dummyML: Automated tabular data analysis pipelines for non-experts

Yipeng Song, Yang S Liu, Dan Metes, Mengzhe Wang, Bo Cao

Submitted to: JMIR Preprints
on: August 30, 2024

Table of Contents

dummyML: Automated tabular data analysis pipelines for non-experts

Yipeng Song[1*] PhD; Yang S Liu[1*] PhD; Dan Metes[2] MSc; Mengzhe Wang[2] MSc; Bo Cao[1] PhD

[1]Department of Psychiatry Faculty of Medicine and Dentistry University of Alberta Edmonton CA
[2]Ministry of Health Government of Alberta Edmonton CA
[*]these authors contributed equally

Corresponding Author:
Bo Cao PhD
Department of Psychiatry
Faculty of Medicine and Dentistry
University of Alberta
4-142A Katz Group Centre for Research
11315 - 87 Ave NW
Edmonton
CA

Abstract

Health researchers and healthcare professionals are interested in exploring predictive analytics and machine learning applications of health data but were challenged by accessing software programming expertise and dealing with the complexity of carrying out machine learning based predictive analysis (e.g., prediction of a future outcome at the individual level). We present an automated machine learning analytical pipeline dummyML, a free and open-source tool designed to reduce the burden of conducting exploratory machine-learning data analysis on tabular data (e.g., data organized into a table, in which information is arranged in rows and columns), for non-experts, and to facilitate meaningful variable explanation.

(JMIR Preprints 30/08/2024:65966)
DOI: https://doi.org/10.2196/preprints.65966

Preprint Settings

1) Would you like to publish your submitted manuscript as preprint?

✓ **Please make my preprint PDF available to anyone at any time (recommended).**
Please make my preprint PDF available only to logged-in users; I understand that my title and abstract will remain visible to all users.
Only make the preprint title and abstract visible.
No, I do not wish to publish my submitted manuscript as a preprint.

2) If accepted for publication in a JMIR journal, would you like the PDF to be visible to the public?

✓ **Yes, please make my accepted manuscript PDF available to anyone at any time (Recommended).**
Yes, but please make my accepted manuscript PDF available only to logged-in users; I understand that the title and abstract will remain v
Yes, but only make the title and abstract visible (see Important note, above). I understand that if I later pay to participate in <a href="http

Original Manuscript

dummyML: Automated tabular data analysis pipelines for non-experts

Yipeng Song[a†], Yang S. Liu[a†], Dan Metes[b], Mengzhe Wang[b], Bo Cao[a*]
[a]Department of Psychiatry, University of Alberta, Edmonton, Canada
[b]Ministry of Health, Government of Alberta, Edmonton, Canada

[†] These authors contributed equally to this work as joint first authors

[*]Correspondence to: Dr. Bo Cao, Department of Psychiatry, University of Alberta, Edmonton, Alberta, Canada (email: cloudbocao@gmail.com), (780) 407-6504.

Keywords: Machine Learning, dummyML, AutoML, Health Data, Predictive Analytics

Abstract

Health researchers and healthcare professionals are interested in exploring predictive analytics and machine learning applications of health data but were challenged by accessing software programming expertise and dealing with the complexity of carrying out machine learning based predictive analysis (e.g., prediction of a future outcome at the individual level). We present an automated machine learning analytical pipeline dummyML, a free and open-source tool designed to reduce the burden of conducting exploratory machine-learning data analysis on tabular data (e.g., data organized into a table, in which information is arranged in rows and columns), for non-experts, and to facilitate meaningful variable explanation.

