Supervised learning: goals and examples

Aaditya Ramdas

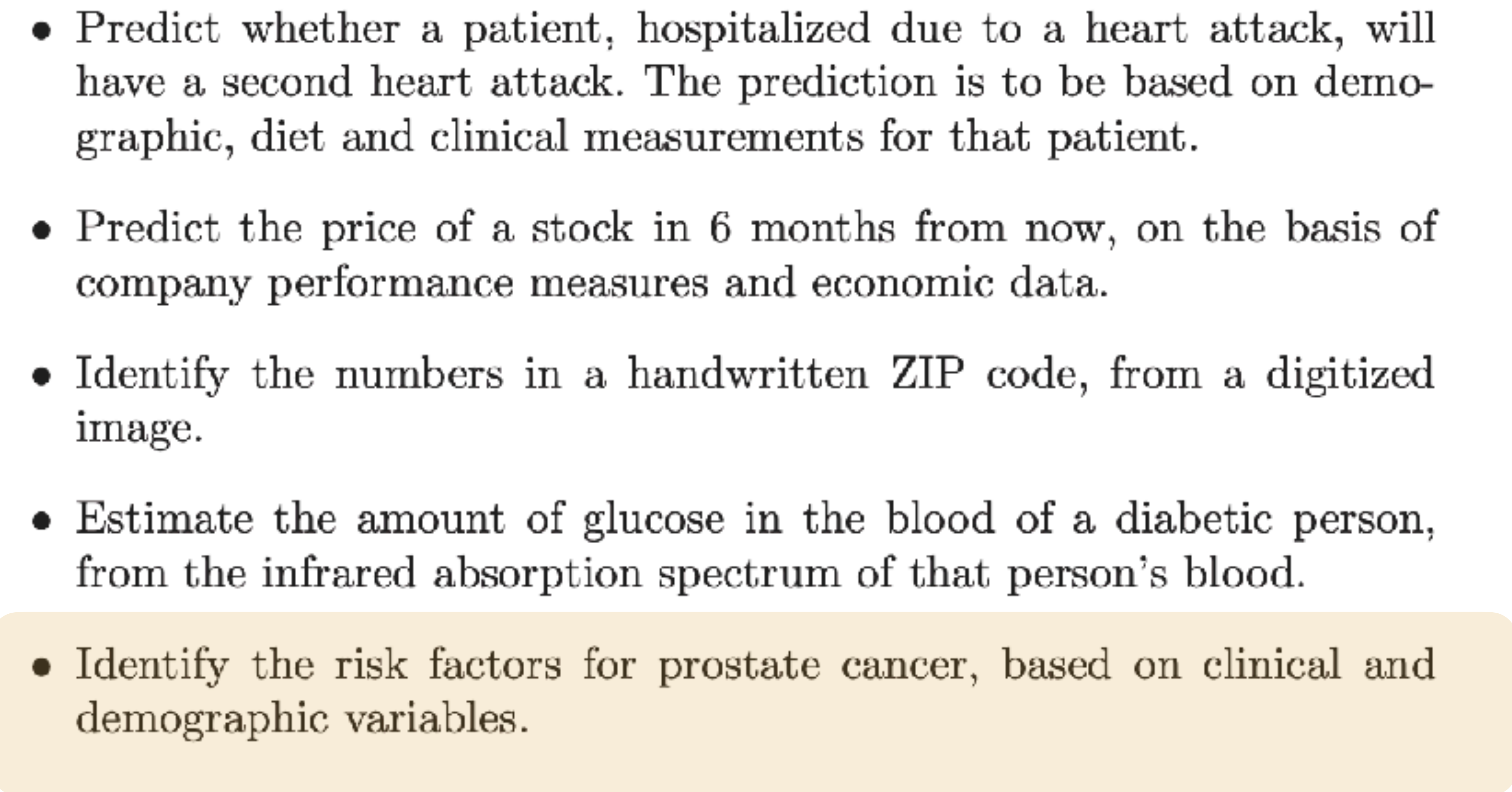
Dept. of Statistics and Data Science
Machine Learning Dept.
Carnegie Mellon University

Outline

1. Examples (1/2 class)

2. Notation (1/2 class)

ESL Chapter 1



- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack. The prediction is to be based on demographic, diet and clinical measurements for that patient.
- Predict the price of a stock in 6 months from now, on the basis of company performance measures and economic data.
- Identify the numbers in a handwritten ZIP code, from a digitized image.
- Estimate the amount of glucose in the blood of a diabetic person, from the infrared absorption spectrum of that person's blood.
- Identify the risk factors for prostate cancer, based on clinical and demographic variables.

Example 1: Email Spam

The data for this example consists of information from 4601 email messages, in a study to try to predict whether the email was junk email, or "spam." The objective was to design an automatic spam detector that could filter out spam before clogging the users' mailboxes. For all 4601 email messages, the true outcome (email type) `email` or `spam` is available, along with the relative frequencies of 57 of the most commonly occurring words and punctuation marks in the email message. This is a supervised learning problem, with the outcome the class variable `email`/`spam`. It is also called a *classification* problem.

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

Our learning method has to decide which features to use and how: for example, we might use a rule such as

$$\begin{array}{ll} \text{if } (\texttt{\%george} < 0.6)\ \&\ (\texttt{\%you} > 1.5) & \text{then } \texttt{spam} \\ & \text{else } \texttt{email}. \end{array}$$

Another form of a rule might be:

$$\begin{array}{ll} \text{if } (0.2 \cdot \texttt{\%you} \ - \ 0.3 \cdot \texttt{\%george}) > 0 & \text{then } \texttt{spam} \\ & \text{else } \texttt{email}. \end{array}$$

Example 2: Prostate Cancer

The data for this example, displayed in Figure 1.1[1], come from a study by Stamey et al. (1989) that examined the correlation between the level of prostate specific antigen (PSA) and a number of clinical measures, in 97 men who were about to receive a radical prostatectomy.

