: t -distribution when we don't know the mean

TBD: CLT, and the fact we don't have to start with a normal

1.4.3 Comparing Distributions

A common task is to compare distribution of certain variables across different groups, datasets or time periods. For instance, can we say that police treats African-Americans differently than white Americans? We can easily compute, for instance, the probability that either group is stopped by the police and compare those figures, but aren't those numbers just a statistical blip, just random noise that will go the other way next day?

In order to answer this, and other similar questions we have to approach it through formal statistical hypothesis testing. We can assume H_0 : both groups are treated equally; and thereafter look at the data, and see if we can reject the hypothesis. Below, we walk through such an analysis and test the sample means using t -test. Note that even if mean values of two samples are similar it does not all the sample properties are equal. For instance, the other distribution may have different variance.

Comparing Means: Is Smoking Ban Associated with Less Smoking?

In the Western World, regulations about smoking have become increasingly restrictive over recent decades. In particular, smoking is banned in many common indoor areas, such as restaurants or workplaces. *SmokeBan* data (AER package in R) provides data for 10,000 workers who work either on a workplace with or without smoking ban, and who are either smokers or non-smokers. A simple analysis suggests that 6098 workplaces with smoking bans are associated with a lower percentage of smokers, 21.2 percent, while 28.96 percent of workers smoke on 3902 workplaces without such a ban. So the difference is 7.76 percentage points. As we have quite a large sample—10,000, it suggests that the effect is real. But is it true? Maybe it is just a statistical fluke and will be reversed in a next survey? Let us answer this question first by simulations, and thereafter by stock t -test.

This task is about comparing two samples: workers on jobs with workplace smoking ban, and on jobs with no such ban. This is called *unpaired* data, there is no obvious correspondence between individuals across the samples. We can answer the question by testing H_0 : average number of smokers in both types of workplaces is the same.

First introduce some notation: let S_i denote the smoking indicator of individual i (0: does not smoke, 1: smokes). Let \mathcal{B} denote the set of those individuals who are working at smoking-ban workplaces, and \mathcal{N} the set of those who work at no-ban workplaces. Define

$$\bar{S}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} S_i \quad \text{and} \quad \bar{S}_{\mathcal{N}} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} S_i \quad (1.4.4)$$

for average smoking probability at smoking-ban and no-smoking-ban workplaces. We will test the hypothesis $H_0 : S_{\mathcal{B}} = S_{\mathcal{N}}$ or equivalently, $H_0 : S_{\mathcal{B}} - S_{\mathcal{N}} = 0$. Assume H_0 holds, i.e. there is not real difference between smoking-ban and no-smoking-ban workplaces. In that case the expected values $\mathbb{E} S_{\mathcal{B}} = \mathbb{E} S_{\mathcal{N}} \equiv \mathbb{E} S$, the overall smoking probability at workplace. We can approximate the latter by sample mean as

$$\bar{S} = \frac{1}{N} \sum_i S_i \quad (1.4.5)$$

where the sum is taken over the whole dataset, in the *SmokeBan* dataset the corresponding value is $\bar{S} = 24.23$ percent.

It is important to realize that the data is generated through a Bernoulli process: it is a discrete process with only two outcomes (smoker or non-smoker), where “smoker” occurs with probability \bar{S} .

TBD: one-sided vs two-sided test

First let's simulate the results. We can create synthetic data by creating 3902 random workers without smoking ban and 6098 ones with smoking ban, both generated using the Bernoulli process with probability 24.23 (this is because according to H_0 we assume they share the same parameter value). Thereafter we compute the mean smoking tendency among our synthetic workers for both types of workplaces. As both sets of workers were created through an identical process, the difference can only be attributed to the random noise. Finally, we repeat the simulations many times, and see how often we get as large a difference as we see in data. If the difference is rare enough (say, it occurs less than in 5% of cases), we can reject H_0 at the corresponding confidence level (5% level in this example).

Table 1.4 shows five example simulations. In all cases the the simulated difference is much smaller than the actual difference 7.76 percentage points. This supports our intuition, telling that the sample is large enough to distinguish a 7-percentage point difference. But this was only 5 simulations. Do the results still hold if we run more trials? Indeed. Figure 1.9 shows a histogram of 10,000 such simulations. The maximum value obtained in these simulations was 3.33 (in absolute value), well below the observed 7.76. For the reference we also record that the average difference over all simulations is 0 and its standard deviation is 0.885 percentage points. We can conclude that chances to observe such a value under H_0 are extremely low, less than 1 in 10^4 . Hence we can reject H_0 at 5% confidence level (we can also reject it at 0.01% confidence level). This

Table 1.4: Simulated smoking habits (percent of smokers) at smoking-ban/no-smoking-ban workplaces for 5 random simulations. The difference in data is 7.76 percent. We can see that simulated values in this example are much smaller than what is visible in the data.

	Ban (pct)	Non-ban (pct)	difference (pct pt)
1	24.96	24.68	0.28
2	24.65	24.88	0.22
3	24.32	24.73	0.41
4	23.76	23.98	0.22
5	24.88	23.70	1.19

suggests that H_0 is not correct. But note that we cannot say it is *incorrect*, we can strictly speaking only claim that it is extremely *unlikely*.

This is about as far as we can get through statistical methods. We can say a hypothesis is “unlikely” at different confidence levels, but we cannot say it is “wrong”.

TBD: Example with insignificant results, continuous data

TBD: Exercise

Although simulation approach we did above is intuitive, it is often too complex way. Fortunately we can replace it with a simple formula. First, note the differences in Figure 1.9 are normally distributed. This is a direct result of Central Limit Theorem, and the fact that sum (or difference) of two normal RV-s is normal:

- Both \bar{S}_B and \bar{S}_N are sums of i.i.d random values, and hence under H_0 they are both approximately distributed as (from CLT)

$$\bar{S}_B \sim N\left(\mathbb{E} S, \frac{\sigma^2}{\sqrt{N_B}}\right) \quad \text{and} \quad \bar{S}_N \sim N\left(\mathbb{E} S, \frac{\sigma^2}{\sqrt{N_N}}\right) \quad (1.4.6)$$

where σ^2 is variance of S . We don’t know $\mathbb{E} S$ but we can approximate it with \bar{S} .

- Moreover, as S follows the Bernoulli process, we can immediately write $\sigma^2 = \mathbb{E} S(1 - \mathbb{E} S)$. Alternatively, we can also just compute the sample variance of S .
- Finally, the difference $d = \bar{S}_B - \bar{S}_N$ is difference of two independent normals, and hence normally distributed with mean 0 and variance $\sigma_d^2 = \frac{\sigma^2}{N_B} + \frac{\sigma^2}{N_N}$.

This permits us to immediately find the confidence interval of d under H_0 using the properties of normal distribution: the CI, related to H_0 is $[-t_{cr} \cdot \sigma_d, t_{cr} \cdot \sigma_d]$. t_{cr} is the critical value, for 95% confidence level it is 1.96. When we plug the numbers into the formula, we get the variance

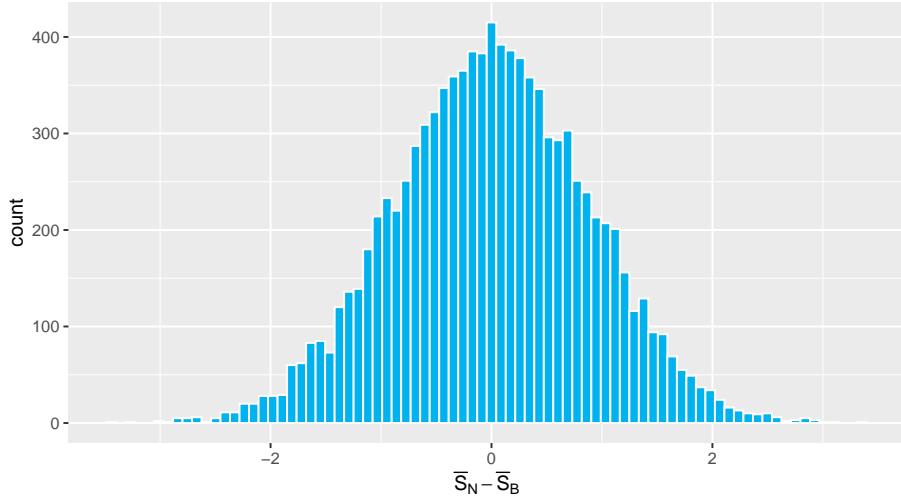


Figure 1.9: Histogram of 10^4 simulation runs. One can see that the most common values are close to 0, values above 3 are extremely rare. The maximum observed value is 3.47 (in absolute value), well below 7.76 in data. We can conclude that observing such a big difference under H_0 is extremely rare.

$$\begin{aligned}\sigma_d^2 &= \frac{\sigma^2}{N_B} + \frac{\sigma^2}{N_N} = \bar{S}(1 - \bar{S}) \left(\frac{1}{N_B} + \frac{1}{N_N} \right) = \\ &= 0.2423 \cdot (1 - 0.2423) \left(\frac{1}{6098} + \frac{1}{3902} \right) = 7.72 \times 10^{-5} \quad (1.4.7)\end{aligned}$$

or

$$\sigma_d = 0.00878 \quad \text{or} \quad 0.878 \quad \text{percentage points.} \quad (1.4.8)$$

This is almost exactly the value we received above through simulations. Now we can compute the 95% CI as

$$[-1.96 \cdot 0.878, -1.96 \cdot 0.878] = [-1.722, -1.722] \quad (\text{pct points}). \quad (1.4.9)$$

The observed difference is way outside of this range, so we can reject H_0 at significance level of 0.05.

As we could see, both approaches resulted in exactly same conclusion. This is to be expected: one approach used explicit simulations to come to the conclusion, the other approach implicitly did the same. Just instead of computing all the random numbers, we used the theoretical results from CLT, Bernoulli distribution, and sum of normals.

Finally, although we can reject H_0 : smoking behavior does not differ with the smoking ban, we cannot tell that smoking ban “causes” workers to smoke less.

It is possible that the observed effect is due to reverse causality (few smokers at workplace make it feasible to introduce the ban) or confounding factors are possible (see more in [Chapter 5](#)). Our conclusion is pure correlational: smoking ban and smoking are “associated”.

TBD: small sample size and t -distribution

Chapter 2

Linear Algebra

In these notes we use vectors and matrices for two main purposes: to hold data, and to simplify algebra—linear algebra (LA) formalism tremendously simplifies algebra and computations for certain types of tasks.

This section covers the basic LA concepts we need. As our usage of LA is heavily matrix-oriented, we cover vectors only superficially. Later we typically assume that vectors are just special matrices with only a single column (or a single row). In a similar fashion we do not use inner and outer product concepts, we treat both these products as just matrix products between row- or column matrices.

2.1 Why Linear Algebra in Machine Learning

The concepts “vector” and “matrix” have (at least) two related meanings. One is a type of data storage, for data that is arranged in one dimension (vector) or in two dimensions (matrix or data frame). The other meaning is vectors and matrices in mathematical, in linear algebra sense. These are numbers, stored in an 1-D or 2-D structure, exactly like the storage structures. However, linear algebra defines a large number of certain mathematical operations on these structures. Here we are interested in the mathematical properties of these objects, but both types are closely related, for instance, many popular computer libraries that support vectors and matrices also implement the corresponding mathematical operations.

As it turns out, a large number of operations we do with ordinary numbers, such as addition, multiplication and inverse generalize easily to matrices as matrix addition, matrix multiplication, and inverse matrix. More importantly, many statistical¹ problems generalize from univariate to multivariate versions using exactly these operations (and we stress that machine learning is in many

¹Here we are mainly concerned with statistics, but the same is also true for many physics and engineering problems.

ways a branch of statistics). For instance, instead of univariate normal distribution, we can use multivariate normal to describe distribution of correlated values. This allows us to handle multivariate problems in a dimension-agnostic way, just deriving, writing, and coding formulas for general N-dimensional case.

For instance, the way to solve linear regression models in any dimensions can be written as

$$\hat{\beta} = (\mathbf{X}^T \cdot \mathbf{X})^{-1} \cdot \mathbf{X}^T \cdot \mathbf{y}. \quad (2.1.1)$$

Even if one does not know what do these symbols and operations mean, one can see that the formula is rather simple. As efficient software implementations of matrix operations are widely available, this formula can almost literally converted into computer code.

As linear algebra is ubiquitous in science and engineering, there exist dedicated well-optimized libraries, and even dedicated hardware to speed up certain linear algebra processes. For instance, ordinary graphics cards with thousand of simple computing cores, originally designed for computer games, are optimized for matrix multiplication because this is how one rotates 3-D scenery. This makes linear algebra a method of choice when implementing and using related methods.

Linear algebra is also the method of choice when presenting statistical methods. Every source, besides the very beginner-oriented texts uses linear algebra, and assumes the reader is familiar enough with the basic concepts. In this sense it is a central component of machine learning language.

Below we walk through the basics of vectors and matrices with the focus on matrix multiplication, inverse matrix, and metric distance.

2.2 Vectors and Vector Spaces

This section briefly introduces vectors, vector spaces, and a few related concepts (in particular *norm* and *distance*) in a non-matrix way. Although later we rely heavily on matrix notations, the concepts in this section do not require matrix formalism.

2.2.1 Vectors

What Are Vectors

Vectors are ordered collections of elements, for our purpose they are just sequences of numbers. Normally we denote vectors by bold lower case letters, so an example vector is $\mathbf{x} = (1, 2, 3)$. Here the vector \mathbf{x} contains three *elements* (also called *components*), 1, 2, and 3, in this order. Order matters, $(1, 2, 3) \neq (2, 1, 3)$. We denote vector components with the same letter as the vector itself, just not in bold, and supplied with the element index. For instance, given the vector \mathbf{x} above, we have $x_1 = 1$ and $x_3 = 3$. We can also use symbols to denote vector elements so another vector example is $\beta = (\beta_1, \beta_2, \beta_3)$. Here we only work with numeric vectors, i.e. with vectors where all elements are

numbers. But the components do not have to be numeric, they may be all kind of objects, including letters, texts, images, functions and other vectors.

An important property of vector is its number of elements, called *dimension*.² Our example vectors \boldsymbol{x} and $\boldsymbol{\beta}$ are 3-dimensional. 3-D vectors are widely used to describe coordinates in our 3-space, e.g. in 3-D computer games. But our vectors can be of any (positive) dimension, including 1-dimensional (just single objects like individual numbers). They may also have very high dimensionality, for instance color images can be stored as vectors of millions of elements. In theoretical applications we can also work with infinite-dimensional vectors.

From our perspective, one of the most important roles of vectors is to hold data. For instance, consider the dataset about 50 U.S. States. A few first observations of it look like:

Population	Income	Illiteracy	LifeExp	Murder	HSGrad	Frost	Area
3615.00	3624.00	2.10	69.05	15.10	41.30	20.00	50708.00
365.00	6315.00	1.50	69.31	11.30	66.70	152.00	566432.00
2212.00	4530.00	1.80	70.55	7.80	58.10	15.00	113417.00
2110.00	3378.00	1.90	70.66	10.10	39.90	65.00	51945.00

We can describe the data points (observations) of this dataset as

$$\begin{aligned} \boldsymbol{x}_1 &= (3615, 3624, 2.1, 69.05, 15.1, 41.3, 20, 5.0708 \times 10^4) \\ \boldsymbol{x}_2 &= (365, 6315, 1.5, 69.31, 11.3, 66.7, 152, 5.66432 \times 10^5) \\ \boldsymbol{x}_3 &= (2212, 4530, 1.8, 70.55, 7.8, 58.1, 15, 1.13417 \times 10^5). \end{aligned} \quad (2.2.1)$$

When stacking these data vectors horizontally on top of each other, we get a data matrix (design matrix). Alternatively, we can look at individual variables as vectors, in that case we have

$$\begin{aligned} \boldsymbol{v}_1 &= (3615, 365, 2212, 2110, 21198, \dots) \\ \boldsymbol{v}_2 &= (3624, 6315, 4530, 3378, 5114, \dots) \\ \boldsymbol{v}_3 &= (2.1, 1.5, 1.8, 1.9, 1.1, \dots) \\ &\dots \end{aligned} \quad (2.2.2)$$

Another comment about the notation: here we are using subscript index not to refer to individual components, but to refer to different vectors. If we refer to an individual component, we can add another index, e.g. $v_{11} = 3615$ and $x_{23} = 1.5$ in the example above. So the first component refers to the vector, and the last one to the component. Note that we denote components (just numbers) with

²We encounter the concept *dimension* in two different meanings. Here it is the dimension of the underlying vector space, or the number of elements in the vector. But often one refers to all vectors as 1-D objects, contrary to matrices that are 2-D objects. This is because vectors are like a 1-D string of numbers and have only a single length, while matrices resemble 2-D rectangle of numbers and have both length and width. One has to understand which is meant by dimension a particular case.

ordinary font while vectors with index are in bold! However, there is a variety of notation used in the literature.

Exercise 2.2.1: Vector dimension

What is dimension of vectors \mathbf{x}_i in (2.2.1) and \mathbf{v}_i in (2.2.2)?

Vector Addition and Scalar Multiplication

If we use vectors just to store data, we may not really need to do any computations with these. Most of the linear algebra is based on two simple operations: addition and multiplication by scalar. With *scalar* we mean here a single number that is not a vector. We denote sum of two vectors by $\mathbf{x} + \mathbf{y}$, and multiplication by scalar α as $\alpha\mathbf{x}$.

When talking about addition and scalar multiplication, we normally mean just the ordinary mathematical operations. But these do not have to be the common addition and multiplication, these can be all kind of operations as long as they satisfy a few axioms, including

1. there is a special element, null vector $\mathbf{0}$, so that $\mathbf{x} + \mathbf{0} = \mathbf{x}$ for all \mathbf{x} .
2. multiplying any vector with scalar 0 will result in null vector: $0\mathbf{x} = \mathbf{0}$ for all \mathbf{x} .
3. multiplying any vector with scalar 1 will retain the original vector: $1\mathbf{x} = \mathbf{x}$ for all \mathbf{x} .
4. the operations follow certain distributive laws: $\alpha(\mathbf{x} + \mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y}$.

For the vector operations we look here, scalar multiplication is performed by multiplying all vector components by the scalar:

$$\alpha\mathbf{x} = \alpha(x_1, x_2, \dots, x_K) = (\alpha x_1, \alpha x_2, \dots, \alpha x_K). \quad (2.2.3)$$

In a similar fashion, vector addition is performed by adding the corresponding components of the vectors:

$$\begin{aligned} \mathbf{x} + \mathbf{y} &= (x_1, x_2, \dots, x_K) + (y_1, y_2, \dots, y_K) = \\ &= (x_1 + y_1, x_2 + y_2, \dots, x_K + y_K). \end{aligned} \quad (2.2.4)$$

As is obvious from this definition, the vectors must have same dimension (here K) to be possible to add those.

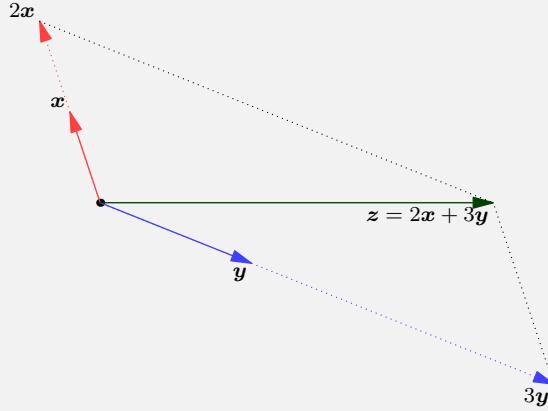
Example 2.2.1: Graphical way to add vectors

Consider two vectors, $\mathbf{x} = (-1, 3)$ and $\mathbf{y} = (5, -2)$. Let's compute $\mathbf{z} = 2\mathbf{x} + 3\mathbf{y}$.

When we just multiply and sum the components we get

$$\mathbf{c} = 2 \cdot (-1, 3) + 3 \cdot (5, -2) = (13, 0).$$

This operation can be represented graphically as



The solid red and blue arrows depict the original vectors x and y , the corresponding dotted arrows are $2x$ and $3y$, and long green solid arrow is their sum $z = 2x + 3y$.

Exercise 2.2.2: What is the capital of France?

Word embeddings is a way to describe words as numeric low-dimensional vectors (typically 100-300 dimensions).^a So in 100-component example the embedding for word *Berlin* (see below in the table) looks like $e(\text{Berlin}) = (-0.562, 0.630, -0.453, -0.299, -0.006, \dots)$. All these numbers correspond to different components, but unfortunately the components are not interpretable in general. Embeddings are computed based on words' co-occurrence in texts. As similar words tend to occur in similar contexts, they tend to have similar embedding vectors. More interestingly, one can also do certain mathematical operations with embedding vectors. For instance, it is well known that $e(\text{king}) - e(\text{man}) + e(\text{woman}) \approx e(\text{queen})$ where $e(\text{word})$ is the embedding vector of *word*.

Below is the first five components (out of 100) for *Berlin*, *Germany*, *France*, and *Paris*^b (the rest of 95 components are not shown).

word	1	2	3	4	5
Berlin	-0.562	0.630	-0.453	-0.299	-0.006
Germany	0.194	0.507	0.287	0.132	-0.281
France	0.605	-0.678	-0.436	-0.019	-0.291
Paris	-0.074	-0.855	-0.689	-0.057	-0.139

Compute $e(\text{Berlin}) - e(\text{Germany}) + e(\text{France})$ and check how close do you get to $e(\text{Paris})$.

^aWhile a 100-D vector may not sound like low-dimensional, typical vocabularies contain between 10,000 and one million different words. Using one-hot encoding would

result in the corresponding number of dimensions, so a few hundred components is much less than that.

^bThe data, *glove.twitter.27B.100d.txt*, based on 2 billion tweets, can be downloaded from [Stanford NLP project](#) website.

All vectors that can be formed from certain elementary vectors using these two operations form a *vector space*. X . For our purpose it is simply a set of all relevant vectors. For a set to be valid vector space, it must be *closed* with respect to these operations. This means that whatever elements of X and real numbers we take, all their sums and products must also be in X . Formally,

- if $\mathbf{x} \in X$, $\mathbf{y} \in X$ and $\mathbf{z} = \mathbf{x} + \mathbf{y}$, then $\mathbf{z} \in X$.
- if $\mathbf{x} \in X$ and $\mathbf{z} = \alpha\mathbf{x}$, then $\mathbf{z} \in X$ for each $\alpha \in \mathbb{R}$.

An intuitive and easy-to-understand example of vector space is \mathbb{R}^2 . In \mathbb{R}^2 vectors are just pairs of real numbers, addition is defined as adding the corresponding components of vectors, and scalar multiplication is defined as multiplying all vector components with the scalar. In this case the scalar multiplication is equivalent to stretching (or squeezing) the vectors while retaining their direction, and vector addition is equivalent to parallel shift of one vector to the “end” of the other one. As a special case, negative of the vector \mathbf{x} , $-\mathbf{x}$, is just the original vector pointing in the opposite way.

Example 2.2.2: Application of 2-D vector space \mathbb{Z}^2

Imagine you are designing a 2-D computer game. We can choose the coordinates in different way, a natural choice is to take the origin $(0, 0)$ to be the bottom-left corner of the screen, and count the horizontal coordinates right, and vertical coordinates up. We also specify the vectors as (horizontal, vertical),^a or (x, y) . As the objects on our screen can only be at certain pixels, we are only interested in integer coordinates, so $x, y \in \mathbb{Z}$ where \mathbb{Z} is the set of all integers, or $(x, y) \in \mathbb{Z}^2$ where \mathbb{Z}^2 is the set of all pairs of integers.

Your player is located at coordinates $\mathbf{p} = (194, 33)$. She shoots an arrow upward that moves 10 vertical pixels per frame. Where is the arrow after 10 frames?

We can write the arrow’s 2-D speed vector as $\mathbf{v} = (0, 10)$. $v_1 = 0$ as the arrow does not move horizontally at all, and $v_2 = 10$ means that it moves up by 10 pixels in one time unit (here the rendering frame). In 10 frames the arrow moves $10\mathbf{v} = (0, 100)$ and hence is located at $\mathbf{p} + 10\mathbf{v} = (194, 133)$. The location at arbitrary frame $t > 0$ can be written as $\mathbf{a}(t) = \mathbf{p} + t \cdot \mathbf{v}$. Here \mathbf{a} , \mathbf{p} , and \mathbf{v} are vectors, locations on screen, and speed on screen respectively; and t is scalar, the frame count from the moment arrow was released.

^aAlghouth *horizontal*, *vertical* may sound an obvious and trivial choice, it conflicts with the traditional way of displaying matrix indices, namely vertical, horizontal. Even more, vertical elements are traditionally counted from top-left corner down.

One can also easily visualize and understand the 3-D vector space \mathbb{R}^3 . Higher-dimensional spaces \mathbb{R}^K are still straightforward, but cannot be visualized.

Vector Space: Base and Linear Independence

The definition of vector space above—closedness with respect to scalar multiplication and vector addition—suggests that all vectors we can form from certain *base vectors* using only vector addition and scalar multiplication form a vector space. So if the base vectors are \mathbf{a} and \mathbf{b} , the the vector space is a set of all possible vectors

$$\mathbf{c} = \alpha \cdot \mathbf{a} + \beta \cdot \mathbf{b} \quad \alpha, \beta \in \mathbb{R} \quad (2.2.5)$$

Figure 2.1 depicts on such example. Two base vectors \mathbf{a} (red) and \mathbf{b} (blue) can be used to compose \mathbf{c} and \mathbf{d} . See also Exercise 2.3.2 about how to compute the corresponding α and β . Such constructs, sums of vectors, multiplied by scalars, are called *linear combinations*. So vector space is made of all possible linear combinations of the base vectors.

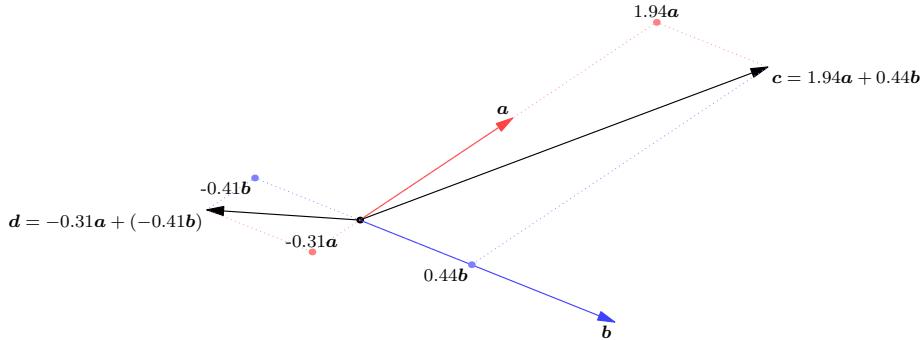


Figure 2.1: Vector space: all vectors on a plane can be computed as a linear combination of two base vectors, here \mathbf{a} (red) and \mathbf{b} (blue). The dotted red and blue arrows show which linear combinations are needed to create vectors \mathbf{c} and \mathbf{d} (black).

The figure shows only two base vectors. We can easily add another one but it turns out to be unnecessary—on the plane depicted on Figure 2.1, two base vectors are sufficient. However, removing either \mathbf{a} or \mathbf{b} will collapse the plane: there is no way to cover a plane using a single vector only. This property of a vector space—how many base vectors are needed to cover the space—is called *dimension* of vector space. Obviously, a single base vector can only cover a line. We can multiply \mathbf{b} with any number but the result will always stay in the line defined by \mathbf{b} . So vector space made of a single base vector, 1-dimensional space, is a line. In an analogous fashion, every linear combination of vector \mathbf{a} and \mathbf{b} will stay on their plane, there is no way to describe a point outside of the plane using only these two vectors. We need a third one that points out of the plane. That would result in a 3-D vector space. These examples are easy to visualize and

understand as our space is 3-D. Mathematically we can easily describe higher dimensional spaces but our imagination fails as soon as we move from three to four dimensions. But even when we cannot imagine high-dimensional space, it serves as an useful tool when working with high-dimensional vectors.

Base vectors and the dimension of vector space are closely related to linear independence. A set of vectors is *linearly independent* if one cannot compute one of these vectors from the others (by using only scalar multiplication and vector addition). Formally, we say that vectors $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n$ are linearly independent if and only if

$$\begin{aligned} \alpha_1 \mathbf{a}_1 + \alpha_2 \mathbf{a}_2 + \cdots + \alpha_K \mathbf{a}_K &= 0 \\ \Updownarrow \\ \alpha_1 = \alpha_2 = \cdots = \alpha_K &= 0 \end{aligned} \tag{2.2.6}$$

It is easy to see that this formal definition is equivalent to the informal claim above. We can easily express one vector in (2.2.6), for instance \mathbf{a}_1 , using other vectors as

$$\mathbf{a}_1 = \frac{\alpha_2}{\alpha_1} \mathbf{a}_2 + \frac{\alpha_3}{\alpha_1} \mathbf{a}_3 + \cdots + \frac{\alpha_K}{\alpha_1} \mathbf{a}_K. \tag{2.2.7}$$

But this is only possible if $\alpha_1 \neq 0$. Hence for us to be able to express at least a single vector in this way, we need at least one α to be non-zero. And by definition, this means our vectors are not linearly independent. In that case it is often said they are *linearly dependent*.

Example 2.2.3: Are these vectors linearly independent?

Let us test if vectors $\begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 4 \end{pmatrix}, \begin{pmatrix} 5 \\ 6 \end{pmatrix}$ are linearly independent.

We can express one vector as a linear combination of others

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} = 2 \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} - \begin{pmatrix} 5 \\ 6 \end{pmatrix},$$

or alternatively write

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} - 2 \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} + \begin{pmatrix} 5 \\ 6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.2.8}$$

Hence these vectors are not linearly independent.

Exercise 2.2.3: Are these vectors linearly independent?

Are vectors $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}, \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix}$ linearly independent?

2.2.2 Norm and Distance

Many machine learning methods need to compute distance between data points. For instance, nearest-neighbors method is concerned with the “closest” data points to the one we want to analyze, while clustering methods make clusters out of observations that are “close”. As we typically think about data points as vectors, all such methods need to compute distance between vectors. We first discuss a generalization of vector length³ called *norm*. Thereafter we define distance between two vectors by just computing the norm of their difference.

Norm

Let us start with the ordinary geometry. Take the example of a 2-dimensional vector on plane. If the vector is given as $(1, 1)$, what is its length? The answer is $\sqrt{2} \approx 1.414$ (see Figure 2.2). The length of the diagonal of a square with unit length sides is 1.414. More generally, we can use the Pythagorean theorem and write the length of vector (x_1, x_2) as $\sqrt{x_1^2 + x_2^2}$. This formula immediately generalized to 3-D space \mathbb{R}^3 , the length of 3-vector $\mathbf{x} = (x_1, x_2, x_3)$ is

$$\sqrt{x_1^2 + x_2^2 + x_3^2}. \quad (2.2.9)$$

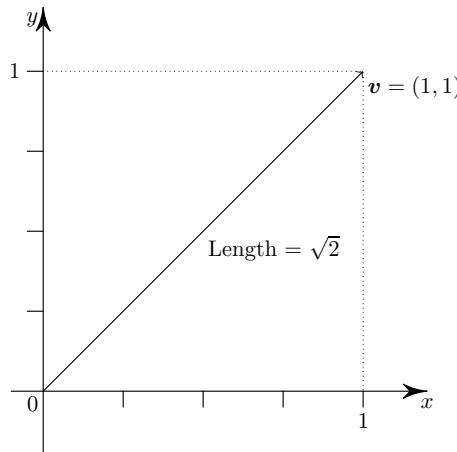


Figure 2.2: Vector \mathbf{v} has both components, v_x and v_y equal to one. From Pythagorean theorem, its length is $\sqrt{2}$. Generalized “length” of a vector is called norm and denoted by $\|\mathbf{v}\|$, in this case $\|\mathbf{v}\| = \sqrt{2}$.

This is our most obvious understanding of length in 3-space. For instance, the box with side lengths of 1, 2 and 2 has diagonal of length $\sqrt{1^2 + 2^2 + 2^2} = 3$. We can generalize the same concept of length further into K -dimensional space \mathbb{R}^K as

$$\sqrt{\sum_{i=1}^K x_i^2} \quad (2.2.10)$$

but unfortunately our imagination cannot keep up when we move above three dimensions.

This “obvious” concept of length is called *Euclidean norm*. We intuitively think in Euclidean terms as this is how the 3-D space we live in is “made”.

³As *length*, here we mean length as length in space, not the number of components (we call the latter vector’s *dimension*).

However, there are many other ways to define length, and sometimes the conventional approach is not the best one. Those more general concepts of are called “norm”, this is why we call the Euclidean length Euclidean *norm*.

Norm of vector \mathbf{x} is typically denoted by $\|\cdot\|$, is a generalization of the concept of length: it is a function that assigns a non-negative real number to every vector. But one cannot just assign each vector an arbitrary number. Valid norm must satisfy three conditions:

1. $\|\mathbf{x}\| \geq 0$; $\|\mathbf{x}\| = 0$ if and only if $\mathbf{x} = \mathbf{0}$.

Norm must be positive, only null-vector is zero norm.

2. $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (triangle inequality)

3. $\|\alpha\mathbf{x}\| = |\alpha| \cdot \|\mathbf{x}\|$ (multiplication be scalar).

In machine learning applications we are often much sloppier, and use measures of “length” that are not valid metric norms. For instance, if our task is to rank texts based on the vocabulary similarity then we can easily violate the assumption 1 (see more in Section 3.2.2 Cosine Similarity).

L_p norm

A rather straightforward generalization of Euclidean norm is L_p norm, also called *Minkowski norm*. It is defined by replacing “2” in the formula for Euclidean norm by a positive parameter p , and the norm is often denoted by adding a small p -subscript to the norm symbol:

$$\|\mathbf{x}\|_p = \left[\sum_{i=1}^K |x_i|^p \right]^{1/p}, \quad p > 0. \quad (2.2.11)$$

Example 2.2.4: L_3 norm of vector (1,1)

Let us compute L_3 norm of $\mathbf{v} = (1,1)$, depicted in Figure 2.2. Remember, its Euclidean, L_2 , norm is $\sqrt{2} \approx 1.414$. Its L_3 norm is

$$\|\mathbf{v}\|_3 = \left[\sum_{i=1}^K |x_i|^3 \right]^{1/3} = (|1|^3 + |1|^3)^{1/3} = \sqrt[3]{2} \approx 1.26.$$

Obviously, Euclidean norm is just L_2 norm. There are two other interesting and popular special cases: Manhattan norm and Chessboard norm.

Manhattan norm (also *taxicab norm*) is L_1 defined as

$$\|\mathbf{x}\|_1 = \sum_{i=1}^K |x_i|. \quad (2.2.12)$$

So Manhattan norm is just the sum of the sides.

Example 2.2.5: Manhattan norm of vector (1,1)

Let's demonstrate Manhattan norm on the same (1,1) vector what we analyzed above. Its L_3 norm is

$$\|\mathbf{v}\|_3 = \left[\sum_{i=1}^K |x_i|^1 \right]^{1/1} = (|1|^1 + |1|^1)^{1/1} = 1 + 1 = 2$$

It is easy to see why Manhattan norm is useful (and why is it called taxicab norm). Imagine you are taking cab in a city where there is a rectangular street grid, for instance in Manhattan. If your destination is 10 blocks east and 10 blocks north, then the cab driver has to drive 20 blocks in all. The “Manhattan-length” of your 10-block-by-10-block ride is 20 blocks. It is also easy to understand that the cabdriver may not be too impressed if you tell her that you are only willing to pay for 14 blocks trip because that is the “correct” Euclidean length...

Chessboard norm also *Chebyshev norm* is L_∞ norm. It can be computed as

$$\|\mathbf{x}\|_\infty = \lim_{p \rightarrow \infty} \left[\sum_{i=1}^K |x_i|^p \right]^{1/p} = \max_i |x_i|. \quad (2.2.13)$$

Although we cannot directly compute L_∞ distance by substituting infinity in the L_p formula (2.2.11), the fact that it amounts to maximum individual component is intuitively fairly obvious to understand. Namely, when we take numbers to p -th power as in $|x_i|^p$, the larger numbers “gain” more from this operation if p is large. At the limit where $p \rightarrow \infty$, all other components are negligible next to the largest one.

The name *chessboard norm* refers to the fact that in chess, it measures the number of moves king needs in order to move to the given number of squares in each direction (Figure 2.3).

Metric distance

Closely related to norm is *metric distance*. We can always define distance between vectors \mathbf{x} and \mathbf{y} as

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|. \quad (2.2.14)$$

So a “distance” between two vectors is the “length” of their difference. For

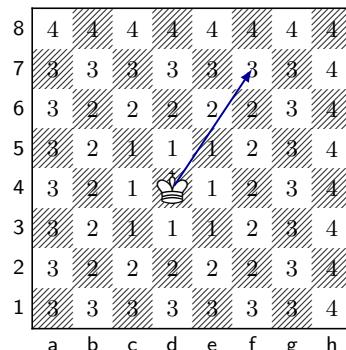


Figure 2.3: King in chess can move one field in every direction. The numbers on the chessboard denote the number of moves king at d4 needs to reach that position. For instance, it needs three moves to get to f7, and hence the vector from d4 to f7 has chessboard norm 3.

suitable “nice” metrics we can also define the opposite

$$\|\mathbf{x}\| = d(\mathbf{x}, \mathbf{0}). \quad (2.2.15)$$

This is possible with L_p norm but not with certain other similarity measures, such as cosine similarity where one cannot define distance from null-vector, $d(\mathbf{x}, \mathbf{0})$, in a consistent manner.

In order for a distance measure $d(\cdot, \cdot)$ to be a proper metric distance, it has to have these three properties:

1. $d(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$ (identity of indiscernibles). Zero distance means the vectors are equal.
2. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ (symmetry). Distance is the same, whichever way you measure it.
3. $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ (triangle inequality). There is no shorter way than the direct route.

Among the distance measure we encounter in these notes, L_p is a proper metric distance but cosine similarity is not.

We illustrate L_p distances by the corresponding unit circles (Figure 2.4). Unit circle is a set of points that are at distance 1 from the origin. In case of Euclidean metric, the unit circle is the familiar circle with radius 1, centered at the origin. In case of the other metrics, the unit circles look different. Some of these have practical applications, for instance walksheds in neighborhoods with grid-like street layout are L_1 circles.

2.3 Matrices

2.3.1 What are matrices

Matrices are some of the central objects in linear algebra. You may imagine matrices as rectangles of numbers, in many ways similar to data frames, that can be indexed based on their rows and columns. As is the case with vectors, the concept “matrix” means two different things, one is data storage on computer, and the other is a mathematical object that has certain mathematical properties.

Here are two examples of matrices:

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \quad (2.3.1)$$

There is no universal way to denote matrices, here we follow one common tradition and use upper case letters in upright sans-serif font like X or Σ . Unlike

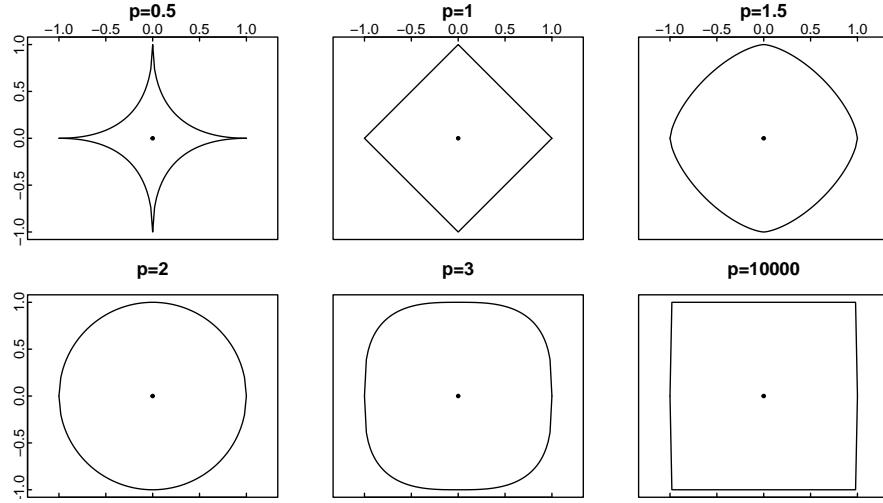


Figure 2.4: Unit circles – points sets of distance 1 from the origin (0,0) (the central dot) in different 2-D L_p spaces. If $p < 2$, the circle looks more like a star, with the Manhattan distance, $p = 1$, being diamond-shaped. If $p > 2$, the circles are more and more box-shaped.

vectors, we do not use bold symbols for matrices. Stacking numbers into rectangles is not of much interest by itself, but it turns out that these rectangles—matrices—make it possible to represent various kinds of data and related operations in a much simpler and more efficient manner.⁴ This is the main reason why linear algebra is ubiquitous in statistics and sciences.

The first matrix A has 3 *rows* and 3 columns, the second matrix B has 3 rows but only a single column. This is *matrix dimension*. Matrix dimension is normally denoted by *rows* \times *columns* and hence matrix A is of dimension 3×3 and B is of 3×1 . But confusingly, *dimension* also means another closely related concept. Namely, sometimes we say that matrices are 2-dimensional while vectors are 1-dimensional objects. This “object dimension” is not to be confused with matrix dimension. Dimension of A is 3×3 while at the same time A is a 2-D object... Normally it is clear from the context what kind of dimension we are talking about.

The individual numbers⁵ the matrices are made of are called *matrix elements* or *matrix components*. These are often denoted by the corresponding lower case

⁴Matrices can be generalized into objects that have 3 or more dimensions, called *tensors*. These are widely used in physics, but also in advanced ML methods, such as neural networks. Modern software, such as *tensorflow* library relies heavily on tensor operations and can employ dedicated hardware, such as GPU or *tensor processing unit* (TPU) for speeding up tensor operations. We do not cover tensors in these notes.

⁵In these notes we only consider numeric matrices. But matrix elements do not have to be just numbers.

letter, supplemented with two *indices*, one for row and the other for column—by convention, matrix elements are indexed first by row and thereafter by column.⁶ For instance, the matrix \mathbf{A} above can be written in more abstract form as

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad (2.3.2)$$

where the a_{11} , the element in the first row and the first column is 1, a_{12} , the element in the first row and the second column is 2, and so on. Sometimes matrix is written by the corresponding elements as $\mathbf{A} = \{a_{ij}\}$ where $i = 1 \dots N$ and $j = 1 \dots M$. This must be understood as we take all the individual elements (numbers) a_{ij} and arrange those into the matrix.

A central role of matrices in machine learning contexts is to hold and manipulate data. For instance, the US States data on page 51 is technically a data frame, but could as well be a matrix. Holding data in matrix form makes it possible to use linear algebra methods, and as we discussed above, this is an excellent option when we are doing multivariate statistics.

A note about matrices and data frames. Both structures look similar, they are both rectangles of data. But while matrices are linear algebra objects, data frames are not. Data frames are a convenient way to store and display heterogeneous data, data where columns can be of different type, not necessarily numbers. Matrices are rectangles of only numbers (as far as these notes are concerned). While matrices in the abstract sense are not related to storage concerns, the computer implementations are. They are normally stored in a different way than data frames to facilitate operations as blocks, while data frames are often designed for easy access by columns in mind.

Matrix Components

Certain combinations of matrix elements have their own names. As these are widely used when discussing matrices, we introduce the most important ones here.

Matrix *diagonal* (also *main diagonal*) are the elements in the form a_{ii} . Consider matrix \mathbf{A} in (2.3.3). Its main diagonal, (a_{11}, a_{22}, a_{33}) , is left white. We usually talk about diagonal in case of square matrices only, but note that it is also defined for non-square matrices. All elements above the main diagonal, i.e. elements a_{ij} where $i < j$, are called *upper triangle* (red in (2.3.3)) and below the diagonal, i.e. elements a_{ij} where $i > j$, are *lower triangle* (blue in (2.3.3)).

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} b_{11} \\ b_{21} \\ b_{31} \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}. \quad (2.3.3)$$

⁶Note that this tradition—rows first and columns second—contradicts with the most common 2-D image data representation: horizontal first and vertical second. This is a frequent source of confusing errors when describing graphical data in matrix form.

Special matrices

There are a number of matrices of special form that are important enough to have a special name. It is important to know a few of those that are used most frequently in the literature.

Square matrix is a matrix with equal number of rows and columns. The matrix A in 2.3.3 is a square matrix while B and C are not.

Symmetric matrix is a matrix where the upper and lower triangle are identical (but mirrored). For instance

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 9 \\ 3 & 9 & 16 \end{pmatrix} \quad (2.3.4)$$

is a symmetric matrix but

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 9 \\ 9 & 3 & 16 \end{pmatrix} \quad (2.3.5)$$

is not.

Formally, A is a symmetric matrix if $a_{ij} = a_{ji}$ for all i, j . Obviously, only square matrices can be symmetric.

Diagonal matrix is a matrix where all elements outside of the main diagonal are zeros. Here is an example of two diagonal matrices:

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} \quad B = \begin{pmatrix} b_{11} & 0 & 0 \\ 0 & b_{22} & 0 \end{pmatrix}. \quad (2.3.6)$$

Note that diagonal matrix may be a non-square matrices as B in the example above, but one almost always refers to square matrices (as matrix A above) when talking about diagonal or non-diagonal matrices.

All square diagonal matrices are symmetric.

Unit matrix (aka *identity matrix*). This is a diagonal square matrix where there are ones on the main diagonal and zeros elsewhere. It is conventionally denoted by I , or I_n for $n \times n$ identity matrix in cases where the dimension is not obvious from the context. Here are two examples:

$$I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The importance of unit matrix is related to it's properties in [matrix multiplication](#) where it is the neutral element, exactly like number one is the neutral

element when multiplying numbers. Matrix-multiplying every compatible matrix A with the unit matrix I results

$$I \cdot A = A \cdot I = A \quad (2.3.7)$$

(see more in Section 2.3.2 below). In particular, this means $I \cdot I = I$.

Vectors as matrices

When working with matrices it is common to treat vectors just as a special kind of matrices, 2-dimensional objects where one dimension is equal to 1. So unlike “true” vectors, vectors-as-matrices have additional properties, namely number of rows and number of columns. However, despite treating vectors as matrices, vectors are typically denoted by lower case letters in slanted bold font. We follow this habit here.

Vector-as-matrix approach gives us two types of vectors: *column vectors* are of shape $N \times 1$ and *row vectors* are of $1 \times N$. For instance, $\mathbf{a} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ and $\mathbf{b} = \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}$ are column vectors while $\mathbf{c} = (-1 \ -2)$ and $\mathbf{d} = (-1 \ 0 \ 1)$ are row vectors.

Normally, if no extra explanation is given, the vectors are assumed to be column vectors. So if we talk about vector \mathbf{x} , we mean a column vector, unless we explicitly state that it is a row vector. Its transpose (see below) however, \mathbf{x}^\top , is a row vector. In order to save space, it is also customary to use row vectors and transposition operator to denote column vector. For instance, to denote a

column vector $\mathbf{z} = \begin{pmatrix} -1 \\ 1 \\ -1 \\ 1 \end{pmatrix}$ we often write $\mathbf{z} = (-1 \ 1 \ -1 \ 1)^\top$.

2.3.2 Matrix operations

Matrix Transposition

A widely used operation, *matrix transposition* is “mirroring” matrix on its main diagonal. We denote transposed matrix by superscript $^\top$ in these notes.⁷ The transposes of the matrices above in (2.3.3) are

$$A^\top = \begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{pmatrix} \quad B^\top = (b_{11} \ b_{21} \ b_{31}) \quad C^\top = \begin{pmatrix} c_{11} & c_{21} \\ c_{12} & c_{22} \\ c_{13} & c_{23} \end{pmatrix}. \quad (2.3.8)$$

Note that transposition swaps the number of rows and columns.

Formally, if $A = \{a_{ij}\}$ where $i = 1 \dots N$ and $j = 1 \dots M$, then $A^\top = \{a_{ji}\}$.

It is easy to see that the transpose of a symmetric matrix is identical to the matrix itself.

⁷The other widely used notation for matrix transposition is apostrophe like A' .

Scalar Multiplication

Matrix multiplication with a scalar (a number) is defined exactly as in case of vectors, by multiplying every matrix element with that scalar. If $\mathbf{A} = \{a_{ij}\}$, then $\lambda\mathbf{A} = \{\lambda a_{ij}\}$. For instance,

$$3 \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 6 & 9 \\ 9 & 6 & 3 \end{pmatrix} \quad -1 \begin{pmatrix} 0 \\ 10 \\ 20 \end{pmatrix} = \begin{pmatrix} 0 \\ -10 \\ -20 \end{pmatrix}. \quad (2.3.9)$$

It is common to denote scalar multiplication either by dot like $\lambda \cdot \mathbf{A}$, or by just $\lambda\mathbf{A}$. We'll use both notations in these notes.

Matrix Addition (and Subtraction)

Matrix addition is defined as elementwise operations: if $\mathbf{A} = \{a_{ij}\}$ and $\mathbf{B} = \{b_{ij}\}$, then $\mathbf{A} + \mathbf{B} = \{a_{ij} + b_{ij}\}$. For instance,

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + \begin{pmatrix} 3 & 2 & 1 \\ 4 & 3 & 2 \\ 5 & 4 & 3 \end{pmatrix} = \begin{pmatrix} 4 & 4 & 4 \\ 8 & 8 & 8 \\ 12 & 12 & 12 \end{pmatrix} \quad (2.3.10)$$

and

$$(1 \ 0 \ -1) - (2 \ 1 \ 0) = (-1 \ -1 \ -1). \quad (2.3.11)$$

Obviously, only matrices with similar dimensions can be added and subtracted.

Matrix Multiplication

Matrix multiplication is among the most important matrix operations, and one of the prime tools for data manipulation. Matrix product can be done manually, and although we almost always use computers in practice, it is important to have the basic understanding of it. In particular, understanding how matrix dimensions play in multiplications, and being able to compute simple products manually are invaluable skills for both coding, debugging, and devising easier and faster ways to solve data problems.

Matrix product is defined in a way that we take a row from the first matrix, column from the second matrix, multiply the corresponding elements, and sum these products. This will be the element of the product matrix at the row (the row number) that was taken from the first matrix, and column (the column number) that was taken from the second matrix. This process must be repeated for every row in the first and for every column in the second matrix. Note that for this to be possible, the number of columns in the first matrix must equal to the number of rows in the second matrix. If this is not the case, the matrices cannot be multiplied.

This definition can be understood as a visual rule: all rows of the first matrix must be multiplied by all columns of the second matrix. Lets multiply

$C = A \cdot B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$. You can imagine a process like this, where rows of A are denoted with blue and columns of B with pink:

$$\begin{pmatrix} c_{11} & \cdot \\ \cdot & \cdot \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ \cdot & \cdot \end{pmatrix} \cdot \begin{pmatrix} 4 & \cdot \\ 2 & \cdot \end{pmatrix} = 1 \cdot 4 + 2 \cdot 2 = 8 \quad (2.3.12)$$

$$\begin{pmatrix} \cdot & c_{12} \\ \cdot & \cdot \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ \cdot & \cdot \end{pmatrix} \cdot \begin{pmatrix} \cdot & 3 \\ \cdot & 1 \end{pmatrix} = 1 \cdot 3 + 2 \cdot 1 = 5 \quad (2.3.13)$$

$$\begin{pmatrix} \cdot & \cdot \\ c_{21} & \cdot \end{pmatrix} = \begin{pmatrix} \cdot & \cdot \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 4 & \cdot \\ 2 & \cdot \end{pmatrix} = 3 \cdot 4 + 4 \cdot 2 = 20 \quad (2.3.14)$$

$$\begin{pmatrix} \cdot & \cdot \\ \cdot & c_{22} \end{pmatrix} = \begin{pmatrix} \cdot & \cdot \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} \cdot & 3 \\ \cdot & 1 \end{pmatrix} = 3 \cdot 3 + 4 \cdot 1 = 13. \quad (2.3.15)$$

and hence $C = \begin{pmatrix} 8 & 5 \\ 20 & 13 \end{pmatrix}$. Note how c_{11} is calculated from the first row of A and the first column of B , c_{12} from the first row of A and the second column of B , and so on. In general, c_{ij} is “made” of i -th row of A and j -th column of B .

More formally, let A be $N \times K$ matrix and B be $K \times M$ matrix. We define the product as

$$C = A \cdot B \quad (2.3.16)$$

where c_{ij} , the element of C at the row i and column j , is defined as

$$c_{ij} = \sum_{k=1}^K a_{ik} b_{kj}. \quad (2.3.17)$$

C dimensions are determined by the number of rows in A and number of columns in B , hence C is $N \times M$.

Let’s repeat the example from above using the formal rule. We have $C = A \cdot B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$. Here $N = K = M = 2$, hence the product will be a 2×2 matrix. We can take the definition directly and compute c_{11} :

$$c_{11} = \sum_{k=1}^2 a_{1k} b_{k1} = 1 \cdot 4 + 2 \cdot 2 = 8. \quad (2.3.18)$$

Analogously we can do all the other elements:

$$c_{12} = \sum_{k=1}^2 a_{1k} b_{k2} = 1 \cdot 3 + 2 \cdot 1 = 5 \quad (2.3.19)$$

$$c_{21} = \sum_{k=1}^2 a_{2k} b_{k1} = 3 \cdot 4 + 4 \cdot 2 = 20 \quad (2.3.20)$$

$$c_{22} = \sum_{k=1}^2 a_{2k} b_{k2} = 3 \cdot 3 + 4 \cdot 1 = 13 \quad (2.3.21)$$

$$(2.3.22)$$

and hence, as above, $C = \begin{pmatrix} 8 & 5 \\ 20 & 13 \end{pmatrix}$. In practice, when multiplying matrices manually, it is easier to follow the visual rule we described above.

Example 2.3.1: Product of non-square matrices

Multiply the following non-quadratic matrices: $C = \begin{pmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

First, note the first matrix has 3 rows and the second has 1 column, hence the result will be a 3×1 matrix. Use the visual rule:

$$\begin{aligned} \begin{pmatrix} c_{11} \\ \cdot \\ \cdot \end{pmatrix} &= \begin{pmatrix} -1 & 1 \\ \cdot & \cdot \\ \cdot & \cdot \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} = -1 \cdot 1 + 1 \cdot 2 = 1 \\ \begin{pmatrix} \cdot \\ c_{21} \\ \cdot \end{pmatrix} &= \begin{pmatrix} \cdot & \cdot \\ 1 & -1 \\ \cdot & \cdot \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} = 1 \cdot 1 - 1 \cdot 2 = -1 \\ \begin{pmatrix} \cdot \\ \cdot \\ c_{31} \end{pmatrix} &= \begin{pmatrix} \cdot & \cdot \\ \cdot & \cdot \\ -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \end{pmatrix} = -1 \cdot 1 + 1 \cdot 2 = 1 \end{aligned} \quad (2.3.23)$$

and hence the answer is $\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$.

Properties of Matrix Product

Matrix product has a number of useful properties, several of which it shares with product of real numbers. Most of these can be proven by following the definition of the corresponding operations.

Multiplication by scalar

$$(\lambda A) \cdot B = A \cdot (\lambda B) = \lambda(A \cdot B). \quad (2.3.24)$$

In case of numbers, this is analogous to the fact that the product does not depend on the order of factors.

Associative property

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C \quad (2.3.25)$$

given the corresponding products exist. This property is shared with ordinary numbers.

Exercise 2.3.1: Test the associative property

Compute the products

$$\left[\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ -1 & 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \right] \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

and

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ -1 & 2 & 3 \end{pmatrix} \cdot \left[\begin{pmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \right]$$

Distributive property For matrices A, B, C we have

$$A \cdot (B + C) = A \cdot B + A \cdot C. \quad (2.3.26)$$

This is also the behavior of ordinary numbers.

Non-commutative Unlike numbers, matrix product is not commutative in general:

$$A \cdot B \neq B \cdot A. \quad (2.3.27)$$

There are many special cases though, for instance multiplication by null matrix and by unit matrix is commutative, as is multiplication by 1×1 matrix, essentially just a number.

As matrix product is non-commutative, we have to distinguish *pre-multiplication* (*left-multiplication*) and *post-multiplication* (*right-multiplication*). For instance, in the expression $A \cdot B$, B is pre-multiplied by A and A is post-multiplied by B . As the product is not commutative, we have to preserve the multiplication type when doing algebra.

Transpose of product Transpose of matrix product

$$(A \cdot B)^T = B^T \cdot A^T. \quad (2.3.28)$$

So a product can be transposed by a) transposing the factors; and b) switching their order.

This property has no real analogue with numbers as transposition of numbers carries little meaning.

Example 2.3.2: Transpose of matrix product

Lets take $A = \begin{pmatrix} 3 & 2 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$. Compute $(A \cdot B)^T$. First, lets

compute the product and transpose it:

$$\mathbf{A} \cdot \mathbf{B} = (3 \ 2 \ 1) \cdot \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = 3 \cdot 1 + 2 \cdot 2 + 1 \cdot 3 = 10. \quad (2.3.29)$$

As this is just a number, its transpose is 10 as well. By using the transposition formula, we have

$$\mathbf{B}^T \cdot \mathbf{A}^T = (1 \ 2 \ 3) \cdot \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix} = 1 \cdot 3 + 2 \cdot 2 + 3 \cdot 1 = 10. \quad (2.3.30)$$

Note that the products $\mathbf{B} \cdot \mathbf{A}$ and $\mathbf{A}^T \cdot \mathbf{B}^T$ would result in a 3×3 matrix instead.

Vector multiplication as matrix product

There are two ways to multiply vectors: inner (dot) product and outer product.

When treating vectors as matrices, we have two types of vectors instead — column vectors and row vectors — and we can treat both types of multiplication as just the matrix product of different types of vectors. Namely, product of a row vector and column vector is equivalent to inner product, while column vector multiplied by row vector is will give the outer product.

Let $\mathbf{a} = (a_1 \ a_2 \ \dots \ a_N)^T$ and $\mathbf{b} = (b_1 \ b_2 \ \dots \ b_N)^T$. Now

$$\mathbf{a}^T \cdot \mathbf{b} = \mathbf{b}^T \cdot \mathbf{a} = a_1 b_1 + a_2 b_2 + \dots + a_N b_N \quad (2.3.31)$$

is the inner product of \mathbf{a} and \mathbf{b} while

$$\mathbf{a} \cdot \mathbf{b}^T = (\mathbf{b} \cdot \mathbf{a}^T)^T = \begin{pmatrix} a_1 b_1 & a_1 b_2 & \dots & a_1 b_N \\ a_2 b_1 & a_2 b_2 & \dots & a_2 b_N \\ \vdots & \vdots & \ddots & \vdots \\ a_N b_1 & a_N b_2 & \dots & a_N b_N \end{pmatrix}. \quad (2.3.32)$$

is their outer product. It is useful to keep in mind that $\mathbf{a}^T \cdot \mathbf{b}$ (row times column) is just a single number but $\mathbf{a} \cdot \mathbf{b}^T$ (column times row) is a $N \times N$ matrix.

Example 2.3.3: Inner and outer product

Let's multiply length-3 vectors of ones: $\mathbf{a} = \mathbf{b} = \mathbf{1}_3$. Their inner product is

$$\mathbf{a}^T \cdot \mathbf{b} = (1 \ 1 \ 1) \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 = 3, \quad (2.3.33)$$

just a number. The outer product is

$$\mathbf{a} \cdot \mathbf{b}^\top = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot (1 \quad 1 \quad 1) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad (2.3.34)$$

a 3×3 matrix.

Other Matrix Operations

Trace of Matrix For a square matrix, its *trace* is the sum of its diagonal elements:

$$\text{Tr } \mathbf{A} = \sum_{i=1}^N a_{ii}. \quad (2.3.35)$$

Example 2.3.4: Matrix trace

$$\text{Tr} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = 1 + 5 + 9 = 15. \quad (2.3.36)$$

Determinant TBD

2.3.3 Inverse Matrix

Column space vector space, generated by the column vectors of the matrix

Column rank dimension of the column space

Row space vector space, generated by the row vectors of the matrix

Row rank dimension of the row space

Full column rank column rank equals to the number of columns

Full rank rank equals to the smallest of either number of rows or number of columns

Theorem 1. Row and column rank are equal (and are called *matrix rank*)

is a scalar function of matrix elements

- Useful descriptor of matrix properties in many contexts

- For 2×2 matrix $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$$\det \mathbf{A} \equiv |\mathbf{A}| = a_{11}a_{22} - a_{12}a_{21}$$

- for 3×3 matrix $B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$

$$\det B \equiv |B| = b_{11}b_{22}b_{33} + b_{12}b_{23}b_{31} + b_{21}b_{32}b_{13} - b_{31}b_{22}b_{13} - b_{21}b_{12}b_{33} - b_{11}b_{23}b_{32}$$

Theorem 2. determinant is non-zero \Leftrightarrow matrix is full rank

Easy to see for a diagonal matrix

- A : square matrix

B is *inverse* A iff

$$BA = I,$$

and is denoted by A^{-1} .

Properties:

- $A^{-1} \cdot A = A \cdot A^{-1} = I$
- A^{-1} is also a square matrix
- A^{-1} is unique

For 2×2 matrix

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

the inverse is

$$A^{-1} = \frac{1}{|A|} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

Definition 1. Matrix is *non-singular* \Leftrightarrow inverse exists

Theorem 3. Matrix is *non-singular* \Leftrightarrow it is full rank

Hence non-singular matrix

- has non-zero determinant
- is full-rank
- does not contain linearly dependent columns or rows (whichever is smaller)
- How does the size of ϵ affect the precision?

Exercise 2.3.2: Find base vector multiplier for a given vector

Equation (2.2.5) shows how all vectors in the vector space can be made of

base vectors:

$$\mathbf{c} = \alpha \cdot \mathbf{a} + \beta \cdot \mathbf{b} \quad \alpha, \beta \in \mathbb{R}.$$

Given vector \mathbf{c} and the base vectors \mathbf{a} and \mathbf{b} and, compute the multipliers α and β .

Hint: attempt to transform (2.2.5) in such a way that vectors \mathbf{a} and \mathbf{b} are combined in a matrix and α and β in a vector, and use matrix multiplication instead of addition. Note also that we are in a 2-D space and hence all vectors only have two components.

Characteristic roots (eigenvalues) are solutions (λ) of the equation

$$\mathbf{A}\mathbf{c} = \lambda\mathbf{c}$$

Characteristic vectors (eigenvectors) are corresponding \mathbf{c} -s.

Intuition:

- Multiplication by \mathbf{A} does not change \mathbf{c}
- Only scales by λ .

Rewrite:

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{c} = \mathbf{0}$$

$\Rightarrow \mathbf{A} - \lambda\mathbf{I}$ must be singular $\Rightarrow |\mathbf{A} - \lambda\mathbf{I}| = 0$.

Matrix:

$$\mathbf{A} = \begin{pmatrix} 30 & 28 \\ 28 & 30 \end{pmatrix}$$

Solve for eigenvalues (using the $\det \mathbf{A} = 0$ condition $|\mathbf{A} - \lambda\mathbf{I}| = 0$):

$$|\mathbf{A}| = (30 - \lambda)(30 - \lambda) - 28^2 = 0$$

The solution:

$$\lambda_1 = 58 \quad \lambda_2 = 2$$

The corresponding eigenvectors:

$$\begin{pmatrix} 30 & 28 \\ 28 & 30 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix}$$

and we have

$$\mathbf{c}_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \quad \mathbf{c}_2 = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

Find eigenvalues, eigenvectors of the unit matrix

$$\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

2.3.4 Eigenvalues

TBD: eigenvalue decomposition

Condition number

While matrix rank is a very convenient mathematical concept—matrix either is full rank or it is not—this is not as clear cut in practice. The problem arises from imprecise data and numerical errors, and all matrix manipulation methods give numerical errors, the larger the matrix, the larger the errors. This means, in practice, that we cannot rely on simple yes/no answer to the full rank question. We need another measure, a continuous measure that tells us how close we are to singularity. Condition number offers such a measure.

Condition number is defined as ratio of the largest and smallest eigenvalues (in absolute value):

$$\kappa \equiv \frac{|\lambda|_{\max}}{|\lambda|_{\min}} \quad (2.3.37)$$

where λ -s are the eigenvalues, $|\lambda|$ -s are the absolute values of eigenvalues (moduli in case of complex eigenvalues), and $|\lambda|_{\max}$ and $|\lambda|_{\min}$ are the largest and the smallest eigenvalue in terms of the absolute value.⁸ We know that singular matrices have (at least) one eigenvalue equal to zero, and hence the condition number is infinite (unless it is a zero matrix). On the other hand, if all the eigenvalues are equal, $\kappa = 1$. So κ constitutes a continuous measure of “how close to singularity” a matrix is.

Example 2.3.5: Condition numbers

As all eigenvalues of the unit matrix

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

are equal to 1, its condition number is 1 as well.

The singular matrix

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

has eigenvalues 16.12, -1.117 and 0. As 0 is the smallest of those (in absolute value), the condition number $\kappa = 16.12/0$ is infinite.

⁸Greene (2003) defines condition number as square root of this expression. This is more convenient in practice as one has to work with smaller values.

2.3.5 Applications: Images in Matrix Form

Wireframe Images Matrices are complex structures that are often hard to understand intuitively. But there are exceptions. One of these is matrix representation of images. We first introduce images in wireframe (line art) form and discuss photos (bitmap images) later.

Let's make an image of \mathbb{B} , the runic letter for “b”. The line art images can be represented in vertices, the “corners” of images, that are connected to other vertices. Let's put the bottom of \mathbb{B} in the origin, and call it A . Let us also make the letter's height equal to two. The vertex coordinates may be

name	x	y	explanation
A	0	0	bottom
B	0	2	top
C	0.6	1.5	upper triangle
D	0	1	middle
E	0.6	0.5	lower triangle
A	0	0	back to bottom

The actual data is the columns x and y . We can put these in matrix B :

$$B = \begin{pmatrix} 0 & 0 \\ 0 & 2 \\ 0.6 & 1.5 \\ 0 & 1 \\ 0.6 & 0.5 \\ 0 & 0 \end{pmatrix}. \quad (2.3.38)$$

We can plot these vertices, and connect them by lines (Figure 2.5). This is a convenient way to represent wireframe images. For more complex images we may add additional data, e.g. color of the vertices, and a list of vertex pairs that are connected (this is called *edgelist*). This image representation format can be easily generalized to higher dimensions, e.g. for 3-D objects.

Image Rotation Prerequisites: Matrix multiplication, Basic trigonometrics

Matrix form is a convenient data representation for various image transformations that can be expressed through linear operators. These include image rotation, scaling and projection on lower-dimensional hyperplanes.

Image rotation can be done by multiplying the object vertex data by *rotation matrix*. We define 2-D rotation matrix as

$$R(\alpha) = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix}. \quad (2.3.39)$$

This matrix will rotate the image (as defined above) *clockwise* by angle α if

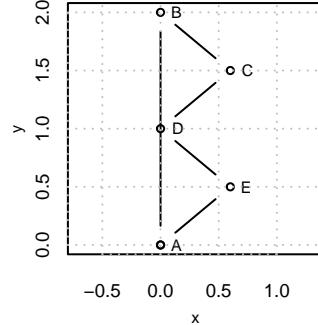


Figure 2.5: Wireframe image of the \mathbb{B} -rune defined by matrix B in (2.3.38). All vertices are plotted and thereafter sequentially connected.

post-multiplied by $R(\alpha)$:⁹

$$B^\alpha = B \cdot R(\alpha). \quad (2.3.40)$$

Note that the 2-D rotation matrix must 2×2 square matrix: two rows are needed to be compatible with 2-column data matrices, and two columns ensure that rotated data still has two columns, one for (rotated) x and one for y .

Let's rotate the vertices of \mathbb{B} -rune, defined above, by 30° . The corresponding rotation matrix will look like

$$R(30^\circ) = \begin{pmatrix} \cos 30^\circ & -\sin 30^\circ \\ \sin 30^\circ & \cos 30^\circ \end{pmatrix} = \begin{pmatrix} 0.866 & -0.5 \\ 0.5 & 0.866 \end{pmatrix} \quad (2.3.41)$$

and the rotated vertices are

$$B^{30} = B \cdot R(30^\circ) = \begin{pmatrix} 0 & 0 \\ 0 & 2 \\ 0.6 & 1.5 \\ 0 & 1 \\ 0.6 & 0.5 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0.866 & -0.5 \\ 0.5 & 0.866 \end{pmatrix} = \begin{pmatrix} 0.00 & 0.00 \\ 1.00 & 1.73 \\ 1.27 & 1.00 \\ 0.50 & 0.87 \\ 0.77 & 0.13 \\ 0.00 & 0.00 \end{pmatrix}. \quad (2.3.42)$$

The rotated matrix is given in Figure 2.6. Note that the vertex metadata is not affected by rotation. Here these are just vertex labels and the connection rule (we just connect all vertices to the next vertex), but these may also include vertex colors and more complex edgelists.