Introduction

Healthcare professionals have seen an exponential increase in biomedical data sets, ranging from electronic health records (EHR) collected in hospitals[1], omics data collected in the laboratory setting[2], administrative data routinely collected by healthcare authorities[3], and many others[4], [5]. Predictive modeling based on machine learning (ML) algorithms is gaining popularity and has shown promising results in analyzing these complex biomedical data sets to support disease diagnosis, biomarker discovery, treatment optimization, and healthcare management[3], [4], [6], [7]. For example, a recent study[3] achieved good accuracy in predicting the onset of diabetes using routinely collected administrative health data on 1,657,395 patients by applying ML-based predictive modeling. However, adopting best practices when building ML models and deploying the models in production requires a high level of expertise in both machine learning and computer programming. These technical requirements pose a barrier to the widespread adoption of machine learning among health data users. Thus, automated data analysis tools based on machine learning are in high demand for data users with domain knowledge in healthcare but limited expertise in ML and programming.

Various automatic machine learning (AutoML) solutions have been proposed to automate the process of raw data preprocessing, predicting variable or feature engineering, model selection, and model deployment[8]. The core idea of AutoML solutions is to select an ML algorithm and the corresponding hyperparameters of this algorithm to achieve optimal performance on a given task. One of the state-of-the-art AutoML systems is Auto-Sklearn[9], which was built on top of Sklearn[10], one of the most widely used machine learning packages for experts and novices. Auto-Sklearn applied Bayesian Optimization[11] to select the hyperparameters in controlling the data preprocessing, feature preprocessing steps, and base machine learning models. For example, for a classification problem, Auto-Sklearn uses 15 base classifiers, 14 feature preprocessing methods, and four data preprocessing methods, resulting in 110 hyperparameters. And these 110 hyperparameters are tuned by Bayesian Optimization. The final output is an ensemble of the 15 base classifiers, and the weights will be determined by the classifier’s performance on hold-out validation data or cross-validation. Another widely used AutoML system is Auto-Weka[12], which is implemented on Java and as a standalone application. Some other popular AutoML systems include TPOP[13] based on genetic algorithms for hyperparameter tuning and TuPAQ[14] based on query optimization.

Despite the development of many sophisticated AutoML systems, it’s underutilized in healthcare research[8]. A key underlying challenge is the lack of transparency in black-box AutoML systems. For example, to solve a classification problem, Auto-Sklearn used Bayesian Optimization to optimize 110 hyperparameters, and the resulting model is an ensemble of 15 base machine learning models. A combination of ML models as an ensemble may complicate the interpretation. Also, combining 14 feature preprocessing methods and four data preprocessing methods produced uninterpretable features that can be very difficult to map back to the original variables. The lack of transparency of these sophisticated AutoML systems impeded their usage among healthcare professionals who are not confident about the automated generated results and worried about the interpretability of the derived highly complex ensemble models.

Another common barrier to applying ML in healthcare research is the skewed distribution of different classes within datasets (e.g., disease versus non-disease). This class imbalance makes it challenging for ML models to learn effectively from the data, potentially introducing biases and impeding model performance and interpretation. Accessing and using algorithms that address the class imbalance in data adds a layer of complexity to users without a deep knowledge of ML and programming.

To make ML analysis accessible, we developed dummyML to address the challenge of model transparency and interpretability, as well as to reduce the technical skills barrier for users, by applying commonly used steps in practical predictive data analysis and providing a web-based graphical user interface (GUI) for the automated data analysis, including user selection of analysis pipelines, model training and saving trained model for future testing and deployment. Users can finish all the data analysis steps by click-and-run without a deep knowledge of programming.

Common data analysis steps in practical predictive data analysis

The components of dummyML were designed following common steps in practical predictive data analysis. 1) Tidy the raw data into tabular form with each row representing a subject and each column representing a variable[15]. 2) Explore the data to have a better understanding of them. 3) Preprocess the data, e.g. remove free-text data, missing value imputation, dummy coding the categorical variables, etc., to prepare it for ML models. 4) Compare multiple types of models, e.g. regularized linear models, random forest, support vector machine and gradient boosting, and select the hyperparameters (i.e., configuration variables) of these different ML models. 5) Validate the selected models' performance for making predictions. 6) Select a single model based on the trade-off between predictive performance and interpretability. 7) Deploy the selected model to make predictions on future data sets.