The goal is to predict the log of PSA (`lpsa`) from a number of measurements including log cancer volume (`lcavol`), log prostate weight `lweight`, age, log of benign prostatic hyperplasia amount `lbph`, seminal vesicle invasion `svi`, log of capsular penetration `lcp`, Gleason score `gleason`, and percent of Gleason scores 4 or 5 `pgg45`. Figure 1.1 is a scatterplot matrix of the variables. Some correlations with `lpsa` are evident, but a good predictive model is difficult to construct by eye.

This is a supervised learning problem, known as a *regression problem*, because the outcome measurement is quantitative.

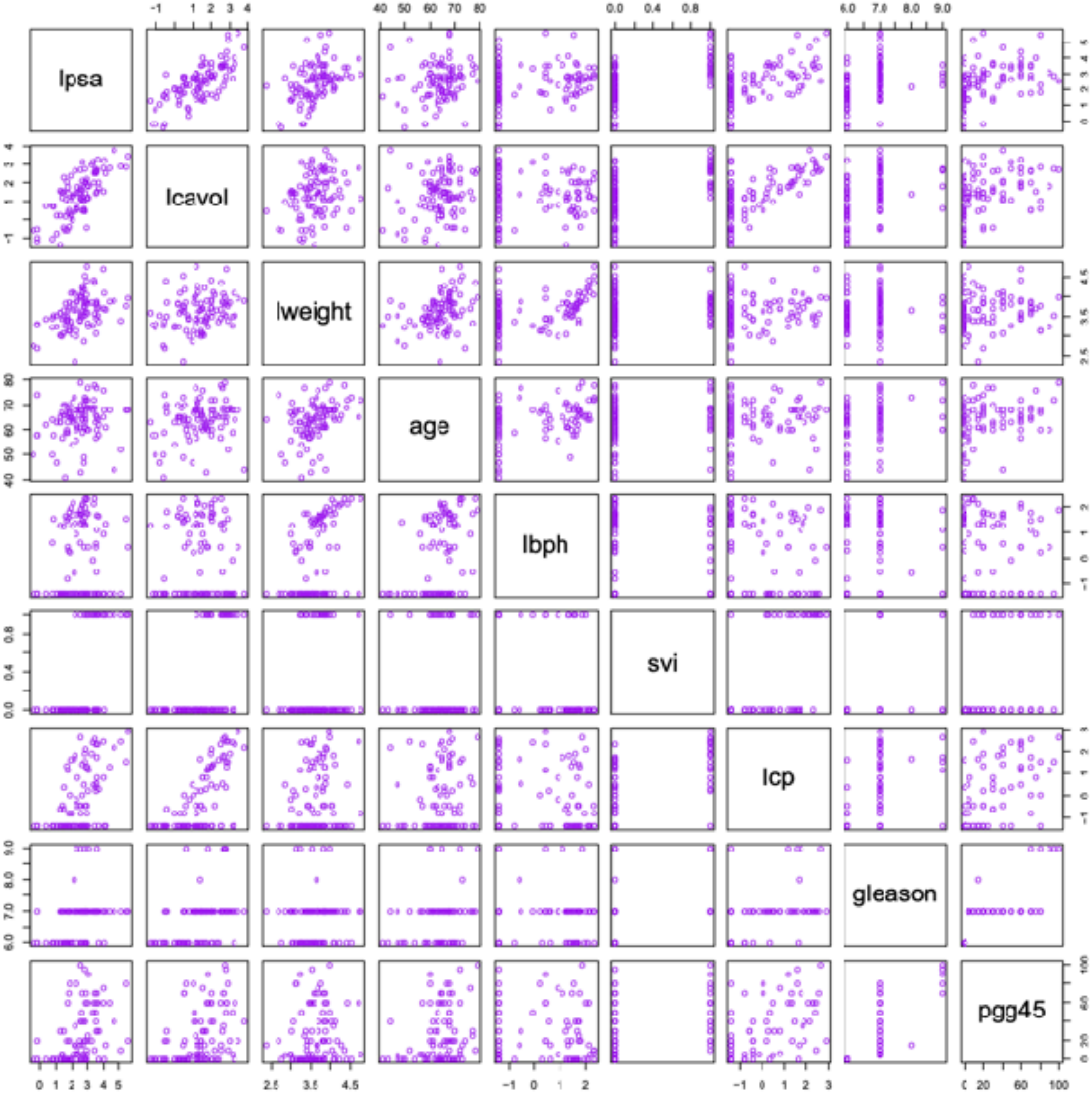
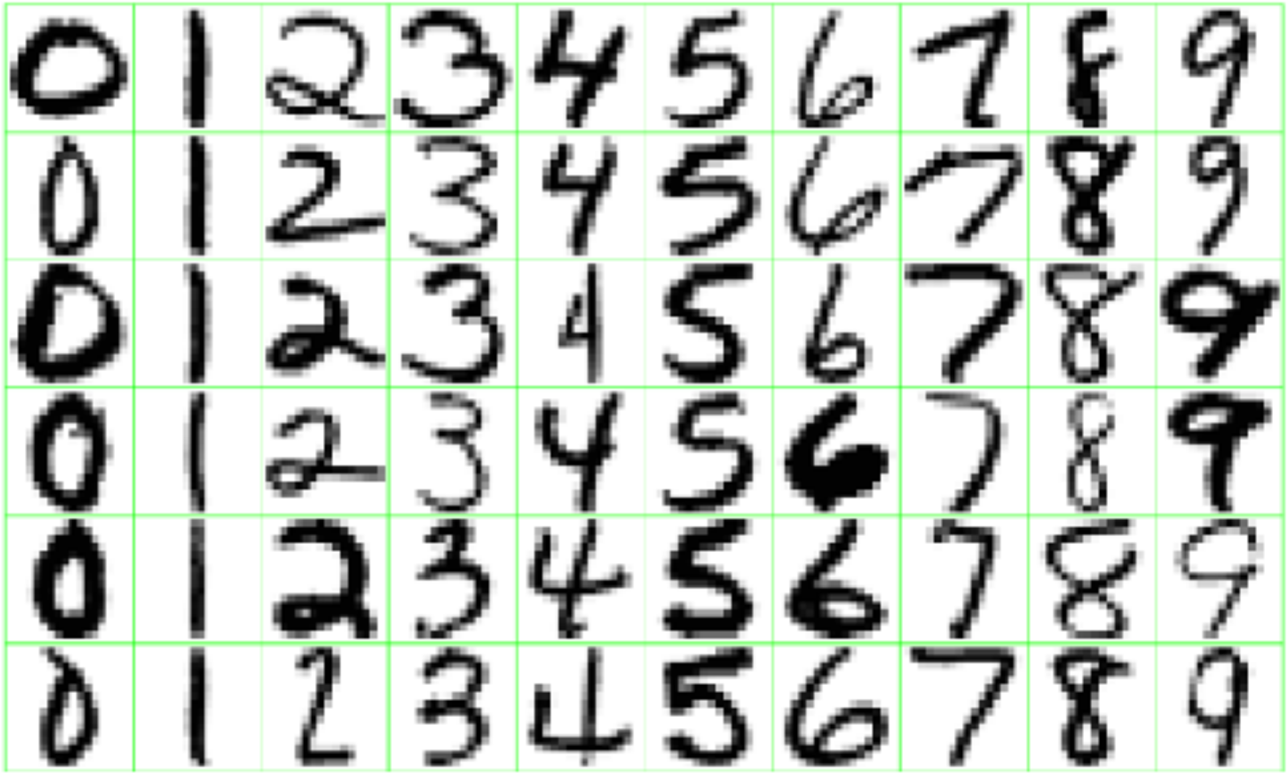


