1

Open Science in Machine Learning

Mikio L. Braun
TU Berlin

Cheng Soon Ong
National ICT Australia

CONTENTS

1.1 Introduction

The advent of Big Data has resulted in an urgent need for flexible analysis tools. Machine learning addresses part of this need, providing a stable of potential computational models for extracting knowledge from the flood of data. In contrast to many areas of the natural sciences such as physics, chemistry and biology, machine learning can be studied in an algorithmic and computational fashion. In principle, machine learning experiments can be precisely defined, leading to perfectly reproducible research. In fact, there have been several efforts in the past of frameworks for reproducible experiments [13, 30, 3, 15]. Since machine

learning is a data driven approach, we need to have access to carefully structured data [25], that would enable direct comparison of the results of statistical estimation procedures. Some headway has been seen in the statistics and bioinformatics community. The success of R and Bioconductor [8, 9] as well as projects such as Sweave [15], and Org-mode [29] have resulted in the possibility to embed the code that produces the results of the paper in the paper itself. The idea is to have a unified computation and presentation, with the hope that it results in reproducible research [14, 24].

Machine learning is an area of research which spans both theoretical and empirical results. For methodological advances, one key aspect of reproducible research is the ability to compare a proposed approach with the current state of the art. Such a comparison can be theoretical in nature, but often a detailed theoretical analysis is not possible or may not tell the whole story. In such cases an empirical comparison is necessary. To produce reproducible machine learning research, there are three main requirements (paraphrased from [32]):

1. software (possibly open source) that implements the method and produces the figures and tables of results in the paper,
2. easily available (open) data on which the results are computed,
3. and a paper (possibly open access) describing the method clearly and comprehensively.

The approach taken by projects that embed computation into the description may not be suitable for the machine learning community, as the datasets may be large and computations may take significant amounts of time. Instead of a unified presentation and computation document, we propose to have independent interacting units of software, data and documentation of the scientific result.

Motivated by the ideals of the free and open source software movement and current trends for open access to research, we flavour our advocacy for reproducible research with that of open science. It has been shown that researchers who are not experts are likely to find solutions to scientific problems [7]. Corresponding to the three requirements for reproducible research above, we advocate open source software, open data and open access to research. Open source software enables non-experts to use the same tools that engineers in professional organisations use. Open data enables non-experimentalists access to measurements made under difficult and expensive experimental conditions. Open access to research publications enables non-specialists to obtain the same information that scientists in well funded institutions can discover. The long term goal is to make (often publicly funded) results freely available to the general public to maximize the potential impact of the scientific discoveries.

In recent years, there has been a move in machine learning to open science. The theoretical contributions are freely available in an open access journal, the Journal of Machine Learning Research, and from the proceedings of several conferences. This journal also now accepts submissions to a special section on open source software. Furthermore, there is an active community on mloss.org which provides a collection of open source software projects in machine learning. For empirical data, there have been several recent projects such as mldata.org, mlcomp.org, tunedit.org and kaggle.com which provide a more computational focus than the earlier efforts such as the UCI machine learning database. We envision that in the future there will be interdependent units of software, data and documentation which interact based on common protocols.

In the rest of this chapter we will relay our design choices and experiences in organizing open source software in machine learning and open access to machine learning data. We briefly introduce the area of machine learning and how to make it reproducible. After

identifying similarities and differences between reproducible research and open science, we advocate open source software and open data. Finally we discuss several issues which have arisen, such as standards for interoperability, and the tradeoff between automation and flexibility.

1.2 What is Machine Learning?

Lying at the intersection of computation, mathematics and statistics, machine learning aims to build predictive models from data. It is the design and development of algorithms that allow computers to evolve behaviours based on empirical data[1]. A more precise definition is given by [18]: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E". Refer to Figure 1.1 for an illustration for supervised learning. The generality of the approach has found application in various fields such as bioinformatics, social network analysis, and computer vision. It is particularly useful when there are relatively large amounts of data, and the desired outcomes are well defined. For example, in bioinformatics, given the genome sequence of a particular individual one might be interested to predict whether this person is likely to get a certain disease. Instead of more traditional approaches in biochemistry of building mechanistic models of the process at hand, machine learning directly tries to infer the prediction rule (diseased or not) from the data using statistical methods. Naturally, expert domain knowledge is captured in this process by constructing features from the known contributing factors of the disease. There are numerous books describing machine learning in great depth [18, 28, 2, 16, 21, 1], and this section focuses on how machine learning algorithms interact with experimental data, and how it is often used in the natural sciences.

While the techniques of machine learning are widely applicable to data rich applications, machine learning researchers tend to focus on the methodological questions. Experiments in computational science generally have the following workflow [17]:

1. Preprocessing of data, using knowledge of the measurement process to remove artifacts.
2. Feature construction and selection, aiming to capture the contributing factors
3. Model construction and parameter estimation from training data, resulting in a predictor. This is often called "training" or "learning".
4. Preparation of test data. This is often done in parallel to the preparation of the training data above.
5. Evaluation and validation of estimated predictor on the test data.

While each step in this workflow is crucial to the success of the data analysis approach, machine learning tends to focus on step 3: choosing the right computational representation of the problem and estimating the model parameters from available data. One key characteristic of machine learning approaches is the focus on "generalisation error", which is the estimated performance of the trained method on future data. This is because many machine learning approaches are so flexible that they could exactly explain all the training data while capturing nothing about the underlying process. This behaviour is called "overfitting". Hence the importance of step 4 and 5 in the above workflow, in checking the

[1] `http://en.wikipedia.org/wiki/Machine_learning`

performance of the trained method on data which was not used during training. Steps 1 and 2 in the above process require close collaboration with domain experts, and it remains an open question whether this creative process of converting human intuition into precise representations on the computer can be fully automated.

1.2.1 Supervised, Unsupervised and Reinforcement Learning

Machine learning problems can be broadly categorised into three approaches, depending on the type of problem that needs to be solved:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

Supervised learning is used when the training data consists of examples of the input features along with their corresponding target values. The example of disease classification above uses the genome sequence as input features, and the target value is a simple binary "yes/no" label. Depending on the type of label required by the problem, this results in classification, regression or structured prediction. Note that it is often advantageous to consider a representation of input features which are more amenable to computation. In the example above, since there are many common sections between the genomes of different individuals, the whole genome sequence may be processed to identify relevant mutations and the mutations are used as features instead. In recent years, many robust and efficient methods have been developed for this well specified problem.