⁹We defined the image by stacking the x and y coordinates of vertices in *columns*. Alternatively, x and y can be stacked in rows. This is equivalent to transposing the image data matrix (B in (2.3.38)) as defined here. Accordingly, the corresponding formula will look like transpose of (2.3.40): $B^T \alpha = R(\alpha)^T \cdot B^T$.

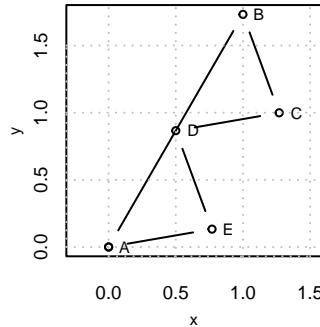


Figure 2.6: The same object as in Figure 2.5 but rotated 30 degrees by multiplication with the corresponding rotation matrix. The rotated vertices are given as B^{30} in (2.3.42).

Exercise 2.3.3: Inverse of rotation matrix

Intuitively, the inverse of a rotation matrix $R(\alpha)$, $R(\alpha)^{-1}$, must be rotation in the opposite direction by a similar amount, i.e. $R(-\alpha)$. Show that for 2-D rotation matrices this is indeed the case, i.e. $R(\alpha) \cdot R(-\alpha) = I$.

It is easy to generalize the rotation matrix into higher dimensions. In the 3-D space, we can rotate the object around each of the three axes, x , y , and z , and hence we have three rotation matrices. For *right-handed* coordinate system these are

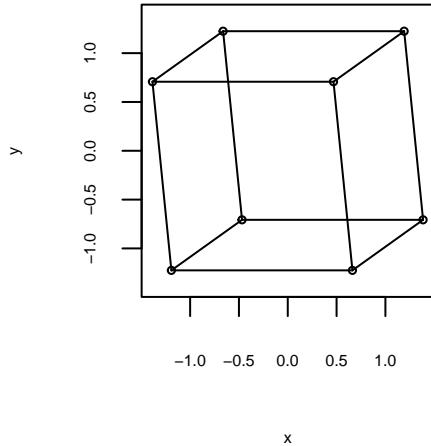
$$R^{xy}(\alpha) = \begin{pmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad R^{zx}(\alpha) = \begin{pmatrix} \cos \alpha & 0 & \sin \alpha \\ 0 & 1 & 0 \\ -\sin \alpha & 0 & \cos \alpha \end{pmatrix}$$

and $R^{yz}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{pmatrix}$. (2.3.43)

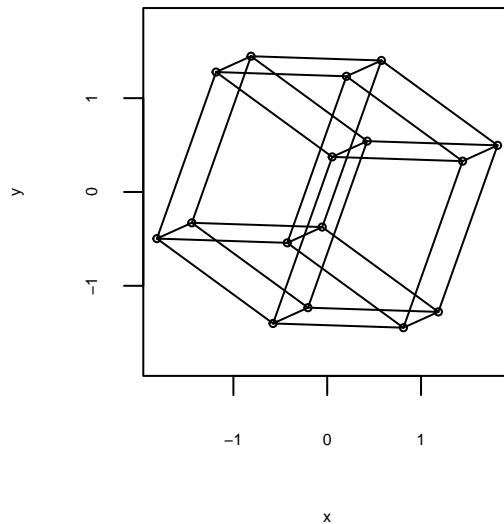
Arbitrary rotations in space can be achieved by sequential application of these matrices. Although our imagination stops here, it is easy to continue into even higher dimensions. However, to display three or higher dimensional objects, we have to project these on the 2-D plane.

Projection Projection (more specifically parallel projection) can be defined as removing some of the coordinates, e.g. dropping z and keeping only x and y to make a 2-D projection of a 3-D object. If we want to project the object onto another plane besides the $x - y$ plane, we may start by rotating it first. As dropping a coordinate is equivalent of deleting the corresponding column, the projection matrix is similar to rotation matrix with the dropped column removed.

Example: cube, rotated and projected on 2-D:



This is easy to understand. But we can make two-dimensional projections not just 3-D spatial objects—we can go into higher dimensions. Here is a similar image of 4-D cube, *tesseract*. This is impossible to understand as our brain has virtually no experience with 4-D world.



Bitmap Images as Matrices Images are normally stored as 2-D matrices of gray values, or stacks of 2-D matrices (3-D tensors) in case of color images, one matrix for each color channel (and sometimes a fourth channel for transparency). Figure 2.7 depicts one such grayscale image. Figure 2.8 shows a closer view on a 10×10 pixel detail, centered at the locomotive's smokestack. The image contains pixels, rectangular squares of different gray value. These gray values

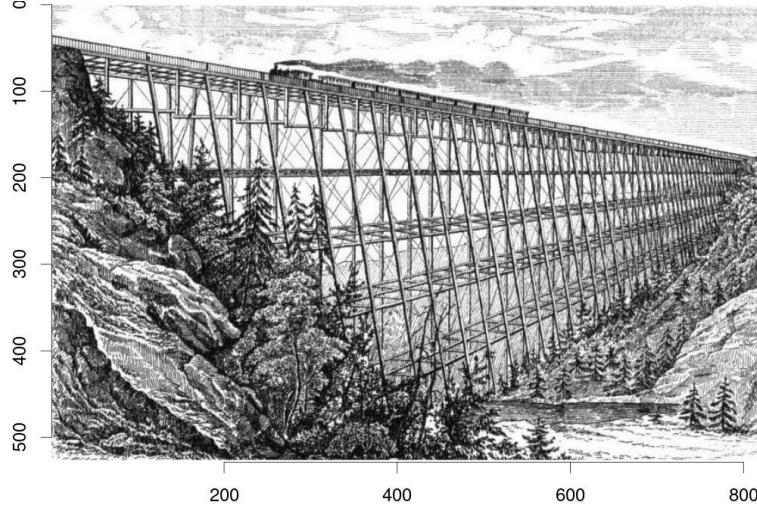
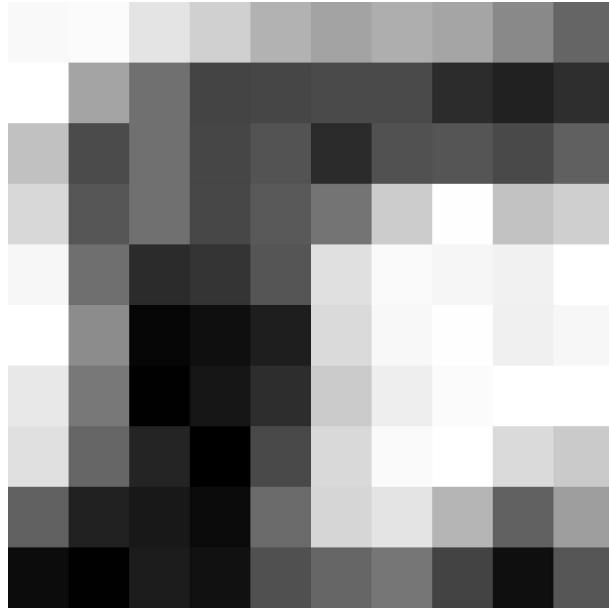


Figure 2.7: Lyman Trestle in Connecticut, around 1876 (from [Wikimedia Commons](#)). It is stored in memory as 820×526 array of gray values. The axes depict the corresponding pixel coordinates.

of pixels is what is stored in the image matrix. The matrix, corresponding to the image, is shown in the lower panel. For instance, the first line of image that transforms from light sky to dark gray of smoke corresponds to the first row of the matrix where values change from 0.98 (light sky) to 0.43 (dark smoke on the right hand side). One can also see that the darkest areas of the smokestack are of value close to zero while the lightest points are of value 1.00.

In practice, it is important to keep in mind that matrices are typically stored as *rows-by-columns*, while images (and plotting coordinates) are typically presented as *width-by-height*. Also, high values may correspond to either dark or low pixel intensity. All this is obviously software-specific.

Transforming Bitmap Images into Coordinate Matrix Form Wireframe image data we discussed above contain vertex coordinates, while color, a vertex attribute, is a secondary consideration. In contrast, bitmaps images only contain the color values in a regular grid but do not explicitly contain the pixel coordinates. In this sense wireframe images are similar to sparse matrices and bitmaps to dense matrices. So in order to rotate a bitmap as matrix, we first have to create it's coordinate matrix. This can be done by creating an (x, y) coordinate pair for each pixel so that the pixels can be described by a triple (x, y, value) . As the pixels are arranged in the matrix in a regular matrix, x and y correspond to either matrix columns and rows, or the way around, depending on the software. Are vertex coordinates and colors will be treated differently, we put the image information into two matrices: a $N \times 2$ coordinate matrix \mathbf{X} , and a $N \times 1$ pixel gray value matrix \mathbf{G} . Note that N is not the height but $height \times width$ of the



(a) Detail (locomotive's smokestack) from Figure 2.7.

	1	2	3	4	5	6	7	8	9	10
1	0.98	0.98	0.90	0.82	0.71	0.66	0.70	0.67	0.56	0.43
2	1.00	0.66	0.47	0.31	0.31	0.33	0.33	0.22	0.17	0.22
3	0.77	0.33	0.47	0.31	0.36	0.21	0.35	0.37	0.32	0.41
4	0.85	0.37	0.47	0.31	0.38	0.48	0.81	1.00	0.77	0.82
5	0.97	0.46	0.21	0.24	0.37	0.89	0.98	0.96	0.95	1.00
6	1.00	0.57	0.07	0.11	0.16	0.86	0.97	1.00	0.95	0.97
7	0.91	0.50	0.05	0.13	0.22	0.81	0.94	0.98	1.00	1.00
8	0.89	0.43	0.18	0.05	0.32	0.86	0.98	1.00	0.86	0.80
9	0.41	0.17	0.14	0.09	0.45	0.85	0.90	0.73	0.41	0.64
10	0.09	0.05	0.16	0.11	0.35	0.43	0.49	0.30	0.11	0.37

(b) The gray level values corresponding to the detail in the upper panel. The high values (near 1.0) correspond to white and low values (near 0.0) correspond to black. One can see that the darkest details of the smokestack are of value 0.05 and the lightest sky is of value 1.00.

Figure 2.8: Detail from the image and the corresponding gray values.

image because each row in these matrices correspond to a single pixel, not to a single row. For instance, the image detail from figure 2.8 will be stored in two matrices,

$$\mathbf{X} = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ \vdots & \vdots \\ 1 & 2 \\ 2 & 2 \\ 3 & 2 \\ \vdots & \vdots \end{pmatrix} \quad \text{and} \quad \mathbf{G} = \begin{pmatrix} 0.98 \\ 0.98 \\ 0.90 \\ \vdots \\ 1.00 \\ 0.66 \\ 0.47 \\ \vdots \end{pmatrix}. \quad (2.3.44)$$

The first column of \mathbf{X} denotes the horizontal position, the column of the original image matrix, and it runs from 1 to the image width for each row. The second column is the vertical position, the image row, it is 1 for each pixel in the first row, 2 for each pixel in the second row and so forth. \mathbf{G} contains the same gray values as the data matrix in Figure 2.8. The pixel coordinates \mathbf{X} are conceptually the same as vertex coordinates for wireframe images, just we are not connecting vertices by lines but instead we use the gray value as the vertex color. The gray values will not change with rotation, just the pixels must be plotted in a different place as the image rotates.

Accordingly the image rotation will consist of two steps:

1. rotate (or otherwise transform) the coordinates \mathbf{X} into \mathbf{X}' , and
2. paint a dot at coordinates (x'_{i1}, x'_{i2}) with the gray value G_i .

Figure 2.9 illustrates this approach with the original image rotated 10 degrees counterclockwise:

Projecting bitmap images As we can rotate bitmap images, we can also project these on 1-D line. Figure 2.10 depicts an image of a page of text that is rotated by 24 degrees. Suppose we want to detect its degree of rotation for further processing. One approach is to rotate the image and project the result onto the vertical axis. If the angle is correct, we should see a clear pattern of dark (text lines) and white (interline gaps). If the angle is wrong, the gaps will be unclear.

We can proceed in a similar fashion as above. First we translate the image into the regular grid coordinate matrix \mathbf{X} and the gray intensity levels \mathbf{G} , and thereafter we project \mathbf{X} onto the vertical line (by discarding the first coordinate component). Thereafter we can either plot the density on the margin, or better, compute the sum of gray levels in narrow intervals.

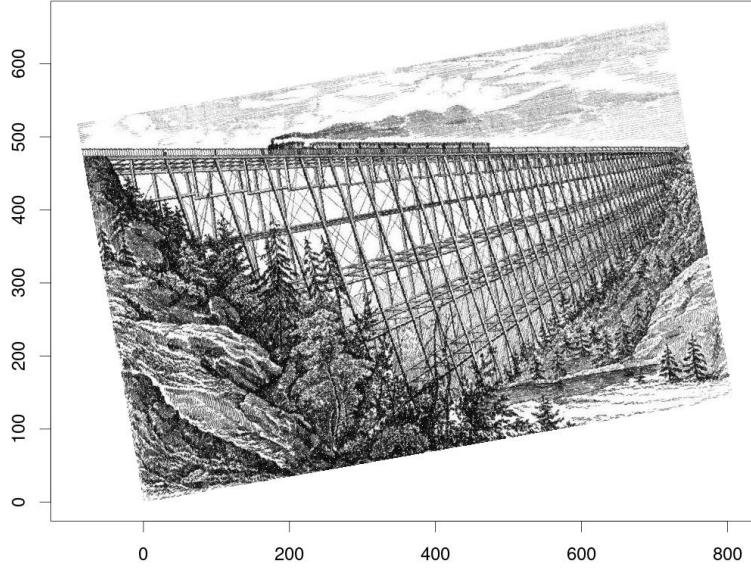


Figure 2.9: The same image rotated 10 degrees.

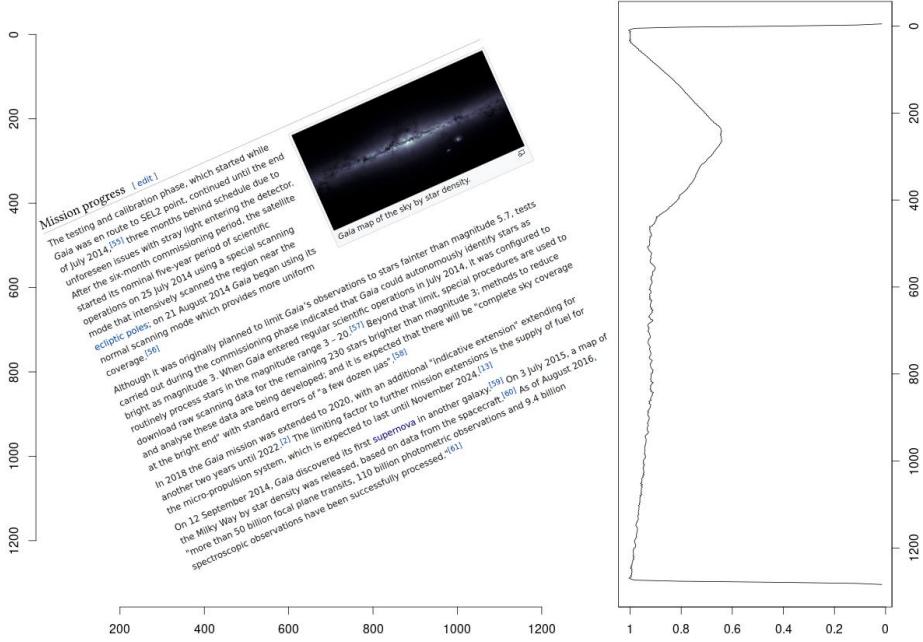


Figure 2.10: Image of a text page (left panel). It is rotated 24 degrees counterclockwise. Right panel depicts the gray value density along the vertical axis. The galaxy image in the form of a triangular dip, centered at row 200, is clearly visible. However, the text lines cannot be distinguished in the plot.

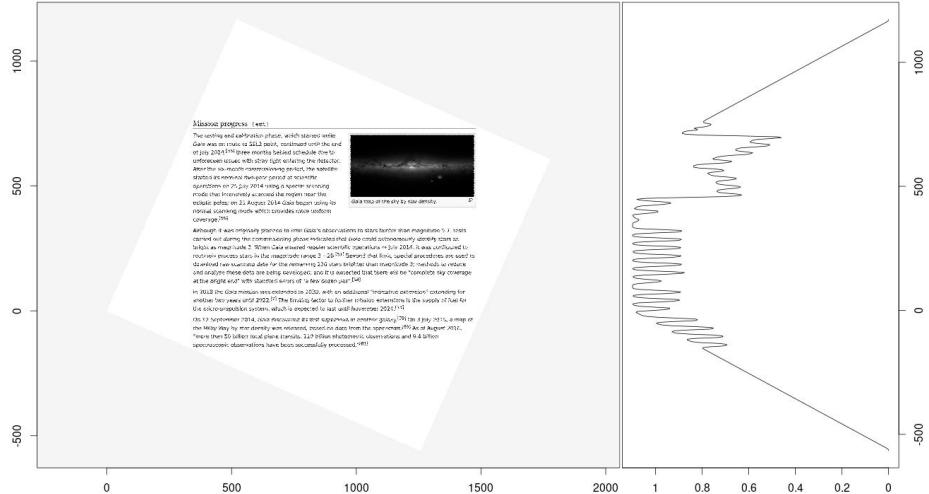


Figure 2.11: The same image as in Figure 2.10 but now rotated into correct position (left panel). The image is now visible as the rectangular dip with vertical sides. Now also the text lines are represented by a regular wavy pattern of lighter and darker stripes. Smooth slopes on both sides of the true image are related to the tilted white background embedded in the image.

Chapter 3

General Machine Learning Strategies

3.1 Numeric Data

This is one of the most common forms of data, and in a way the easiest one to work with. Most machine learning and other analytical methods are designed for numerical data, even more, typical mathematical operations we want to do, such as multiplication and addition, can only be done using numeric data. However, numeric data is not just numbers. It can come in various forms, and not all forms of numeric data works with all methods.

Table 3.1: Recent house sales

id	price (\$ 1000)	m ²	crime (per 1000)
a	800	200	0.2
b	1500	400	0.1
c	?	200	0.1

3.2 Metric Distance: A Revisit

Section 2.2.2 introduced the concept of *metric*. We mainly discussed Euclidean and other L_p -related metrics. However, in machine learning applications it is often useful to let data decide the way we measure distance. This gives rise to various data-based transformations, including feature normalization and Mahalanobis distance.

Many ML applications also permit violations of the strict assumptions behind the distance metric. We may not be particularly concerned if triangle inequality does not hold, or if distance is zero only for identical vectors. We want instead a simple and good enough method to rank vectors. This gives us the popular cosine similarity. Below we discuss a number of more popular such approaches.

3.2.1 Data-Driven Metrics

Imagine you are using the nearest neighbors method to predict house prices. Your dataset contains two training examples (a and b) and house c where you have to predict the price (See Table 3.1). Nearest neighbors (see Section 7.1) predicts the house c to have the same price as the more “similar” one of the training examples a and b .

But which house is more similar? Clearly, house a is of the same size while b is in a similar neighborhood. Obviously, we can choose a distance metric and compute distance. For instance, the Euclidean distance $d_E(\mathbf{c}, \mathbf{a}) = 0.1$ and $E(\mathbf{c}, \mathbf{b}) = 200$ and hence house a is more similar to house c than house b . But does this way of deciding similarity make sense? If we measure house size in km² and crime rate per million instead of per 1000 residents, we would come to the opposite conclusion. So we can identify two separate issues here:

1. Ranking according to Euclidean metric (as well as other L_p metrics) is not robust with respect to change of units.
2. We don’t know how we should weight difference in size relative to the difference in crime rate.

The first problem is actually a specific manifestation of the second problem. If we were able to address the weighting, the first problem would also vanish, as the weights would presumably depend on the measurement unit and make the ranking unit-invariant.

One popular approach to address this problem is to use a metric that is derived from data. There are several popular approaches, all of these normalize the units with respect to certain variation in data. However, despite that these distance metrics are “data driven”, they are not necessarily more correct than other units. Sometimes the preferred metric can be deduced from the nature of the problem, but other times one has just to experiment and find the best approach.

Feature normalization

Perhaps the most popular approach is to *normalize* the features: transform the features into mean-zero and variance-one version. This would constitute an answer to the house-price-problem along these lines: “we think that customers value one standard deviation in house size about the same as one standard deviation in the neighborhood crime rate”.

Consider a design matrix $\mathbf{X} = (\mathbf{x}_{\bullet 1}, \mathbf{x}_{\bullet 2}, \dots, \mathbf{x}_{\bullet K})$ where $\mathbf{x}_{\bullet j}$ is the j -th column (feature) of \mathbf{X} . Hence $\mathbf{x}_{i\bullet} = (x_{i1}, x_{i2}, \dots, x_{iK})^\top$ is the feature vector for observation i . Now instead of using the original features $\mathbf{x}_{i\bullet}$, we transform these into

$$\tilde{\mathbf{x}}_{i\bullet} = \frac{\mathbf{x}_{i\bullet} - \bar{\mathbf{x}}_{\bullet j}}{\text{sd } \mathbf{x}_{\bullet j}} \quad \forall i \in 1 \dots N, j \in 1 \dots K. \quad (3.2.1)$$

Here $\bar{x}_{\bullet j} = \frac{1}{N} \sum_{i=1}^N x_{ij}$ denotes the average value of the feature j , and $\text{sd } \mathbf{x}_{\bullet j}$ is, in a similar fashion, the standard deviation over i of all x_{ij} -s. N is the total number of cases. Obviously, each feature component $\tilde{x}_{\bullet j}$ is now mean zero and standard deviation of 1. Note that normalization is done for each feature separately, but using all the observations for that component. The result, obviously, does not depend on the units any more as both the numerator and denominator in 3.2.1 are scaled by an equal amount when we change units. We can say that we are introducing new unit, $\text{sd } \mathbf{x}_{\bullet j}$, to measure feature j .

Example 3.2.1: Data normalization

Consider the matrix below. It contains three columns, $x_{\bullet 1}$, $x_{\bullet 2}$ and $x_{\bullet 3}$. The first two have similar spread (2) but different means (2 and 12 respectively), while the third one also has a different scale.

	$x_{\bullet 1}$	$x_{\bullet 2}$	$x_{\bullet 3}$
1	1	11	10
2	2	12	20
3	3	13	30
mean $\bar{x}_{\bullet j}$	2	12	20
std.dev $x_{\bullet j}$	1	1	10

The table also lists the corresponding mean values and (sample size corrected) standard deviations. When normalizing the data, we simply subtract the corresponding mean from each variable in data, and divide the resulting difference by the corresponding standard deviation:

	$x_{\bullet 1}$	$x_{\bullet 2}$	$x_{\bullet 3}$
1	1	-1	-1
2	2	0	0
3	3	1	1

One can see that all three variables are now equal—they are all $(-1, 0, 1)$. This is rather intuitive: they all have one observation in the middle, and two other in each side of the mean by an equal amount.

Another comparison of normalized and non-normalized features is given in Figure 3.1. The left panel depicts the original feature space where spread of \mathbf{x}_2 is much larger than that of \mathbf{x}_1 . The dotted circle denotes a set of equidistant points (using Euclidean distance in \mathbb{R}^2) from the dark blue point in its center. One can see that the circle encompasses the green dot but not the yellow dot—hence the dark blue dot is closer to the green dot than to the yellow dot. On the right panel we see the normalized version of the same data. The visual impression confirms that both features are spread roughly equally. The solid circle depicts a set of equidistant points from the dark blue dot in the new feature space. In this space, the dark blue dot is closer to the yellow dot than to the green one. Both panels also show the circles in the other feature space, now transformed to ellipses.

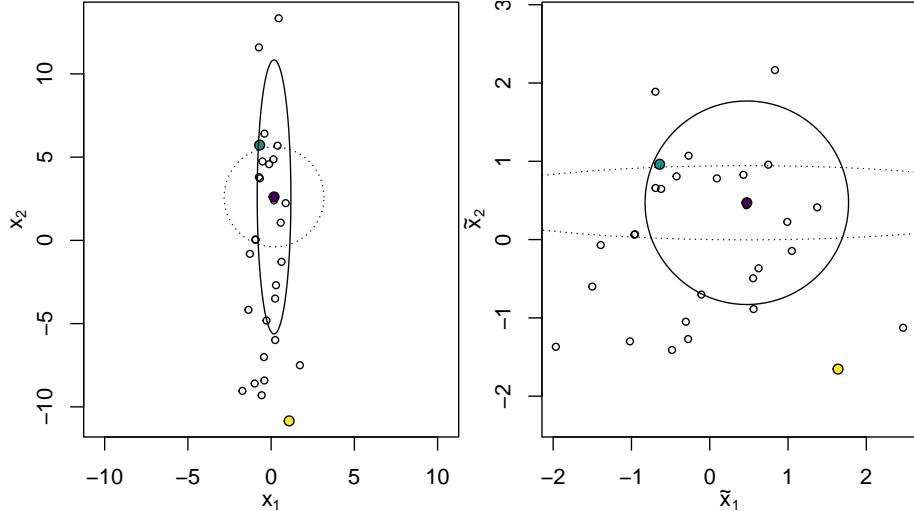


Figure 3.1: Non-normalized features (left) and normalized features (right). The same three cases are marked with different red on both images. The dotted line depicts a circle in the original feature space, the solid line is circle in normalized feature space.

Figure 3.2 gives a related example using Boston housing data. Both the left and right panel depict exactly the same data: neighborhood crime rate versus

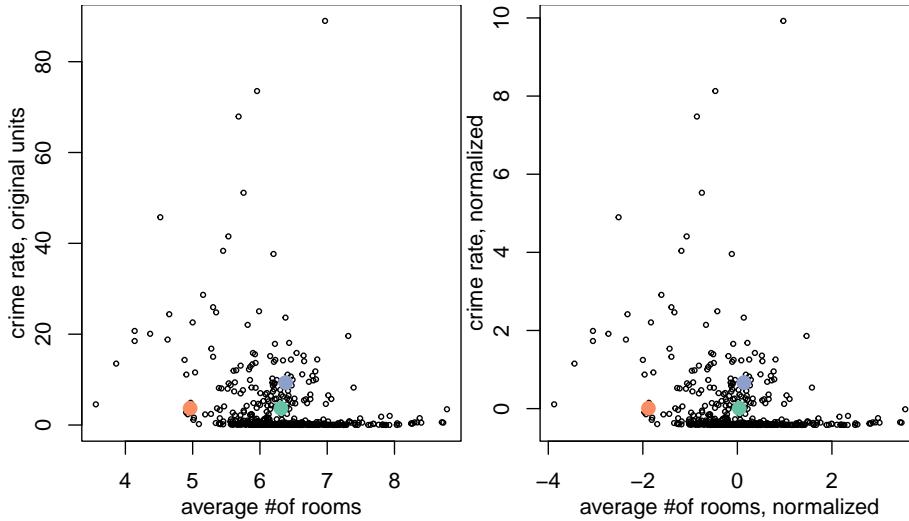


Figure 3.2: Boston housing data: neighborhood crime rate (crim) versus average number of rooms (rm). Non-normalized (left) versus normalized features (right). While the images look exactly the same, the Euclidean distance rankings are different: the nearest (colored) neighbor the the green neighborhood is the red on the left, and blue on the right panel.

the average number of rooms. However, on the left panel we use the original features while on the right panel we use normalized features. Although the images remain exactly the same, the Euclidean distances differ. For instance, the distance between the green, and the orange and blue neighborhoods is 1.349 and 5.666 on the left panel, and 1.92 and 0.666 on the right panel. Hence the closest colored neighbor to the green on the left figure is the orange, and on the right figure it is the blue.

TBD: a better figure where one can see scaling

From technical point of view, normalization is a good option if the features are roughly independent, and their distribution is roughly symmetric and does not have fat tails. This assures that the variance is stable and the mean is in the middle of the observations.

More conceptually, normalization is justified in such cases where standard deviation is a relevant scale unit. In case of house price example, this is the case if people consider both house size and neighborhood security a relevant measure, and standard deviation of the respective variables is a good proxy for how people value these two factors. However, if the customers never care about crime, except for the worst few neighborhoods, feature normalization may not be a good approach.

Another common usage case is to transform values that are measured in arbitrary and hard-to-understand units into more easily understandable (and

comparable) ones. For instance, we may ask people to report their support for a government policy on a scale from 1 (very much against it) to 5 (very much in favor of it). One unit in this scale is hard to understand, but a claim like “those whose support is one standard deviation above the mean...” carries more meaning.

There is one more technical reason why it is advisable to normalize features—if the design matrix contains columns of very different scale, its condition number will be high and hence the convergence may suffer.

Min-max scaling An easy alternative to normalization is min-max scaling. It is conceptually similar to normalization, just instead of normalizing by the standard deviation of the feature column, we divide by its range. For example, we can create a new set of features that are all in equal range as follows:

$$\tilde{\mathbf{x}}_i = \{\tilde{x}_{ij}\} = \frac{x_{ij} - \min_i x_{ij}}{\max_i x_{ij} - \min_i x_{ij}} \quad \forall i \in 1 \dots N, j \in 1 \dots K, \quad (3.2.2)$$

where min and max are taken over the particular feature $x_{\bullet j}$. In this example all the features will be converted into $[0,1]$ interval, but we can shift and scale these into another interval instead, say $[-0.5, 0.5]$, if needed. Min-max scaling works well if the features are independent and have uniform-like distribution with no tails—all values end abruptly at the boundary. This ensures that the minima and maxima are stable.

As min-max scaling is very similar to the feature normalization, its advantages and disadvantages are similar too.

Mahalanobis distance Prerequisites: [Vector Norm 2.2.2](#), [Eigenvalues and eigenvalue decomposition 2.3.4](#), [feature normalization 3.2.1](#), covariation matrix.

This is a generalization of feature normalization in case where the features may be correlated. Consider Figure 3.3. Here the two features \mathbf{x}_1 and \mathbf{x}_2 do not just have a different variance, they are also clearly correlated. If we use just feature normalization, we will change the picture somewhat, but we cannot address the fact that the data points are clearly clustered around the diagonal line.

Mahalanobis transformation, in contrast, stretches and rotates the data in a way that is aligned with the axis of the data. In the left panel of Figure 3.3, the solid ellipse depicts the equidistant points from the central dark blue dot. The ellipse is elongated along the long axis of the correlated data, and compressed along its short axis. So Mahalanobis distance measures distance with respect to the extent of the point cloud in each particular direction, not just along the coordinate axes as is the case with feature normalization.

Mahalanobis distance can be done and understood easily using matrix notation and eigenvalue decomposition. Consider \mathbf{X} to be a $N \times K$ data matrix and $\mathbf{x}_{i\bullet}$ and $\mathbf{x}_{j\bullet}$ to be two rows (observations) from that data. The Mahalanobis

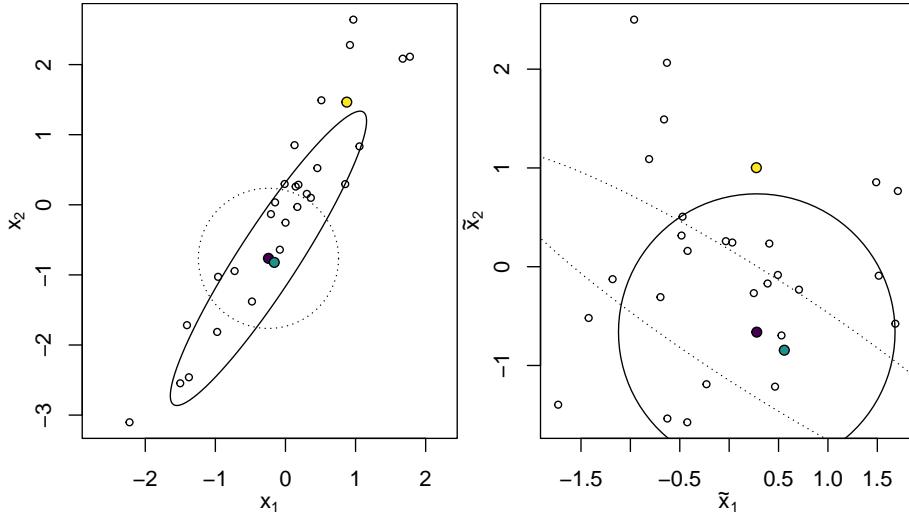


Figure 3.3: Original features (left) and Mahalanobis-transformed features (right). The same three cases are marked with different colors on both images. The dotted line depicts a circle in the original feature space, the solid line is circle in Mahalanobis feature space.

distance between the observations $\mathbf{x}_{i\bullet}$ and $\mathbf{x}_{j\bullet}$ is defined as

$$d_E(\mathbf{x}_{i\bullet}, \mathbf{x}_{j\bullet}) = \sqrt{(\mathbf{x}_{i\bullet} - \mathbf{y}_{\bullet})^\top \Sigma^{-1} (\mathbf{x}_{i\bullet} - \mathbf{x}_{j\bullet})} \quad (3.2.3)$$

where Σ is the covariance matrix of \mathbf{X} .

Mahalanobis distance is equivalent to transforming the data matrix into

$$\tilde{\mathbf{X}} = \left(\mathbf{X} - \mathbf{1}_N \cdot \bar{\mathbf{x}}^\top \right) \Sigma^{-\frac{1}{2}} \quad (3.2.4)$$

where $\bar{\mathbf{x}}$ is the vector of column means and $\mathbf{1}_N \cdot \bar{\mathbf{x}}^\top$ is accordingly a matrix of column means.

Mahalanobis transformation is essentially the same as transforming data to [principal components](#) (see Section 10.3). Thereafter it measures Euclidean distance in such a rotated and stretched feature space. In case the features are uncorrelated, Mahalanobis distance is equivalent to Euclidean distance in normalized data.

Mahalanobis distance is a good measure for data where the data variation is a meaningful distance measure, and not just along the features as in case of normalization, but also along the axes of variation in data.

Example 3.2.2: Mahalanobis transformation of iris data

We demonstrate the Mahalanobis transformation using Iris data. Figure 3.4 shows an analogous transformation as in Figure 3.3 for 2-D slice of iris data. The different species, denoted by different colors, are reasonably well separated on both figures. However, in the original coordinates (petal length and width, left panel) the data points form an elongated cloud where the different species cluster at different location in this cloud. In transformed coordinates, the points are stretched out along the minor axis, increasing the distance between dots for similar species.

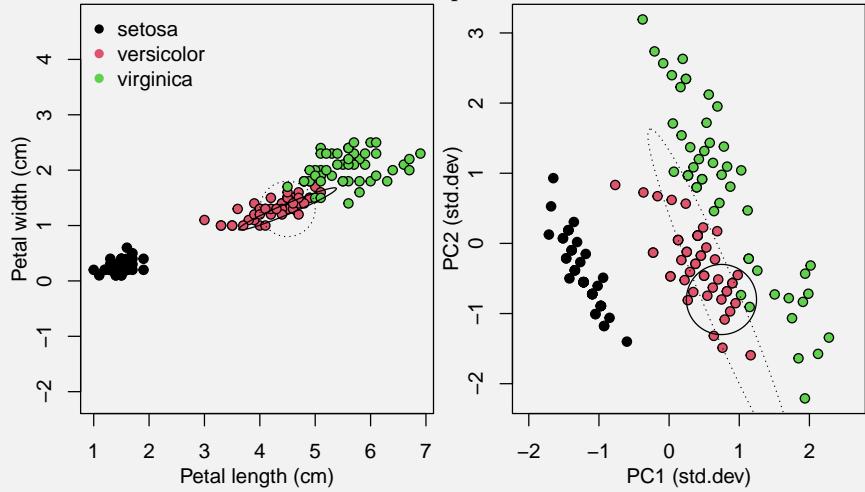


Figure 3.4: Iris data: petal width versus petal length in the original coordinates (left panel) and in the corresponding Mahalanobis-transformed coordinates (right panel).

In transformed coordinates the species do not form as tight clusters any more as in the original coordinates, making categorization more difficult. This fact is also visible from the example circles, the circle that centers on a red observation in Mahalanobis coordinates (solid line) includes two green points while the circle in the original coordinates (dotted line) includes only a single green dot while capturing most of the red ones. For k -NN to work well, it should be possible to draw such circles around most datapoints that contain many dots of the correct color and few of other colors. This is easier in the original coordinates.

Note that one of the major reason for data transformation, features measured in incomparable units, does not hold here as both features are measured in centimeters.

3.2.2 Cosine similarity and angular distance

Prerequisites: [Vector Norm 2.2.2](#)

Cosine similarity is a similarity measure that is not based on L_p distance. It is widely used when assessing similarity in features that are not numeric, such as when comparing texts.

Cosine similarity between vectors \mathbf{x} and \mathbf{y} is defined as

$$c(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} \quad \mathbf{x} \neq \mathbf{0}, \mathbf{y} \neq \mathbf{0}, \quad (3.2.5)$$

where $\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \cdot \mathbf{x}}$ is the Euclidean norm. It is easy to see that $c(\mathbf{x}, \mathbf{x}) = 1$.

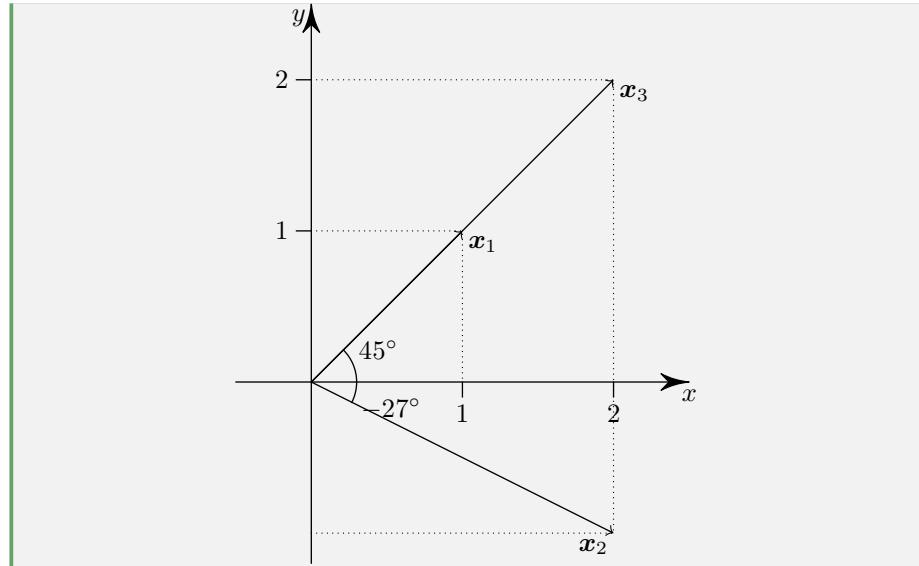
It's name, *cosine* similarity, originates from the fact that inner product of vectors equals to the product of their norms, multiplied by the cosine of the angle between them:

$$\mathbf{x}^\top \cdot \mathbf{y} = \|\mathbf{x}\| \cdot \|\mathbf{y}\| \cdot \cos \phi, \quad (3.2.6)$$

where ϕ is the angle between vectors \mathbf{x} and \mathbf{y} . Hence cosine distance equals just to the cosine of the angle between the vectors. Note that it is *solely the angle* between the vectors. Cosine distance is agnostic to the length ([norm](#)) of the vectors (as long as this is positive). It is a measure in similarity in direction the vectors point to, and not a measure of the length of the vectors. For instance, when analyzing texts using bag-of-words (see Section 8.1.3), this amounts to comparing word frequencies in the texts. The number of words (text size) is irrelevant. Such an approach may be very well suited when we try to understand the topic of the text while the texts itself may be of very different size.

Example 3.2.3: Cosine similarity in \mathbb{R}^2

The easiest way to understand cosine similarity is to analyze it in \mathbb{R}^2 plane. Look at the vectors \mathbf{x}_1 and \mathbf{x}_2 on the figure below. $\mathbf{x}_1 = (1, 1)$ and hence it points 45° upward. $\mathbf{x}_2 = (2, -1)$ and accordingly points 27° downward, and hence the angle between the two vectors is $45 + 27 = 72^\circ$.



Now calculate cosine similarity. First, the Euclidean norms are

$$\begin{aligned}\|\mathbf{x}_1\| &= \left\| \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\| = \sqrt{\begin{pmatrix} 1 \\ 1 \end{pmatrix}^\top \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix}} = \sqrt{1+1} \approx 1.414 \\ \|\mathbf{x}_2\| &= \left\| \begin{pmatrix} 2 \\ -1 \end{pmatrix} \right\| = \sqrt{\begin{pmatrix} 2 \\ -1 \end{pmatrix}^\top \cdot \begin{pmatrix} 2 \\ -1 \end{pmatrix}} = \sqrt{4+1} \approx 2.236.\end{aligned}\quad (3.2.7)$$

Now we can plug the numbers into the cosine similarity definition (3.2.5):

$$c(\mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{x}_1^\top \cdot \mathbf{x}_2}{\|\mathbf{x}_1\| \cdot \|\mathbf{x}_2\|} \approx \frac{\begin{pmatrix} 1 \\ 1 \end{pmatrix}^\top \cdot \begin{pmatrix} 2 \\ -1 \end{pmatrix}}{1.414 \cdot 2.236} = \frac{2-1}{3.162} = 0.316. \quad (3.2.8)$$

We can easily check that 0.316 is cosine of 71.6° . Hence the computed cosine similarity is equal to the cosine of the angle between \mathbf{x}_1 and \mathbf{x}_2 . (The difference is related to rounding errors.)

TBD: Example where two vectors of different size are at same similarity with a third one

Cosine similarity has a few very favorable properties, in particular it is easy to compute, involving just multiplications, additions, and one division. In case of sparse matrices, only non-zero components need to be considered. All this makes it very well suitable for analyzing high-dimensional data, such as words in texts.

Unlike the distance measures above, cosine similarity is not a metric distance as larger value means not more distant but more similar data vectors. The

maximum similarity, distance between identical vectors is 1 while the minimum similarity, distance between opposite vectors, is -1. This is sufficient to order vectors according to their similarity, and often this is all we need.

In case one needs a difference measure instead of similarity measure, one can use cosine distance $d_{cos}(\mathbf{x}, \mathbf{y}) = 1 - c(\mathbf{x}, \mathbf{y})$. Cosine distance is zero in case of vectors that point in the same direction, the maximal possible distance is 2 when two vectors point in exactly opposite direction. Another option is to use *angular distance*, defined as

$$d_a(\mathbf{x}, \mathbf{y}) = \frac{\cos^{-1} c(\mathbf{x}, \mathbf{y})}{\pi}, \quad (3.2.9)$$

instead of cosine distance. However, there is little gain from selecting a more computationally demanding metric if our task is just to rank vectors according to similarity.

Exercise 3.2.1: Cosine, angular distance are not proper metric distances

Show that neither cosine nor angular distance are proper [metric distances](#) as defined in Section [2.2.2](#).

Chapter 4

Regression Models

4.1 Linear Regression

4.1.1 The Problem: Why We Want Linear Regression

In both research and applied analysis we are often interested in relationship— are larger values of x associated with larger or smaller values of y ? Or maybe we know that the relationship exists but we may want to quantify it—how much larger values of y are associated with larger values of x ?

As a example, let's analyze the length and width of iris sepals (from iris data). Figure 4.1 (right panel) displays the length and width of the sepals for *setosa*, an iris species. Here it is not surprising at all to see that longer leaves are wider too. However, the exact form of the relationship may not be obvious. As leaves get longer, do they get wider at an increasing pace and become more rounded? Or perhaps the way around—very long leaves are more elongated? And sometimes we do not have any particular reason to expect to see an increasing or decreasing relationship. In any case, we can plot the data like here and ask see what kind of relationship do we see here.

More formally, we sometimes want to see and test if two variables are related. Are these variables related? How strong is the relationship? Are x and y more strongly related than x and z ? Alternatively, we may know the value of one variable and want to use this knowledge to predict the value of something else. For instance, we may want to predict house price based on its size. But there are many ways how two variables can be related. Figure 4.2 shows a few different options. Which of these curves is “correct”? Which of these is “better”?

Obviously, there is no general answer to the “correct” and “better” question. It depends on the process, data and the problem. But we want to construct a tool that can be used to assess the following questions:

- are these two variables related? Yes or no?
- how strong is the relationship? For many real world problems, such as prediction, it is not just enough to say that there is a relationship, we

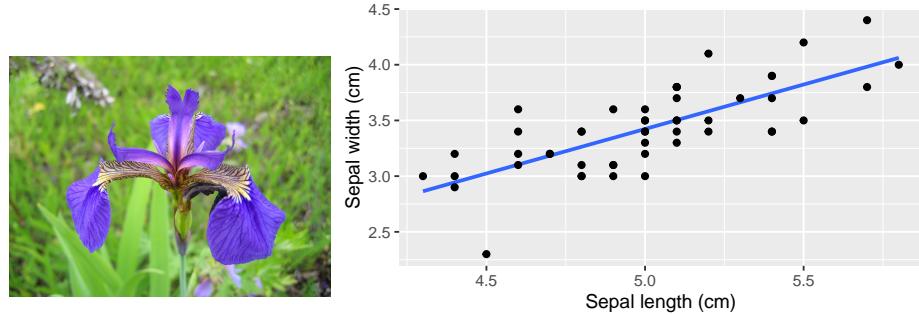


Figure 4.1: Flower of iris setosa (left). Sepals are the big purple falls spreading downward (the thinner upward growing ones are petals). The right panel displays the relationship between the width and length of sepals. Unsurprisingly, longer petals area also wider. The blue trend line is computed using linear regression. Flowers of setosa species in iris data.

need a numeric value.

- is the number statistically significant? Maybe it is just by chance that the numbers look related?
- how does the relationship in one dataset compare to that in the other dataset? Is it stronger or weaker?
- and finally, the tools should be intuitive and easy to use.

We stress here that in order to answer the questions above we need a mathematical tool. Just eyeballing the data and deciding which curve is the “best” is not precise enough and does not scale (but it is an almost necessary starting point!). The tool should have clear mathematical formulation and clear assumptions so we can judge if it is appropriate in each case. It should also be flexible, allowing various tweaks to be incorporated to address problems of different flavor. And finally, it should be simple to implement and use on computer.

4.1.2 Simple Regression

TBD: History

Introduction

Linear regression is perhaps the most popular tool to answer these questions. It checks all the boxes in the list above offering both yes/no-style answers and quantitative answers. It is also simple, intuitive, and easy to use.

Linear regression is a statistical model. Here the *model* means a specification how the endogeneous variable y is related to the explanatory variables x . A

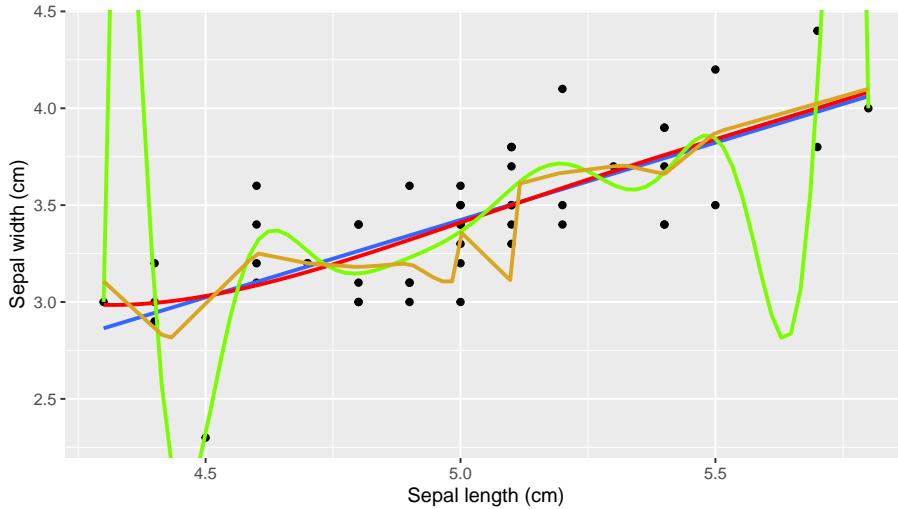


Figure 4.2: The same data as in Figure 4.1. Besides the original trend line (blue), several other possible relationships are shown.

model is a necessary tool if we want to “ask data” about the true relationship. In typical applications we consider here, we need a *statistical model*, a model that contains both a deterministic and a stochastic part. The former is what we are (mainly) interested in, the part of the relationship we can reliably describe and use for inference and prediction. The other part is rarely used beyond evaluating model’s performance, but it is a necessary component that takes care of the stochastic nature of data. In large classes of common applications we simply cannot reliably predict the outcome based on the observed features. A statistical model helps to handle such unreliability in a consistent and precise manner.

In regression models we describe the value of the *outcome variable* y using *explanatory variable* x . There are many other way to call these variables, e.g. *endogenous variable*, *dependent variable*, *target*, or just “ y ” for the outcome, and *exogenous variables*, *independent variables*, *features*, *predictors*, or just “ x ” for x .

Note that this approach means data is treated in a fundamentally asymmetric way as data is partitioned into explanatory variables and the endogenous variable. Sometimes this is a natural approach, for instance if our task is to predict y based on x , or if we are interested how x affects (causes) y . In other cases, it may be less relevant. For instance, it is not obvious why we should treat width and length of leaves in an asymmetric manner.¹

¹Such asymmetric treatment of data is common to a large class of models, commonly called *supervised learning* methods in machine learning literature. Certain other methods (*unsupervised learning*), for instance clustering or principal component analysis do not follow

Despite of being an old (over 200 years) method, it is still immensely popular, and it is hard to see anything replacing it any time soon. Linear regression is definitely not everything a data scientist has to know, there are just too many problems (for instance, natural language processing or image analysis) that cannot be tackled with linear regression. But linear regression wins almost always in terms of simplicity and interpretability. It also forms either a benchmark or a component for many other statistical models, including neural networks.

Setup

The linear regression model is defined as

$$y_i = \beta_0 + \beta_1 \cdot x_i + \epsilon_i. \quad (4.1.1)$$

Here y is the outcome variable, x is the explanatory variable, and ϵ is the *error term* (also called *disturbance term*, or *noise term*). The index i indicates individual observations, typically rows in the data frame. It stresses that each observation i has a different value for y , x and ϵ , but they all share the same parameters β_0 and β_1 .

TBD: Interpret/explain ϵ

Example 4.1.1: Linear regression with Iris data

Let us demonstrate the linear regression using Iris data, the same dataset that was used in Figure 4.1. The first few lines of the data look like

Table 4.1: Example cases from Iris dataset

Sepal length	Sepal width
5.1	3.5
4.9	3.0
4.7	3.2
4.6	3.1
5.0	3.6

Let's pick sepal width as the outcome y and sepal lenght as the explanatory variable x . So we can put (4.1.1) in a more specific form as

$$\text{Sepal width}_i = \beta_0 + \beta_1 \cdot \text{Sepal length} + \epsilon_i. \quad (4.1.2)$$

This is what we know. We know all x and y values and we know our model, but we don't know β_0 , β_1 and ϵ .

When estimating the model we find the best values of β_0 and β_1 where *best* means the best in the linear regression sense. The solution here is $\hat{\beta}_0 = -0.569$ and $\hat{\beta}_1 = 0.799$ (see more in [4.1.10 Solving the Linear Regression Model](#)). The “hat” on top of $\hat{\beta}$ stresses that these are not “true” values but our best estimates based on data. So we can rewrite the definition (4.1.1)

this distinction.

for the first few observations in the data as

$$\begin{aligned} 3.5 &= -0.569 + 0.799 \cdot 5.1 + e_1 \\ 3 &= -0.569 + 0.799 \cdot 4.9 + e_2 \\ 3.2 &= -0.569 + 0.799 \cdot 4.7 + e_3 \\ &\dots \end{aligned} \tag{4.1.3}$$

Note that the parameter values (β_0 and β_1) are the same for all observations i , but the variable values differ for each i . We have replaced ϵ by e to stress that we don't know the correct ϵ , but we can estimate e from (4.1.3).

Example 4.1.2: How fast does the Universe expand?

By early 20th century, it was clear that certain nebulae in sky are outside of our Milky Way galaxy and astronomers attempted to use those to determine the Solar motion in space. By late 1920s, there was already data for both velocity and distance for 24 “extragalactic nebulae”, i.e. galaxies. In 1929, Edwin Hubble published a paper where he plotted velocity versus distance for those 24 objects (Hubble, 1929).

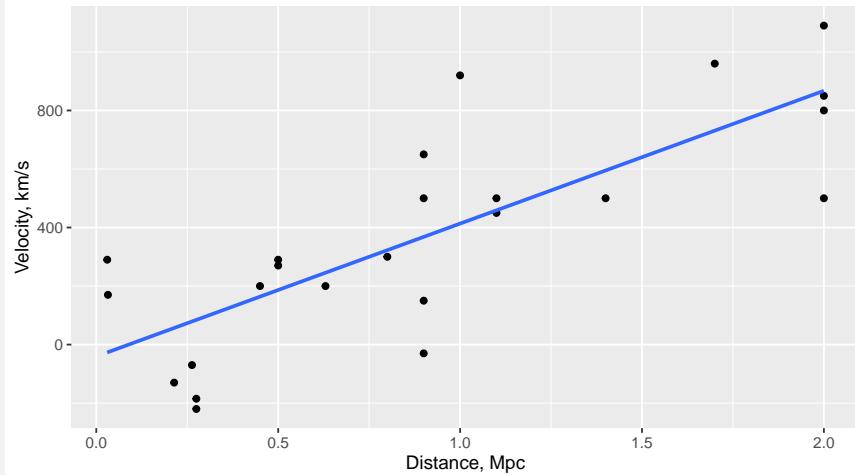


Figure 4.3: The original Hubble diagram. Hubble estimated the slope to be approximately $500 \text{ km s}^{-1} \text{ Mpc}^{-1}$. Despite the less-than-impressive data, and the fact that he badly overestimated the slope, it is considered one of the most important cosmological discoveries of all time.

On the figure, the more distant galaxies are clearly moving faster away from us. This suggests that the Universe is expanding, Hubble estimated the expansion rate to be approximately $500 \text{ km s}^{-1} \text{ Mpc}^{-1}$ (modern estimates are approximately $72 \text{ km s}^{-1} \text{ Mpc}^{-1}$).^a The rate, nowadays called “Hubble constant”, can be estimated from the same data using linear regression:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-40.4360	83.4480	-0.48	0.6328
R	453.8600	75.2457	6.03	0.0000

The estimate of distance R (measured in Mpc) indicates that 1 Mpc more distant galaxies move 454 km/sec faster (away from us). We can reverse this rate and ask “how long time it takes for a galaxy, moving 454 km/sec, to reach to 1 Mpc = 3.09×10^{19} km distance? This gives us roughly 2 billion years, the number of years since the Big Bang (modern estimates are 13.8 billion years). This was an important piece of evidence supporting the idea that the Universe is young.

^aparsec (pc = 3.09×10^{13} km or 3.26 light years) is a distance where the Earth orbit's radius is visible as 1" arc. The closest stars are 1.3pc away from us, Milky Way disk is 50,000pc in diameter. Mpc = 1 000 000 parsec.

The two essential parts of the model is the deterministic part $\beta_0 + \beta_1 x$, and the stochastic part ϵ . In most applications we are primarily interested in the deterministic part. By itself it describes y as a linear function of x because as we have $y = \beta_0 + \beta_1 \cdot x$, the modeled x - y relationship will form a straight line on graph. Parameter β_0 is called *intercept* (also *constant*), parameter β_1 is commonly called *slope*, but often one refers to both parameters together as “betas” or “coefficients”.

In the situation where we typically use linear regression (and other statistical models) we usually do not know the “correct” values of β_0 and β_1 but we know our data, i.e. all the explanatory and outcome variable pairs (y_i, x_i) for $i = 1 \dots N$. Hence our first task is to find β_0 and β_1 (see more in [4.1.10 Solving the Linear Regression Model](#)) before we can use the model for anything else. Sometimes we are interested in the the parameter values itself as these may carry policy-relevant meaning (see more in [Interpretation](#)). In other cases we do not care much about the betas, but want to use those to predict other interesting outcomes, such as y values (see more in [Prediction](#)).

Prediction

Prerequisites: Expectation, expectation as a linear operator

We start with explaining prediction as it is useful to be able to predict y values to understand interpretation.

Imagine we have somehow figured out the “right” values of β_0 and β_1 . What would be our predicted y_i , typically denoted by \hat{y}_i ? Let’s look at the model definition ([4.1.1](#)) again. We know the values of β_0 and β_1 (we just figured these out). We also know x_i , $i = 1 \dots N$ (this is our data). But we don’t know ϵ_i , as that is an unobserved stochastic error. We have no idea about its true value and hence we cannot compute y_i . But we can compute its expectation $\mathbb{E} y_i$

instead, given we assume something about the error terms:

$$\mathbb{E} y_i = \mathbb{E}[\beta_0 + \beta_1 \cdot x_i] + \mathbb{E} \epsilon_i. \quad (4.1.4)$$

As β_0 , β_1 and x_i are known values, their expectations are just these values. We are left with $\mathbb{E} \epsilon_i$. Normally we just assume

$$\mathbb{E} \epsilon_i = \mathbb{E} \epsilon = 0. \quad (4.1.5)$$

What does this assumption mean?

- The first part of the assumption, $\mathbb{E} \epsilon_i = \mathbb{E} \epsilon$, means that the expected value of ϵ is equal for all observations. Typically we also add that they are distributed in the same way for all i , but that is less important.
- The second part, $\mathbb{E} \epsilon = 0$, means that we assume that it's expectation is exactly zero, and hence its mean in a finite sample (like our data) is also close to zero. This may sound like a strong assumption but it is actually pretty harmless in most cases. Were we assuming something else, say $\mathbb{E} \epsilon = a$ for some constant a , this would amount of shifting y values up by a . But we already have the intercept term, β_0 that plays a similar role and shifts y values up and down. As a result, the β_0 would decrease by amount a , so that $\beta_0 + a$ will remain constant. Our predictions would not change.²

With these assumptions in place, we can just write our predictions as

$$\hat{y}(x) = \mathbb{E} y_i = \beta_0 + \beta_1 \cdot x. \quad (4.1.6)$$

It may be written different forms, for instance

$$\hat{y}(x_i) \equiv \hat{y}_i \equiv \hat{y}(x_i; \beta_0, \beta_1) \quad (4.1.7)$$

where the first form make dependency on x explicit, the second form uses the index i for a short-hand notation, and the third version also indicates the prediction depends on the model parameters.

In cases where we know the true values y_i (for instance, on training data), we can also compute the prediction errors, typically called *residual terms*, *deviations*, or *residual errors*:

$$e_i = y_i - \hat{y}_i = y_i - (\beta_0 + \beta_1 \cdot x_i). \quad (4.1.8)$$

Obviously, the better the model, the smaller are the residual terms. But in general we face trade-offs—we cannot make residual error for one observation smaller without making it larger for another observation at the same time.

²Here we assume that the model in fact includes the constant term. If this is not the case, the assumption may have major implications. See

TBD: reference to the example

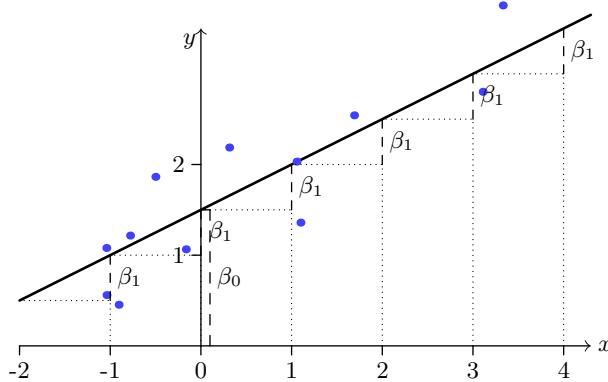


Figure 4.4: Interpretation of regression parameters. The blue dots represent the data points and the thick line is the regression line. Intercept β_0 represents the vertical intercept of the thick regression line, i.e. the predicted y value at the point where $x = 0$. Slope β_1 corresponds to the “climb” of the line when x increases by one unit. Random data.

Note that here we are predicting the expected value of y , $\mathbb{E}y$. We may also choose to predict something else instead, e.g. median or other quantiles of y , its minimum value, or probability that y is positive.

TBD: example where we predict something a little bit realistic (Hubble data?)

Interpretation One of the big advantages of linear regression is its interpretability. There are other interpretable models, such as logistic regression, but none can compete with linear regression in terms of ease and simplicity. In many situations we are less interested in the predicted values and more interested in understanding the underlying process, and in such cases linear regression is often the obvious choice.

To interpret a model means to “understand” the parameter values and being able to tell a story what do these values mean. Simple regression has two parameters, intercept β_0 and slope β_1 . The meaning of these parameters can easily be understood when analyzing the predicted values. From (4.1.6) we see that if $x = 0$, the predicted $\hat{y}(0) = \beta_0$. Hence intercept indicates the expected (predicted) value of y if $x = 0$ (See Figure 4.4). To understand what does slope, β_1 , describe, we can compute

$$\hat{y}(x+1) - \hat{y}(x) = [\beta_0 + \beta_1 \cdot (x+1)] - [\beta_0 + \beta_1 \cdot x] = \beta_1. \quad (4.1.9)$$

Hence slope tells us how much does our prediction \hat{y} grow when x increases by 1 unit.

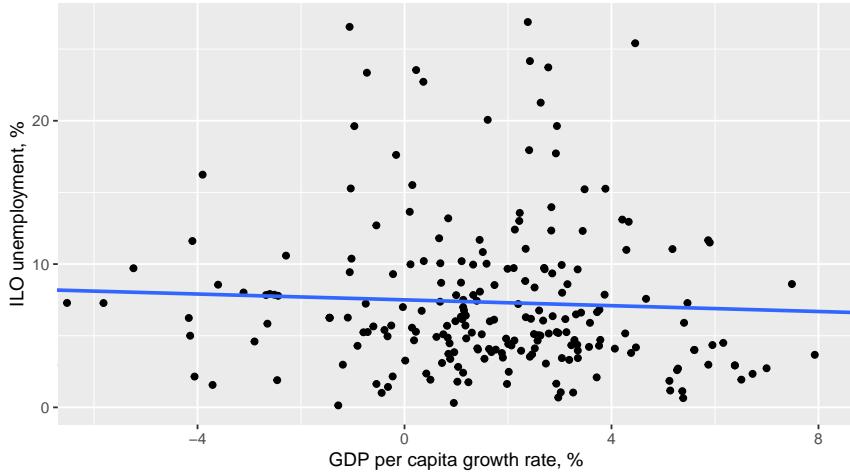
Example 4.1.3: Unemployment versus GDP growth

Figure 4.5: Relationship between unemployment and GDP growth across countries in 2016, and the corresponding regression line. World Bank data.

Figure 4.5 shows the relationship across countries between unemployment and GDP growth where unemployment is measured as percentage of labor force, and GDP growth in percentages. We can model the relationship as

$$\text{unemployment}_i = \beta_0 + \beta_1 \cdot \text{GDP growth}_i + \epsilon_i \quad (4.1.10)$$

where i denotes different countries. The corresponding linear regression results are

	Estimate	Std. Error	t-value	Pr(> t)
Intercept	7.503	0.380	19.728	0.000
growth	-0.102	0.103	-0.988	0.324

The column *Estimate* shows the estimated values, $\hat{\beta}_0$ and $\hat{\beta}_1$. Here “intercept” means that expected unemployment for a zero-growth country is 7.6 percent. For each additional percent of growth, that number falls by 0.1 pct points. For instance, if economic growth is 2%, the model predicts the unemployment rate to be 7.4%. The estimate for *growth* seems surprisingly small (and is not statistically significant), but remember the data describes the cross-section of countries in 2016, a period of rather robust growth, and not relationship over time for an individual economy.

Note that linear regression (nor any other statistical model) does not allow to make causal claims. The *growth* estimate -0.102 cannot be interpreted that more growth *causes* less unemployment, at least not based on this data.

Note that while these interpretations are always correct in the mathematical sense, they may sometimes carry little meaning. For instance, the regression line in Figure 4.1 is given by parameters $\beta_0 = -0.569$ and $\beta_1 = 0.799$. The intercept means that zero-length sepals are -0.569 cm wide. This does not make any sense, but as none of our flowers have sepal length less than 4cm, it does no harm when we use our model for the actual 4-6cm long flowers. But extrapolation for small flowers may be very misleading as this example suggests. The slope parameter β_1 means that for each unit (i.e. centimeter) sepals are longer, they are 0.799 units (i.e. centimeters) wider in average. This number is reasonable and tells us something about the shape of the flowers.

Table 4.2: Software output table from sepal length–sepal width regression. Different software package label the columns slightly differently, and may provide slightly different outputs, but the main information is very much the same.

	Estimate	Std. Error	t-value	Pr(> t)
Intercept	-0.569	0.522	-1.091	0.281
‘Sepal length’	0.799	0.104	7.681	0.000

Interpreting the regression table The statistical software we use for linear regression typically outputs not just coefficient values but a complete table of results. Table 4.2 shows an example of such a table, computed fro the *setosa* sepal length–sepal width regression. The first column, “Estimate”, presents the same estimated coefficients we discussed above. Here we discuss the other columns in this table. As it turns out, all these columns are very important.

The second column in the table is labelled “Std. Error”. This is the standard error of the estimate. As the points do not line up exactly, we need to include certain randomness in the model (this is term ϵ in (4.1.1)). Intuitively, depending which data points we exactly measure, our regression coefficients will be slightly larger or slightly smaller. Under mild assumptions, these coefficients follow t -distribution (t -distribution is rather similar to normal distribution) and this column provides their standard error. You can imagine we sample the data from population many-many times. Each time we get a slightly different sample, and hence we get slightly different estimated values. Standard error describes the variability of the estimates obtained in this way. We don’t want to (or even cannot) do many samples, so “Std. Error” is computed using the mathematical properties and underlying assumptions instead.

In this table we can see that the intercept’s standard error is 0.522 while the sepal length coefficient’s error is 0.104. Hence the latter is much more precisely determined by our data than the former.

The next column is labelled “ t -value”. This is the t -value for the coefficient:

$$t = \frac{\text{Estimate}}{\text{Std. Error}}. \quad (4.1.11)$$

It is a number computed just from the two previous columns. This is related to the most common hypothesis test that is done in context of linear regression: H_0 : Estimate = 0. You can imagine the data where x and y (i.e. sepal length and sepal width) are not related. But just because randomness in the data, we always see a slight relationship. One can show that if H_0 is correct, the t value is t -distributed. Large t values are unlikely under H_0 . In the table the intercept's t -value is -1.091 and that for sepal length is 7.681. Hence it is much more likely to see such an intercept just by random chance.

Finally, the last column “ $\text{Pr}(>|t|)$ ” shows just how likely it is to get such a t -value under H_0 . This number is computed from the “ t -value” column. The probabilities are 0.281 and 0, so just by playing with random data, we can get an intercept of similar size in more than 25% of cases. However, the chances of getting similar value for sepal length coefficient are much smaller and essentially 0.

TBD: Example with intercept 0

Correlation and causation One has to keep in mind that these parameters cannot be interpreted causally. The regression parameter that connects sepal width and length cannot be interpreted as for every centimeter the sepal grows in length, it grows 0.799 centimeters in width. Linear regression³ only computes the average relationship: in our data, longer leaves are also wider. Data alone do not tell why. Human language easily slips into semi-causal interpretation, and humans are prone to interpret a relationship causally even when explicitly stated that this may not be true. Occasionally this leads to a completely wrong interpretation. The causality-agnostic language has a special phrase, *associated with*, to denote the correlational relationship like what is computed in linear regression. So our sepal results may be phrased as *1 cm longer leaves are associated with 0.799 centimeters more width*.

Formal Definition of Linear Regression

Now we have done all the preparatory work to define the linear regression model.

Let us revisit the definition of residual term (4.1.8). We noted above that better models have lower prediction errors in general, but we cannot drive all of them down to zero at the same time. Instead, we somehow have to address the trade-offs we face. In case of linear regression, we define the “best” model as the one that minimizes the sum of squared residuals. This is why linear regression is often called “least squares”—the word “least” refers to minimization and “squares” refers to the fact that we minimize squared errors.⁴

³This is not an issue specific to linear regression but a common problem with all statistical models. In order to establish causality based on statistical analysis, we need very specific information that is typically not present in what we call “data”. See more in Chapter 5.