User interfaces of automated data analysis pipelines

We implemented the automated data analysis pipeline by closely mimicking the above commonly used data analysis steps. In this way, automated data analysis is as transparent as practical predictive data analysis. The proposed automated data analysis pipelines, named "dummyML", have two user interfaces, programming-based and GUI based. A Python package named "dummyML" was implemented for the proposed automated data analysis pipeline. Users with Python programming experience can use this package to achieve maximum flexibility in doing automated data analysis with minimum effort in writing codes. A separate web application named "dummyML_web" was developed to provide a web-based GUI for users without programming experience in Python. Users can still finish all the automated data analysis steps and deploy the trained models for predicting future data by click-and-run without deep programming knowledge, after proper setup by themselves or with the help of an IT technician on a local PC or online service.

Design of dummyML

The dummyML is more like an automated data analysis pipeline rather than an AutoML system. Our implementation was set apart from other AutoML systems by the following principles.

Mimicking practical predictive data analysis steps. The implemented dummyML closely mimicked the commonly used data steps in practical predictive data analysis. It is not as sophisticated as other well-established AutoML systems and has some limitations, i.e., feature engineering is not encouraged even though the option is provided in dummyML, only commonly used data preprocessing steps are available, and can only choose from widely used standard models. However, the advantage of following practical prediction data analysis steps is that it is easy to interpret the data using a framework that health researchers are already familiar with.

Designed for non-experts. The implemented dummyML consists of two user interfaces. Non-experts with experience in Python programming can use the programming interface to achieve maximum flexibility in automated data analysis with a few lines of code. Non-experts without experience in Python programming can use the web-based GUI to finish all the steps in automated data analysis by click-and-run without any programming. Furthermore, the users can easily deploy the trained models in a web application to make predictions for other users.

Implementation details

Data exploration. The automated data analysis pipeline requires the data to be in tabular form; each row indicates a subject and each column indicates a separate variable. We applied the Python package ydata-profiling[16] to generate basic summary statistics, e.g., missing patterns, quantile statistics, correlation statistics, and histogram plots, to facilitate data exploration.

The GUI for the data exploration part is depicted in Figure 1. Click the checkbox “A sample of the raw data” will show a random sample of the raw data, and click the checkbox “Data summary” will show an HTML page containing the basic summary statistics to explore the raw data.

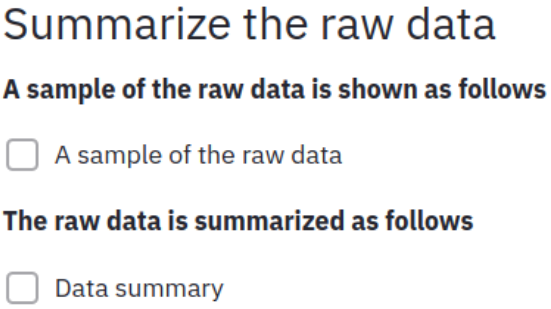

Figure 1: The GUI for the data exploration section in the proposed automated data analysis pipeline.

Data preprocessing. We implemented a minimal but viable data preprocessing pipeline to prepare the data for the ML model's compatible features. 1) The variables with limited unique integers (sometimes, a categorical variable is coded as integers rather than the original string format) can be transformed into categorical variables. 2) The categorical variables are one-hot coded with missing values as a separate level. 3) Samples and variables with large amounts of missing values are excluded from the data analysis. 4) Median imputation is used to impute the missing values in quantitative variables. 5) All the above steps are saved for preprocessing future data.

Figure 2 depicts the GUI for the data preprocessing section. The user can either use the default values for the arguments to control the data preprocessing behavior or set up the values according to their understanding of the data. The GUI also provides the meaning of the arguments, tips and tricks for setting these arguments, key steps in the data preprocessing section, and a link to the source code.