In other applications, training data may not have corresponding labels. This more exploratory mode of learning is called *unsupervised learning* and is often used to discover structure within the data. The scientist may be interested in discovering groups of similar examples (called clustering), determining the distribution of the data (called density estimation), or finding low dimensional representations (called principle component analysis or manifold learning). In contrast to supervised learning, unsupervised learning methods do not depend on manual human annotation to obtain the target outputs. This allows a higher degree of automation in the data generation process, but the final evaluation of the method and the results becomes considerably more difficult. Naturally, there has been a spectrum of approaches, collectively called semi-supervised learning [5], which try to combine the benefits of both supervised and unsupervised methods.

The approaches above, including the computational science workflow, assume a "passive" application of machine learning. First the data are collected, then the computational approach is applied to analyze the resulting data. However, data collection may be expensive or difficult, and furthermore the experiment may include choosing different conditions. Approaches called *active learning* and *reinforcement learning* are concerned with finding suitable actions to take in a given situation. For example, a classifier may actively choose which individuals it would like to obtain a label for during training. Or a robot may choose the action of moving to a new location before collecting more data. As will be discussed later, these more interactive data collection paradigms pose novel conceptual challenges to reproducible research.

1.2.2 The Role of the Data Set

The data set plays an important role in machine learning, because it typically tackles problems where a formal explanation of the data analysis task is not possible. One could even say that this is one of the distinguishing features of the machine learning approach.

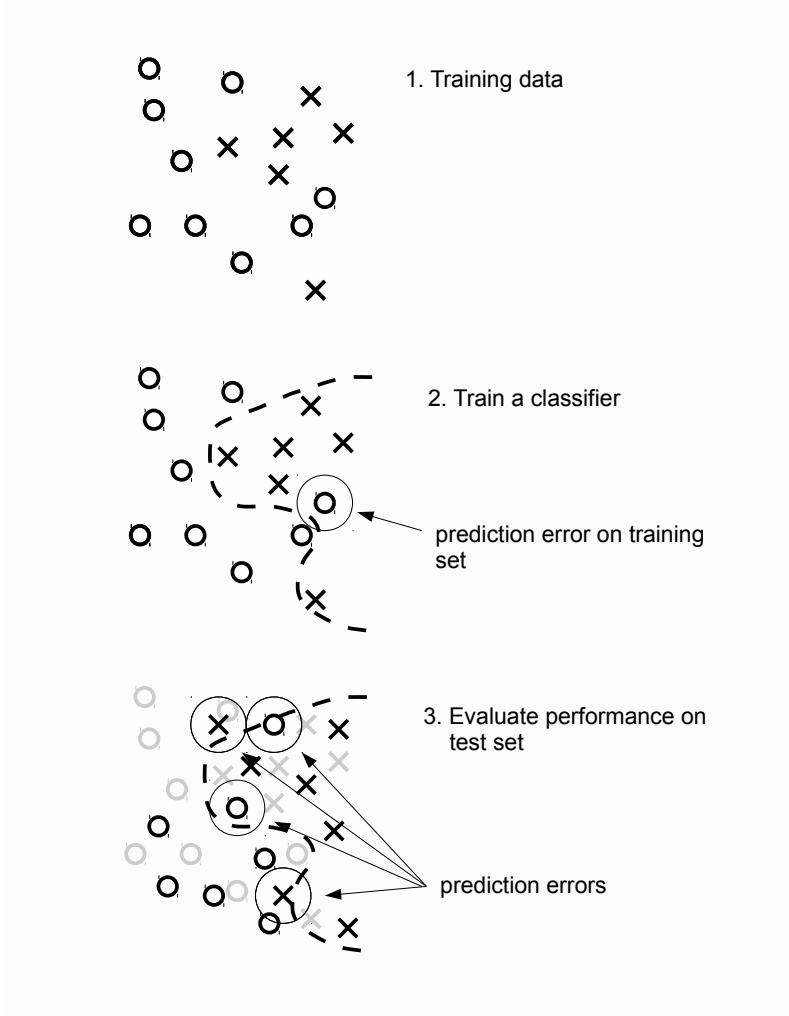

FIGURE 1.1
The different stages in a supervised learning application. The learning data is presented as training data set. On this training set, a classifier is trained to separate the two classes. Already on the training set, the classifier might choose the predict different labels than the ones that are specified if it assumes that there is noise in the data set. The classifier is eventually evaluated on an independent test set not available at training. This test set is used to estimate the expected test error on unseen examples. Again, depending on the level of noise, the test error might be non-zero for the optimal decision boundary. However, if the test error is much larger than the training error, one says that the learning method has *overfitted* to the training data.

In theoretical computer science, for instance, problems are typically formally defined. An introductory problem taught in basic computer science courses is the problem of finding the shortest path in a graph. This can be precisely defined in mathematical terms, and the value of the shortest path can be formally verified. This means that for a given algorithm one can prove that it indeed solves the problem. In addition, one may also prove theorems about other aspects of the shortest path problem like the time it takes to compute a solution.

For many machine learning problems such a definition may not be possible. Consider the problem of handwritten character recognition where the goal is to correctly classify images of handwritten digits. The problem here is that there is no formal specification of what exactly the handwritten, digitized image of a character looks like. The performance of the machine learning algorithm can only be measured relative to the collected dataset, which in the example above the set of images of handwritten numbers.

1.2.3 Applied Statistics, Data Mining, Artificial Intelligence

As with any human defined separation between fields, what is labeled as machine learning [18, 28, 2, 16, 21, 1] instead of applied statistics, data mining or artificial intelligence, tends to have more to do with the community that a researcher belongs to rather than any particular technical difference. Nevertheless, we shall attempt to outline some general trends in the different communities in this section. It also serves as a brief guide to further literature for interested readers.