⁴The term “linear regression” and “linear least squares” are usually treated as synonyms. However, we do not necessarily have to minimize sum of squared errors. We may choose to minimize other functions of the residual terms, for instance sum of absolute values of the errors. The result is still linear, and still has the property of regression to mean (Galton, 1886), but it is usually called “median regression”, not linear regression.

We can minimize the sum of squared errors (SSE) by selecting good values for β_0 and β_1 . This gives us an informal definition of linear regression: it is the linear model (4.1.1), and parameters β_0 and β_1 , chosen in the way that SSE is minimized.

Formally, we can write the sum of squared errors as

$$\begin{aligned} SSE(\beta_0, \beta_1) &= \sum_{i=1}^N e_i^2 = \\ &= \sum_{i=1}^N (\hat{y}_i - y_i)^2 = \\ &= \sum_{i=1}^n [y_i - (\beta_0 + \beta_1 \cdot x_i)]^2. \end{aligned} \quad (4.1.12)$$

Here we write SSE as $SSE(\beta_0, \beta_1)$ to stress that it's value depends on β -s. The first line of (4.1.12) is the definition of SSE, and the others follow from the definition of residuals and predicted value. The “correct”, i.e. optimal β -s are those that minimize $SSE(\beta_0, \beta_1)$, formally written as

$$(\hat{\beta}_0, \hat{\beta}_1) = \arg \min_{(\beta_0, \beta_1)} \sum_{i=1}^N e_i^2 = \arg \min_{(\beta_0, \beta_1)} \sum_{i=1}^n [y_i - (\beta_0 + \beta_1 \cdot x_i)]^2. \quad (4.1.13)$$

Here we denote the optimal values for β_0 and β_1 by $\hat{\beta}_0$ and $\hat{\beta}_1$. This can be understood as we play around with β_0 and β_1 until we have achieved the smallest possible SSE, and then we call the corresponding values $\hat{\beta}_0$ and $\hat{\beta}_1$. Note that these values are our *estimates*, not necessarily the “true” values of β_0 and β_1 . The true values are unknown, the estimates (denoted by hat) are the closest we can get based on the data. In case of simple regression, one can get fairly far by manually computing SSE while playing with the β -s. Alternatively, one can rely on non-linear optimization (see Section 9.2). Linear regression turns out to be even simpler, as here we can solve the best β -s analytically (see Section 4.1.10). This is the only statistical model where it is possible and no doubt, this has also contributed to its popularity.

Example 4.1.4: SSE for the iris sepals regression

Let's compute a few SSE values for iris sepals, for the same data we used in Figure 4.1. Pick first $\beta_0 = 0$ and $\beta_1 = 1$, i.e. we assert that sepals are as wide as they are long. Table 4.3 shows the relevant calculations. The first 4 rows show the first four lines of data. The two first columns are the data, sepal length and sepal width. The third column, \hat{y} is the predicted width, and given our choice of β -s, it is exactly equal to sepal length. The fourth column, $e = \hat{y} - \text{Sepal width}$ is the residual. As we are predicting way too large width, the residuals are all positive. The final column, e^2 , contains the squared values of the corresponding residuals. The last row is the sum of the corresponding columns. Here we are only interested in the

last number, the sum of squared errors.

Table 4.3: Computing SSE for iris data. Sepal length and Sepal width are the actual datapoints. \hat{y} is the predicted width, given $\beta_0 = 0$ and $\beta_1 = 1$. e is the corresponding deviance and e^2 is squared deviance, “squared error”. The last line gives the sum of all rows.

	Sepal length	Sepal width	\hat{y}	e	e^2
1	5.10	3.50	5.10	1.60	2.56
2	4.90	3.00	4.90	1.90	3.61
3	4.70	3.20	4.70	1.50	2.25
4	4.60	3.10	4.60	1.50	2.25
...
sum	250.30	171.40	250.30	78.90	127.91

In this example we have $SSE = 127.91$.

4.1.3 Model evaluation: MSE, RMSE, R^2

One of the first questions after estimating your model is how good job does it do. Is this model actually better than just predicting the average value for everyone? Just how much better is the model? A natural answer to this comes from the least squares model definition: how big is its SSE?

There are several problems with using SSE as a model goodness indicator though:

- SSE grows as we add more datapoints to the model. So a large value of SSE may either mean that the model does not describe the data well, or that we estimate it on a large dataset.
- SSE is measured in squared units, so for instance if y is measured in dollars, MSE will be in dollars-squared. This is hard to interpret.
- It is also hard to compare models on different kind of data. If units of measurement are different, the SSE will also be different even on the same dataset. And sometimes the units are inherently different. For instance, based on SSE we cannot tell if income is modeled in a more precise manner than temperature.

All these three issues have fairly simple solutions.

In order to fix the first issue—SSE growing with dataset size—we can use not sum of squared errors but *mean squared error* (MSE) instead:

$$MSE = \frac{1}{N} \sum_{i=1}^N e_i^2 = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2. \quad (4.1.14)$$

You may notice that the formula for MSE resembles that of variance, just instead of using the average value \bar{y} , we use the predicted value \hat{y}_i to compute the deviations.

The solution to the second problem is easy too: instead of MSE we can use its square root, called root-mean-squared-error (RMSE):

$$RMSE = \sqrt{MSE}. \quad (4.1.15)$$

RMSE is measured in the same units as y and hence easily interpretable. If MSE resembles variance, its square root resembles standard deviation and we can say something like “typically, our predictions are off by RMSE”. Obviously, both MSE and RMSE can also be used to define linear regression in analogous fashion as SSE in (4.1.13). A set of betas that minimizes SSE will also minimise MSE and RMSE.

The solution to the last issue is a little bit more involved but it leads us to the well known R^2 , perhaps the most popular measure of goodness in regression models. We start with the observation that SSE, the sum of errors “left over” by the model, does not tell much about the model’s performance unless we know how spread-out were the observations (y -s) to begin with. So we define *total sum of squares* (TSS)

$$TSS = \sum_{i=1}^N (y_i - \bar{y})^2 \quad (4.1.16)$$

where \bar{y} is the average of y . Note the difference between TSS and SSE as defined in (4.1.12): while SSE computes the error terms as the difference between the true y_i and the model prediction \hat{y}_i , TSS computes it as the difference between y_i and the average, \bar{y} . So TSS is a convenient measure of the total spread in the data. It is very much equivalent to the variance, just multiplied by N .

This gives us a measure of model goodness: if SSE/TSS is small, the model “explains away” most of the variation in the data. For instance, if $SSE/TSS = 0.2$, the model only “leaves behind” 20% of the original variation. Traditionally one looks at this measure reverted: R^2 is defined as

$$R^2 = 1 - \frac{SSE}{TSS}. \quad (4.1.17)$$

In this hypothetical example $R^2 = 0.8$, and the model explains 80% of the variation in data. Let’s think a second what this means. In one extreme case where our model is completely useless, and our predictions are no better than just predicting the mean value for every data point, we have $SSE = TSS$ and hence $R^2 = 0$. In another extreme case where our model is able to predict every single observation exactly, we have $SSE = 0$ and hence $R^2 = 1$. So R^2 gives us a convenient and easy-to-interpret measure of prediction goodness: which percentage of the total variation is explained by the model. Small R^2 indicate the model is not good (from the predictive perspective) and high R^2 shows that it predicts well.

Example 4.1.5: R^2 for the iris sepals regression

We follow up Example 4.1.4 and compute R^2 of the corresponding regression. However, instead of picking arbitrary parameter values (0 and 1 in Example 4.1.4), we compute the regression estimates. These are $\beta_0 = -0.569$ and $\beta_1 = 0.799$ (see Example 4.1.1). First we present a similar table to compute deviations and SSE as in Example 4.1.4:

Table 4.4: Computing R^2 for iris data. The table is analogous as in Example 4.1.4, just this time using the actual regression coefficient values instead of 0 and 1.

	Sepal length	Sepal width	\hat{y}	e	e^2
1	5.10	3.50	3.50	0.00	0.00
2	4.90	3.00	3.34	0.34	0.12
3	4.70	3.20	3.18	-0.02	0.00
4	4.60	3.10	3.10	0.00	0.00
...
sum	250.30	171.40	171.40	0.00	3.16

As we have picked the regression estimates for β_0 and β_1 now, the deviations e are small, and we see both positive and negative values now. The table shows that $SSE = 3.159$, a much smaller value than 127.91, the number we got when using arbitrary values in the previous example.

In order to compute R^2 , we also need TSS (4.1.16):

$$TSE = \sum_i (\text{Sepal width}_i - \overline{\text{Sepal width}})^2 \quad (4.1.18)$$

where $\overline{\text{Sepal width}}$ is the average value of sepal width. Plugging in the data, we find $TSS = 7.041$, and hence

$$R^2 = 1 - \frac{3.159}{7.041} = 0.551 \quad (4.1.19)$$

R^2 is easily interpretable. As TSS is the total variation in the data, and SSE is the variation left unexplained by the model, then SSE/TSS is the percentage of the total variance left unexplained by the model. And hence R^2 is the opposite of it, the percentage of total variation in data that is explained by the model. In Example 4.1.5, the simple regression model explains 55.1 percent of the total variation.

R^2 is not an universal measure. It is computed from prediction errors e and hence it focuses on prediction. If our task is to predict, we should strive to as high R^2 as possible. But if our primary focus is inference, interpretation of β -s, the high R^2 value is of less importance. Different type of data lead to different R^2 values. In social sciences, it is common to observe R^2 in a range of 0.2...0.3 for ordinary regressions. If we are interested at changes over time, we often find R^2 less than 0.05, and in contrary, if we are predicting future behavior by

current behavior, R^2 may exceed 0.9.

Finally, note that when defining SSE, TSS and R^2 we did not make use of the fact that we are working with *linear* regression. In fact, all these measures are well defined for all supervised learning models with continuous outcomes. This includes nearest neighbors, trees and related methods, and neural networks, as long as the variable of interest is continuous.

4.1.4 Multiple Regression

Prerequisites: simple regression; linear algebra: vectors, matrices, vectors as matrices, matrix multiplication

What is Multiple Regression

In case of simple regression we were concerned with a case where a single explanatory variable x was associated with outcome y . Now we look at the case where there are more such explanatory variables. For instance, in order to predict income, we may want to include education, but also age, gender and place of residence (rural or urban). This is a *multiple regression* approach.

Conceptually it is similar to looking at direct effects only and eliminating indirect effects. Imagine we are interested of the effect of education on income.⁵ We estimate a model of the form

$$\text{Education}_i = \beta_0 + \text{Income}_i + \epsilon_i \quad (4.1.20)$$

But education and income may be related through different channels, one is “direct effect” where education directly influences the income (e.g. through a different job) or the way around (income determines what kind of education one can afford). This is the direct effect (Figure 4.6, left panel). But this is not the only way these two variables are related. For instance, education also influences one’s choice of where to live, e.g. in urban or rural area, and income differs by location. This is the indirect effect: education influences location, and location in turn influences income. The opposite causality is plausible as well where income determines where to live, and location determines educational choice. It is the same indirect effect. When working with simple regression, we allow the location choice to change when education changes and hence what we measure is a sum of direct and indirect effect. (Obviously, there are more factors than just location choice that influence education and income, so it may be better to talk about indirect effects in plural.) This is manifested by the fact that we do not include any information about location in the model.

But this is not the case of multiple regression where we estimate a model of a form

$$\text{Education}_i = \beta_0 + \text{Income}_i + \text{Location}_i + \epsilon_i. \quad (4.1.21)$$

⁵As “effect” we mean association, all sorts of relationships, including the causal effect. When not doing causal inference we usually talk about “effect”. When analyzing causal influence we talk about “causal effect”.

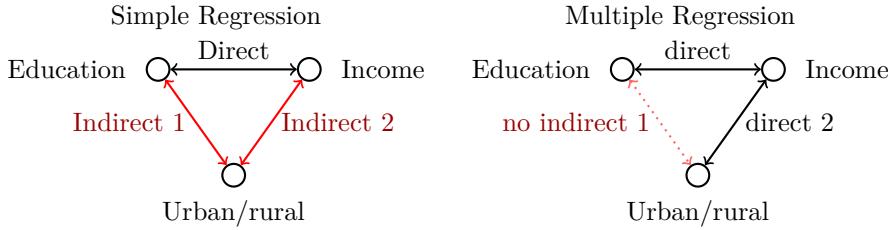


Figure 4.6: Analyzing the effect of education on income using simple regression (left) and multiple regression (right). Simple regression includes indirect effect, the red line from education over urban/rural location to income. Multiple regression fixes the urban/rural choice and in this way breaks the line between education and location. Only direct effect (black line) is left. Two-way arrows stress that the causality may run in both directions across the links.

Here we include location as an additional explanatory variable and hence it cannot change freely any more. It is forced to have the value it has in the data. As a result, the first indirect effect between Education and Urban/rural choice (Figure 4.6, right panel) is broken. Education is not allowed to influence the location choice. What is left are two direct effects—from education to income, and from location choice to income (or the way around as we cannot tell which way does causality go). More realistically, there are typically always variables we cannot control, so we just remove some of the indirect effects but still leave others in the model.

When is it be advantageous to do multiple regression? Direct effects are easier to interpret, and it is easier to base policy implications on those. For instance, the direct effect suggests that the key to good earnings is a good education while indirect effect suggests it is a location choice.⁶ In both cases the message is clear but different. In the first case it is “learn”, in the other case “move”. In multiple regression case we can actually disentangle these two and tell which one is more important. In simple regression case we cannot do that. But simple regression may be more appropriate in other circumstances. For instance, if you want to know whether better educated individuals earn more then location does not matter. What you want is just the simple regression analysis.

Technically, multiple regression is very similar to simple regression, just we allow several explanatory variables to influence the outcome y at the same time. Say we are interested in the effect of K explanatory variables. Now instead of (4.1.1) we write

$$y_i = \beta_0 + \beta_1 \cdot x_{1i} + \beta_2 \cdot x_{2i} + \cdots + \beta_K \cdot x_{Ki} + \epsilon_i. \quad (4.1.22)$$

For instance, in the income–education example above, we may have $K = 4$: x_1 is education, x_2 is age, x_3 is gender, and x_4 is place of residence. The outcome y is income. Exactly as in case of simple regression, we call y the outcome variable,

⁶Here we talk about unidirectional causal effects. It is commonly not the case.

x_k are explanatory variables and ϵ is error term. The unknown parameters β_k are sometimes called slopes but more often just “betas”. And finally, index i stresses that each observation i has a different value for y , x_1 , x_2 , ... x_K and ϵ , but they all share the same parameters $\beta_0 \dots \beta_K$.

From now on we follow vector-based formalism that make notation, mathematics, and numerical computations substantially simpler. We stack all explanatory variables x_i into a vector \mathbf{x} , a shortcut for $\mathbf{x} = (x_1, x_2, \dots, x_K)^\top$.⁷ Now the multiple regression definition 4.1.22 can be written as

$$y_i = \beta_0 + \boldsymbol{\beta} \cdot \mathbf{x}_i + \epsilon_i. \quad (4.1.23)$$

To simplify the notation further, it is common to add the constant “1” as the first (0-th) component of \mathbf{x} and define $\mathbf{x} = (1, x_1, x_2, \dots, x_K)^\top$. Note that, strictly speaking, it is not correct to refer \mathbf{x} now as “data” or “variables”, unless you are willing to refer to a constant as “data”. But this trick helps us to simplify notation even further, and we still call it somewhat sloppily “data”. So the regression model in its final vector form is written as

$$y_i = \boldsymbol{\beta}^\top \cdot \mathbf{x}_i + \epsilon_i. \quad (4.1.24)$$

Note that whatever is the number of variables K , the vector form (4.1.24) remains the same. Vectors allow us to abstract away from K , both in notation and in computations.

TBD: Predictions

Formal Definition of Multiple Regression

We can easily generalize the definition for simple regression 4.1.12 to multiple regression. We just use the multiple regression predictions to define the sum-of-squared-errors (SSE):

$$\begin{aligned} SSE(\boldsymbol{\beta}) &= \sum_{i=1}^N e_i^2 = \\ &= \sum_{i=1}^N (\hat{y}_i - y_i)^2 = \\ &= \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \cdot \boldsymbol{\beta})^2. \end{aligned} \quad (4.1.25)$$

The notation we use stresses that SSE depends on the parameter vector $\boldsymbol{\beta}$. The solution $\hat{\boldsymbol{\beta}}$ is just the parameter vector that minimizes $SSE(\boldsymbol{\beta})$:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} SSE(\boldsymbol{\beta}). \quad (4.1.26)$$

⁷Remember that $\mathbf{x} = (x_1, x_2, \dots, x_K)^\top$ is a shorthand for $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{pmatrix}$.

TBD: Interpretation in multiple regression

4.1.5 Categorical Variables

So far we have assumed that all our variables are numeric and hence the multiplication $\beta \cdot x$ is possible. But there are many types of data that are not numeric. For instance, gender is often recorded as dichotomous label “male” or “female”. Home type may be one of “rental apartment”, “condo”, “single-family home” or “other”. And some variables, although coded as numbers, are not really numbers. For instance, family status may be coded as 1–single, 2–married, 3–divorced, etc. Such variables cannot be directly included into regression models (or any other statistical models). And if done so, like with the marital status variable above, the results are probably wrong and misleading (this is the case if non-interval categories are coded as numbers).

Consider the *Males* dataset (R package *Ecdat*) that contains wages of 545 young men in 1980s. We are going to describe the wage as a function of marital status, and ethnicity. In the dataset we have marital status coded as “yes” and “no” (meaning married and non-married), and ethnicity coded as “black”, “hispanic”, and “other”. To give you better idea of the data, Table 4.5 shows a small sample of it. Note that *wage* refers to log hourly wage.

Table 4.5: Example of Males data

wage	married	ethn
1.20	no	other
1.52	yes	other
1.46	no	black
1.69	yes	black
1.12	no	hisp
2.22	yes	hisp

Note: *wage* is log hourly wage.

We would like to estimate a regression model along the lines:

$$\log w_i = \beta_0 + \beta_m \cdot m_i + \beta_e \cdot e_i + \epsilon_i \quad (4.1.27)$$

where *w* is wage, *m* is marital status and *e* is ethnicity. However, we cannot use the existing variables in a model like this as both marital status and ethnicity are not numbers but categories. Hence we have to convert the variables into numeric ones. The most popular approach to transform categorical variables into numerical ones is by creating *dummy variables (dummies)*. This amounts to getting a series of 0/1 variables, one for each category of the original variable.

Let’s start with *married*. This is a simple two-category variable with two possible values, “yes” and “no”. An obvious choice is to convert it to binary 0/1 variable where “0” refers to “no” and “1” refers to yes. Let’s call the variable *m* (Table 4.6).

Table 4.6: Extended Males data with dummy variable m denoting status “married”

wage	married	ethn	m
1.20	no	other	0
1.52	yes	other	1
1.46	no	black	0
1.69	yes	black	1
1.12	no	hisp	0
2.22	yes	hisp	1

We can use m directly in the regression model. If we do this, we get the following results:

	Estimate	Std. Error	t-value	Pr(> t)
Intercept	1.5524	0.0105	147.28	0.0000
m	0.2203	0.0159	13.85	0.0000

The basic interpretation of the result is the same as always:

- “Intercept” gives the average outcome value where all other explanatory variables are 0. In this case this means intercept corresponds to the average log-wage where $m = 0$, i.e. average log-wage for those who are not married. Non-married men earn 1.55 log units in average.
- “ m ” describes gain by those who have one unit larger m . So those who have $m = 1$ have average log-income larger by 0.22 units. Or in a more plain language, the married men earn more by 0.22 (in log terms), in total 1.77.

So we managed to include a categorical variable into our model. The interpretation tells us how much do the corresponding categories’ outcome differ, in average. It was rather easy in case of two categories.

Exercise 4.1.1: Do union members have higher wages?

The Males data also includes union membership (either “yes” or “no”). We can create analogous dummy $u = 1$ if someone is a union member and 0 otherwise. When running a simple regression

$$\log(wage_i) = \beta_0 + \beta_1 \cdot u_i + \epsilon_i \quad (4.1.28)$$

we get the following results:

	Estimate	Std. Error	t-value	Pr(> t)
Intercept	1.605	0.009	174.87	0.000
u	0.179	0.019	9.65	0.000

Use this table to answer the following questions:

- What is the log-wage for non-union members? (in average)
- What is it for union members? (in average)?
- How big is the difference in favor of the union members?

Next, let's move to multiple categories.

Dummy variables In case of regression models, both y and \mathbf{x} must be numeric. If they are not, we have to transform these in numeric form before we can begin the actual calculations.

TBD: binary, multi-category dummies

Non-linear regression Linear regression assumes a linear relationship between y and *the parameters* β , not necessarily between y and \mathbf{x} .

TBD: What it is and why OLS is called linear

TBD: Polynomial regression

4.1.6 Interactions Effects

(also *cross-effects*) is the way to design regression models that do not just to handle variables independently, but to allow different outcomes for certain joint combinations of the variables. This is one of the most widely used methods to add flexibility to regression models.

Artificial example

Let's look at an artificial example.⁸ Consider an analysis where we are interested in income as a function of cognitive skills and social skills.⁹ Assume we have collected data on personal income, performed a test for cognitive skills (such as IQ test), and assessed the social skills too. Let us measure both skills in a binary fashion: low (0) and high (1). Take a look at the four individuals (a , b , c , and d) in Table 4.7.

We focus on the first four columns for now. The baseline individual a , the one with low social and low cognitive skills, earns \$40,000 a year. The next one, individual b , has low social skills but high cognitive skills and makes \$60,000, i.e. \$20,000 more than individual a . This suggest that the effect of cognitive skills is \$20,000. No surprise, cognitive skills are valuable. However, when we compare individuals c and d , we see that adding cognitive skills for someone who

⁸See Deming (2017).

⁹Cognitive skills are skills that required for conscious mental work, such as reading, learning, math. These can be measured with standard tests, such as IQ or AFQT. Social skills are skills we use in human communication and persuasion, and include a plethora of small-scale behavioral habits that are hard to assess and train consciously.

TBD: find a few good papers.

Table 4.7: Example skill-income data.

1	2	3	4	5	6	
	Annual id	Social income, \$	Skills Social	Cognitive	Interaction Social × Cognitive	Captured by
a	40,000	0	0	0	0	β_0
b	60,000	0	1	0	0	$\beta_0 + \beta_2$
c	50,000	1	0	0	0	$\beta_0 + \beta_1$
d	100,000	1	1	1	1	$\beta_0 + \beta_1 + \beta_2 + \beta_3$

already has high level social skills improves her income by \$50,000. Cognitive skills are even more valuable for someone who has high social skills.

This effect cannot be captured by a standard multiple regression model. If we were to estimate the data by a model like

$$\text{income}_i = \beta_0 + \beta_1 \cdot \text{social skills}_i + \beta_2 \cdot \text{cognitive skills}_i + \epsilon_i, \quad (4.1.29)$$

we will interpret β_2 as the effect of cognitive skills, no matter what is the level of social skills. If we run such a linear regression on these data, we get $\beta_1 = \$25000$ and $\beta_2 = \$35000$. These values correspond to the average effect of social skill for low- and high cognitive-skilled individuals, and for the average cognitive skills effect for low- and high social-skilled individuals. But what if we want to capture the fact that higher social skills are related to a larger effect of cognitive skills?

As a solution, we can amend the model (4.1.29) by introducing an *interaction term*, $\beta_3 \cdot \text{social skills} \times \text{cognitive skills}$. From practical perspective, interaction term is equivalent in computing a new variable, social skills \times cognitive skills (see column 5 in Table 4.7), and adding it into the regression model as just another feature (modern software typically has handy shortcuts for this operation). So the corresponding linear regression model with an interaction effect will look like

$$\begin{aligned} \text{income}_i = & \beta_0 + \beta_1 \cdot \text{social skills}_i + \beta_2 \cdot \text{cognitive skills}_i + \\ & + \beta_3 \cdot \text{social skills}_i \times \text{cognitive skills}_i + \epsilon_i. \end{aligned} \quad (4.1.30)$$

When we estimate this regression model, we get $\beta_0 = \$40000$, $\beta_1 = \$10000$, $\beta_2 = \$20000$ and $\beta_3 = \$30000$.

Unfortunately, interaction effects make regression models harder to interpret. The basic interpretation remains the same: β tells how much larger is the expected outcome for those who have the variable's value larger by one unit. However, now the variable values are not independent any more. We cannot have $\text{social skills} \times \text{cognitive skills} = 1$ if the person has $\text{social skills} = 0$. So we cannot just conclude that “those with high social skills earn 10000 more than those with low social skills” as β_1 suggests. Now the effect size depends on the level of cognitive skills.

Interpreting the Interaction Effects

In order to interpret the results, let us start with predicting the outcomes for each individual. As even experienced researchers get confused by the effects, it is helpful to write down the table of dummies for each four categories (Table 4.8). It basically replicates the first four columns of Table 4.7. The four categories are a , b , c and d , the first one possessing only low skills, the last one has both high social and high cognitive skills. Importantly, we have also marked the interaction effect here. Each of the dummy column corresponds to one variable in the regression model (4.1.30) and hence to the respective β in (4.1.30).

Table 4.8: Predicting Interaction Effects.

category	Skills		Interaction		Captured by
	Social	Cognitive	Social \times Cognitive		
a	0	0	0		β_0
b	0	1	0		$\beta_0 + \beta_2$
c	1	0	0		$\beta_0 + \beta_1$
d	1	1	1		$\beta_0 + \beta_1 + \beta_2 + \beta_3$

Consider the first individual a who has low social and low cognitive skills. She has all the explanatory variables equal to 0, so her predicted income will just be $y_a = \beta_0 = \$40000$ (the last column in the table). Next, for the individual b who has low social skills but high cognitive skills, we have $y_b = \beta_0 + \beta_2 = \60000 as b has cognitive skills dummy equal to unity, and cognitive skills correspond to β_2 in (4.1.30). Individual c has low cognitive skills but high social skills and hence her income is $y_c = \beta_0 + \beta_1 = \50000 . Finally, d has both high social and high cognitive skills, and hence her $social \times cognitive$ is high too. Her income is accordingly $y_d = \beta_0 + \beta_1 + \beta_2 + \beta_3 = \100000 . The summary of modeled effects are in the last column in Table 4.7.

In order to interpret the effect, we compute the income differences. The income difference between b and a is $\beta_2 = \$20000$. The income difference between c and d is captured by sum of two coefficients, $\beta_2 + \beta_3 = \$50000$ as individual d has both cognitive skills and the interaction effect non-zero. So in conclusion we can interpret the interaction effect β_3 as the *additional effect of cognitive skills* for those individuals who have high social skills.

Sometimes it is worthwhile to present the effect in a graphical form (Figure 4.7). The figure depicts two lines: the red line describes the relationship between cognitive skills and income for low-social skill individuals, and the blue line depicts the relationship for high-social skill individuals.

Interaction effects are a popular way to add flexibility to the linear regression and other similar models. The result is not linear any more in the original features ($social \times cognitive$ is not a linear term!) but it is still a linear function in the extended feature set where $social \times cognitive$ forms a separate feature. The added flexibility comes with a cost—more complex interpretation. While the model (4.1.30) is not hard to interpret, we have lost the beauty of the original

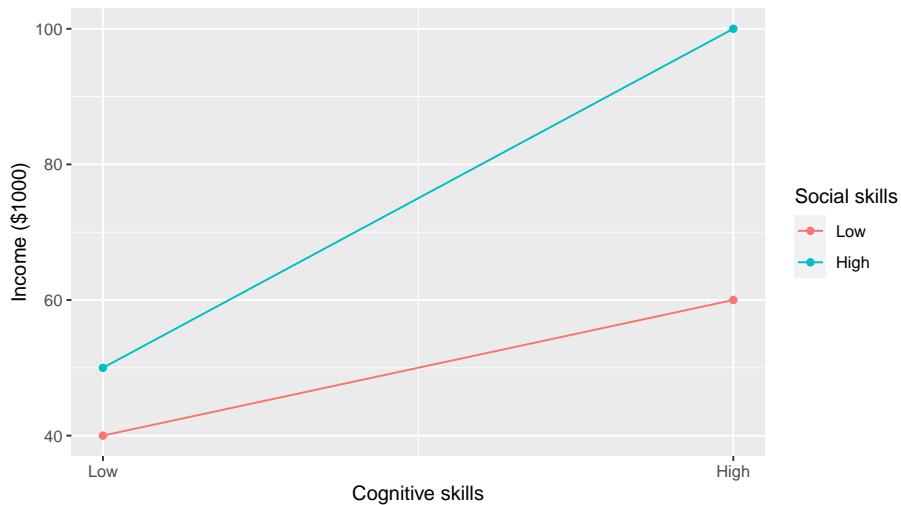


Figure 4.7: Interpretation of interaction effects

model: β_1 and β_2 are not any more the universal effects of social and cognitive skills. The effect of one factor depends on the level of another factor.¹⁰

One can easily extend the interaction effects to multi-category variables, and to continuous variables. It is also easy to introduce 3-way interactions but those are substantially more demanding to interpret. However, if we are only interested in prediction then it is not a major concern.

Example 4.1.6: Importance of social skills

Deming (2017) analyzes the effect of cognitive and social skills on wage. He uses NLSY^a data to establish cognitive skills, and workplace occupational requirements to associate jobs with social skills. Both skills variables are **standardized**, i.e. their average value is zero. He uses a linear regression model of the form

$$\log \text{wage}_i = \beta_0 + \beta_1 \cdot \text{cognitive skills}_i + \beta_2 \cdot \text{social skills}_i + \beta_3 \cdot \text{cognitive skills}_i \times \text{social skills}_i + \beta_4 \cdot \mathbf{X}_i + \epsilon_i \quad (4.1.31)$$

where besides of skill \mathbf{X} are the other individual characteristics. His results are

¹⁰In certain literature, in particular in psychology, the sentence is often phrased as “the effect of one variable is *moderated* by another one”.

variable	effect	std.error
cognitive skills	0.206***	0.007
social skills	0.049***	0.006
cognitive×social	0.019***	0.006
R^2	0.344	

where *** means the estimate is significant at 1% confidence level. These outcomes have the following interpretation:

- One unit larger cognitive skills^b are related to 0.206 units larger log income (i.e. $e^{0.206} = 1.220$ times larger wage, see [log-transformation](#)) for those with social skills equal to zero (i.e. average social skills).
- One unit larger social skills are associated with 0.049 units larger log wage (i.e. $e^{0.049} = 1.05$ times larger wage) for those with cognitive skills zero (i.e. average cognitive skills).
- There is an additional log wage premium 0.019 (i.e. $e^{0.019} = 1.019$ times larger wage) for workers with both social and cognitive skills one unit above the mean. If both skills are two units above the mean, the premium is four times as large.

This is the central results of the study: cognitive skills are more valuable for workers with high social skills. Equivalently, this can be put in the other way around: social skills are more valuable for workers with high cognitive skills.

^aNational Longitudinal Survey of Youth

^b“Unit” in case of standardized features is its standard deviation.

TBD: Why, when do we want interactions

Interaction Effects and Intersectionality

Interaction effects parallel *intersectionality*. Intersectionality is a view that many important experiences by individuals who belong to multiple groups cannot be described as only a sum of experiences by people that belong to one (and only one) of those groups. The concept of intersectionality is typically used in context of discrimination. For instance, the experience of a black women may not be described as a sum of experience of black (men) and (white) women. A workplace treats black men equally to white men, and white woman equally to white men, may still treat black women in a unfair fashion: the trait of black and the trait of women “intersect”.

We can transform this example into a regression model. Assume “treatment” here means wage the workers of the particular group receive (we can use other treatments, such as promotion, or hiring too). We can write a linear regression model for wage as

$$w_i = \beta_0 + \beta_1 \cdot race_i + \beta_2 \cdot sex_i + \epsilon_i \quad (4.1.32)$$

where w_i is wage of individual i . If we model income in this way, the “treatment”

(i.e. wage) is just sum of the treatment of the corresponding race β_2 and gender β_3 . However, if we add the interaction effect

$$w_i = \beta_0 + \beta_1 \cdot race_i + \beta_2 \cdot sex_i + \beta_3 \cdot race_i \times sex_i + \epsilon_i \quad (4.1.33)$$

then the treatment is “made of” three components: treatment of the corresponding race group, gender, and the “intesectional effect” β_3 . The members of particular race and gender may receive wage that differs from the sum of just race effect and just gender effect.

Note also that in linear regression we are always working with average values, e.g. looking for average salaries of whites and non-whites, and of men and women. Such aggregated approach has some parallels with group prejudices. It is easy to look at, say, β_2 only and claim that this number describes *all* women. This is not correct, the number describes the difference between male average and female averages for *all* man and women *in this sample*. Also, neither model (4.1.32) nor model (4.1.33) address the cause of the difference.

4.1.7 Feature Transformation

Prerequisites: Fat-tailed distributions, log-normal distribution

Standardization TBD: standardized features

Log-transformation Many types of data, such as income or price, have a well-defined lower bound but no obvious upper bound. The corresponding distributions tend not to look normal but are more similar to log-normal (Figure 4.8) and hence violate the assumptions we need to compute standard errors. An obvious remedy is to analyze log-income instead of income and in empirical literature, income analysis is almost universally done in log form. In such case, transforming your outcome variable into log form has two main advantages, one theoretical and one data-driven.

1. If distribution of log-income is more similar to normal, the issue of violating the normality assumption is likely small. This is the theoretical advantage.
2. Second advantage is data driven, and is typically correct in this type of data. Namely, log transformation improves the predictive power of the model (increases R^2), often by a substantial amount. This, in turn, is related to the fact that this type of data is often created not by additive processes but by multiplicative processes (see below).

Let’s analyze the effect of log-transform in context of simple regression. When transforming the outcome to $\log y$, we can write the OLS model as

$$\log y_i = \beta_0 + \beta_1 \cdot x_i + \epsilon_i. \quad (4.1.34)$$

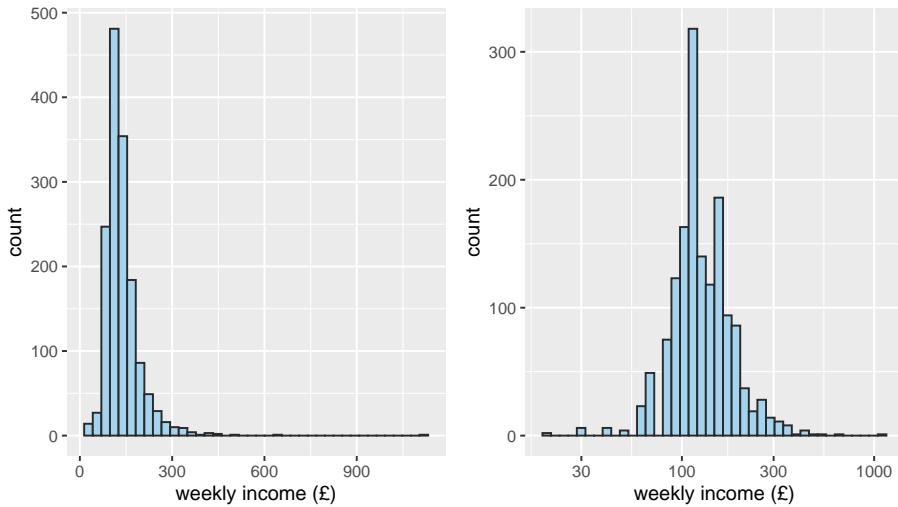


Figure 4.8: Distribution of UK household income in early 1980-s. Income distribution (left panel) does not look normal but has a long thin tail of high-income households reaching up to weekly income 1000£. Log income (right panel) is fairly close to normal as logarithm spreads low-income observations and squeezes the high-income ones. Ecdat package data.

Taking exponent of both sides we can transform it back to non-log form:

$$y_i = e^{\beta_0} \cdot e^{\beta_1 \cdot x_i} \cdot e^{\epsilon_i}. \quad (4.1.35)$$

This is not a linear model but a multiplicative model: y is not a sum but a product of three different terms:

1. e^{β_0} is the value of y in case both $x = 0$ and $\epsilon = 0$. This is the analogue of intercept.
2. e^{β_1} describes how much (how many times) will y increase as x increases (but remember: no causality!). For one unit larger x , the outcome y is e^{β_1} times larger.
3. and finally, e^{ϵ_i} is the (multiplicative!) error term.

The second advantage, in other words, is an empirical regularity: it appears that fat-tailed outcome can typically be better explained by multiplicative models, not additive models.

Finally, we also discuss the interpretation. The basic interpretation of the model is always the same but as our outcome now is $\log y$, we have to restate it that one unit larger x is associated with β_1 units larger $\log y$. When transforming the model back into non-log form (outcome is y , not $\log y$), we can restate the

interpretation as one unit larger x is associated with e^{β_1} times larger y . In case where β_1 is small, $e^{\beta_1} \approx 1 + \beta_1$ and we can say that it describes how many percent larger y we tend to observe when x is larger by one unit.¹¹ For instance, if $\beta = 0.1$, $e^\beta = 1.105 \approx 1.1$. Remember, this is a multiplicative effect and hence we can say that one unit larger x is associated with 10 percent larger y .

Log-log transformation In certain type of data, it may be advantageous to log-transform not just y but also x and hence to look at the model

$$\log y_i = \beta_0 + \beta_1 \log x_i + \epsilon_i. \quad (4.1.36)$$

In order to find the interpretation of β_1 , we can again take exponent of both sides:

$$y_i = e^{\beta_0} \cdot x_i^{\beta_1} \cdot e^{\epsilon_i}. \quad (4.1.37)$$

This models suggests it is worthwhile to look at a case where x is larger by a certain proportion, say by α percent. In that case y will be larger $(1 + \alpha)^{\beta_1}$ times. In case the percentage is small (for example, 1 percent, i.e. $\alpha = 0.01$), this is approximately equal to $(1 + \alpha)^{\beta_1} \approx 1 + \alpha\beta$. Hence we can interpret it as *how many percent y is larger when x is larger by one percent*. This figure is often called *elasticity*. Compare the interpretation of log-transformed and log-log transformed data. In the former case we find *percentage increase per unit increase in x* , in the latter *percentage increase per one percent increase in x* . Table 4.9 summarizes the interpretation of the regression coefficients. As multiple regression model can include both log-transformed and not log transformed predictors, different model estimates may have to be interpreted in different ways.

Table 4.9: Interpretation of transformed and non-transformed regression coefficients

Type	Interpretation of β_1
linear-linear ($y \sim x$)	one unit larger x is associated with β_1 unit larger y
log-linear ($\log y \sim x$)	one unit larger x is associated with β_1 percent larger y (only holds for small β_1 values)
log-log ($\log y \sim \log x$)	one percent larger x is associated with β_1 percent larger y

Example 4.1.7: Linear, log-linear, and log-log transformations

Here we use linear regression to analyze the relationship between price and mass of diamonds. Figure 4.9 shows the relationship for no transformation, log-transformation, and log-log transformation. Just visual impression suggests that the latter is the best.

¹¹Note that this interpretation does not hold if one uses decimal logarithm instead of natural logarithm.

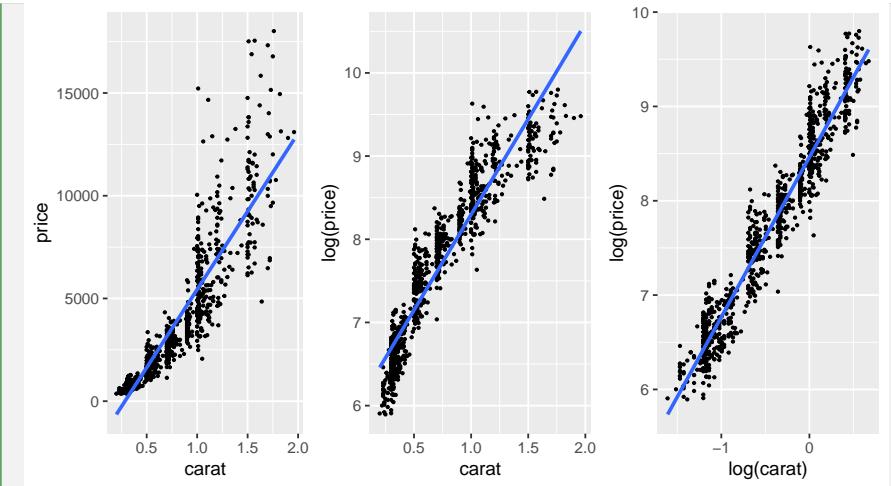


Figure 4.9: Diamond mass (carat) and price data, including the corresponding regression lines. Left panel shows the linear model in price and carat. One can see that the line does not capture the convex pattern in data. In the middle panel, the model is linear in log price and carat. Now the data pattern is concave and the line again fails to capture it well. On the right panel we log-transform both variables, and the result looks very good visually.

Next we analyze how do the corresponding linear regression models look like:

Table 4.10: Results of three different regression models: linear-linear, log-linear, and log-log.

.	object.
Intercept	-2164.441*** 98.660	6.000*** 0.023	8.462*** 0.011
carat	7601.749*** 116.904	2.297*** 0.027	
log(carat)			1.694*** 0.015
# obs	965	965	965
R ²	0.8145	0.8815	0.9311

We can see that the log-log model has the best predictive power (highest R^2) while linear-linear has the worst. The corresponding regression coefficients can be interpreted as follows: for linear-linear model, β_1 mean that one carat heavier diamonds are 7601.749 dollars more expensive. In for log-linear model, 1 carat heavier diamonds are $e^{2.297} = 9.949$ times more expensive. Finally, log-log model can be interpreted that 1 percent heavier diamonds are 1.694 percent more expensive.

TBD: other kind of feature engineering

4.1.8 Assumptions in OLS Models

We assume:

1. Above we introduced the assumption $\mathbb{E}\epsilon = 0$. This is effectively normalization, and we need this assumption only if we are interested in the value of the constant term.
2. \mathbf{x} and ϵ are independent: $\mathbf{x} \perp\!\!\!\perp \epsilon$. This is needed to get the correct estimates of $\boldsymbol{\beta}$, in particular if we are interested in causal inference.

If we have a good estimate for $\boldsymbol{\beta}$, we easily compute $\mathbb{E}y_i$ as

$$\hat{y}_i = \mathbb{E}y_i = \mathbf{x}'_i \cdot \boldsymbol{\beta} + \mathbb{E}\epsilon = \mathbf{x}'_i \cdot \boldsymbol{\beta} \quad (4.1.38)$$

where \hat{y}_i means *predicted value* of y , based on the features \mathbf{x}_i .

4.1.9 Linear Regression: The matrix approach

In both theoretical and practical applications the regression models are often presented in the matrix form. It often makes the presentation more clear but more importantly storing and handling data in matrix form tremendously simplifies solving the multiple regression model. Even more, in matrix form there is no real difference between simple and multiple regression, not in the way it is presented nor how it is solved by computer.

Let's look again at the linear regression model in vector form, (4.1.24):

$$y_i = \mathbf{x}'_i \cdot \boldsymbol{\beta} + \epsilon_i$$

or, more explicitly,

$$y_i = (1 \quad x_1^i \quad x_2^i \quad \dots \quad x_K^i) \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \end{pmatrix} + \epsilon_i \quad \text{for } i = 1 \dots N \quad (4.1.39)$$

(Remember that the first component of the \mathbf{x} is normally taken to be number 1 corresponding to the intercept β_0 .) Note we assume we have N observations indexed by i and $K + 1$ unknown parameters $\beta_0, \beta_1, \dots, \beta_K$. Alternatively we may say we have a single unknown vector parameter $\boldsymbol{\beta}$. $K + 1$ parameters correspond to K explanatory variables if we do not count the constant column as a variable (we usually do not).

Note that the first term of the matrix product in (4.1.39) is just a row vector of the data for the first observation. Hence we can take all the N rows

corresponding to each observation in the data, and arrange these underneath each other like this:

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}}_{N \times 1} = \underbrace{\begin{pmatrix} 1 & x_1^1 & x_2^1 & \dots & x_K^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_K^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^N & x_2^N & \dots & x_K^N \end{pmatrix}}_{N \times (K+1)} \cdot \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \end{pmatrix}}_{(K+1) \times 1} + \underbrace{\begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{pmatrix}}_{N \times 1}. \quad (4.1.40)$$

This is the matrix form. It can be written more briefly as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \quad (4.1.41)$$

where \mathbf{y} is typically called *outcome vector*, \mathbf{X} is *design matrix*, $\boldsymbol{\beta}$ is the same parameter vector as in case of vector-form formula (4.1.24), and $\boldsymbol{\epsilon}$ is *disturbance vector*.

Exercise 4.1.2: Matrix form

Show that (4.1.40) and (4.1.39) are equivalent.

This formula may need several comments:

- we have the design matrix \mathbf{X} not transposed in the matrix notation, unlike in the vector form where \mathbf{x}_i^\top is transposed. This is because we arrange the data for one observation *horizontally* in the design matrix and we normally denote horizontal vectors with transposition sign.
- \mathbf{X} is design matrix, the matrix that incorporates all the data already converted into the numeric form and potentially also normalized and rescaled. In a way it is data matrix, but it is (in general) not the matrix of unmodified original data.

In a similar fashion, we can stack all the prediction vectors in (??) as

$$\underbrace{\begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{pmatrix}}_{N \times 1} = \underbrace{\begin{pmatrix} 1 & x_1^1 & x_2^1 & \dots & x_K^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_K^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^N & x_2^N & \dots & x_K^N \end{pmatrix}}_{N \times (K+1)} \cdot \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \end{pmatrix}}_{(K+1) \times 1} \quad (4.1.42)$$

and in the simpler matrix form

$$\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}. \quad (4.1.43)$$

In order to use the matrix approach, we have to transform the data into matrix form—create the *design matrix*. Design matrix is our final data in a

numeric matrix form. It is “final” in the sense that it only contains the data that is actually used in the model, and it must be in a form that can be directly plugged into the formulas. In particular, this means all the non-numeric features must have been transformed to numeric ones; if feature normalization is desired this must be done and so forth. Not all models gain from construction of the design matrix, and there are models that contain more than one design matrix. But design matrix is the primary form of feeding data into linear regression and many other models.

Broadly, creating the design matrix proceeds through the following steps:

1. select only the relevant explanatory variables (i.e. no outcome variable) from your original data. Add information from different datasets if needed.
2. convert all these variables into a desired numeric form. This may include converting non-numeric variables into numeric (e.g. instead of $gender \in \{M,F\}$ we may use $female \in \{0,1\}$). It also includes converting numbers into another form if needed, e.g. continuous age into age groups or income into log income.
3. You may have to add additional columns, in particular constant—a column of number ones—as this is not normally included in your data.
4. stack the observations on top of each other as a matrix. The result looks much like a data frame but it must be a matrix in the mathematical sense of the word.

Example 4.1.8: Convert data to design matrix

Assume we have a dataset that looks like

name	age	position	salary
Liu Bei	28	manager	77,000
Sun Ren	23	employee	55,000
Thorgerd Egilsdóttir	26	employee	66,000
Freydíð Eiríksdóttir	30	senior manager	123,000

We want to describe salary as a function of position and age using a linear model

$$\text{salary}_i = \beta_0 + \beta_1 \cdot \text{age}_i + \beta_2 \cdot \text{manager}_i + \epsilon_i. \quad (4.1.44)$$

Hence we want to estimate effect of $K = 2$ variables (age and manager status) using $N = 4$ observations. We need to learn $K + 1 = 3$ unknown parameters.

Before moving any further we have to decide how to code the variables. While we can keep *age* as it is—it is already a numeric variable—we have to change *position* to a numeric form. We may follow the ordinary procedure of creating dummies and code it as *manager* = $\mathbb{1}(\text{person is manager})$ and the modified data will now be

name	age	manager	salary
Liu Bei	28	1	77,000
Sun Ren	23	0	55,000
Thorgerd Egilsdóttir	26	0	66,000
Freydíð Eiríksdóttir	30	1	123,000

In vector form the same equation would look

$$\text{salary}_i = (1 \quad \text{age}_i \quad \text{manager}_i) \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} + \epsilon_i \quad (4.1.45)$$

where $i \in \{1, 2, 3, 4\}$.

The design matrix will look like

$$\mathbf{X} = \begin{bmatrix} 1 & 28 & 1 \\ 1 & 23 & 0 \\ 1 & 26 & 0 \\ 1 & 30 & 1 \end{bmatrix} \quad (4.1.46)$$

The first column, \mathbf{x}_0 , is the constant. This is something normally needed in a linear model and here we add it to the data. The second column is *age*, left unchanged. The third column is *manager*, equal to unity if the person is manager and zero otherwise. Note we have removed *name* as we don't use this in our model, and *salary* as this is our outcome variable:

$$\mathbf{y} = \begin{bmatrix} 77,000 \\ 55,000 \\ 66,000 \\ 123,000 \end{bmatrix}. \quad (4.1.47)$$

Now we can write the model (4.1.44) in matrix form as

$$\begin{bmatrix} 77,000 \\ 55,000 \\ 66,000 \\ 123,000 \end{bmatrix} = \begin{bmatrix} 1 & 28 & 1 \\ 1 & 23 & 0 \\ 1 & 26 & 0 \\ 1 & 30 & 1 \end{bmatrix} \cdot \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \end{bmatrix} \quad (4.1.48)$$

This is the expression (4.1.40) written for this particular data. Now we may solve this for β .

4.1.10 Solving the Linear Regression Model

Prerequisites: You should be able to follow [matrix calculus](#) rules but not necessarily understand all of it. You know what is gradient, and it's notation.

We start our solution by writing the least squares condition (??) in matrix form. Using the properties of matrix multiplication, we can write the residual sum of squares, SSE (??), as

$$SSE = \sum_{i=1}^n (\hat{y}_i - y_i)^2 = (\hat{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta})^T (\hat{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta}). \quad (4.1.49)$$

Note that this result, SSE , is a scalar.

This is the number we want to make as small as possible by choosing a suitable $\boldsymbol{\beta}$ value. It is often called the *loss function*, and sometimes denoted by L or $L(\boldsymbol{\beta})$ in order to stress that it depends on the parameter vector.

Let's open the parentheses (remember: matrix multiplication is not commutative, so we have to keep track of left and right multiplication):

$$\begin{aligned} (\hat{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta})^T (\hat{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta}) &= \\ &= \mathbf{y}^T \mathbf{y} - \mathbf{y}^T \mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{y} + \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X}\boldsymbol{\beta} = \\ &= \mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X}\boldsymbol{\beta} \end{aligned} \quad (4.1.50)$$

because $\mathbf{y}^T \mathbf{X}\boldsymbol{\beta}$ is scalar and hence $\mathbf{y}^T \mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{y}$.

Next, in order to find the optimal solution, we use the well-known result from the ordinary calculus, namely that the derivative at the optimum equals to zero. Just in case of matrices, the derivative is called gradient and instead of number 0 we have zero vector $\mathbf{0}$. As we are optimizing over a vector value $\boldsymbol{\beta}$ now, we have to use the rules of matrix calculus. As SSE is a scalar, the result will just be a column vector.

The derivative can be computed as follows:

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}} SSE &= \frac{\partial}{\partial \boldsymbol{\beta}} [\mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X}\boldsymbol{\beta}] = \\ &= -2 \mathbf{X}^T \mathbf{y} + (\mathbf{X}^T \mathbf{X} + (\mathbf{X}^T \mathbf{X})^T) \boldsymbol{\beta} = \\ &= -2 \mathbf{X}^T \mathbf{y} + 2\mathbf{X}^T \mathbf{X}\boldsymbol{\beta} = 2\mathbf{X}^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y}). \end{aligned} \quad (4.1.51)$$

We use three ideas here: a) $\mathbf{y}^T \mathbf{y}$ does not depend on $\boldsymbol{\beta}$ and hence the derivative is $\mathbf{0}$. b) we use (A.2.4) for the term $\frac{\partial}{\partial \boldsymbol{\beta}} (\mathbf{y}^T \mathbf{X}\boldsymbol{\beta})$; c) $\frac{\partial}{\partial \boldsymbol{\beta}} (\boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X}\boldsymbol{\beta})$ is computed using (A.2.11).

From the optimality condition $\nabla SSE(\boldsymbol{\beta}) = \mathbf{0}$ we get

$$2\mathbf{X}^T (\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) = \mathbf{0} \quad \text{or} \quad (4.1.52)$$

$$\mathbf{X}^T \mathbf{y} = \mathbf{X}^T \mathbf{X}\boldsymbol{\beta}. \quad (4.1.53)$$

By left-multiplying both sides by $(\mathbf{X}^T \mathbf{X})^{-1}$, we get

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}. \quad (4.1.54)$$

So we have derived a simple analytic solution to the linear regression problem. Note that in order for the solution to exist,

$$(\mathbf{X}^T \mathbf{X})^{-1}$$

must exist $\Rightarrow \mathbf{X}$ must be full rank.

The solution involves three matrix multiplications (cheap operations) and one matrix inversion (expensive operation). In practice, the analytic solution is the preferred one if the dimension of $\mathbf{X}^T \mathbf{X}$ is small, i.e. we have thousands or fewer features. In case of more features, Gradient Descent may become faster as that only requires to repeatedly compute the gradient (4.1.51) but not the inverse.

4.2 Logistic Regression

The previous section introduced linear regression, one of the central workhorses for inferential analysis. The main requirement for the linear regression is the outcome variable, y , is continuous, or at least close to continuous. However, for a large class of problems, this is not the case. Logistic regression is designed for tasks where the outcome is categorical, in particular binary. For instance, the questions like who survived the shipwreck, which tweets were retweeted, and which drills got stuck are good candidates for logistic regression analysis.

4.2.1 What Is Logistic Regression And What Is It Good For?

Consider policymakers during economically challenging times. Unemployment is large and work is nowhere to be found. Government is spending a lot of money on benefits and the voices that are concerned about the effect on workers' motivation and governments coffers are growing in strength. But actually—it is not just that work is nowhere to be found. There are plenty of jobs available. But unfortunately those jobs require different skills, skills that most unemployed do not possess. So government comes up with idea to upskill the unemployed instead of just paying benefits. It announces a subsidized training program where all unemployed are welcome to participate. But who will actually end up joining this program?

Let us analyze how the participation depends on age using “Treatment” dataset (R package *Ecdat*). The dataset describes various labor market-relevant variables, such as education, income and unemployment, but from our perspective the most important information is age and whether one participated in a job training program. Figure 4.10 displays the relationship. The graph looks a bit weird, this is because there are only two values for participation—either 1 (participated) or 0 (did not participate). In order to avoid too much overlap, we have knocked to points a bit off from their true location.

We need a model that can handle

- Model probability $\text{Pr}(\text{target} = 1)$
- the probability must be in $[0,1]$
- Linear regression models the value, not the probability!
- Binary target
 - The outcome variable has two and only two distinct values
 - * attends school
 - * gets a job
 - * defaults a loan
 - * mail is spam
 - * survives

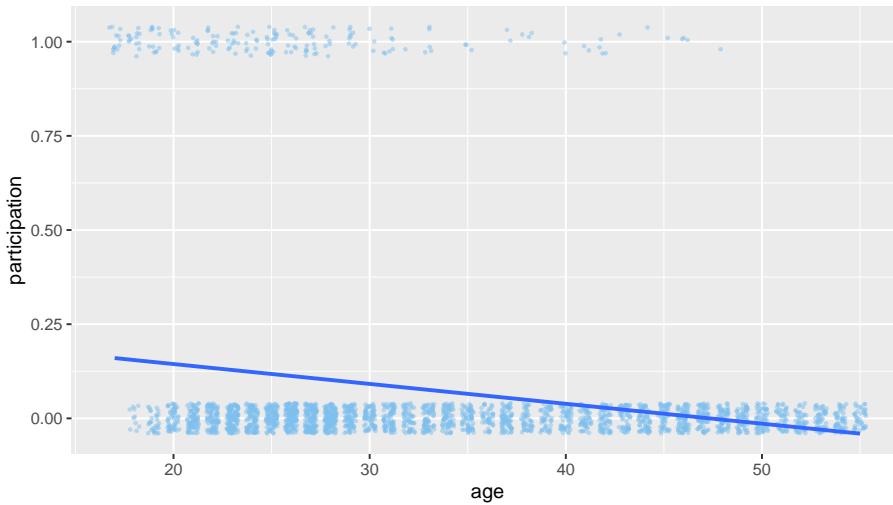


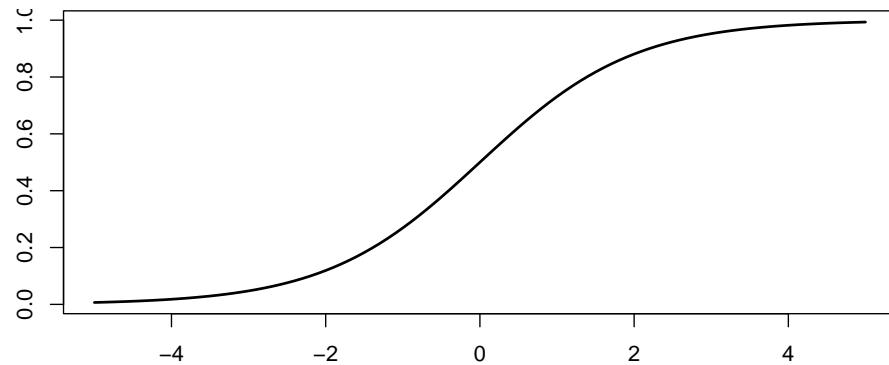
Figure 4.10: Participation as a function of age. Treatment data. In order to avoid overlap on the plot, the points are moved slightly off from their true location.

- Models $\Pr(\text{target} = 1)$
- Normally solved with maximum likelihood (ML)
- Logistic regression is for **binary target**
- Linear regression is for continuous numeric target
- Logistic regression is unrelated to log-transformations

Logistic or linear regression?

- Who survived Titanic disaster?
- How good is your GPA?
- Who gets admitted to an elite school?
- Will the tweet be retweeted?
- How many times will the tweet be retweeted?

$$\Lambda(x) = \frac{e^x}{e^x + 1} = \frac{1}{1 + e^{-x}}$$



```
summary(m <- glm(treat ~ age, data=Treatment, family=binomial()))

##
## Call:
## glm(formula = treat ~ age, family = binomial(), data = Treatment)
##
## Deviance Residuals:
##      Min        1Q     Median        3Q       Max
## -0.7325  -0.4667  -0.2903  -0.1400   3.1224
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 1.03426   0.32995  3.135  0.00172 **
## age         -0.12294   0.01223 -10.052 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1345.3 on 2674 degrees of freedom
## Residual deviance: 1187.1 on 2673 degrees of freedom
## AIC: 1191.1
##
## Number of Fisher Scoring iterations: 7
```

How much will likelihood increase per 1 year of age?

- This is not what -0.1229415 is
- It tells that the underlying linear predictor increases by -0.1229415
- which tells us nothing

Odds ratio effect:

$$e^{-0.1229415} = 0.8843154$$

is the multiplicative odds ratio effect

- one year older \Rightarrow only 88% as likely to be treated

```
##   factor      AME      SE      z      p    lower   upper
##   age -0.0075 0.0008 -9.1811 0.0000 -0.0090 -0.0059
```

- How much will probability of treatment grow (in average) per 1 year of *age*
- one year older persons are 0.75 pct points less likely to participate
- *AME*: average marginal effect: every person has a different number, this is their average

Add marital status to the model (variable "married")

- Estimate the participation probability using logistic regression
- Compute the logistic marginal effects
- Interpret the results
- Thereafter add ethnicity ("ethn") and interpret.

Remember 4.2.1: Linear regression vs logistic regression

Here we list the main differences between linear versus logistic regression:

Model Linear regression models the outcome value:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Logistic regression models the outcome probability:

$$\Pr(y_i = 1) = \Lambda(\beta_0 + \beta_1 x_i)$$

Here x_i is the predictor, y_i is outcome, and β -s are unknown parameters to be estimated; $\Lambda(x) = 1/(1 + \exp(-x))$ is the *logistic function* (sigmoid function).

Usage Linear regression can be used where the outcome y is *continuous variable* (e.g. height, income, duration).

Logistic regression can be used where outcome is *binary variable* (e.g. found a job, survived shipwreck, earthquake occurs).

Interpretation Linear regression: β_1 means *one unit larger x is associated with β_1 unit larger y (if other predictors the same)*.

Logistic regression: cannot easily interpret β_1 as this is a non-linear model. Need to compute marginal effects (or odds ratios).

Prediction Linear regression: predict outcome value

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

Logistic regression: predict outcome probability

$$\widehat{\Pr}(y_i = 1) = \Lambda(\hat{\beta}_0 + \hat{\beta}_1 x_i)$$

Predict outcome category (classification/categorization):

$$\hat{y}_i = \begin{cases} 1 & \text{if } \widehat{\Pr}(y_i = 1) > 0.5 \\ 0 & \text{if } \widehat{\Pr}(y_i = 1) < 0.5 \end{cases}$$

Chapter 5

Causality

Many important questions involve *causal inference*: we want to say something about what is a cause of an airplane crash, illness, or entrepreneurial success. This is very valuable knowledge as this allows us to avoid crashes, cure illnesses, and be successful in job. In use such knowledge extensively in our everyday lives, both instinctively (you pull your hand away from hot pan when you get burned) and deliberately (you turn on light when you cannot read any more). Such act can only successful if we know that touching hot surfaces causes burns, and flipping the switch causes the room to become light. In these cases we know and understand the process very well, or at least well enough to successfully employ to improve our everyday experience. But there are many important situations when we don't know well what will be the outcome of our acts or decisions.

It also turns out that causality is more complex than just flipping a switch. Sometimes the cause is just a binary on-off events, such as flipping a switch, or dropping a glass on floor so it breaks. Other times these are not just on-off events but can have a different quantity, *dosis*, such as the amount of training pilots receive. Sometimes we want to measure the size of the outcome, for instance when we are interested in the effect of college education on salary. In other cases the size of outcome carries little meaning (in case of breaking a glass it matters little how many pieces it breaks into) but we may be interested in the probability of the outcome, and how does this depend on the dosis of the cause. Often the cause is not a single event but multiple events linked in chains. For instance, in case of airliner accident this may include weak training of pilots, combined with management's reluctance to address technical problems of aircraft, and bad weather on landing.

In this section we are mainly concerned with quantity of a single cause embedded in such chains. For instance, *how much less likely are planes to crash if pilots have x% more training?* In this example, we are not just interested in "pilot training" but a certain amount (*dosis*) of pilot training. The question itself is often clear and well defined and can in principle be answered from data. However, as it turns out, the data with necessary structure is extremely

rare. The best answers come from randomized experiments, but these are often expensive, unethical, or just impossible. Airplane accidents are a good example of important causal questions where we cannot conduct experiments.

Because suitable experimental data is hard to find, we have to resort on other types of data. Unfortunately, this leads to both more complex econometric methods, and less reliable answers.

5.1 What is cause?

People normally have a pretty good intuitive understanding of what is cause. Two events A and B are causally related if the latter at least partly depends on the former. The dependency can be understood as if you remove the cause A , then B will not occur, or even if it occurs, it will be somewhat different. The *cause A* also has to precede *effect B* in time (although we may debate if this has to apply in non-physical world). However, the intuitive understanding is in many ways limited and does not cover several different ways how one event may influence another.

5.1.1 Sufficiency and Necessity

The intuitive concept of causality applies well if A is both necessary and sufficient for B to occur.

But if this is not the case then the concept of cause get murkier. For instance, the supersonic airliner Concorde crashed in 2000 because another airplane (Continental DC-10) dropped a large piece of metal on the runway. A few minutes later the accelerating Concorde run over the debris. This caused its tire to rupture, a piece of rubber hit the wing and broke the fuel tank in there. What was the cause of the accident? Improper maintenance of DC-10 or dangerous design of Concorde? Planes should not leave debris on runway, no doubt. But planes airliners should also be able survive tire ruptures (this was not the first tire rupture of Concorde). Both of these factors were necessary but alone they were not sufficient. Such factors are often referred to as *contributing causes*. In a similar fashion, when counting the death from a certain disease, how should one count a case where someone had more than one critical medical condition, including the disease of interest? For instance, would someone who dies of lung malfunctioning while having both pneumonia and acute COVID-19, count as COVID-19 death? At least some persons would have probably survived a single condition, either just pneumonia or just COVID-19.

Alternatively, an event may be sufficient but not necessary. If two kids are throwing rocks to a window almost instantaneously, the first rock will break the glass and the second one will just go through the already broken pane. Will the second rock still be the cause? After all, when we say that only the first rock is the cause, then when we remove the cause the outcome will still be the same—a broken window. So should we blame the second child as much as we blame the first one?

5.1.2 Measuring the Amount of Cause and Effect

In many cases the cause is essentially a binary “is there/is not there” quantity. Dropping an unboiled egg on the floor causes it to break. We can say that one unit of the cause (dropping the egg) causes one unit of outcome (smashed egg). Here the concept of quantity and quantitative effect is rather useless. We can just stay with the ordinary everyday language.

In other cases the quantity carries an important meaning. For instance, a vaccine is made of a number of different ingredients, the most important of which is called “active ingredient”. This is the substance that actually helps to fight the infection. For instance, the Pfizer coronavirus vaccine contains 30 µg of viral RNA, its most important ingredient. As this would be barely visible to human eye, the vaccines normally contain a lot of “fillers”, such as salts, sugar, and water. However, from the causality viewpoint, the important question is *how much less likely it will be to contract the disease if one takes x µg of the active ingredient?* This question involves two quantities: the quantity of the active ingredient, *dosis* of the cause, measured in µg; and quantity of the effect, here measured in terms of probability difference. In this case we expect to see a negative relationship: more micrograms of the active ingredient will make it less likely to contract the disease.

Sometimes we are only interested in the quantity of outcome but not in the dosis of the cause. E.g. we may ask *how much more (or less) likely are hurricanes because of global warming?* In this case we take the dosis of global warming as given and only ask about how it affects the probability of hurricanes. For instance, a valid answer might be “10 percent”, i.e. the hurricanes are 10% more likely now than in the past due to global warming.

Such questions get harder to understand if we add the uncertainty measures we may have in the answer. Instead of a simple “10%”, one may now give the confidence intervals: *we are 95% confident that global warming has increased the probability of hurricanes between 5 and 15%*. Note that such a claim contains two unrelated probability measures: confidence of our results (95%), and the probability of hurricanes (growth between 5 and 15%). Such double use of probability needs some probability literacy, and even for the literate it needs a second or two to understand the sentence. This is the language of science, this is very much the only type of results science can produce, but undoubtedly, complexity of claims like this has contributed to general skepticism of global warming and scientific results in general.

5.2 Causality with data: three explanations

Let us now leave (fortunately) rare air disasters and return to situations where we can collect “data”, i.e. we can observe a multitude of similar cases where we can measure various factors. For example, assume we collect data about flu shots and health in a large hospital. The hospital records contain a long list of people, and for each one we know whether they received a flu shot or not,

id	flu shot	flu
1	0	1
2	0	0
3	1	0
4	1	0

Table 5.1: Example data about flu shots. id is the individual id, flu shot is a dummy denoting whether the person get a flu shot, and flu denotes whether she got flu (1) or not (0). The table contains four observations only, but there can easily be thousands more.

and whether they got flu or not. The data may look like in Table 5.1. We are interested in the effect of *treatment* (here flu shot) on *outcome* (here getting sick with flu). This example only contains four observations but we can imagine a similar dataset of thousands of lines. Here we are interested in the flu shot as a binary on-off event, either someone got flu shot or did not. We are not interested in the dosis (the amount of active substance) or type of the flu shot. In a similar fashion, we record outcome as a binary variable: flu or no flu. We do not measure severity of flu, or between different strains of the virus. However, we are interested not just in binary flu or no-flu, but in the probability to

Whatever the size of the table, it can be summarized for our purpose in just two numbers: average flu for those with flu shot (0 according to the table) and for those without flu shot (0.5). So the individuals with flu shot are 50pct points less likely to have flu, at least in average in this data.

As this problem is framed, the data tends to make people to believe that flu shots are indeed effective. If we want to generalize from these 4 observations alone, we measure the difference in the flu rate for the no-flu shot group and the flu shot group. In this example it is 50 percentage points, so one may want to conclude that flu shots make the flu risk substantially (50 percentage points) lower.

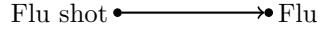
However, this conclusion is premature. This empirical regularity—flu shot is associated with 50 pct points lower probability to get flu—can be explained in three fundamentally different ways, each involving very different reasoning and very different implications.