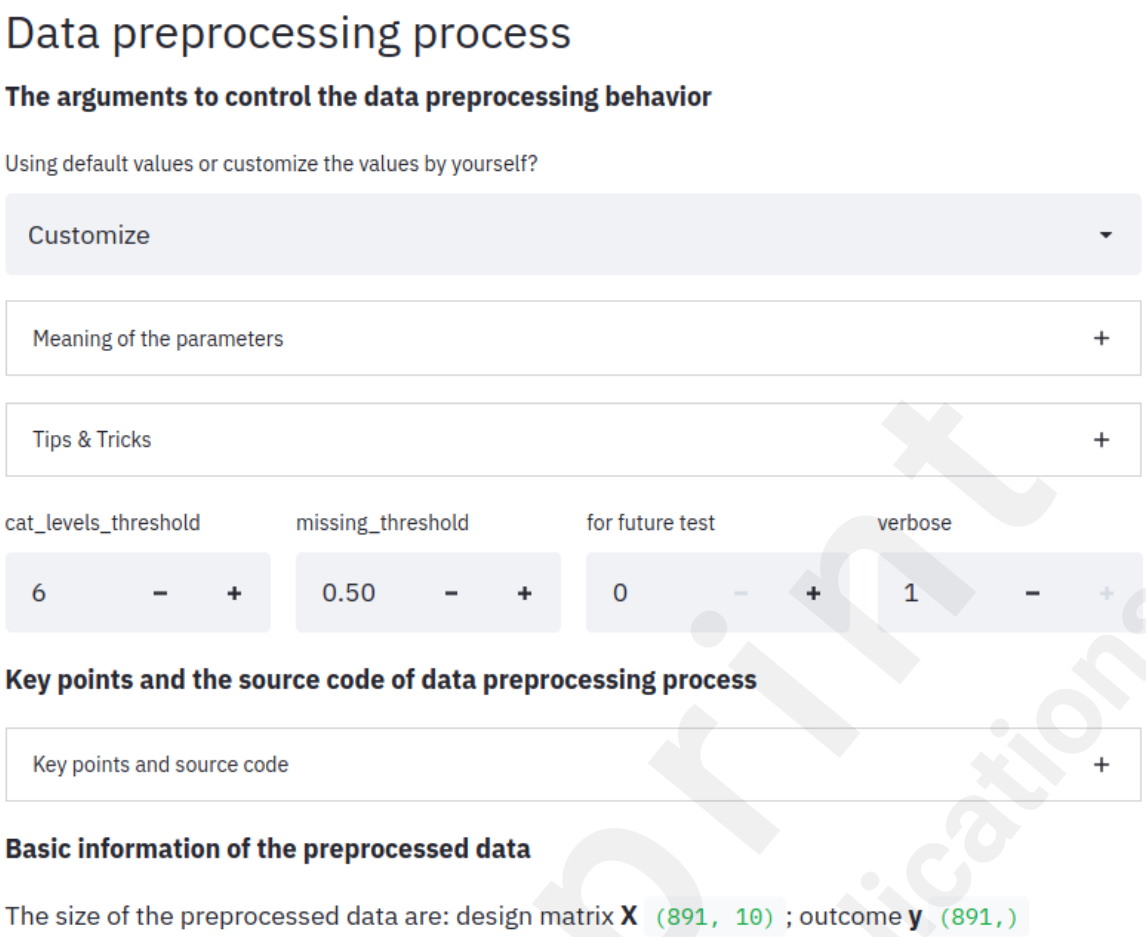

Figure 2: The GUI for the data preprocessing section of the proposed automated data analysis pipeline.

