MATH 829: Introduction to Data Mining and Analysis
Overview

Dominique Guillot

Departments of Mathematical Sciences
University of Delaware

February 8, 2016

Supervised vs unsupervised learning

Supervised learning: outcome variable to guide the learning process

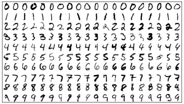
- Set of *input* variables (predictors, independent variables).
- Set of *output* variables (response, dependent variables).
- Want to use the input to predict the output.
- Data is *labelled*.

Unsupervised learning: we observe only the features and have no measurements of the outcome.

- Only have features.
- Data is *unlabelled*.
- Want to detect structure, patterns, etc.

Examples

- Handwritten digits



- You are provided a dataset containing images (16 x 16 grayscale images say) of digits.
- Each image contains a single digit.
- Each image is labelled with the corresponding digit.
- Can think of each image as a vector in $X \in \mathbb{R}^{256}$ and the label as a scalar $Y \in \{0, \dots, 9\}$.
- Idea: with a large enough sample, we should be able to *learn* to identify/predict digits.

Examples (cont.)

Gene expression data: rows = genes, columns = sample.

ESL, Figure 1.3.

- DNA microarrays measure the expression of a gene in a cell.
- Nucleotide sequences for a few thousand genes are printed on a glass slide.
- Each "spot" contains millions of identical molecules which will bind a specific DNA sequence.
- A target sample and a reference sample are labeled with red and green dyes, and each are hybridized with the DNA on the slide.
- Through fluoroscopy, the log (red/green) intensities of RNA hybridizing at each site is measured.

Question: do certain genes show very high (or low) expression for certain cancer samples?

Spam data

- Information from 4601 email messages, in a study to screen email for “spam” (i.e., junk email).
- Data donated by George Forman from Hewlett-Packard laboratories.

TABLE 1.1. *Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between* spam *and* email.

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

ESL, Table 1.1.

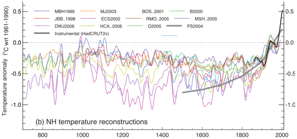
- Each message is labelled as spam/email.
- Want to predict the label using characteristics such as word counts.
- Which words or characters are good predictors?

Note: labelling data can be very tedious/expensive. Not always available.

Examples (cont.)

Inferring the climate of the past:

- We have about 150 years of instrumental temperature data.
- Many things on Earth (proxies) record temperature indirectly (e.g. tree rings width, ice cores, sediments, corals, etc.).
- Want to infer the climate of the past from overlapping measurements.



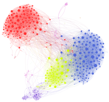
IPCC AR4.

Examples (cont.)

Clustering:

 →  

Wikipedia - Chire.



- Unsupervised problem.
- Work only with features/independent variables.
- Want to label points according to a measure of their similarity.

Modern problems

In modern problems:

- Dimension p is often very large.
- Sample size n is often very small compared to n.

In classical statistics:

- It is often assumed a lot of samples are available.
- Most results are asymptotic ($n \to \infty$).
- Generally not the right setup for modern problems.

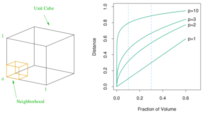
How do we deal with the $p \gg n$ case?

The curse of dimensionality

- Consider a hypercube with sides of length c along the axes in a unit hypercube. Its volume is c^p. To capture a fraction r of the unit hypercube:

$$c^p = r.$$

Thus, $c = r^{1/p}$.
- A small sample of points in the hypercube will not cover a lot of the space.
- If $p = 10$, in order to capture 10% of the volume, we need $c \approx 0.8$!



ESL, Figure 2.6.

The $p \gg n$ case: sparsity

- A modern approach to deal with the $p \gg n$ case is to assume some form of **sparsity** inside the problem.

Examples:

1. Predict if a person has a disease $Y = 0, 1$ given its gene expression data $X \in \mathbb{R}^p$ with p large. Probably only a few genes are useful to make the prediction.
2. The spam data: many of the English words are probably not useful to predict spam. A small (but unknown) set of words should be enough (e.g. "win", "free", "!!!", "money", etc.).

   "You know what Toby, when the son of the deposed king of Nigeria emails you directly, asking for help, you help! His father ran the freaking country! Ok?"

   -Michael Scott, The Office.
3. Climate reconstructions: large number of grid points, few annual observations. Can exploit conditional independence relations within the data.

The $p \gg n$ case: sparsity

A linear regression problem: suppose we try to use linear regression to estimate Y (response) using X (predictors)

$$Y = \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon.$$

- Classical statistical theory guarantees (under certain hypotheses) that we can recover the regression coefficients β if n is large enough (consistency).
- In modern problems n/p is often small.
- What if we assume only a small percentage of the "true" coefficients are nonzero?
- Obtain consistency results when $p, n \to \infty$ with $n/p = \text{constant}$.
- How do we identify the "right" subset of predictors?
- We can't examine all the $\binom{p}{k}$ possibilities! For example, $\binom{1000}{25} \approx 2.7 \times 10^{49}$!

Programming

We will use **Python** to program, analyse data, etc. during the semester.

- Free. Open-source.
- Interpreted.
- Full programming language. Very flexible. Object oriented. Powerful.
- A LOT of scientific packages.

Can use either Python 2.7 or 3.5.

If you have used Python before: make sure you have `numpy`, `scipy`, `matplotlib`, `scikit-learn`.

If you haven't used Python before: I recommend downloading *Anaconda Python* 3.5 from *Continuum Analytics* (https://www.continuum.io/). It's free and comes with a lot of packages already installed.

Warning: If you use a mac, you probably don't want to use the version of Python that came with the computer.

Getting started with Python

Editor:

- Can use Idle.
- Can use IPython + text editor.
- Can use full IDE like Spyder.

- Getting started: very good tutorial at

http://www.scipy-lectures.org/

Take a look at Sections 1.2, 1.3, 1.4.

- A lot of good videos on YouTube.
- If you are a Matlab user: take a look at

http://mathesaurus.sourceforge.net/matlab-numpy.html

- Short intro: `intro_to_python.py`.