Applied statistics [6, 10, 34, 4] tends to focus more on theoretical understanding of the statistical properties, and traditionally has been focused on regression type estimation problems. Data mining [22, 36] has a more business oriented origin, and has historically been focused on finding relationships in data in an unsupervised learning fashion. One example is *association rule mining*, which discovers interesting relations between items in databases, and has been popular in market basket analysis. In contrast to machine learning, data mining aims to discover structure in a given dataset, which makes evaluation of the discovery more challenging as it is hard to know the true structure in a given dataset.

Machine learning is often considered to be a sub-field of artificial intelligence [27, 12] where artificial intelligence is concerned with the study and construction of computer algorithms that exhibit intelligent behaviour. In addition to learning, there is significant research on reasoning and planning, which has traditionally been based on mathematical logic. In many real world problems, such as commuting to work, the solution involves multiple steps which have to be in a particular order.

1.3 Machine Learning and Reproducibility

When it comes to reproducibility, an interesting aspect in machine learning is that reproducibility can be achieved to a higher degree by automation than in other sciences. The reason is that all components of the research are available on a computer. Unlike, say, experimental biology, where one has to physically construct an experiment, machine learning is mostly about data.

As explained in the previous section, machine learning is concerned with creating learning methods that perform well on certain application problems, and that can be verified independently by third parties. Therefore, a research result consists of the method found, the data set it has been evaluated on, and a full description of the experimental setup, including feature extraction and estimation of free model parameters. Requiring reproducibility

therefore implies requiring publication of the method, the data, and the experimental set up.

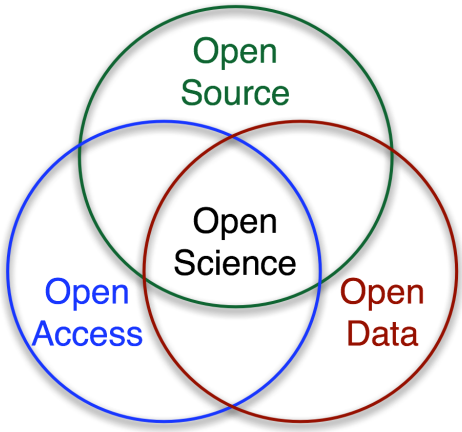

FIGURE 1.2
Open Science = Open Source Software + Open Access Papers + Open Data

In statistics and for easily computable problems, there has been significant progress in reproducible research which follows the vein of literate programming [13]. The proposal is to embed the code for calculating the results of the paper directly in the paper itself, hence simplifying management of code and data, and significantly improving reproducibility of the paper's results. In recent years, there has been a trend in machine learning to analyse large scale data, where a significant investment in time and computational infrastructure is required to produce the results reported in the paper. The model of embedding code and data into the generation of a PDF paper does not scale to such issues, and a restructuring of what is means to have reproducible results is required.

Since data mining has traditionally been applied to business intelligence problems, it has been difficult to obtain access to such private and sensitive data. In artificial intelligence, similar to the challenges faced when trying to reproduce research in reinforcement learning, there are open questions on how to compare and validate solutions. We will discuss this further in the Challenges section.

For machine learning, we believe that by adopting the procedures and concepts of open source, open data, and open access, one can create an environment which supports reproducibility, encourages collaboration between researchers, and removes many of the limitations and delays inherent in the current scientific environment.

1.3.1 Openness

The idea of open source software, which emerged in the 1980s, has interesting connections to the problem of reproducibility, although its motivations and goals are ultimately different. Open source is less about reproducibility, but more an attempt to create a process for collaborative software creation, which is not so different from the way science is organized.

Open source software was the first in a whole series of movements about openness to ease collaboration. Originally, open source software emerged as a countermovement to the increasingly commercial nature of writing software. Software essentially became trade secrets. While open source software was first restricted to academia, it eventually became

1.	Free redistribution
2.	Source code
3.	Derived works
4.	Integrity of the author's source code
5.	No discrimination against persons or groups
6.	No discrimination against fields of endeavor
7.	Distribution of license
8.	License must not be specific to a product
9.	License must not restrict other software
10.	License must be technology-neutral

TABLE 1.1
Attributes of Open Source Software from [23]

a widely accepted alternative to commercial closed source software. The Linux operating system has helped a lot in this respect as it was one of the first large scale pieces of software completed in this way.

The open source model for collaboration has since been copied in other areas as well, of which the Open Data, Open Access, and Open Research movements are the most relevant for our current discussion. As we will discuss in much more detail in the context of open source in the next chapter, the main aspect of these approaches is to create the legal and organizational foundations for collaborative research. The licenses are just one facet of this approach.

So openness is not mainly about reproducibility, but about collaboration. To achieve reproducibility, it would suffice to publish the relevant pieces of information under a classical copyright license, which would allow others to reproduce the results, but would not allow them to directly reuse the code and data.

One could argue whether openness is required for scientific progress, but we believe that the community processes that come with openness significantly simplify scientific collaboration and therefore help to speed up scientific progress as a whole.

1.4 Open Source Software

The basic idea of open source software is very simple, programmers or users can read, modify, and redistribute the source code of a piece of software. The underlying idea is both to make software freely available, and to establish a collaborative community where people contribute to software they find interesting, weeding out bugs if they can, without relying on a software company to take care of this.

The Open Source Initiative (OSI) has compiled a definition of open source according to the criteria listed in Table 1.1. Note that this includes not discriminating against certain persons or groups (by restricting the use to certain countries) or uses (to include non-academic uses). Software which violates any of these requirements is not considered open source. For example, software projects that restrict usage to "non-commercial use only" or "research only" violates OSI definitions and should not be labeled open source.

