types of such heterogeneous data. In cases like these, rather than describing the dataset as a matrix with m rows, we will describe it as a set containing m elements: $\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}\}$. This notation does not imply that any two example vectors $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ have the same size.

In the case of supervised learning, the example contains a label or target as well as a collection of features. For example, if we want to use a learning algorithm to perform object recognition from photographs, we need to specify which object appears in each of the photos. We might do this with a numeric code, with 0 signifying a person, 1 signifying a car, 2 signifying a cat, etc. Often when working with a dataset containing a design matrix of feature observations $\boldsymbol{X}$, we also provide a vector of labels $\boldsymbol{y}$, with y_i providing the label for example i.

Of course, sometimes the label may be more than just a single number. For example, if we want to train a speech recognition system to transcribe entire sentences, then the label for each example sentence is a sequence of words.

Just as there is no formal definition of supervised and unsupervised learning, there is no rigid taxonomy of datasets or experiences. The structures described here cover most cases, but it is always possible to design new ones for new applications.

5.1.4 Example: Linear Regression

Our definition of a machine learning algorithm as an algorithm that is capable of improving a computer program's performance at some task via experience is somewhat abstract. To make this more concrete, we present an example of a simple machine learning algorithm: **linear regression**. We will return to this example repeatedly as we introduce more machine learning concepts that help to understand its behavior.

As the name implies, linear regression solves a regression problem. In other words, the goal is to build a system that can take a vector $\boldsymbol{x} \in \mathbb{R}^n$ as input and predict the value of a scalar $y \in \mathbb{R}$ as its output. In the case of linear regression, the output is a linear function of the input. Let $\hat{y}$ be the value that our model predicts y should take on. We define the output to be

$$\hat{y} = \boldsymbol{w}^\top \boldsymbol{x} \tag{5.3}$$

where $\boldsymbol{w} \in \mathbb{R}^n$ is a vector of **parameters**.

Parameters are values that control the behavior of the system. In this case, w_i is the coefficient that we multiply by feature x_i before summing up the contributions from all the features. We can think of $\boldsymbol{w}$ as a set of **weights** that determine how each feature affects the prediction. If a feature x_i receives a positive weight w_i,