FIGURE 1.1. *Scatterplot matrix of the prostate cancer data. The first row shows the response against each of the predictors in turn. Two of the predictors,* `svi` *and* `gleason`*, are categorical.*

Example 3: Handwritten Digit Recognition

The data from this example come from the handwritten ZIP codes on envelopes from U.S. postal mail. Each image is a segment from a five digit ZIP code, isolating a single digit. The images are 16×16 eight-bit grayscale maps, with each pixel ranging in intensity from 0 to 255. Some sample images are shown in Figure 1.2.

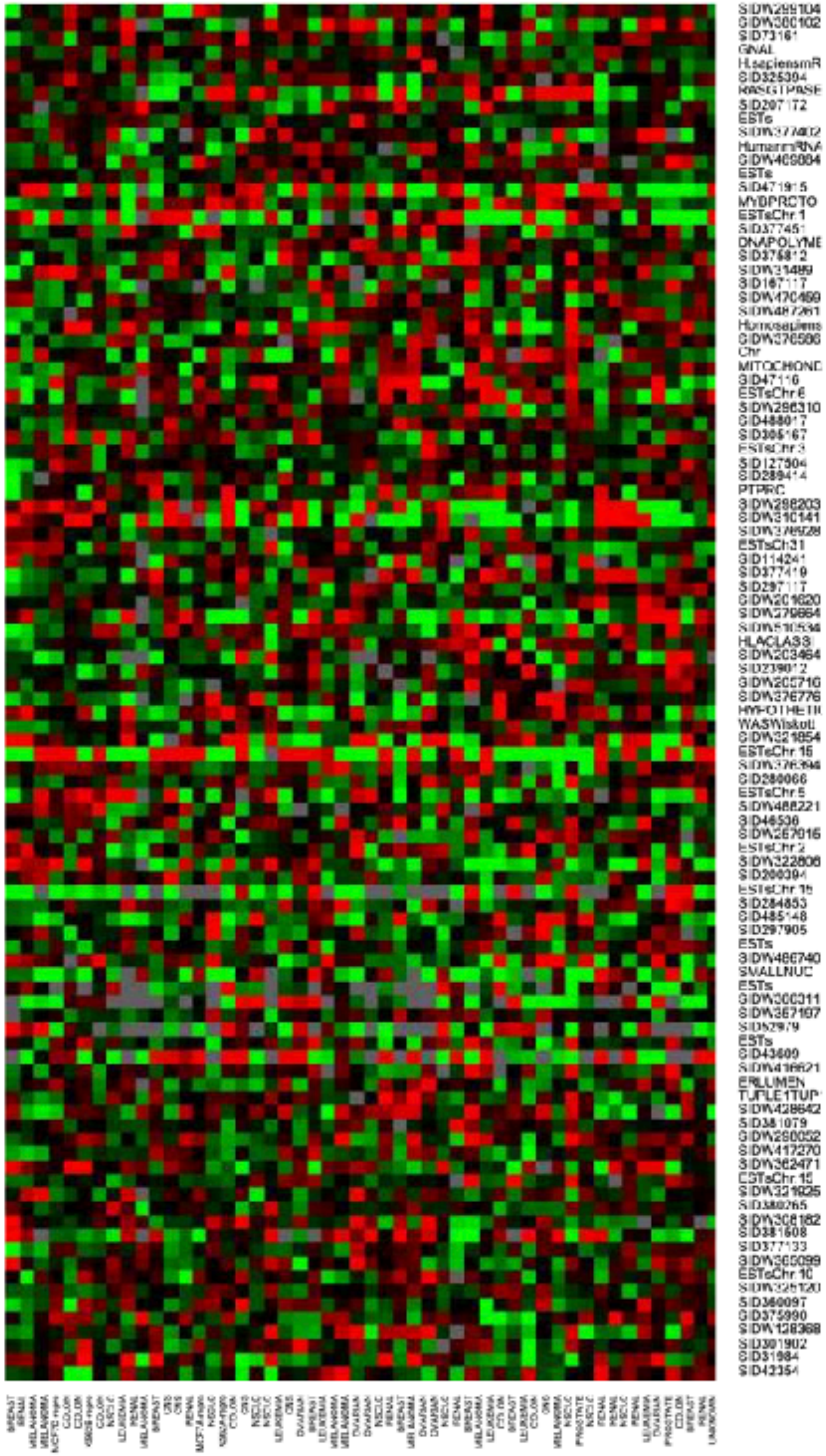
The images have been normalized to have approximately the same size and orientation. The task is to predict, from the 16×16 matrix of pixel intensities, the identity of each image $(0, 1, \ldots, 9)$ quickly and accurately. If it is accurate enough, the resulting algorithm would be used as part of an automatic sorting procedure for envelopes. This is a classification problem for which the error rate needs to be kept very low to avoid misdirection of mail. In order to achieve this low error rate, some objects can be assigned to a "don't know" category, and sorted instead by hand.