License	Apache	BSD/MIT	GPL	LGPL	MPL/CDDL	CPL/EPL
Reciprocity	No	No	Yes	Maybe	No	No
Modification release	No	No	Yes	Yes	Yes	Yes
Patent	Yes	No	No	No	Yes	Yes
Jurisdiction	Silent	Silent	Silent	Silent	California	New York
Freedom	PR	Free	PR	PR	Free	PR

TABLE 1.2
The rights of the developer to redistribute a modified product. A comparison of open source software licenses listed as "with strong communities" on `http://opensource.org/licenses/category`. The reciprocity term of GPL states that *if* derivative works from a GPLed licensed software are distributed in binary form, *then* the recipient of the binary form must also be given the source code of the derivative work licensed under the same GPL license. Other important questions are whether the source code to modifications must be released (Modification release); whether it provides an explicit license of patents covering the code (Patent); the legal jurisdiction the license falls under (Jurisdiction); freedom to adapt licence terms (Freedom) (PR = Permission Required from license drafter). Apache: License used by the Apache web server; BSD: License under which the BSD Unix variant is released; MIT: developed by the MIT; GPL/LGPL: (lesser) GNU General Public License; MPL: License used by the Mozilla web browser; CDDL: Common Development and Distribution License developed by Sun Microsystems based on the MPL; CPL: Common Public License published by IBM; EPL: Eclipse Public License used by the Eclipse Foundation, derived from the CPL.

1.4.1 Open Source Licenses

Since traditional copyright is the default in most country jurisdictions, open source software has to come with an explicit copyright license, that gives permissions for others to exercise the exclusive rights of copyright. This permission is sometimes given under certain terms and conditions. Since individual licenses might be hard and costly to enforce, people have quickly begun to use a number of standard licenses. Organizations like the Free Software Foundation (`www.fsf.org`) have also stepped in to defend the GPL, LGPL and AGPL licenses to set a legal precedent.

Table 1.2 collects key features of these licenses. A detailed comparison of the different styles of licences is beyond the scope of this chapter, and the interested reader is refered to other resources [20]. The main differences to consider is whether one wants to ensure that derived work is again open source or not. Also note that there is always the option of releasing the software under a different license by the authors themselves. That way, for example, companies can buy software from the authors by paying for an alternative software license.

Another complication in choosing the right license is that some licenses are not compatible with one another in the sense that one cannot combine two pieces of software which have conflicting licenses because it would then be impossible to satisfy both licenses at the same time.

Generally speaking, the BSD/MIT style licenses are the most admissible. They basically state that you are free to reuse the software as long as the original copyright notices and the license stay intact. The GNU Public License (GPL) requires that any derivative work is also published under the GPL. Variants of this exist like the Lesser GPL (LGPL), which just linking by a non-LGPLed work against a LGPLed library without modifying the library is not defined as a derived work, or the Affero GPL which even extends the notion of derived

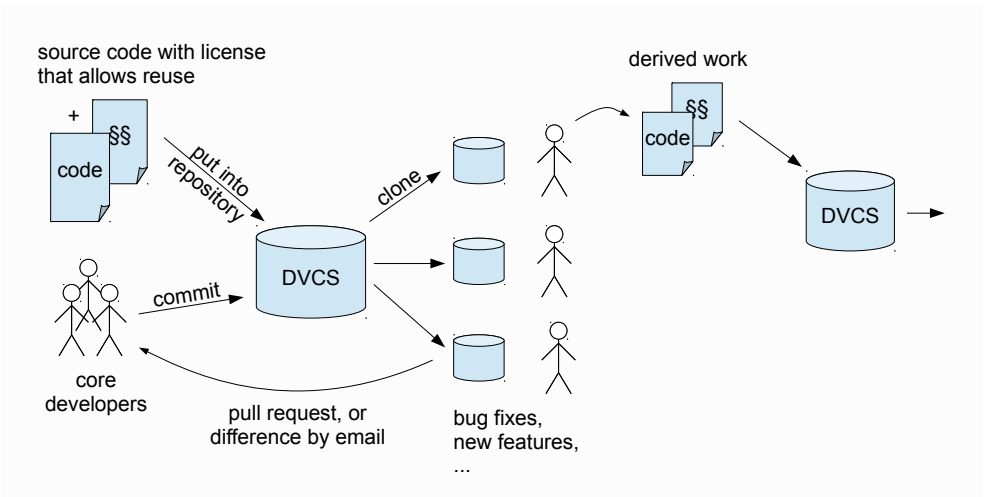

FIGURE 1.3
An overview of the Open Source collaboration model. The source code is released with a license that permits collaboration. Typically, the source code is put into a distributed version control system (DVCS) which tracks changes and lets people access the source code more easily. A group of core developers (also known as "committers") control which changes are incorporated in the main code base. Other users can clone the source code, for example to fix bugs. Changes are offered to the core developers in the form of so-called "pull requests", or by email. The source code can be incorporated into other projects which may again be published under an open source model.

work to include software that uses the original software over some sort of network interface communication.

1.4.2 The Open Source Collaboration Model

As discussed in Section 1.3.1, while openness and reproducibility are not equivalent, the open source software movement has developed a number of standards and processes which are also relevant for reproducible research.

One important component of open source, which is often overlooked, is that open source is not simply a publishing model but comes with a community that emphasizes collaborative work. The book by Raymond [26] gives a good overview over this matter. Figure 1.3 illustrates this process.

Since the source code is freely available and may be modified by others, in principle anyone with the necessary skill and motivation is able to contribute to a project. The level of involvement ranges from fixing minor bugs, to proposing new features, and finally joining the project as a main developer. In order to control the overall direction and consistency and quality of the source code, projects are typically organized into different layers of users. The way this typically works is that anyone may suggest minor fixes or additions, but only a selected number of people are able to actually push changes to the main source tree. Such users are called "committers" from the technical term for adding a revision to a source control system. Within the inner circle, any kind of organization is possible from strictly hierarchical organizations to more or less egalitarian organizations.

Source code version control systems play an important role to facilitate collaboration in open source projects. Two major categories of them are available today: centralized and distributed. A source code version control system is a database which tracks the changes to source code such that one can always revert changes or go back in time to earlier versions easily. Typical examples are CVS (`http://www.nongnu.org/cvs/`) or subversion (`http://subversion.apache.org/`).

One restriction of these systems is that there is only one central data base or repository to track the changes which makes coordination in large multi-module projects difficult. Distributed version control systems (for example, git `http://git-scm.com/`, mercurial `http://mercurial.selenic.com/`, or bazaar `http://bazaar.canonical.com/`) remove this restriction by making it easy to set up local copies of existing repositories (this act is often referred to as "cloning"). People can locally work on the code and offer their changes for integration with the main repository (a process often called "pull request"). In addition, a number of web services exist to give a more user-friendly web interface to these version control systems for managing collaborators, controlling access rights, and organising software projects. Examples include sourceforge (`http://sourceforge.net/`), github (`http://github.com/`), Google code (`http://code.google.com/`), bitbucket (`http://bitbucket.org/`), and many others.