Select and fit multiple types of ML models. For the majority of ML models, there are associated hyperparameters to control the model's complexity. The hyperparameters of an ML model must be properly selected to achieve good prediction performance on unseen data, future data, or artificially left test data. A complex model may overfit the training data, where the resulting model fits the data well but performs poorly on the unseen data, and an oversimplified model may underfit the data and not make meaningful predictions. A simple example is the Lasso model (linear regression model with Lasso penalty), in which the strength of the Lasso-type penalty controls the model's complexity. When the Lasso penalty is very large, all the coefficients shrink to zero, and underfitting happens; when the Lasso penalty is very small, all the coefficients are nonzero. Overfitting may happen when data is of high dimensional,

In practical predictive data analysis, we need to select both the type of ML model and the hyperparameters associated with the model. And, unlike the previously mentioned AutoML systems, e.g., the ensemble of 15 classifiers in the Auto-Sklearn, we prefer to use standard ML models, which usually give satisfactory results when properly tuned. Standard ML models were also widely tested in research and production, and interpretation methods were established. We included the following standard models (Table 1) in the automated data analysis pipelines for both classification and regression. Three additional models specially designed for imbalanced classification, where the number of positive cases in the outcome overwhelmed the negative cases, were also included as standard models. After the preprocessed data is split into training and test data sets, the specified standard models are tuned to select the corresponding hyperparameters. The list of standard models and the corresponding hyperparameter tuning method are shown in Table 1.

Model	Hyperparameter	Hyperparameter tuning method	Feature importance measure
Linear model	None	None	Coefficient
Linear model with Lasso type penalty	Strength of Lasso type penalty	Grid search	Coefficient
Linear model with ridge type penalty	Strength of ridge type penalty	Grid search	Coefficient
Linear model with ElasticNet type penalty	Strength of Lasso type penalty; the ratio of L1/L2 penalty	Grid search	Coefficient
Support vector machine	Regularization strength; Kernel	Bayesian Optimization	Permutation importance
Neural network (multi-layer perceptron)	Number of hidden layers, hidden layer sizes, learning rate, L2 regularization	Bayesian Optimization	Permutation importance
Gradient boosting	Maximum depth of base tree; number of trees	Bayesian Optimization	Impurity-based feature importance
Random forest	Maximum number of features for splitting; number of trees	Bayesian Optimization	Impurity-based feature importance
Random under-sampling integrated in the learning of AdaBoost	Maximum depth of base tree; number of trees	Bayesian Optimization	Impurity-based feature importance
Balanced random forest model	Maximum number of features for splitting; number of trees	Bayesian Optimization	Impurity-based feature importance

Table 1: Standard models included in the dummyML package and the corresponding hyperparameters and hyperparameter tuning method.

The GUI for the select and fit multiple types of ML models is depicted in Figure 3. The user can specify the standard models for comparison. Also, the user can either use the default values for the arguments to control the data preprocessing behaviour or set up the values according to their own understanding of the data. The meaning of the arguments, the tips & tricks for setting these arguments, the key steps in the data preprocessing section, and the link to the source code are also

available in the GUI.