Example 4: DNA Expression Microarrays

Here is how a DNA microarray works. The nucleotide sequences for a few thousand genes are printed on a glass slide. 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. The result is a few thousand numbers, typically ranging from say -6 to 6, measuring the expression level of each gene in the target relative to the reference sample. Positive values indicate higher expression in the target versus the reference, and vice versa for negative values.

A gene expression dataset collects together the expression values from a series of DNA microarray experiments, with each column representing an experiment. There are therefore several thousand rows representing individual genes, and tens of columns representing samples: in the particular example of Figure 1.3 there are 6830 genes (rows) and 64 samples (columns), although for clarity only a random sample of 100 rows are shown. The figure displays the data set as a heat map, ranging from green (negative) to red (positive). The samples are 64 cancer tumors from different patients.



The challenge here is to understand how the genes and samples are organized. Typical questions include the following:

(a) which samples are most similar to each other, in terms of their expression profiles across genes?

(b) which genes are most similar to each other, in terms of their expression profiles across samples?

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

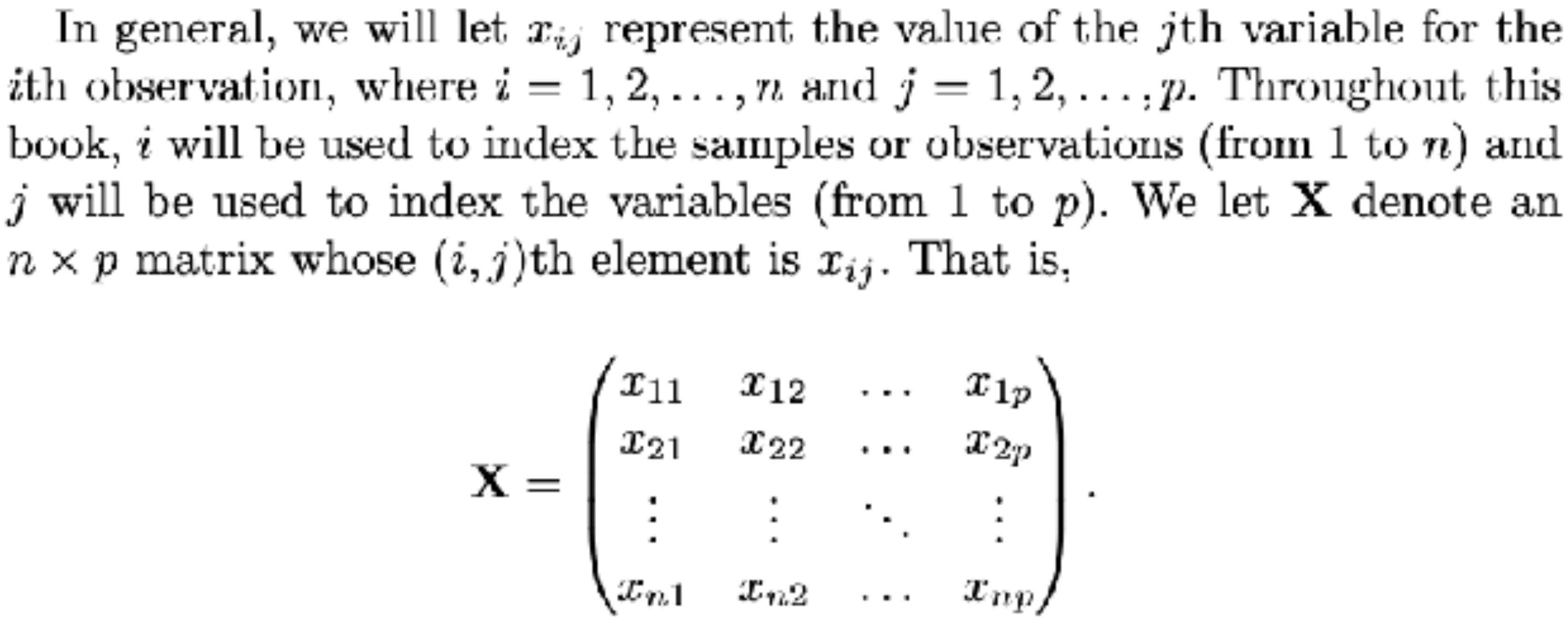
We could view this task as a regression problem, with two categorical predictor variables—genes and samples—with the response variable being the level of expression. However, it is probably more useful to view it as *unsupervised learning* problem. For example, for question (a) above, we think of the samples as points in 6830–dimensional space, which we want to *cluster* together in some way.