Note the specific choice of words—*associated with*. This is a common way to say what the table tells while avoiding any misleading causal claims. It literally means that those with flu shot have lower probability to conduct flu. That is all it means. In particular, it does not mean that the lower probability is *because of* the flu shot. It does not mean that expanding the flu shot program to a larger part of population would lower the illness rate.

Choosing an appropriate vocabulary, and being able to understand and correct the common misconceptions is extremely important when working with causal inference.

Next, we discuss the three possible explanations.

Model 1: Flu shot causes (no) flu To start with, it is possible that flu shot has a direct impact on the flu, in particular on the probability to get flu. Schematically, we can write it as a *causal diagram*



The example data above suggests that if this causal interpretation is correct, flu shot is highly effective by lowering the flu probability by 50pct points. Hence the policymakers should encourage more people to get a flu shot.

This is the easy-to-understand explanation we discussed above, and it is something people intuitively tend to do, because of how the problem is framed. While not necessarily true, this is definitely a strong candidate explanation for the effect we see in data.

Model 2: Flu causes (no) flu shot Alternatively, the exact same data can be generated if it is flu instead that has an effect on flu shot. For instance, people who do not feel well may avoid flu shot because of the need to go out, while the healthy ones will get it. The causal diagram will run the opposite way:

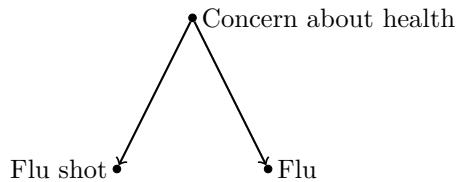


The result, in terms of data, may look exactly the same as in Table 5.1. The example situation is often referred as *self-selection*, the case where people select into treatment depending on the outcome. In our example the flu shot may be completely worthless, but because these are healthy people who get it (self-select into treatment), the previous causal model still indicates as if it is highly effective. Hence in the current example, self-selection biases the estimate upward—makes flu shot to look more effective than it actually is. It may even make a harmful treatment to look effective.

This is not just question for academic research but of immediate policy relevance. If flu shot is worthless, there is no reason to recommend it to more people.

What is the reason we may get a completely wrong result? The problem is in the causal model, not the data collection or analysis. If we use Model 1 to analyze data generated by Model 2, we may get wrong results.

Model 3: A third factor causes both flu and flu shot As a third possibility, there may be other reasons some people get flu and flu shot. For instance, those who are more concerned about their health may take flu shot, but they also wash hands, wear appropriate clothing and live otherwise more healthy life. As a result they do not get flu even if the flu shot itself is worthless. The causal diagram will look like



This is another example of self-selection where people who are less likely to get flu self-select into treatment and those who are more likely, will select into no-treatment. As a result, the estimated effect will be upward biased.

Note that unlike in the previous case, the self-selection here is not based on outcome but on *confounding factors*, other variables that explain whether someone takes flu shot. If we can incorporate confounding factors into the model, we can eliminate the problem. But when working with human behavior, we only rarely have information about all relevant background characteristics. In the example above, while data about flu shots and flu may be abundant in medical records, information on general health behavior (such as how often someone washes her hands) is much more fragmentary or completely missing.

If Model 3 turns out to be the correct causal model, there is again little reason to suggest more people to get a flu shot. The health authorities should instead recommend washing hands and following other guidelines to avoid the disease.

Unfortunately, the typical data, such as in Table 5.1, does not provide any guidance about which of the causal model is correct. They can all be correct and influence our results in different ways. The table is not enough to provide causal explanation. There are a number of strategies one may follow to establish causality. Randomized Controlled Trials are considered the best option, followed by natural experiments, case-control studies, and other methods.

Exercise 5.2.1: Does smoking cause lung cancer?

Lung cancer was historically a very rare disease. However, by 1960-s, it had become the most common cancer type in the West, and it was clearly correlated with smoking. But does smoking cause cancer?

Explain the correlation between smoking and cancer using all three causal models: smoking causes cancer, cancer causes smoking, and confounding factors cause both smoking and cancer.

Solution: see Example 5.2.1 below.

Example 5.2.1: Smoking and lung cancer

By 1960s, cigarettes were the dominant way of consuming tobacco and one could easily see that the rapid growth of tobacco smoking was accompanied with the explosive growth of lung cancer, with a 20-year lag. But a lot else had also changed by 1960, including urbanization, transportation and the chemical environment of our homes. So the correlation was not a proof of

smoking being harmful.

There were many doctors already back then who suggested that smoking causes cancer. In particular, they argued that the tars and nicotine in the tobacco smoke disturbs the growth-control mechanism in lung cells. This is the first causal model.

However, in particular the tobacco industry responded with the “smoking gene” explanation. They argued that smoking itself is harmless, but people who smoke tend to have “smoking gene”, certain set of genes that makes them likely to smoke, but also likely to get cancer. This is the third model with smoking gene being the confounding factor.

Finally, it is also possible that cancer causes smoking: in particular, cancer in very early stages, before it is diagnosed or before it even can be diagnosed, makes people to itch for a cigarette. This is the second explanation, the reverse causality.

5.3 Strategies for Causal Inference

5.3.1 We Know Which Model is Feasible

Sometimes it is possible to eliminate one or two explanations based on different type of information. Often we know that the decision to take or not to take treatment preceded the outcome in time, and hence Model 2 is infeasible. We may also know plausible physical or physiological mechanisms that can carry influence from treatment to outcome while there are no plausible confounding factors.

Unfortunately, in many important applications this is not true. In case of social processes and human behavior, one can often provide multiple plausible explanations supporting all three causal models.

Example 5.3.1: Do parachutes help to survive a “gravitational challenge”?

Smith and Pell (2003) take an absurd example and discuss the effect of parachutes on survival while jumping from aircraft. As explained above, we have three possible causal models:

1. Parachutes cause survival. This is the obvious explanation no-one (except Smith and Pell, 2003) can argue with. There is also plenty of medical evidence about how our bodies react to rapid acceleration.
2. Survival causes parachute use. Here we can eliminate this potential mechanism as we know the decision to use or not to use parachute must have preceded survival—parachutists were alive when leaving airplane. Hence causality cannot flow this way.
3. A third factor can cause both parachute use and survival. Although in principle this is possible, it is hard to come up with any plausible mechanism that might cause such an observed link (Smith and Pell (2003) suggest this is mental health).

Hence in this case we are able to eliminate both other causal mechanisms, except the first one. Note that the elimination was not based on data (a table of observations of parachute use and survival) but of more general knowledge about parachutes, decisions, and how our bodies work.

5.3.2 Randomized Controlled Trials

Randomized Controlled Trials (RCT)-s are considered the “gold standard” of causal inference. The idea of RCT is to randomly assign individuals into treatment group (those who receive flu shot) and control group (those who do not receive flu shot). Most importantly, RCT assigns treatment through a random mechanism. This means that treatment status depends on the random status, and hence it cannot depend on the outcome (as stipulated by model 2) or by confounding factors (as suggested by model 3). Hence we can immediately eliminate models 2 and 3. Only model 1 will remain as a feasible explanation.

For instance, we can imagine conducting a RCT regarding the flu shot efficiency. We need a large number of participants who are willing to give their explicit consent to join the experiment. Next, we randomize all participants into the treatment and control group. If possible, the trials are *double-blind*, i.e. neither the volunteer who receives a shot, nor the nurse who administers it know whether the syringe contains placebo or vaccine (syringes may be labeled, and the information about what each label contains is not released before the experiment is over). Those in the treatment group will receive flu shot, the others will receive placebo. Placebo may be essentially the same injection as the vaccine, just containing no active substance (the injection is mainly water, salt, and other unrelated substances). Later on collects the participants’ health information through the flu season, and when this is done, the treatment/control group information is released. Now we can analyze whether the flu shot was effective.

Unfortunately, RCT-s are not without their downsides.

- RCT-s are often unethical, expensive or infeasible. For instance, it may be considered unethical to pay different workers different wage for similar work, even if it allows us to get valuable information about how work motivation depends on income.
- Spillover effects: even if such an experiment is conducted, its results may not answer the original question. Human may react to the fact that they are in either treatment or control group, and in many cases we cannot design placebos. For instance, when analyzing the effect of content of education, it is impossible to design a “placebo education” program in a way that the participants do not understand what they are doing.
- RCT-s are often too expensive or completely infeasible. For instance, how does tax rate influence macroeconomic performance. An RCT would mean randomizing world countries into low-tax and high-tax regime, and to ensure the governments are conforming with this over a decade or more. This is clearly impossible.

- RCT-s may also be infeasible if cases of interest are rare and the delay in creating a suitable sample may be unacceptable. A similar situation may occur even while the cases are fairly common but the set-up is considered too high-risk (such as suicide attempts). In such cases we want to act immediately and not follow the data collection protocol.
- Randomization attempts to ensure that the treatment and control group are similar in all respects, except that the former receive treatment. However, just because of the random nature of randomization, this may not be true in small samples.
- It is hard to experiment with humans who may drop out of study and otherwise violate the assigned protocols.
- RCT analysis assumes the treatment is pre-determined and does not depend on outcome. However, in many applications, such as psychotherapy, the standard practice adjusts treatment depending on outcome.

Example 5.3.2: Effect of pneumonia vaccine

Bonten *et al.* (2015) analyze the efficacy of polysaccharide conjugate vaccine against pneumococcal pneumonia (effect of a specific vaccine on a particular type of pneumonia). This is a good example how a relatively straightforward RCT application—a new drug—is very complicated to conduct in practice.

The authors enrolled 84,496 elderly (65 year old or older). The eligibility criteria was no previous pneumococcal vaccinations and no immunocompromising conditions. The sample was followed with home visits next two years. The randomization was done by randomizing syringes in the shipment box with vaccine/placebo. No participant (including medical workers) was aware of the placebo status. The study included 42,240 vaccinated and 42,256 placebos. The pneumonia data was collected 2008-2013 in different medical centers. Participants who received other related vaccines, or developed other diseases, such as lung cancer, were excluded from the analysis. The participants were frequently visited in order to detect any side effects.

There were 3232 medical visits where the pneumonia was suspected. In total, the analysts detected 89 relevant pneumonia cases among the vaccinated and 178 among the placebo group, this means vaccine efficacy is roughly 50%. They also analyzed the efficacy for different pneumonia subtypes, the effect was fairly similar. The efficacy for all pneumonia cases was smaller but positive, consistent with the idea that the vaccine does not work against other types of disease. There were too pneumonia-related deaths to make any conclusions about the efficacy of the vaccine, and no evidence about adverse effects like chronic medical conditions.

In summary, the main difficulty was the small number of relevant pneumonia cases that necessitated both the enormous sample and a very long study period.

5.3.3 Natural Experiments

(Also *quasi-experiment*). Sometimes in a situation where it is not feasible to conduct a RCT, either nature or human institutions may provide a situation that is similar to a randomized experiment.

Examples:

- Resettlement of Karelians in Finland at the end of WW2. In WW2, Soviet Union captured a large swath of Finnish territory. The Finnish inhabitants, mostly peasants, were resettled in various places in Finland where there was land available. This created essentially a random experiment where the population of various villages increased in a rapid and random manner.
- WW2 German missiles on London neighborhoods. The precision of Nazi V2 missiles was good enough to hit London, but inside of the city, it hit pretty much random locations. This creates an experiment where certain blocks were randomly destroyed. How does such destruction affect urban life and development over time?
- Collapse of bridge
- Opening a new college that attract new type of students.
- Curriculum changes from one year to another
- [Correia et al. \(2020\)](#) analyze the effect of 1918 influenza pandemic on regional mortality and post-epidemic economic development. They use the fact that cities and states in the US implemented the interventions—closing businesses, banning gatherings and promoting hygiene—at different point of time. The authors argue that timing of these measures is as good as random, i.e. not related to the unobserved mortality and economic trends in any systematic way so we can consider this as an experiment.

Example 5.3.3: Do more extensive public health measures during pandemic help economy?

The 1918 flu epidemic was perhaps the largest pandemic the humans experienced in the 20th century killing according to estimates approximately 50 million people in slightly over a year.^a In the US, the public health response was largely left to the individual cities to decide, and typically included school closures, public gathering bans, and isolation and quarantine, and may also contained altered work schedules, business closures, face mask ordinances and other measures ([Markel et al., 2007](#)). What makes the response to a natural experiment is that the cities implemented these responses (non-pharmaceutical interventions, NPI-s) were implemented in different time in different cities.

In the following we describe the study by [Correia et al. \(2020\)](#). The authors analyze two relationships:

1. How does 1918 flu mortality influence the subsequent economic recovery and development?
2. How do the NPI-s implemented during the pandemic influence economy? Note that NPI-s have potentially two effects: first through their

effect on mortality, and second through direct influence on economy of certain NPI-s, such as bans on public gatherings or closing of businesses.

As natural experiment is not a RCT, we do not have well-defined control and treatment groups. Instead, we have a number of cities that implemented different NPI-s at different point of time. Importantly, for the first question, as authors argue, mortality was not related to economic shocks. This means there were no hidden confounding factors that determined both mortality and the economic development later. The only effect from mortality to economy was the direct effect: mortality influenced behavior, economic decisions, and hence economic growth. For the second question, authors employ the variation of type and timing of NPI-s. In a similar fashion, they argue that “variation across cities is unrelated to economic fundamentals”, i.e. there were no hidden confounders that determined both timing of NPI-s and economic recovery later. The only effect from NPI-s was through it’s influence on mortality, morbidity, human behavior and hence economy. These two arguments form the identifying assumptions for the models.

Formally, they estimate models of the form

$$\begin{aligned} y_{st} &= \alpha_s + \tau_t + \beta_t M_{s,1918} + \mathbf{X}_s \gamma_t + \epsilon_{st}^M \quad t \neq 1918 \\ y_{st} &= \alpha_s + \tau_t + \beta_t NPI_{s,1918} + \mathbf{X}_s \gamma_t + \epsilon_{st}^{NPI} \quad t \neq 1918 \end{aligned} \quad (5.3.1)$$

where y is an economic development indicator for city s in year t , α and τ are constants, M is mortality, NPI is NPI, and \mathbf{X} are all other city-specific covariates. Formally, the identifying assumptions are

$$M_{1918} \perp\!\!\!\perp \epsilon^M \quad \text{and} \quad NPI_{1918} \perp\!\!\!\perp \epsilon^M. \quad (5.3.2)$$

The authors find substantial effects in both models. Increased mortality has substantial negative economic effects, including fall in employment, manufacturing output, bank assets and investments in durable goods. In contrary, earlier and more forceful public health interventions do not lead to worse economic outcomes but the way around, more employment, output and assets. (Some of the results are not statistically significant though).

^aIn comparison, the First World War that ended in 1918 killed approximately 10 million in over four years.

5.3.4 Case-Control Study

In many contexts where it is not possible to conduct a RCT, one may compare different cases with different outcomes, and see if there are more “treated” cases in one group. For instance, one can compare those with and without diagnosed lung cancer and certain age group, and analyze if the group with diagnosed cancer contains a larger percentage of smokers.

Unlike RCT, case-control studies cannot unambiguously establish causality.

They are similar to other observational studies that establish correlation, but in order to eliminate other causal models we still need additional information.

Example 5.3.4: Case-Control Study of Flue Vaccine Efficacy

Ferdinands *et al.* (2014) conduct a case-control study to analyze influenza vaccine efficacy for children. They enroll 216 children (from 6 month to 17 year olds) who are admitted in intensive care units with acute respiratory problems in selected hospitals. 44 of these children are diagnosed flu and 172 are not. Those with flu form “cases” while those without flu are “controls”. As both groups are selected from children who are in a similar situation, admitted into intensive care with respiratory problems, they are broadly similar.

The authors analyze what proportion of cases and controls have been vaccinated against influenza, and find that complete flu vaccination is much less prevalent among cases (odds ratio 0.26). They conclude that vaccination is “associated with a three-quarters reduction in the risk of life-threatening influenza illness in children”.

5.3.5 Controlling for Confounding Factors

One of the most prevalent approaches is to explicitly control for all available confounding factors.

Unfortunately, all relevant information is rarely present. But examples include cases where we know what the selection is based on, e.g. when selection is done by caseworkers based on very limited information only, and the limited information is available for the researcher.

TBD: Expand, add examples

5.3.6 Explicit Modeling of Selection Process

Sometimes we can model the selection process based on theoretical considerations.

TBD: Heckman’s method.

5.4 Causal inference in linear regression framework

Prerequisites: [Simple regression 4.1.2](#)

This section discusses some of the causality aspects more formally in a linear regression framework. Linear regression is just a simple and popular framework but the central ideas here carry over to all other statistical models. In particular the fundamental problem, “curse of counterfactual”, is always there, no matter which model we are using.

Many important causal questions, for instance

- does the drug cure illness?
- how would going to college influence my income?
- does advertisement work?

can be written as linear regression problems in the form

$$y_i = \beta_0 + \beta_1 T_i + \epsilon_i \quad (5.4.1)$$

where y is the outcome (illness, income, or whether someone buys a product), T is *treatment*, the indicator if the person attended college, took the drug, or was shown an advertisement. β_0 and β_1 are (in general unknown) parameters. Here we discuss the case where T can be measured as a binary 0/1 indicator variable (for instance, whether the person took the drug or received placebo instead). The central parameter of interest in (5.4.1) is β_1 . This tells us how much larger (or smaller) y would be if $T = 1$ instead of $T = 0$. This is exactly the causal effect we are interested in, the effect we can use for policy design. The disturbance term ϵ captures the individual-specific effects, e.g. responsiveness to drugs and illnesses, or learning ability. These individual specific effects do not depend on T .

5.4.1 Counterfactual and Identifying Assumption

How can we compute β_1 ? Denote the outcome y_i as a function of the treatment status T by $y_i(T)$ (we use the linear model (5.4.1)). From that expression we immediately see that $\beta_1 = y_i(1) - y_i(0)$, i.e. the causal effect is just the difference between the outcome when treated, $y_i(1)$, and when non-treated, $y_i(0)$, and this is true for every single individual i . This feels almost like a trivial thing to do. And it were indeed trivial if only we could measure both $y_i(1)$ and $y_i(0)$. But unfortunately we can never, never ever, observe $y_i(1)$ and $y_i(0)$ at the same time. Someone (or something) is either treated or not treated. Nothing can be in two treatment states at the same time.¹ In our observed world the treatment status is either $T_i = 0$ and only $T_i = 0$, or $T_i = 1$ and only $T_i = 1$. The corresponding $y(T)$ is called *actual outcome*, and the other, the one we don't observe, is *counterfactual outcome* (or just *counterfactual*). The actual outcome is easy to handle: this is what you observe. Just measure it. All the trouble with causal inference is to come up with a suitable proxy for the counterfactual. We stress here once again that the only way to "measure" counterfactual is by using a proxy. It is fundamentally impossible to measure the counterfactual value. This is the "curse of counterfactual".

Example 5.4.1: Former outcome as counterfactual

It may be tempting to use the pre-treatment observations as proxies for post-treatment counterfactuals. This may or may not be correct but in any case it requires additional justification. For instance, returning to the question of effect of college degree on income, we might consider taking

¹We stay in the macroscopic world and do not discuss quantum superposition, Schrödinger's cat and related topics here.

pre-college income as counterfactual for post-college income. This is obviously absurd. Before starting college most students were fresh high school graduates with little or no work experience and often with no job. Had they not attended college, they would be working in most case, and have 4 years of experience by now. A 18-year old person without work experience will not form a valid proxy for a 23-year old with 4 years of experience.

See more at [Before-After Estimator](#) in Section 5.4.2.

As counterfactual is not observed, it is hard to say anything about it based on data, or at least based on data alone. Coming up with a convincing counterfactual always includes certain assumptions, usually referred to as *counterfactual assumptions* or *identifying assumptions*. For instance, such an assumption may state that in average, the treatment group outcome would be the same as that of the control group, if the members of the treatment group had not received treatment. In practice, these assumptions must always be backed up with knowledge about the selection process. The just cited assumption seems credible if we performed a randomized controlled trial (RCT). After all, randomization is done for this exact purpose: to make the treatment and control group look exactly the same in all known and unknown dimensions. However, it will not be credible at all if we are comparing salaries of college graduates and non-graduates. College graduates and non-graduates differ in many respects, not just by the fact that one group has spent several years of their life as students. Note that the knowledge about selection process is not usually called data. Just obtaining more “data”, i.e. observations about the treated and non-treated outcomes, does not allow us to make credible conclusions about the causal effect. We need *both* data *and* knowledge about the selection process.

What happens if the identifying assumptions are wrong? Take (5.4.1) as the point of departure. For simplicity, that model only contains a single explanatory variable, the treatment status T . The data we collect consists of tuples in the form (T, y) , i.e. every case is a pair of two numbers, the treatment status and the corresponding outcome. How can we estimate β_1 ? Intuitively, in a large sample, the average outcome for the treated, $\bar{y}(1)$ and for the untreated, $\bar{y}(0)$, should tell us something about the effect. In particular, we are tempted to interpret their difference $\bar{y}(1) - \bar{y}(0)$ as the treatment effect. (We look at a large sample in order to avoid issues with sampling noise.)

The intuitive concept of “average over a large sample” corresponds to mathematical concept of expected value, so we can replace the large sample average with the corresponding expected value. We denote the corresponding conditional averages by $\mathbb{E}[y|T = 1]$ for the treated and $\mathbb{E}[y|T = 0]$ for the non-treated. Using the model (5.4.1) we can express these values as

$$\mathbb{E}[y|T = 1] = \mathbb{E}[\beta_0 + \beta_1 T + \epsilon|T = 1] = \beta_0 + \beta_1 + \mathbb{E}[\epsilon|T = 1] \quad (5.4.2)$$

and

$$\mathbb{E}[y|T = 0] = \mathbb{E}[\beta_0 + \beta_1 T + \epsilon|T = 0] = \beta_0 + \mathbb{E}[\epsilon|T = 0]. \quad (5.4.3)$$

Expected value of constants β_0 and β_0 are just these two constants, β_1 drops out from the second equation because in that case it is multiplied by $T = 0$.

But the last terms, the conditional expectations of ϵ , are critical. $\mathbb{E}[\epsilon|T = 1]$ is the expected value of the error term for the treated individuals, $\mathbb{E}[\epsilon|T = 0]$ is the same for the non-treated individuals. In general, these two differ, i.e. $\mathbb{E}[\epsilon|T = 1] \neq \mathbb{E}[\epsilon|T = 0]$.

When we now compute the difference between the two expected values (5.4.2) and (5.4.3), we get

$$\mathbb{E}[y|T = 1] - \mathbb{E}[y|T = 0] = \beta_1 + \mathbb{E}[\epsilon|T = 1] - \mathbb{E}[\epsilon|T = 0]. \quad (5.4.4)$$

Obviously, this equals to the correct value β_1 only in case of

$$\mathbb{E}[\epsilon|T = 1] = \mathbb{E}[\epsilon|T = 0]. \quad (5.4.5)$$

This condition is known as *mean independence assumption*, denoted by $\mathbb{E}\epsilon \perp\!\!\!\perp T$.² This is the technical way to state the identifying assumption for cross-sectional estimator.

Example 5.4.2: Expected value of unobserved characteristics

For instance, when returning to our previous example about college degree and income, the unobserved factors ϵ that influence wage may include socio-economic background, cognitive and non-cognitive skills, health, geographic location (such as country and rural/urban location) and so on. So $\mathbb{E}[\epsilon|T = 1]$ is the expected value of such factors for college graduates and $\mathbb{E}[\epsilon|T = 0]$ is the expected value of the same factors for those who did not attend college. We know that college graduates tend to have higher socio-economic status, they are more likely urban and living in high-income countries, and they possess more cognitive skills. And we have no reasons to believe that the other factors are similar between these two groups, hence most likely $\mathbb{E}[\epsilon|T = 1] \neq \mathbb{E}[\epsilon|T = 0]$

Identifying assumptions always have two sides: the technical requirements—which technical properties in the data are required to make the estimator valid; and an intuitive side—what do these technical requirements mean. So (5.4.5) states the technical side of the identifying assumption and Example 5.4.2 describes its intuitive content. As the example suggests, the assumption is probably not fulfilled when comparing college graduates and not graduates.

5.4.2 A Few Popular Estimators

Prerequisites: Simple regression 4.1.2, interactions in linear regression 4.1.6, independent random variables 1.3.3, conditional expectations 1.3.5.

There are many ways to estimate causal effect β_1 . Here we introduce a few simple and popular methods: cross-sectional estimator, before-after estimator,

² $\mathbb{E}\epsilon \perp\!\!\!\perp T$ typically denotes *independence*, a stronger condition than mean independence. See [independent random variables](#) in Section 1.3.3

and differences-in-differences estimator. While the two former are simple and popular in media, the latter one is based on slightly more credible assumptions, and is often used in research. These are all based on *fixed effects* approach and assume that certain values or trends are invariant.

Cross-Sectional Estimator

TBD: just difference in means versus OLS

The idea with cross-sectional estimator is very simple: we assume that the difference between treated and non-treated outcomes is due to the treatment, and only due to the treatment, at least in average. If this is the case then the untreated cases (controls) form a valid counterfactual, and we get correct estimates by just computing the average difference between the treated and the controls. Formally, we assume $\mathbb{E}[\epsilon|T = 1] = \mathbb{E}[\epsilon|T = 0]$ and hence the effect of interest is

$$\beta_1 = \mathbb{E}[y|T = 1] - \mathbb{E}[y|T = 0] \quad (5.4.6)$$

(see (5.4.4)). As discussed above, this assumption seems a credible one in case of RCT-s, and a lot less credible in case where different type of people can freely decide whether to get treatment. For instance, it is extremely hard to justify that college graduates and non-graduates are similar in every way except graduation. We have many good reasons to think that ϵ and T are systematically related. Smarter students with better socio-economic background are overrepresented among college graduates, and both background and innate skills help to get well-paid jobs later in life too.

Example 5.4.3: Cross-sectional estimator of college effect is biased

Let us continue Example 5.4.2. The arguments there suggest that college graduates are drawn from more favorable ends of distribution of ϵ and hence T is positively correlated with ϵ . This means $\mathbb{E}[\epsilon|T = 1] > \mathbb{E}[\epsilon|T = 0]$. As a result the estimator (5.4.4) is upward biased:

$$\mathbb{E}[y|T = 1] - \mathbb{E}[y|T = 0] = \beta_1 + \mathbb{E}[\epsilon|T = 1] - \mathbb{E}[\epsilon|T = 0] > \beta_1 \quad (5.4.7)$$

The bad news is that we don't know by how much biased is the estimate, and based on data alone we cannot tell. Here "data" means a table of college graduation status and income for a large number of individuals. Education is one of the many unfortunate examples where it is hard to find plausible information to break the curse of counterfactual and actually compute the effect.

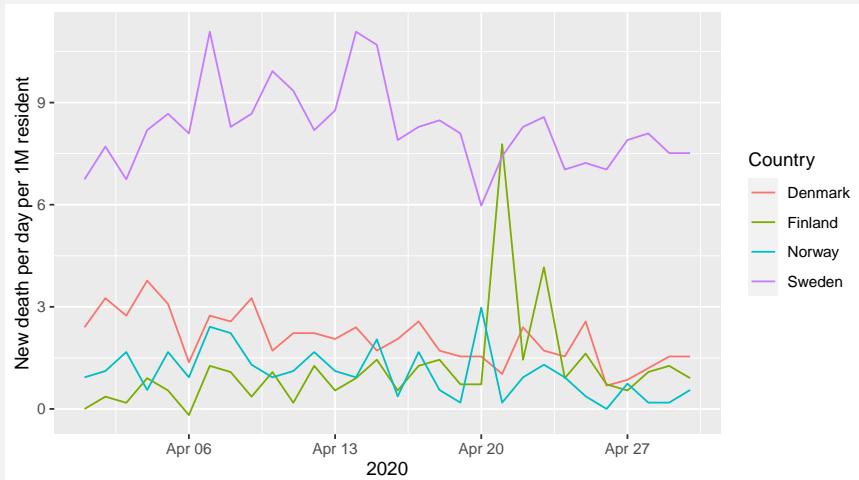
The intuitive side of the assumption was already stated above but we repeat it here: for the cross-sectional estimator to be valid, the average unobserved characteristics of the treated and controls must be similar. If this is the case, then controls make valid counterfactuals for the treated as they are otherwise similar, except that they received the treatment.

Example 5.4.4: COVID-19 stay-at-home orders in Nordic countries

Nordic countries^a are small but highly developed countries in Northern Europe. They are rather similar in terms of their institutions, featuring extensive social safety net, high taxes, effective governance and little corruption. The population of Denmark, Finland and Norway is approximately 5 million, that of Sweden is 10 million. During the COVID-10 pandemic in 2020, most European countries issued stay-at-home orders, banned public gatherings and closed non-essential businesses. However, almost nothing was done in Sweden where only gatherings of more than 500 were forbidden, and face masks remained rare.

This suggest to use a cross-sectional estimator where we compare Sweden to other Nordic countries. As Swedish policy was clearly exceptional, we can define it as treatment, so in this case treatment means “not introducing stay-at-home orders”. Non-treatment would then be what other countries did, to issue such orders.

We focus here on the first wave of COVID in April 2020. The data we use originates from <https://raw.githubusercontent.com/datasets/covid-19/master/data/time-series-19-covid-combined.csv>. The figure below shows the daily number of new deaths in all these three countries.



The figure clearly indicates that the death rate per million residents was much higher in Sweden fluctuating between 6 and 9 over this period. In the comparison countries, it mainly stayed below 3, with the exception of a short peak in Finland.

The CS estimate of the effect is just the difference between the corresponding average values. The average death rate in Sweden is 8.252, in other Nordic countries it is 1.432 and hence their difference, the effect, is 6.82 (deaths per day per million residents).

^aHere we consider Finland, Denmark, Norway and Sweden.

In practice, it is often useful to use linear regression in the form of (5.4.1) instead. Linear regression approach has two main advantages:

- We can easily include other covariates, e.g. demographic variables such as age distribution in case of COVID death rate. Adding more covariates when just comparing means can be done in a very limited fashion only. One has to split data not just along the treatment/control group but also along other covariates, and we will rapidly run into curse of dimensionality.
- Linear regression supplies the confidence intervals and statistical significance figures with no additional work. True, we can get the same result when performing *t*-test on the treatment and control samples directly, so this is just a minor convenience.

**Example 5.4.5: COVID-19 stay-at-home orders in Nordic countries:
regression approach**

Here we replicate the results of Example 5.4.4 with linear regression. As we define the treatment to be “no lockdowns”, it is equivalent to being “Sweden”, so we can write the model (based on (5.4.1)) as

$$deaths_i = \beta_0 + \beta_1 Sweden_i + \epsilon_i. \quad (5.4.8)$$

The dummy $Sweden_i$ must be understood as for every observation we use a 0/1 dummy that tells if this is an observation about Sweden. The results are in the table below:

	Estimate	Std. Error	t-value	Pr(> t)
Intercept	1.432	0.121	11.845	0.000
Sweden	6.820	0.242	28.210	0.000

The results must be interpreted as follows: *Intercept* is the average daily death rate in case $Sweden = 0$, i.e. the death rate outside Sweden (compare with the results in Example 5.4.4). *Sweden* is the effect of treatment, i.e. not introducing lockdowns. As the table indicates, the effect is very large compared to the intercept (6.82 versus 1.432) and highly significant. So the model suggests that the “no-lockdown” treatment resulted 5.8-fold increase in death rate. This seems like a very large effect.

Can we conclude that lockdowns elsewhere were a very good idea? Not so fast. First, is the identifying assumption credible? In this case it is $\mathbb{E}[\epsilon | Sweden = 1] = \mathbb{E}[\epsilon | Sweden = 0]$, i.e. the omitted variables for Sweden are similar to those in the other Nordic countries (in average). While there are good reasons to believe that Sweden is somewhat different from its Nordic neighbors, it is hard to believe the difference in death rate should be that big. After all, the standard deviation of the error term is just 1.14. This is much smaller than the effect 6.82. So even if the mean independence

assumption may not be completely correct, it seems that any bias here is dwarfed by the effect size.

But before we offer any policy conclusions, note two caveats. First, we are talking about a large difference in a small rate (a few cases per million). Maybe it does not matter that much. And second, we do not know what did Sweden benefit from this policy. Did its economy perform better? Did the population maintained better mental health? We cannot give solid policy advice before having an answer to those questions.

Before-after estimator

Before-after estimator (BA) is similar to the cross-sectional one, but instead of comparing two different groups, the treated and the non-treated, we compare the same cases before the treatment and after the treatment. If the disturbance term does not change over time (in average) then the difference is the causal effect.

While the main approach remains very similar to CS estimator, it is useful to introduce a slightly different notation. Assume there are two time periods: $t = 0$ is time before treatment and $t = 1$ is time after treatment. The corresponding treatment indicator for individual i is T_{it} with $T_{i0} = 0$ before and $T_{i1} = 1$ after treatment. So we may write

$$y_{it} = \beta_0 + \beta_1 T_{it} + \epsilon_{it} \quad (5.4.9)$$

Here y_{it} is the outcome of individual i at time t , and two indices for ϵ_{it} indicates that the error term may also differ for the same individual over the successive time periods. We only consider observations that are treated between time 0 and 1, so $T_{i0} = 0$ and $T_{i1} = 1$ for all individuals i . The expected outcome after the treatment, $\mathbb{E}[y|t = 1]$, is now

$$\mathbb{E}[y|t = 1] = \mathbb{E}[y|T = 1] = \beta_0 + \beta_1 + \mathbb{E}[\epsilon|t = 1] \quad (5.4.10)$$

where we use the fact that “after”, at $t = 1$, everyone is treated, i.e. $T = 1$. In a similar fashion, the expected outcome before the treatment is

$$\mathbb{E}[y|t = 0] = \mathbb{E}[y|T = 0] = \beta_0 + \mathbb{E}[\epsilon|t = 0] \quad (5.4.11)$$

and the estimated effect

$$\mathbb{E}[y|T = 1] - \mathbb{E}[y|T = 0] = \beta_1 + \mathbb{E}[\epsilon|t = 1] - \mathbb{E}[\epsilon|t = 0]. \quad (5.4.12)$$

This captures the correct value, the causal effect β_1 , only if $\mathbb{E}[\epsilon|t = 1] - \mathbb{E}[\epsilon|t = 0] = 0$. To put it in words, this means that the expected disturbance term before and after treatment is similar. Or more plainly—there is no unobserved trend. This is the identifying assumption for the before-after estimator. The requirement is pretty obvious—the BA estimator is just the outcome difference over time, and this is the causal effect only if there is no other time

differences interfering with the effect. For instance, if we are using before-after estimator to assess the effect of college degree, we have to assume that if a person had not attended college, her income would have stayed the same. (Note: we only look at those who attended college in this estimator.) This is a completely unrealistic assumption, similar to the claim that high-school students and young workers without college degree in their mid-20s would earn exactly the same. However, in other cases before-after estimator may be justified.

Example 5.4.6: President's approval: before and after September 11th

September 11th terror attacks were a major shock for the U.S. society, and the effect was immediately reflected in the president's approval ratings. The figure below depicts the Presidents (G. W. Bush) approval rating between From July till November, 2001. The dashed vertical line is the September 11th terror attacks. Tremendous support to president is immediately obvious from the huge increase in the approval rate.

TBD: Explain data, data source, also in the repo

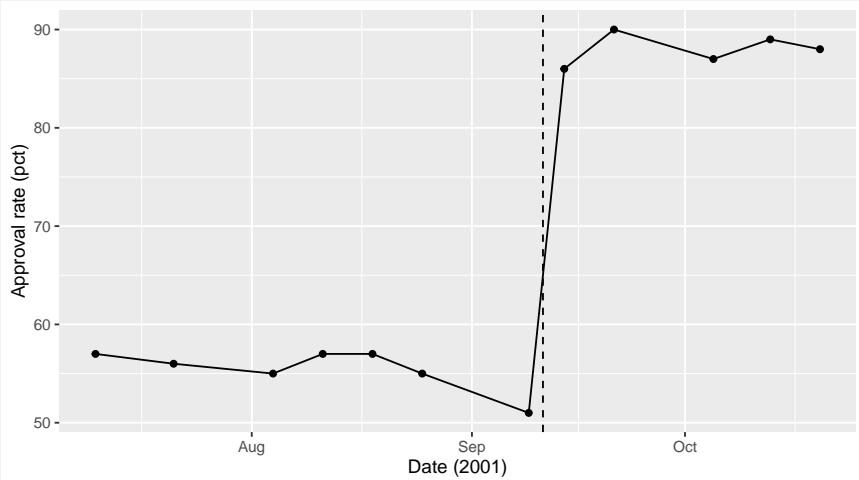


Figure 5.1: U.S. President G. W. Bush approval rating through summer and fall 2001. The dashed vertical line corresponds to September 11 terrorist attacks.

The president's approval rate was hovering around 55% in August and early September, and between 85 and 90% in late September and October. The Sept 7-10 polls indicated that 51% of the respondents approved his performance but just a week later, September 14-15, it was at 86%. The sharp jump corresponds exactly to the terror attacks, and is missing anywhere else in this data.

G. W. Bush's average approval rate from early July till early September was 55.4% and from mid-September till end of October it was 88%. The difference, 32.6pct points, is the BA estimate of the effect.

Instead of period means, we can also just look at the last pre-attack poll (support 51%) and the first post-attack poll (support 86%). Using this approach the effect is 35 pct points.

Note that the identifying assumption—no trends besides the effect of treatment—seems to be violated here as the approval rate seems to be heading downhill through the summer. But as the 9/11 effect clearly dwarfs the trend, we consider it to be a minor issue.

In a similar fashion like what we did with cross-sectional estimator, we can also use linear regression to compute the BA estimate. This can be done by using (5.4.13), typically one also has to create the auxiliary treatment indicator T based on time of the observations.

Example 5.4.7: Presidents approval: before and after September 11th, the regression approach

Let us revisit the George W. Bush's approval ratings in 2001. First we compute the treatment indicator T . Here it is related to time, $T = \mathbb{1}(date > 2001-09-11)$, i.e. it equals to one for all observations after September 11th, and to zero for all observations before that date. However, in order to stress the fact that the treatment is related to observation after the event (and in order to distinguish between treatment group and post-even observations for differences-in-differences estimator), we label it *After* instead. Thereafter we use (5.4.13) as the regression model. The complete data including *After*, 12 observations, is in the table below:

date	Approval, pct	After
2001-07-10	57	0
2001-07-21	56	0
2001-08-04	55	0
2001-08-11	57	0
2001-08-18	57	0
2001-08-25	55	0
2001-09-09	51	0
2001-09-14	86	1
2001-09-21	90	1
2001-10-05	87	1
2001-10-13	89	1
2001-10-20	88	1

Table 5.2: G.W.Bush approval ratings through the first fall of his presidency. The last column, After, is the post September-11 indicator.

Now we adapt model (5.4.13). We use *After* in place of T and drop the index i as the data is just about a single person, so we have

$$y_t = \beta_0 + \beta_1 \text{After}_t + \epsilon_t. \quad (5.4.13)$$

When we estimate this model, we get the following results:

.	object	..
Intercept	55.429	<i>0.734***</i>
After	32.571	<i>1.137***</i>
# obs		12
R ²		0.9880

Table 5.3: DiD regression estimate for the effect of 9/11 terror attacks on presidents approval rating. Standard errors in italics.

The model shows that before the attacks, the approval rate was $\beta_0 = 55.429\%$, and after the attacks it was larger by $\beta_1 = 32.571$ pct points. This is the BA estimate, it is easy to see that it has extremely large t value. The approval level after the attacks is predicted to be $\beta_0 + \beta_1 = 88\%$. These are exactly the same numbers we have in Example 5.4.6. Here linear regression recovers the mean differences.

Note that credibility of the estimator, in this case almost the same thing as the credibility of the identifying assumption, relies on our knowledge of the events through the last decades of 20th and the first decades of 21st century. We know that no other president has seen such a boost in the approval rate, and there has been no unexpected events comparable to September 11th attacks.

Differences-in-differences estimator

Differences-in-differences (also diff-in-diff or DiD) estimator combines the cross-sectional and before-after estimators. The former is biased if the treatment and control groups differ in a way we cannot take into account (i.e. $\mathbb{E}[\epsilon|T = 1] \neq \mathbb{E}[\epsilon|T = 0]$), and the latter if there is an uncontrolled trend in the treated group (i.e. $\mathbb{E}[\epsilon|t = 1] \neq \mathbb{E}[\epsilon|t = 0]$). DiD compares the time trend for the treated group and non-treated group, or what amounts to the same, it compares treated-nontreated differences before and after the treatment. This relaxes the assumptions behind the cross-sectional and before-after estimators and replaces these with a different identifying assumption: time trends for the treated and non-treated groups are the same. However, we pay for the more relaxed assumptions with more stringent data requirement: now we need four data points, both for the treated and for the non-treated before and after the point of time of treatment.

Let us first take a hypothetical example. Imagine there is a federal country that contains a number of provinces. In year 2015 certain provinces decided to substantially boost the public education by investing in schools, teachers and outreach. We consider these additional investments to be treatment T , so some provinces were in the treatment group $T = 1$ while others that did not invest are in the control group $T = 0$. According to survey data from 2014, before the treatment began, the average schooling level in the treatment provinces $\bar{y}(T = 1, t = 2014) = 9$ years and in control provinces $\bar{y}(T = 0, t = 2014) = 8$

years. Another survey from 2020, five years into the treatment, found that $\bar{y}(T = 1, t = 2020) = 11$ and $\bar{y}(T = 0, t = 2020) = 9$. (See Figure 5.2.)

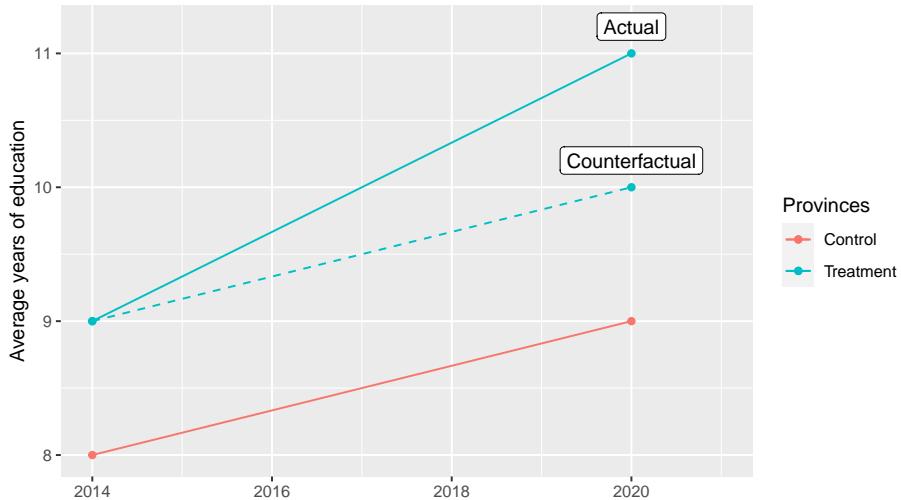


Figure 5.2: Hypothetical education data. Both the levels and trends differ for the treatment and control provinces. Dashed line denotes the counterfactual assumption, the difference between the actual and counterfactual value is the DiD estimate, here 1 year of extra schooling.

We can immediately see that the data does not support the CS and BA identifying assumptions. As the treatment and control groups differ already in 2014, before the treatment even began, it is hard to argue that they would not differ in 2020 if some of the provinces were not investing in education. In a similar fashion, in case of BA estimator, the assumption that without such an investment, the education level of 2020 would be the same as in 2014 for the treatment provinces is not convincing. After all, in control provinces the level is increasing with no treatment whatsoever! What DiD method assumes is that the *trend difference* is due to treatment. So without the treatment, the treatment provinces would have followed the dashed trajectory on the figure, leading up to the counterfactual of 10 years by 2020. However, as the actual outcome was 11 years, the difference, 1 year of extra schooling, is the DiD effect.

Let us now look at this idea more formally. We have two groups, control $T = 0$ and treatment $T = 1$ and two time periods: before $t = 0$ and after $t = 1$. Imagine we have measured these four data points as a (control before), b (treatment before), c (control after) and d (treatment after) (See Table 5.4). So a and b are measured before anyone was treated, and if we want to use cross-sectional assumption, we would expect those to be equal.

The values c and d describe the control and treatment group outcomes after treatment, at $t = 1$. Their their difference, $d - c$, is both due to the treatment

Table 5.4: Four datapoints

time	control ($T = 0$)	treatment ($T = 1$)
before ($t = 0$)	a	b
after ($t = 1$)	c	d
trend	$c - a$	$d - b$
difference in trend	$(d - b) - (c - a)$	

effect and other, unobserved, difference. But we assume that the unobserved difference after the treatment equals to that before the treatment, $b - a$. This is the identifying assumption. Hence the difference what is left over when we subtract the pre-treatment difference $b - a$ is the treatment effect:

$$\beta = \text{pre-treatment difference} - \text{post-treatment difference} = (d - c) - (b - a). \quad (5.4.14)$$

Alternatively, we can look at the difference over time. The values in the column *control*, a and c , describe the control group outcomes before and after the treatment. In case of before-after estimator they should be equal³ but now we allow a time trend $c - a$. The next column, *Treatment*, shows the outcomes for the treatment group where the time trend is $d - b$. This time trend is caused by both treatment and other, unobserved, trend. But we assume that the unobserved trends for the control and treatment group are the same, $c - a$. Hence the difference what is left over when we subtract the control group time trend from the treatment group time trend is the treatment effect:

$$\begin{aligned} \beta &= \text{treatment group trend} - \text{control group trend} = \\ &= (d - b) - (c - a) = (d - c) - (b - a). \end{aligned} \quad (5.4.15)$$

As both of these approaches gave us the same estimate, we can conclude that both assumptions are equivalent. So the identifying assumption for the DiD model can be summarized as:

Unobserved differences between the treatment and control groups are similar before and after treatment (in average)

or

Unobserved time trends for the treatment and control groups are the same.

³As above, as no-one in the control group is ever treated, both $Y(0|T = 0, t = 0)$ and $Y(0|T = 1, t = 1)$ are observable so no counterfactual assumption is needed here.

Example 5.4.8: COVID-19 Epidemic and Presidents Approval

Political leaders often enjoy a strong support during the time of crisis. Did the same also apply to the US president Donald Trump in spring 2020, during the COVID-19 epidemic? Let's answer this question with polling data. But as presidents' rating ebbs and flows over time, we compare Trump with Barack Obama using differences-in-differences approach. As spring 2020 was Trump's fourth year in office, we compare his approval trend with that of Barack Obama in 2016, fourth year of Obama's second term in office. The identifying assumption here is that the approval rate trends for Trump in 2020 were similar to those of Obama in 2016, had the COVID epidemic not happened.

A sample of the data is in the table below:

Table 5.5: Approval ratings of presidents Obama and Trump during their fourth year in office. Polling data from RealClearPolitics. The displayed period, from mid-January to mid-April centers on mid-March, the weeks in 2020 where the world, including the US, rapidly realized the magnitude of the unfolding health crisis.

poll	date	approve	president
The Economist/YouGov	2016-01-17	43	Obama
Bloomberg	2016-03-21	50	Obama
NBC News/Wall St. Jrnl	2016-05-17	51	Obama
ABC News/Wash Post	2020-01-22	47	Trump
Economist/YouGov	2020-02-10	45	Trump
Reuters/Ipsos	2020-04-13	46	Trump

We choose a single day, March 15th as the day of "treatment". By March 15th 2020 the coronavirus epidemic had become the leading issue in US media and politics. The number of infected and dead was increasing rapidly and within a week California ordered the first state-wide lockdown. Hence "before" are polls conducted before March 15th, and "after" are later polls. The treatment group is made of Trump, as the pandemic occurred on his watch. Obama did not experience anything similar in 2016 and hence he forms the control group. When we compute the group/time period averages, we get an analogue to the Table 5.4:

Table 5.6: The effect of COVID-19 pandemic on president's approval rate

Time	Control (Obama)	Treatment (Trump)	Difference (Trump - Obama)
Before (before March 15)	45.9	45.1	-0.86
After (after March 15)	48.1	46	-2.11
Difference (After-Before)	2.21	0.96	-1.251

We can see that the average approval rate for both presidents between mid-January and mid-March was fairly similar around 45% while Obama was enjoying a 0.86 pct point lead. However, by end of March–early April Obama’s lead had increased to 2.11 pct points. The difference of these two figures is the effect estimate—−2.15 pct points. Alternatively, we can look at the growth of the popularity of both presidents over the same time period. During this time, Obama gained 2.21 pct points of approval while Trump 0.96 pct points. By construction, the difference is exactly the same number, −1.25.

We may depict this estimator graphically by plotting two lines, one for Obama and one for Trump, for two time points, “before” and “after”:

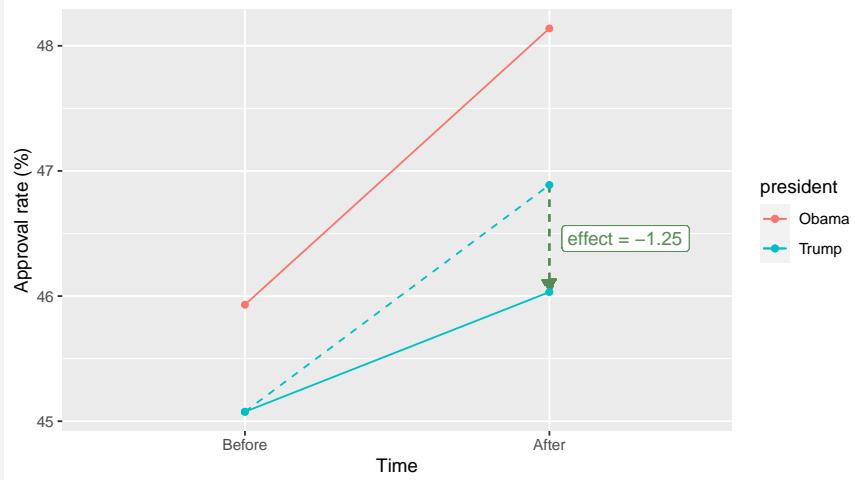


Figure 5.3: Average presidents’ approval rate before and after March 15th of their 4th year in office. We can see that Obama’s approval increased by more than two percentage points over this period while that of Trump grow by slightly less than one point. The dashed blue line depicts the counterfactual—the path of Trump approval rate, if it had been similar to that of the Obama’s. The difference between the counterfactual and the actual approval (green dashed line), −1.25 points, is the effect.

This analysis suggests the epidemic was actually hurting the president’s standing. Do we believe the result is correct? It is certainly plausible—unlike Korean president Jae In Moon for instance, Trump was not a leading figure in driving the nation’s response to the virus. However, our belief should fundamentally depend on whether we believe in the identifying assumption: without the virus, Trump’s approval in 2020 had followed a similar trend as Obama in 2016.

Table 5.4 treats the data as if we have just a single observation in each 4 cells of the table. But we may have more data, and we may have additional variables we may want to control for, for instance political preferences, age, place of residence, and other characteristics of the respondents. In this case

we can use linear (or other type) regression instead of the tabulation. As in the examples with presidents' approval, we may observe multiple polls for both before and after period. Linear regression can easily capture the trend with a term $\beta_1 \cdot \text{after}$, and difference between the treatment and control groups by $\beta_2 \cdot \text{treatment}$. However, if we use these two terms only, we assume the trends are equal for both groups and hence by construction the effect is zero. So we also need an interaction term of the form $\beta_3 \cdot \text{after} \times \text{treatment}$ (see Section 4.1.6) to allow the trends between the groups. So the regression model will look like

$$y_{it} = \beta_0 + \beta_1 \cdot \text{after}_{it} + \beta_2 \cdot \text{treatment}_{it} + \beta_3 \cdot \text{after}_{it} \times \text{treatment}_{it} + \epsilon_{it}. \quad (5.4.16)$$

Here β_0 captures the baseline effect, the average outcome for the control group before treatment; β_1 captures the difference in the baseline trend, the outcome growth for the control group from "before" to "after"; β_2 captures the baseline difference between treatment and control groups (before treatment); and finally, β_3 is the estimated difference in time trends for the treatment and control group. The last figure, β_3 , is exactly the DiD estimate we are looking for. Hence, in order to estimate DiD using linear regression, you have to include:

- a) intercept β_0 ,
- b) a term for after-treatment time period $\beta_1 \cdot \text{after}$,
- c) a term for the treatment group $\beta_2 \cdot \text{treatment}$,
- d) an interaction effect for treatment group after the treatment $\beta_3 \cdot \text{after} \times \text{treatment}$.

The latter is the estimate of interest. We may add additional controls here, such as respondents' political preferences, state, and so forth.

Example 5.4.9: President's approval rating: the regression approach

Let's return to the example of Obama's and Trump's approval rating. We select a the time period from mid-January to mid-April of the fourth year of their presidency, as in Example 5.4.8. If fact, our dataset contains 149 polls for this period (See Table 5.5), so we have many observations for each table cell. We estimate the following model:

$$y_{it} = \beta_0 + \beta_1 \cdot \text{after}_{it} + \beta_2 \cdot \text{Trump}_{it} + \beta_3 \cdot \text{after}_{it} \times \text{Trump}_{it} + \epsilon_{it}. \quad (5.4.17)$$

We get the following results:

.	object	..
Intercept	45.931	<i>0.433***</i>
after	2.208	<i>0.582***</i>
presidentTrump	-0.856	<i>0.539</i>
after × presidentTrump	-1.251	<i>0.785</i>
# obs	149	
R ²	0.2065	

Table 5.7: DiD regression estimate for the effect of COVID-19 epidemic on the US president's approval rating. Standard errors in italics.

As expected, the regression approach gave us exactly the same numbers, including the main effect: $\text{after} \times \text{Trump} = -1.25$ as the table-based approach.

Unless we introduce additional controls, linear regression just compares the averages. But unlike the table above, we now also have standard errors. These suggest that *after*, the spring-2016 trend for Obama, is indeed statistically significant. However, none of the Trump-related effects is statistically significant. The polling average for Trump is a little bit less than that for Obama, and his polling numbers have been lagging even more over the spring, but both effects are small and may well be just sampling noise. Hence, we can conclude that the epidemic did not give Trump any noticeably boost, and may instead have hurt him slightly.

To recap, let's list here all the predicted values:

- The baseline approval rate, for Obama, before mid-March, was $\beta_0 = 45.93\%$.
- For Trump, the approval rate was slightly lower, $\beta_0 + \beta_2 = 45.075\%$.
- After mid-March, Obama experienced a mild increase of $\beta_1 = 2.208$ pct points.
- After mid-March, Trump's growth was somewhat smaller than Obama's, $\beta_1 + \beta_3 = 0.957$ pct points, leading the average approval rate to $\beta_0 + \beta_1 + \beta_2 + \beta_3 = 46.032\%$.
- The main effect of interest here, the difference in springtime growth, is $\beta_3 = -1.251$ pct points. However, this is not statistically significant.

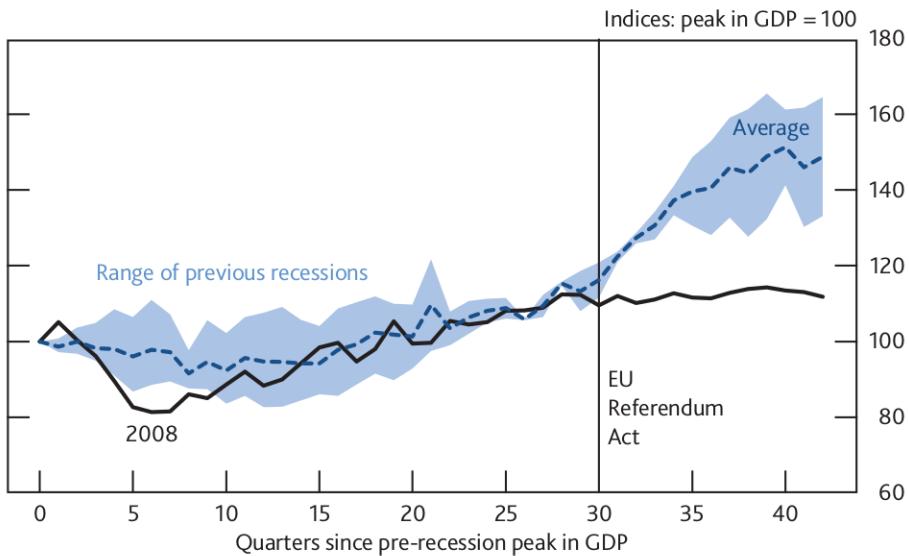
DiD estimators are sometimes used in a less formal context. Figure 5.4, taken from [of England \(2019\)](#), p 14), compares the 2009 recession and the following recovery with “previous recessions”. One can see that while the previous recessions were followed by a substantial investment growth, this has not been the case since the Great Recession. After the Brexit referendum decision was made in 2015 (marked as the *EU Referendum Act* on the figure), the investment level has remained essentially flat. The authors conclude that “weak investment appears to primarily reflect Brexit and associated uncertainty”, a conclusion that also receives support from investor surveys.

5.5 Cognitive Illusions in Causal Inference

Humans brains are developed to serve us well in typical everyday situations we encountered through the past hundreds of thousands of years. However, accurately establishing causality based on observational or experimental data has apparently not been an important survival task for our ancestors. This manifests in cognitive biases related to causal inference.

Consider a binary treatment-binary outcome data, such as the flu shot-flu example in Section 5.2. This data can always be displayed as a four-cell contingency table (Table 5.8). It depicts four potential outcomes, for instance a denotes the count of cases where both the treatment and outcome are absent. This is typical data humans observe, it is much more rare to be able to directly manipulate the treatment in order to conduct something that resembles a RCT. Evolution has taught us to deduce causality from such case counts.

Business investment after previous recessions^(a)



Sources: ONS and Bank calculations.

- (a) Chained-volume measure. Recessions are defined as at least two consecutive quarters of negative GDP growth. Previous recessions include those beginning in 1973, 1975, 1980 and 1990. A recovery ends if a second recession occurs in the period shown.

Figure 5.4: Effect on Brexit referendum on the business investments in UK: an example of graphical DiD approach.

We are inclined to believe “treatment” causes “outcome” if we see many cases in cells *a* and *d* while *c* and *b* remain relatively empty, and there are no obvious confounding factors. It should be clear to the reader by now that such data alone is not enough to establish causality. However, very often this is all we have, and we have to use such information to successfully live in the environment we live in. Remember—we are talking about a time frame of hundreds of thousands of years, most of which our ancestors spent as hunter-gatherers in African savannas.

It turns out that humans are more likely to believe treatment causes outcome ([Matute et al., 2015](#)) if

1. The outcome is very likely, say, 75% or more. This is called *outcome-density bias*.
2. The treatment is very likely (*cause-density bias*).

Both biases are related to the cell *d* in the table being well populated compared to the other cells, and hence reinforce each other. People are likely to believe in bogus causal relationships if both treatment and outcome are very likely, for instance when they take a homeopathic pill every few hours while the ailment goes away rapidly. A simple remedy to counter this bias is to recommend people

Table 5.8: Four potential outcomes in binary treatment/binary outcome data. “0” and “1” denote presence and absence of treatment and outcome, the letters in cells are the corresponding case counts.

Treatment	Outcome	
	0	1
0	a	b
1	c	d

to lower the frequency of treatment. It makes the d -cell case count smaller and hence the bias will be less strong.

5.6 Causality and complex social problems

Sometimes the causal chains are much more complex and harder to predict than is apparent when someone first encounters the problem. This is often the case if the question are related to humans and social problems. Below we walk through an example, namely usage of mandatory bicycle helmet laws.

5.6.1 Effect of bike helmet laws

It is well established that bicycle helmets substantially reduce head injuries during certain type of crashes. Is this evidence enough to justify mandatory helmet laws (MHL-s)? It turns out we need much more evidence.

The fact that helmets help to prevent head injuries is best established through mechanical experiments where model heads, with and without helmet, are dropped to hard surface. One can find information for both about frontal impact ([Cripton et al., 2014](#)) and for oblique impact ([Mills and Gilchrist, 2008](#)). In such experiments the researchers have full control over the environment, such as impact speed, type of helmet, hair and skin properties and so on, and in this sense they answer the exact question they are designed for very well. The question in the above-cited papers is about head injury when hitting hard surface at given speed.

Obviously, head injury is not a random process that only depends on the presence of helmet—there are many more decisions involved. For a start, one has to decide whether to cycle or not. Thereafter one chooses the route, speed, and makes other decisions about biking like distance from curb, whether to pass someone, etc. Also the other road users choose their behavior, such as motorists must decide speed and distance when passing a cyclist. So there are many reasons to believe that such studies do not give the complete picture.

1. Typical crashes occur at different angles and surfaces, and not necessarily in conditions similar to that of the laboratory environment. But as the experiments get better, we can assume the laboratory models get increasingly close to the real cases.

2. in certain circumstances the helmet may get stuck and hurt the wearer more than would be the case without helmet (rotational injuries). So far, the non-experimental evidence from actual accidents tends to indicate that helmets help to prevent head injuries by a substantial degree ([Amoros et al., 2012](#)), so the cases where helmets hurt are probably rare in practice. However, we are outside of controlled experiment realm now.
3. cyclists may act differently depending on whether they wear or do not wear helmets. In particular, wearing helmet can lead to more risky behavior (risk compensation). [Fyhri et al. \(2018\)](#) does not find any effect of wearing helmet on cyclists' speed in a field experiment. However, the study was limited in terms of number of participants (31) and situations encountered on the road. More research is needed here but it is much harder to simulate realistic situations here.
4. MHL-s may discourage cycling. As crashes are rare, the net health effect may be negative if, in order to avoid rare crashes, people avoid the healthy exercise in the first place. However, we may debate if authorities should strive toward fewer crashes, or more healthy population.

The discouragement may occur through several mechanisms:

- (a) helmets are considered inconvenient, either to wear, or to carry around
- (b) the authorities are using scaring tactics to make the point for helmets. This may make people afraid of biking in first place instead of choosing helmets.
- (c) helmet requirement makes bike shares less harder to implement.

These effects are very hard to pinpoint in experiments.

5. If MHL discourages cycling, it also makes cycling less safe through following mechanisms:
 - (a) “safety in numbers”: the less bikes there are on street, the less the motorists expect to encounter them, the less prepared they are to notice cyclists in traffic and hence the more likely are the accidents.
 - (b) some people may choose driving over cycling increasing the amount of motorized traffic.
 - (c) fewer cyclists also means less political will to invest in cycling infrastructure.
6. helmets may also cause drivers to behave differently and behave more (or less) risky with respect to cyclists. For instance, [Walker \(2007\)](#) finds that cars passed cyclists with helmets significantly closer in average.

Note that despite of the large number of potential mechanisms, the net effect may be dominated by just one or two major ones while all the others are of very little importance. The problem is that we don't know how strong are each of these effects, and hence the policy will remain largely uninformed.

Chapter 6

Assessing model goodness

6.1 Categorization

We discussed model evaluation in the context of linear regression above in Section 4.1.3. Here we focus on evaluating categorization. However, RMSE is not an appropriate model goodness indicator for categorization.

In case of continuous outcomes, such as income or light intensity, a good model will be the one that predicts values very close to the true observed ones. Hence the measure should be based on $|\hat{y}_i - y_i|$, the difference between the observed and predicted values. But this approach does not really work for categorization for two main reasons:

1. Categories are nominal or ordinal measures (see [Section 1.1.1, Measures: Possible Mathematical Operations](#)). Hence subtraction $\hat{y}_i - y_i$ may not even be defined, as, for instance, in case of predicting gender based on user purchase history.
2. Second, even if we somehow define the difference, the result will be either 0 if our prediction is correct, or 1 if it is not correct (or a number of other integer values if we have more categories). This does not align well with $|\hat{y}_i - y_i|$, the distance between prediction and the true value.

A number of solutions are based on confusion matrix.

6.1.1 Confusion matrix and related concepts

Instead of attempting to measure distance between the predicted and true values, we just tabulate and count all types classification errors. This is the essence of *confusion matrix*. It turns out that this simple approach allows to assess the models in many relevant ways.

Confusion matrix

A simple and easy way to assess the goodness of predictions is to create a cross table of the actual and predicted classes. Such a table is called *confusion matrix* and it is a central concept in many categorization-related goodness measures. Here we discuss confusion matrix in case of two categories only but it easily generalizes to a larger number of classes.

Assume we have in total T cases from two categories: P positive cases “+”, and N negative cases “-”. One can imagine this is a medical diagnosis where the negative ones do not suffer from the disease while the positive ones were diagnosed the disease. These are “actual categories”, created either manually or in another way, possibly through expensive testing or diagnosis, so we know these are correct. Now we use a model to predict the category for each case. The model predicts \hat{P} cases as positive and \hat{N} cases as negative. Obviously, we would like to have every single case predicted correctly but this is seldom the case. In order to get a good overview about our prediction results, we can create a 2×2 cross-table where we present the counts for actual and predicted classes (Table 6.1). The table indicates how many actual positive cases were predicted as positive, how many as negative, and so on. This is confusion matrix.

Table 6.1: Example confusion matrix for two categories, labeled here as “-” and “+”. The table entries are counts: TP , true positives, refers to positive cases that were also predicted to be positive, P is the number of actual positive cases. See explanations in the text.

Actual	Predicted		
	-	+	Total
-	TN	FP	N
+	FN	TP	P
Total	\hat{N}	\hat{P}	T

In case of two categories, the core of confusion matrix contains four cells:

- *True positives (TP)* are cases that are actually positive, and are correctly predicted as positive. We like TP to be large.
- *True negatives (TN)* are actually negative and are predicted as negative. We like TN to be large as well.
- *False positives (FP)*, also *type-I errors*, are cases that are actually negative but were predicted as positive. We would like FP to be zero.
- *False negatives (FN)*, also *type-II errors*, are cases that are actually positive but were predicted as negative. We would like FN to be zero as well.

In case of confusion matrix, these concepts typically refer to the corresponding counts, e.g. FP is the number of cases we incorrectly predict as positive. However, these may also refer to probabilities, e.g. FP may be a probability that we predict a case incorrectly as positive. Obviously, in case of a good model we have high values of TP and TN while the counts of FP and FN are small. Table 6.1 also includes one-way counts: P is the total number of actual positives, N the number of actual negatives, \hat{P} number of predicted positives and \hat{N} that of predicted negatives. Finally, T denotes the total number of cases.

Although a 2×2 table seems simple, confusion matrix is actually surprisingly confusing. It is partly related to the notation and language. In particular, *true positives* refer to cases that are actually positive, and are predicted as positive; not to the “ground truth”, the cases that are actually positive as one may think. This is why we introduce the “actual” status here, to distinguish between the actual positives P and “true” positives TP . In addition, N typically denotes the total number of cases, not just the number of actual negatives. Here we denote the total number of cases by T .

Based on these numbers, we define a number of model goodness measures:

- *Accuracy*: percentage of correct answers

$$\text{Accuracy} = \frac{TP + TN}{T}$$

Accuracy is an easy and intuitive summary measure: what percentage of our predictions turn out to be correct. However, it is not very informative in case of very inequally sized categories as even a naive model that always predicts the largest class can achieve high accuracy.

- *Recall* is the percentage of actual positives that are correctly identified (recalled) as positives

$$\text{Recall} = \frac{TP}{P} = \frac{TP}{TP + FN}.$$

If our main concern is to capture all positives, recall may be a good measure. However, it is easy to fool: if we predict positive for every case, we get $Recall = 1$ but the model is hardly of any use.

- *Precision* is a sort of mirror image of recall: percentage of predicted positives that turn out to be correct.

$$\text{Precision} = \frac{TP}{\hat{P}} = \frac{TP}{TP + FP}.$$

Precision may be a good measure if avoiding false positives is a major concern. As the other measures, this can also be fooled easily: if we ensure that only the most likely cases are labelled as positive, we an ensure that precision is high.

- *F score* is an attempt to find a balance between recall and precision. It is just the harmonic mean of these measures

$$F = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

Example 6.1.1: Accuracy, Precision, Recall, F-score

Dataset *Treatment* contains information about individual participation in a labor market program (treatment), and various background information, such as age, previous unemployment, and income. Here we use that information to estimate a logistic regression model to predict the treatment status based on age, previous real income and previous unemployment. The original data has 185 treated individuals out of 2675 individuals in total, while our model predicts 134 as treated. When we create confusion matrix, a cross-table of actual and treated values, the results looks like this:

Actual	Predicted		total
	non-treated	treated	
non-treated	2452	38	2490
treated	89	96	185
total	2541	134	2675

From the table we can compute all four model performance measures:

$$\text{Accuracy } A = \frac{TP + TN}{T} = \frac{96 + 2452}{2675} = 0.953 \quad (6.1.1)$$

$$\text{Recall } R = \frac{TP}{P} = \frac{96}{185} = 0.519 \quad (6.1.2)$$

$$\text{Precision } P = \frac{TP}{hat{P}} = \frac{96}{134} = 0.716 \quad (6.1.3)$$

$$\text{F-score } F = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2}{\frac{1}{0.716} + \frac{1}{0.519}} = 0.602 \quad (6.1.4)$$

So our model has very high accuracy but not impressive recall and precision. This is because the groups are of very different size: only 185 (6.9 percent) of individuals are treated, and hence if we would predict "non-treated" for everyone, we would still get accuracy 93.1 percent. So despite of the impressive accuracy, the model does not do actually much better than a naive guess. $R = 0.519$ tells that we only catch slightly over 50% of the treated individuals correctly, and $P = 0.716$ shows that only around 70% of predicted treatment cases are correct. Finally, F -score is predictably between R and P .

