Introduction

Hiroshi Shimodaira[*]

January-March 2020

Prerequisites You should have taken Informatics 1; in particular Inf1 — Data and Analysis is relevant to this thread. We use some Calculus, Linear Algebra, and Probability in this course, but we will revise and refresh what is needed as we go along.

Books There is no required textbook, but some good books include:

- Simon Rogers and Mark Girolami, *A First Course in Machine Learning*, 2nd Ed., CRC, 2016.
- Tom Mitchell, *Machine Learning*, McGraw Hill, 1997.
- Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2003.

Some more advanced books include:

- Ian H. Witten and Eibe Franke, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2005.
- Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
- David J.C. MacKay, *Information Theory, Inference, and Learning Algorithms*, CUP, 2003.
- R Duda, P Hart and D Stork, *Pattern Classification*, 2nd Ed., Wiley, 2001.
- Kevein P. Murphy, *Machine Learning*, The MIT Press, 2012.
- T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*, 2nd Ed., Springer 2009. (`http://statweb.stanford.edu/~tibs/ElemStatLearn`)

This book demonstrates some of the ideas we cover in this course, applied in a ‘Web 2.0’ setting:

- Toby Segaran, *Programming Collective Intelligence*, O’Reilly, 2007.

Online resources Wikipedia (`http://en.wikipedia.org/`) has good coverage of several of the topics that we will cover.

[*] This note is heavily based on notes inherited from Steve Renals and Iain Murray.

1.1 Motivation

Given some data, what might we do with it? In the algorithms and data structures thread you will learn about how to design and analyse algorithms and data structures to *sort* data and to *search* for items in a data set, in a correct, efficient and simple manner. In this thread we are more concerned with algorithms and models that can make *predictions* about new data items, given the data that has already been observed. For example, we may want to:

- assign a new data item to its class (*classification*);
- predict the next item in a sequence (*time series prediction*);
- predict an output given a new input (*regression*).

Léon Bottou (`http://leon.bottou.org/research/largescale`) puts things in an interesting way:

> For the sake of the argument, assume that there are only two categories of computers. The first category, the *makers*, directly mediate human activity. They implement the dialogue between sellers and buyers, run the accounting department, control industrial processes, route telecommunications, etc. The makers generate and collect huge quantities of data. The second category, the *thinkers*, analyse this data, build models, and draw conclusions that direct the activity of the makers.

In this thread we are concerned with the thinkers. Let’s look at some examples.

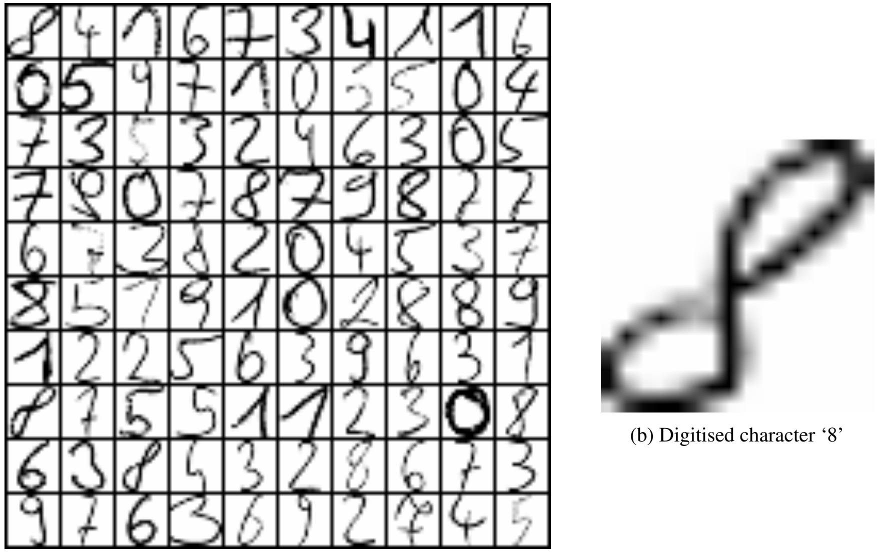
Handwritten digit recognition Figure 1.1a shows some examples of the digits 0–9, written by people, then scanned and digitised. Each digit *image* is associated with one of the ten digit *classes*. If an image of a new digit is observed, the task is to classify it correctly. To do this, we must decide on a representation for the images, and learn a way of associating such a representation to a class. This is a classification problem.

