Scaffolding the Generation of Machine Learning Models with SciRise

Aneesha Bakharia
The University of Queensland, Brisbane, Australia
aneesha.bakharia@gmail.com

Abstract

While many robust and feature-rich Machine Learning libraries exist, novice users find it difficult to understand a standard data science workflow and produce generalisable models. In this position paper, I argue that a carefully designed user interface can alleviate most of the issues encountered by novice users and introduce the design principles that underpin the development of SciRise. SciRise is being actively developed as an open source Jupyter Notebook widget that generates code for Scikit-Learn and Auto-SKLEARN. Challenges and future directions for SciRise are briefly also discussed.

1 Introduction

Scikit-Learn [Pedregosa *et al.*, 2011] is an algorithm rich Machine Learning (ML) library for the Python programming language. Scikit-Learn has a well-documented API, is robust and has fast implementations of many supervised and unsupervised algorithms. Scikit-Learn also offers many features to assist with data pre-processing (i.e., scaling data and normalisation), ML experiment setup (i.e., splitting of data into train, test and validation sets) and model selection (i.e., GridCV). Novice users however find is difficult to directly program Scikit-Learn and generate ML models the are able to generalise to unseen data. Novice users while keen to learn data science skills may not have the required programming knowledge and dont understand fundamental ML experiment setup concepts. These key concepts include the importance of model selection (i.e., automatic algorithm parameter selection) and model validation. New domain specific languages and ML libraries are not required to address the issues that novice users encounter. I argue that instead of making new languages and libraries, that careful and user-centered interface design is able to address the issues that novice users encounter.

A key aim for SciRise is to scaffold the ML experiment setup process for novice users, teach ML concepts and support the user as they transition to writing their own code using the Scikit-Learn API. The SciRise user interface takes a declarative template-based approach. Various ML visual tools do exist, such as Orange but use a flow chart user interface metaphor. A flow chart user interface has not been used by SciRise, because it would be difficult to expose the user to the Scikit-Learn API and ultimately teach the user to write their own code. Within the SciRise user interface, the user is able to clearly see each step in the experiment, algorithms and parameters are labeled using the Scikit-Learn API, parameters and options can be edited and a preview of generated code is available.

2 Design principles

In this section the design principles that have driven the design and implementation of the SciRise tool are discussed. I believe that it is important to share these design principles to make the reasons behind the opinionated design decisions transparent. The design principles may also be of interest to other Machine Learning and Deep Learning visual tool designers.

2.1 Encapsulate ML experiment design

While an experienced Data Science professional is able to design ML experiments, novice users find the task challenging. Classic ML experiment design, involves splitting a dataset into train, test and validation sets; selecting the appropriate algorithms and specifying good initial parameters for the algorithms. It is very important that novice users understand the importance of model validation, as the consequences will be a model that is unable to generalise to unseen data and prone to overfitting. The SciRise tool for this reason uses a declarative template-based approach to scaffold the creation of ML experiments.

Experiments using both the classic test, train and validation set split as well as k-fold cross-validation (CV) are supported and map directly to the functionality provided by Scikit-Learn. In future releases the full set of available SciKit-Learn cross-validation features will be included (i.e., Leave One Out (LOO), Leave P Out (LPO), Random permutations cross-validation (i.e., Shuffle and Split), Stratified k-fold and Stratified Shuffle Split). ML experiment templates are available as starting points for supervised learning. Users are easily able run an ML experiment without being required to customise or configure the process but can also start to adjust the settings as they become more experienced and knowledgeable.

2.2 Include context-sensitive help

Projects like Scikit-Learn and many other ML libraries include useful and detailed API documentation but don't include explanations of why a particular feature is important. The "What?" and "Why?" is very important to novice users of ML algorithms e.g., "What is model selection and why is it important?". SciRise therefore includes context-sensitive help that answers the "What?" and "Why?" question for each setting and option that a user is able to choose. Links are also provided to the Scikit-Learn API and other useful introductory resources.

2.3 Recommend ML algorithms and parameters

Model selection in terms of choosing an algorithm with high accuracy for the task and then specifying good values for model parameters, is a complex task especially for novice users. Scikit-Learn includes very useful Grid-CV functionality but many novice users are don't know that the functionality exists. Scikit-Learn's Grid-CV is able to perform cross-validation across algorithms and their parameters. Auto-SKLEARN [Feurer *et al.*, 2015] provides a very simple API to perform regression and classification, ensuring that the optimal algorithm and associated model parameters are used, but the chosen algorithm and its parameters are not transparent. SciRise in providing a user interface for both Scikit-Learn and Auto-SKLEARN, aims to make the use of model selection both transparent and understandable for a novice user. Users will gain an understanding of the available parameters that must be tuned, just by interacting with the SciRise user interface.

2.4 Encourage experimentation

As mentioned in the introduction, SciRise provides a user interface to take advantage of Scikit-Learn and Auto-SKLEARN. The user interface has not been designed to only provide an abstraction layer over the exposed API's from both SciKit-Learn and Auto-SKLEARN. The key aim has been to make the API's from both tools accessible to novice users while still ensuring data science best practice and encouraging user experimentation. In SciRise, the user is able to actively explore alternate configuration settings and API options. It is hoped that experimentation will lead to an understanding of all supported features and algorithms in Scikit-Learn and eventually allow the user to use to modify the code generated by SciRise for more complex ML projects.

2.5 Integrate with Existing Data Science Tools

SciRise has been implemented as a Jupyter Notebook widget rather than a standalone application. This decision was made both in a conscious and deliberate manner to promote the use of a data science workflow. The Jupyter Notebooks offer a number of advantages particularly for novice users. In the case of SciRise, a Jupyter Notebook widget provides the ideal environment to embed the SciRise user interface and allow users to build ML models. As users become more familiar with SciRise, they may wish to generate and directly edit the Scikit-Learn code. Implementation as a Jupyter Notebook widget means that the generated code can be inserted into a new cell and directly edited. A key advantage is that Jupyter Notebooks which use the SciRise widget, when shared with other users will allow model experimentation.

2.6 Generate High Quality Python Code

SciRise includes functionality to generate Scikit-Learn and Auto-SKLEARN code. Code is generated internally to execute the ML models and also available for export. Users are able to choose between supervised and unsupervised learning, customise options provided within the Jupyter Notebook, view model results and generate Python programming code. The generated code is structured and includes detailed comments. The generated code provides a useful starting point for users wishing to investigate more complex ML experiments.

2.7 Python programming experience not required

While SciRise is able to generate code, no programming experience is required. SciRise has been designed to expose the full functionality of both Scikit-Learn and Auto-SKLEARN via a graphical user interface. The full range of Scikit-Learn algorithms are available for experimentation by novice users including data pre-processing, clustering, classification, regression, model selection and dimension reduction.

3 Challenges

A key challenge in implementing SciRise, is user interface responsiveness (i.e., the time taken to return results). On one hand SciRise is designed to encourage experimentation and exploration but on the other hand the time taken for Grid-CV and Auto-SKLEARN to complete model and parameter selection will vary depending upon the size of the dataset. The current approach taken by SciRise is to make the user aware that the process will not return results immediately. Eventually, a technique to determine an estimate of execution time will be implemented.

4 Future Directions

Future directions include conducting a comprehensive usability study and creating templates to scaffold the creation of advanced Deep Learning network architectures that use computational graph ML libraries (e.g., TensorFlow and Keras).

References

[Feurer *et al.*, 2015] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2962–2970. Curran Associates, Inc., 2015.

[Pedregosa *et al.*, 2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.