PERSPECTIVE

Open Access

Data Science for Molecular Diagnostics Applications: From Academia to Clinic to Industry

Mona Riemenschneider,¹ Joachim Wienbeck,² André Scherag,³ and Dominik Heider^{4,*}

Abstract

The development of computational approaches for predictive modeling of diseases, for example, in cancer diagnostics or prognostics, or drug resistance predictions in infectious diseases, has opened a new era in precision medicine. Clinical decision-support-systems have been designed for assistance in molecular diagnostics (MDx) or companion diagnostics (CDx) to enhance therapeutic success. These systems are typically based on statistical or machine learning models, built upon clinical or corporate patient data. However, these software systems are not ready to use in MDx or CDx settings—due to regulatory requirements, such as specific ISO standards, which are needed for regulatory approval. As these standards are generally not fulfilled in academic software, there is currently only a limited use of such systems in clinical routine. Owing to the fact that academia cannot solve this problem on its own, we propose a scheme where the three key players, that is, academia, clinics, and industry, work more closely together in software development processes for medical devices.

Keywords: clinical studies; diagnostics; machine learning; regulation

Development of Prediction Models

The development of computational approaches for diagnosis, prognosis, and monitoring has opened a new era in precision medicine.¹ Systems medicine approaches consider all facets of data, for example, omics data, pathways, and gene regulations in an integrative manner.² Clinical decision-support-systems (cDSS) have been designed for assistance in molecular diagnostics (MDx) or companion diagnostics (CDx) to enhance therapeutic success. For example, cDSS are regularly applied to predict drug resistances that frequently occur under treatment.³ One of the most popular tools in this area is *geno2pheno*,⁴ a guide to choose an appropriate antiretroviral treatment for each individual patient, which has already entered clinical practice in Europe. Overall, there is great potential in the development of sophisticated cDSS based on statistical and machine learning techniques.

Machine learning techniques are also applied in other clinical fields, for example, in cancer.⁵ For instance, Listgarten et al.⁶ provide a model for predicting the susceptibility to breast cancer using single nucleotide polymorphisms. The model can be obtained by building classifiers by means of support vector machines, decision trees, or naive Bayes.⁶ Moreover, Ertle et al.⁷ used machine learning approaches to predict liver cancer and precursor states like fibrosis, using noninvasive computational approaches.

In general, the building of such classification models is roughly separated into two major tasks: (1) the training process that defines the construction of the model and (2) testing to assess the models' performance. Typically, classifiers are evaluated based on specific measures, such as accuracy or the area under the receiver operating characteristics curve.⁸ However, it is well known that all measures have specific pitfalls, for example, accuracy is

¹Department of Bioinformatics, Straubing Center of Science, Straubing, Germany.

²QIAGEN GmbH, Hilden, Germany.

³Research Group for Clinical Epidemiology, University Hospital Jena, Jena, Germany.

⁴Department of Mathematics and Computer Science, University of Marburg, Marburg, Germany.

*Address correspondence to: Dominik Heider, PhD, Department of Mathematics and Computer Science, University of Marburg, Hans-Meerwein-Str. 6, Marburg 35032, Germany, E-mail: dominik.heider@uni-marburg.de

biased toward the majority class in unbalanced data. For further information on classification evaluation, we refer to Lever et al.⁹ One important problem in this field is the acceptance of machine learning models by medical practitioners. Although standard statistical models such as logistic regression¹⁰ are widely accepted, modern approaches, for example, random forests or deep learning, are barely known and not accepted in MDx or CDx scenarios at the moment. This is mainly due to the fact that these approaches come with a kind of uncertainty, that the clinical interpretation of their results is sometimes more difficult, and that many well-known statistical performance measures, such as statistical power for building up appropriate clinical cohorts, are only poorly

understood in these models. Moreover, and maybe unnecessary to mention, the models should somehow match to the (quality of the) available data, for example, considering linear and nonlinear approaches as well as missing data.

However, the main pitfall of computational models for precision medicine developed in academia is that these academic researchers (e.g., PhD students and postdocs) typically have only temporary contracts and most software is developed by individuals on a one-person-one-project basis. Thus, these researchers develop software in a prototype-centered manner, meaning that they develop software for quickly publishable results. However, to enable future clinical

routine use of the software, at least two additional perspectives have to be considered: regulatory aspects of software development, which treats software as a medical device, and the necessity to evaluate the clinical patient-relevant benefits of the software in properly designed clinical trials to ultimately prove the efficacy and effectiveness of the MDx or CDx software.

Development of MDx Software

Creating a software that is meant to be used in MDx or CDx from the results of academic and clinical research means that various formal aspects of MDx software development have to be considered that may have been labeled as unimportant up to this point (see Fig. 1). The following paragraphs give a short overview over some of the aspects that are relevant for MDx or CDx software products; it is by no means an extensive guideline or reference.

