to process large amounts of data on a budget, there are two basic strategies: *out-of-core learning* and *parallelization over a cluster*.

Out-of-core learning describes learning from data that cannot be stored in main memory, but where the learning takes place on a single computer (or even a single processor within a computer). The data is read from a source like the hard disk or the network either one sample at a time or in chunks of multiple samples, so that each chunk fits into RAM. This subset of the data is then processed and the model is updated to reflect what was learned from the data. Then, this chunk of the data is discarded and the next bit of data is read. Out-of-core learning is implemented for some of the models in `scikit-learn`, and you can find details on it in the online user guide. Because out-of-core learning requires all of the data to be processed by a single computer, this can lead to long runtimes on very large datasets. Also, not all machine learning algorithms can be implemented in this way.

The other strategy for scaling is distributing the data over multiple machines in a compute cluster, and letting each computer process part of the data. This can be much faster for some models, and the size of the data that can be processed is only limited by the size of the cluster. However, such computations often require relatively complex infrastructure. One of the most popular distributed computing platforms at the moment is the `spark` platform built on top of Hadoop. `spark` includes some machine learning functionality within the `MLLib` package. If your data is already on a Hadoop filesystem, or you are already using `spark` to preprocess your data, this might be the easiest option. If you don't already have such infrastructure in place, establishing and integrating a `spark` cluster might be too large an effort, however. The `vw` package mentioned earlier provides some distributed features and might be a better solution in this case.

Honing Your Skills

As with many things in life, only practice will allow you to become an expert in the topics we covered in this book. Feature extraction, preprocessing, visualization, and model building can vary widely between different tasks and different datasets. Maybe you are lucky enough to already have access to a variety of datasets and tasks. If you don't already have a task in mind, a good place to start is machine learning competitions, in which a dataset with a given task is published, and teams compete in creating the best possible predictions. Many companies, nonprofit organizations, and universities host these competitions. One of the most popular places to find them is Kaggle, a website that regularly holds data science competitions, some of which have substantial prize money attached.

The Kaggle forums are also a good source of information about the latest tools and tricks in machine learning, and a wide range of datasets are available on the site. Even more datasets with associated tasks can be found on the OpenML platform, which

hosts over 20,000 datasets with over 50,000 associated machine learning tasks. Working with these datasets can provide a great opportunity to practice your machine learning skills. A disadvantage of competitions is that they already provide a particular metric to optimize, and usually a fixed, preprocessed dataset. Keep in mind that defining the problem and collecting the data are also important aspects of real-world problems, and that representing the problem in the right way might be much more important than squeezing the last percent of accuracy out of a classifier.

Conclusion

We hope we have convinced you of the usefulness of machine learning in a wide variety of applications, and how easily machine learning can be implemented in practice. Keep digging into the data, and don't lose sight of the larger picture.