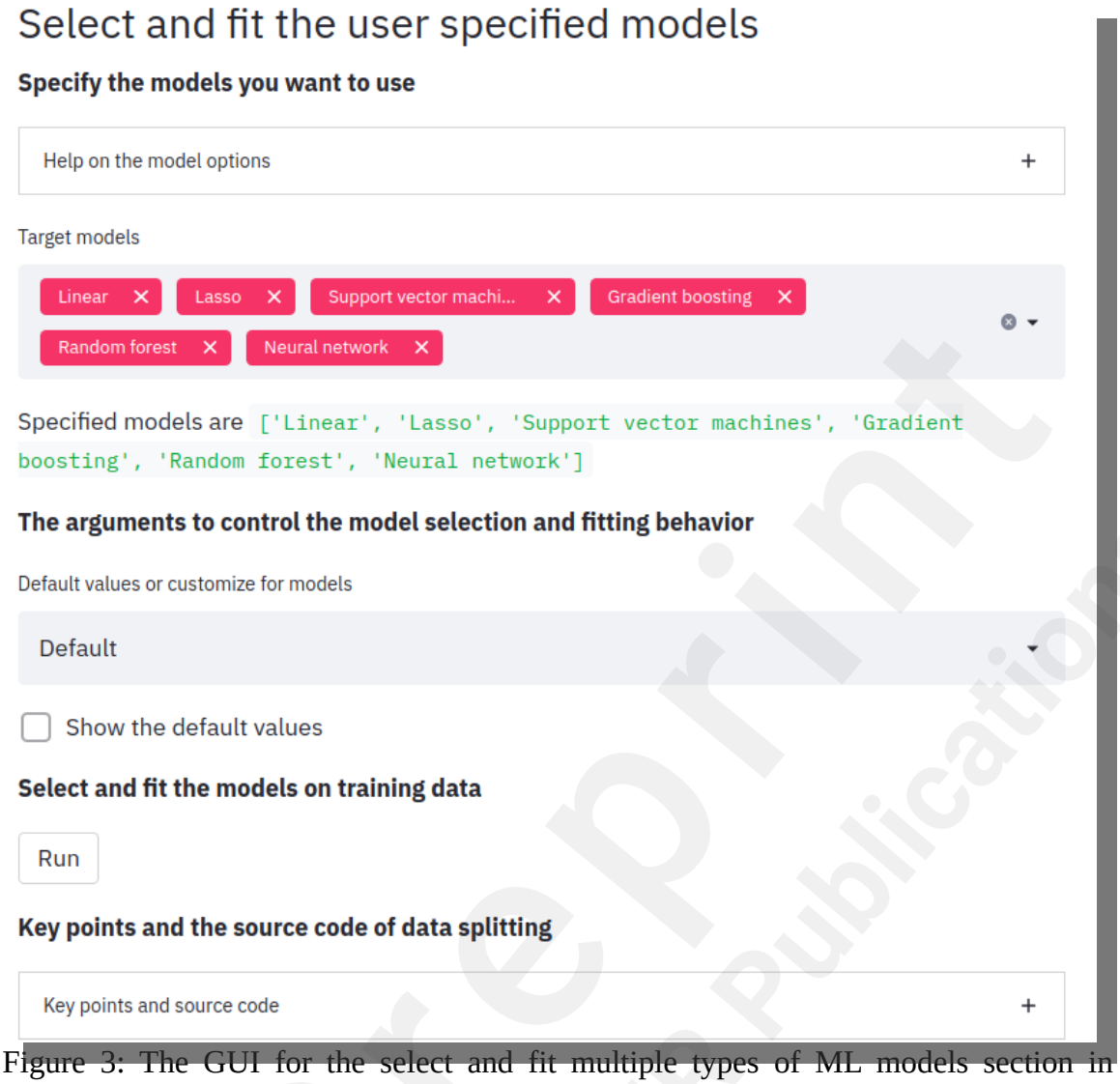

Figure 3: The GUI for the select and fit multiple types of ML models section in the proposed automated data analysis pipeline.

Validate the selected models. After the models are selected and fitted, they are further validated on the test set or through K-Fold cross-validation (CV) (default K= 10). The models are validated with respect to most commonly used metrics, i.e., sensitivity, specificity, balanced_accuracy, recall, precision, f1_score and AUC for classification models and R squared, mean squared error (MSE) and mean absolute error (MAE) for regression models. The scoring algorithms are native to Sklearn [10], see sklearn.metrics API documentation for details.

Figure 4 depicts the GUI for validating the selected models section. The user can either use the default values for the arguments to control the validation process or set up the values according to their own understanding of the data. The GUI also provides the meaning of the arguments, tips and tricks for setting these arguments, key steps in the data preprocessing section, and a link to the source code.