Outline

1. Examples (1/2 class)

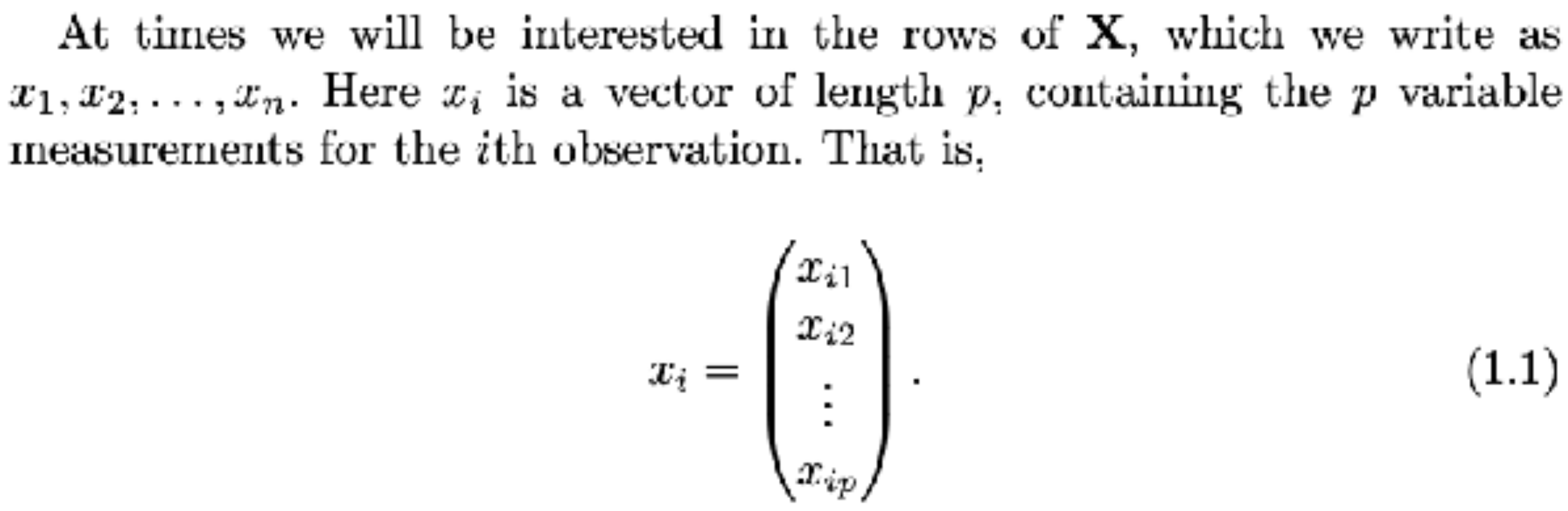
2. Notation (1/2 class)

ISL Chapter 1



In general, we will let x_{ij} represent the value of the jth variable for the ith observation, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, p$. Throughout this book, i will be used to index the samples or observations (from 1 to n) and j will be used to index the variables (from 1 to p). We let $\mathbf{X}$ denote an $n \times p$ matrix whose (i, j)th element is x_{ij}. That is,

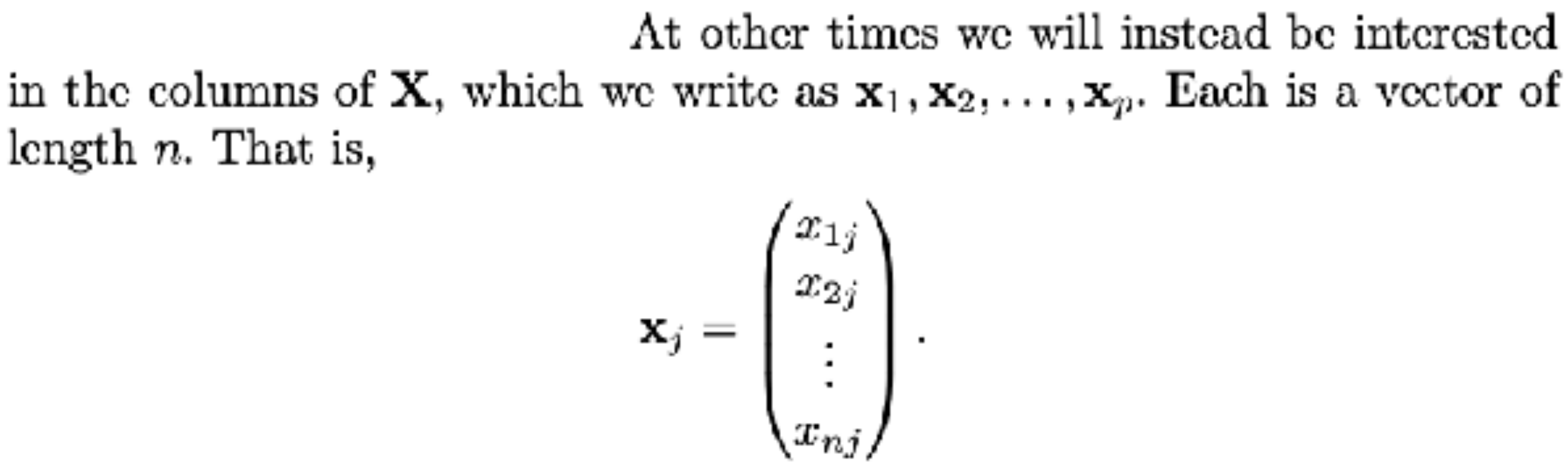
$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1p} \\ x_{21} & x_{22} & \ldots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{np} \end{pmatrix}.$$



At times we will be interested in the rows of $\mathbf{X}$, which we write as $x_1, x_2, \ldots, x_n$. Here x_i is a vector of length p, containing the p variable measurements for the ith observation. That is,

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}. \tag{1.1}$$

(Vectors are by default represented as columns.)