Load forecasting For the electricity industry, being able to predict the demand for power — as far in advance as possible — is extremely important in order to schedule maintenance, buy fuel as cheaply as possible, minimise overhead, and so on. A typical problem is to predict the hourly electricity demand several days in advance. There are many factors to consider such as weather conditions, time of day, day of the week, holidays, and so on. The problems to be addressed are how the factors are to be represented, and given such a representation how to estimate the load. This is a regression problem.

Predicting currency exchange rates Given a history of currency exchange rates (the exchange rate recorded each day, hour, minute, . . .) how accurately can the next value in the time series be predicted?

Recommender systems Given my shopping history, how can an online store recommend items to me that I might want to buy?

Spam Filtering Filter out the unwanted spam mail messages, whilst not losing any of the non-spam messages.



(a) Set of digitised characters

(b) Digitised character ‘8’

Figure 1.1: Handwritten digits, captured by a camera and digitised. This data was collected and distributed by Alex Seewald, `http://alex.seewald.at/digits/`

Speech recognition What is the sequence of words that corresponds to an acoustic signal?

Medical diagnosis Given a set of measurements relating to a patient, what condition should be diagnosed?

Robot navigation Given a set of observations about the world, what actions should a robot take to achieve a goal?

1.2 Data

In this thread we are interested in discovering patterns in data, and using those patterns to make predictions or perform classifications. People are naturally good at exploiting patterns, Witten and Frank give several examples: hunters seek patterns in animal migrations, farmers seek patterns in crop growth, politicians look for patterns in the electorate, lovers look for patterns in their partners’ responses to them. Indeed, the process of science involves finding patterns in data, making predictions based on those patterns, and verifying the predictions.

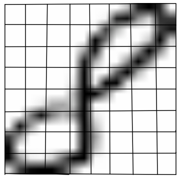


Figure 1.2: Preprocessed digit image on an 8x8 grid

Data comes from many places:

- Sensors such as microphones, cameras, machine monitoring
- Social/economic records such as currency exchange rates, prices
- Artificially created such as possible parses of a sentence, possible object code generated by a compiler

And it needs to be kept in a machine readable electronic form, but it can come in different forms:

- Numeric: numbers such as sound pressure levels obtained from a microphone
- Nominal: categories such as a red, blue and green; or the words in a language
- Ordinal: symbolic data that can be ranked such as grades from ’very easy’ to ’very difficult’

1.2.1 Example: Handwritten digits

As an example let’s look at how we might represent a handwritten digit, such as the digit 8 in Figure 1.1b. First, we need to extract an image of an individual digit, since the raw data is likely to be sequences of characters (e.g., an address on an envelope, a handwritten cheque). This process, called segmentation, is in itself non-trivial! Once the individual digit images are extracted they need to be normalised, for example they need to be scaled to be the same size (imagine that each digit is scaled to fit in a rectangle of a fixed size). These *preprocessing* operations result in a set of segmented and normalised digit images (Figure 1.1a). We will assume that there exists an automatic process that can perform the segmentation and normalisation operations.

Binary representation

Preprocessing removes quite a lot of the variability between images: the location and scale of the images are now the same. Once the images are all fitted to similar sized boxes, the next task is to find a way to represent them in away that is suitable for the process or machine that will associate digit images with the 10 possible classes (0–9). One way to do this is to divide each box into a grid. Figure 1.2 shows the digit image from Figure 1.1b on an 8×8 grid. We could represent each cell in the grid by a binary value: 0 if there is no ink in that location, 1 if there is. This would result in something like the one shown in Figure 1.3 for the example image.

We now have a grid of numbers. How can we represent such a grid? One way is as a 64-bit ($8 \times 8 = 64$) binary number:
0000011100001001000110100001110000111000111100001001100011110000
This has the advantage of being compact, but treating the bits as part of a single number means that

0	0	0	0	0	1	1	1
0	0	0	0	1	0	0	1
0	0	0	1	1	0	1	0
0	0	0	1	1	1	0	0
0	0	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	0	0	1	1	0	0	0
1	1	1	1	0	0	0	0

Figure 1.3: Binary representation of the digit image with an 8x8 grid

some are more significant than others—changing the first bit has a bigger effect than changing the last bit.

Another way is as a 8-by-8 matrix, e.g. $B = (b_{ij}),\ 1 \leq i \leq 8,\ 1 \leq j \leq 8$, so that each element b_{ij} corresponds to the grid (i, j), i.e., the cell at i'th row and j'th column.