These model goodness measures have multiple names, and they are closely related to other similar measures. A number of examples are listed here:

- Accuracy-based measures

Misclassification rate is just the opposite of accuracy:

$$\text{Misclassification rate} = 1 - A \quad (6.1.5)$$

- Recall-based measures

True positive rate and **sensitivity** are just another names for recall

Specificity is recall for negative outcomes:

$$\text{Specificity} = \frac{TN}{TN + FP} = \frac{TN}{N} \quad (6.1.6)$$

False positive rate is 1 minus specificity:

$$\text{False positive rate} = 1 - \text{specificity} = \frac{FP}{N} \quad (6.1.7)$$

Note that for each model we can only compute a single accuracy measure but two P and R measures: one for positive and one for negative cases. Typically it is obvious from the problem which cases we should analyze. For instance, in case of medical diagnosis, the “positive” cases often mean the illness, and we may be concerned about catching as many cases as possible (we need a high recall). Alternatively, if the treatment is expensive and potentially harmful, we may be interested to ensure all of the cases we identify are actually correct (we look for high precision).

Exercise 6.1.1: Accuracy, Precision, Recall

Look at categorizing cases into two color categories: Red and Yellow. Consider the confusion matrix:

		Predicted	
		Red	Yellow
Actual	Red	10	20
	Yellow	10	60

Assuming Yellow is the positive, compute A , P , R and F -score.

ROC curve

Categorization models normally do not just predict the class, but the *probability* that the observation belongs to each class. It is customary to take probability 0.5 as the threshold between the categories. If the predicted probability is less

than the threshold, it belongs to one, if it is above the threshold, it belongs to the other category. Say, probability 0.25 corresponds to “spam” and 0.6 to “no-spam” category.

While value 0.5 is intuitive and exactly in the middle of the range, we do not have to pick this value. If different type errors are associated with different costs, we may be much more willing to err in one side than another and pick a threshold noticeably different from 0.5. For instance, a judge may consider 0.9 as a too low confidence to sentence someone for a felony, because such decision, if wrong, will have serious consequences for the defendant. Obviously, our predictions change if we pick a different threshold, and different models may show different behavior here. ROC curve (*receiver operating characteristics*) is a way to make the type-I/type-II error trade-off explicit, and help the user to choose between different models and thresholds.

ROC curve is also based on confusion matrix related concepts, true positive rate, *TPR*, (i.e. recall) and *false positive rate*, *FPR*, defined as

$$\text{False positive rate} = \frac{FP}{N}.$$

While *TPR* measures the percentage actual positives that are identified correctly, *FPR* measures the the percentage of actual negatives, incorrectly identified as positives. Obviously, we want out model to show high *TPR* (ideally 1) and low *FPR* (ideally 0).

ROC curve makes these tradeoffs explicit. It plots TPR agains FPR for different thresholds. A typical ROC curve is shown in Figure 6.1. The figure shows to models, linear probability model and logistic regression, addressing labor market training participation as a function of age, experience unemployment, and other individual characteristics. Typically all models offer two extreme choices: $FPR = TPR = 0$ and $FPR = TPR = 1$. The first corresponds to the case where all observations are predicted to be negative, the other to the cases where these are predicted to be positive. (Here it is not the case for LPM as the predicted probabilities may be outside $[0,1]$ interval.) Obviously, we are interested in the cases in the middle where *FPR* is low but *TPR* approaches to one. The figure suggests that logistic regression clearly outperforms LPM at low *FPR* values. For instance, if *FPR* = 0.1, *TPR* for LPM is approximately 0.9, but for logit 0.95.

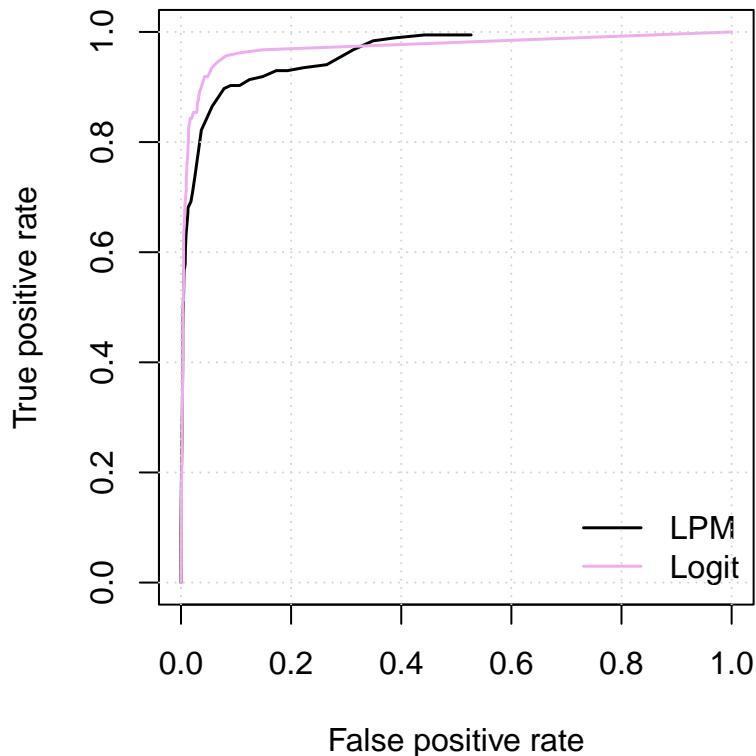


Figure 6.1: Example ROC curve for linear probability model (black) and logistic regression (pink). The figure suggest that in most cases logit outperforms LPM as it is able to achieve higher TPR over TPR range 0 to 0.3.

Example 6.1.2: Computing ROC curve

Let's look at a case where we have two classes: "0" (negative) and "1" (positive). We are working with four cases only. We run our model and the algorithm predicts the following probabilities:

case	Pr(positive)	true value
1	0.3	0
2	0.4	1
3	0.6	0
4	0.7	1

Denote the probability threshold value by θ , i.e. if $\text{Pr}(\text{positive}) > \theta$, we predict the case to be positive, otherwise it is assigned to the negative category. If we pick $\theta = 0.5$, our algorithm predicts the cases 3,4 to be positive and 1,2 to be negative. This is the most intuitive approach, but may not be the best if type-I/type-II errors have very different price.

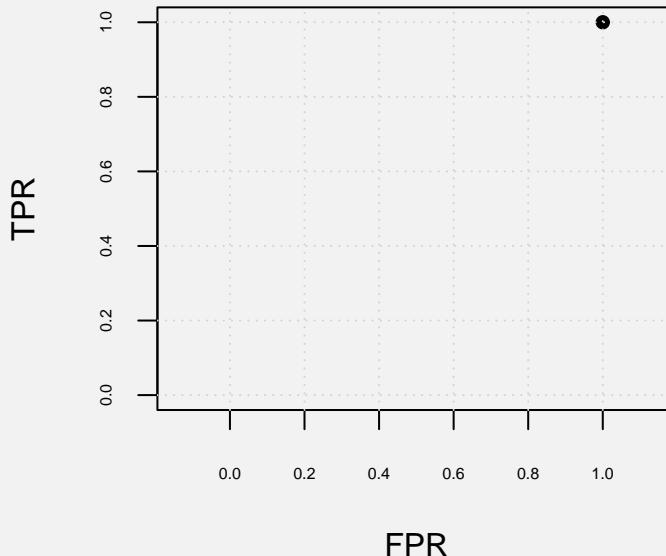
ROC curve is what makes these tradeoffs explicit. Let's start with an extreme threshold, $\theta = 0$. Now the algorithm predicts everything to be positive^a and the confusion matrix will look like

		Predicted	
		1	0
Actual	1	2	0
	0	2	0

Note that here $P = 2$ and $N = 2$. Obviously, we get all the actual positives, but we get all actual negatives wrong. This corresponds to the true positive and false positive rates

$$\text{TPR} = \frac{TP}{P} = \frac{2}{2} = 1 \quad \text{FPR} = \frac{FP}{N} = \frac{2}{2} = 1 \quad (6.1.8)$$

and will give us a data point on the ROC curve:



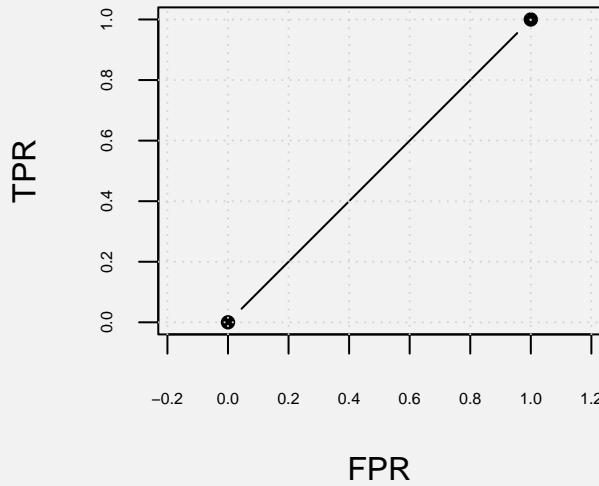
On the other extreme, we can take threshold $\theta = 1$. Now none of the cases is predicted positive and the confusion matrix will be

		Predicted	
		1	0
Actual	1	0	2
	0	0	2

All the negatives are correct but all positives are wrong, and the true positive and false positive rates are accordingly

$$\text{TPR} = \frac{2}{2} = 0 \quad \text{FPR} = \frac{2}{2} = 0. \quad (6.1.9)$$

This will correspond to the lower-left corner of the ROC curve:



Note that now the ROC curve is already a curve, not just a single point. Such two extreme cases are always possible with a naive model that predicts either all cases positive or negative, and hence do not tell us much about the underlying model.

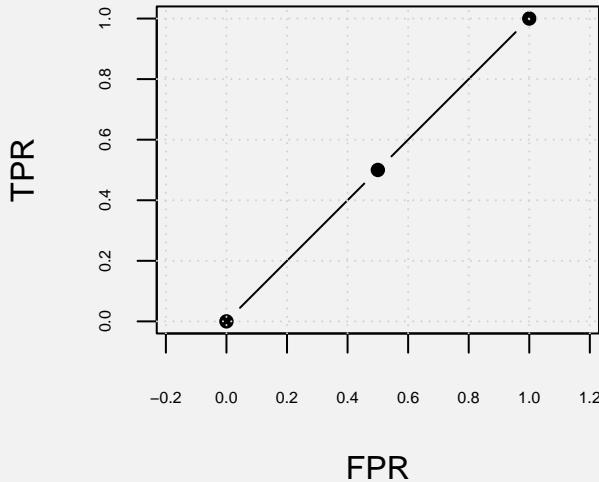
As a third example, take the threshold $\theta = 0.5$ and hence cases 1 and 2 will be predicted negative and cases 3, 4 positive. The confusion matrix is

		Predicted	
		1	0
Actual	1	1	1
	0	1	1

Now half of the predictions are correct and a half are wrong:

$$\text{TPR} = \frac{1}{2} = 0.5 \quad \text{FPR} = \frac{1}{2} = 0.5. \quad (6.1.10)$$

We get another point in the middle of the same ROC curve:



As above, we got a point on the same line that denotes random outcomes. This indicates that our model performs exactly as good as a naive model that randomly predicts half of the cases negative and the other half positive.

^aIt predicts literally *everything* to be positive only for such models, like logistic regression, where predicted probabilities are in the interval (0,1). This may not be the case for e.g. [k-NN](#) (predicted probability may be 0) or for LPM (predicted probability may be negative).

Limitations of the confusion matrix approach

While confusion matrix offers us a large number of intuitive indicators for the model performance, it is oblivious about the confidence of our estimators. As long as the predictions do not change, the increased confidence in the predictions is not reflected in the results. This makes it hard to compare models on small datasets as small changes in categorization results may obscure more important underlying confidence effects.

6.2 Overfitting and Validation

6.2.1 What is overfitting

Estimating a models makes it to fit data as well as possible. But model can only fit training data, the data it sees during the estimation. Sometimes the difference

is small, as simple models may be quite rigid and all data looks reasonably similar. But in case some regions in the data space are only sparsely covered in training data, a flexible model may produce results that are substantially worse than another, less flexible model. This is referred as *overfitting*. Note that in high-dimensional space, the data is always sparse, and there are always large uncovered regions, though not always do we need to make predictions in such regions.

Consider the artificial example in Figure 6.2. The left panel shows the upward trending relationship between age and income. The right panel shows a number of model fits for the relationship on the same data. All model contain polynomials of age in the form¹

$$y_i = \beta_0 + \beta_1 \cdot \text{age}_i + \beta_2 \cdot \text{age}_i^2 + \beta_3 \cdot \text{age}_i^3 + \cdots + \epsilon_i. \quad (6.2.1)$$

The first model, of degree 0, contains just the constant term β_0 , essentially assuming that income is independent of age. Degree 1 is the linear model, degree 2 is a quadratic relationship, and so on. The higher the polynomial degree, the more flexible the model. This is because higher degree models contain more base functions, and by combining more functions we are able to match the data better.

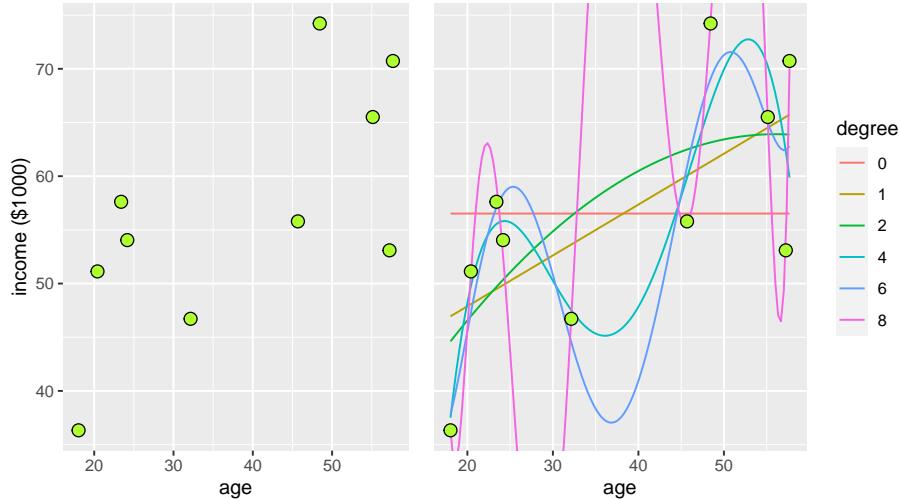


Figure 6.2: Artificial age-income data. Left panel shows just the data points, the right panel displays the same data and a number of polynomial regression models with various polynomial degrees.

Which model is the best one? Just by looking at the image, one may suggest

¹Actually, these example models here are created using orthogonal polynomials, not simple polynomials. In case of ordinary polynomials, high-order *age* terms will introduce a lot of multicollinearity and the numeric precision will be insufficient.

that degree 0 is too inflexible, the data seems to follow an increasing trend and a constant cannot capture it. The linear and quadratic model (degree 1 and 2) both capture the trend well. The quadratic model also captures the falling trend before age 35, linear model, obviously, only shows a constant trend. The 4th-degree polynomial seems somewhat more wobbly, and the sixth-degree model seems to fluctuate quite a lot. And the 8th-degree polynomial has gone completely wild and jumps up and down way outside of the image. However, note that the higher degree, the closer the model follows the actual data points. The wild 8th-degree line seems to be very close to most of the data points. This can be confirmed by computing the corresponding *RMSE*-s:

degree	0	1	2	4	6	8
RMSE	10.72	7.86	7.67	5.57	4.93	0.80

This is easy to understand intuitively: a more flexible model is more able to fit the actual data points. This is why 8-th degree model fits the data noticeably better. But the price is paid in the gaps between data points. There is nothing that limits the model's wobbles in those gaps, and hence flexible models can jump wildly. But what matters more—the better precision at the data points where we have the data, or unrealistic behaviour between the data points? At the end we are interested in model performance exactly in these gaps—after all, there is little need to make predictions at the data points as we know the answers at those points anyway.

But in order to formally evaluate the effect of the apparent misbehavior between data points, we have to know more. In this particular case the 8th-degree predictions for, say, a 30-year old seem completely out of touch with the reality, a linear fit feel much more realistic. But sometimes we may have reasons to believe in the more flexible model, and sometimes we just don't know what to expect.

Consider next a two-dimensional example in Figure 6.3. The figure depicts a categorization problem where the coordinate pairs (x_1, x_2) are classified into either red or blue. The dots are the training data (previous observations) and seem to show a red area at lower left and blue area at the upper right part of the figure. The decision boundary between these two areas follows a wavy pattern.

The left panel uses a simple linear decision boundary (estimated with logistic regression). This one is apparently underfitting: it does not capture the blue and red waves, clearly systematic, that cross over into the other color. On the right panel we can see a much more elaborate boundary (calculated with nearest neighbors) that carves out every single red and blue dot. This seems to be a too complex boundary and suggests we are overfitting.

Split your data into training and testing data

- use *only* training data for training
- use testing data for testing
 - Testing: just the final test. No further use of testing data allowed.

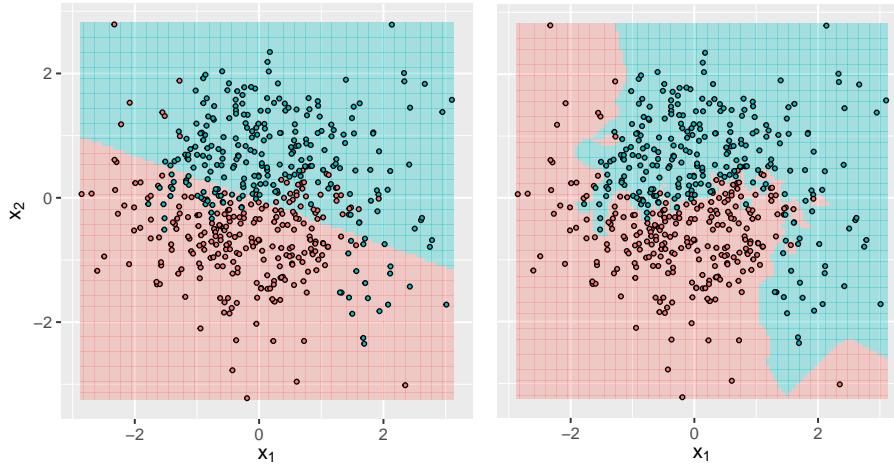


Figure 6.3: Categorization task with blue and red dots. Logistic regression (left panel) does not capture the red and blue “peninsulas” extending into the other color area. This is underfitting. Nearest neighbors (with $k = 1$) carves out a separate island for every single dot. The decision boundary seems too complex. This is overfitting.

- great if large dataset
 - if new data arrives over time
 - 80% for training, 20% testing
1. Split your data into working and testing (holdout) data
 - Testing data is only for the final test
 2. Split your working data into training and validation data
 - use the validation data for hyperparameter tuning
 - model selection
 - compare different models on validation data
 3. report the final performance using testing data.

Chapter 7

Machine Learning Models

7.1 k -Nearest Neighbors

Prerequisites: [Metric distance](#)

Nearest neighbors is one of the simplest and most intuitive machine learning methods. We predict the value, or a class, of a new observation as the class of the most similar observation in the training data set.

7.1.1 Introductory Example

Imagine we have data as depicted on Figure 7.1, left panel. It contains yellow and violet training observations, and our task is to categorize the empty unknown data points into one of these color categories. Intuitively, it is reasonable to assume that points that are “close” on the image should have similar color. So if an empty circle is fairly close to a violet training observation and far from everything yellow, we should consider violet as a good prediction for the unknown class.

This intuitive approach is the basis for *nearest neighbor* classification: we just categorize an unknown data point into the category that corresponds to the category of its closest neighbor. This has been done on the right panel of Figure 7.1: it divides the figure into tiny squares (101×101 squares) and categorizes the center of each square into either yellow or violet by looking at its closest neighbor’s color. The squares that coincide with the unknown data will tell us how the model will categorize these data points.

This baseline approach is very sensitive to individual outliers in the data. If we have a violet point sitting deep inside the yellow territory, we would immediately think that everything in the close neighborhood of the outlier also belongs to the violet class. Nearest neighbors does not allow for reasonable smoothing. Fortunately, a remedy here is very easy. Instead of using the category of the nearest neighbor as the predicted class, we can smooth the picture somewhat by using, say, 5 nearest neighbors, and finding the category that is preferred in this

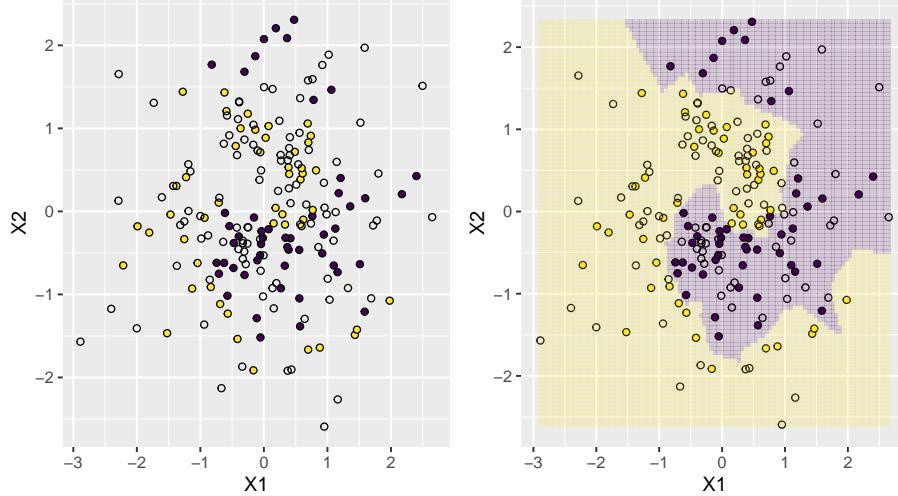


Figure 7.1: Example data: some of the datapoints are categorized into yellow and violet, but some are not (left panel). Intuitively, the empty circles should be classified according to a colored one nearby. This is the intuition of the nearest neighbor method. On the right panel, all the points that are closer to a violet one are painted violet and those that are closer to a yellow one are colored yellow. All the empty circles now lie in one of these areas of solid color and can be categorized either as yellow or violet.

group (often referred to as *majority voting*). If 3 out of the 5 closest neighbors are yellow and 2 are violet, we will pick yellow. This results in a noticeably smoother pictures (Figure 7.2 shows exactly the same data categorized using 5 and 25 nearest neighbors).

This is the essence of k -nearest neighbors (k -NN). In case of only two categories, k is often chosen to be an odd number in order to avoid ties in majority voting.

7.1.2 What is Distance

However, the simple and intuitive method is not without it's issues. As soon as we leave the 1-dimensional world, it may not be clear any more which observations are closer to each other. The nice example in Figure 7.1 was somewhat deceiving by making you to believe that what looks close on the image is also close in the data space. But take a simple example. Assume you are predicting house prices and you have data of recent sales like in Table 7.1, including the price (in \$1000), size (m^2), and neighborhood crime rate (incidents per 1000 residents). Your task is to predict the price of the house c that is similar to a in terms of neighborhood crime rate, and similar to the house b in terms of size. Which one is more similar? Your prediction will be very different depending on which one you choose as the nearest neighbor. Obviously, there is no cor-

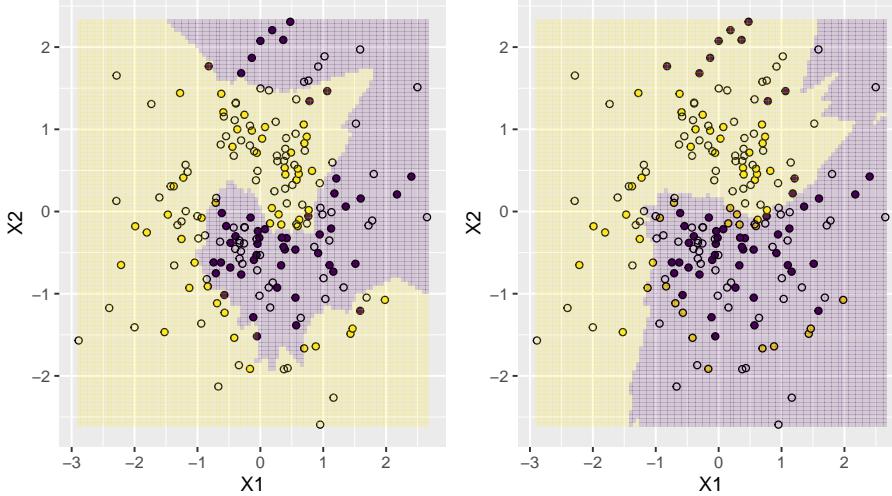


Figure 7.2: The same data points as in Figure 7.1, but now categorized based on 5 (left) and 25 (right) nearest neighbors. We can see that in the latter case, there are several groups of points that are embedded in the area of different color.

Table 7.1: Recent house sales

id	price (\$ 1000)	m ²	crime (per 1000)
a	800	200	0.2
b	1500	400	0.1
c	?	200	0.1

rect way to tell. We have to weight the different features somehow by using an appropriate distance metric.

Moreover, the previous example used simple numeric features, but this may not always be so. How can you tell which text is closer to another one? Which customer is more similar to a third one? In these cases we don't even have numeric measures to start with, and our decisions about creating those add an additional layer of assumptions into the model.

Obviously, one can always choose a pre-determined distance metric, either Euclidean or another one. Even more, k -NN does not require the metric to be a valid metric in the sense of vector spaces, it is enough if it allows us to order the observations by "closeness" in a consistent way. This opens the option for cosine similarity (more about it later).

7.2 Trees and tree-based methods

Decision trees is a popular algorithm for both regression and classification. Trees are easy to implement and understand, and in some sense they are easy to interpret. Our mental decision-making is often based on tree-like way of thinking.

Trees are also a popular way to “grow forests”, a large collection different tree-based methods combined into a single ensemble method.

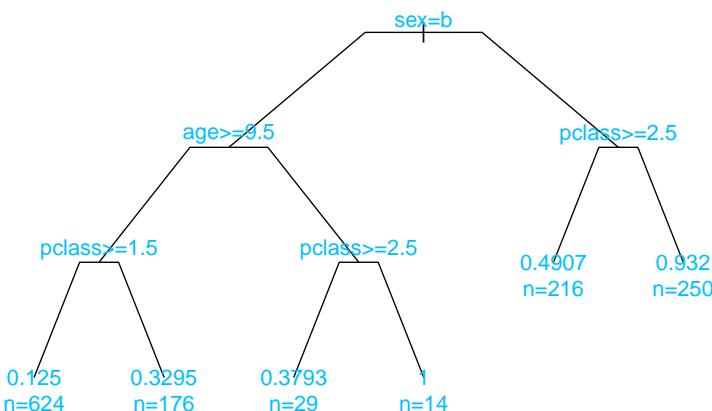
7.2.1 Decision Trees

As an example of using decision trees in game situations, think about the “animal game”, a children game where one player thinks an animal, and the other player have to guess it by asking question. The questions must be worded in a way that the answer is always “yes” or “no”. These yes-no answers build a tree-like structure, and if written down, one can display it as a decision tree.

However, the decision trees in such situations are based on our pre-existing knowledge and may be quite inefficient. In this section we discuss how to build decision trees based on data, how to do it in an efficient manner, and how to make ensemble methods (forests) out of many trees.

```
##      pclass survived     sex      age
## 1:      1          1 female 29.0000
## 2:      1          1 male   0.9167
## 3:      1          0 female 2.0000
## 4:      1          0 male   30.0000
## 5:      1          0 female 25.0000
## 6:      1          1 male   48.0000
```

left: true; “b”: female

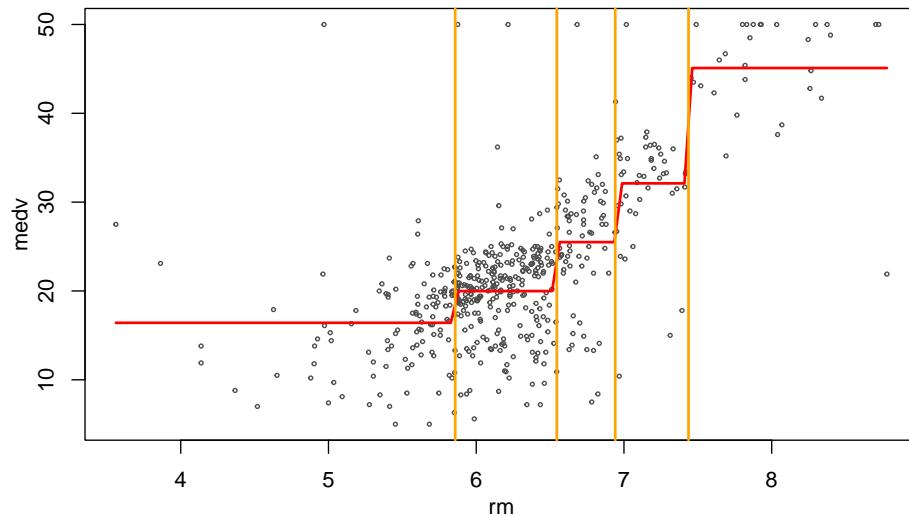


Advantages

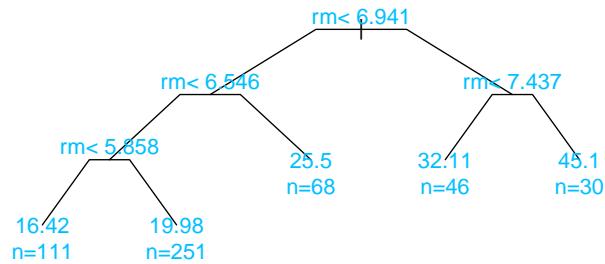
- popular, easy to understand
- reflects logic of decision-making
- arbitrary complexity
- classification and regression
- can be made compact (pruning)

Downsides

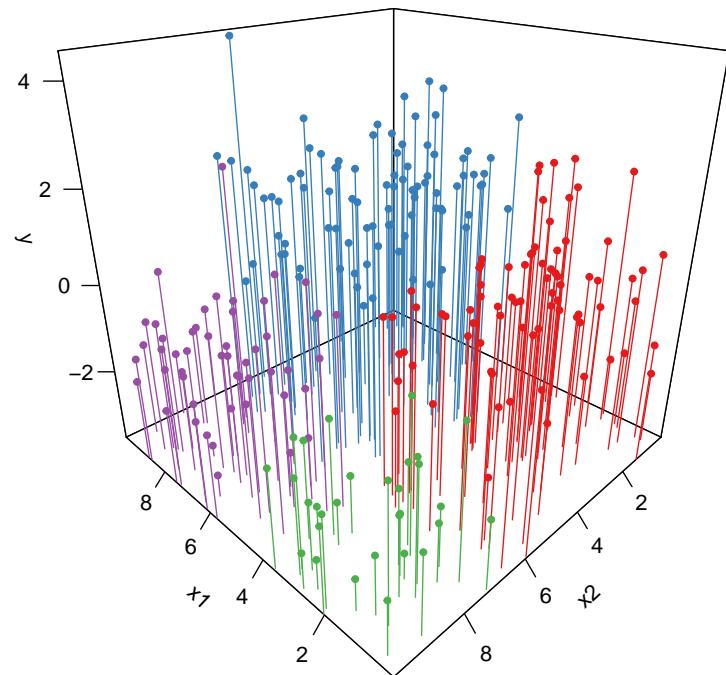
- sometimes hard to interpret
- Split your feature space into rectangles
- Predict the mean value in each rectangle
- Choose such a split that minimizes the total variance

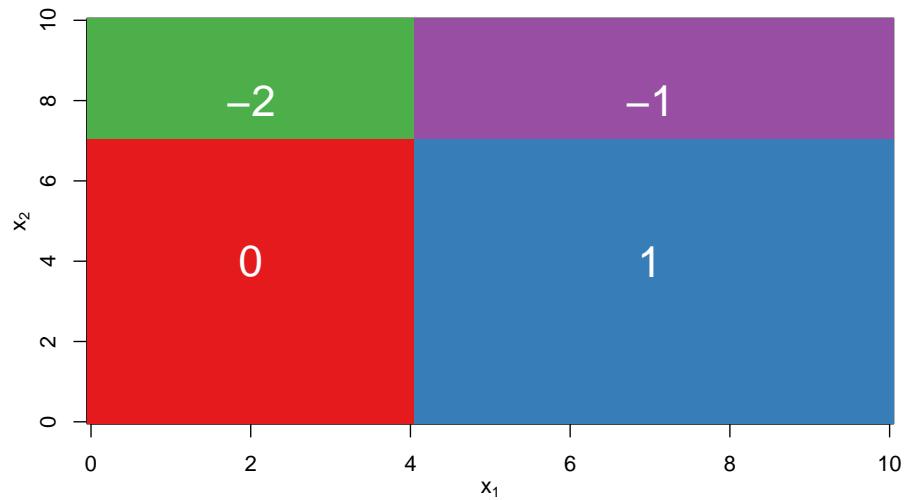


4 nodes, 5 leafs

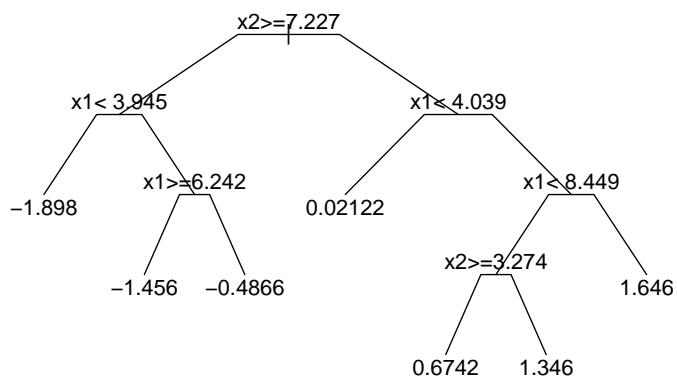


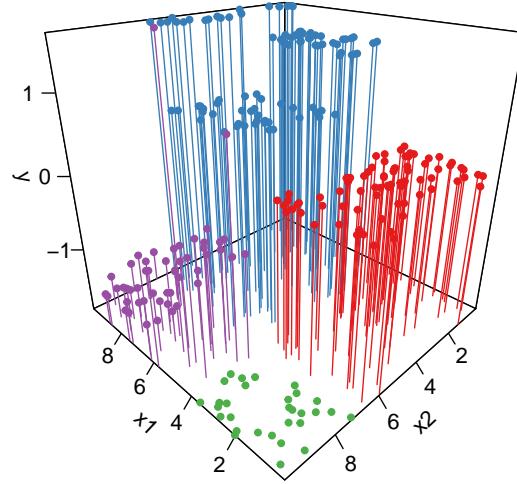
$$y = \mathbb{1}(x_1 > 4) - 2 \cdot \mathbb{1}(x_2 > 7) + \epsilon, \text{ 250 observations:}$$





(Yes \Leftrightarrow No)





We have predictors \mathbf{X} , outcome \mathbf{y} :

- $\mathbf{y}: N \times 1$
- $\mathbf{X}: N \times K$

Split the feature space into M rectangular regions \mathcal{R}_m , $m = 1, 2, \dots, M$.

Predict a constant value in each region:

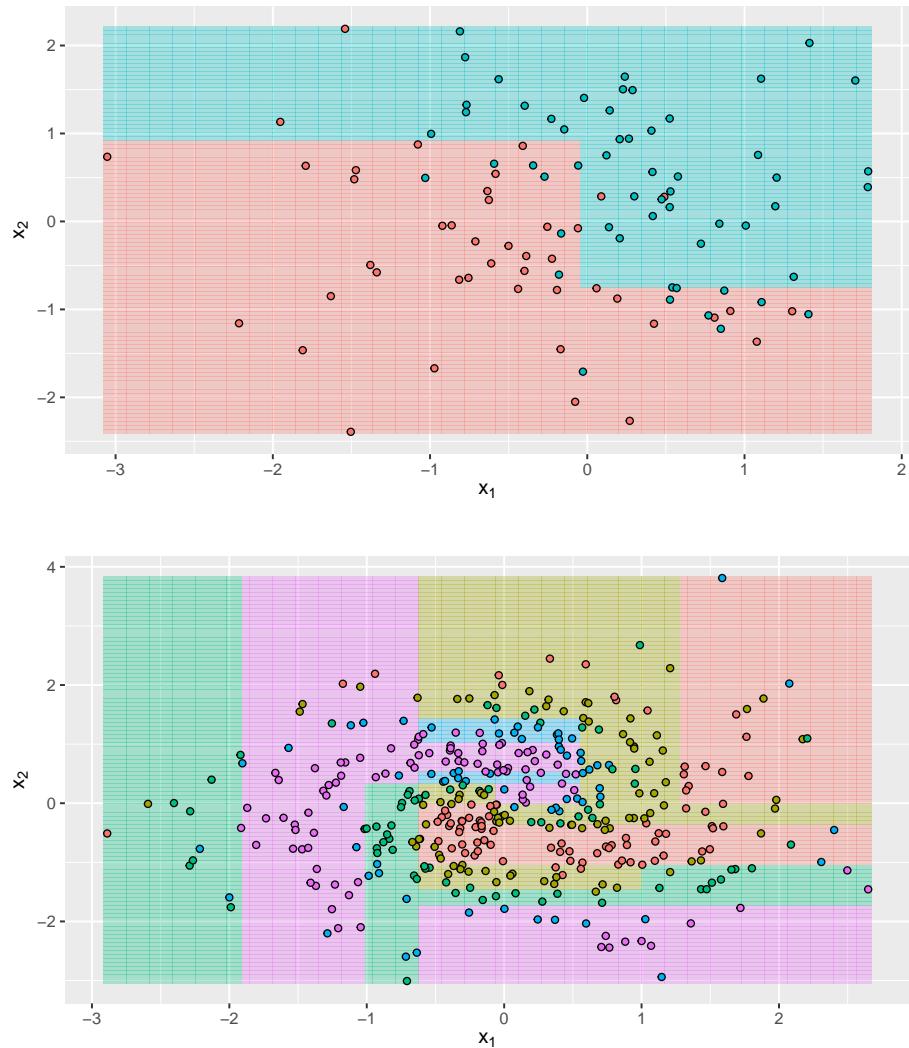
$$\hat{y}(x) = \sum_m c_m \cdot \mathbb{1}(\mathbf{x}_i \in \mathcal{R}_m)$$

Where c_m is the predicted value for region m .

c_m is the average in region m

$$c_m = \frac{1}{n_m} \sum_{i: x_i \in \mathcal{R}_m} y_i$$

- Split the feature space into rectangular blocks
- In each block predict the class in majority
- Try to do the splitting in a way that the rectangles are as “pure” as possible



Try all possible splits

- This is not possible

Second best

- Recursive binary splitting

(binary attributes)

Take data $(\mathbf{x}_i, y_i), i = 1 \dots N$.

1. If we have only $y_i = 0$ or $y_i = 1, i = 1 \dots N$, return a homogeneous leaf
2. otherwise:

- (a) Find “best splitting attribute” j
- (b) Split data into two sets along x^j value
- (c) Remove x^j from data
- (d) repeat the procedure on both branches.

- Data:

$$(\mathbf{X}, \mathbf{y}) = \left(\begin{pmatrix} \mathbf{x}'_{1\bullet} \\ \mathbf{x}'_{2\bullet} \\ \vdots \\ \mathbf{x}'_{N\bullet} \end{pmatrix}, \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \right) = \left((\mathbf{x}_{\bullet 1}, \mathbf{x}_{\bullet 2}, \dots, \mathbf{x}_{\bullet K}), \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \right)$$

K attributes of length N , a single endogeneous variable y of length N .

Assume binary features $x_{ij} \in \{0,1\}$.

```

1   function growTree((X, y))
2     let N = number of cases in X
3     if  $y_i = 0$  for all  $i = 1, 2, \dots, N$ :
4       return leaf(0)
5     if  $y_i = 1$  for all  $i = 1, 2, \dots, N$ :
6       return leaf(1)
7     let j = bestAttribute((X, y))
8       # the best way to split (X, y) into two parts
9       # one corresponding to  $\mathbf{x}_j = 0$ , the other to  $\mathbf{x}_j = 1$ 
10    let D = X-j
11    # remove feature j from data
12    return node(j, growTree((Di, yi) for i :  $x_{ij} = 0$ ),
13                  growTree((Di, yi) for i :  $x_{ij} = 1$ ))

```

Minimize quadratic loss inside of regions:

$$\hat{y}_j = \arg \min_{\hat{y}} \sum_{i: x_i \in R_j} (\hat{y} - y_i)^2$$

First split in 1 dimension:

$$s_1 = \arg \min_s \left[\min_{\hat{y}_1} \sum_{i: x_i < s} (\hat{y} - y_i)^2 + \min_{\hat{y}_2} \sum_{i: x_i \geq s} (\hat{y} - y_i)^2 \right]$$

n -th split among d dimensions: choose, s, d to minimize

$$\min_{\hat{y}_1} \sum_{i: x_{id} < s} (\hat{y} - y_i)^2 + \min_{\hat{y}_2} \sum_{i: x_{id} \geq s} (\hat{y} - y_i)^2$$

Information

$$i(x) = -\log_2 \Pr(x)$$

- \log_2 : measured in *bits*
- \log : measured in *nats*

How much information do we gain from an answer?

$$I([2,3]) = \log\left(\frac{2}{2+3}\right) = 1.322 \text{ (bits)}$$

$$I([4,0]) = 0.0 \text{ (bits)}$$

$$I([3,2]) = 0.737 \text{ (bits)}$$

Consult Witten et al, ch 4.

Requirements for information measure:

1. Only “Yes”/Only “No” $\Rightarrow 0$
2. $N_{\text{Yes}} = N_{\text{No}} \Rightarrow$ maximum information
3. Should be applicable to multiclass situations
4. Hierarchical decisions consistent:

Three classes:

- One-step process $[A, B, C]$ must give the same information as
- Two-step process $[A, (B,C)] + [B,C]$.

Example:

class	A	B	C	total
N	2	3	4	9

$$I([A,B,C]) = I([A, (B,C)]) + \frac{7}{9} \cdot I([B,C])$$

Leads to *Entropy*

$$H(x) = - \sum_x \Pr(x) \log_2(\Pr(x))$$

Measure of uncertainty

- 3 classes:
 - hot, cold, all right
- Given the features,
 - 3 × hot
 - 4 × cold
 - 5 × alright

$$N = 12$$

$$H = - \left[\frac{3}{12} \cdot \log_2(3/12) + \frac{4}{12} \cdot \log_2(4/12) + \frac{5}{12} \cdot \log_2(5/12) \right] = 1.555$$

Regularizing Trees

Decision trees are easy to overfit. This manifests in too deep and complex trees that can easily achieve 100% accuracy on training data. There are three common solutions for overfitting.

1. Set maximum depth. Only allow the tree to grow until it reaches a given depth.
2. Stop growing trees if split not significant. This requires a certain threshold value, in terms of MSE or entropy improvement, so when the best split will not improve the overall goodness of the model by at least this much, the tree will stop.
This approach is too shortsighted though, as a mediocre split now may make it possible to achieve a very good split later.
3. *Pruning* is a more far-sighted (albeit more complex) alternative to decisions based on split significance. The idea of *const complexity pruning* is similar to lasso regression, instead of minimizing MSE of the tree, we choose the loss function that penalizes the number of terminal nodes \mathcal{T} :

$$L(\lambda) = \sum_{m=1}^{||\mathcal{T}||} \sum_{i:i \in \mathcal{R}_m} (y_i - \hat{y}_i)^2 + \alpha ||\mathcal{T}|| \quad (7.2.1)$$

Thereafter we can build large trees for different α and cross-validate for the best α .

7.2.2 Ensemble Methods

So far we discussed just trees–decision-making based on a single decision tree. However, it turns out one can get better estimators when combining multiple trees into “forests”. In this way we build *ensemble methods*. The idea of ensemble methods is very simple: instead of relying on a single model, we use a number of models and see what do they all predict. If they agree, this is good news. If they disagree, we just pick the solution that most of the models agree on (we do “majority voting”). In case of regression outcome we just aggregate the predictions of different models.

While ensemble methods are often based on various kinds of trees, one can do ensembles of other types of models as well, e.g. by combining different neural network models, different data sources, or different pre-processing. A major reason why ensemble methods work well with trees is that trees are typically created by the greedy splitting algorithm. As this is suboptimal (it is shortsighted), it leaves a lot of potential information behind. Ensemble methods introduce more variation in how the trees are built, and in this way help to gain some of that information back.

Bagging

Bagging (bootstrap aggregating) is basically averaging predictions from a large number of bootstrapped samples (random samples taken from the training

data). The basic reason why bagging works is as follows: as our data is a random sample for the population, the trees (or other models) built based on it are random too and hence predict random results. But we can make random results to be more stable if we average a large number of them.

If we have B training sets, we can build B different trees, one for each training set, and get B different predictors $\hat{y}^1, \hat{y}^2, \dots, \hat{y}^B$. The final predictor is just the average of these, $\hat{y} = \frac{1}{B} \sum_i \hat{y}^i$ (or the majority category in case of classification). However, as we normally only have a single training set, we can rely on bootstrapping to build more training sets. In case of bagging there is no need to prune the trees, the aggregation functions as regularization.

The number of trees, B , is a hyperparameter that should be tuned, a good value may be around 100.

Random Forests

The downside with bagging is that it tends to rely on the same main features for all trees and hence gets too little variation in the model. Random forests solve this issue by adding random feature selection. Each time the bagging algorithm considers a split, it only considers K' features to use for splitting, instead of the full set of K features. This forces the individual trees to use different features. In practice, random forests are typically more precise method than bagging.

Random forests have two hyperparameters (in addition to the individual tree parameters): number of trees, and number of features K' to include into individual splits. A good choice tends to be $K' = \sqrt{K}$ where K is the original number of features in the data. In case of $K \approx 1000$ features in the original data, each split is done on a subset of $K' \approx 30$ features only. This leaves the majority of features out, and forces the trees to use information that may otherwise not be used.

Boosting

1. build a simple tree
 - may just be a stump
2. predict
3. you get correct and incorrect results
4. compute weights:
 - (a) weights for this tree: better accuracy, higher weights
 - (b) weights for each observation: wrong get more weight by α .
5. repeat many times
6. your prediction is the weighted average of all trees.

Unfortunately, as is the case with other models, ensemble improve prediction accuracy at the expense of interpretability. We cannot present a bagging or random forest model as a decision tree any more.

7.3 Naïve Bayes

Naive Bayes is a classification method that is based on very simplistic (naive) independence assumption, and on the Bayes theorem. The independence assumption is unrealistic in many cases, but tremendously simplifies the computations and makes it able to handle high-dimensional cases. It turns out this tradeoff—computational simplicity againsts unrealistic assumptions—pays off in many types of problems.

7.3.1 Before We Begin: A Single Word-Based Bayesian Classifier

Prerequisites: [Bayes theorem 1.3.2](#)

Imagine you open your mailbox and see the following email:

*I have very urgent and confidential business proposition for you.
On 26th December 2004 an Oil Consultant/Contractor with the
Myanmar National Petroleum Corporation, Mr. A Y Mustafa ...
Mr. A Y Mustafa died from an automobile crash ...
I am looking for a foreigner who will stand in as the next of kin
to Mr. Mustafa ...*

*Yours truly,
Mr. M. Lwin*

Further below it is explained how you can get \$4 million in a few days if you agree with Mr. Lwin's proposal. This would make your day! (Or maybe even your life!)

But before you reply to this “urgent proposal”, maybe you should check if this email is instead spam? After all, the phrase “confidential business proposition” may catch your eyes as somewhat weird. Will these words help us to build a spam filter? Our final task is to build such an spam filter based on the Naive Bayes model, but before we get there, let's build a Bayesian spam filter based on a single phrase only.

In the categorization applications we discuss here we can never be 100% certain about the correct category, so instead of directly modeling a spam/non-spam decision, we model the probability that an email that contains the phrase “confidential business proposition” (*CBP*) is spam. Formally, we are interested in the conditional probability

$$\Pr(S|CBP) \tag{7.3.1}$$

where S is the spam status ($S = 1$ for spam and $S = 0$ for no spam), and CBP is the CBP status ($CBP = 1$ if the email contains the phrase and $CBP = 0$ if it does not). As both S and CBP can have two values, we have for four probabilities in total:

$$\begin{array}{ll} \Pr(S = 0|CBP = 0) & \Pr(S = 0|CBP = 1) \\ \Pr(S = 1|CBP = 0) & \Pr(S = 1|CBP = 1). \end{array} \tag{7.3.2}$$

The first row represents the probabilities that the email is not spam ($S = 0$) while not containing the phrase ($CBP = 0$) and while containing the phrase ($CBP = 1$). In the second row we have the probability of spam ($S = 1$). Exactly as in the upper row, the first probability refers to the case where the email is spam while not containing the phrase, while the second one represents the case that an email that contains the phrase is spam. Obviously, as every email is either spam or non-spam, the corresponding probabilities must sum to unity: for emails without the phrase $\Pr(S = 0|CBP = 0) + \Pr(S = 1|CBP = 0) = 1$ and the same for emails with the phrase, $\Pr(S = 0|CBP = 1) + \Pr(S = 1|CBP = 1) = 1$. For simplicity, we often use the shorter notation (7.3.1) to represent all four probabilities.

Next, we can use Bayes theorem to express the probabilities in (7.3.1):

$$\Pr(S|CBP) = \frac{\Pr(CBP|S) \cdot \Pr(S)}{\Pr(CBP)}. \quad (7.3.3)$$

As above, CBP represent the phrase status (email does contain or does not contain CBP) and S represents the spam status (email is spam or not spam), so (7.3.3) actually represents four different probabilities, exactly as does (7.3.1) above. To fix the ideas, let's focus on $\Pr(S = 1|CBP = 1)$, the probability that an email containing “confidential business proposal” is spam. This version of (7.3.3) is

$$\Pr(S = 1|CBP = 1) = \frac{\Pr(CBP = 1|S = 1) \cdot \Pr(S = 1)}{\Pr(CBP = 1)}. \quad (7.3.4)$$

Let us now go over of all the probabilities on the right-hand-side of (7.3.4).

- $\Pr(S = 1)$ is the prior for spam, the unconditional probability of the email being spam. It is just the percentage of spam emails in our data. Note that while computing the spam percentage is simple, it requires a training dataset, a manually labeled set of emails.
- $\Pr(CBP = 1|S = 1)$ is the percentage of spam emails that contain the phrase. This is the main source of information that includes both spam and content data and allows us to improve our predictions. It is straightforward to calculate this probability by simply selecting all the spam emails and computing the percentage that contain “confidential business proposal”.
- Finally, the normalizer $\Pr(CBP)$ is just the unconditional probability that emails contain the phrase, be they spam or not. This is the simplest probability to compute, we don't even need labeled training data.

So all these probabilities can be computed easily if we have training data, a dataset of suitable size where the emails are already labeled into spam and no-spam ones. The rest is essentially just tabulating, and using the Bayes theorem as the final step.

Example 7.3.1: Bayesian spam filter based on a single word

Assume you have labeled training data of 1000 emails, 400 of these are spam and 600 are not spam. We focus on a single word, “viagra” in these emails. Denote the “viagra” status by $V = \mathbb{1}(\text{email contains “viagra”})$. The table below shows the counts of all types of emails:

	$V = 0$	$V = 1$	Total
$S = 0$	500	100	600
$S = 1$	150	250	400
Total	650	350	1000

Based on this table, let us compute $\Pr(S = 1|V = 1)$, the probability that an email is spam, given it contains “viagra”. When using (7.3.4), we first have to find the following probabilities:

- $\Pr(V = 1|S = 1)$, probability of “viagra” in spam emails. From the table we can see that it is $250/400 = 5/8 = 0.625$.
- The prior, $\Pr(S = 1)$, the proportion of spam emails. It is $400/1000 = 2/5 = 0.4$.
- The normalizer, $\Pr(V = 1)$, the probability to see “viagra” in emails. It is $350/1000 = 7/20 = 0.35$.

Based on these numbers we can easily compute

$$\begin{aligned}\Pr(S = 1|V = 1) &= \frac{\Pr(V = 1|S = 1) \cdot \Pr(S = 1)}{\Pr(V = 1)} = \\ &= \frac{\frac{5}{8} \cdot \frac{2}{5}}{\frac{7}{20}} = \frac{5}{7} \approx 0.714.\end{aligned}\quad (7.3.5)$$

The Bayesian update based on a single word made the final probability 0.714, close to 2-fold increase over the prior $\Pr(S) = 0.4$. Such substantial improvement was only possible because in this example data the word “viagra” is very common in spam emails. If a word is rare, it can only identify a small number of spam emails. If it is very spam-specific, it gives us large precision but the recall will remain low as most of the spam emails are left unidentified.

Exercise 7.3.1: Probability of spam given no “viagra”

Use the data as in the Example 7.3.1. Compute the probability $\Pr(S = 1|V = 0)$, the probability that the email is not spam if it does not contain “viagra”. How much larger is the posterior compared to the prior?

7.3.2 Naive Bayes Classifier

Prerequisites: [Bayes theorem 1.3.2](#), [Independence 1.3.2](#)

Obviously we should not base our spam filter just on a single word. It would not catch much spam, and it may remove good emails that for the same reason contain a similar expression. A much better approach would be to look at many words, potentially at all the words we know about. But let's start with adding another phrase, *lottery winner* (*LW*), to our spam filter. So now our model contains two indicators, *CBP* and *LW*, both of which can be 0 or 1, and the probability for spam can now be expressed as

$$\Pr(S = 1 | CBP, LW) = \frac{\Pr(CBP, LW | S = 1) \cdot \Pr(S = 1)}{\Pr(CBP, LW)}. \quad (7.3.6)$$

Here we have simplified the notation a little bit:

- The prior, $\Pr(S)$, is unchanged. This is just the unconditional probability that an email is spam.
- $\Pr(CBP, LW | S = 1)$ can be any of these four probabilities, depending on which phrases the email contains:
 1. $\Pr(CBP = 0, LW = 0 | S = 1)$: probability that a spam email does not contain either “confidential business proposition” nor “lottery winner”.
 2. $\Pr(CBP = 0, LW = 1 | S = 1)$: probability that a spam email does not contain “confidential business proposition” but contains “lottery winner”.
 3. $\Pr(CBP = 1, LW = 0 | S = 1)$: probability that a spam email contains “confidential business proposition” but does not contain “lottery winner”.
 4. $\Pr(CBP = 1, LW = 1 | S = 1)$: probability that a spam email contains both “confidential business proposition” and “lottery winner”.
- Similar reasoning also applies to the normalizer $\Pr(CBP, LW)$. It is a list of four probabilities, corresponding to the case that any combination of these two phrases occur in the email.
- And finally, the question of interest itself, $\Pr(S = 1 | CBP, LW)$, also contains four possibilities: it is the probability that the email is spam, given it contains or does not contain any combination of “confidential business proposition” and “lottery winner”.

So in order to use the Bayesian approach with two phrases, we need 9 probabilities in all: one for the prior, four for the conditional probabilities, and four for the normalizers. All these should be computed from the labeled training

data. This is straightforward to do, given we have a labeled training data set of suitable size.

But unfortunately the story does not stop here. If we use three phrases instead of two, we have 8 combinations for both the conditional probability and for the normalizer: one set of four when the new phrase is present and another set of four when it is absent. In case of four phrases we have 16 and so on. It is easy to see that when we analyze K phrases or words, we have 2^K different combinations. In case of a realistic text, K may easily exceed 10,000. So a complete Bayesian approach will involve $\sim 2^{10,000}$ different combinations. This is infeasible to do.¹ We run into the curse of dimensionality.

But it is easy to avoid the curse of dimensionality by introducing additional assumptions. The most popular one, and the one that is the basis of Naive Bayes method, assumes that the presence of words in the data is independent given the email is spam or no spam. And remember—in case of independent events, we can just multiply the corresponding probabilities. So in case of our two phrases we can write the conditional probability as

$$\Pr(CBP, LW|S) = \Pr(CBP|S) \cdot \Pr(LW|S). \quad (7.3.7)$$

As explained above, this formula represents eight different probabilities, four different combinations of presence of these two phrases in spam and non-spam emails. To fix the ideas, let's look at $\Pr(CBP = 1, LW = 1|S = 1)$, i.e. probability that a spam emails contains both these phrases. The assumption means that if e.g. $\Pr(CBP = 1|S = 1) = 0.1$ and $\Pr(LW = 1|S = 1) = 0.1$, then $\Pr(CBP = 1, LW = 1|S = 1) = 0.01$. If 10% of spam emails contain CBP and 10% of spam emails contain LW , then 1% of spam emails contain both phrases. Our assumption tells that this is true for spam emails, and also for non-spam emails, but not necessarily for all emails. This is what the conditioning on spam status S does in (7.3.7).

Let's first look at the good news. Where we formerly had to calculate four different probabilities, one for each combination of CBP and LW , now we only need two: one for CBP and one for LW . And even better, when we add another feature, we only need one additional probability. So in case of 10,000-word vocabulary, we only need 10,000 different probabilities. This is fast and trivially fits into the computer memory. So we have circumvented the curse of dimensionality in case of the conditional probability (we'll discuss what to do with the normalizer below).

But what does this assumption mean and why is it called “naïve”? Independence of random variables means that one RV does not contain information about the other one. If we learn about the first phrase, “confidential business proposition”, this does not tell us anything about the presence of the other phrase, “lottery winner” in case we are just looking at the spam emails. So the

¹Sometimes it is not appreciated just how incredibly big are these numbers. For comparison, the age of Universe is approximately $10^{18} \approx 2^{54}$ seconds, and the visible universe contains $\sim 10^{90} \approx 2^{300}$ elementary particles. $2^{10,000}$ is just way way beyond of what fits into our universe, so there is no way we can ever collect and analyze this much data. (As before, we do not talk about quantum computing).

method ignores the fact that the words may be correlated, and not just correlated but the correlation may carry different meaning than the individual words. For instance, when tokenizing “New York” into individual words, we treat the resulting “new” and “york” as separate independent identities. The model does not understand that “New York” carries a distinct meaning, very different from what these two single words carry. These are the bad news, and this is why the approach is called “naive”. But in this case the good news outweigh the bad ones, as we now have a model that actually can be computed.

Let us now build a full two-class Naive Bayes model. Assume we have a vocabulary of size K of words W_1, W_2, \dots, W_K where W_i denotes the presence (if $W_i = 1$) or absence (if $W_i = 0$) of word i in the document. We are looking for a category S where $S \in \{0,1\}$ denotes whether the document is spam or not. We can write the independence (naive) assumption as

$$W_i \perp\!\!\!\perp W_j | S. \quad (7.3.8)$$

This assumption means we describe the words as picked randomly from different distributions, one for $S = 0$ and one for $S = 1$. As we discussed above, the assumption is unrealistic as it ignores the relationship between words but it resolves the curse of dimensionality problem. In fact, the Naive Bayes method scales surprisingly well.

Remember that by definition of [independent events](#) their joint probability is equal to the product of individual probabilities. For two words we have $\Pr(W_1, W_2) = \Pr(W_1) \cdot \Pr(W_2)$. The same applies for more than two probabilities, $\Pr(W_1, W_2, \dots, W_K) = \prod_{j=1}^K \Pr(W_j)$ (see [1.3.2](#)). It also applies for conditionally independent probabilities as conditional probabilities are just word probabilities in spam or non-spam category, so we can write

$$\Pr(W_1, W_2, \dots, W_K | S) = \prod_{j=1}^K \Pr(W_j | S). \quad (7.3.9)$$

This is the conditional probability we need in the Bayesian approach, and above we discussed that when using the independence assumption, we only have to find $2K$ probabilities from the training data, $\Pr(W_j = 1 | S)$ and $\Pr(W_j = 0 | S)$ for $j = 1, \dots, K$, in order to compute the joint probability.

Let us now solve the spam email problem by using the independence assumption ([7.3.9](#)). We want to estimate $\Pr(S = 1 | W_1, W_2, \dots, W_K)$, the probability of the email being spam, given the words it contains or does not contain. We start

with expressing this probability through Bayes theorem:

$$\begin{aligned} \Pr(S = 1|W_1, W_2, \dots, W_K) &= \frac{\Pr(W_1, W_2, \dots, W_K|S = 1) \cdot \Pr(S = 1)}{\Pr(W_1, W_2, \dots, W_K)} = \\ &= (\text{here we use the independence assumption (7.3.9)}) = \\ &= \frac{\Pr(W_1|S = 1) \cdot \Pr(W_2|S = 1) \cdots \Pr(W_K|S = 1) \cdot \Pr(S = 1)}{\Pr(W_1, W_2, \dots, W_K)} = \\ &= \frac{\Pr(S = 1) \cdot \prod_{j=1}^K \Pr(W_j|S = 1)}{\Pr(W_1, W_2, \dots, W_K)}. \quad (7.3.10) \end{aligned}$$

Now the numerator is factorized into a product of $K + 1$ factors in the form of $\Pr(W_k|S)$ and $\Pr(S = 1)$. These are just K conditional probabilities in the form *probability the word k exists/does not exist in spam emails*. So we need just K numbers, as probability of absence of the word is just $1 - \text{probability of presence of it}$. These conditional probabilities can easily be calculated based on word counts in spam/non-spam documents.

But this only solves one part of our problem: the normalizer $\Pr(W_1, W_2, \dots, W_K)$ is still there, and it is still intractable.² But fortunately we can eliminate the normalizer in a simple way. Let's also compute the probability that the email is not spam. Using the same approach as in (7.3.10), we get

$$\Pr(S = 0|W_1, W_2, \dots, W_K) = \frac{\Pr(S = 0) \cdot \prod_{j=1}^K \Pr(W_j|S = 0)}{\Pr(W_1, W_2, \dots, W_K)}. \quad (7.3.11)$$

As above, we have a numerator that contains K word frequencies in non-spam emails, the prior $\Pr(S = 0)$, and an intractable normalizer in the denominator. But note that these two expressions, (7.3.10) and (7.3.11), contain the exact same normalizer. So we can just leave the normalizer out and still compare these probabilities! However, if we leave out the normalizer, they are not probabilities any more. These are now called *likelihoods* instead and they are not valid probabilities as we do not normalize these. In particular, they do not sum to unity, and they may exceed 1. So we cannot interpret the results as probabilities, but as we left out the same denominator, we can still say that the largest likelihood corresponds to the largest probability. If likelihood for spam is larger, we call this email spam, if the likelihood for non-spam is larger we say it is non-spam.

So we can now follow these steps to predict whether the email is spam. All of it can be done on training data.

1. Compute the priors $\Pr(S = 0)$ and $\Pr(S = 1)$.

²Note that we assume independence of *conditional* distribution $W_i \perp\!\!\!\perp W_j|S$, not independence of unconditional distribution which would be written $W_i \perp\!\!\!\perp W_j$. The former means that words are independent in each class, spam and non-spam; the latter means that the words are independent when combining both classes. The latter is not a result of the former.

2. For each word W in the vocabulary, compute the conditional probabilities $\Pr(W = 1|S = 1)$ and $\Pr(W = 1|S = 0)$.
3. Compute the numerators (likelihoods) of the naive Bayes formula (7.3.11)

$$\begin{aligned} \mathcal{L}(S = 1|W_1, W_2, \dots, W_K) &= \Pr(S = 1) \cdot \prod_{j=1}^K \Pr(W_j|S = 1) \\ &\quad \text{and} \\ \mathcal{L}(S = 0|W_1, W_2, \dots, W_K) &= \Pr(S = 0) \cdot \prod_{j=1}^K \Pr(W_j|S = 0) \end{aligned} \tag{7.3.12}$$

where $\mathcal{L}(\cdot)$ denotes the corresponding likelihood.

4. Which likelihood is larger, $\mathcal{L}(S = 1|\mathbf{W})$ or $\mathcal{L}(S = 0|\mathbf{W})$? (\mathbf{W} denotes a vector of all vocabulary words, $\mathbf{W} = (W_1, W_2, \dots, W_K)^\top$.) This is the prediction. If likelihood for spam is larger we have a spam email, if non-spam likelihood is larger, it is not a spam.

In this example we only consider the presence of words $\Pr(W_j = 1|S = 1)$, but one may also add information about absence of words $\Pr(W_j = 0|S = 1) = 1 - \Pr(W_j = 1|S = 1)$. Obviously, these probabilities differ for spam and non-spam cases.

In practice, it is better to work with logarithms of likelihoods (*log-likelihoods*) instead of the likelihoods as the latter typically contain too small numbers. The corresponding log-likelihoods are

$$\begin{aligned} \ell(S = 1|\mathbf{W}) &\equiv \log \mathcal{L}(S = 1|\mathbf{W}) = \log \Pr(S = 1) + \sum_{j=1}^K \log \Pr(W_j|S = 1) \\ \ell(S = 0|\mathbf{W}) &\equiv \log \mathcal{L}(S = 0|\mathbf{W}) = \log \Pr(S = 0) + \sum_{j=1}^K \log \Pr(W_j|S = 0). \end{aligned} \tag{7.3.13}$$

Normally the prediction will be the category that corresponds to a larger probability, and this also means category that has the largest log-likelihood. So we can write both equations in (7.3.13) as

$$\hat{S} = \arg \max_S \ell(S) = \log \Pr(S) + \sum_{j=1}^K \log \Pr(W_j|S) \tag{7.3.14}$$

where \hat{S} means the predicted category.

Let us repeat the algorithm above for log-likelihood. To compute Naive Bayes, we need:

1. Compute the log prior probabilities $\log \Pr(S = 0)$ and $\log \Pr(S = 1)$.
This amounts to two probabilities.

2. For each word W in the vocabulary, compute the log conditional probabilities $\log \Pr(W = 1|S = 1)$ and $\log \Pr(W = 1|S = 0)$.

All together we have two probabilities for each word, or $2 \cdot K$ probabilities in total.

3. Compute the Naive Bayes log-likelihoods (7.3.13)

$$\ell(S = 1|W_1, W_2, \dots, W_K) = \log \Pr(S = 1) + \sum_{j=1}^K \log \Pr(W_j|S = 1)$$

and

$$\ell(S = 0|W_1, W_2, \dots, W_K) = \log \Pr(S = 0) + \sum_{j=1}^K \log \Pr(W_j|S = 0) \quad (7.3.15)$$

where $\ell(\cdot)$ denotes the corresponding log-likelihood.

4. Which log-likelihood is larger, $\ell(S = 1|\mathbf{W})$ or $\ell(S = 0|\mathbf{W})$? This is the prediction.

In case of typical texts where the vocabulary size is $\sim 10,000$ we have to compute tens of thousand of log-probabilities. This is an easy task for modern computers given we have suitable training data.

Example 7.3.2: Email classification

Assume we have categorized the following emails as spam/no-spam:

1. viagra is good in life: spam
2. life is good: no spam
3. viagra in life: no spam

Hence the training data tells us that the unconditional probability of spam $\Pr(S = 1) = 1/3$ and the unconditional probability of non-spam $\Pr(S = 0) = 2/3$.

These three emails contain vocabulary (in alphabetical order) “good”, “in”, “is”, “life”, “viagra”; and the corresponding [document-term-matrix](#) (see Section 8.1.3) is in table below:

	good	.in	is	life	viagra
x_1	1.00	1.00	1.00	1.00	1.00
x_2	1.00	0.00	1.00	1.00	0.00
x_3	0.00	1.00	0.00	1.00	1.00
N_W	2.00	2.00	2.00	3.00	2.00
$\Pr(W = 1 S = 1)$	1.00	1.00	1.00	1.00	1.00
$\Pr(W = 1 S = 0)$	0.50	0.50	0.50	1.00	0.50

Table 7.2: DTM of the three example emails (rows \mathbf{x}_1 , \mathbf{x}_2 and \mathbf{x}_3) the corresponding word counts (N_W), and conditional probabilities of word in spam ($\Pr(W = 1|S = 1)$) and no-spam ($\Pr(W = 1|S = 0)$) emails.