At other times we will instead be interested in the columns of $\mathbf{X}$, which we write as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$. Each is a vector of length n. That is,

$$\mathbf{x}_j = \begin{pmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{nj} \end{pmatrix}.$$

Using this notation, the matrix $\mathbf{X}$ can be written as

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_p \end{pmatrix},$$

or

$$\mathbf{X} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix}.$$

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}.$$

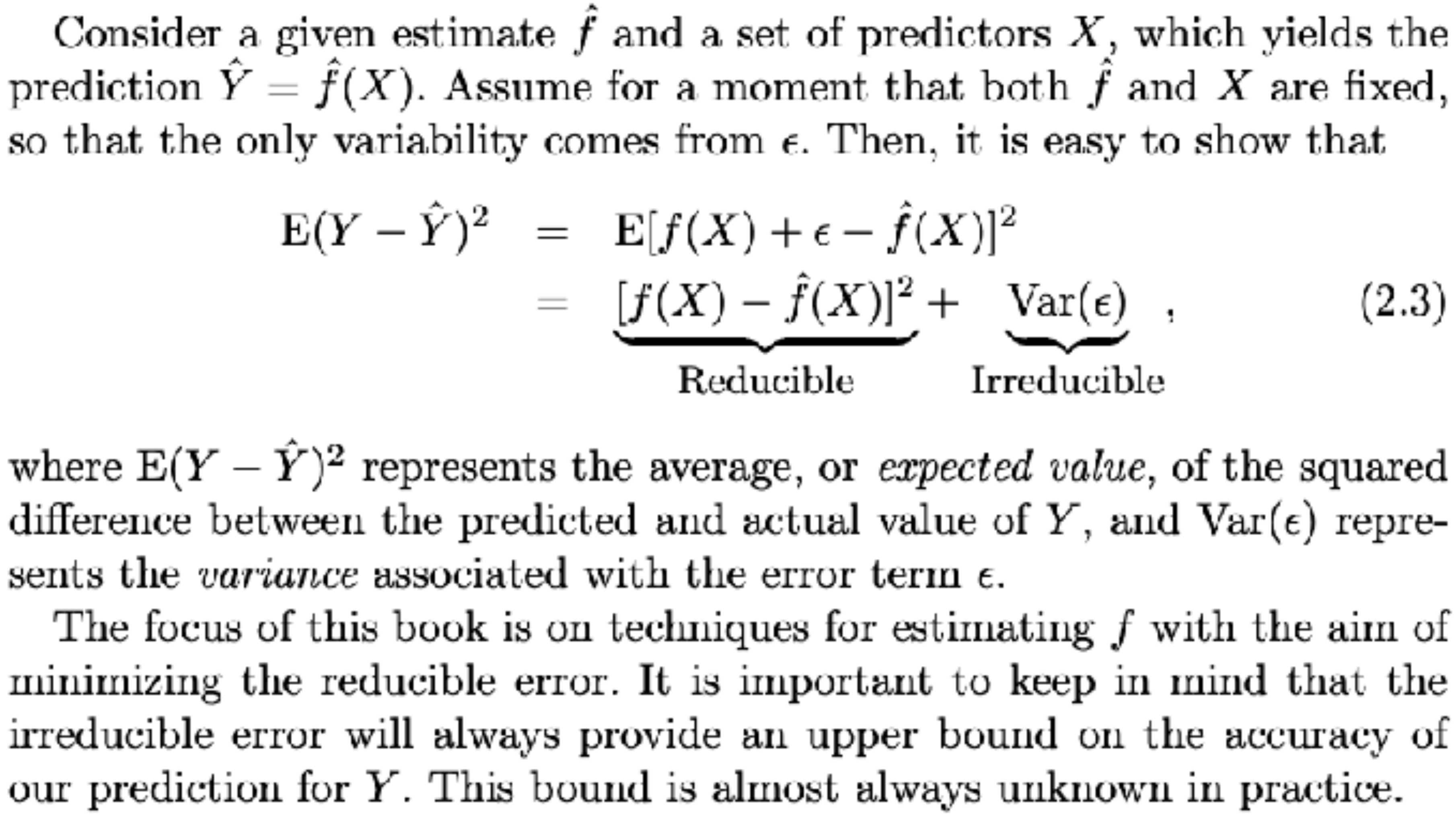
Then our observed data consists of $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, where each x_i is a vector of length p. (If $p = 1$, then x_i is simply a scalar.)

More generally, suppose that we observe a quantitative response Y and p different predictors, $X_1, X_2, \ldots, X_p$. We assume that there is some relationship between Y and $X = (X_1, X_2, \ldots, X_p)$, which can be written in the very general form

$$Y = f(X) + \epsilon. \tag{2.1}$$

Here f is some fixed but unknown function of $X_1, \ldots, X_p$, and ϵ is a random *error term*, which is independent of X and has mean zero. In this formulation, f represents the *systematic* information that X provides about Y.