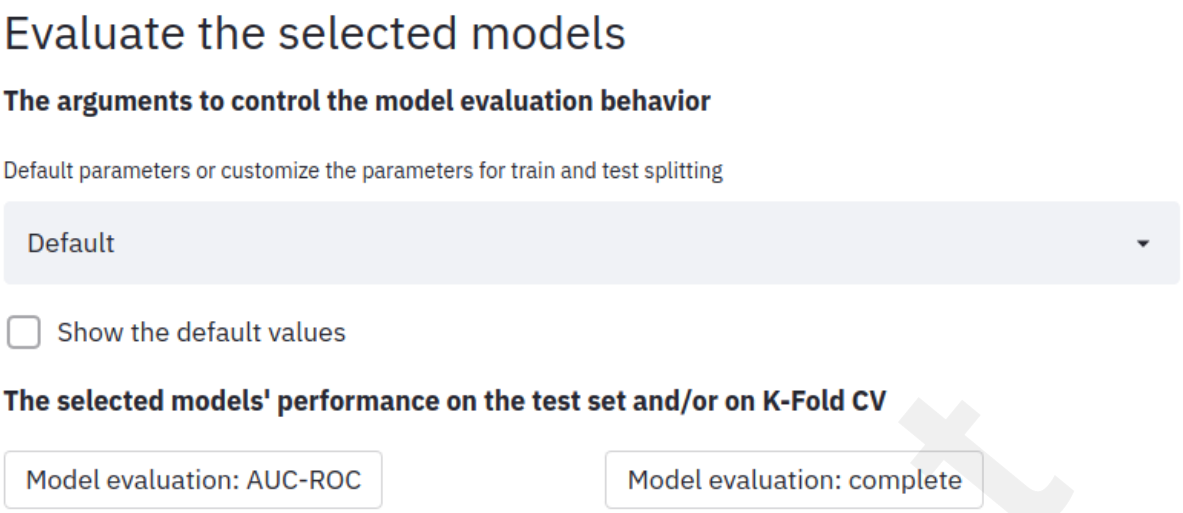

Figure 4: The GUI for the validation of the selected models section in the automated data analysis pipeline.

Save the selected models and explore the results After selected models are evaluated, they can be saved for other uses. We can also try to interpret the selected models by examining the variables' contributions to the prediction. The measures used to evaluate the contribution of each variable in the prediction model are in Table 1.

Real-world dummyML example

We further demonstrate the functionality of the dummyML pipelines on automated data analysis using an Electronic Healthcare Records (EHR) data set. MIMIC-III[1] is one of the largest open-source EHR data sets from emergency departments in a large tertiary care hospital. MIMIC-III contains comprehensive measurements on diagnosis, medical procedures, medication, lab tests, vital signs, clinical notes, survival data, length of stay, etc. Harutyunyan and colleagues [17] proposed building several benchmark data sets based on MIMIC-III to compare the existing models and newly developed models' performance on several commonly used tasks in practice. We used the automated data analysis pipelines on the benchmark data set to predict in-hospital mortality. The results are shown in Table 2.

	clf_linear	clf_lasso	clf_ridge	clf_gb	clf_rf
sensitivity	0.719251337	0.556149733	0.703208556	0.302139037	0.248663102
specificity	0.795248078	0.843815514	0.79944095	0.983577918	0.985674354
balanced_accuracy	0.757249708	0.699982623	0.751324753	0.642858477	0.617168728
recall	0.719251337	0.556149733	0.703208556	0.302139037	0.248663102
precision	0.314619883	0.317557252	0.314217443	0.70625	0.694029851

f1_score	0.437754272	0.404275996	0.434351775	0.423220974	0.366141732
AUC	0.836937634	0.783886314	0.840339204	0.860499183	0.850289801

Table 2: The results of the test set of the benchmark MIMIC-III data sets using the proposed automated data analysis pipeline.

Discussion

dummyML is a free and open-source automated data analysis pipeline for building predictive modeling on tabular data. It is intuitive and easy for healthcare professionals with limited expertise in ML and programming to use. The adoption of dummyML will enable health professionals to make full use of their ever-growing healthcare data sets by applying advanced ML techniques.

dummyML is designed for non-experts with or without Python programming experience. Non-experts with limited Python programming experience can make full use of the developed dummyML Python package to achieve maximum flexibility in the automated analysis of their data sets. Non-experts without Python programming experience can also automate the analysis of their data sets by click-and-run without any deep knowledge of programming. In addition, the provided web application makes it easy for non-experts to deploy their trained models in production. No programming effort or knowledge about web development is needed for model deployment. dummyML has been used as a core analytical tool by the authors for academic research (e.g., [18]) and serves as a starting point for efficient data exploration. The tool is open-source and thus can be modified by the research community to fit specific future analytical needs.