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

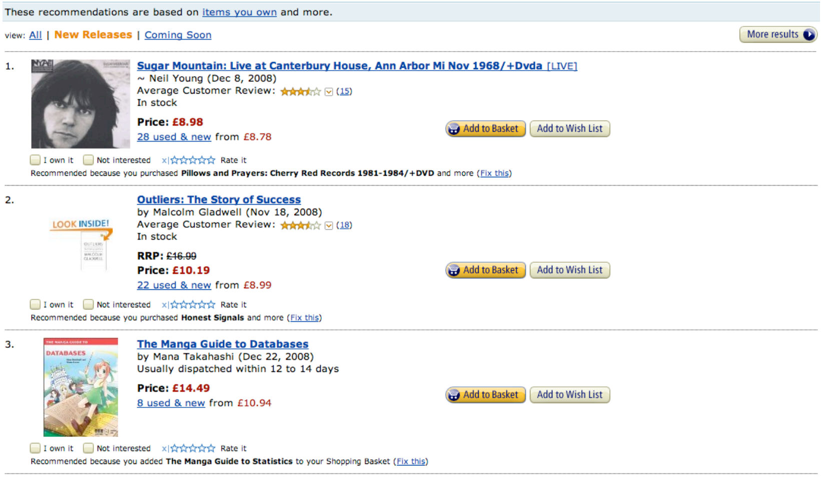
Binary vector representation

Recognising digit images will always require a way of measuring the similarity between images (we assume that images of the same digit class are in some way similar). Thus we would like a representation that tells us that the above binary grid is similar to this one:

0	0	0	0	0	1	1	1
0	0	0	0	1	0	0	1
0	0	0	1	1	0	1	0
0	0	0	1	1	1	0	0
0	0	0	1	1	0	0	0
0	1	1	1	0	0	0	0
1	0	0	1	1	0	0	0
1	1	1	1	0	0	0	0

A natural way to do this is by converting the corresponding 8-by-8 matrix into a 64-element vector [1]: $(0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, \ldots, 0, 0, 0)^T$
Notation: $(\cdot)^T$ means transpose, allowing us to write a column vector on a single line to save space. Having the two images represented as vectors, **a** and **b** for example, we can now tell how they are similar to each other by calculating the distance (e.g. Euclidean distance) between **a** and **b**.

[1] The conversion shown here was done in row-wise, i.e., the first row of the matrix was copied into the vector first, followed by copying the second row, and so on. It could be done in column-wise, of course.



These recommendations are based on items you own and more.

view: All | New Releases | Coming Soon

1. Sugar Mountain: Live at Canterbury House, Ann Arbor Mi Nov 1968/+Dvda [LIVE]
~ Neil Young (Dec 8, 2008)
Average Customer Review: (15)
In stock
Price: £8.98
28 used & new from £8.78

□ I own it □ Not interested ×|☆☆☆☆☆ Rate it
Recommended because you purchased **Pillows and Prayers: Cherry Red Records 1981-1984/+DVD** and more (Fix this)

2. Outliers: The Story of Success
by Malcolm Gladwell (Nov 18, 2008)
Average Customer Review: (18)
In stock
RRP: ~~£16.99~~
Price: £10.19
22 used & new from £8.99

□ I own it □ Not interested ×|☆☆☆☆☆ Rate it
Recommended because you purchased **Honest Signals** and more (Fix this)

3. The Manga Guide to Databases
by Mana Takahashi (Dec 22, 2008)
Usually dispatched within 12 to 14 days
Price: £14.49
8 used & new from £10.94

□ I own it □ Not interested ×|☆☆☆☆☆ Rate it
Recommended because you added **The Manga Guide to Statistics** to your Shopping Basket (Fix this)

Figure 1.4: Automatically generated recommendations from `http://www.amazon.co.uk`

You will have come across 2- and 3-dimensional vectors when you have studied vector geometry. The concept can be extended to higher dimensions and the mathematics (scalar product, distance, and so on) remains the same. We are representing the images in a high-dimensional space (64 dimensions in this case), where each dimension corresponds to a pixel value.

Real-valued vectors

It was not completely accurate to write that we assigned the binary number to each grid location as '0 if there is no ink in that location, 1 if there is'. In fact what we did was to choose a threshold on how much grey colour there was in each location. If a location was above threshold we assigned it value 1, otherwise it was assigned 0.

An alternative to hard thresholding would be to set the value for each location to a real number, between 0 and 1, such that 0 is completely white and 1 is completely black, with numbers in between corresponding to the amount of grey. In this case our vector might look like:
$(0.00, 0.00, 0.00, 0.00, 0.02, 0.55, 0.66, 0.78, 0.00, 0.00, 0.00, 0.01, 0.70, \ldots, 0.00, 0.00, 0.00)^T$

This type of representation is used in many problems, and we will be dealing with such multidimensional vector representations in a lot of this course. They are often referred to as *feature vectors*.