Now we receive a new email: *no viagra no life*. We want to categorize it based on our training data above. We convert it into BOW representation using the existing vocabulary:

	good	in	is	life	viagra
\mathbf{x}_4	0	0	0	1	1

Table 7.3: The new email as BOW. Note that as the word “no” is missing in the training vocabulary, we ignore it here as we have no way of telling what would the corresponding probabilities be.

These tables together are sufficient to compute the likelihoods. Although we normally prefer log-likelihood, let’s first do the likelihoods:

$$\begin{aligned}
 \mathcal{L}(S = 1|\mathbf{x}_4) &= \Pr(S = 1) \times \\
 &\quad \times \Pr(\text{good} = 0|S = 1) \times \Pr(\text{in} = 0|S = 1) \times \\
 &\quad \times \Pr(\text{is} = 0|S = 1) \times \Pr(\text{life} = 1|S = 1) \times \\
 &\quad \times \Pr(\text{viagra} = 1|S = 1) = \\
 &= 0.333 \times 0 \times 0 \times 0 \times 1 \times 1 = 0 \quad (7.3.16)
 \end{aligned}$$

The product contains three factors of 0—our training data has no spam examples that do not contain words “good”, “in” and “is”, and hence the probability not to contain those words given spam is 0. Next we compute the likelihood for no spam:

$$\begin{aligned}
 \mathcal{L}(S = 0|\mathbf{x}_4) &= \Pr(S = 0) \times \\
 &\quad \times \Pr(\text{good} = 0|S = 0) \times \Pr(\text{in} = 0|S = 0) \times \\
 &\quad \times \Pr(\text{is} = 0|S = 0) \times \Pr(\text{life} = 1|S = 0) \times \\
 &\quad \times \Pr(\text{viagra} = 1|S = 0) = \\
 &= 0.667 \times 0.5 \times 0.5 \times 0.5 \times 1 \times 0.5 = 0.042. \quad (7.3.17)
 \end{aligned}$$

Hence this email falls into category “no spam” as 0.042 is larger than 0.

We can repeat the exercise with log-likelihood, this is the method we normally use. First we compute log likelihood for spam:

$$\begin{aligned}
 \ell(S = 1|\mathbf{x}_4) &= \log \Pr(S = 1) + \\
 &\quad + \log \Pr(\text{good} = 0|S = 1) + \log \Pr(\text{in} = 0|S = 1) + \\
 &\quad + \log \Pr(\text{is} = 0|S = 1) + \log \Pr(\text{life} = 1|S = 1) + \\
 &\quad + \log \Pr(\text{viagra} = 1|S = 1) = \\
 &= -1.099 - \infty - \infty - \infty + 0 + 0 = -\infty \quad (7.3.18)
 \end{aligned}$$

The product contains three factors of 0—our training data has no spam examples that do not contain words “good”, “in” and “is”, and hence the probability not to contain those words given spam is 0. Next we compute the likelihood for no spam:

$$\begin{aligned}\ell(S = 0 | \mathbf{x}_4) &= \log \Pr(S = 0) + \\ &\quad + \log \Pr(\text{good} = 0 | S = 0) + \log \Pr(\text{in} = 0 | S = 0) + \\ &\quad + \log \Pr(\text{is} = 0 | S = 0) + \log \Pr(\text{life} = 1 | S = 0) + \\ &\quad + \log \Pr(\text{viagra} = 1 | S = 0) = \\ &= -0.405 - 0.693 - 0.693 - 0.693 + 0 - 0.693 = -3.178.\end{aligned}\quad (7.3.19)$$

The conclusion is “no spam” again as $-3.178 > -\infty$. Obviously, the resulting figures are just logs of the likelihoods above.

Exercise 7.3.2: Categorize using Naive Bayes

Use the training data in Example 7.3.2, and categorize the sentence “life is life”.

It is very easy to generalize Naive Bayes to more than two classes—you just need to compute the log-likelihoods for each class, and as pick the class with the largest log-likelihood as the prediction.

Smoothing

In the examples above we did not talk much about how to compute the probabilities, such as $\Pr(W = 1 | S = 1)$. Intuitively, one may want just to use the proportion of documents where the word is present (as we did in the examples above):

$$\Pr(W = 1 | S = 1) = \frac{N_{W=1|S=1}}{N_{S=1}} \quad (7.3.20)$$

where $N_{W=1|S=1}$ is the count of spam-emails where the word is present, and $N_{S=1}$ is the total number of spam emails. This intuitive approach is justified if the counts are large. But if we observe just a few cases of the word, these probabilities may be far from the truth. A common manifestation of this problem is the case where we observe only a single instance of the word. Obviously, this means it only belongs to a single class, say spam. So while $\Pr(W = 1 | S = 1) > 0$, the probability $\Pr(W = 1 | S = 0) = 0$. Now whenever you encounter a document that contains this word, we have $\mathcal{L}(S = 1) > 0$ while $\mathcal{L}(S = 0) = 0$ (see (7.3.12)). Hence our prediction will be $S = 1$ for sure. The other words in the text play no role because zero remains zero even when multiplied by a positive probability. But intuitively, how can we base our prediction on a single observation of a single word, while ignoring everything else in the message, and claim we are 100% certain in our conclusion? Note that replacing likelihood

by log-likelihood only shifts the problem from zero-likelihood to minus-infinity-likelihood but does not offer any solution. For instance, imagine we are doing a spam filter and we have seen the word “viagra” just once, in a spam email. Hence every new email that contains this word will be categorized as spam because $\Pr(viagra = 1|S = 0) = 0$. No buts, no ifs.

But typically it is not just a single word that occurs in our corpus only once. Imagine the word “conference” also appears only once, in a valid email. So every message containing “conference” will unambiguously be categorized as valid. But now what should we do with an email that contains both “viagra” and “conference”? Based on our reasoning above, it has zero probability to belong to either category.

Obviously, it is problematic to rely on a single rare value for categorization. Rare words are, by definition, rare, and hence may or may not occur just by chance. Additionally, the NB classifier fails completely if the text to be categorized contains two rare words, each pointing to a different class. In that case all likelihoods will be zero and the prediction will either be arbitrary, or undefined. Note that more data is not a solution to the rare word problem: with more data, one also includes increasingly rare words in the sample, so we always have words that are only represented 1-2 times in the corpus.

The problem arises because we are drawing too strong conclusions from too little data. Clearly, a single “viagra” and a single “conference” is not enough to claim we have 100% certainty to categorize the email. As this certainty originates from the probability calculation (7.3.20), that approach must be incomplete. Intuitively, it is easy to see what is wrong with that formula—it does not take into account the sample size. If we find 0 valid ones out of total $N = 1$ emails that contain “viagra” we have probability 0. If we find 0 valid emails out of $N = 1000$ we still have probability 0. But clearly in the latter case we are much more confident in the result.

A popular solution to this problem is called smoothing. Smoothing is equivalent to Bayesian estimation³ of the probabilities. Instead of taking the strictly frequentist approach (7.3.20),⁴ we should take a Bayesian approach where we include a prior for the probability of interest. A Bayesian prior (beta-prior) is equivalent to adding a small positive number to the counts.

Take the spam example. Let the prior for $\Pr(W = 1|S = 1) = 0.5$, i.e. we have no prior reason to believe that the word W is either represented or not represented in either of the classes. This can be done by picking a parameter $\alpha > 0$ and instead of computing the empirical probabilities as

$$\Pr(W = 1|S = 1) = \frac{N_{W=1|S=1}}{N_{S=1}} \quad (7.3.21)$$

³“Bayesian” here refers to Bayesian statistics, not to Naive Bayes estimator. Naive Bayes is based on Bayesian theorem but Bayesian statistics includes much more than this method.

⁴This claim is a bit misleading. While frequentists may be happy with (7.3.20), they are not happy with how we use this probability afterwards. In particular, ignoring uncertainty of computed values is not a correct way of doing frequentist statistics.

we compute these as

$$\Pr(W = 1|S = 1) = \frac{N_{W=1|S=1} + \alpha}{N_{S=1} + 2\alpha}. \quad (7.3.22)$$

Now in case we have not seen the word in spam, the corresponding probability will not be 0, but $\alpha/(N_{S=1}+2\alpha)$ instead. α describes our confidence in the prior, small α mean little confidence and large α means a lot of confidence. Obviously, the more observations from we have (the larger N), the less α matters and the result is close to the pure frequentist probability 0. Data overrides the prior. But it is never exactly 0 and hence does not corrupt the estimator. Note that the term 2α in denominator takes into account that we have two classes. If we haven't seen any example from spam, i.e. $N_{S=1} = 0$, then $\Pr(W = 1|S = 1) = \alpha/(2\alpha) = 1/2$, i.e. the prior. In practical applications, α is a hyperparameter and should be tuned using cross-validation or another similar approach. A good value for α tends to be small, something like 0.1. As the denominator in (7.3.22) is typically large (in thousands), the term 2α in denominator plays a little role. The term that matters is α in the numerator.

Finally, one can take different priors for different words and classes, and also have a prior on $\Pr(C = c)$. See [Murphy \(2012, p. 87\)](#) for more information.

Example 7.3.3: Email classification with smoothing

Let us revisit the example [7.3.2](#) and add smoothing to that model. Let's pick $\alpha = 0.2$. We have the following data

	good	in	is	life	viagra
x_1	1.00	1.00	1.00	1.00	1.00
x_2	1.00	0.00	1.00	1.00	0.00
x_3	0.00	1.00	0.00	1.00	1.00

Table 7.4: DTM of the three example emails from example [7.3.2](#). The first row (the first email) is spam, the following two are not spam.

However, now we have to adjust the way we compute the probability by using [7.3.22](#). For *good* we get

$$\begin{aligned} \Pr(\text{good} = 0|S = 0) &= \frac{1 + \alpha}{2 + 2\alpha} = 1.2/2.4 = 0.5 \\ \Pr(\text{good} = 1|S = 0) &= \frac{1 + \alpha}{2 + 2\alpha} = 1.2/2.4 = 0.5 \\ \Pr(\text{good} = 0|S = 1) &= \frac{0 + \alpha}{1 + 2\alpha} = 0.2/1.4 \approx 0.143 \\ \Pr(\text{good} = 1|S = 1) &= \frac{1 + \alpha}{1 + 2\alpha} = 1.2/1.4 \approx 0.857. \end{aligned} \quad (7.3.23)$$

Remember that in the case of no smoothing in example [7.3.2](#) the probabilities were $\Pr(\text{good} = 1|S = 0) = 0.5$ and $\Pr(\text{good} = 1|S = 1) = 0.5$. Now

the former, 0.5, remains unchanged while the latter is moved away from the original extreme value 0. It is easy to see that this type of smoothing moves the calculated probabilities toward 0.5, toward the uninformative “middle ground”.^a The fewer observations we have, the larger is the shift. This process makes intuitively sense: the fewer observations we have the less certain we are in the calculated probabilities, and the more we are willing to admit that we don’t know well what do these words mean. “We don’t know” is a way to say that we should pick probabilities close to 0.5.

It is also easy to see why did we write 2α in the denominator. One α applies to the presence, and another to the absence of the word. This ensures that the probability of presence and absence of the word sum to unity.

The next table shows all the probabilities:

	good	in	is	life	viagra
$\Pr(W = 0 S = 0)$	0.50	0.50	0.50	0.08	0.50
$\Pr(W = 1 S = 0)$	0.50	0.50	0.50	0.92	0.50
$\Pr(W = 0 S = 1)$	0.14	0.14	0.14	0.14	0.14
$\Pr(W = 1 S = 1)$	0.86	0.86	0.86	0.86	0.86

Table 7.5: Smoothed probabilities for the DTM above for $\alpha = 0.2$.

Afterwards we predict in exactly the same way as earlier, just using the smoothed probabilities instead of original ones.

^aYou can see this by taking a very large α value, say 100.

7.4 Neural Networks

Neural networks are one of the fastest developing classes of machine learning models. These have achieved tremendous progress in a variety of fields, in particular image and natural language processing. Modern neural networks can recognize images, faces, fingerprints, can convert pictures of text to text, and understand spoken language.

While exciting methods, these are also some of the most demanding ones with the number of trainable parameters in millions or even billions.⁵ This makes training and using such models slow and computation-intensive. One also needs a large amount of training data which may be challenging to obtain. Another major downside of neural networks is lack of interpretability. As ML models are increasingly employed as decision-making aides, there is an increasing interest for model transparency. Complex models as neural networks are, are essentially black boxes even for data scientists who train them.

7.4.1 Feed-Forward Networks

Biological Origins

Humans have been puzzled about how brain works since ancient times. The first step in solving the puzzle was understanding the basic working mechanisms of *neurons*, the brain cells. In a very simplistic view, each neuron receives signals from other neurons through its dendrites, and outputs a signal to the dendrites of other neurons through its axon. Normally its axon does not send any signal, but if large enough number of its input dendrites become active, then the neuron “fires”, i.e. sends a signal through its axon to another neuron.

Next, we model this simple neuron using mathematical tools and create an artificial neuron. These artificial neurons are the fundamental building blocks of all computational neural networks. The simplest networks, made of such artificial neurons, are called *perceptrons*. We start by introducing *and-perceptron* and *or-perceptron*, both are essentially made of a single neuron. Thereafter we move to more complex perceptrons with *hidden layers*.

Perceptron

Perceptrons (feed-forward neural network) is a simple neural network that can be built using such artificial neurons. Perceptrons can perform various prediction tasks but cannot compete with more specialized networks, such as convolutional networks for image processing or LSTM blocks for natural language processing. However, the more specialized models almost always contain feed-forward layers.

AND and OR Perceptron Imagine we have a neuron that gets two inputs (e.g. from other neurons) and outputs a signal only if both of these inputs are “high”.

⁵The largest language model as of 2021, *Switch-C*, contains $1.57 \cdot 10^{12}$ parameters.

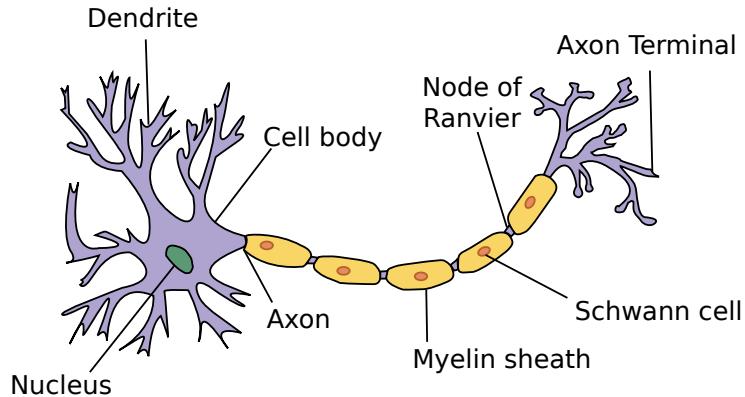


Figure 7.3: Schematic look of neuron. The cell has a long “tail”, axon, that is connected to the dendrites of other neurons. If a neuron receives enough electrical signals on its dendrites, it will “fire”, i.e. send a signal out of its axon.

By User:Dhp1080 - "Anatomy and Physiology" by the US National Cancer Institute's Surveillance, Epidemiology and End Results (SEER) Program, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=1474927>

We can model the neuron activity by the following way:

1. Label the inputs x_1 and x_2 . These are the signals it receives through “dendrites”, and these can be “low”, 0, or “high”, 1.
2. Let the neuron add both signals x_1 and x_2 . We are a little more general here and compute the weighted sum $z = w_1x_1 + w_2x_2$ where w_1 and w_2 are weights, so the neuron can assign different importance to different signals.
3. Let the neuron fire (output 1) if $z > \bar{z}$ where \bar{z} is a threshold, otherwise it will output 0. So we can write output $y = \mathbb{1}(z > \bar{z})$.

Such a process activates the neuron if the input values are large enough (given w_1 and w_2 are positive), and it outputs 1 when active. We can write it formally as

$$y = \mathbb{1}(w_1x_1 + w_2x_2 > \bar{z}). \quad (7.4.1)$$

Figure 7.4 depicts this perceptron as a simple neural network. The figure indicates some of the most important components of neural networks. The inputs are entering through *input layer*. The input nodes do not really do any operations, these are simply the feature vectors that are fed into the model. In this example we only have two inputs, but in complex networks, such as image processing, the input layer may contain millions of nodes corresponding to the input pixels.

In this example, input layer feeds its data directly to the output layer. Output layer contains a single node (neuron) that does two operations: first the linear transformation $z = w_1x_1 + w_2x_2$ and thereafter activation $y = \mathbb{1}(z > \bar{z})$.

w_1 and w_2 are typically called *weights* and \bar{z} is called *bias*. Both of these operations are done almost universally by all nodes (except input nodes) in all networks. However, in practice it is rare to use indicator function (step function) for activation. The most popular function in practice is ReLU, [see below](#).

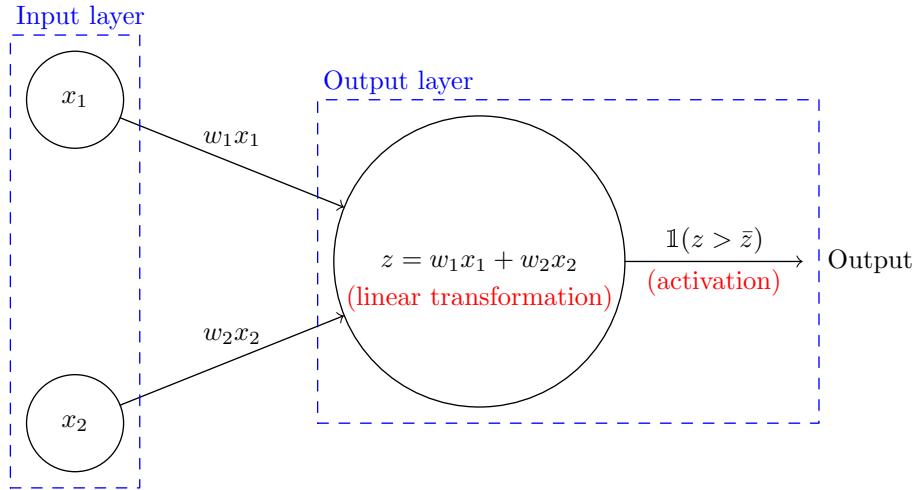


Figure 7.4: And Perceptron. The inputs x_1 and x_2 form the input layer, the single computing node forms the output layer. While the input layer only provides output to the node, the output node itself performs two operations: linear transformation $z = w_1x_1 + w_2x_2$, and activation, $y = \mathbb{1}(z > \bar{z})$.

This simple perceptron can perform logical *AND* and *OR* operations when choosing the weights and the bias accordingly. *AND* and *OR* operations are binary logical operations that take two inputs x_1 and x_2 , both of which may only have two values, 1 or 0, and always return either 1 or 0, depending on the inputs. See Table 7.6. Logical *AND* is 1 only if both inputs are 1, logical *OR* is 1 if any of the inputs is 1, and logical *XOR* (exclusive or) is 1 if exactly one of the inputs is one. This table can be understood as a dataset where we observe two features (inputs) x_1 and x_2 , and predict any of the outputs *AND*, *OR* or *XOR* (or maybe all three outputs at the same time). Unlike traditional datasets, this small table is comprehensive data, i.e. there are no more possible combinations of inputs, and the outputs are always exactly the same given inputs. No stochastic noise is possible here.

In order to perform the listed operations, we need to find suitable parameters, a triple of numbers (w_1, w_2, \bar{z}) . A possible solution for *AND*-perceptron is to choose $w_1 = w_2 = 1$ and $\bar{z} = 1.5$. For instance, if $x_1 = 1$ and $x_2 = 0$, then $z = 1 \cdot x_1 + 1 \cdot x_2 = 1 < \bar{z}$ and hence $y = 0$. However, if both $x_1 = x_2 = 1$, then $z = 2 > \bar{z}$ and so $y = 1$. Obviously, there are infinitely many possible solutions, for example, when keeping $w_1 = w_2 = 1$, every $\bar{z} \in (1,2)$ will produce correct results.

Table 7.6: AND, OR and XOR operations

inputs		outputs		
x_1	x_2	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Exercise 7.4.1: OR-perceptron

Use the same perceptron model as in Figure 7.4. Construct *OR*-perceptron: find a suitable set of (w_1, w_2, \bar{z}) that performs *OR*-operation.

[Solution](#) on page 295.

The neural networks and network operations are normally presented in vector form (and very often in matrix or even tensor form). In these examples, the input layer is feeding a vector $\mathbf{x} = (x_1, x_2)^\top$ to the output layer, and output layer is performing and operation

$$y = \mathbb{1}(\mathbf{x}^\top \cdot \mathbf{w} > \bar{z}) \quad (7.4.2)$$

where $\mathbf{w} = (w_1, w_2)^\top$ is the single neuron's vector of weights.

XOR-perceptron and hidden layers Can we use the same perceptron structure to perform *XOR*-operation? For *XOR*-perceptron, using the same model, we need values $\mathbf{w} = (w_1, w_2)$ and \bar{z} that represent the *XOR* column in Table 7.6. Using the vector notation (7.4.2), we can write four equations, one for each row of data in the table:

$$\begin{aligned} (0 \ 0) \cdot \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 0 < \bar{z} \quad (1 \ 0) \cdot \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = w_1 > \bar{z} \\ (0 \ 1) \cdot \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = w_2 > \bar{z} \quad (1 \ 1) \cdot \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = w_1 + w_2 < \bar{z}. \end{aligned} \quad (7.4.3)$$

The first expression shows us that \bar{z} is positive; the second and third equation show that both w_1 and w_2 are larger than \bar{z} ; but the fourth equation, corresponding the the $x_1 = 1$ and $x_2 = 1$ case contradicts the previous results by requiring that $w_1 + w_2 < \bar{z}$. Hence *XOR*-perceptron, using the model of Figure 7.4, is not possible.

A solution is to use a more complex network by introducing a *hidden layer* between the input and output layers (Figure 7.5). Now instead of connecting inputs \mathbf{x} to the output node y , we connect inputs to hidden layer nodes. In *XOR*-perceptron, it contains two nodes, h_1 and h_2 . Both of these nodes work in a similar fashion as the output node in the *AND*-perceptron: they take in

inputs, perform a linear transformation in the form $\chi = \mathbf{x}^\top \mathbf{w}$, and activation in the form $h = \mathbb{1}(\chi > b)$. However, their outputs h are not the network final outputs, but are fed to the output layer node y instead. That one performs exactly the same way as earlier, just it takes its inputs not from the input layer but from the hidden layer.

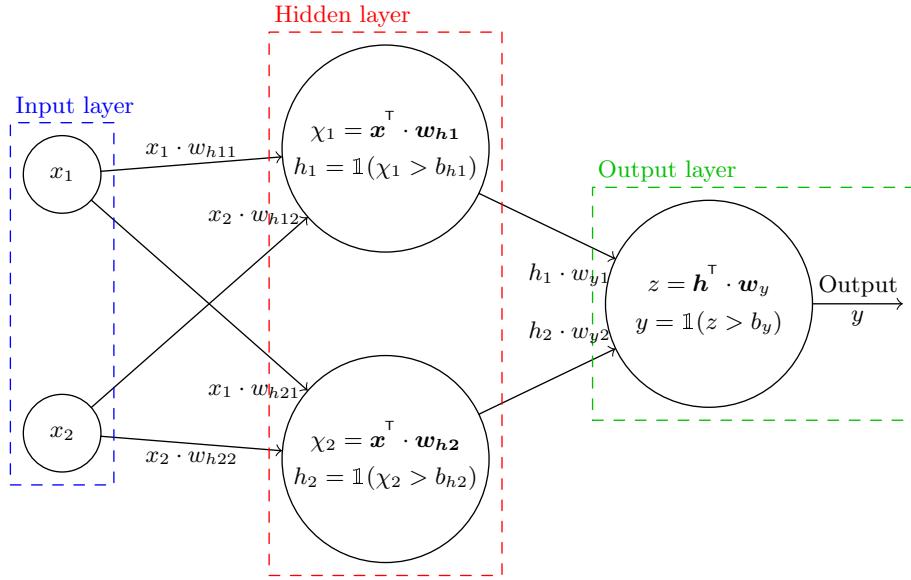


Figure 7.5: XOR Perceptron. The inputs x_1 and x_2 form the input layer, but now both input layer nodes are connected to both hidden layer nodes h_1 and h_2 , and not to the output layer. Both hidden layer nodes perform linear transformation and activation, using different weights \mathbf{w}_h and biases b_h . The single output layer node behaves exactly like in case of AND-perceptron, just it gets its inputs from the hidden layer, not from the input layer.

Why is hidden layer called “hidden”? This is because we cannot observe these values in our data. Both inputs \mathbf{x} and outputs y are observable in labeled training data. But we usually have no information about the “correct” values of the hidden layer values. Even more, there may be no such thing as hidden layers in the actual data generating process, our hidden layer nodes are just convenient mathematical tools that do not correspond to anything in reality. However, in certain cases hidden layers may represent concepts that are interpretable. For instance, one has found that in image processing tasks, some layers and nodes tend to recognize lines, arcs, light areas and similar basic image features.

Let us now finish the *XOR* perceptron. There are many ways to set the parameters in a way that our network performs *XOR* operation. One particular way is to notice that $x_1 \text{ } XOR \text{ } x_2 = (x_1 \text{ OR } x_2) - (x_1 \text{ AND } x_2)$. Can we somehow, using the network in Figure 7.5, perform this subtraction? Yes we can. We can take the input nodes x_1, x_2 and the hidden layer node h_1 and convert these into

an *AND*-perceptron by setting $\mathbf{w}_{h1} = (1 \ 1)^\top$ and $b_{h1} = 1.5$. Thereafter we can take x_1, x_2 , and the second hidden layer node h_2 and make an *OR*-perceptron by setting $\mathbf{w}_{h2} = (1 \ 1)^\top$ and $b_{h2} = 0.5$. And finally we can make the output layer node y to subtract *AND* from *OR* by setting $\mathbf{w}_y = (-1 \ 1)^\top$. The activation process must leave both values “1” and “0” unchanged, so we can pick $b_y = 0.5$.

Table 7.7 lists all the parameters for the perceptron in Figure 7.5. All in all we have 9 parameters. Obviously there is an infinite number of possibilities to choose the parameters, e.g. if we multiply all the weights and biases by a (positive) constant, the results are unaffected. We can also swap the role of h_1 and h_2 , and introduce many other changes without affecting the predictions of our network. Although such flexibility may seem advantageous, it also suggests that neural networks are prone to overfitting and should normally be used with some form of regularization. It is also obvious that *XOR* binary logical operation does not have anything like “hidden layer”. There is no way to measure the “true values” of χ_1 and χ_2 , these values are just a trick to make the perceptron be able to perform *XOR*. So at least in this perceptron, the hidden values remains “hidden”, or perhaps even non-existent.

Table 7.7: XOR-perceptron parameters

node	weights	bias
h_1	1, 1	1.5
h_2	1, 1	0.5
y	-1, 1	0.5

Exercise 7.4.2: Use the perceptron for *XOR*

Show that the perceptron with parameters as given in Table 7.7 can perform *XOR*. In particular, show that $0 \text{ } XOR \text{ } 1 = 1$.

Hint: set $x_1 = 0, x_2 = 1$, and compute all the hidden values χ_1, h_1, χ_2, h_2 and z . Do you get $y = 1$?

Computations the network nodes do are usually written (and coded) in matrix (or tensor) form. We can explain this by re-visiting the hidden layer of XOR perceptron example. The first hidden node χ_1 does the linear transform $\chi_1 = \mathbf{x}^\top \cdot \mathbf{w}_{h1}$ and the second hidden node $\chi_2 = \mathbf{x}^\top \cdot \mathbf{w}_{h2}$. We can combine these two transformations into

$$\begin{pmatrix} \chi_1 \\ \chi_2 \end{pmatrix} = \begin{pmatrix} \mathbf{w}_{h1}^\top \cdot \mathbf{x} + b_{h1} \\ \mathbf{w}_{h2}^\top \cdot \mathbf{x} + b_{h2} \end{pmatrix}. \quad (7.4.4)$$

And simplify this into a matrix form

$$\boldsymbol{\chi} = \mathbf{W}_h \cdot \mathbf{x} + \mathbf{b}_h. \quad (7.4.5)$$

Here χ is the vector of linear terms inside of the hidden layer nodes, W_h is matrix of hidden layer weights, where rows are the nodes and columns are the elements that correspond to the input vector. Finally, b_h is a vector of hidden layer biases.

Activation

Above we explained that all nodes two both the linear operations (that can be expressed in matrix form), and activation. There are three popular activation functions:

ReLU (rectified linear unit) is perhaps the most popular activation function. It is essentially a line with kink at 0, and defined as

$$\text{ReLU}(x) = \max(x, 0) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases} \quad (7.4.6)$$

It has the advantage of being simple and being piecewise linear, and hence easy to compute. Its derivative is either 0 or 1, although is not differentiable at 0. It seems to matter little in practice, although most theoretical optimization literature seem to only handle differentiable (Lipschitz continuous) functions (see Bottou *et al.*, 2018). Relu is perhaps the most popular activation function for neural networks everywhere, except in the output layer.

Softmax or Logistic Softmax (multinomial logit) is defined as logistic transformation of input vector \mathbf{x} . It's i -th component is

$$\Lambda(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (7.4.7)$$

Its one-dimensional version is the same as logistic function, and is often called *sigmoid* function in ML literature. It is often denoted by $\sigma(x)$. See Figure 7.6

Softmax has a very useful property: namely, whatever the value of x_i , e^{x_i} is always positive. And hence $\sum_j e^{x_j}$ is always positive, and hence the components of $\Lambda(\mathbf{x})$ sum to unity. So softmax is a transformation that converts all kind of inputs into valid probabilities and hence it is the most popular output layer activation function for categorization problems. Whatever comes to the output layer, it always outputs a valid probability vector.

No activation Finally, third popular activation function is no activation (or identity function as activation function). This is useful for regression models. In fact, neural network with no hidden layers and identity activation is equivalent to linear regression.

TBD: multiple outputs

TBD: outputs: prediction vs regression

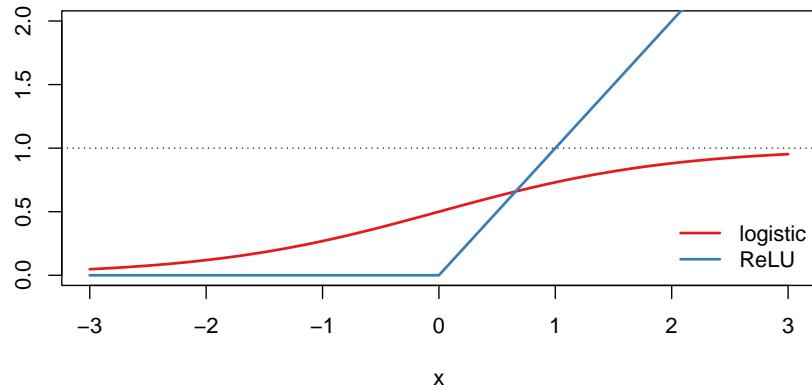


Figure 7.6: Popular activation functions for neural networks

7.4.2 Convolutional Neural Networks

Convolutional neural networks are networks that incorporate *convolutions*, certain kind of weighted sums over spatially arranged data. Convolutional filters are well-known methods to manipulate and enhance images or other signals, and have been widely used long before the neural networks became popular.

In the following sections, we explain what are convolutions, how they can be used to detect image elements, and how they can be incorporated into neural networks.

Convolutions and convolutional filters

Convolution is essentially a weighted sum or average over certain spatial region of the data. It is a function of two sources of information—data and weights. *Spatial* in this context means not necessarily space, but all other kind of arrangements where it makes sense to talk about distance between data points. This includes images where we can talk about neighboring pixels, or pixels that are farther apart, or time series, where we can talk about observations that are taken in next day, versus two days ago.

For instance, a convolution of series of daily temperatures T_t may involve sum of temperature T of the given day t , plus two days before and after that day. The weights may be larger for the given day and smaller for the other days, for instance $w_{-2} = w_2 = 0.25$, $w_{-1} = w_1 = 0.5$ and $w_0 = 1$. Here the index for weights denotes how many days off we are from the central day of interest. Convolution is often denoted by $*$, and for the daily temperatures we may write the convolution as $T * w$. As we can choose an arbitrary day t as the day of

interest, the result depends on t and we write $(T * w)(t)$:

$$(T * w)(t) = w_{-1}T_{t-2} + w_{-1}T_{t-1} + w_0T_t + w_1T_{t+1} + w_2T_{t+2}. \quad (7.4.8)$$

The vector of weights $w = (w_{-2}, w_{-1}, w_0, w_1, w_2)$ is called *kernel*, and also *filter*.

Example 7.4.1: 1-D convolution

Imagine we are measuring temperature on Pluto. But it is far away and we only get time at the expensive telescope once a month to do the measurements. We measure the temperature in five consecutive months and get $T_1 = 40K$ (-233C), $T_2 = 45K$, $T_3 = 45K$, $T_4 = 50K$, and $T_5 = 45K$. As our measurements are imprecise, we may want to smooth the individual observations over time (compute moving average). But we also want to put more weight on the current month's measurement and less weight on the neighboring months, as the measured differences may reflect true processes on Pluto. So we choose weights $w_{-1} = 0.25$, $w_0 = 0.5$, and $w_1 = 0.25$. The convoluted (averaged) temperatures are:

$$\begin{aligned}(T * w)(2) &= w_{-1}T_1 + w_0T_2 + w_1T_3 = 43.75 K \\ (T * w)(3) &= w_{-1}T_2 + w_0T_3 + w_1T_4 = 46.25 K \\ (T * w)(4) &= w_{-1}T_3 + w_0T_4 + w_1T_5 = 47.50 K.\end{aligned}$$

In this way we can use convolutions for smoothing noisy observations. This is also why we want the weight to sum to unity in this case—we do not want our smoothed temperature values to be systematically biased.

For 2-D case, convolutions are defined in a similar fashion. For instance, we may convolve surface temperature over certain geographic area by adding measurements in this area while giving more distant measurements lower weight (or maybe larger weight, depending on our task). The main difference between 1-D and 2-D case is that now we need a 2-D weight structure. So instead of a simple sum, now we need a double sum

$$(T * W)(k,l) = \sum_i \sum_j w_{ij} T_{k+i,l+j}, \quad (7.4.9)$$

where the weight matrix $W = \{w_{ij}\}$. For instance, when doing a similar temperature measurements, but now over a rectangular grid, we may set the weight matrix to

$$W = \begin{pmatrix} \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \\ \frac{2}{16} & \frac{4}{16} & \frac{2}{16} \\ \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \end{pmatrix}. \quad (7.4.10)$$

As in case of time series smoothing, we chose the weights to sum to unity in order to avoid systematic bias in temperature. But in other applications, the

weights do not have to sum to one. We also do not have to choose a 3×3 kernel, it may be of any dimension, including even dimension like 2×2 , or non-square, such as 2×3 .

In neural networks, 2-D convolutions are one of the main workhorses for image processing. Let us demonstrate 2-D convolution by constructing a filter that detects vertical edges in images.

We pick a very simple 3×4 image, that include a 2×2 black box in the top-right corner (see Figure 7.7 left). For simplicity, the image only has two possible pixel values, 0 (white) and 1 (black). We choose 2×2 kernel with weights as

$$W = \begin{pmatrix} -1 & 1 \\ -1 & 1 \end{pmatrix}. \quad (7.4.11)$$

One can easily imagine convolutions as moving this kernel window over the image, pixel-by-pixel. At each location one has to multiply the pixel values with the corresponding kernel weights and add the results.

The image has one vertical edge in the upper-middle of the image, the one that separates the black and white area. We start moving the convolution filter across the image the top-left corner (violet frame). This results in zero value, as all the pixel values are 0. Next, we move the kernel window right by one pixel (olive frame). In that position the negative filter weights overlap with 0-values pixels while positive weights overlap with 1-value pixels. As a result, the filter returns 2. Finally, in the rightmost position (green frame), both positive and negative weights overlap with equal pixel values (1) and hence the output is again 0. The filter only “fires” if there is a vertical edge on the image. The figure only depicts convolution along the upper row of the image. If we lower the filter down by one pixel, then the all-zero lowermost row does not contribute to the output, and we get values 0, 1, and 0. The filter still “fires” for the vertical edge in the middle, but not it fires only “partially” because the vertical edge is only partially in the filter’s window.

The result of moving over the image is a new 2-D layer of “vertical edgeness” of the image (ther “Result” box in the figure). In this example, the largest edgeness value is in the middle-upper position, the position where the window exactly overlaps the vertical edge. Values 0 correspond to the case where the image does not contain any vertical edges, and value 1 corresponds to the case where the filter window only partially contains a vertical edge. Now if we want to know if any particular location contains a vertical edge, we have to see what are the values of the result layer. Large value represent to a vertical edge.⁶

It is possible to hand-craft more different filters. For instance we can add a filter for horizontal edges, a filter for corners, for diagonal lines and for other objects. However, in case of neural networks we normally do not hand-craft the filters but let the networks to learn those. For example, the network may learn that many vertical edges suggests that the picture is about a building, while human face may be better visible when using curved lines. Even more,

⁶Large *positive* values represent an edge between white pixels left and black pixels right. Large *negative* values correspond to the opposite case.

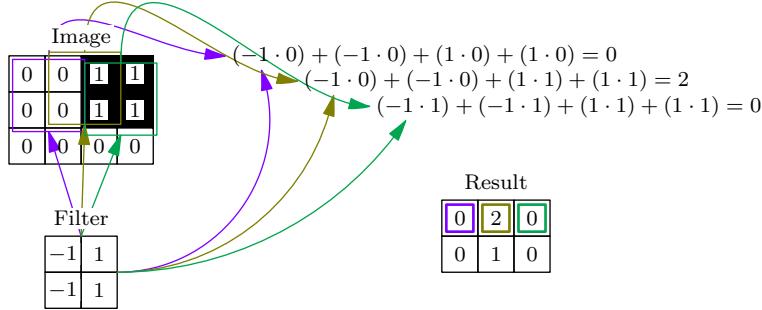


Figure 7.7: Vertical edge detection using 2D convolution filter. Moving the convolution filter across the image, the top-left corner (violet frame) results in zero, as all the pixel values are 0. In the next position (olive frame) the negative filter weights overlap with 0-values pixels while positive weights overlap with 1-value pixels. As a result, the filter returns 2. Finally, in the rightmost position (green frame), both positive and negative weights overlap with equal pixel values (1) and hence the output is again 0. The filter only “fires” if there is a vertical edge on the image.

normally we allow the network to learn a large number (e.g. 64) filters, so it does not just rely on vertical edges, but can look at a large number of different details.

Note that filters are not limited to 2×2 size, they may be a lot larger, and they do not have to be of square shape.

Padding, Pooling, and Strides

Pooling After running the data through a convolutional filter, we have another matrix that tells how well did each place in the image correspond to what the filter captures. In the example in Figure 7.7 we can see that the edges of the image do not correspond to vertical edges, the lower-middle of the image corresponds somewhat, and top-middle corresponds the best. Often we are not interested in such a detailed knowledge. For instance, we may be interested in the location of the cleanest vertical edge while willing to ignore the other less-clean representation nearby.

In this case we may run the resulting data through a *pooling layer*. Popular max pooling finds the maximum value in a small area, e.g. in a 2×2 square on the layer. In this example, max pooling will result in value 2, indicating that there is a clear vertical edge somewhere in that region. It will however ignore 0-s and 1-s, so we do not learn that there is also places that do not represent vertical edges. If we are interested in the latter, we may choose average pooling instead. This will correspond to average “edgeness” of that region on the image.

Padding It is obvious from Figure 7.7 that the resulting layer is smaller than the original image layer. We have to fit the whole filter onto the image, and as

soon as its dimension is more than 1, we have fewer points in the output than in the input. We can proceed in different ways:

- We can just accept that this is the case, and that the layers get smaller and smaller as the data proceeds through successive convolutional layers.
- Alternatively we can “pad” the layers with certain values, e.g. some predetermined values, or values from nearby pixels. This is called *padding*.

Strides Finally, we do not have to move the convolution window by a single pixel each time. We may choose another step size, called *stride*. A large stride may be useful if the image is fuzzy, or if the features do not change rapidly from pixel to pixel. Large strides are a way to lower the resolution in the middle of the network. One may also choose different strides for horizontal/vertical movement.

From 2-D Images to Multi-Layer Images

The previous discussion was concerned around a single 2-D image layer. This describes well first convolution layer in black-and-white images that contain only a single color layer. But in case of color images (3 or 4 color layers), or in case of subsequent convolutional layers, the input data is not 2-D any more. In that case the kernel must be 3-dimensional: width, height, and number of layers. Normally we only adjust the width and height, and let the kernel include all image layers. This is because the layers are not spatially organized, there is no reason to say that, e.g. red and green layer should be next to each other while transparency layer is “further away”. So just include all layers into the kernel, and let the training process sort out which layers are important for each filter.

This explains the number of parameters we need for a convolutional layer. A kernel that is *width* pixels wide and *height* pixels tall, and processing an image of *layers* different layers, contains $(\text{width} \times \text{height} \times \text{layers} + 1)$ parameters (+1 is for bias). The whole convolutional layer takes

$$(\text{width} \times \text{height} \times \text{layers} + 1) \times \text{filters} \quad (7.4.12)$$

parameters where *filters* is the number of filters.

Chapter 8

Different Types of Data

Introductory machine learning problems are often presented using well-behaved numeric-only datasets. Here we look at some of the different data types and explain how to use these for ML models.

8.1 Text as Data

As literate humans, we receive quite a lot of information through written text. So text *is* data and it contains a lot of information. But to process text on computer poses a number of challenges. Text is non-numeric to start with, and unlike images where we can understand the pixel values, it is not obvious what might be the features in case of text.

Below we discuss two simple ways of using text, *bag-of-words* (BOW) and *term-frequency-inverse-document-frequency* (TF-IDF). To be more precise, TF-IDF is a BOW post-processing method. But before we get to text itself, we have to talk about pre-processing, such as tokenization because raw text is often not the best choice for processing.

8.1.1 Preprocessing: Tokenization, Stemming, and Lemmatization

Text processing typically starts with simple methods to simplify the text by removing various features that are not relevant for current task. For instance, we may consider case of the word irrelevant, and we may say that *speaking* and *speaks* is essentially the same word if we do for instance topic modeling. But case may be very important for other task, e.g. for extracting names from text.

Tokenization

Text is normally analyzed at word level. This means we have to split it into words. This is relatively easy with English and other European languages where

space and certain punctuation symbols are reliable word boundary markers. But other languages may not contain similar markers and hence the process is much more complex.

Even in English, we have a number of corner cases. For instance, what should we do with *don't*? Should we retain it as “*don't*”, convert it to “*do not*”, or just remove the apostrophe and make it into “*dont*”? In contrary, what about names like *Kuala Lumpur*? Should it be retained as a single word containing space, or split into two words? We have to make such decisions depending on the task at hand. As the result may deviate from what we commonly call “words”, this process is usually called *tokenization* instead, and the results are *tokens*, not “words”.

But we do not *have* to use words. We may break written text down to individual characters, character pairs, or syllables instead. The smaller units may be useful where we have to guess the meaning of words from how they are written, from prefixes and suffixes they contain, or if they contain syllables that also occur in other, known words.

Stemming

Stemming is a process where common prefixes and suffixes are removed from the words. For instance, when encountering the word *studying*, we may remove the suffix *-ing* leaving the stem *study*. More advanced stemming algorithms may leave only the part of stem that never changes, in this example, this would be *stud*. Note also that the never-changing-part of a word may differ between written and spoken language, or be completely missing (like in case of *go* and *went*).

Stemming helps us to see the common stems of related words, and we can for instance build a search engine that finds both *study* and *studying* when the user enters either of them. This is often what the users want when they search.

Lemmatization

However, stemming is a simplistic method that often fail to produce consistent stems for closely related words. For instance, a simple stemming algorithm may turn word *studying* into stem *study*, but the word *studies* will produce *studi*. As these stems are not identical, the search engine may not understand that the corresponding words are similar.

Lemmatization is a method that is in principle similar to stemming, but instead of just removing the standard suffixes, it uses morphological analysis of words to deduce the *lemma*, the standard base form of all the related words. Lemma is the standard form of words that is listed in dictionary. In the example above, the lemma of both *study* and *studies* is *study*. Needless to say, lemmatization is much more complex to implement than stemming and needs some sort of dictionary lookups.

Stopwords and Other Simplification

We often want to start by simplifying text. Typically one converts all words to lower case, removes punctuation, perhaps normalizes the grammatical constructs, and removes *stopwords*, common words like *and*, *not*, *but* that carry little information.

Obviously, whether capitalization, punctuation and stopwords are just a noise or actually helpful depends on the task. If you are predicting the topic of a news article, the common stopwords carry little information. However, when deducing authorship of a text, the subtle differences in stopword usage or the exact grammatical form of words used may turn out to be quite important.

8.1.2 *n*-grams

n-grams are just ordered sequences of *n* words (or other objects, such as letters or sentences). For instance, in case of document “The waiter opened the gate a little and looked out”, we can create the following bigrams (2-grams): (*the*, *waiter*), (*waiter*, *opened*), (*opened*, *the*), (*the*, *gate*), (*gate*, *a*), (*a*, *little*), (*little*, *and*), (*and*, *looked*), (*looked*, *out*). *n*-grams can be used in a similar fashion as tokens, their main advantage is that they preserve the order of the words. For instance, it is harder to deduce meaning of two tokens (unigrams) “do” and “not”, while a bigram (do, not) has much more distinct meaning.

However, texts contain many more different *n*-grams than unigrams, and hence working with *n*-grams typically needs more resources and training data.

8.1.3 Bag of Words and Document-Term-Matrix

Prerequisites: [Metric distance](#), [vector norm 2.2.2](#), [cosine similarity 3.2.2](#).

Statistical methods only work on numerical data, so before we can apply any common ML method on text we have to convert it into a numeric representation. *Bag of words* (BOW) is a simple and popular approach to transform texts into a numeric vector form, in essence just a frequency table of words in the text. An essential property of BOW is that it does not preserve the order of words. One can imagine throwing all the words into a bag, so for each document it will just contain the counts of the words but not their order. We obviously lose a lot of information by such “bagging”, sometimes it is useful, sometimes it is undesirable. Normally we work on many documents which may be short (like tweets) or long (like books). We can construct a BOW for each of these and stack them into a matrix, called *Document-Term-Matrix* (DTM).

DTM can be constructed in different ways, here we explain how to create it based on word counts (or more precisely, token counts), but one can choose to remember just the presence of words, not their counts. One can also construct bag-of-characters or bag-of-bigrams instead of bag-of-words.

In this form we represent documents as vectors of word counts in the vocabulary: first we collect all words in all the documents we analyze. This collection

is called *vocabulary*. Say there are V words in the vocabulary. Now we can represent each document as a vector of length V , $\mathbf{x} = (x_1, x_2, \dots, x_V)^\top$, where each component x_j equals to the count of that word j in the text. If the word is not present, we set $x_j = 0$.

BOW-s are numeric vectors we can use for various mathematical operations. For instance, we can compute both Euclidean distance or cosine similarity between BOW-s. When stacking BOW-s horizontally underneath each other, we get a DTM, essentially a design matrix where features are the words, and feature values are the word counts in each document (observation).

Example 8.1.1: DTM of Laozi quotes

Let us create a DTM of two Laozi quotes: “*Knowing others is wisdom, knowing yourself is Enlightenment*”, and “*Mastering others is strength. Mastering yourself is true power*”.

These quotes together form vocabulary of size 10 (in alphabetical order) *enlightenment, is, knowing, mastering, others, power, strength, true, wisdom, yourself*. The first quote only contains words *enlightenment, is, knowing, others, wisdom, yourself* and hence the corresponding BOW is $\mathbf{x}_1 = (1, 2, 2, 0, 1, 0, 0, 0, 1, 1)^\top$. We must keep the word counts in a consistent order, normally in the same order as the words are in the vocabulary. The number “1” in the first position indicates that the word *enlightenment* is present 1 times, but for instance the words *is, knowing* are there two times, and words *mastering, power, strength, true* are not present at all. The second quote contains *enlightenment, is, knowing, others, wisdom, yourself* and hence its BOW is $\mathbf{x}_2 = (0, 2, 0, 2, 1, 1, 1, 1, 0, 1)^\top$. The vocabulary and both BOW-s are shown in the table below.

Table 8.1: Two BOW-s \mathbf{x}_1 and \mathbf{x}_2 , corresponding to the two Laozi quotes in the text. Both BOW-s, stacked horizontally underneath each other as in this table, form a numeric DTM that can be used in various machine learning models.

	enlightenment	is	knowing	mastering	others	power	strength	true	wisdom	yourself
\mathbf{x}_1	1	2	2	0	1	0	0	0	1	1
\mathbf{x}_2	0	2	0	2	1	1	1	1	0	1

Let us also compute Euclidean distance and cosine similarity between these two quotes. For the Euclidean distance we need their difference $\mathbf{x}_1 - \mathbf{x}_2 = (1, 0, 2, -2, 0, -1, -1, -1, 1, 0)^\top$ and hence $d_e(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^\top \cdot (\mathbf{x}_1 - \mathbf{x}_2)} = \sqrt{13} = 3.606$. For cosine similarity, we need to compute the inner product $\mathbf{x}_1^\top \cdot \mathbf{x}_2 = 6$ and both norms, $\|\mathbf{x}_1\| = \sqrt{12} = 3.464$ and $\|\mathbf{x}_2\| = \sqrt{13} = 3.606$, and hence $c(\mathbf{x}_1, \mathbf{x}_2) = 6 / (\sqrt{12} \cdot \sqrt{13}) = 0.48$. In some applications we may

also want to remove the stopword *is*.

The advantage of DTM is its simplicity. Creating a BOW for given task requires no training data and can be done fast and easily. Sparseness of typical DTM helps to increase the algorithm speed and decrease the memory requirements, as typical vocabularies contain 10,000 to 100,000 words while most of the documents do not contain most of the words. Hence the DTM-s are in practice mostly filled with zeros, and we can use sparse matrices as the underlying data structures.

DTM-s have two major disadvantages. First, they are large. Typical word-based DTM-s contain 10,000–100,000 words, and hence every document, even just a tweet of a single word, contains this many numbers. This may make large DTM-s slow and sluggish.¹ Second, by construction they do not store the order of words. Whatever the order of words, the data looks identical. A potential way to address this issue is to use a bag of bigrams instead of BOW. As bigrams retain the order of words, such a bag will contain much of the information of the original word order. However, as there are many more bigrams compared to words, the dimensionality problem will get worse.

A potential solution to the dimensionality problem stems from the typical word distribution in natural languages. While there is a core of very frequent words, most of the words in a vocabulary are rare. Note that in practice there is always a large number of rare words. If we increase the sample size (the number or size of documents), we sample a larger number of formerly rare words so those are not rare in our BOW any more. But in actual applications, larger documents will always contain many even less common words, misspellings, names and acronyms, so the problem of a large number of infrequent words does not go away. But often we can just disregard such words—seeing a word only a few times is arguably too little to make any inference about its role for language models. Hence a common strategy is to exclude all words that are less frequent than a given threshold.

8.1.4 TF-IDF

As DTM is effectively a numeric design matrix, we can use it with a plethora of traditional ML methods. For instance, we may categorize texts using k -NN and cosine similarity. Unfortunately, this measure may not perform very well in practice. First, often the words that are common in both texts are popular ones that carry little distinctive power. Even if we remove obvious stopwords, there are still many common words that occur repeatedly in most of the texts. Second, less popular words tend to be used in a “bursty” way, so if one word is already used in a document, it will be very likely used again. Just word counts will put too much weight on a few words that are used many times in the texts.

¹This sounds like contradicting the praise of simplicity and sparsity in the previous paragraph, but it is not. Sparsity often helps, but it is not a cure against all inefficiencies. Small dense matrices are still much more efficient than large sparse matrices.

As a remedy, one may use term-frequency inverse document frequency (*TF-IDF*) transformation. There are many different specific definitions of TF-IDF in the literature, here we follow the approach of [Murphy \(2012, p 482\)](#).

First, we transform the word counts into *tf* form as

$$\text{tf}(x_{ij}) = \log(1 + x_{ij}) \quad (8.1.1)$$

where x_{ij} is the count of word j for the text i . This transformation suppresses large counts of single words, but is still more informative than just binary contains/does not contain features. By adding 1 to x_{ij} we ensure TF of a missing word is zero, and TF for every word present in document is positive.

Second, for each word j in the vocabulary, we define *idf* as logarithm of the inverse of number of documents that contain the word j . Normally we adjust the inverse a bit, e.g. we take inverse of 1 plus the number of documents in order to avoid cases where a vocabulary word is not found in any document.² Formally, we may define *idf* as

$$\text{idf}(j) = \log \frac{N}{1 + \sum_{i=1}^N \mathbb{1}(x_{ij} > 0)}. \quad (8.1.2)$$

$\mathbb{1}(x_{ij} > 0)$ is the indicator function that equal to one if the document i contains word j , and hence $\sum_{i=1}^N \mathbb{1}(x_{ij} > 0)$ is the count of documents that contain the word j . We also use the total number of documents N as numerator. This is a form of normalization where the words that are present in all documents will have the fraction of $N/(1 + N) \lesssim 1$ and hence its logarithm $\text{idf} \lesssim 0$. In the opposite end, IDF for a word that is found in none of the documents is $\log N$ and for a word in a single document only, $\text{idf} = \log N/2$. Hence *idf* assigns to words that are present in many documents low weight, and words that are present in only a few documents high weight. Note that *idf* assigns a single value for each word across all documents. IDF is a word-specific value while TF-vectors are different for each BOW.

Finally, the TF-IDF transformation for word j is defined as

$$\text{tf-idf}(x_{ij}) = \text{tf}(x_{ij}) \cdot \text{idf}(j) \quad j \in 1 \dots K. \quad (8.1.3)$$

Note that TF-IDF is not made of single document alone—it is a transformation of the complete DTM \mathbf{X} . While each single BOW (a row in the original data matrix \mathbf{X}) has been transformed into a row of TF-IDF matrix $\tilde{\mathbf{X}}$, the transformation needs information about how common are the words across all documents. In some ways it is similar to [feature normalization](#).

²There may be several reasons that a word that is in no document finds its way to the vocabulary. For instance, one may use a standard vocabulary, derived from other documents. Also, training data typically contains words that are in no validation document.

Example 8.1.2: TF-IDF of Laozi quotes

Let us TF-IDF transform the DTM of the Laozi quotes, *Knowing others is wisdom, knowing yourself is Enlightenment* and *Mastering others is strength. Mastering yourself is true power*, presented in Table 8.1.

Table 8.2 shows the results. The columns (features) are the $V = 10$ vocabulary words and two first lines represent the DTM as in Table 8.1. The following two lines, tf_1 and tf_2 show the corresponding tf -terms, essentially just log-transforms of the DTM. The row idf is the IDF term. It is $\log 2/(1+2) \approx -0.41$ for words that are in both documents and $\log 2/(1+1) = 0$ for words that are in a single document only. Although we do not have any such example, it would be $\log 2 = 0.69$ for words that are in none of the quotes. Finally, the last rows $tf\text{-}idf_1$ and $tf\text{-}idf_2$ are just the corresponding tf -terms multiplied by idf . These two rows form the TF-IDF-transformed data matrix \tilde{X} .

Table 8.2: Example vocabulary, bag-of-word vectors, and TF-IDF transformation for quotes: “*Knowing others is wisdom, knowing yourself is Enlightenment*” and “*Mastering others is strength. Mastering yourself is true power*”.

	enlightenment	is	knowing	mastering	others	power	strength	true	wisdom	yourself
\mathbf{x}_1	1.00	2.00	2.00	0.00	1.00	0.00	0.00	0.00	1.00	1.00
\mathbf{x}_2	0.00	2.00	0.00	2.00	1.00	1.00	1.00	1.00	0.00	1.00
tf_1	0.69	1.10	1.10	0.00	0.69	0.00	0.00	0.00	0.69	0.69
tf_2	0.00	1.10	0.00	1.10	0.69	0.69	0.69	0.69	0.00	0.69
idf	0.00	-0.41	0.00	0.00	-0.41	0.00	0.00	0.00	0.00	-0.41
$tf\text{-}idf_1$	0.00	-0.45	0.00	0.00	-0.28	0.00	0.00	0.00	0.00	-0.28
$tf\text{-}idf_2$	0.00	-0.45	0.00	0.00	-0.28	0.00	0.00	0.00	0.00	-0.28

The TF-IDF-transformed data \tilde{X} is a similar numeric data matrix like DTM and can be used in different ML models as any other numeric data. It gives sometimes quite a substantial improvement in the modeling accuracy. It is also easy and fast to perform, involving just a few operations that can be easily vectorized.

TBD: Word embeddings

TBD: Converting text to features. BOW/embeddings + n-grams + part-of-speech + other kind of features

Chapter 9

Machine Learning Techniques

9.1 Loss Function and Non-Linear Optimization

Statistical problems typically require estimation of certain parameters (fitting the model) based on data. The parameters may be interesting itself, or these may be needed for predictions, hypothesis testing or other reasons. For instance, in case of linear regression, these are parameter β , in case of regression trees these are the splitting and stopping rules for each branch. More complex models, such as neural-network based image recognition tasks can be imagined as many layers of linear regression models on top of each other and can contain millions or hundreds of millions of parameters.

If we move beyond the simplest cases, it is completely infeasible to find the best parameter values manually. We need methods to do this on computer, to do it fast, and in a reliable fashion. Typically this proceeds through *non-linear optimization*, a technique where a certain function (called *loss function* or *objective function*) is minimized or maximized by manipulating the parameters. The parameter value that results in the smallest loss value is the best parameter, the solution.¹

Next we will look at some details of non-linear optimization. These notes will only give a brief overview of the methods in order to prepare you for applications.

9.1.1 Loss Function

A large class of statistical models involves a function, *loss function*, that describes how “bad” is the model. For instance, linear regression is defined through sum of squared errors as

$$SSE(\beta) = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = (\mathbf{y} - \mathbf{X}\beta)^\top (\mathbf{y} - \mathbf{X}\beta). \quad (4.1.49)$$

¹Not all models require such non-linear optimization. For instance, k -nearest neighbors do not contain any parameters, and Naive Bayes parameters (conditional probabilities) can be computed directly from the data without any optimization.

SSE is a measure of model “badness” and hence $SSE(\beta)$ can be understood as the loss function. It depends on the parameter vector β and hence we can manipulate β to get larger or smaller losses. Non-linear optimization is a technique (more precisely, a set of many different techniques) that systematically manipulate the parameter in order to find the smallest loss.²

Why do we want the smallest loss? If we are interested in prediction, then SSE is one of the most obvious measures of the model predictive power (or rather lack of it as large SSE corresponds to low power). So in this case it is almost trivially true that small loss is equivalent to good model. If we are interested in inference, we may need additional assumptions regarding the “true” model and data.

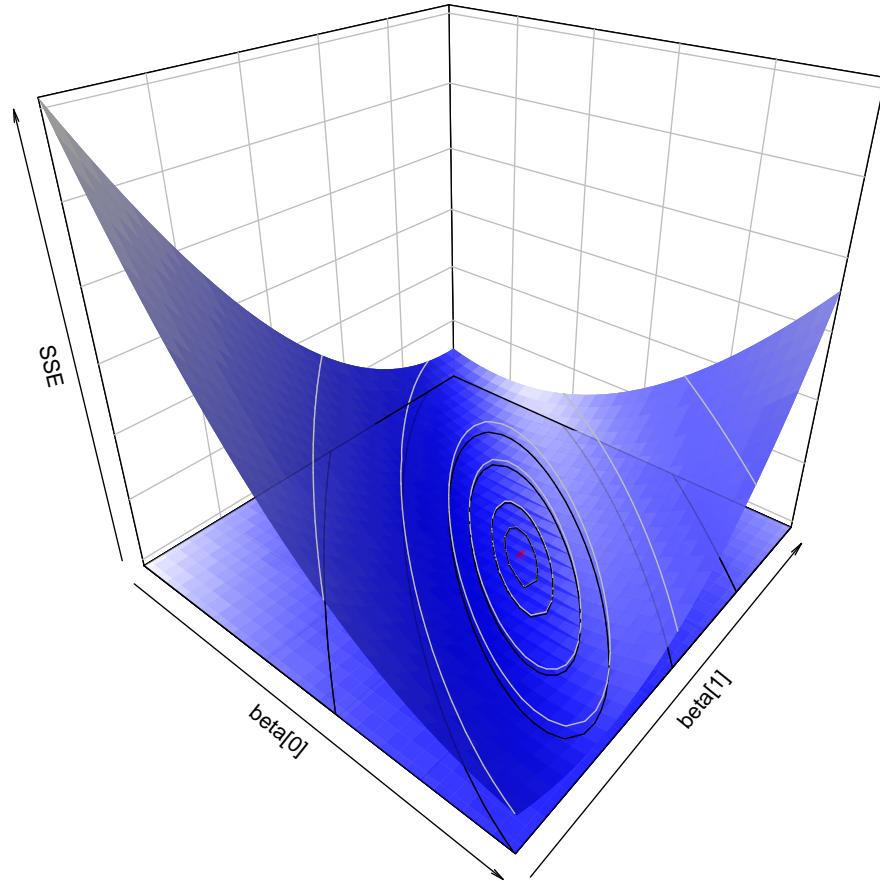


Figure 9.1: SSE as a function of β_0 and β_1 .

²As a side note—we do not actually need non-linear optimization for linear regression. This is the only statistical model where we can do the optimization analytically and directly compute the solution.

Let us illustrate the loss function with a figure. Figure 9.1 shows the Hubble regression (see Example 4.1.2 on page 99). The regression is in a form

$$v_i = \beta_0 + \beta_1 R + \epsilon_i \quad (9.1.1)$$

where v is velocity of galaxies (km/s) and R is distance (Mpc). So this model only has two parameters and hence it can be visualized. On one axis of the figure we display β_0 , on the other β_1 and the vertical axis describes SSE . The loss function describes an elongated parabolic surface with minimum at $\beta^* = (-40.4, 453.9)$. This is the solution to the linear regression problem shown in Example 4.1.2 above.

Sometimes we want to maximize a function instead of minimizing it, in that case it is often referred to as *objective function* instead of *loss function*. More strictly speaking, objective function is a more general term, it is a function that should be either minimized or maximized (i.e. *optimized*) in order to find the solution. So loss function is also an objective function.

9.1.2 Maximum Likelihood (ML)

Maximum Likelihood is a M -estimator where the parameter estimate is established by maximizing its *likelihood* (in practice, almost always, log-likelihood). Likelihood is essentially the probability to observe the data, given it's parameter values. This assumes we have established the data generating process (DGP) for the observations. For instance, in case of a sample of random variables (RV), DGP is essentially the distribution we assume the data is originating from. In case of a regression model we may assume that our independent variables (features) are given and exogenous, while the response variable is calculated based on the features and a random disturbance term. Note that with *data*, we mean all data that goes into the model, including the explanatory and response variables.

A few examples

Below are a few examples of the probability to observe the data, equivalent to the likelihood function.

Coin Toss We toss a coin 4 times. We get H twice and T twice. We know it is a fair coin. This is a binomial process. The probability to observe k heads in n coin tosses where probability of a head is p is

$$\Pr(X = k) = C_k^n p^k (1 - p)^{n-k}. \quad (9.1.2)$$

(Note: if you are doing numeric computations, you may prefer the R function `pbisom()` instead.) The multiplier C_k^n counts how many different ways there are to observe k heads in n tosses. This is the binomial probability mass function (pmf).

The probability to observe two heads and two tails is accordingly

$$\Pr(H,H,T,T) = C_2^4 0.5^2 (1 - 0.5)^2 = 0.375. \quad (9.1.3)$$

Independent Normals We are given a sample X_1, X_2, \dots, X_n of independent draws from standard normal distribution. Note the normality of the data must either be assumed, or in rare cases somehow learned (e.g. through theoretical considerations).

Note that as we are dealing with a continuous distribution, we cannot directly write down the probability of the data. However, as the density function (pdf) gives us the probability per unit interval, we can write the probability of data per unit interval:

$$\Pr(X_1, X_2, \dots, X_n) = \phi(X_1) \cdot \phi(X_2) \cdots \phi(X_n) \quad (9.1.4)$$

where $\phi(\cdot)$ is the normal pdf. Note that the assumption of the independence allows us to write the probability as a product of individual probabilities.

Examples Involving Unknown Parameters

The previous examples did not contain any unknown parameters, so there was little left to be estimated. We can make these examples more interesting by including a parameter.

Coin Toss We toss a coin 4 times. We get H twice and T twice. We don't know whether it is a fair coin. What is the best estimate for p , the probability to receive a head?

Let's start with the probability to observe the data. As in (9.1.2) and (9.1.3) we have:

$$\Pr(H,H,T,T) = C_2^4 p^2 (1-p)^2 = 6 p^2 (1-p)^2 \equiv \mathcal{L}(p). \quad (9.1.5)$$

$\mathcal{L}(p)$ is the *likelihood function*. It is essentially the same probability, just we stress that the main arguments of interest are the parameters. Binomial process only has a single parameter p .

Which value of p will give the highest $\mathcal{L}(p)$ value? Let's start with a grid search (Figure 9.2). As one can see, the optimal value is 0.5. This is not surprising, as intuitively the best estimate for p is simply the sample mean.

In practice, one almost always works with log-likelihood instead of likelihood. This is for two reasons, first the likelihood values are in realistic setting often too small for current computers to represent; and also log-likelihood has a number of attractive statistical properties. Obviously, as log is a monotonic function, the optimum parameter value will remain the same. In this simple example, we can analytically solve for the optimal value:

$$\ell(p) \equiv \log \mathcal{L}(p) = \log 6 + 2 \log p + 2 \log(1-p). \quad (9.1.6)$$

The optimum can be solved through a simple calculus. The optimum condition is

$$\frac{\partial}{\partial p} \ell(p) = \frac{2}{p} - \frac{2}{1-p} = 0 \quad (9.1.7)$$

from where follows that the optimal $\hat{p} = 0.5$.

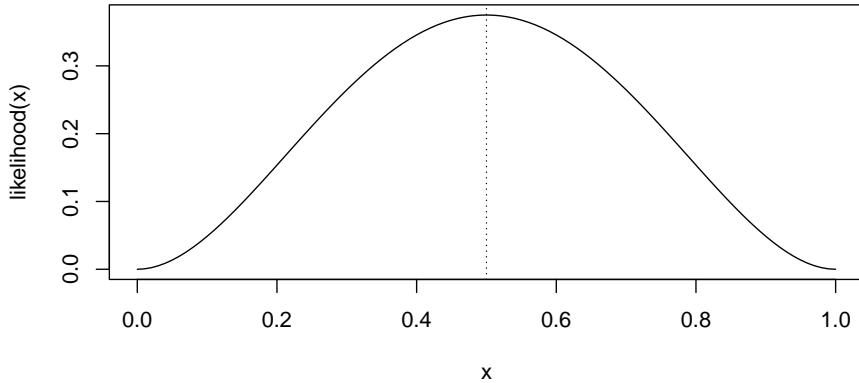


Figure 9.2: Likelihood value for the coin toss, depending on the head probability p

Mean of Normals We are given a sample X_1, X_2, \dots, X_n of independent draws from a normal distribution with variance one. Unlike in the case above, we don't know what is the mean of the distribution. Let's estimate it by ML.

For a single observation we have

$$\Pr(X = x; \mu) = \phi(x - \mu) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x - \mu)^2\right). \quad (9.1.8)$$

As before, for the numeric analysis you may prefer the R function `dnorm` instead of this expression.

For n independent normals

$$\begin{aligned} \Pr(X_1 = x_1, X_2 = x_2, \dots; \mu) &= \\ &= \phi(x_1 - \mu) \cdot \phi(x_2 - \mu) \cdots \phi(x_n - \mu) = \\ &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x_1 - \mu)^2\right) \times \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x_2 - \mu)^2\right) \times \cdots \\ &\quad \times \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x_n - \mu)^2\right) \end{aligned} \quad (9.1.9)$$

This is essentially the likelihood function $\mathcal{L}(\mu)$. For log-likelihood we have

$$\begin{aligned}\ell(\mu) &= \\ &= -\frac{1}{2}(\log 2 + \log \pi) - \frac{1}{2}(x_1 - \mu)^2 - \frac{1}{2}(\log 2 + \log \pi) - \frac{1}{2}(x_2 - \mu)^2 - \\ &\quad \cdots - \frac{1}{2}(\log 2 + \log \pi) - \frac{1}{2}(x_n - \mu)^2 = \\ &= -\frac{n}{2}(\log 2 + \log \pi) - \frac{1}{2} \sum_{i=1}^n (x_i - \mu)^2.\end{aligned}\quad (9.1.10)$$

Our next problem is to maximize this log-likelihood. First, note that the first term in the last row in (9.1.10) does not contain the parameter μ . Hence it drops out when we take derivative of it with respect to μ . Second, note that what is left is maximizing $-\frac{1}{2} \sum_{i=1}^n (x_i - \mu)^2$ which is equivalent to minimizing sum of squared deviations. Hence least squares estimator is equivalent to ML estimator in case of normal disturbances.