What makes MDx or CDx software special when compared with other software is the severity of the consequences of malfunctions or inappropriate usage. Diagnostic decisions that are made or supported by software will often have significant impact on health or even life of the involved patients. Because of this, almost all countries have set up regulations that are supposed to ensure a high quality of the software. These regulations typically apply to the entire lifecycle of the software, from the initial planning to the final retirement.

The goal of all regulations is to ensure that the error rate in the diagnostic process is as low as possible. Therefore, typical questions governing the regulations are as follows:

- Is it clear what the software is supposed to do?
- How is it ensured that the software does what it was intended to do?
- How is misuse (intentional and unintentional) of the software prevented?
- How does the manufacturer ensure that all quality measures are obeyed?
- How does the manufacturer ensure that issues showing up during productive usage are handled appropriately?
- Have risks been considered?

To answer these (and other) questions in a structured and comparable manner, the regulations usually ask for certain processes to be implemented and documents to be created during the software development cycle. The minimal requirement will always be some sort of a quality management system implemented at the sites involved in the software development process.

This is usually based on ISO 9001 that sets up standards for a general quality management system, and ISO 13485, which defines quality management system standards for development of medical products. Somewhat more specific requirements are defined in IEC 62304, which specializes on the software lifecycle process for medical software. Moreover, risk management (ISO 14971) has to be taken into account. Risk management is extremely important for medical devices, as these devices are used for diagnostics, thus affect the way of treatment and may finally harm patients if predictions are incorrect. Hence, a rigorous risk assessment is necessary. Two other standards also apply to software as a medical device, although they are not limited to it: IEC 60601-1 adds requirements mainly about network, software interfaces, and hardware, and IEC 62366 adds requirements about ergonomics.

What often requires some rethinking for software development is that not only the quality of the final product is important and has to be explained to potential auditors, but also the organization and documentation of the process that is employed to create the final product. Defining the process for the creation of the software, documenting this definition, and making sure that the software development is actually performed according to the corresponding guidelines are an integral part of the software development for MDx or CDx software.

The standards mentioned before leave significant room for interpretation when they are used as base for MDx or CDx software development. Defining and documenting the actual process require a deliberate balance between effort, regulatory compliance, flexibility, accuracy, and other aspects. This is where skill, experience, and “art” (i.e., creative aspects of the software development processes) of the persons defining the process can easily make a significant difference in terms of person months and millions of dollars costs, for medium sized software projects.

A prominent example for regulations that apply to MDx or CDx software is the Food and Drug Administration (FDA) premarket approval. The FDA does not dictate which processes to follow and which documents to create; it does, however, issue recommendations on which documents to submit and what the processes should cover. Following these recommendations makes life a lot easier for both sides, the manufacturers and the FDA inspectors.

Ideally (from a process-oriented point of view), any MDx or CDx software should be planned, designed,

implemented, and tested in this order. This represents the classic “waterfall” approach of software development. There are good reasons for adopting this approach—it is important that all quality criteria are defined *before* the development starts, not *after* it has been found out what the software really does. Also the traceability, which links each requirement to the software items that implement it, the test that verifies these, and the validation of the requested (and vice versa), is easy to show when following a waterfall approach. For larger and more complex software projects, it has shown that a waterfall approach shows important drawbacks. Agile methods such as Scrum¹¹ or Kanban¹² have almost fully replaced strict waterfall processes for these. Defining an agile process that at the same time fulfils the requirements for full traceability and a documentation that meets regulatory requirements has shown to be a rather challenging task, but possible without sacrificing quality or regulatory compliance, as demonstrated in certification of MDx software from different companies, such as QIAGEN and Illumina.

When turning research results into an MDx or CDx product that includes software components, these software components very often already exist in some or the other form—these are mostly the software components that the research was done with, or derived of these. As research often develops quite dynamically, the software that is created for the specific research item is usually not planned and created in a way that would satisfy MDx or CDx regulatory needs. This occurs for a good reason, as the overhead that MDx or CDx software development requires would make research very inflexible and tedious. Although it is possible and often tempting to “convert” a research software item into an MDx or CDx software, it is often a better choice to start over again and create a new compliant software from scratch, taking the experience related to the research software as one important input for the new product.

In a software development environment that is known to be very focused on rules and regulations, it seems surprising how much “art” is involved when choosing the right path for building the application. Time to market and development costs are seriously affected, so this “art” often makes the difference between success and failure of a new product.