1.2.2 Example: Recommender Systems

You can get recommendations for things like films, games, music by asking your friends and by reading reviews. And you know some of your friends have tastes similar to your own (and some of them don't). But what if there is a new film your friends haven't seen yet? Or if none of your friends have quite the same likes and dislikes as you? A *recommender system* (also called *collaborative filtering*) works by looking at large groups of people and seeing which among them buy or rent or download similar things to you. Such a system can then combines the lists of things that they like, to provide a list of things that you might like.

The film rental company Netflix uses recommender systems to suggest films you might like to watch. They instituted the Netflix Prize (`http://www.netflixprize.com`) to improved the quality of recommender systems. The description provided by Netflix gives a good idea of what recommender systems do, and the framework in which they are constructed:

> Netflix is all about connecting people to the movies they love. To help customers find those movies, we've developed our world-class movie recommendation system: Cinematch. Its job is to predict whether someone will enjoy a movie based on how much they liked or disliked other movies. We use those predictions to make personal movie recommendations based on each customer's unique tastes. And while Cinematch is doing pretty well, it can always be made better.
>
> Now there are a lot of interesting alternative approaches to how Cinematch works that we haven't tried. Some are described in the literature, some aren't. We're curious whether any of these can beat Cinematch by making better predictions. Because, frankly, if there is a much better approach it could make a big difference to our customers and our business.
>
> So, we thought we'd make a contest out of finding the answer. It's 'easy' really. We provide you with a lot of anonymous rating data, and a prediction accuracy bar that is 10% better than what Cinematch can do on the same training data set. (Accuracy is a measurement of how closely predicted ratings of movies match subsequent actual ratings.)

(The bar was finally beaten in summer 2009, after 3 years of competition.)

What does the data used by a recommender system look like? You can think of it as a large matrix of people (recommenders) vs. items (e.g. films). Each cell (i, j) of the matrix gives the value that person i gives to item j. In the case of Netflix, this is a whole number from 1–5, representing the rating that the person gave to the film (1-star to 5-stars); for a website recommender it might be 1 (liked), 0 (no preference), -1 (didn't like); for an online store it might be 2 (bought), 1 (browsed), 0 (didn't buy).

In many cases this matrix is sparse: most people have not seen most films.

1.3 Learning

We are interested in learning from the data that we have observed, in order to make predictions about new data points. But what do we mean by learning? Chambers' dictionary defines the verb *learn* as follows:

> to be informed; to get to know; to gain knowledge, skill, or ability.

It is not obvious to see how this translates to the computational setting. What does it mean to inform a computer? How can we determine or measure whether a computer has gained a skill? Rather than worrying about such philosophical issues, we will use the following definition, based on that in Tom Mitchell's book:

> *Machine learning is the study of models and algorithms that improve automatically through experience.*

There are still some problems with this definition:

1. What do we mean by *improve*?
2. What is *experience*?

For a system to improve then it must have some kind of purpose, for example classifying digit images to the correct digit class. If the system improves, then it classifies more accurately. Putting it another way, it makes fewer errors. In machine learning we need to be able to measure improvement in terms of a mathematical function that may measure the error, or measure how well the system models the data. The choice of this function, and the way the system is modified to improve the function, is the at the core of machine learning.

By experience we mean the observed data. In the case of digit recognition, experience corresponds to a set of images together with their correct classes. In the case of a film recommender system, it is a set of ratings given to films by a large group of users.

1.4 Notation

The following is the default notation used in the course. Note that different versions may be used sometimes for the ease of readability.