This problem is easy to be solved analytically. From the optimum condition we know:

$$\frac{\partial}{\partial \mu} \ell(\mu) = -\frac{\partial}{\partial \mu} \left(\frac{1}{2} \sum_{i=1}^n (x_i - \mu)^2 \right) = \sum_{i=1}^n (x_i - \mu) = 0\quad (9.1.11)$$

and hence ML estimator $\hat{\mu} = \sum_i x_i / n$. Not surprisingly, the intuitive estimator of distribution mean, the sample mean, also turns out to be its ML estimator.

TBD: elliptical curves, overshooting, flat areas, local minima

TBD: learning rate

TBD: feature scaling

9.2 Gradient Ascent

Prerequisites: Vectors and matrices, gradient

Gradient Ascent (GA) and its mirror image Gradient Descent (GD), is a popular method to find maxima and minima (collectively called *optima*) of functions. Gradient ascent is widely used in various machine learning applications, it also serves as a basis for many more complex methods, such as Newton-Raphson. While one can easily find analytic solutions for the optimum of simple functions, such as quadratic function, this is not possible for more complex cases. The “more complex” includes almost all objective functions we encounter in machine learning practice, the linear regression being the only notable exception. So we have to rely on numerical computations, usually referred as *non-linear optimization*, to find the solution. GA is one of the most popular of these methods.

The GA idea in a 2-D case is depicted in Figure 9.3.

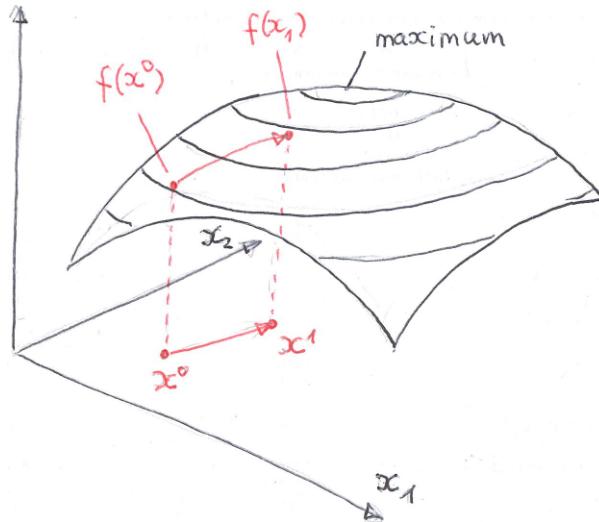


Figure 9.3: One step of Gradient Ascent. We start from an initial guess \mathbf{x}^0 and take a step along the gradient. This moves us uphill to \mathbf{x}^1 . The function $f(\mathbf{x})$ is depicted by the surface overlaid by (rather circular) level sets.

1. Start with an initial guess \mathbf{x}^0 of the location of the maximum. Note: I denote by superscript 0 the initial vector, its components are denoted by subscripts: $\mathbf{x}^0 = (x_1^0, x_2^0)$.
2. Compute the gradient $\nabla f(\mathbf{x}^0)$.
3. Now take a step in the direction of the gradient. This leads to a new location $\mathbf{x}^1 = \mathbf{x}^0 + R \cdot \nabla f(\mathbf{x}^0)$. Scalar R , *learning rate*, determines the length of the step. As the gradient is pointing uphill, the function value at \mathbf{x}^1 is larger than at \mathbf{x}^0 .
4. Now repeat the process choosing \mathbf{x}^1 as the starting point. This gives you the next approximation \mathbf{x}^2 .
5. Repeat until gradient is close to zero and the function value does not improve any more.

GA is similar to climbing a hill in **whiteout conditions** (or in total darkness). The ground is white, the sky is white, and you cannot see more than a step or two around you. How will you get to the top of the hill? Using GA! You start wherever you are (this is your \mathbf{x}^0). You feel which way the ground is rising (this is your gradient). Now you take a few steps in that direction (this is your \mathbf{x}^1). You repeat the process until you have reached flat ground. This is the hilltop.

Let's take a concrete 2-dimensional example. Let's make one step of GA for the function $f(\mathbf{x}) = x_1 \cdot \log x_2$. You can easily see its gradient is $\mathbf{g}(\mathbf{x}) = (\log x_2, x_1/x_2)'$.

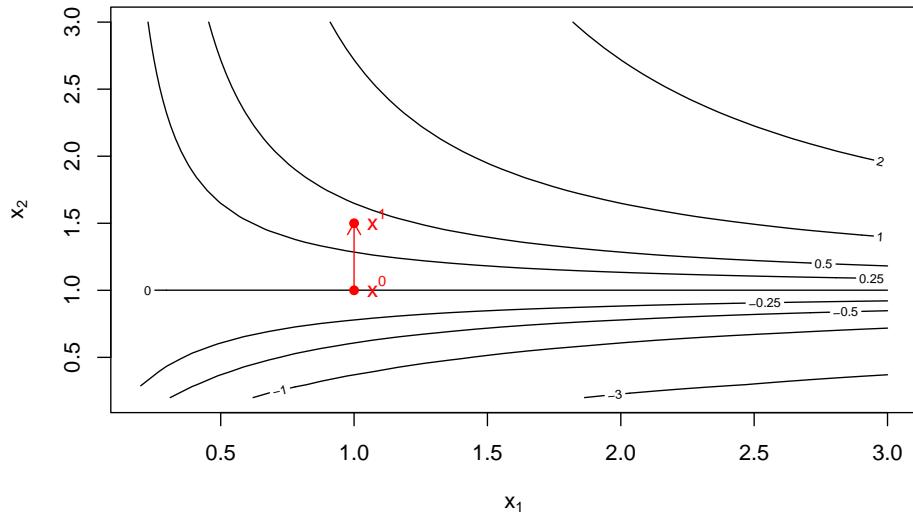


Figure 9.4: One Gradient Ascent step for function $f(\mathbf{x}) = x_1 \cdot \log x_2$. At the initial point $\mathbf{x}^0 = (1,1)'$, the gradient $(0,1)'$ points straight up. We move in that direction by the amount (learning rate) $R = 0.5$. This leads us to $(1,1.5)'$, our next approximation for the maximum.

1. Pick a starting point. Let's choose $\mathbf{x}^0 = (1,1)'$. The function value at that point is $f(\mathbf{x}^0) = 0$. See Figure 9.4.
2. The gradient at this point is obviously $\nabla f(\mathbf{x})|_{\mathbf{x}=(1,1)'} = (\log 1, 1/1)' = (0,1)'$.
3. Now take a step from \mathbf{x}^0 along the gradient. But first we have to choose the learning rate R . Let's pick $R = 0.5$. Now we have

$$\mathbf{x}^1 = \mathbf{x}^0 + R \cdot \nabla f(\mathbf{x}^1) = (1,1)' + 0.5 \cdot (0,1)' = (1,1.5)' \quad (9.2.1)$$

The new function value is $f((1,1.5)') = 1 \cdot \log 1.5 = 0.405$. Indeed, we moved uphill.

4. Now we can repeat the process until reach the maximum – although note that this particular function does not posess a maximum. So we stop here ☺

This particular example is illustrated on Figure 9.4 by contour plot. Note that at \mathbf{x}^0 , the gradient points straight up as the function is constant along x_1 at our initial point $(1,1)'$, and hence we move along the direction of steepest ascent at that point. However, as R is relatively large, the steepest ascent direction at \mathbf{x}^1 is slightly different.

Now let's describe the algorithm in more detail for the general n dimensional case.

1. Pick an initial value of the parameter $\mathbf{x}^0 = (x_1^0, x_2^0, \dots, x_n^0)'$. It is always good idea to choose parameters as close to the actual maximum as you can as this may have a huge impact on speed. (Hint: the current optimum is most likely close to the place where it was in your previous run!) Often though you have little guidance about how to choose good starting values.
2. Compute the gradient of your objective function $\nabla f(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}^0$.
3. Take a step from \mathbf{x}^0 in the gradient direction. The step should be neither too long (you may overshoot and land on the other side of the hill) nor too short (it takes too long time to find the maximum). So we employ the learning rate R and take the step of length $R \cdot \nabla f(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^0}$. This will land you in a new place we call \mathbf{x}^1 :

$$\mathbf{x}^1 = \mathbf{x}^0 + R \cdot \nabla f(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^0} \quad (9.2.2)$$

This is our new best bet for the location of maximum. It is probably not a perfect place, but unless your code is wrong (or the function constant at \mathbf{x}^0), it is a better place than \mathbf{x}^0 .

Note: here is the only point of difference between GA and GD algorithms. If we want to move downhill, we have to take a step to the direction of the steepest descent, i.e. a step to the *opposite to the gradient*, and hence the updating rule is $\mathbf{x}^1 = \mathbf{x}^0 - R \cdot \nabla f(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^0}$.

Choice of a good R value is important. This is a parameter of your model, although not one that is directly related to the process you are modeling. Such parameters are typically called *hyperparameters*. Unlike many other hyperparameters, such as choice of k for k -nearest neighbors, this one will probably not affect your final results, just the speed of convergence (and it may determine if your model will converge in the first place).

4. Are we in the correct place? There are several ways to check this:
 - (a) Gradient is very small. Say, $\|\nabla f(\mathbf{x}^1)\| < \epsilon^g$, where $\|\cdot\|$ is norm (length) of the vector, and ϵ^g is a small number, say 10^{-6} . Small gradient indicates that the function is flat at that point, and differentiable functions are flat at maximum.
 - (b) Function value does not grow much any more. Say, $|f(\mathbf{x}^1) - f(\mathbf{x}^0)| < \epsilon^f$ where ϵ^f is another small number.
 - (c) One may suggest more conditions, for instance about relative size of gradient.

Typically we only need one of the criteria above: if gradient is close to zero, the function stops growing, and vice versa.

These conditions are called *stopping criteria*. In practice, you always need an additional, bail-out criterion: stop if the process has been repeated too many times already. This is because we too often choose too small learning rate, run into numerical problems, or have coding errors.

5. If we are in correct place, stop here. If not, set $\mathbf{x}^0 \leftarrow \mathbf{x}^1$ and repeat from step 2.

Finally, a note about Gradient Ascent and Gradient Descent. Depending on the task, you may need to go downhill instead of uphill, for instance when finding the place of minimal loss. The GD algorithm is almost exactly the same as GA. The only exception is that you have to take the step toward steepest descent, not steepest ascent. This is just the opposite direction of gradient, and hence the update step would look like

$$\mathbf{x}^1 = \mathbf{x}^0 - R \cdot \nabla f(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^0} \quad (9.2.3)$$

(note the minus sign). There are two trivial ways to turn the GA problem into a GD problem or the way around:

1. flip the sign of your objective function: instead of minimizing $f(\mathbf{x})$, maximize $-f(\mathbf{x})$.
2. change the algorithm in a way to take a step into the opposite direction of the gradient. If you don't want to change your code, just use negative learning rate R .

TBD: SGD

9.3 OLS Example

Linear Regression “Least Squares” means

$$\min_{\boldsymbol{\beta}} L(\boldsymbol{\beta}) = \sum_i (\hat{y}_i - y_i)^2$$

where

$$\hat{y}_i = \hat{\boldsymbol{\beta}}' \mathbf{x}_i$$

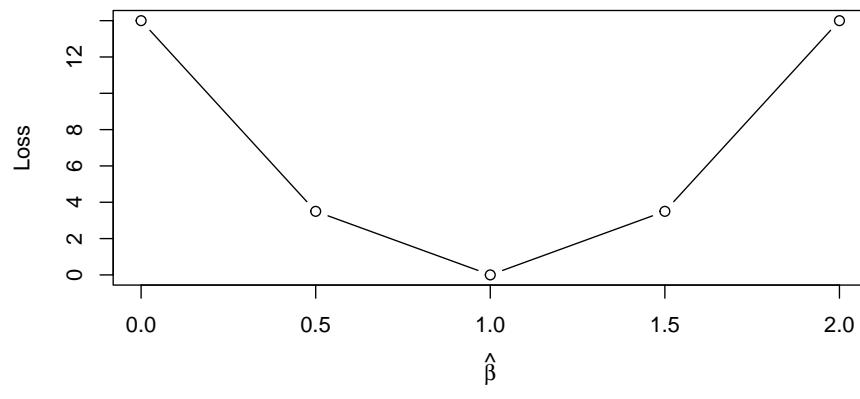
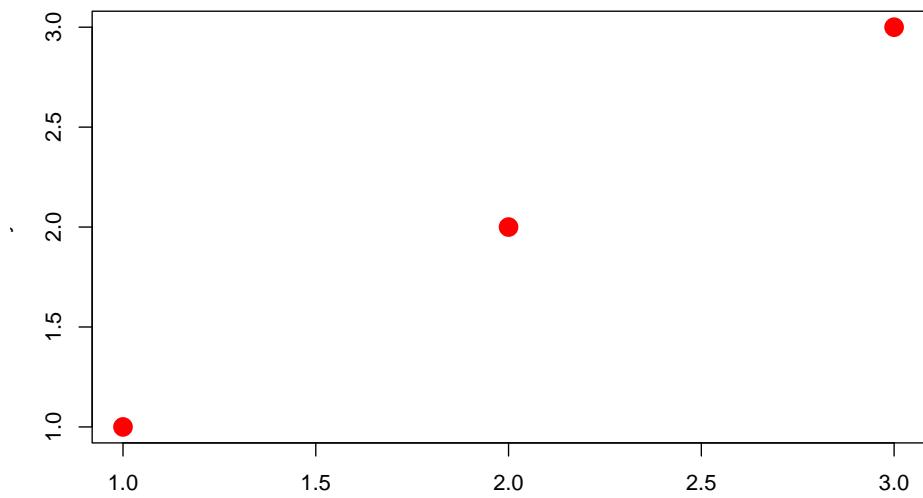
- find $\boldsymbol{\beta}$ that minimizes $L(\boldsymbol{\beta})$
 - L is “loss function” (objective function, cost function)
 - **how?**

Example: Predict September Arctic Sea Ice by March Extent
 Trial-and-Error Exercise
 (Grid Search)

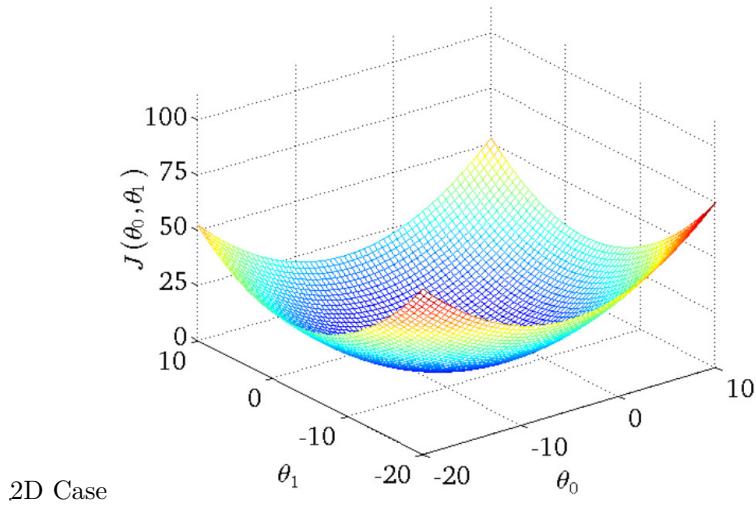
- OLS model $y_i = \beta x_i + \epsilon_i$
- Use the data at right
- Find the optimal β
 - Calculate $L(0), L(0.5), L(1), L(1.5), L(2)$.

- Plot $L(\beta)$ versus β

data:



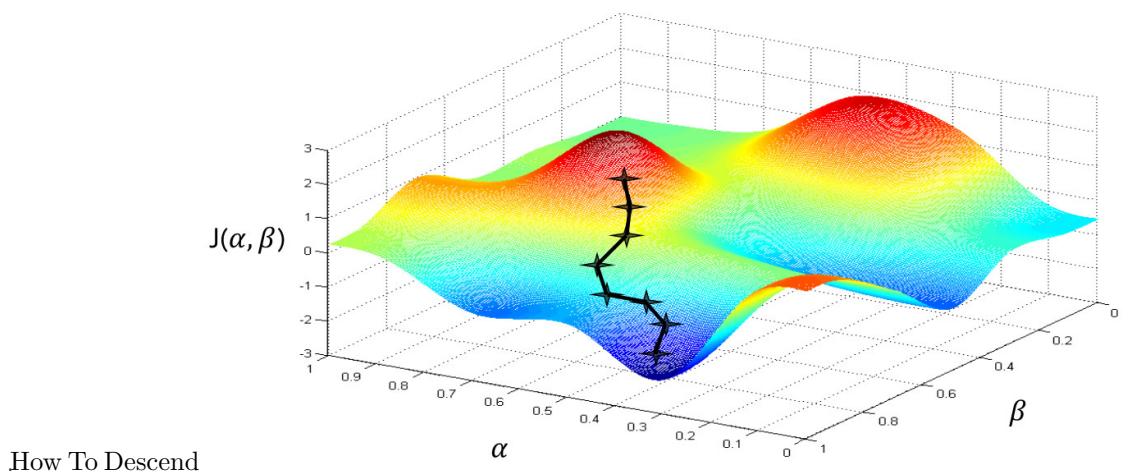
$L(\beta)$



9.4 Gradient Descent

How To Minimize $L(\beta)$?

1. Start with a β value
2. Calculate $L(\beta)$.
3. Change β so it decreases L
4. Repeat 2-4 until we are at minimum



9.4.1 What Is Gradient

What is Gradient Vector of first derivatives of the function with respect to it's arguments

- Direction where the function's growth is steepest
- “Speed” of growth
 - Per unit interval

Example:

$$f(\beta) = \beta^2 \Rightarrow \nabla f(\beta) = 2\beta$$

- positive if $\beta > 0$
 - $f(\cdot)$ grows when β grows
- negative if $\beta < 0$
 - $f(\cdot)$ grows when β decreases
- zero if $\beta = 0$
 - We are in a (local) optimum

Two-Dimensional Example

$$\begin{aligned} f(\beta_1, \beta_2) &= e^{-\beta_1^2 - \beta_2^2} \\ \frac{\partial f(\beta_1, \beta_2)}{\partial \beta_1} &= -2f(\beta_1, \beta_2)\beta_1 \\ \frac{\partial f(\beta_1, \beta_2)}{\partial \beta_2} &= -2f(\beta_1, \beta_2)\beta_2 \end{aligned}$$

In vector form

$$\frac{\partial f(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}'} = -2f(\boldsymbol{\beta})\boldsymbol{\beta}$$

Linear Regression Example Example of Linear Regression:

- In non-matrix form

$$\begin{aligned} L(\boldsymbol{\beta}) &= \sum_i (y_i - \beta_1 x_{1i} - \beta_2 x_{2i})^2 \\ \frac{\partial}{\partial \beta_1} L(\boldsymbol{\beta}) &= 2 \sum_i (y_i - \beta_1 x_{1i} - \beta_2 x_{2i}) \cdot x_{1i} \\ \frac{\partial}{\partial \beta_2} L(\boldsymbol{\beta}) &= 2 \sum_i (y_i - \beta_1 x_{1i} - \beta_2 x_{2i}) \cdot x_{2i} \end{aligned}$$

- In matrix form

$$\begin{aligned} L(\boldsymbol{\beta}) &= (\mathbf{y} - \boldsymbol{\beta}\mathbf{X})'(\mathbf{y} - \boldsymbol{\beta}\mathbf{X}) \\ \frac{\partial}{\partial \boldsymbol{\beta}} L(\boldsymbol{\beta}) &= 2(\mathbf{y} - \boldsymbol{\beta}\mathbf{X})'\mathbf{X} \end{aligned}$$

9.4.2 How to Optimize

How To Improve β Move in (the opposite) direction of gradient $\nabla f(\beta)$

- Gradient: in which direction the function grows most
- Climbing a snowy mountain in fog
- $-\nabla f(\beta)$: direction the function decreases most

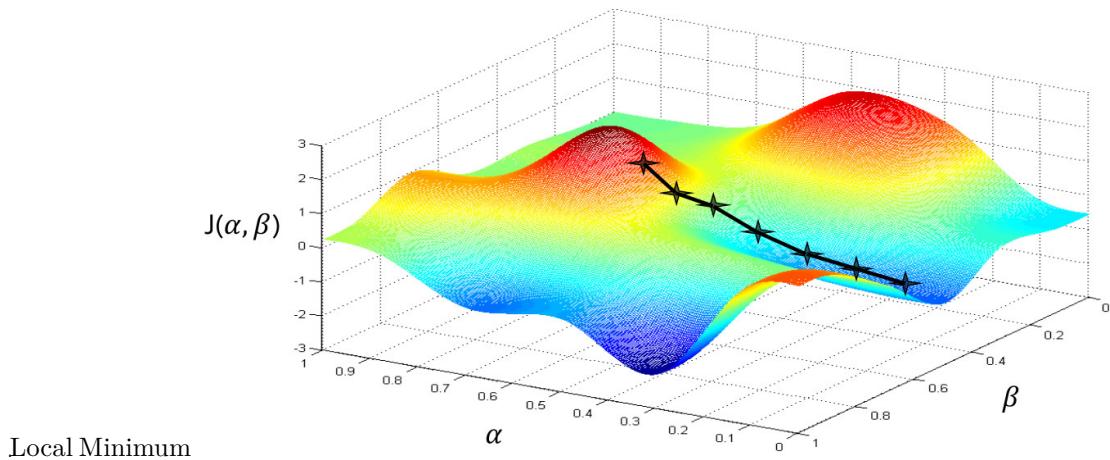
$$\beta_{n+1} = \beta_n - R \frac{\partial}{\partial \beta'} L(\beta)$$

- R : step size (learning rate)
 - Should be small
 - Can be made adaptive
 - Can be calculated
- Stop when gradient close to zero ...
- or when the objective function does not decrease any more
- or when too many iterations

Algorithm

1. Set $n = 0$ and β^0 to a value
 - $\beta^0 = \mathbf{0}$ is sometimes a good choice
2. Choose R (a small number)
3. Calculate $L(\beta^n)$
4. Calculate gradient $\nabla L(\beta^n)$
5. Is gradient close to $\mathbf{0}$?
 - Yes – stop
6. Calculate $\beta^{n+1} = \beta^n - R \cdot \nabla L(\beta)$
7. Calculate $L(\beta^{n+1})$
8. Did $L(\beta)$ decrease substantially?
 - No – stop
9. Is n too large?
 - Yes – stop
10. set $n := n + 1$, repeat 4

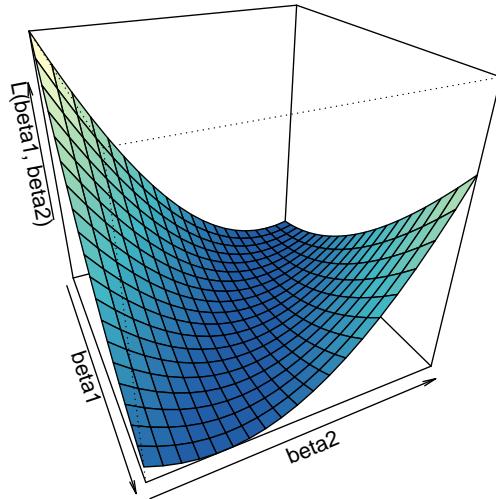
9.4.3 Problems

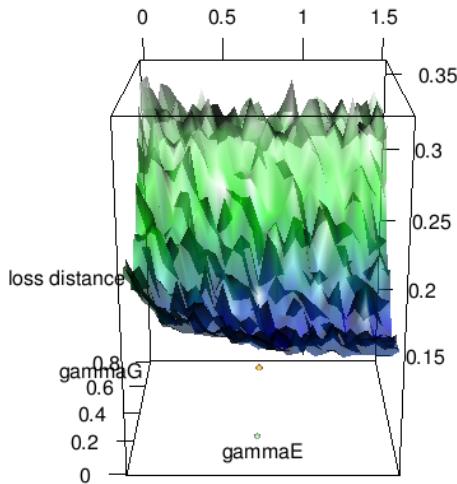


.Convexity Function $f(\mathbf{x})$ is convex iff:

$$\forall \mathbf{x}_1, \mathbf{x}_2, \quad t \in (0,1)$$

$$\begin{aligned} f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) &< \\ &< tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2) \end{aligned}$$





Noisy Objective Function

9.5 Key Concepts

Key Concepts

- Cost Function (Loss Function)
- Non-Linear Optimization
- Gradient Descent
- Local/Global minima
- Convexity
- Learning Rate
- Feature Scaling

9.5.1 Resources

- Khan Academy's "Why gradient is the direction of steepest ascent" <https://www.khanacademy.org/math/multivariable-calculus/multivariable-derivatives/gradient-and-directional-derivatives/v/why-the-gradient-is-the-direction-of-steepest-ascent>

9.6 Feature Selection and Regularization

9.6.1 Feature Selection

In applied analysis it is quite common to have datasets with a large number of features. Often we have little knowledge about the importance of many of these. Many of these may be highly correlated and many can show certain importance in our models. For instance, when using a large international survey data, such as World Value Survey, one may find the variable *country* is highly correlated with *domestic language*, language used to fill out the survey, and the id of the team member. It may also be somewhat unexpectedly related to certain lifestyle questions, e.g. there are probably very few affirmative answers to the question “do you have a boat” in an arid landlocked area. Such closely correlated variables may cause various problems with data modeling. To name a few

- Large and unstable parameter values and large standard errors. This is mainly a problem for inferential modeling and may obscures the interesting effects that are in fact there.
- Overfitting. This is how the same issue manifests in predictive modeling.
- Model does not converge, or converges into a sub-optimal solution. While linear regression (almost) always works well, more complex model are much more demanding in terms of data properties. Even if data looks good globally, we may run into a trouble in a region where the correlation is high.

In case of a smaller well-documented dataset, it may be possible to manually select the interesting features. But this approach does not scale to larger datasets where we have thousands of similar variables we do not understand well. For instance, imagine analyzing urban movements using millions of cellphone calls, or doing sports analytics with thousands of datapoints about athletes’ movements. In such cases it is not obvious what to include or exclude in the model.

Feature selection is a method (more like a set of several methods) that helps to include only the “best” features in the model. It is in some ways similar to *regularization*, a method that does not directly select features but manipulates the model parameters and may achieve a comparable effect.

Consider the following example. We generate one variable $\mathbf{x} \sim N(0,1)$ and form a number of other highly correlated variables:

$$\begin{aligned}\mathbf{x} &\sim N(0,1) \\ \mathbf{x}_1 &= \mathbf{x} + \epsilon \\ \mathbf{x}_2 &= \mathbf{x} + \epsilon \\ &\dots\end{aligned}$$

where $\epsilon \sim_{i.i.d} N(0, 0.1)$. We generate the outcome y as $y = \mathbf{x} + \mathbf{u}$, where $\mathbf{u} \sim N(0, 0.3)$. Thereafter we estimate linear regression model in the form

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + e_i$$

i.e. we model \mathbf{y} using a number of highly correlated variables. Importantly, we keep the number of cases very low, the example below is made for $N = 12$ training data observations and 7 different \mathbf{x} vectors.

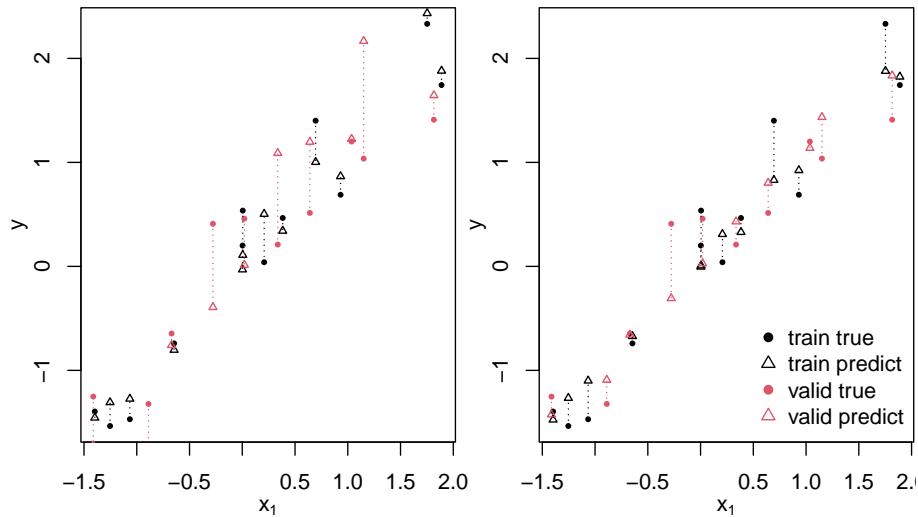


Figure 9.5: Linear regression (left panel) versus ridge regression (right panel). Solid dots represent data and empty triangles are predictions. Black is training and red testing data. Linear regression make much more noisy predictions than regularized ridge regression. There are 6 other highly correlated features not visible in this figure.

Figure 9.5 shows an artificial example with 7 highly correlated predictors. Solid black are the training data and red are validation data, solid dots represent training and empty triangles testing data, and the dotted lines between those are residual errors. The left figure depicts the linear regression results. One can see that errors in training data are mostly small but for testing data the errors are much larger, in particular for data points that are far from training observations. The fact that the model behaves much better on training than on testing data is also confirmed by the corresponding R^2 values, 0.957 and 0.541 respectively.

The right panel show results for a penalized ridge regression. Ridge suppresses the noise from highly correlated variables and correctly finds that all predictors contain essentially the same information. Hence the predictions (triangles) form almost perfect line on the figure. The corresponding R^2 values are 0.934 and 0.853. The flexibility in the linear model largely captured the noise, making the model less flexible forced it to focus on the signal instead.

9.6.2 Regularization

More complex models we use on large datasets are often very flexible. This flexibility may easily lead to enormous overfitting, and technical problems, such as lack of convergence.

Regularization is a simple way to force flexible models to be less flexible in order to improve their performance on unseen data. It can be done in various ways like

- by adding a penalty term to the objective function
- by using a Bayesian prior over the parameter values
- by terminating iterative optimization (such as stochastic gradient descent) early

One can show that under certain assumptions, all these methods produce similar results.

Chapter 10

Unsupervised Learning

10.1 Introduction

In the previous sections we were working with supervised learning. In case of supervised learning, our task is to predict outcome y based on features \boldsymbol{x} . As we have labeled training data, we can tell the algorithm for each case how “far off” was the prediction from the true value. Later we use the trained model to do similar predictions on unlabeled data. As we have analyzed the prediction errors on training data, we have some confidence to assume the errors are similar on unknown data. Formally, our task is to estimate the function $f : X \rightarrow Y$ where X is the feature space and Y is the target space. A good result is such a function where the predicted value $\hat{y} = f(\boldsymbol{x}_i)$ is close to the true value y_i .

Unsupervised learning is the case where we don’t know the “correct” labels y_i and hence we cannot tell if our predictions are close or not to the true values. Even more, there is often no such things as true labels, cases can be categorized in many different ways and all of these may be correct in some sense.

Instead of trying to predict “correct” values that we do not know or that may even not exist, unsupervised learning is used to discover and exploit various traits in data. The common examples include:

- Cluster analysis: we group data into clusters, groups of cases that are reasonably similar to each other while being different from cases in other clusters. A small number of such clusters can thereafter be used as data simplification. We may use a single “representative” in each cluster instead of individual values and in this way to tremendously reduce the complexity of data.

For instance, the consumers can be categorized into a small number of clusters, and afterwards one may design a different marketing strategy for each cluster. It may be infeasible to have a large number of such strategies but unsupervised learning helps us to reduce the complexity to a manageable number.

- Principal component analysis: we analyze which kind of values tend to occur together in the data. This allows us to find certain combination of features, principal components, that carry most of the information. The principal components may give us novel insight into the problem, but it also allows to remove the less important traits and simplify the data in this way.
- Market basket analysis: as in PCA, we attempt to find values that tend to occur together. However, our task will be to construct claims like “consumers who bought x usually also buy y .
- Also usually not considered as unsupervised learning, various descriptive graphs and tables play a similar role. They help us to discover the traits in the data, their limits, and structure.

10.2 Cluster Analysis

Prerequisites: Metric distance, vector norm 2.2.2,

Cluster analysis is a way to partition data points into a number of groups, “clusters”. We want the data in cluster to be similar in some sense while data in different clusters may differ.

First we discuss the basic idea of cluster analysis and thereafter introduce perhaps the most popular clustering algorithm, k -means.

10.2.1 Idea

Cluster analysis attempts to find natural groupings in the data. As it is an unsupervised learning method, we do not normally provide it with any pre-defined grouping information. All this will be derived directly from data. This also applies to the number of clusters—we normally do not know what is the correct number. Even more, there may not be anything like the “correct number”, one can look at the data in different ways, just some of the approaches may be more insightful regarding the current problem. The key of deciding which observation goes to which cluster is similarity—observations in the same cluster should be more similar than observations in different clusters. There are many ways of deciding which cases are more similar.

- market segmenting: find groups of customers interested in certain type of products
 - we want to use a small number of marketing strategies
- market segmentation: find voters, susceptible for a certain type of propaganda
- land use analysis/urban planning



Figure 10.1: Original data (left) and cluster centers (right). This artificial dataset contains five clearly separated groups and hence it is exceptionally well suited for cluster analysis. The left panel shows the original data, the right panel the five cluster centers as computed by k-means. Note that the ordering of clusters is different.

- group neighborhoods from crime perspective → policing
- diagnose illness based on symptoms

10.2.2 Cluster Analysis More Generally

In order to split data into clusters we need four things: suitable data, distance metric, loss function, and an algorithm that can actually compute the clusters.

First, we obviously need data. In the example below we imagine data as points on the 2-D x - y -plane, but in general these are in high-dimensional space and cannot be easily visualized. Normally we imagine data in a form of a numeric design matrix but in certain cases it may also be in a different format. (For instance, one can compute string distance between words, and in that case the data may be in the form of character strings.)

Second, in order to measure similarity, we need something like distance metric (see [metric distance](#) in Section 2.2.2). If the design matrix is numeric, we may rely on Euclidean or other L_p type metrics, but we may also carve out our own dedicated metric. For instance, when comparing portraits, we may want to design a metric that only looks at the faces and ignores the background. In case of more than a single feature, we also have to weight the features somehow, i.e. feature scaling matters (see [Data-Driven Metrics](#) in Section 3.2.1).

If we want to compare different ways to split partition data into clusters we also need a measure of how good or bad the result is. We may do this by

defining a per-cluster loss function (see Section 9.1 Loss Function). Let's write the loss function for cluster C as $L(C)$. The cluster C should be understood as a set of data points: we can compute the loss as soon we know which observations belong to it. The loss function typically penalizes intra-cluster distance as we would prefer all member points to be close to each other.

Finally, if we want to compute the clustering, not just to compute the “badness” (loss) of the result, we also need an algorithm that can find a good set of clusters based on the data, distance, and loss we selected above. It should try different ways to put data into clusters C_1, C_2, \dots and pick such an arrangement that produces minimal loss (it should minimize the loss function). Ideally it should find the smallest possible loss but if this is not feasible, a good enough solution may do. The algorithm should return the partition—which observations go to which cluster. Formally:

$$\{C_1, C_2, \dots, C_K\} = \arg \min_{C_1, C_2, \dots, C_K} \sum_{k=1}^K L(C_k).$$

Unfortunately there are typically way too many possible ways to partition data into clusters, so it is in general not possible to find the best way. But there are many algorithms that typically work well enough and provide a good solutions, that may not be the best though.

TBD: Some sort of example, perhaps 1-D k-means

10.2.3 k -Means Clustering

k -means is one of the simplest and most popular clustering algorithms. It is intuitive, fast, and always provides a solution, although the solution may sometimes be suboptimal.

When one chooses in-cluster variance as the loss measure, we have

$$L(C) = \frac{1}{||C||} \sum_{i', i \in C} (\mathbf{x}_i - \mathbf{x}_{i'})' (\mathbf{x}_i - \mathbf{x}_{i'})$$

where $||C||$ is number of observations in the cluster (see Section 0.1, Sets). Hence the total loss

$$\min_{C_1, C_2, \dots, C_k} \sum_{i=1}^k \frac{1}{||C_i||} \sum_{j', j \in C_i} (\mathbf{x}_j - \mathbf{x}_{j'})' (\mathbf{x}_j - \mathbf{x}_{j'})$$

k -means partitions data into pre-specified k clusters where cluster membership is mutually exclusive—each data point belongs to one and only one cluster. The cluster membership is determined based on the distance between the data point and cluster centers, and the algorithm repeatedly re-assigns the observations to the closest cluster, and thereafter re-computes the cluster centers. These two steps are computed repeatedly until it results in a stable partition—each observation belongs to its closest cluster. It may sound somewhat surprising that such a simple idea works very well, but in most cases it does.

Next, we explain the algorithm in more detail and provide an example.

The k -means algorithm

1. Select the desired number of clusters, k . This must be decided before the algorithm starts.
2. Next, we need to find a *centroid*¹ for each of the k clusters. We can do this in various ways, for instance we can pick a random data point as the centroid for each of the clusters (just pay attention to that each cluster should get a *different* data vector as its centroid).
3. Now assign each actual data point to the cluster with the closest centroid. This immediately causes the cluster partition to clear up as more similar observations tend to fall into the same cluster. As a result we now know for each observation which cluster does it belong to.
Note that “closest” assumes we have decided for a distance metric, in case of k -means we normally use Euclidean distance.
4. Now we compute new cluster centroids by just averaging the data vector components for each cluster.
5. And now we just repeat from 3 until the partition converges, i.e. there are no more changes in the partition $\{C_1, C_2, \dots, C_K\}$.

The algorithm works surprisingly well and always produces a result. Figure 10.2 illustrates how the algorithm works in a simple case. However, sometimes the result may be suboptimal, so it is advisable to run the k -means algorithm several times with different random starting points.

TBD: k-means gets stuck, perhaps as an exercise

Determining the number of clusters

k -means expects the user to provide the required number of clusters. This is easy in case we know enough about the underlying data structure, but sometimes we need more guidance from the algorithm itself. A popular way to find the “best” number of clusters is by using *elbow plot*. The idea of the elbow plot is the following: we allocate the data points to clusters by minimizing the sum of squared errors (or another loss function) within the clusters. In case we choose too few clusters, we have many mis-allocated points and hence the loss is large. But as soon as we pick the correct number of clusters, the loss should fall substantially. Increasing k even further will not substantially change the loss. So one expects to see a kink, the “elbow” on the plot at the correct value of k .

Figure 10.3 displays such a clear kink at $k = 5$. This is the same data as depicted on Figure 10.1. That artificial dataset is extremely well suited for cluster analysis. However, when we move to typical real datasets, the kinks may

¹*Centroid* is similar to average or mean value, just in case of multi-dimensional objects (like data vectors) we call the average “centroid”. You can easily visualize it as the “middle point” of a point cloud.

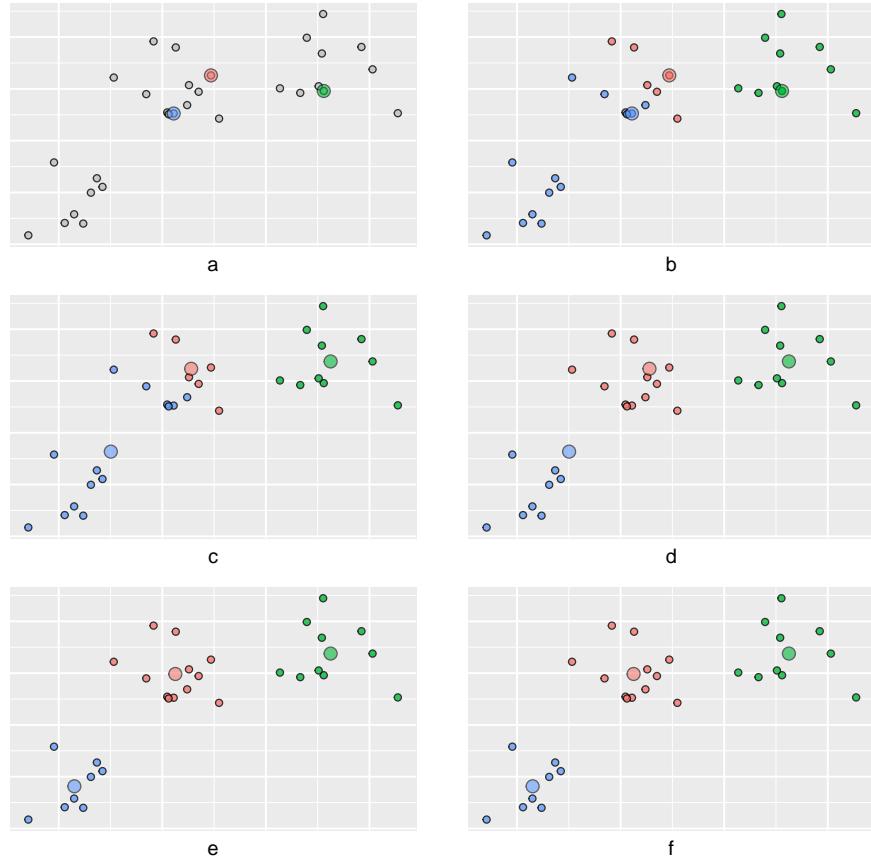


Figure 10.2: k -means algorithm at work. a) Top-left panel shows the plain data with no cluster information, however, the random data points are picked as cluster centers (denoted by color). b) Top-right panel has all data points colored according to the closest random cluster. Already at this stage the k -means algorithm starts to separate data into different clusters. c) Mid-left panel shows the updated cluster centers: based on the previous image, the new cluster centers are centroids of the points that belong to the same cluster (same color). d) We update clusters (colors) again based on the closest cluster center. Now all of the middle blob belongs to a single cluster. e) One more update of cluster centers will position the red and blue cluster center in the middle of the respective clusters. f) Next update of clusters (colors) does not change anything. The process has converged.

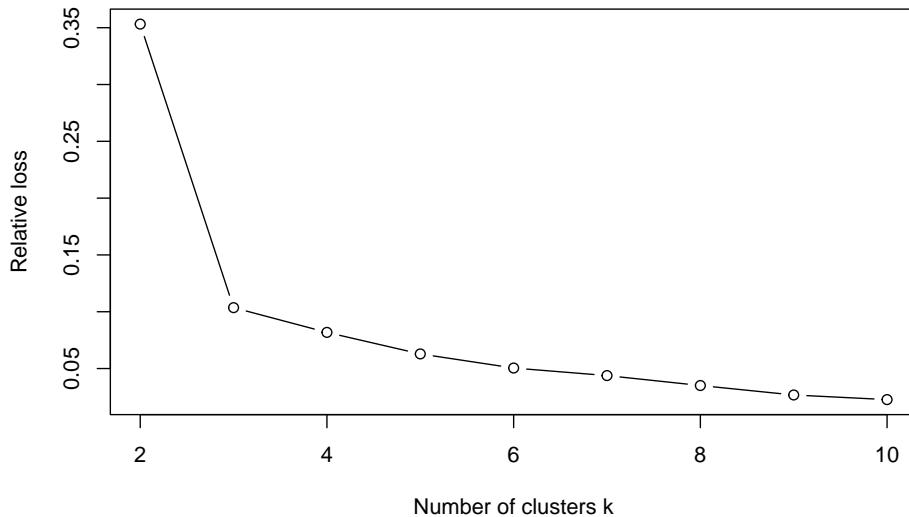


Figure 10.3: Elbow plot for the same data as used in Figure 10.1. The vertical axis denotes the relative loss, within-cluster loss as a percentage of total loss. One can see that at $k = 5$, the relative loss reaches essentially zero. This is the ‘‘elbow’’ of the plot and the best number of the clusters.

be much more vague, or completely missing. Figure 10.4 depicts a similar elbow plot for diamonds data. As the data (left panel) do not display any clear cluster structure, the relative loss on the elbow plot (right panel) keeps getting smaller even when we add clusters. The apparent kink at $k = 3$ does not correspond to any clearly distinct clusters (figures on the left panel). The clusters are probably not a useful way to think about this data.

10.3 Principal Component Analysis

Principal Component Analysis (PCA) is another popular method to find patterns in data. It is in some way similar to cluster analysis, but unlike the latter, PCA is not concerned about points located close in space, but rather about points placed near hyperplanes in hyperspace. It is used in a wide variety of applications, including exploring and analyzing correlated features, designing new features, and compressing data.

10.3.1 Motivation

There are several motivations that lead to more-or less similar concept of PCA. We list here three problems, the first more a social science problem, and two other more technical.

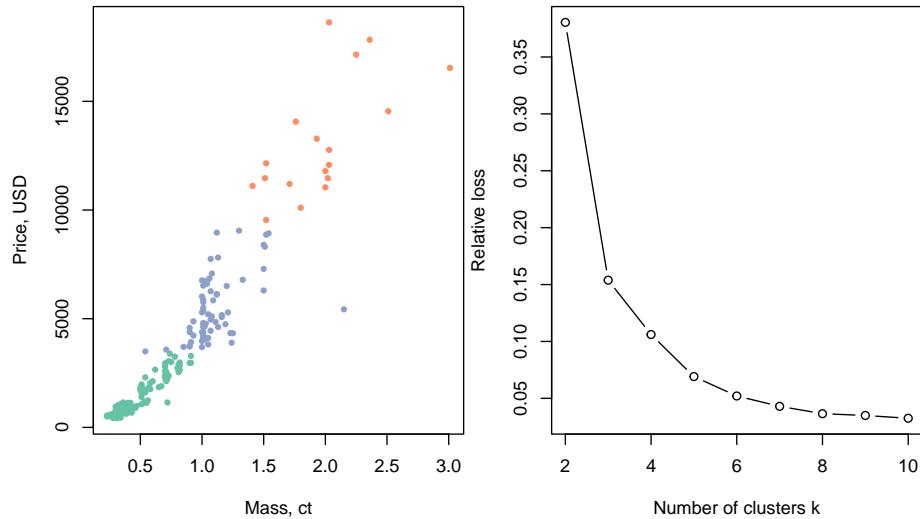


Figure 10.4: Elbow plot for diamonds data. The left panel depicts the diamonds data, the data points are colored according to the detected cluster id. The right panel shows the elbow plot, if anything it suggests $k = 3$ is the optimal number of clusters. However, as the left figure indicates, the detected clusters are rather indistinct. This data is not well suited for clustering.

1. How to measure vague concepts? If we are interested in measures like income or age, the measurement is easy. Pretty much everyone knows their age and people have fairly good understanding what income is (even though they may not know their income, or may be unwilling to tell). But how liberal are you? Or how extrovert are you? The respondents may have an idea in both cases and may be willing to answer something like “rather not liberal” or “fairly extrovert”. But now we are measuring their idea about their liberalism (called *perceived liberalism*) and not how liberal they actually are.

Instead of asking about liberalism directly, the survey typically ask for a number of questions that we think are closely related to liberalism. For instance, one may ask

1. Do you support gun rights?
2. Do you support free abortion?
3. Should the government take more care of the environment?
4. Is cross-border crime the most serious threat nowadays?

As people traditionally understand the concept “liberalism”, we may want to add the second and third answer with a positive weight, and the first and the fourth answer with negative weight. But is this the correct approach? And what should the weights be? Is “liberalism” even a useful concept on this data, if the answers to these questions are pretty much just random then we perhaps do not really have liberals and conservatives in the first place?

2. How do we aggregate similar measures? The second motivation is quite similar, just originate from the technical world. Imagine you have a number of similar measure—similar but not precisely the same. For instance, you are working with natural disaster data and you want an estimate of the destruction hurricanes are causing. But FEMA² does not provide a single number of “destructiveness” of a hurricane. Instead it lists a number of different costs:

- Total Individual Assistance (IA) - Applications Approved
- Total Individual and Households Program - Dollars Approved
- Total Housing Assistance - Dollars Approved
- Total Other Needs Assistance - Dollars Approved
- Total Public Assistance Grants - Dollars Obligated
- Dollars obligated to emergency work
- Dollars obligated to permanent work

These numbers are clearly related to the destruction—more destruction probably means all these numbers are larger. But if I need a single number then which one should I take? Or should I take the average? But does average over number of applications and dollars approved and obligated make any sense?

3. How to get rid of a lot of redundant data Finally, there is a data compression problem. Imagine you are collecting cellphone data over time and geographic districts. For each district and each time period you record

- Total number of outgoing phone calls
- Total number of incoming phone calls
- Total number of text messages
- Total number of multimedia messages
- Total number of data connections
- Total seconds of outgoing phone calls
- Total GB of data transfer
- ...

Obviously, all these numbers are highly correlated. A busy afternoon in a large city has all these figure up in millions while there is hardly anything in a tiny rural place in the middle of night. Do we really have to store and analyze all these numbers? Can we only keep one of these and drop the others? But which one? Or should we take average again? But does average over counts, seconds and GB-s even make sense?

All of these tasks are different sides of the same problem: we have a lot of correlated data and we are looking ways to simplify and understand it. While in social sciences the understanding part has been traditionally in the focus, in technical fields it is more often simplification that we are looking for. But in all cases we are looking for fewer dimensions: in the first example we want to reduce four answers into a single liberalism measure, in the second example we try to reduce seven different cost measures to a “destructiveness” measure, and

²The U.S. Federal Emergency Management Agency

in the final example we may want to come up with 1-2 numbers that capture the “cellphone activity”.

There are many applications where one may want to collapse a large number of dimensions into a smaller more manageable numbers:

- Genome data: there is a tremendous numbers of parameters to measure genes
- Document and image classification: the algorithm may come up with hundreds of different categories, and we are just interested in a handful
- Product recommendation: as above, we may only be interested in a small number of product categories, not in thousands.

In a similar fashion, there are numerous reasons why we want fewer dimension.

- Curse of dimensionality
- High K - low N
 - run out of degrees of freedom
- Overfitting
- Visualization
 - Hard to do for more than 3 dimensions
- Interpretation
 - High-dimensional variation hard to interpret (understand)
 - * Collapse msisdns, calls, active firms.. into “activity”
- Statistical
 - Better statistical properties
- Visualization
 - Visualize
 - Understand
- Computation
 - Compress data
 - Simplify computation
- Look at answers to *all relevant* question
- Group the questions with similar answers to factors or principal components
- Look at factor loadings

- Large loading \Rightarrow important variable
- Factors denote the relevant concepts
 - If you are lucky ☺
- Data compression

Example 10.3.1: PCA with 2-D data

The easiest way to get an intuitive understanding of PCA is to use 2-D highly correlated data. Figure 10.5 below shows a such synthetic dataset.

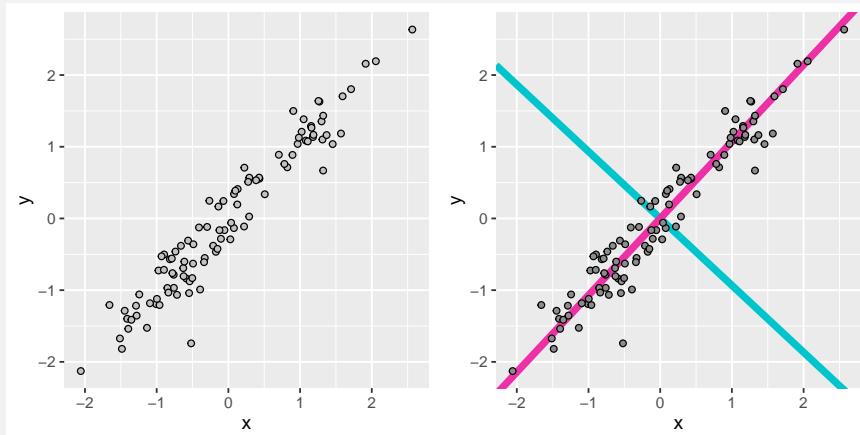


Figure 10.5: Highly correlated data that is perfectly suited for PCA. The left panel depicts only the datapoints, the right panel adds the principal components (PC1 red and PC2 blue).

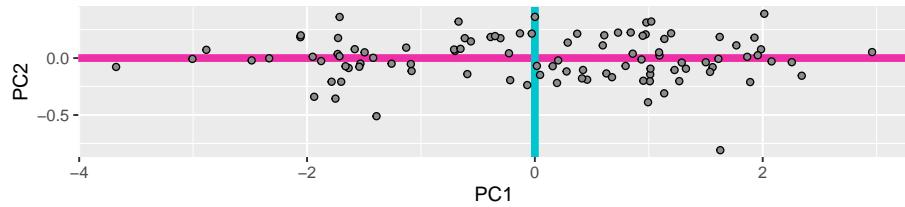
In this data, x and y are highly correlated, one can imagine the image as the dots lying on the 45° line, with a small perturbations that push them up or down off the line. This is indeed how PCA looks at the image (right panel).

Components:

```
## Importance of components:
##                 PC1     PC2
## Standard deviation    1.430 0.1936
## Proportion of Variance 0.982 0.0180
## Cumulative Proportion 0.982 1.0000
```

Rotation:

```
##          PC1         PC2
## x -0.6822721 -0.7310983
## y -0.7310983  0.6822721
```



variable	PC_1	PC_2
individual applications approved	-0.203	-0.200
approved individual/household total	-0.402	-0.414
approved housing assistance	-0.397	-0.409
approved other assistance	-0.353	-0.365
obligated to public assistance total	-0.426	0.427
obligated to emergency work	-0.400	0.385
obligated to permanent work	-0.413	0.393

- Note: no distinction b/w x and y
- Don't attempt OLS...

Data:

$$\mathbf{M} = \begin{pmatrix} x & y \\ 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{pmatrix}$$

Now

$$\mathbf{M}'\mathbf{M} = \begin{pmatrix} 30 & 28 \\ 28 & 30 \end{pmatrix}$$

(Note: $\mathbf{M}\mathbf{e} = \lambda\mathbf{e}$) Solve for eigenvalues:

$$|\mathbf{M}'\mathbf{M}| = (30 - \lambda)(30 - \lambda) - 28^2 = 0$$

The solution:

$$\lambda_1 = 58 \quad \lambda_2 = 2 \quad (10.3.1)$$

The corresponding eigenvectors:

$$\begin{pmatrix} 30 & 28 \\ 28 & 30 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix} \quad (10.3.2)$$

and we have

$$\mathbf{e}_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \quad \mathbf{e}_2 = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

The eigenvector matrix

$$\mathbf{E} = \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$$

Is a rotation matrix for angle $\cos \phi = 1/\sqrt{2}$ or 45° .

Rotated data:

$$\mathbf{ME} = \begin{pmatrix} x & y \\ 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 3/\sqrt{2} & 1/\sqrt{2} \\ 3/\sqrt{2} & -1/\sqrt{2} \\ 7/\sqrt{2} & 1/\sqrt{2} \\ 7/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}$$

- Eigenvalue decomposition rotates data matrix
 - Orthonormal eigenvectors are the new base
- The largest eigenvalue corresponds to the most important dimension

10.3.2 Principal Component Regression

One widely used application of PCA is in the regression analysis. One can use the principal components instead of the original variables. This may give two advantages:

- Sometimes the principal components have clear interpretation, and hence the resulting coefficients have more meaningful interpretation than when using the original variables.
- Often the less important principal components add little value to the regression, so we can ignore those and get a simpler model.

Note that PC regression may not give any gains if the components that describe little variance in data still describe a lot of variance in the target variable.

Example 10.3.2: Principal component regression with 2-D data

Here we demonstrate principal component regression using 2-D data. Consider the data below:

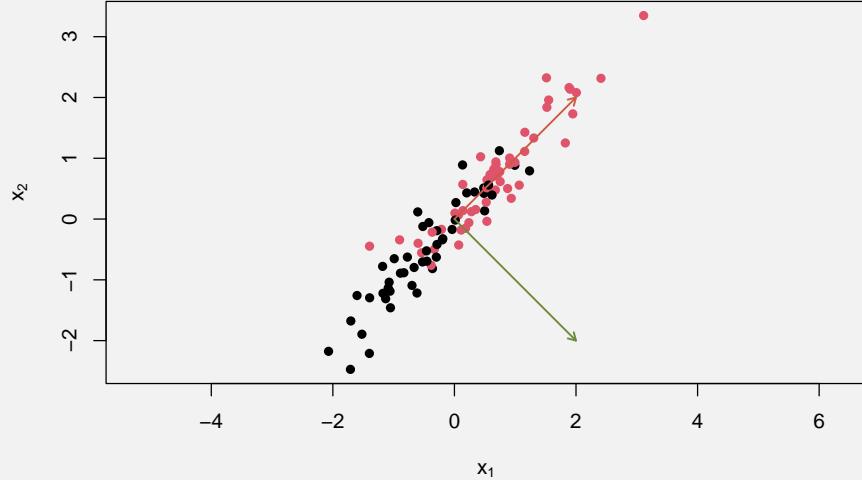


Figure 10.6: Narrow band of data points where color is associated with the bottom-left–top-right direction. This does not correspond exactly to any of the features x_1 and x_2 , but instead to their linear combination (rotated data). The principal components are depicted as arrows, with PC1 (orange) describing the direction maximum variation, and PC2 (dark green) the perpendicular direction.

The data contains two numeric features, x_1 and x_2 , and a color label, “black” or “red”. Our task is to predict the color based on x_1 and x_2 .

It is easy to see that red dots dominate in the top-right corner of the figure. We can use a simple logistic regression model

$$\Pr(\text{color}_i = \text{red}) = \Lambda(\beta_0 + \beta_1 x_1 + \beta_2 x_2). \quad (10.3.3)$$

The results are

	Estimate	Std. Error	z value	$\Pr(> z)$
(Intercept)	-0.0177	0.2550	-0.07	0.9445
x_1	1.0210	0.8682	1.18	0.2396
x_2	0.8609	0.8373	1.03	0.3038

In a counterintuitive fashion, neither x_1 nor x_2 show much significance here while we can clearly see that the red dots are clustered in the top-right corner. Even more, the point estimate for x_2 is negative although these are larger values that are associated with red color. The problem here is the fact that both features contain essentially the same information, and hence the design matrix is ill conditioned. The accuracy on training data, 0.78, is

acceptable though for such a noisy image. We can cure the problem with principal component analysis. Doing PCA on the data matrix gives us

Table 10.1: Principal components of data in Figure 10.6

	PC1	PC2
x1	-0.71	0.71
x2	-0.71	-0.71

Table 10.2: Proportion of variance explained by components in Table 10.1

	1	2
variance	1.95	0.05
proportion	0.98	0.02
cumulative	0.98	1.00

As is evident from the tables, $PC1$ loads both x_1 and x_2 of equal amount and hence points to North-East, while $PC2$ contains a positive quantity of x_1 and a negative quantity of x_2 , and hence points South-East (see Figure 10.6). As $PC2$ very little information (its contains only 2% of the total variance), we can drop $PC2$ and use a simpler model

$$\Pr(\text{color}_i = \text{red}) = \Lambda(\beta_0 + \beta_1 \cdot PC1). \quad (10.3.4)$$

The results are as follows:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.2123	0.2548	0.83	0.4046
PC1	-1.3707	0.2798	-4.90	0.0000

The results show that $PC1$ is now highly significant and of reasonable size. Accuracy on training data is the same, 0.78. But we got a simpler, model with more easily interpretable model.

Example 10.3.3: How are conservative family values and identity related to willingness to do good for society?

The worldview of people can be described with different dimensions, including family values (conservative versus liberal), and identity (global versus local), and many other. But how are these two value sets associated with the willingness to contribute to the society? Let's analyze this based on the World Value Survey, a large world-wide opinion survey.

We estimate a linear regression model in the form

$$\text{contribute}_i = \alpha + \boldsymbol{\beta}^\top \cdot \text{globalist values}_i + \boldsymbol{\gamma}^\top \cdot \text{gender values}_i + \epsilon_i \quad (10.3.5)$$

where *family values* is a vector of family-values related opinion, and *democratic values* is a vector of authoritarianism-democracy related viewpoints. The examples of family values include^a

doingGoodImportant It is important to do something for good of society.
trustUN how much confidence do you have in United Nations?

worldCitizen I see myself as a world citizen.

partOfNation I see myself as part of the nation.
maleLeaders men make better political leaders than women
maleExecutives men make better business executives than women
collegeBoy university education is more important for a boy than for a girl
When we run such a regression, we get the results

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-1.7280	0.0239	-72.24	0.0000
trustUN	0.0021	0.0048	0.44	0.6569
worldCitizen	0.1433	0.0053	26.81	0.0000
partOfNation	0.2503	0.0073	34.37	0.0000
maleLeaders	0.0772	0.0059	13.06	0.0000
maleExecutives	0.0342	0.0063	5.45	0.0000
collegeBoy	-0.0767	0.0057	-13.52	0.0000

The results suggest that those with stronger both global and national identity feel it more important to be good for society. The same is true for those who think men make better leaders, but not for those who believe higher education matters more for boys. The predictive power of the model is low with $R^2 = 0.041$.

Instead of estimating the effect of responses to individual questions, we can combine the answers into principal components, and include those in the regression instead. The principal components are

	PC1	PC2	PC3	PC4	PC5	PC6
trustUN	0.06	-0.43	-0.82	0.37	0.04	-0.01
worldCitizen	0.02	-0.67	0.05	-0.70	0.25	0.02
partOfNation	-0.06	-0.60	0.53	0.50	-0.32	-0.01
maleLeaders	-0.59	-0.01	0.06	0.16	0.50	-0.61
maleExecutives	-0.61	-0.00	-0.01	0.08	0.19	0.76
collegeBoy	-0.52	0.00	-0.21	-0.31	-0.74	-0.21

The first two components are easy to interpret: PC_1 loads strongly with all three variables that describe the conservative gender roles, hence a large PC_1 value describes liberal gender values (as the loadings are negative). PC_2 loads on the globalist/national feelings and describes someone who does not trust UN and does not feel any attachment neither to the world nor her nation.

The importance of the components is

	1	2	3	4	5	6
variance	2.01	1.29	0.96	0.78	0.58	0.37
proportion	0.34	0.22	0.16	0.13	0.10	0.06
cumulative	0.34	0.55	0.71	0.84	0.94	1.00

Let's re-run the regression using the two first principal components only. Together these describe over 50% of the variance, and third component is also much harder to interpret. So we estimate a regression model

$$\text{contribute}_i = \beta_0 + \beta_1 \cdot PC1_i + \beta_2 \cdot PC2_i + \epsilon_i. \quad (10.3.6)$$

The results are

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-2.4423	0.0046	-531.80	0.0000
PC1	-0.0349	0.0032	-10.79	0.0000
PC2	-0.1854	0.0040	-45.87	0.0000

The results indicate that $PC1$ —individuals who have more liberal viewpoints about the gender roles—are less likely to think it is important to do something good for the society. But the effect of $PC2$ is much stronger—those who do not trust UN and do not feel belonging to a group do not think it is important to do good. The effect of the latter component is much stronger than that of the former, suggesting that feeling of attachment and identity is much more important factor in determining someone's willingness to contribute to the public good. The model's explanatory power is weak though, with $R^2 = 0.031$.

^aWVS opinion questions usually state the claim in a very conservative fashion and allow the respondents either to agree or disagree with it. Here the answers are re-coded in a way that larger positive numbers always denote more support for the claim.

10.4 Comparison of Clustering and PCA

As methods of unsupervised learning, but cluster analysis and PCA share a number of traits. Both can be used to discover certain patterns in data, and given such patterns, to simplify, compress, and interpret the data.

But they also differ in a number of ways. In case of clustering, we are primarily interested in homogeneous subgroups. An example case is in Figure 10.7. The left panel of the figure contains 5 reasonably distinct groups that we can capture using clustering methods, such as k -means. We can use this group information in several ways:

- If the clusters correspond to the structural properties of the data, this helps us to interpret and understand the it.
- We can also design different measures to address different groups. For instance, different patients may require different treatment even with similar diagnosis.
- We can compress the data by replacing individual observations with the corresponding cluster center. Note that this does not constitute a dimensionality reduction: cluster center vectors are still of the same dimension

as individual data vectors.

- Sometimes the group membership itself is of interest: which cases go together?

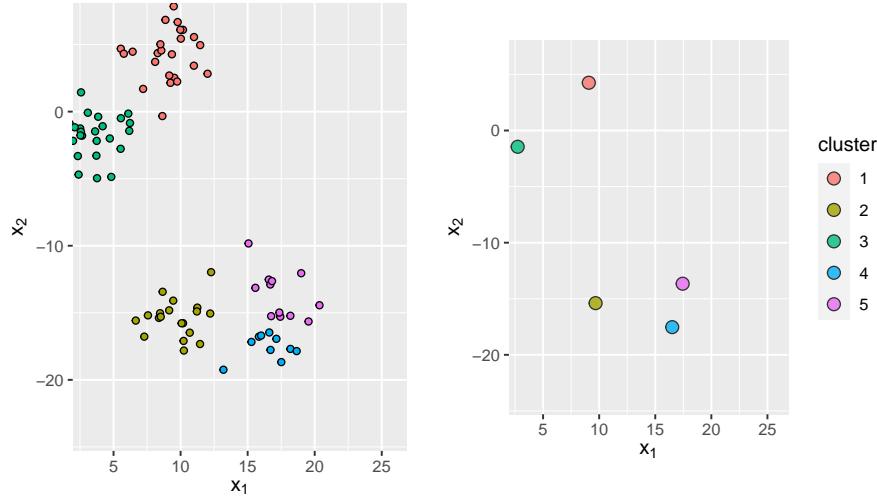


Figure 10.7: How cluster analysis treats data: homogenous subgroups (left panel) can be replaced by their corresponding cluster centers (right panel). In this way we can reduce the original 100-observation dataset to 5 different "types". These types can be either interpreted, one can design separate measures for each type, for instance marketing strategies in case of customer types, and one can also replace each observation with the cluster center in order to compress the data.

In case of principal components, our main task is to find a low-dimensional representation of data that contains most of the original information. This representation has a number of applications:

- Sometimes the reduction process itself reveals interesting properties of the data that can be interpreted.
- We can genuinely reduce the dimensionality of data by removing the (rotated) dimensions that carry little information. Unlike clustering, this process genuinely shrinks the data dimensionality. But the individual observations still remain distinct and are not replaced by certain average observations.

Lower-dimensional data is both easier to analyze and compress.

These methods also work well on different types of data. Cluster analysis is designed for data that contains well-separated relatively homogeneous "blobs" while principal component analysis can handle datasets that form an elongated cloud.

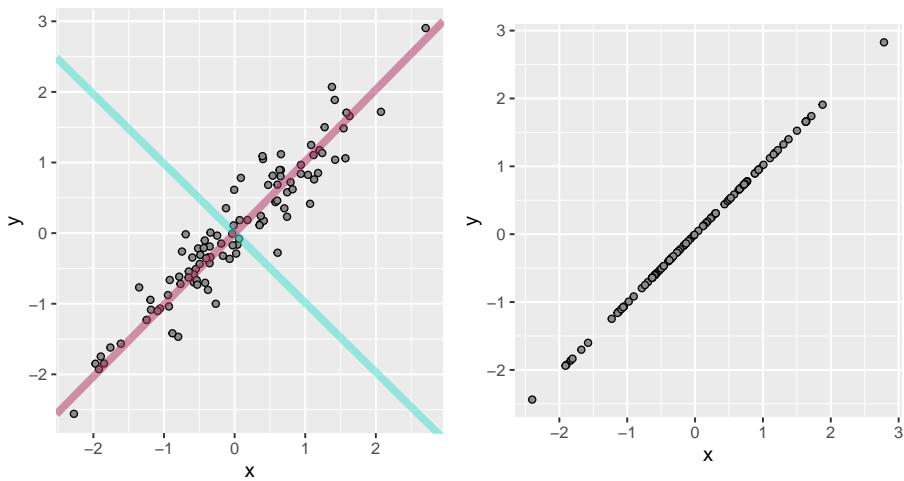


Figure 10.8: How PCA treats data: the dimension of maximum variance is treated as PC1 (red line, left panel) that carries most of the information. The other dimension (blue line) carries little information and is collapsed, resulting in rotated 1-D data (right panel). We may be interested in factor loadings, the relationship between the original features x_1 and x_2 , and the resulting principal components for interpretation and understanding. We may also prefer to analyze and store the reduced-dimensional data (right panel) instead of the original one.