Open source software goes beyond simply making source code available. By publishing the full source codes used in a scientific study, reproducibility can be supported in significant ways. But there are more benefits from the open source approach, which we will discuss in the context of machine learning.

1.4.3 Machine Learning and Open Source Software

By making machine learning methods available to others by source code, fair comparison of methods is much easier. Instead of having to reconstruct algorithms from the descriptions in the papers, one can simply reuse existing software. Just as in open source, exposing all the details of the computation also helps to uncover problems in the methods much more quickly. Adopting an open source approach can help to transfer research results to distant academic disciplines or even the industry much faster because people can build on existing software and integrate them in their own products.

The open source approach has the potential to further transform the way scientists collaborate, because it allows people to share their work much sooner. Traditionally, researchers keep their results private until they are published, based on the misconcieved fear that others may steal their results. This often leads to significant delays because the review process takes months, sometimes years until a paper is published. Adopting an open source approach, researchers could put their work in progress into a version control system that would track individual contributions of researchers and also provide timestamps to resolve precedence between different research groups. Using the open source collaboration model brings us eventually to the Open Research approach where all work in progress is made public to invite collaboration already at an early stage.

A number of different initiatives have been started in recent years to support the use of open source software in machine learning. These efforts are a first step towards making it easier and more rewarding for researchers to publish their code under an open source license.

Originally started as the "Machine Learning Tools Satellite Workshop" in December 2005, researchers in the machine learning community first came together a day before the annual Neural Information Processing Systems conference (`nips.cc`) to discuss possible ways to support machine learning. A year later, a second workshop took place at the NIPS conference, with a closing discussion which led to the position paper "The Need for Open

Source Software in Machine Learning" [31]. Among the obstacles identified against machine learning open source software was that writing software is not considered a scientific contribution in academic circles, and hence there is no incentive for researchers to publish source code. Furthermore, researchers may not be good programmers, and there is entrenched behaviour of reviewers accepting papers which may not be reproducible, where the sloppiness may hide more subtle problems. For more industrial laboratories, there is a common misconception among management that open source software conflicts with commercial interests. In fact open source software *is* commercial software [35], particularly in terms of the federal regulations in the United States of America.

To recognise the contribution of good software in academic currency, a special machine learning open source software track at the Journal of Machine Learning Research (JMLR), and a community website `mloss.org` where people can register their machine learning open source software projects, were created. The main motivation for the special track at the JMLR was to give people a way to publish software. Otherwise it would not seem wise to spend a significant time on publishing software because this effort is not captured in the usual metric used to measure scientific productivity.

Currently, 436 projects are listed on `mloss.org`, and the JMLR has published 45 papers on the special MLOSS track, which demonstrates that the initiative has been very well received by the community. So far, the initiative has been highly successful, but has focused mostly on the "method" side of the problem to make machine learning research more reproducible. Unfortunately, there has been little interaction in terms of common standards and interfaces which would make it much easier to exchange pieces of software.

One problem is also that an open source project has a much different life cycle than a scientific publication. The main difference is that once a paper is published, although one usually continues to research on the topic, the publication stays fixed. A successful open software project, on the other hand, lives on and ideally attracts an active user and developer base. The problem is that an open source project can require a significant amount of work to keep running, which is then not reflected in the above publication model.

1.5 Open Access

In recent years, it has become accepted that open access is a desirable and viable publication model for papers. Open access benefits researchers, institutions, nations and society as a whole. For researchers, it brings increased visibility, usage and impact for their work. Institutions enjoy the same benefits as researchers but in aggregated form. Countries also benefit because open access increases the impact of the research in which they invest public money and therefore there is a better return on investment. Society as a whole benefits because research is more efficient and more effective, delivering better and faster outcomes for us all [2].

Not only is open access a desirable avenue for research output, but it is in fact practical and economically viable. Enabled by low-cost distribution on the Internet, open access literature is digital, online, free of charge, and free of most copyright and licensing restrictions. For example, Creative Commons (`creativecommons.org`) lays out a flexible range of protections and freedoms for authors, artists, and educators. Many journals (more than 8000 according to `www.doaj.org`) have adopted the open access model. In fact NIH supports open access to research funded via its grants, but the publishers are fighting back.

[2] `www.openoasis.org`

As mentioned in the introduction, machine learning has multiple open access publication venues, including its flagship journal the Journal of Machine Learning research, and the proceedings of conferences such as the International Conference on Machine Learning (ICML), Neural Information Processing Systems (NIPS), the Conference on Uncertainty in Artificial Intelligence (UAI) and the International Conference on Artificial Intelligence and Statistics (AISTATS).

1.6 Open Data

Based on the model provided by open source software and open access papers, the approach has been extended to other areas, most notably Open Data (`http://opendatacommons.org/`). As mentioned above, data sets are very important in machine learning because they define learning problems which cannot be defined formally. A new well-designed data set has the potential to spark a completely new line of research.

Historically, machine learning publications mostly focused on new data analysis methods, therefore data sets were often compiled or used for publications which presented new methods. Another typical way to publish data sets consists in organizing a challenge. Here, the challenge organizer puts together a data set, keeping part of the data private and inviting others to develop methods for their data sets during a given challenge runtime. Afterwards, the methods are ranked on the private data based on the published performance measure. The top performing methods are often invited to publish in a special issue of a journal, or in a workshop.

Over time, such data sets are often collected in data set repositories with the goal of making it easier to find relevant data sets and existing results. However, there still is not an open exchange in the same way as there is with source code.

Part of this problem might be that while authorship is usually clear with source code, the number of people involved in data acquisition is often much larger, and often more interdiscplinary. Privacy and legal considerations may be much more complex for data related to people such as medical information.

1.6.1 Machine Learning Data Set Repositories

Recall that open science consists of three components: open source software, open access papers and open data. While mloss.org provides the "method" software, the actual experimental protocols for a particular paper are not available, and neither is the data used for producing the results and figures.