Mean-squared error of predictions (regression)



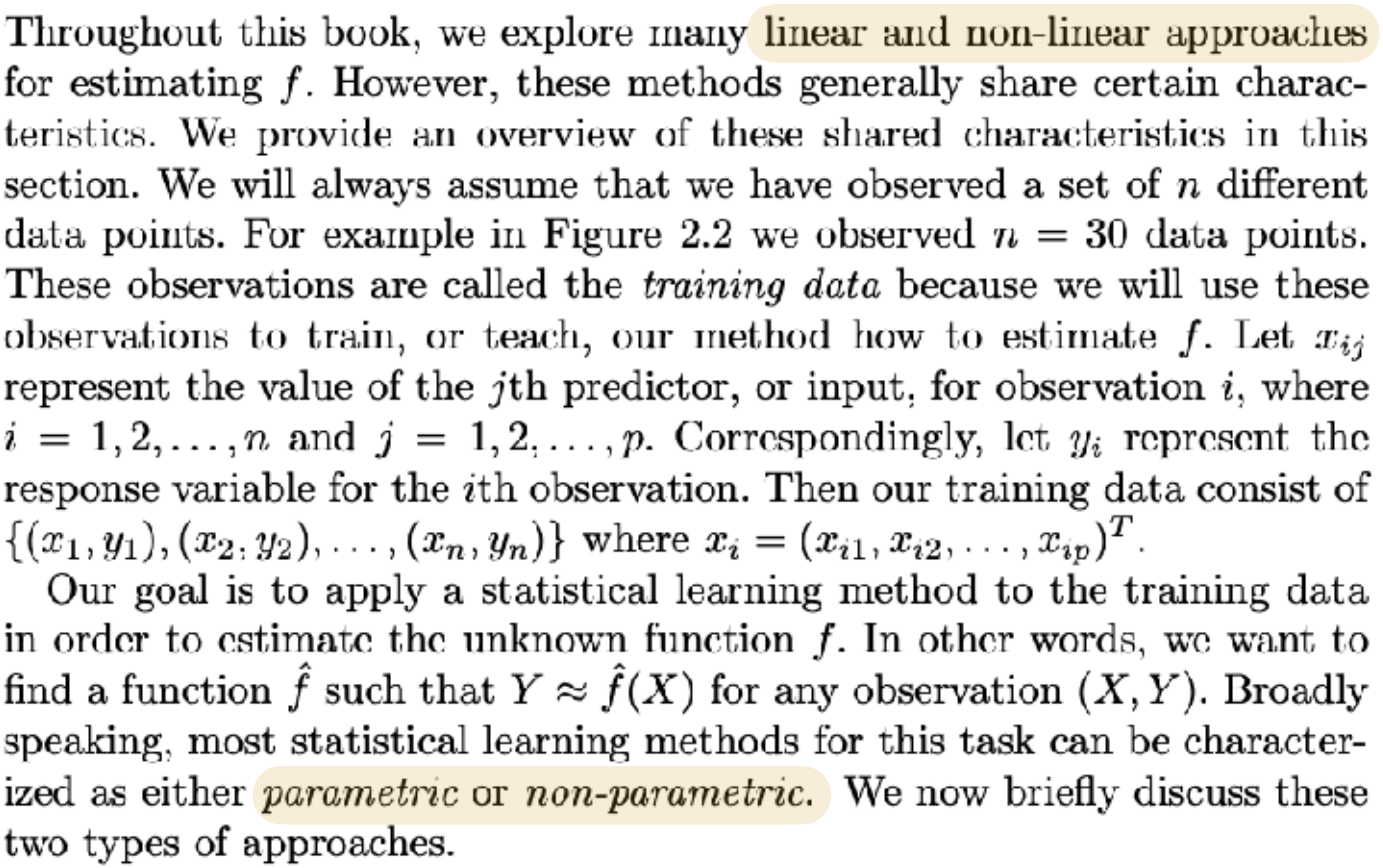
Consider a given estimate $\hat{f}$ and a set of predictors X, which yields the prediction $\hat{Y} = \hat{f}(X)$. Assume for a moment that both $\hat{f}$ and X are fixed, so that the only variability comes from ϵ. Then, it is easy to show that

$$\begin{aligned} \mathrm{E}(Y - \hat{Y})^2 &= \mathrm{E}[f(X) + \epsilon - \hat{f}(X)]^2 \\ &= \underbrace{[f(X) - \hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{Irreducible}} , \end{aligned} \qquad (2.3)$$

where $\mathrm{E}(Y - \hat{Y})^2$ represents the average, or *expected value*, of the squared difference between the predicted and actual value of Y, and $\mathrm{Var}(\epsilon)$ represents the *variance* associated with the error term ϵ.

The focus of this book is on techniques for estimating f with the aim of minimizing the reducible error. It is important to keep in mind that the irreducible error will always provide an upper bound on the accuracy of our prediction for Y. This bound is almost always unknown in practice.

How do we estimate f ?



Throughout this book, we explore many linear and non-linear approaches for estimating f. However, these methods generally share certain characteristics. We provide an overview of these shared characteristics in this section. We will always assume that we have observed a set of n different data points. For example in Figure 2.2 we observed $n = 30$ data points. These observations are called the *training data* because we will use these observations to train, or teach, our method how to estimate f. Let x_{ij} represent the value of the jth predictor, or input, for observation i, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, p$. Correspondingly, let y_i represent the response variable for the ith observation. Then our training data consist of $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$.

Our goal is to apply a statistical learning method to the training data in order to estimate the unknown function f. In other words, we want to find a function $\hat{f}$ such that $Y \approx \hat{f}(X)$ for any observation (X, Y). Broadly speaking, most statistical learning methods for this task can be characterized as either *parametric* or *non-parametric*. We now briefly discuss these two types of approaches.

Parametric Methods

Parametric methods involve a two-step model-based approach.

