different outcomes. While identifying topics can be helpful, any conclusions you draw from an unsupervised model should be taken with a grain of salt, and we recommend verifying your intuition by looking at the documents in a specific topic. The topics produced by the `LDA.transform` method can also sometimes be used as a compact representation for supervised learning. This is particularly helpful when few training examples are available.

Summary and Outlook

In this chapter we talked about the basics of processing text, also known as *natural language processing* (NLP), with an example application classifying movie reviews. The tools discussed here should serve as a great starting point when trying to process text data. In particular for text classification tasks such as spam and fraud detection or sentiment analysis, bag-of-words representations provide a simple and powerful solution. As is often the case in machine learning, the representation of the data is key in NLP applications, and inspecting the tokens and *n*-grams that are extracted can give powerful insights into the modeling process. In text-processing applications, it is often possible to introspect models in a meaningful way, as we saw in this chapter, for both supervised and unsupervised tasks. You should take full advantage of this ability when using NLP-based methods in practice.

Natural language and text processing is a large research field, and discussing the details of advanced methods is far beyond the scope of this book. If you want to learn more, we recommend the O'Reilly book *Natural Language Processing with Python* by Steven Bird, Ewan Klein, and Edward Loper, which provides an overview of NLP together with an introduction to the `nltk` Python package for NLP. Another great and more conceptual book is the standard reference *Introduction to Information Retrieval* by Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze, which describes fundamental algorithms in information retrieval, NLP, and machine learning. Both books have online versions that can be accessed free of charge. As we discussed earlier, the classes `CountVectorizer` and `TfidfVectorizer` only implement relatively simple text-processing methods. For more advanced text-processing methods, we recommend the Python packages `spacy` (a relatively new but very efficient and well-designed package), `nltk` (a very well-established and complete but somewhat dated library), and `gensim` (an NLP package with an emphasis on topic modeling).

There have been several very exciting new developments in text processing in recent years, which are outside of the scope of this book and relate to neural networks. The first is the use of continuous vector representations, also known as word vectors or distributed word representations, as implemented in the `word2vec` library. The original paper "Distributed Representations of Words and Phrases and Their Compositionality" by Thomas Mikolov et al. is a great introduction to the subject. Both `spacy`

and `gensim` provide functionality for the techniques discussed in this paper and its follow-ups.

Another direction in NLP that has picked up momentum in recent years is the use of *recurrent neural networks* (RNNs) for text processing. RNNs are a particularly powerful type of neural network that can produce output that is again text, in contrast to classification models that can only assign class labels. The ability to produce text as output makes RNNs well suited for automatic translation and summarization. An introduction to the topic can be found in the relatively technical paper "Sequence to Sequence Learning with Neural Networks" by Ilya Suskever, Oriol Vinyals, and Quoc Le. A more practical tutorial using the `tensorflow` framework can be found on the TensorFlow website.

CHAPTER 8
Wrapping Up

You now know how to apply the important machine learning algorithms for supervised and unsupervised learning, which allow you to solve a wide variety of machine learning problems. Before we leave you to explore all the possibilities that machine learning offers, we want to give you some final words of advice, point you toward some additional resources, and give you suggestions on how you can further improve your machine learning and data science skills.

Approaching a Machine Learning Problem

With all the great methods that we introduced in this book now at your fingertips, it may be tempting to jump in and start solving your data-related problem by just running your favorite algorithm. However, this is not usually a good way to begin your analysis. The machine learning algorithm is usually only a small part of a larger data analysis and decision-making process. To make effective use of machine learning, we need to take a step back and consider the problem at large. First, you should think about what kind of question you want to answer. Do you want to do exploratory analysis and just see if you find something interesting in the data? Or do you already have a particular goal in mind? Often you will start with a goal, like detecting fraudulent user transactions, making movie recommendations, or finding unknown planets. If you have such a goal, before building a system to achieve it, you should first think about how to define and measure success, and what the impact of a successful solution would be to your overall business or research goals. Let's say your goal is fraud detection.