Several repositories focusing on machine learning data sets exist, for example, the UCI machine learning repository (`http://archive.ics.uci.edu/ml/`), or the DELVE repository (`http://www.cs.toronto.edu/ delve/`). These sites have quite a long history, the DELVE site being run since 1995. Both sites host a number of standard benchmark sets which have been used in hundreds of publications.

Still, both sites do not allow for the level of interactivity which would be require to become a main repository for open data exchange. Both sites are rather static, one cannot simply add a data set. The sites contain mostly data sets which are generally considered to be too easy, with the focus lying mostly on regression and binary classification. DELVE in particular has been mostly unmaintained in the last few years.

When designing `mldata.org`, we had the goals in mind to create a community run web site where people can publish data sets. The web site has mechanisms to stimulate

interaction between users, such as tagging, discussions, and ratings. The whole data set can be edited in a wiki-like fashion, such that the community can continually improve the archived data.

Another goal was to provide standardized means for benchmarking. The DELVE repository has been rather ambitious in this respect, but to our knowlegde, it is currently referred to only for the data sets. As we will discuss in more depth in Section 1.7, building and establishing a standard framework for benchmarking data sets is a difficult task, but this problem has to be solved ultimately to make machine learning research reproducible.

Our web site `mldata.org` supports four kinds of information: raw data sets, learning tasks, learning methods, and challenges. A raw data set is just some data, while the learning task also specifies the input and output variables and the cost function used in evaluation. A learning method is the description of a full learning pipeline, including feature extraction and learner. One can upload predicted labels for a data set and a task to create a solution entry which automatically evaluates the error on the predicted labels. Finally, a number of learning tasks can be grouped to create a challenge.

Most of this data is text. We did not attempt a full formal specification of the learning method, but as a first step, we defined a general file exchange format for supervised learning based on HDF5, a structured compressed file format. It is similar to an archive of files but has additional structure on the level of the files, such that users can directly store and access matrices, or numerical arrays. Using the specified file format is not mandatory, but using it unlocks a number of additional features like a summary of the data set, and converting the data set into a number of other formats.

The website went live in 2007. To jump start the community, we uploaded hundreds of freely available data sets. So far, our experience with the web site is mixed. As we will discuss in the final section of this chapter, achieving an acceptable level of interoperability has to be balanced against the complexity of the system. Here, we are still in a process to find the optimal mix.

1.6.2 Business Models around Machine Learning Data Sets

In recent years, other approaches to disseminating data sets have also arrived. The idea is less about open data and providing a service to academia, but more about building a platform between researchers who know data analysis methods on companies which have interesting data.

These recent approaches are often organized around competitions, where the data sets and prize money are provided by companies. One example is kaggle (`http://kaggle.com/`), where companies can set up their own competition. Such web sites became quite popular after the famous Netflix Prize challenge (`http://www.netflixprize.com/`). Netflix, a provider of streaming video, set up a competition where the person who could improve over Netflix existing recommendation algorithm would win one million dollars.

However, the Netflix Prize also highlights some of the dangers of competitions on live business data. Netflix was eventually sued over privacy concerns. Using competition data, researchers seemed to be able to de-anonymize the data sets by correlating the data with other sites. Netflix finally chose not to run a second competition [11].

Ultimately, such changes can also be seen as a cheap way for companies to outsource data analysis work to graduate students in machine learning and related fields. For competitions with many participants, the final prize money might be significantly less than what would have to be paid for the joint work of all participants.

1.7 Future Challenges

We have discussed an approach to support reproducible research in the area of machine learning based on adopting concepts and processes from open source software and extensions. We believe that the combination of open source software, open data, and open access leads to an environment where researchers can efficiently exchange their results and reuse the work of others. Still, a large number of challenges still exist which we will discuss in the following.

1.7.1 Interoperability and Standards

Due to the reasons mentioned above, it is desirable to have independent units of data, software and computational resources that interact with one another. Furthermore, within a particular application pipeline, different parts of the pipeline such as the feature construction and classifier training may be provided by code from different software projects. One major challenge when building a long workflow is to ensure that when replacing a feature construction method with a novel approach, the whole pipeline still functions as expected. This requirement goes beyond simple replicability, but it is necessary in order to build large complex systems capable of solving real world problems. To achieve this, the community would have to agree on certain standards or protocols of communication between the different parts of a data processing pipeline.

As mentioned before `mloss.org` and `mldata.org` are only first steps towards the goal of open science in machine learning. We briefly review several other projects which work towards the same goals.

In the area of statistics, a lot of integration has already been realized in the form of the R programming language[3]. R is an open source reimplementation of the commercial S programming language and is similar in scope to other data analysis centered programming language environments like the commerical Matlab or the Python based scipy. It provides specialized data types for dealing with all kinds of data, and comes with a large library of standard statistical and data analysis functions, as well as libraries for plotting and visualization. In addition, it has a central package repository called CRAN[4] which makes it very easy for researchers to publish their code and for others to install and use it.

In statistics, R is the de-facto standard, meaning that practically all papers also publish their methods in high quality code, often including data sets as well. R also comes with very good documentation support, generating code which can be used with the LaTeX typesetting system, including example code snippets. It is also possible to package data sets together with the code which is a very good way to publish moderately sized data sets.

Finally, there are also the Sweave project[5], and knitr[6] which are systems where you can combine R code with the LaTeX code to typeset your paper such that the paper itself is turned into the code to produce your analysis results, leading to a very high level of reproducibility.