Chapter 11

Applications

11.1 Recommender Systems

Recommender systems are in some ways similar to ordinary supervised ML methods. Their aim is to predict users’ “product rating” over different products, and recommend those with highest rating. The rating can be numeric, like in cases where users literally rate movies on 1-5 scale, or it may be a binary “rating”, e.g. the indicator if someone bought or did not buy a product.

However, recommenders also differ from the standard supervised models in several important aspects:

- In ordinary supervised models we base the estimates on some sort of universal user characteristics, such as age or education. Recommenders instead rely heavily on other recommendations made by users.
- Recommendation data is typically sparse. While we can collect common background information for most users, the users typically only rate a small minority of products. Hence we cannot rely on traditional methods, at least not without imputing the missing data.

Recommenders also have problems related to shifting human taste, e.g. after listening 10 songs of a certain genre, the user may get bored of this and look for something different. Such shifts are very hard to model.

11.1.1 Collaborative Filtering

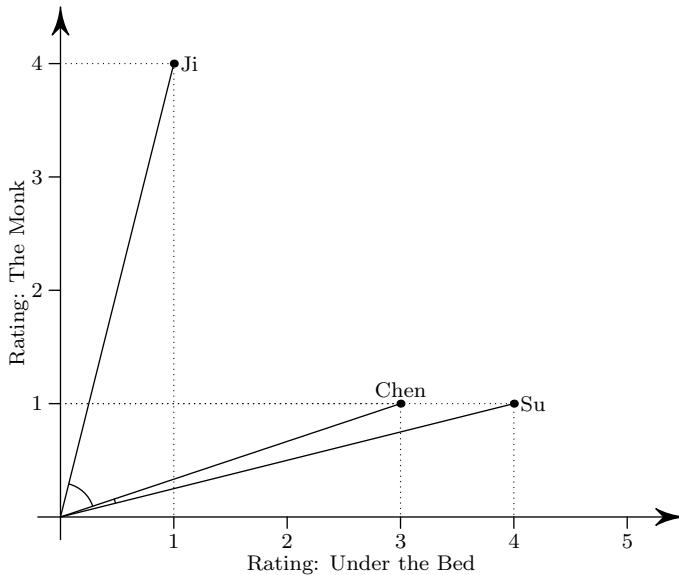
The idea of collaborative filtering is the following: when predicting ratings by user i , we find a set of users who are “most similar” to the user i , and base our decision on their ratings. The “most similar” here means users who are similar in terms of how do they rate products, not in terms of the background characteristics like education and age. This approach is in many ways similar to k -NN or local regression, just that is adapted to sparse data, and where we impute many datapoints.

We start with a trivial example. Consider people rating movies. There are three people: *Ji*, *Chen* and *Su*, and they are rating two movies, *Under the Bed* and *The Monk*. Imagine their ratings are

Name	Movie	Rating	Average	Centered rating
Ji	Under the Bed	1	2.5	-1.5
	The Monk	4		1.5
Chen	Under the Bed	3	2.0	1.0
	The Monk	1		-1.0
Su	Under the Bed	4	2.5	1.5
	The Monk	1		-1.5

Average is the users' average rating over all movies they have rated.

As this data is complete (all users rated all movies), we can easily depict the users in the 2-D rating space:



The raters are depicted as vectors pointing from the origin (0,0) to the corresponding ratings (*Under the Bed* rating on horizontal axis and *The Monk* rating on the vertical axis). A quick visual inspection also tells that Chen and Su are more “similar” than, for instance, Chen and Ji. Based on this quick picture we may already say that if Su liked a third move very much, Chen may also like it. But we are less certain what will Ji think about it as his tastes seem to be different.

In practice it is better to use centered cosine similarity, not Euclidean distance (our eyes implicitly measure Euclidean distance). The figure shows the angles between Chen and Ji, and Chen and Su. The angles (or cosines of the angles) lead to exactly the same conclusion in this case as the Euclidean

distance. For centered cosine similarity we have to compute the centered ratings by just subtracting the user's average value from her ratings (see the last column in the table).

* add centered figure to the figure

The centered data does not look particularly informative in the 2-D case as all the vectors are aligned on exactly the same NW-SE line. However, in a 3-D or in a higher dimensional space this is not so any more. But even in the 2-D case we see Chen and Su ratings pointing in one direction and Ji's rating pointing to the other direction. This tells us that Chen and Su are rather similar: both rate "Under the Bed" better than "The Monk". In case of "Ji" this is the opposite. The main advantage of centered distance is, however, that when we impute the missing ratings as user averages, the centered versions of the imputed values end up being 0. 0-length components do not change the cosine similarity, so computed user similarity is less affected by our imputations.

- * impute missings as user average ratings
- * centered cosine similarity
- * find most similar users (e.g. 10 of them)
- * use their ratings to compute the predictions
- * suggest the highest predicted values

Chapter 12

Responsible Data Science

As artificial intelligence is used more and more widely in our everyday lives, we encounter more and more situations that leave the technical/statistical side of AI. In this chapter we discuss these question with focus on social and ethical issues.

12.1 Explainable AI

A few simple statistical models can be explained fairly easily. For instance regression models are relatively easy to understand, but example-based k -NN and simpler decision trees are also not that complicated. However, many more advanced models, in particular neural networks, are essentially black boxes.

But humans often want explanations. Imagine a situation where the bank turns down your loan application. Why? Will you be happy with the clerk explaining that “I just pushed the button and this is what the computer told me...” If at the same time another, at least superficially similar customer who also happens to be of a different race got her load approved, then the situation may look quite troublesome for the bank. An explanation is urgently needed but what even constitutes a suitable explanation?

[Liao et al. \(2020\)](#) discuss the types of explanations that users of AI applications expect and need. They distinguish four broad types:

1. Global: explain the model. Explain the model using a global structure, e.g. weighting of features, as an approximate decision tree, or other decision rules.
2. Local: explain predictions for a particular instance. Which features of the particular case made the model predict what it predicts?
3. Counterfactual: how will prediction change when we change features? Which features should we change to get a certain prediction?

4. Example-based: provide examples of similar cases where the model provided similar predictions, and slightly different cases where the predictions were different.

Many users ask related questions in order to understand better the limitations of the model (What is permissible? How can I improve the training?) Another important application is to manipulate inputs in order to avoid undesirable outcomes. If the model predicts that the product will not be successful, then we want to know what should we change in order to make it a success.

12.2 Ethical Questions

As any other tool, statistics and machine learning can be used for good and for evil. Ethical dilemmas are nothing new to us, but as technology opens new avenues, we sometimes have to ask the age-old questions in a totally new context. Even if we can do this, should we do it? And how should we proceed in delicate cases?

12.2.1 Who Are Represented in Big Data?

Statistical models are trained on data and hence reflect the properties of the underlying data. When collecting dedicated survey data, such as World Value Survey, the researchers usually spend quite a bit of effort to ensure that the data is representative, and carefully document the sampling methods and resulting sampling weights in case certain populations or geographic regions are over/underrepresented. See Section 1.2.3 Sampling Process.

Unfortunately, such steps are often left undocumented in case of Big Data. Even worse, it is often unclear what would the theoretical population and sample frame even be in many cases. For instance, large NLP models are typically trained with text downloaded from internet. However, what would be a good representative sample of text? We know that different people leave behind a different amount of text depending on their habits and internet access. But should we strive to an equal representation of people? Text—language, is after all most cases produced as a part of communication between several persons. So perhaps it is appropriate that the loud voices are over-represented in a text corpus? As internet is also extremely complicated, it would be difficult to compute the sampling weights.

So it is not surprising at all that complex models that are trained on complex real data will re-produce various unfavorable traits that we encounter in the real world. And if we do not want the model to reflect such views, then what kind of views do we want it to reflect? For instance, should we refer to a certain group of immigrants as *undocumented* or *illegal*? As people have different opinion about the appropriate language here, addressing this question is necessarily political (Bender *et al.*, 2021).

12.2.2 Big Data, Big Inequality?

[Boyd and Crawford \(2012\)](#) discuss access to Big Data. Big Data is mainly collected by players in the internet industry, such as social media or online retail companies. These firms will have both the data and the resources for analysis, and they will decide who else have access to data. It probably leads to inequality in terms of research access where those with resources (prestigious universities and rich private research labs) will have access, and the other cannot easily participate in the relevant debate. Neither can they evaluate the quality of published big data-based research. The fact that the private gatekeepers do not follow similar transparency and public access requirements as the public sector data collectors will hamper analysis of topics that the data collectors find inconvenient.

12.2.3 Fairness: Different Measures are Incompatible

[Angwin et al. \(2016\)](#) analyze an algorithm, Correctional Offender Management Profiling for Alternative Sanctions (COMPAS), widely used in the U.S. criminal justice system to predict whether the defendant is likely to commit a new crime. Their analysis turned up “significant racial disparities”. In particular, whites who were labeled as “high risk” by the algorithm did not re-offend in 23.5% of cases, while African-Americans who were labeled “high-risk” did-not re-offend in 44.9% of cases. To put it differently, substantially more low-risk African-Americans were mis-categorized into high-risk category than the corresponding whites. Unfortunately it is not just a label for academic researchs—the perceived riskiness of recidivism influences sentences, right to bail, probation and other measures that have important real world effects on individual lives. The proponents of the COMPAS score have countered the criticism by demonstrating that at given score, both whites and African-Americans have similar probability to re-offend. So COMPAS score is a fair measure.

The problem boils down to different concepts of fairness. [Angwin et al. \(2016\)](#) criticism centers on *group fairness*, i.e. requirement that similar *groups of people*, here defined by race, should be treated similarly ([Jacobs and Wallach, 2021](#)). So whites who do not re-offend should have the same mis-classification rate as blacks who do not re-offend. This is clearly violated with COMPAS score. However, the its advocates rely on *individual fairness*, requirement that similar *individuals* to be treated equally. Unfortunately, these two concepts of fairness are not compatible in general ([Kleinberg et al., 2016](#)). Except in very specific cases, such as perfect predictions, it is possible to be fair either in one way or the other way, but not in both ways at the same time.

Consider the following example. There are two groups of people, Reds and Greens. The individual risk R can be either low L or high H . There is also a test X with two possible outcomes, either $X = 0$ or $X = 1$. Assume depending on the test score, the probability the individual is high-risk is

$$\Pr(H|X = 0) = 0.3 \quad \Pr(H|X = 1) = 0.8. \quad (12.2.1)$$

So when we predict, we predict all $X = 1$ to be High Risk and everyone with $X = 0$ to be of Low Risk. Assume the test is equally good for both reds and greens, so we can instead write

$$\begin{aligned} \Pr(H|X=0, \text{red}) &= 0.3 & \Pr(H|X=1, \text{red}) &= 0.8 \\ \Pr(H|X=0, \text{green}) &= 0.3 & \Pr(H|X=1, \text{green}) &= 0.8 \end{aligned} \quad (12.2.2)$$

so the test is perfectly fair in the individual fairness sense.

However, assume the base rates differ between Reds and Greens:

$$\Pr(H|\text{Red}) = 0.7 \quad \Pr(H|\text{Green}) = 0.4. \quad (12.2.3)$$

So there are more high-risk individuals among reds than among greens. Without any test data we are tempted to categorize all Reds to H and all Greens to L (this will probably considered a major violation of equal treatment). However, if we do the test, the Red/Green label does not carry any information, and we can perfectly ignore it from the prediction perspective. The corresponding confusion matrices for the both groups are in Table 12.1.

Table 12.1: Confusion matrices for Reds and Greens that correspond to the example case. We assume both groups contain 100 individuals. As the predictions are based on test X , $X = 0$ is equivalent to predicted class L and $X = 1$ to the predicted class H .

Reds				
		Predicted		
		$X = 0$	$X = 1$	Total
Actual	L	14	16	30
	H	6	64	70
Total		20	80	100

Greens				
		Predicted		
		$X = 0$	$X = 1$	Total
Actual	L	56	4	60
	H	24	16	40
Total		20	80	100

One can check that both confusion matrices correspond to the assumptions above. In particular, the fraction of high-risk individuals among Reds is larger than among Greens; precision for the high-risk individuals $\Pr(H|X=1) = 0.8$;

and precision for the low-risk individuals $\Pr(L|X = 0) = 0.7$. These probabilities are equal for both Reds and Greens so if, for instance, we observe that someone has $X = 0$, then the person has 70% of chance of being of low risk, no matter which color they belong to. Knowledge of group membership does not help to make better predictions given we have the test score. So policymakers who have to make decisions based on this test have no incentive to incorporate the group membership into their calculations.

Unfortunately, the equal treatment principle that is central to group fairness is now violated. False positive rate for Reds, $FPR_{Red} = 16/30 = 0.533$ while $FPR_{Green} = 4/60 = 0.067$. So eight times more low-risk Reds are misclassified as high-risk! Analogous calculation for false negative rate tells that seven times more high-risk Greens are misclassified as low-risk ones. If being classified as high-risk will bring real-world consequences, we are bringing such consequences unfairly to eight times more Reds than Greens. And many more Greens will unfairly be treated as low-risk even if they are not.

The problem is neither the test, machine learning algorithms, nor the method. The central problem here are the different concepts of fairness, and the fact that these are incompatible. Here we took a well-balanced test as given and showed that the equal treatment principle is violated. We can also take equal treatment as given and show that the corresponding test will not be balanced.

Part of the problem is the basing some of the fairness calculations on the group labels that are irrelevant as predictors. If we believe that group labels should not be used for prediction, and they do not carry any information (as in this example), then why do we want the fairness to be based on the “irrelevant” group labels? There are no good answers. But it just feels “fair”.

But wherever is the fundamental problem, the policymakers are facing an inconvenient choice. They have to decide between

- Ignoring the equal treatment principle
- Ignoring the score-balancing requirement
- Not using the risk predictions at all. However, in the example above this easily leads to perfect color discrimination where all Reds are believed to be of high risk and all Greens of low risk.

Obviously, one can also use a combination of these options.

12.3 Human Versus Algorithmic Decision-Making

Algorithms are often criticized as “obscure”, in particular when the inner workings of those are not published. Sometimes it is claimed that we should not use algorithms at all as algorithms are no less biased than humans. However, such claims miss a few important points. While complex algorithms are always obscure, human mind is no more transparent. While we can publish the inner details of algorithms (although not necessarily understand these), this is not possible in case of human brains. We can also analyze the data, and access possible problems there, but again, this is not possible to do with humans.

Instead of discussing the “obscurity” and “biasedness”, we should ask if algorithms can do better decisions than the relevant humans. For instance, can judges make better decisions if they have access to an algorithmic result? ([Kleinberg et al., 2018](#)) show that this is indeed the case in case of NYC judges. The judges have to decide whether to jail or release arrested criminals, and the authors show that judges’ decision is much affected by seemingly random factors. They tend to keep too many low-risk defendants in jail while releasing too many high-risk defendants. Algorithm would achieve a similar crime reduction with 20-40% smaller jail rate, or alternatively, at a similar jail rate it had achieved 25-15% smaller crime rate.

Appendix A

Mathematics

These notes assume you are reasonably familiar with basic calculus and a few other mathematical concepts, such as logarithm. Below is a list of the most important rules with little explanations.

A.1 High-School Mathematics

A.1.1 Logarithm

Definition: a is *logarithm* of x if $e^a = x$ where $e = 2.71828\dots$, and we write $a = \log x$. For instance, $\log 7.389 \approx 2$ as $e^2 \approx 7.389$.

Note: e -based logarithms as defined above are also called *natural logarithms*, and sometimes denoted by \ln instead of \log . Often the notation \log is reserved for *decimal logarithms*, defined as $\log_{10} x = a$ if $10^a = x$. Sometimes (in information theory for instance) we also use *binary logarithms* where $\log_2 x = a$ if $2^a = x$. Notation differs in different fields and between different authors. In these notes, *logarithm* always means natural logarithm and is denoted by \log . If needed, other logarithms are denoted by \log_{10} or \log_2 by explicitly writing their base.

Properties

$$\log x^\alpha = \alpha \log x \quad \text{and} \quad \log(x y) = \log x + \log y \quad (\text{A.1.1})$$

Logarithms of different base can easily be converted as

$$\log_b x = \frac{\log x}{\log b} \quad (\text{A.1.2})$$

Limits involving logarithm

TBD: $\lim_{x \rightarrow 1} \log x = x - 1$ etc

TBD: $\lim_{\epsilon \rightarrow 0} (1 + \epsilon)^\alpha = 1 + \alpha\epsilon$

$$\lim_{x \rightarrow 0} x \cdot \log x = 0 \quad (\text{A.1.3})$$

Proof A.1.1

Write $x \log x = (\log x)/(1/x)$ and use L'Hospital's rule.

A.1.2 Differentiation**Definition**

Derivative in case of univariate function is defined as

$$f'(x) \equiv \frac{df(x)}{dx} = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}. \quad (\text{A.1.4})$$

The definition in case of multivariate calculus is defined in an analogous way. If we only change one variable at a time, we treat the others as constants, and essentially we are back at the univariate case. Just the result is called *partial derivative* now, and denoted with ∂ symbol:

$$\frac{\partial f(x, y, z, \dots)}{\partial x} = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x, y, z, \dots) - f(x, y, z, \dots)}{\Delta x}. \quad (\text{A.1.5})$$

List of common differentiation rules

Here is a (non-exhaustive) list of the common rules for calculating the derivative of simple functions, and combinations of functions.

$$\frac{d}{dx} x^\alpha = \alpha x^{\alpha-1} \quad (\text{A.1.6})$$

$$\frac{d}{dx} \log x = \frac{1}{x} \quad (\text{A.1.7})$$

$$\frac{d}{dx} \log_b x = \frac{1}{\log b} \frac{1}{x} \quad (\text{A.1.8})$$

A few general rules for differentiation:

$$\frac{d}{dx} [\lambda f(x) + \mu g(x)] = \lambda f'(x) + \mu g'(x) \quad \text{differentiation is a linear operator} \quad (\text{A.1.9})$$

$$\frac{d}{dx} f(g(x)) = f'(g(x)) \cdot g'(x) \quad \text{chain rule} \quad (\text{A.1.10})$$

$$\frac{d}{dx} f(\lambda x) = \Lambda f'(x) \quad \text{application of chain rule} \quad (\text{A.1.11})$$

A.2 Matrix calculus

As linear algebra is the language of statistics, we may often want to do optimization in matrix form too. This requires *matrix calculus*. It is essentially ordinary

calculus, supplemented with a set of rules how to collect all the resulting objects that arise when differentiating the complements back into matrices and vectors.

Start simple. If we have a matrix, we can define its derivative with respect to a scalar just as a similar matrix where we have differentiated each component with respect to that scalar:

$$\frac{\partial}{\partial \lambda} \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1K} \\ x_{21} & x_{22} & \dots & x_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{NK} \end{bmatrix} = \begin{bmatrix} \frac{\partial \lambda}{\partial x_{11}} & \frac{\partial \lambda}{\partial x_{12}} & \dots & \frac{\partial \lambda}{\partial x_{1K}} \\ \frac{\partial \lambda}{\partial x_{21}} & \frac{\partial \lambda}{\partial x_{22}} & \dots & \frac{\partial \lambda}{\partial x_{2K}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \lambda}{\partial x_{N1}} & \frac{\partial \lambda}{\partial x_{N2}} & \dots & \frac{\partial \lambda}{\partial x_{NK}} \end{bmatrix}. \quad (\text{A.2.1})$$

Essentially we take derivatives of each element of a collection (a collection we call matrix) and put these back into a similar collection. So little changes if we differentiate matrices with respect to a scalar.

Differentiation with respect to a vector requires additional rules, however. Let's take the simplest case: differentiate a scalar function $f : \mathbb{R}^K \rightarrow \mathbb{R}$ with respect to a column vector: $\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x})$. Note that as $f(\mathbf{x})$ is a function of the *vector* \mathbf{x} , it can instead be written as $f(x_1, x_2, \dots, x_K)$ if \mathbf{x} has K components. It is also a *scalar function*, i.e. it associates a single number with the input vector \mathbf{x} . We define the derivative with respect to the column vector \mathbf{x} as:

$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(\mathbf{x}) \\ \frac{\partial}{\partial x_2} f(\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_K} f(\mathbf{x}) \end{bmatrix} \quad (\text{A.2.2})$$

and w.r.t. the row vector as

$$\frac{\partial}{\partial \mathbf{x}'} f(\mathbf{x}) = \left[\frac{\partial}{\partial x_1} f(\mathbf{x}) \quad \frac{\partial}{\partial x_2} f(\mathbf{x}) \quad \dots \quad \frac{\partial}{\partial x_K} f(\mathbf{x}) \right]. \quad (\text{A.2.3})$$

So we simply take the partial derivatives of the function with respect to all individual components of the vector, and stack the results in a vector of the same shape. So derivative of a scalar function with respect to a vector is a vector of similar shape. It's not too bad so far.

This rule has a nice application. In case of both \mathbf{x} and $\boldsymbol{\beta}$ are $K \times 1$ column vectors, $\mathbf{x}^T \boldsymbol{\beta} = \boldsymbol{\beta}^T \mathbf{x} = \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_K x_K$ is a scalar. Hence

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{x}^T \boldsymbol{\beta} = \frac{\partial}{\partial \mathbf{x}} \boldsymbol{\beta}^T \mathbf{x} = \begin{bmatrix} \frac{\partial}{\partial x_1} (\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_K x_K) \\ \frac{\partial}{\partial x_2} (\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_K x_K) \\ \vdots \\ \frac{\partial}{\partial x_K} (\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_K x_K) \end{bmatrix} = \boldsymbol{\beta}. \quad (\text{A.2.4})$$

This is very similar to the ordinary calculus where $\frac{\partial}{\partial x} x \cdot \boldsymbol{\beta} = \boldsymbol{\beta}$.

Things get more complex if we want to differentiate a *vector function* w.r.t a vector. Let's stay with the cases that are easier to represent: derivative of a column vector function w.r.t a row vector, and the way around. A vector function is a function that associates a vector with each argument value. Let's look at a function $\mathbf{f} : \mathbb{R}^K \rightarrow \mathbb{R}^N$, i.e. it associates a N -dimensional vector with each K -dimensional argument. The concept of vector function is simply a shorthand of writing

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{pmatrix} \quad (\text{A.2.5})$$

i.e. it is a suitably stacked collection of N scalar functions of a vector arguments, which in turn, can be written with no vector notation at all as

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(x_1, x_2, \dots, x_K) \\ f_2(x_1, x_2, \dots, x_K) \\ \vdots \\ f_N(x_1, x_2, \dots, x_K) \end{pmatrix}. \quad (\text{A.2.6})$$

Now we have to take a derivative of this stack of functions w.r.t the row vector \mathbf{x}^T . We just take each individual (vertical) component, differentiate it as in (A.2.3), and stack the resulting row vectors vertically. This gives us a matrix:

$$\frac{\partial}{\partial \mathbf{x}^T} \mathbf{f}(\mathbf{x}) \left[\begin{array}{cccc} \frac{\partial}{\partial x_1} f_1(\mathbf{x}) & \frac{\partial}{\partial x_2} f_1(\mathbf{x}) & \dots & \frac{\partial}{\partial x_K} f_1(\mathbf{x}) \\ \frac{\partial}{\partial x_1} f_2(\mathbf{x}) & \frac{\partial}{\partial x_2} f_2(\mathbf{x}) & \dots & \frac{\partial}{\partial x_K} f_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_1} f_N(\mathbf{x}) & \frac{\partial}{\partial x_2} f_N(\mathbf{x}) & \dots & \frac{\partial}{\partial x_K} f_N(\mathbf{x}) \end{array} \right]. \quad (\text{A.2.7})$$

So the derivative of N -dimensional column vector w.r.t K dimensional row vector is a $N \times K$ matrix. This is a nice result that can be used in several applications. If \mathbf{A} is a $N \times K$ matrix

$$\frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}'} = \mathbf{A} \quad \text{and} \quad \frac{\partial \mathbf{x}'\mathbf{A}}{\partial \mathbf{x}} = \mathbf{A}. \quad (\text{A.2.8})$$

(You simply have to write down the definition of $\mathbf{A}\mathbf{x}$, and use (A.2.3) to get the result).

Additional useful results without proofs:

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{x}^T = \begin{bmatrix} \frac{\partial x_1}{\partial x_1} & \frac{\partial x_1}{\partial x_2} & \cdots & \frac{\partial x_1}{\partial x_K} \\ \frac{\partial x_2}{\partial x_1} & \frac{\partial x_2}{\partial x_2} & \cdots & \frac{\partial x_2}{\partial x_K} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_K}{\partial x_1} & \frac{\partial x_K}{\partial x_2} & \cdots & \frac{\partial x_K}{\partial x_K} \end{bmatrix} = \mathbf{I} \quad \frac{\partial}{\partial \mathbf{x}^T} \mathbf{x} = \mathbf{I} \quad (\text{A.2.9})$$

$$\frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}^T} = \mathbf{A} \quad \frac{\partial \mathbf{x}^T \mathbf{A}}{\partial \mathbf{x}} = \mathbf{A} \quad (\text{A.2.10})$$

$$\frac{\partial \mathbf{x}^T \mathbf{x}}{\partial \mathbf{x}} = \|\mathbf{x} + \mathbf{x}\| = 2\mathbf{x} \quad \frac{\partial \mathbf{x}^T \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A}^T)\mathbf{x}. \quad (\text{A.2.11})$$

All these results can be proven by using the definition of matrix multiplication, and the differentiation wrt vector (A.2.2) and (A.2.3).

A.2.1 Gradient

Prerequisites: Calculus, and a basic understanding of multivariate calculus

What is Gradient

Gradient is generalization of derivative for functions on \mathbb{R}^n . While the derivative describes the slope of the function in 1-dimensional case, gradient indicates both the slopes (along different axes) and the direction of the steepest ascent for functions of n variables (functions on \mathbb{R}^n). Below, we only look at the scalar functions $\mathbb{R}^n \rightarrow \mathbb{R}$, i.e. the function take an n -dimensional input but return a scalar value.

Perhaps the easiest way to understand this is to think about a hilly landscape. Elevation is a function $\mathbb{R}^2 \rightarrow \mathbb{R}$: from two inputs (longitude and latitude) to a single number (elevation). Gradient tells at which rate the ground rises, and where is the direction of the steepest climb.

Let us take a simple example

$$f(\mathbf{x}) \equiv f(x_1, x_2) = x_1 \cdot \log x_2 \quad (\text{A.2.12})$$

where \mathbf{x} is the 2-dimensional input vector $(x_1, x_2)'$. Two-parameter functions can easily be visualized as surfaces, $f(\mathbf{x})$ is depicted in Figure A.1.

For this function, the partial derivatives are $\frac{\partial}{\partial x_1} f(\mathbf{x}) = \log x_2$ and $\frac{\partial}{\partial x_2} f(\mathbf{x}) = x_1/x_2$. In a way, gradient, commonly denoted by $\nabla f(\mathbf{x})$ or sometimes $\partial f(\mathbf{x})/\partial \mathbf{x}$, is just a compact way to write this in vector form:

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \log x_2 \\ x_1/x_2 \end{pmatrix}. \quad (\text{A.2.13})$$

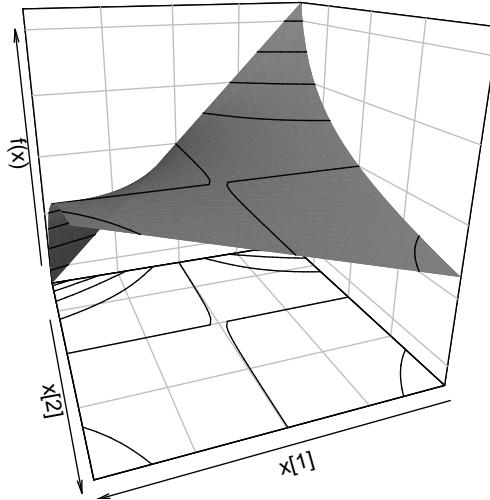


Figure A.1: $f(\mathbf{x}) = x_1 \cdot \log x_2$ as function of x_1 and x_2 . The level sets, contours of equal values, are plotted both on the surface and on bottom of the figure box.

This is a 2×1 vector. So gradient is just a habit to stack the partial derivatives into a vector. Compared to the function itself, gradient is harder to visualize as it has two values (it's range is in \mathbb{R}^2). Figure A.2 shows two options for visualizing $\nabla f(\mathbf{x})$.

In case of n -dimensional argument functions $\mathbb{R}^n \rightarrow \mathbb{R}$, we have n partial derivatives $\frac{\partial}{\partial x_1} f(\mathbf{x}), \frac{\partial}{\partial x_2} f(\mathbf{x}), \dots, \frac{\partial}{\partial x_n} f(\mathbf{x})$ and we stack these into the gradient as

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial}{\partial x_1} f(\mathbf{x}) \\ \frac{\partial}{\partial x_2} f(\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_n} f(\mathbf{x}) \end{pmatrix} \quad (\text{A.2.14})$$

so the gradient is $n \times 1$ vector. Note that gradient is a function too, although not a scalar valued one but a vector valued function $\mathbb{R}^n \rightarrow \mathbb{R}^n$: it associates each n -dimensional argument value \mathbf{x} to a n -dimensional vector $\nabla f(\mathbf{x})$. This is analogous with the derivative in one-dimensional case $\mathbb{R} \rightarrow \mathbb{R}$, that one is also an one-dimensional function $\mathbb{R} \rightarrow \mathbb{R}$.

While gradient itself is a function, when we calculate its value for any particular argument of \mathbf{x} , the result will be a vector (not function). This is exactly analogous to the ordinary derivative, which is a function but when calculated at a particular x value it is a number.

As an example, let's take the function $f(\mathbf{x})$ we defined above and let's cal-

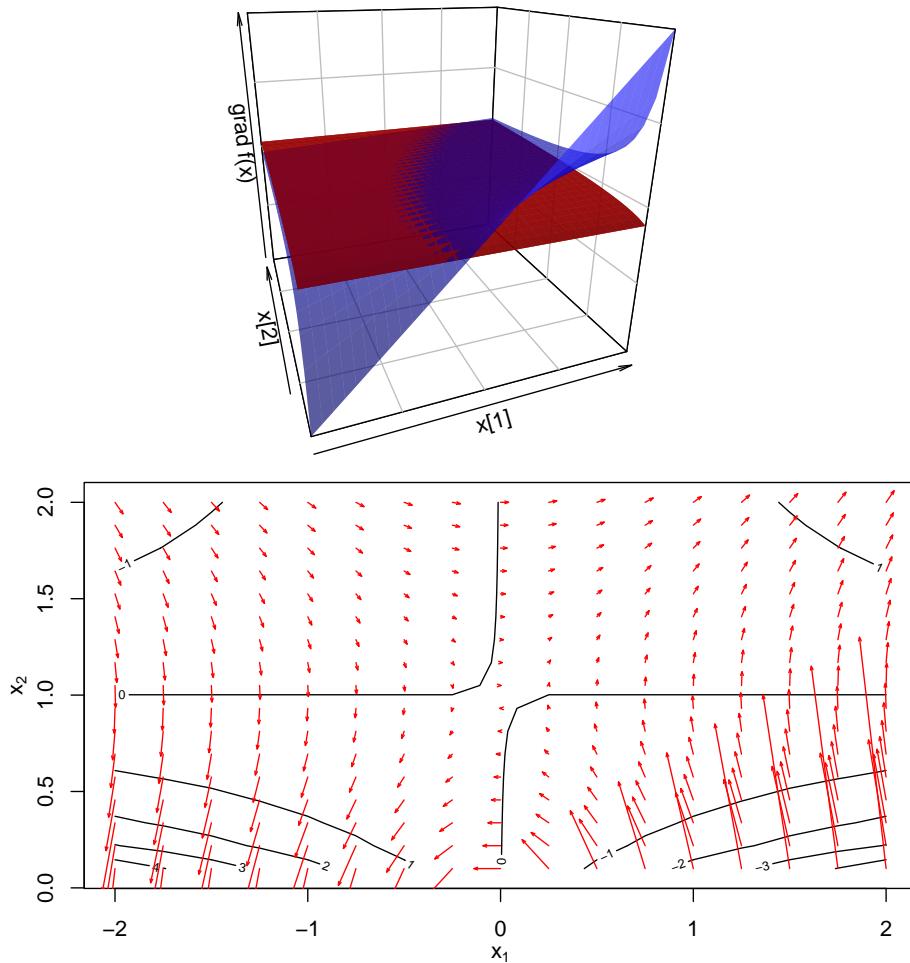


Figure A.2: Gradient of $f(\mathbf{x})$, depicted as two surfaces (upper panel). The blue surface corresponds to $\frac{\partial}{\partial x_2} f(\mathbf{x}) = x_1/x_2$, the red one to $\frac{\partial}{\partial x_1} f(\mathbf{x}) = \log x_2$. The lower panel depicts the gradient as arrows plotted on the levels (contours) of the function. The length of the arrows is proportional to the gradient length, their direction is equal to the gradient direction. One can easily see that the norm of gradient is proportional to the steepness of the function surface, and gradient points to the direction of the steepest climb.

culate it's value, and it's gradient's value at $\mathbf{x} = (1, 2)'$:

$$f(\mathbf{x})|_{\mathbf{x}=(1,2)'} = 1 \cdot \log 2 \approx 0.693. \quad (\text{A.2.15})$$

This is seldom written in such a long way, almost all texts use a shorter but somewhat misleading version of $f((1,2)')$, or just $f(1,2)$ instead. A similar notation applies to gradient. It's value at $(1,2)'$ can be written as

$$\nabla g(\mathbf{x})|_{\mathbf{x}=(1,2)'} = \begin{pmatrix} \log x_2 \\ x_1/x_2 \end{pmatrix} \Big|_{\mathbf{x}=(1,2)'} = \begin{pmatrix} \log 2 \\ 1/2 \end{pmatrix}. \quad (\text{A.2.16})$$

As before, this value is often written in shorter but imprecise way as

$$\nabla f \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} \log 2 \\ 1/2 \end{pmatrix}. \quad (\text{A.2.17})$$

It is imprecise because $f \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ is a constant, $\log 2$, and gradient of a constant is always 0. However, it is a widely used shortcut in the literature. It is important to distinguish between gradient of a function, calculated at a fixed argument value; and between a function, computed at the same fixed argument values. When using the shortcut notation we have to understand that $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ is the argument, gradient is computed with respect of the argument, and afterwards evaluated at this value of the argument.

However, it appears gradient is much more than a handy way to write a number of derivatives in a compact form. It naturally generalizes a number of properties of 1-dimensional derivatives.

Gradient and Direction of Steepest Ascent

As components of gradient are just ordinary partial derivatives, each component indicates the slope of the function along that axis: how much will the function value grow if we move along the axis, while keeping the location on the other axes constant. If one component is large and another is small, we have a steep hill in the first direction while it is pretty flat along the other axis.

Let's look at linear functions, simple even surfaces with no curvature whatsoever. A linear function may look something like the plane depicted on Figure A.3. If the surface is rising rapidly along x_2 while staying constant along x_1 , the direction of fastest climb is just along x_2 . Analogously, if the function grows along x_1 while staying flat along x_2 , we have to move toward x_1 . Such a situation is depicted on Figure A.4 although here the first gradient component $\partial f(\mathbf{x})/\partial x_1 < 0$ and hence we have to move toward smaller values of x_1 instead if we want to climb uphill. Obviously, if none of the gradient components are zero, we have to move somewhere in-between of these two directions. This is shown on Figure A.5. It is also intuitive, that the "somewhere in-between" should be closer to the steeper gradient than to the smaller gradient component.

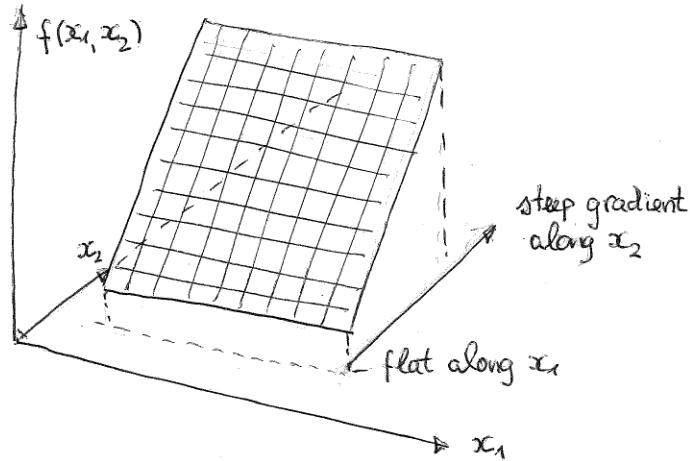


Figure A.3: Function increasing along x_2 while constant along x_1 .

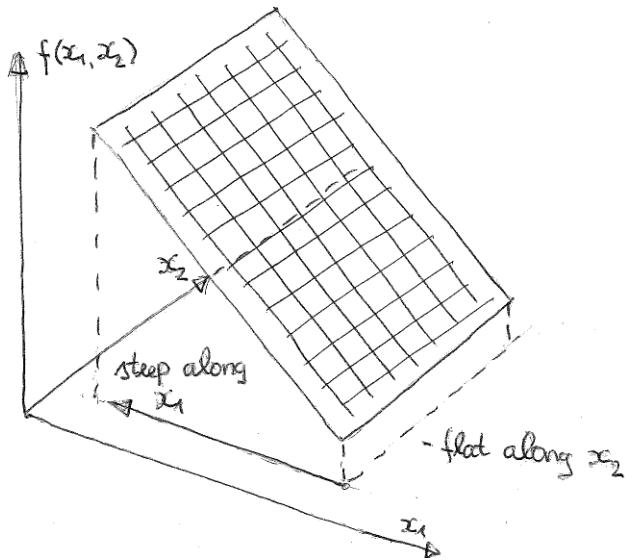


Figure A.4: Function increasing along $-x_1$ while constant along x_2 .

It is easy to show that the exact direction of the steepest climb is the same as the direction of the gradient vector. Let's choose a point (x_1, x_2) where the function's value is $f(x_1, x_2)$. Now move away from this point by $(\Delta x_1, \Delta x_2)$. This causes the function to grow by

$$\Delta f \equiv f(x_1 + \Delta x_1, x_2 + \Delta x_2) - f(x_1, x_2) \approx g_1 \cdot \Delta x_1 + g_2 \cdot \Delta x_2 \quad (\text{A.2.18})$$

where g_1 and g_2 are the corresponding gradient components, calculated at (x_1, x_2) . However, for not to go too much wild, we'll change the coordinates in

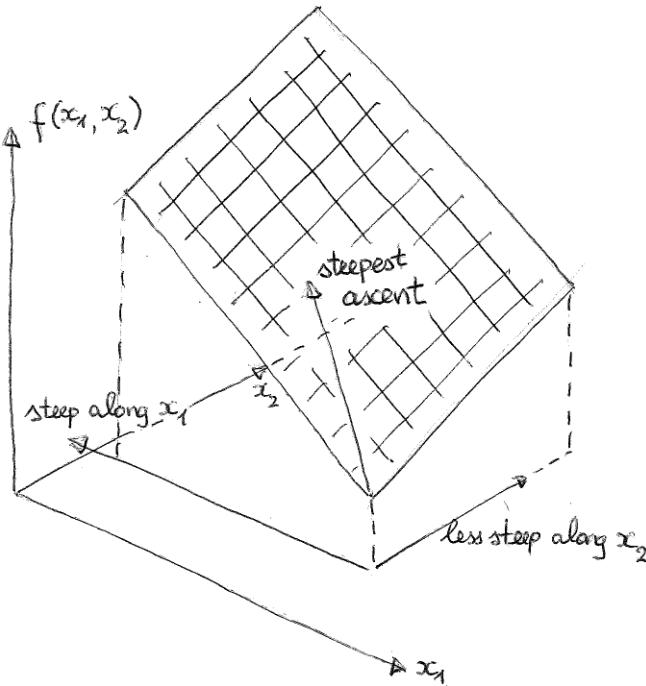


Figure A.5: Function increasing both $-x_1$ and x_2 . The direction of steepest climb is somewhere between these two gradient components.

this way that the total move will be of length one. Hence $\Delta x_1^2 + \Delta x_2^2 = 1$, or alternatively $\Delta x_2 = \sqrt{1 - \Delta x_1^2}$, and the change of the function can be written as

$$\Delta f \approx g_1 \cdot \Delta x_1 + g_2 \cdot \Delta x_2 \quad \text{or} \quad \Delta f \approx g_1 \cdot \Delta x_1 + g_2 \cdot \sqrt{1 - \Delta x_1^2}. \quad (\text{A.2.19})$$

Which Δx_1 results in the largest value of Δf ? The optimality condition gives us

$$\frac{\partial \Delta f}{\partial \Delta x_1} = g_1 + g_2 \frac{-2\Delta x_1}{2\sqrt{1 - \Delta x_1^2}} = g_1 - g_2 \frac{\Delta x_1}{\Delta x_2} = 0 \quad (\text{A.2.20})$$

where we used the fact that $\sqrt{1 - \Delta x_1^2} = \Delta x_2$. The solution is

$$\frac{\Delta x_1}{\Delta x_2} = \frac{g_1}{g_2}. \quad (\text{A.2.21})$$

In other words, this means that the direction vector $(\Delta x_1, \Delta x_2)$ must be parallel to the gradient vector (g_1, g_2) .

Appendix B

Exercise Solutions

Solution (1.2.1). We have data $x = (1, 2, 3, 3, 3, 5, 5, 10)$.

1. Mean is

$$\bar{x} = (1 + 2 + 3 + 3 + 3 + 5 + 5 + 10)/8 = 32/8 = 4$$

2. Median is 3 as the “middle” of the data is between two “3”-s (there are three numbers smaller than 3 and three numbers larger than 3).
3. mode is 3 as this is the most frequent number.

If one data point is missing, we cannot compute mean. For median, we can tell that it must be between 3 and 5: if the missing data point is smaller than 3, the median is still 3. If it is larger than 5, there are 3 numbers no larger than 3 and 3 numbers no smaller than 5 in the data, and hence median is between 3 and 5 (potentially equal to either 3 or 5). So we have bounds on the median. The model will be either 3 (if the missing value is not 5), or it is a bimodal dataset with modes both 3 and 5 (if the missing value is 5).

Solution (1.2.2). According to media, the net worth of Bill Gates is \$105 billion, almost three times the total wealth of Iceland. Hence an option would be to grant Bill Gates Iceland citizenship and in this way to make him an “Icelander”. This will make the average wealth of Icelanders to grow from \$95,000 to \$370,000 per person.

The problem is the meaning of the expression “all Icelanders”. People understand it intuitively that it applies to everyone (everyone individually), but here it is (most likely deliberately) used in a different sense, something like “everyone combined together”.

Solution (1.2.3). The naive answer would be to place armor in the most damaged parts of the airplanes. However, note that we face a missing data problem here: we can only observe those places that actually return to the base. We don’t know where were those planes hit that did not return.

However, we can still guess: as anti-aircraft fire is very imprecise, there is no reason to believe that engines and cockpits were not hit. The fact that we do not see much damage in those parts hints that those are the weakest points. If engines or cockpit are hit, the plane won't return. Hence you should recommend to armor those areas that are *not damaged!*

Solution (1.3.1). Let 0 correspond to the case where there were no 6-s on the dice, and 1 if there were at least one 6. We can write the RV as

$$X = \begin{cases} 1 & \text{if } X \in \{(1,6), (2,6), (3,6), (4,6), (5,6), (6,6), \\ & (6,5), (6,4), (6,3), (6,2), (6,1)\} \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.0.1})$$

Solution (1.3.2). Here is the 6×6 table of all possible outcomes with sum 6 highlighted in blue:

		Sum of two dies					
		Die 2					
		1	2	3	4	5	6
Die 1	1	2	3	4	5	6	7
	2	3	4	5	6	7	8
	3	4	5	6	7	8	9
	4	5	6	7	8	9	10
	5	6	7	8	9	10	11
	6	7	8	9	10	11	12

There are clearly 5 ways to get sum 6, hence the corresponding probability $\Pr(Z = 6) = 5/36$.

Solution (1.3.3). If the die is fair, all sides have probability $1/6$. Hence the expected value

$$\mathbb{E} D = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \dots + \frac{1}{6} \cdot 6 = \frac{1}{6}(1 + 2 + \dots + 6) = 3.5$$

Solution (1.3.4). a) First we have to find the expected value: $\mathbb{E} X = 0.25 \cdot (-1) + 0.5 \cdot 0 + 0.25 \cdot 1 = 0$. It is also immediately obvious as the values are symmetric around 0. Next, let us do a table, similar to Table 1.3:

x	$\Pr(X = x)$	$X - \mathbb{E} X$	$(X - \mathbb{E} X)^2$
-1	0.25	-1	1
0	0.50	0	0
1	0.25	1	1

Note that as $\mathbb{E} X = 0$, the columns 1 and 3 are the same. Variance, the expected value of the last column is $\text{Var } X = \mathbb{E}(X - \mathbb{E} X)^2 = 0.25 \cdot 1 + 0.25 \cdot 1 = 0.5$.

b) We already know $\mathbb{E} X$, so we have to find $\mathbb{E} X^2$: $\mathbb{E} X^2 = 0.25 \cdot (-1)^2 + 0.5 \cdot 0^2 + 0.25 \cdot 1^2 = 0.5$. The variance is $\text{Var } X = \mathbb{E} X^2 - (\mathbb{E} X)^2 = 0.5 - 0 = 0.5$.

As the example demonstrates, in case of $\mathbb{E} X = 0$, the variance is equal to just $\mathbb{E} X^2$.

TBD: continuous case

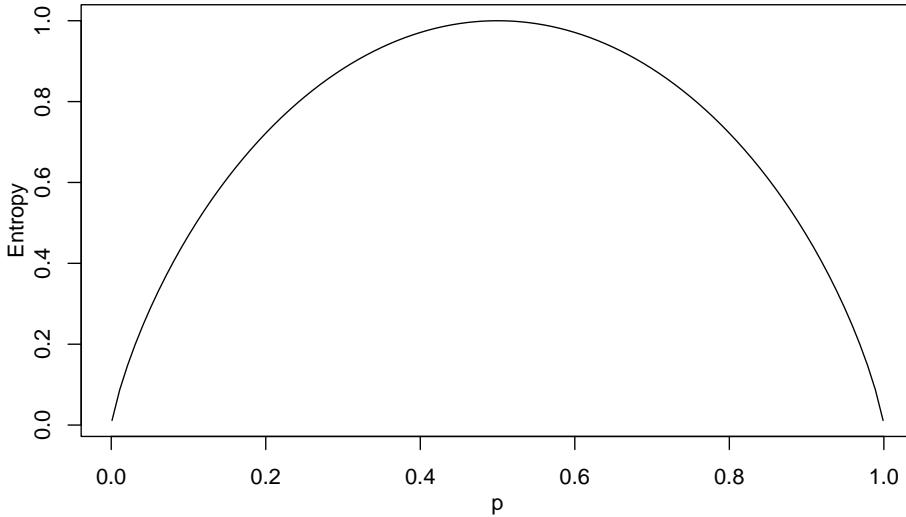
Solution (1.3.5). X must be ordinal for the comparison in (1.3.17) to have a meaning.

Solution (1.3.6). Standard uniform as specified in (1.3.23) has density 1 in the interval $[0,1]$. So all values between 0 and 1 are equally likely, and values outside of this interval are impossible. Hence the lower 2.5% quantile is the lowest 2.5% end of this interval, $q_{0.025} = 0.025$ and upper $q_{0.975} = 0.975$.

Solution (1.3.7). Bernoulli distribution has two states: the event happens with probability p and does not happen with probability $1 - p$. Inserting this into the definition of entropy (1.3.24), we get

$$\mathbb{H}(X) = -p \log_2 p - (1 - p) \log_2(1 - p). \quad (\text{B.0.2})$$

The entropy as a function of p will look like



The entropy is 0 at both ends of the curve: we are (almost) certain the event either does not happen ($p = 0$) or happens ($p = 1$), and hence there is no uncertainty. The largest uncertainty is in the middle where both outcomes are equally likely, and we can gain 1 bit of information.

Solution (2.2.1). Dimension is just the number of components in the vector. Hence \mathbf{v}_i is of dimension 8. The table of data itself does not reveal how long are \mathbf{x}_i -s, but as the data is about “50 U.S. States”, its dimension must be 50.

Solution (2.2.2). We can compute individual components as $e(\text{Berlin})_1 - e(\text{Germany})_1 + e(\text{France})_1 = -0.562 - 0.194 + 0.605 = -0.151$, $e(\text{Berlin})_2 - e(\text{Germany})_2 + e(\text{France})_2 = 0.630 - 0.507 - 0.678 = -0.555$, and for the following 3 components we have -1.176 , -0.450 , and -0.016 . So the vector

$$e(\text{Berlin}) - e(\text{Germany}) + e(\text{France}) = (-0.151, -0.555, -1.176, -0.450, -0.016)$$

while

$$e(\text{Paris}) = (-0.074, -0.855, -0.689, -0.057, -0.139)$$

As one can see, the result is not exact, but broadly agrees in terms of size and sign of the components, unlike any other word listed here.

Solution (2.2.3). As in case of Example 2.2.3, we can express

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = 2 \cdot \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} - \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix}, \quad (\text{B.0.3})$$

or

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} - 2 \cdot \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} + \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \quad (\text{B.0.4})$$

Hence these vectors are not linearly independent.

Solution (2.3.1). The first product:

$$\begin{aligned} \left[\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \right] \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} &= \begin{pmatrix} -2 & 2 \\ -2 & 2 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \\ &= \begin{pmatrix} 2 & -2 & 2 \\ 2 & -2 & 2 \end{pmatrix}. \quad (\text{B.0.5}) \end{aligned}$$

The second product:

$$\begin{aligned} \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \cdot \left[\begin{pmatrix} -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \right] &= \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{pmatrix} = \\ &= \begin{pmatrix} 2 & -2 & 2 \\ 2 & -2 & 2 \end{pmatrix}. \quad (\text{B.0.6}) \end{aligned}$$

These are indeed equal.

Solution (2.3.2). As we are in 2-D space, we can write (2.2.5) using components as

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \alpha \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} + \beta \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \quad (\text{B.0.7})$$

Note that this expression is equivalent to

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \cdot \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \quad (\text{B.0.8})$$

Hence $\begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ can be isolated using the inverse

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}^{-1} \cdot \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \quad (\text{B.0.9})$$

Solution (2.3.3). By the properties of trigonometric functions we have

$$R(-\alpha) = \begin{pmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{pmatrix}$$

and hence

$$\begin{aligned} R(\alpha) \cdot R(-\alpha) &= \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} \cdot \begin{pmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{pmatrix} = \\ &= \begin{pmatrix} \cos^2 \alpha + \sin^2 \alpha & \cos \alpha \sin \alpha - \sin \alpha \cos \alpha \\ \sin \alpha \cos \alpha - \cos \alpha \sin \alpha & \sin^2 \alpha + \cos^2 \alpha \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{aligned}$$

Solution (4.1.1). TBD

Solution (4.1.2). Let's multiply X and β in (4.1.40) using the ordinary matrix multiplication rules:

$$\begin{pmatrix} 1 & x_1^1 & x_2^1 & \dots & x_K^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_K^2 \\ \vdots & \vdots & \ddots & & \vdots \\ 1 & x_1^N & x_2^N & \dots & x_K^N \end{pmatrix} \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \end{pmatrix} = \begin{pmatrix} \beta_0 & \beta_1 x_1^1 & \beta_2 x_2^1 & \dots & \beta_K x_K^1 \\ \beta_0 & \beta_1 x_1^2 & \beta_2 x_2^2 & \dots & \beta_K x_K^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \beta_0 & \beta_1 x_1^N & \beta_2 x_2^N & \dots & \beta_K x_K^N \end{pmatrix} \quad (\text{B.0.10})$$

and hence we can write (4.1.40) as

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} \beta_0 & \beta_1 x_1^1 & \beta_2 x_2^1 & \dots & \beta_K x_K^1 \\ \beta_0 & \beta_1 x_1^2 & \beta_2 x_2^2 & \dots & \beta_K x_K^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \beta_0 & \beta_1 x_1^N & \beta_2 x_2^N & \dots & \beta_K x_K^N \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{pmatrix}. \quad (\text{B.0.11})$$

Each line of this equation is equivalent to (4.1.39) after performing the matrix multiplication there, and there are N lines. Hence (4.1.40) is equivalent to (4.1.39) for each $i = 1 \dots N$.

Solution (6.1.1). We have

$$TN = 10, TP = 60, FP = 20, FN = 10 \quad (\text{B.0.12})$$

and $T = 100$. Hence

$$\begin{aligned} A &= \frac{TN + TP}{T} = \frac{70}{100} = 0.7 \\ P &= \frac{TP}{TP + FP} = \frac{60}{80} = 0.75 \\ R &= \frac{TP}{TP + FN} = \frac{60}{70} \approx 0.86 \\ F &= \frac{2}{\frac{1}{P} + \frac{1}{R}} = 0.8 \end{aligned} \quad (\text{B.0.13})$$

Solution (3.2.1). Section 2.2.2 lists three properties of distance metric. The first one is

$$d(\mathbf{x}, \mathbf{y}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{y}.$$

This property is not satisfied for cosine-related distances: $d_{cos}(\mathbf{x}, \mathbf{y}) = 0$ means cosine is 1 (or angle is 0), but this is true for all vectors that point to the same direction, not just for equal vectors.

Another issue arises from the fact that these distances are not defined for null vector, hence distance between null-vector and any other vector is undefined.

Solution (7.3.1). Re-write the Bayesian expression for the case of no-viagra-no-spam:

$$\Pr(S = 0 | V = 0) = \frac{\Pr(V = 0 | S = 0) \cdot \Pr(S = 0)}{\Pr(V = 0)} \quad (\text{B.0.14})$$

Based on the table in Example 7.3.1, we can compute the necessary probabilities:

- $\Pr(V = 0 | S = 0)$, probability of no “viagra” in no-spam emails. From the table we can see that it is $500/650 = 10/13 \approx 0.769$.
- The prior, $\Pr(S = 0)$, the proportion of legitimate emails. It is $600/1000 = 3/5 = 0.6$.
- The normalizer, $\Pr(V = 0)$, the probability not to see “viagra” in emails, $650/1000 = 13/20 = 0.65$.

Inserting the values in (B.0.14), we get

$$\begin{aligned} \Pr(S = 0 | V = 0) &= \frac{\Pr(V = 0 | S = 0) \cdot \Pr(S = 0)}{\Pr(V = 0)} = \\ &= \frac{\frac{10}{13} \cdot \frac{3}{5}}{\frac{13}{20}} = \frac{200}{169} \frac{3}{5} \approx 0.71. \end{aligned} \quad (\text{B.0.15})$$

Based on the information that the email contains no word “viagra”, we update the prior 0.6 by $200/169 \approx 1.18$ times.

Solution (7.3.2). TBD

Solution (7.4.1). An easy solution is $w_1 = w_2 = 1$, $\bar{z} = 0.5$.

Solution (7.4.2). From the Table 7.7 we have weights

$$\mathbf{w}_{h1} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \mathbf{w}_{h2} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \mathbf{w}_y = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (\text{B.0.16})$$

and biases

$$b_{h1} = 1.5 \quad b_{h2} = 0.5 \quad b_y = 0.5. \quad (\text{B.0.17})$$

We can compute the h_1 node values:

$$\chi_1 = \mathbf{x}^\top \cdot \mathbf{w}_{h1} = (0 \quad 1) \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 1 \quad \text{and} \quad h_1 = \mathbb{1}(\chi_1 > b_{h1}) = \mathbb{1}(1 > 1.5) = 0. \quad (\text{B.0.18})$$

Analogously, for the second hidden node h_2 we have

$$\chi_2 = \mathbf{x}^\top \cdot \mathbf{w}_{h2} = (0 \quad 1) \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 1 \quad \text{and} \quad h_2 = \mathbb{1}(\chi_2 > b_{h2}) = \mathbb{1}(1 > 0.5) = 1. \quad (\text{B.0.19})$$

So we have $\mathbf{h} = (h_1, h_2)^\top = (0, 1)^\top$. Now we can perform a similar operation with the output layer:

$$z = \mathbf{h}^\top \cdot \mathbf{w}_y = (0 \quad 1) \cdot \begin{pmatrix} -1 \\ 1 \end{pmatrix} = 1 \quad \text{and} \quad y = \mathbb{1}(z > b_y) = \mathbb{1}(1 > 0.5) = 1. \quad (\text{B.0.20})$$

So we have $0 \text{ XOR } 1 = 1$.

Bibliography

- Amoros, E., Chiron, M., Martin, J.-L., Thélot, B. and Laumon, B. (2012) Bi-cycle helmet wearing and the risk of head, face, and neck injury: a french case-control study based on a road trauma registry, *Injury Prevention*, **18**, 27–32.
- Angwin, J., Larson, J., Mattu, S. and Kirchner, L. (2016) Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks., ProPublica website, proPublica.
- Bender, E. M., Gebru, T., McMillian-Major, A. and Shmitchell, S. (2021) On the dangers of stochastic parrots: Can language models be too big?, in *Conference on Fairness, Accountability, and Transparency (FAccT '21)*, pp. 610–623.
- Bonten, M. J., Huijts, S. M., Bolkenbaas, M., Webber, C., Patterson, S., Gault, S., van Werkhoven, C. H., van Deursen, A. M., Sanders, E. A., Verheij, T. J., Patton, M., McDonough, A., Moradoghli-Haftvani, A., Smith, H., Mellelieu, T., Pride, M. W., Crowther, G., Schmoele-Thoma, B., Scott, D. A., Jansen, K. U., Lobatto, R., Oosterman, B., Visser, N., Caspers, E., Smorenburg, A., Emini, E. A., Gruber, W. C. and Grobbee, D. E. (2015) Polysaccharide conjugate vaccine against pneumococcal pneumonia in adults, *New England Journal of Medicine*, **372**, 1114–1125, pMID: 25785969.
- Bottou, L., Curtis, F. and Nocedal, J. (2018) Optimization methods for large-scale machine learning, *SIAM Review*, **60**, 223–311.
- Boyd, D. and Crawford, K. (2012) Critical questions for big data, *Information, Communication & Society*, **15**, 662–679.
- Correia, S., Luck, S. and Verner, E. (2020) Pandemics depress the economy, public health interventions do not: Evidence from the 1918 flu, Tech. rep., SSRN.
- Cripton, P. A., Dressler, D. M., Stuart, C. A., Dennison, C. R. and Richards, D. (2014) Bicycle helmets are highly effective at preventing head injury during head impact: Head-form accelerations and injury criteria for helmeted and unhelmeted impacts, *Accident Analysis & Prevention*, **70**, 1 – 7.

- Deming, D. J. (2017) The growing importance of social skills in the labor market, *Quarterly Journal of Economics*.
- Ferdinands, J. M., Olsho, L. E. W., Agan, A. A., Bhat, N., Sullivan, R. M., Hall, M., Mourani, P. M., Thompson, M. and Randolph, A. G. (2014) Effectiveness of Influenza Vaccine Against Life-threatening RT-PCR-confirmed Influenza Illness in US Children, 2010–2012, *The Journal of Infectious Diseases*, **210**, 674–683.
- Fyhri, A., Sundfør, H., Weber, C. and Phillips, R. (2018) Risk compensation theory and bicycle helmets – results from an experiment of cycling speed and short-term effects of habituation, *Transportation Research Part F: Traffic Psychology and Behaviour*, **58**, 329 – 338.
- Galton, F. (1886) Regression towards mediocrity in hereditary stature., *The Journal of the Anthropological Institute of Great Britain and Ireland*, **15**, 246–263.
- Greene, W. H. (2003) *Econometric Analysis*, Prentice Hall.
- Hubble, E. (1929) A relation between distance and radial velocity among extra-galactic nebulae, *Proceedings of the National Academy of Sciences*, **15**, 168–173.
- Jacobs, A. Z. and Wallach, H. (2021) Measurement and fairness, *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*.
- Kleinberg, J., Lakkaraju, H., Leskovec, J., Ludwig, J. and Mullainathan, S. (2018) Human decisions and machine predictions, *The Quarterly Journal of Economics*, **133**, 237–293.
- Kleinberg, J., Mullainathan, S. and Raghavan, M. (2016) Inherent trade-offs in the fair determination of risk scores, Tech. rep., arXiv.
- Liao, Q. V., Gruen, D. and Miller, S. (2020) Questioning the ai: Informing design practices for explainable ai user experiences, in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, Association for Computing Machinery, New York, NY, USA, CHI '20, p. 1–15.
- Markel, H., Lipman, H. B., Navarro, J. A., Sloan, A., Michalsen, J. R., Stern, A. M. and Cetron, M. S. (2007) Nonpharmaceutical interventions implemented by US cities during the 1918-1919 influenza pandemic, *JAMA*, **298**, 644–654.
- Matute, H., Blanco, F., Yarritu, I., Díaz-Lago, M., Vadillo, M. A. and Barbería, I. (2015) Illusions of causality: how they bias our everyday thinking and how they could be reduced, *Frontiers in psychology*, **6**, 1–14.
- Mills, N. and Gilchrist, A. (2008) Oblique impact testing of bicycle helmets, *International Journal of Impact Engineering*, **35**, 1075 – 1086.

- Murphy, K. P. (2012) *Machine Learning: A Probabilistic Perspective*, MIT Press, Cambridge, MA.
- of England, B. (2019) Inflation report, february 2019, Tech. rep., Bank of England, London, UK.
- Smith, G. C. and Pell, J. P. (2003) Parachute use to prevent death and major trauma related to gravitational challenge: systematic review of randomised controlled trials., *BMJ*, **327**, 1459–1461.
- Walker, I. (2007) Drivers overtaking bicyclists: Objective data on the effects of riding position, helmet use, vehicle type and apparent gender, *Accident Analysis & Prevention*, **39**, 417 – 425.

Index

- $\mathbb{1}()$, *see* indicator function
- accuracy, 169
- actual outcome, 147
- alternative hypothesis, 35
- and, 211
- angular distance, 93
- associated with, 105
- associated with, 138
- atomic event, 24
- average, *see* mean
- axon, 209
- bag of words, 223
- bagging, 192
- before-after estimator, 153
- Bernoulli process, 44
- bimodal, 8
- bit, 32
- bootstrap, 192
- case-control study, 145
- causal diagram, 139
- cause, 136
- cause-density bias, 163
- central limit theorem, 45
- central tendency, 24
- centroid, 253
- Chebyshev norm, 59
- chessboard norm, 59
- compound event, 18
- conditional expectation, 149
- conditional probability, 20
- confidence interval, 13, 38, 40
- confidence level, 35, 38, 39
- confounding factor, 163
- confounding factors, 140
- confusion matrix, 168
- contributing cause, 136
- convolution, 216
- filter, 217
- kernel, 217
- cosine similarity, 60, 91, 270
- counterfactual, 147
- counterfactual assumption, *see* identifying assumption
- counterfactual outcome, *see* counterfactual
- coverage error, 13
- cross-sectional estimator, 150
- cumulative distribution function, 29
- decision boundary, 178
- degrees of freedom, 11
- dendrite, 209
- dependent variable, *see* outcome variable
- design matrix, 51, 125, 125
- distance metric, 183
- document-term-matrix, 203, 223
- dosis, 135, 137
- double-blind experiment, 142
- DTM, *see* document-term-matrix
- dummies, 113
- effect, 136
- elasticity, 122
- elbow plot, 253
- endogenous variable, *see* outcome variable
- ensemble method, 192
- entropy, 31
- Euclidean norm, 57
- event, 18

expectation, 7, 22, 25, 148
 expected value, *see* expectation
 explanatory variables, 97
 external validity, 13

 F-score, 170
 false negative, 37, 168
 false positive, 37, 168
 false positive rate, 171, 172
 fat tails, 87
 feature normalization, 85
 feature normalization, 226
 feature selection, 245
 feed-forward neural network, 209
 fixed effects, 150

 gradient, 237,
 textbf{283}
 gradient ascent, 234
 gradient descent, 234, 237
 group fairness, 275

 hidden layer, 212

 identifying assumption, 148, 149, 153,
 156
 independent events, 22, 200
 indicator function, ix, 226
 individual fairness, 275
 intersectionality, 119
 interval measure, 2
 iris data, 95

 k -nearest neighbors, 176, 181, 182

 lemma, 222
 lemmatization, 222
 likelihood, 201
 linear regression
 polynomial regression, 177
 linear independence, 56
 linear probability model, 172
 linear regression
 mean squared error, 107
 prediction, 100
 residual, 105
 SSE, *see* sum of squared errors

 sum of squared errors, 106, 107
 total sum of squares, 108
 Lipschitz continuity, 215
 log-likelihood, 202
 logistic function, 215
 loss function, 229, 252
 L_p -norm, 58

 Mahalanobis distance, 88
 majority voting, 182, 192
 manhattan norm, 58
 matrix, 60
 column, 61
 component, *see* element
 condition number, 73, 88
 diagonal, 62
 diagonal matrix, 63
 dimension, 61
 eigenvalue decomposition, 88
 element, 61
 identity matrix, *see* unit matrix
 index, 62
 lower triangle, 62
 multiplication, *see* product
 post-multiplication, 68
 pre-multiplication, 68
 product, 65
 rotation matrix, 74
 row, 61
 square, 62
 square matrix, 63
 symmetric matrix, 63
 trace, 70
 transposition, 64
 unit matrix, viii, 63
 upper triangle, 62
 mean, 7
 mean independence, 149
 measure, 2
 median, 7
 metric, 84
 metric distance, 59
 min-max scaling, 88
 Minkowski norm, *see* L_p -norm
 mode, 8
 MSE, *see* mean squared error

- multimodal, 8
- naive bayes, 198
- nat, 32
- natural experiment, 144
- nearest neighbors, 84, 181
- neural networks
- activation, 210
 - bias, 211
 - input layer, 210
 - perceptron, 209
 - weights, 211
- neuron, 209
- nominal measure, 2
- non-linear optimization, 229
- null hypothesis, 35, 38
- objective function, 231
- observed value, 23
- optimization, 231
- or, 211
- ordinal measure, 2
- outcome, 138
- outcome variable, 97
- outcome-density bias, 163
- overfitting, 177
- p*-value, 36, 38
- p.m.f, *see* probability mass function
- padding, 220
- paired data, 43
- PCA, *see* principal component analysis
- penalty, 247
- percent, 3
- percentage point, 4
- pooling, 219
- population, 11
- population variance, 11
- precision, 169, 197
- principal component analysis, 255
- probability density function, 25
- probability mass function, 28
- pruning, 192
- quantile, 39
- quasi-experiment, 144
- R-squared, 108
- R^2 , 108
- random variable, 22, 38
- randomized controlled trial
- see* RCT, 142
- ratio measure, 2
- RCT, 142
- realization, 23, 38
- recall, 169, 197
- regression
- constant, *see* intercept
 - cross-effect, *see* interaction effect
 - deviation, 101
 - error term, 98, 112
 - explanatory variable, 112
 - interaction effect, 115, 161
 - interaction term, 116
 - intercept, 100, 102
 - log transform, 120
 - log-log transform, 122
 - multiple regression, 110
 - outcome variable, 112
 - residual, 101
 - slope, 100, 102
 - standardized features, 118, 120
- regularization, 245
- ReLU, 215
- RMSE, *see* root mean squared error
- robust statistic, 7
- ROC curve, 172
- root mean squared error, 108, 178
- RV, *see* random variable
- sample, 11
- sample space, 18
- sample variance, 9
- sampling error, 13
- scalar, 52
- self-selection, 139, 140
- sensitivity, 171
- sigmoid function, 215
- significance level, 36, 38
- simple event, 18
- softmax, 215
- specificity, 171
- standard deviation, 9

standard error, **9**
 statistical hypothesis, **34**
 statistical model, **96**
 stemming, **222**
 stopwords, **223**, 225
 stride, **220**

 t -statistic, **36**
 t -value, **104**
 tesseract, **77**
 test statistic, **36**, 38
 TF-IDF, **226**
 token, **222**
 tokenization, **222**
 treatment, **138**
 true negative, **168**
 true positive, **168**
 true positive rate, **171**, 172
 type-I error, **37**, 38, 168
 type-II error, **37**, 38, 168

 uniform distribution
 discrete, **29**
 unimodal, **8**

 variance, **11**, **26**
 vector, **50**
 column vector, **64**, 69
 component, *see* element
 dimension, **51**
 element, **50**
 inner product, 69
 linear combination, **55**
 norm, **57**, **58**, 91
 outer product, 69
 row vector, **64**
 row vector, 69
 vector space, **54**
 dimension, **55**
 Venn diagram, **19**
 vocabulary, **203**, 224

 word embeddings, **53**

 xor, **211**