1. First, we make an assumption about the functional form, or shape, of f. For example, one very simple assumption is that f is linear in X:
$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p. \tag{2.4}$$
This is a *linear model*, which will be discussed extensively in Chapter 3. Once we have assumed that f is linear, the problem of estimating f is greatly simplified. Instead of having to estimate an entirely arbitrary p-dimensional function $f(X)$, one only needs to estimate the $p+1$ coefficients $\beta_0, \beta_1, \ldots, \beta_p$.

2. After a model has been selected, we need a procedure that uses the training data to *fit* or *train* the model. In the case of the linear model (2.4), we need to estimate the parameters $\beta_0, \beta_1, \ldots, \beta_p$. That is, we want to find values of these parameters such that
$$Y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p.$$
The most common approach to fitting the model (2.4) is referred to as *(ordinary) least squares*, which we discuss in Chapter 3. However, least squares is one of many possible ways to fit the linear model. In Chapter 6, we discuss other approaches for estimating the parameters in (2.4).

Non-Parametric Methods

Non-parametric methods do not make explicit assumptions about the functional form of f. Instead they seek an estimate of f that gets as close to the data points as possible without being too rough or wiggly. Such approaches can have a major advantage over parametric approaches: by avoiding the assumption of a particular functional form for f, they have the potential to accurately fit a wider range of possible shapes for f. Any parametric approach brings with it the possibility that the functional form used to estimate f is very different from the true f, in which case the resulting model will not fit the data well. In contrast, non-parametric approaches completely avoid this danger, since essentially no assumption about the form of f is made. But non-parametric approaches do suffer from a major disadvantage: since they do not reduce the problem of estimating f to a small number of parameters, a very large number of observations (far more than is typically needed for a parametric approach) is required in order to obtain an accurate estimate for f.

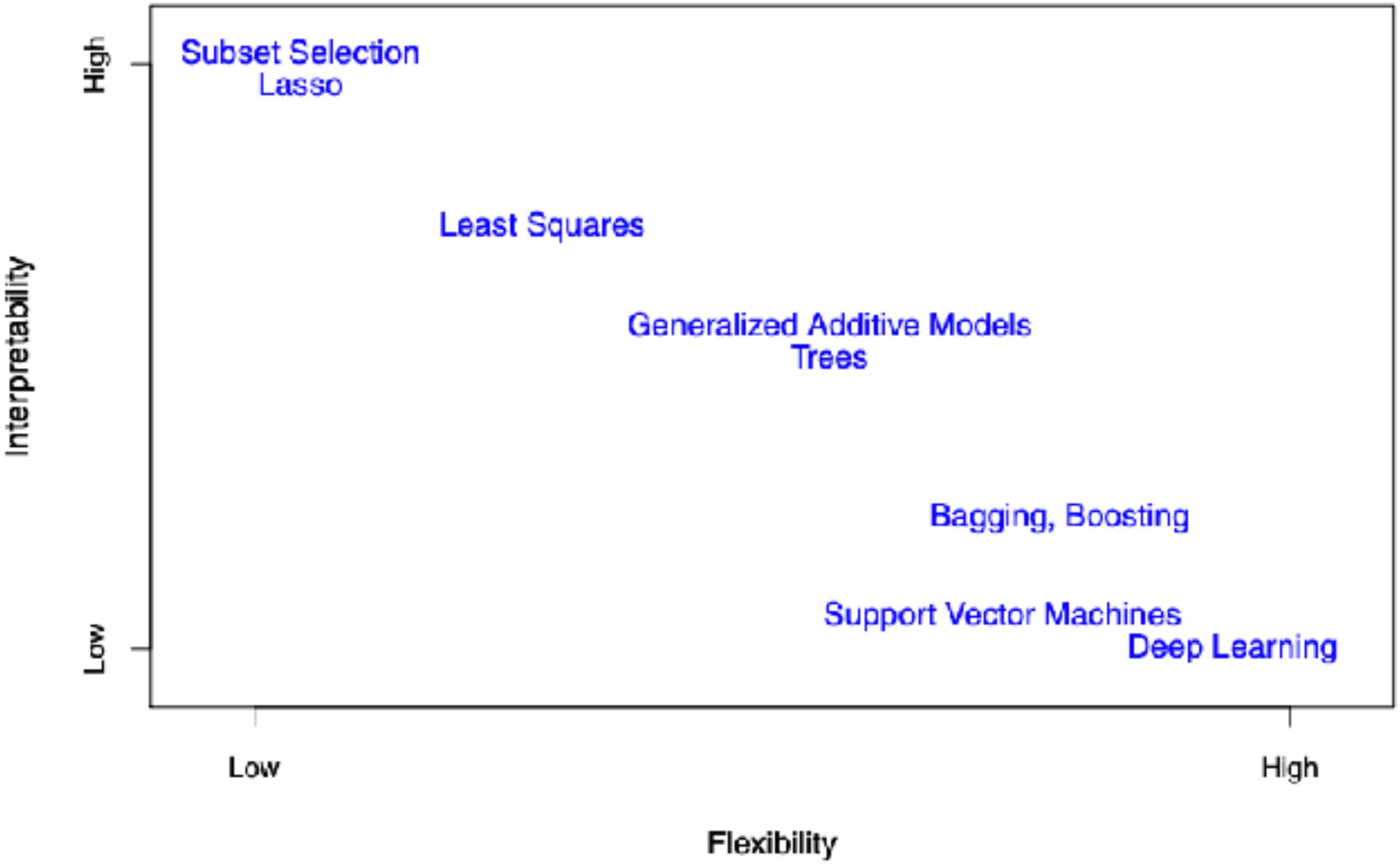


FIGURE 2.7. *A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods. In general, as the flexibility of a method increases, its interpretability decreases.*

Next class

K - nearest neighbors
Bias-variance tradeoffs