The success of R hinges on the homogeneity of the research community. For machine learning, the set of tools, programming languages, and approaches has always been too diverse to be integrated in the same tight fashion easily. For example, for real-time, or large scale applications, the performance of R is insufficient. Therefore, while the R example shows

[3] `http://r-project.org`
[4] `http://cran.r-project.org`
[5] `http://www.statistik.lmu.de/~leisch/Sweave/`
[6] `http://yihui.name/knitr/`

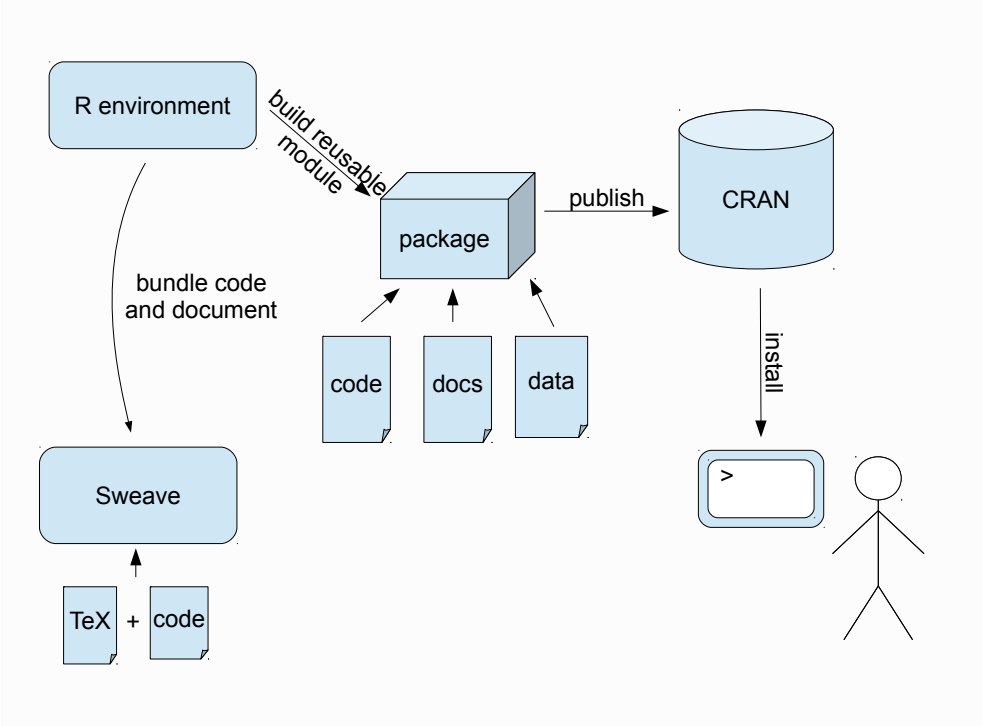

FIGURE 1.4
Overview of the R environment. R provides a very rich environment for data analysis, but one of the main strengths of the system is the central package repository called CRAN, which allows users to publish their code together with documentation and data sets easily, which can then be installed painlessly by other users. Another component are libraries like Sweave or knitr, which let users combine code and the LaTeX code used to typeset document to achieve a very high level of integration and reproducibility.

the benefit of a tightly integrated infrastructure adopted by the majority of the community, achieving this complexity for other areas will typically be much more challenging.

For machine learning, one would have to integrate different programming languages like Matlab, Python, C, or Java, support different data formats and data storage backends like databases, files, web services, and also different operating systems.

One way to approach this problem is to develop formal abstractions and descriptions to encode feature preprocessing and other operations and to provide an interface which others can plug into. Two examples of currently active projects are the ExpML project by Vanschoren et al. [33] who have developed an XML schema for doing exactly this. The goal is to provide support for this XML schema in the major existing machine learning platforms such that experiment descriptions can be automatically executed. The project is also working on setting up an experiment repository in the same sense as `mldata.org` and `mloss.org`. Other examples are `tunedit.org` and `mlcomp.org`. These websites provide computing facilities for people to run their methods on data sets, and to collect benchmark results for a large number of algorithms.

Another project is the "Protocols and Structures for Inference" project by Mark Reid (`http://psi.cecs.anu.edu.au/`) which also defines an interface language for common machine learning interactions. Here, the focus is less on reproducibility and open data, but more on laying the groundwork for improved interoperability between the different pieces of code.

1.7.2 Automation vs. Flexibility

From our experience building `mldata.org`, we observed the following unfortunate trade off between automation and flexibility: We built an general representation of tabular style data in HDF5 which captures many different possible feature types, such as categorical, real valued or strings. Using this HDF5 representation, we could easily automate conversion between several popular machine learning formats such as csv files, libsvm formats, and matlab binaries. However, this led to a large proportion of the datasets on mldata.org being of tabular form, and many users assumed that this was the only acceptable structure.

On the other hand, mldata.org also accepts any file format as a dataset. This flexibility means that we are unable to automatically convert between formats that are convenient for different programming languages, and the dataset is then less appealing to users.

In general, there is always the danger to create something so complex and complicated to make the system practically unusable. A formal description of a machine learning experimental setup quickly evolves to become a full domain-specific language (DSL), just another programming language for the user to learn.

One way to approach this dilemma is to focus on common cases and simple examples first, to keep the system simple and user-friendly. However there will always be cases which cannot be represented in such a system.

1.7.3 Non-static data

In traditional machine learning settings, data is considered in a "batch", that is the whole dataset is available and is fixed. However, in many recent application areas such as social network analysis, there is a stream of data, and the corresponding research area of online learning that continuously updates the predictor has emerged. Defining reproducibility in such a setting is challenging.

Furthermore, in the setting of reinforcement learning, the algorithm has a choice of which data to receive and may even intervene in the environment. Apart from highly contrived simulated examples, it is an open problem on how to define reproducibility in such a setting.

1.8 Outlook

An open letter to the U.S. congress signed by 25 Nobel laureates in 2004 states: *"Open access truly expands shared knowledge across scientific fields, it is the best path for accelerating multi-disciplinary breakthroughs in research."*. This sentiment has extended to open data in recent years, quoting the Open Knowledge Foundation [19]: *"The more data is made openly available in a useful manner, the greater the level of transparency and reproducibility and hence the more efficient the scientific process becomes, to the benefit of society."*. In a data driven field such as machine learning, the easy availability of methods and data are cornerstones of reproducibility and scientific progress.

We believe that open source goes way beyond simply making your source code available to others under a license which invites collaboration, but is in fact a whole process for open collaboration, not unlike science. Science has always favored open collaboration through publication of scientific results. Isaac Newton is attributed with the famous quote *"If I have seen further it is by standing on the shoulders of giants."*, which reflects the importance of sharing scientific results to accelerate scientific growth. The open science model poses an interesting inspiration to transform the scientific progress in the information age.