Evaluation Using Clinical Trials

So far we have only focused on the proper academic and industrial/regulatory aspects of MDx and CDx software development. Similar to the development pro-

grams of diagnostic tests (e.g., Ref.¹³), the already mentioned aspects usually cover that the technical properties are well understood and documented and result in reproducible outputs (phase I) while the software has also been applied in setting in which clear/gold-standard samples (e.g., from diseased/healthy individuals) have been evaluated (phase II). However, similar to the development of biomarkers or other medical devices with a stronger potential for a health benefit or harm, some of the future MDx or CDx software will require data from properly planned clinical phase III/IV trials—questions that are sometime summarized as health technology assessment (HTA). Although phase III studies have to be carried out in the same manner and setting as it would be in later clinical and diagnostic practice, phase IV trials aim at showing whether the application of the software leads to a measurable clinical outcome benefit in a broader population. Both late phases are used to demonstrate safety and efficacy/effectiveness of the software when compared with the standard of care. Related studies require proper study planning, conduct, and statistical analysis, which are largely prespecified in the protocol and the statistical analysis plan. It can be expected that the results of such HTA evaluations will become increasingly important when questions of reimbursement by healthcare providers are considered.

Conclusions

The challenge of producing MDx or CDx software hits most students unprepared when they enter industrial employment. Putting more emphasis on regulatory aspects, controlled development processes, and the necessity to demonstrate patient safety and efficacy in the education of students would provide these with the tools to master this “art.” Future students that are exposed to a wider multidisciplinary perspective will pave the way for a better and more direct transfer of academic software into MDx and CDx software, and thus, to a better treatment of patients. Thus, we propose the following solutions for the challenges in academic settings: (1) department should hire permanent technical academic staff as software quality managers, who support the development of software for at least PhD students and postdocs, (2) academic software development for MDx and CDx applications should be carried out in close collaboration with clinical and industrial partners (e.g., as mentors for a subsequent spin-off), and (3) formal software development courses should be part of students curricula.

Authors' Contributions

D.H. designed the concept and supervised the work. M.R., A.S., J.W., and D.H. wrote the article. All authors approved the final article.

Author Disclosure Statement

No competing financial interests exist.

References

1. Hawgood S, Hook-Barnard IG, O'Brien TC, et al. Precision medicine: beyond the inflection point. *Sci Transl Med*. 2015;7:300ps17.
2. Batra R, Alcaraz N, Gitzhofer K, et al. On the performance of *de novo* pathway enrichment. *NPJ Syst Biol Appl*. 2017;3:6.
3. Riemenschneider M, Hummel T, Heider D. SHIVA—a web application for drug resistance and tropism testing in HIV. *BMC Bioinformatics*. 2016;17:314.
4. Lengauer T, Sander O, Sierra S, et al. Bioinformatics prediction of HIV coreceptor usage. *Nat Biotechnol*. 2007;25:1407–1410.
5. Beck AH, Sangoi AR, Leung S, et al. Systematic analysis of breast cancer morphology uncovers stromal features associated with survival. *Sci Transl Med*. 2011;3:108ra113.
6. Listgarten J, Damaraju S, Poulin B, et al. Predictive models for breast cancer susceptibility from multiple single nucleotide polymorphisms. *Clin Cancer Res*. 2004;10:2725–2737.
7. Ertle JM, Heider D, Wichert M, et al. A combination of α -fetoprotein and des- γ -carboxy prothrombin is superior in detection of hepatocellular carcinoma. *Digestion*. 2013;87:121–131.
8. Fawcett T. An introduction to ROC analysis. *Pattern Recognit Lett*. 2006; 27:861–874.
9. Lever J, Krzywinski M, Altman N. Points of significance: classification evaluation. *Nat Methods*. 2016;13:603–604.
10. Lever J, Krzywinski M, Altman N. Points of significance: logistic regression. *Nat Methods*. 2016;13:541–542.
11. Beedle M, Schwaber K. *Agile Software Development with Scrum*. Englewood Cliffs, NJ: Prentice Hall. 2002.
12. Anderson DJ. *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press. 2010.
13. European Medicines Agency. *Pre-Authorisation Evaluation of Medicines for Human Use*. Doc. Ref. CPMP/EWP/1119/98/Rev.1. London: Penguin. 2009.

Cite this article as: Riemenschneider M, Wienbeck J, Scherag A, Heider D (2018) Data science for molecular diagnostics applications: from academia to clinic to industry, *Systems Medicine* 1:1, 13–17, DOI: 10.1089/sysm.2018.0002.

Abbreviations Used

cDSS = clinical decision-support-systems
CDx = companion diagnostics
EMA = European Medicines Evaluation Agency
FDA = Food and Drug Administration
HTA = health technology assessment
MDx = molecular diagnostics

Publish in Systems Medicine

- Immediate, unrestricted online access
- Rigorous peer review
- Compliance with open access mandates
- Authors retain copyright
- Highly indexed
- Targeted email marketing

liebertpub.com/sysm