x	A scalar
$\mathbf{x}$	A column vector : $\mathbf{x} = (x_1, x_2, \ldots, x_D)^T$, where D is the dimension of vector
$\mathbf{x}^T$	Transpose of vector $\mathbf{x}$, meaning a row vector if $\mathbf{x}$ is a column vector
x_n or x_d	n'th sample (scalar), or d'th element of vector $\mathbf{x}$
$\sum_{n=1}^{N} x_n$	Summation, i.e., $x_1 + x_2 + \ldots + x_N$
$\prod_{n=1}^{N} x_n$	Product, i.e., $x_1 x_2 \cdots x_N$
$\mathbf{x}_n$	n'th sample (vector): $\mathbf{x}_n = (x_{n1}, x_{n2}, \ldots, x_{nD})^T$
$\lVert\mathbf{x}\rVert$	Euclidean norm or L^2 norm: $\lVert\mathbf{x}\rVert = \sqrt{\sum_{d=1}^{D} x_d^2} = \sqrt{\mathbf{x}^T\mathbf{x}}$, also known as magnitude
$\lVert\mathbf{x}\rVert_1$	L^1 norm, i.e. $\sum_{d=1}^{D} \lvert x_d\rvert$
$\mathbf{u} \cdot \mathbf{v}$	dot (inner or scalar) product of $\mathbf{u}$ and $\mathbf{v}$, i.e., $\mathbf{u} \cdot \mathbf{v} = \lVert\mathbf{u}\rVert\,\lVert\mathbf{v}\rVert \cos\theta = \mathbf{u}^T\mathbf{v} = \Sigma_{d=1}^{D} u_d v_d$
$\mathbf{A}$	A matrix: $\mathbf{A} = (a_{ij})$. If $\mathbf{A}$ is a M-by-N matrix, $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_N)$
A_{ij}	Element of matrix $\mathbf{A}$ at i'th row and j'th column, i.e. a_{ij}
$\mathbf{I}$ or $\mathbf{I}_d$	Identity matrix of size d by d (ones on diagonal, zeros off)
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$, i.e. $\mathbf{A}^T = (a_{ji})$
$\mathbf{A}^{-1}$	Inverse of matrix $\mathbf{A}$, i.e. $\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$
$\lvert\mathbf{A}\rvert$ or $\det(\mathbf{A})$	Determinant of matrix $\mathbf{A}$
$\mathrm{tr}(\mathbf{A})$	Trace of matrix $\mathbf{A}$
$\{\mathbf{x}_n\}_1^N$	A set of samples: $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
$\bar{x}$	Sample mean of x
s^2, s_x^2, or $\mathrm{Var}(x)$	Sample variance of x
s_{xy}	Sample covariance of x and y. NB: $s_{xx} = s_x^2$
μ	Population mean
σ^2	Population variance
$\mathbf{\Sigma}$	Population covariance matrix (cf. summation operator $\sum$)
σ_{ij}	(i, j)-element of a covariance matrix, i.e. $\mathbf{\Sigma} = (\sigma_{ij})$. NB: $\sigma_{ii} = \sigma_i^2$
$\exp(x)$	The natural exponential function of x, i.e. e^x
$\ln(x)$	The natural logarithm of x
δ_{ij}	Kronecker delta, i.e. $\delta_{ij} = 1$ if $i = j$, 0 if $i \neq j$
$x \propto y$	x is proportional to y

Exercises (linear algebra and calculus)

Try the following questions without using calculators.

1. Consider two column vectors such that $\mathbf{a} = (1, 2, 3)^T$ and $\mathbf{b} = (-3, 3, -1)^T$.

   (a) Find $\mathbf{a} + \mathbf{b}$.

   (b) Find $\mathbf{a} - \mathbf{b}$.

   (c) Find $||\mathbf{a}||$, $||\mathbf{b}||$, and $||\mathbf{a} - \mathbf{b}||$.

   (d) Find $\mathbf{a}^T\mathbf{b}$.

   (e) Find $\mathbf{a}\mathbf{b}^T$.

   (f) What is the geometric relationship between $\mathbf{a}$ and $\mathbf{b}$?

2. Let $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ and $B = \begin{pmatrix} 4 & -1 \\ -2 & 3 \end{pmatrix}$.

   (a) Find $A - B$.

   (b) Find AB and $A^T B^T$.

   (c) Find BA and $B^T A^T$.

   (d) Find $|A|$, i.e. $\det(A)$.

   (e) Find A^{-1}.

3. Let A is a m-by-n matrix and B is a n-by-ℓ matrix, prove $(AB)^T = B^T A^T$.

4. Let A, B, C are m-by-n, n-by-ℓ, and ℓ-by-k matrices, respectively. Rewrite $(ABC)^T$ in a form without parentheses. What is the dimension of the resultant matrix?

5. Derive a formula for the Euclidean distance between the origin $(0, 0)$ and a line $y = ax + b$, where a and b are arbitrary constants.

6. Find the derivative of each of the following functions.

   (a) $\sum_{n=0}^{N} a_n x^n$, where $a_n, n = 0, \ldots, N$ are constants

   (b) e^{x^3}

   (c) $e^{\cos(x)}$

   (d) $\frac{e^{-x^2}}{x^2}$

   (e) $\frac{1}{1+e^{-x}}$

   (f) x^x