Acknowledgements

The authors wish to thank Luis Ibanez and Lydia Knüfing for useful comments and criticisms, which resulted in significant improvements in the paper.

1.9 Authors' Bio

Cheng Soon Ong (NICTA) and Mikio L. Braun (TU Berlin) have been working on changing the existing policies in research with respect to publishing software and data since 2005. Along with Soeren Sonnenburg, they have organised several workshops on the matter starting with the Machine Learning Tools Satellite Workshop (December 2005), which has been followed by two NIPS Workshops in December 2006, 2008 and an ICML Workshop in 2010. Together with 13 co-authors, most of them well-known senior researchers in the area of machine learning, they have written the position paper "The Need for Open Source Software in Machine Learning" which appeared in JMLR in October 2007, resulting in the Open Source Software track at JMLR. They also advocate open science through community websites mloss.org and mldata.org.

Cheng Soon Ong is currently a senior researcher at the Bioinformatics Group of NICTA in Australia. His PhD work on kernel methods was completed at the Australian National University in 2005. He had a short postdoc stint at the Statistical Machine Learning Group, in NICTA, Canberra, followed by a longer one at at both the Max Planck Institute of Biological Cybernetics and the Fredrich Miescher Laboratory in Tübingen, Germany. From 2008 to 2011, he was a lecturer at ETH Zurich. Prior to the PhD, he researched and built search engines and Bahasa Malaysia technologies at Mimos Berhad, Malaysia. He has a B.E. (Information Systems) and a B.Sc. (Computer Science) from the University of Sydney, Australia.

Mikio L. Braun received pre-diplomas (B.Sc.) in Mathematics and Computer Science and a diploma (M.Sc.) in Computer Science from the University of Bonn. From 2000 to 2004, he was a research assistant and Ph.D. student in the CVPR group headed by Prof. Buhmann at the University of Bonn. In May 2004, he joined the Intelligent Data Analysis Group headed by Klaus-Robert Müller at the Fraunhofer Institute FIRST in Berlin, Germany. In January 2007, he moved to the Technical University of Berlin branch of the IDA Group. In 2010, he co-founded TWIMPACT, a start-up company focused on real-time analysis of social media data.

Bibliography

[1] David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[2] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] J.B. Buckheit and D.L. Donoho. Wavelab and reproducible research. Technical report, Stanford, 1995.

[4] Peter Bühlmann and Sara van de Geer. *Statistics for High-Dimensional Data*. Springer, 2011.

[5] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, 2006.

[6] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Taylor and Francis, 1994.

[7] Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani, editors. *Perspectives on Free and Open Source Software*. MIT Press, 2007.

[8] Robert Gentleman. Reproducible research: a bioinformatics case study. *Stat Appl Genet Mol Biol.*, 2005.

[9] Robert Gentleman and Duncan Temple Lang. Statistical analyses and reproducible research. Technical Report 2, Bioconductor Project Working Papers, 2004.

[10] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, 2001.

[11] Neil Hunt. Netflix prize update. http://blog.netflix.com/2010/03/this-is-neil-hunt-chief-product-officer.html, March 2010.

[12] Marcus Hutter. *Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability*. Springer, 2010.

[13] Donald E. Knuth. Literate programming. *The Computer Journal*, 27(97), 1984.

[14] Jelena Kovacevic. How to encourage and publish reproducible research. In *ICASSP*, 2007.

[15] F. Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In *COMPSTAT*, pages 575–580, 2002.

[16] Stephen Marsland. *Machine Learning: An Algorithmic Perspective*. CRC Press, 2009.

[17] Jill P. Mesirov. Accessible reproducible research. *Science*, 327:415–416, 2010.

[18] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[19] Jennifer C. Molloy. The open knowledge foundation: Open data means better science. *PLoS Computational Biology*, 9(12):e1001195, 2011.

[20] Andrew Morin, Jennifer Urban, and Piotr Sliz. A quick guide to software licensing for the scientist-programmer. *PLoS Computational Biology*, 8(7):e1002598, 07 2012.

[21] Kevin P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012.

[22] Robert Nisbet, John Elder IV, and Gary Miner. *Handbook of Statistical Analysis and Data Mining Applications*. Academic Press, 2009.

[23] Open Source Initative. `http://www.opensource.org/docs/osd`.

[24] Roger D. Peng. Reproducible research in computational science. *Science*, 334:1226–1227, 2011.

[25] Carl Edward Rasmussen, Radford M. Neal, Geoffrey Hinton, Drew van Camp, Michael Revow, Zoubin Ghahramani, Rafal Kustra, and Rob Tibshirani. `http://www.cs.toronto.edu/ delve/`.

[26] Eric S. Raymond. *The Cathedral and the Bazaar*. O'Reilly Media, 1999.

[27] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition, 2009.

[28] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

[29] Eric Schulte, Dan Davison, Thomas Dye, and Carsten Dominik. A multi-language computing environment for literate programming and reproducible research. *Journal of Statistical Software*, 46(3), 2012.

[30] Matthias Schwab, Martin Karrenbach, and Jon Claerbout. Making scientific computations reproducible. *Computing in Science & Engineering*, 2(6):61–67, 2000.

[31] Sören Sonnenburg, Mikio L. Braun, Cheng Soon Ong, Samy Bengio, Leon Bottou, Geoffrey Holmes, Yann LeCun, Klaus-Robert Müller, Fernando Pereira, Carl Edward Rasmussen, Gunnar Rätsch, Bernhard Schölkopf, Alexander Smola, Pascal Vincent, Jason Weston, and Robert C. Williamson. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8:2443–2466, 2007.

[32] Victoria Stodden. The legal framework for reproducible research in the sciences: Licensing and copyright. *IEEE Computing in Science and Engineering*, 11(1):35–40, 2009.

[33] Joaquin Vanschoren, Hendrik Blockeel, Bernhard Pfahringer, and Geoffrey Holmes. Experiment databases - a new way to share, organize and learn from experiments. *Machine Learning*, 87(2):127–158, 2012.

[34] Larry Wasserman. *All of Statistics*. Springer, 2004.

[35] David A. Wheeler. Open source software is commercial. *Journal of Software Technology*, 14(1), 2011.

[36] Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 3rd edition, 2011.