Although the dummyML is not as sophisticated as other well-established AutoML systems, by mimicking the steps of practical predictive data analysis, the model selection process and the selected models in dummyML enjoy the same level of transparency and interpretability as practical predictive data analysis. Transparency and interpretability are two of the most important factors in healthcare data analysis. In addition, as shown in the real data analysis of predicting the in-hospital morbidity in one of the largest open-source EHR data, MIMIC-III, the well-tuned standard models work well.

Acknowledgments

This research was undertaken, in part, thanks to funding from the Canada Research Chairs program, Alberta Innovates, Mental Health Foundation, MITACS Accelerate program, Simon & Martina Sochatsky Fund for Mental Health, Howard Berger Memorial Schizophrenia Research Fund, the Abraham & Freda Berger Memorial Endowment Fund, the Alberta Synergies in Alzheimer's and Related Disorders (SynAD) program, University Hospital Foundation and University of Alberta, and NARSAD Young Investigator Grants - Brain and Behavior Research Foundation.

Author contributions

YS - Conceptualization, Writing - Original draft, software; YSL - Writing -Original draft, Writing – Editing; DM - Writing - Editing; MW - Writing - Editing; Bo Cao - Supervision, Writing - Editing.

Software availability

Instructions and software packages are openly available at the following URL.

dummyML package: https://gitlab.com/YipengUva/end2endml_pkg
dummyML webUI: https://gitlab.com/YipengUva/end2endml_web

References

[1] A. E. W. Johnson *et al.*, "MIMIC-III, a freely accessible critical care database," *Sci. Data*, vol. 3, 2016, doi: 10.1038/sdata.2016.35.

[2] A. D. Hingorani, T. Shah, M. Kumari, R. Sofat, and L. Smeeth, "Translating genomics into improved healthcare," *BMJ*, vol. 341, no. nov05 1, 2010, doi: 10.1136/bmj.c5945.

[3] M. Ravaut *et al.*, "Development and Validation of a Machine Learning Model Using Administrative Health Data to Predict Onset of Type 2 Diabetes," *JAMA Netw. Open*, 2021, doi: 10.1001/jamanetworkopen.2021.11315.

[4] B. Cao *et al.*, "Treatment response prediction and individualized identification of first-episode drug-naïve schizophrenia using brain functional connectivity," *Mol. Psychiatry*, vol. 25, no. 4, 2020, doi: 10.1038/s41380-018-0106-5.

[5] D. Bzdok and A. Meyer-Lindenberg, "Machine Learning for Precision Psychiatry: Opportunities and Challenges," *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, vol. 3, no. 3. 2018, doi: 10.1016/j.bpsc.2017.11.007.

[6] D. B. Dwyer, P. Falkai, and N. Koutsouleris, "Machine Learning Approaches for Clinical Psychology and Psychiatry," *Annual Review of Clinical Psychology*. 2018, doi: 10.1146/annurev-clinpsy-032816-045037.

[7] A. L. Neves *et al.*, "Using electronic health records to develop and validate a machine-learning tool to predict type 2 diabetes outcomes: A study protocol," *BMJ Open*, vol. 11, no. 7, 2021, doi: 10.1136/bmjopen-2020-046716.

[8] J. Waring, C. Lindvall, and R. Umeton, "Automated machine learning: Review of the state-of-the-art and opportunities for healthcare," *Artificial Intelligence in Medicine*, vol. 104. 2020, doi: 10.1016/j.artmed.2020.101822.

[9] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Auto-Sklearn: Efficient and robust automated machine learning," in *Automated machine learning: Methods, systems, challenges*, 2019.

[10] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, 2011.

[11] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems*, 2012, vol. 4, pp. 2951–2959, Accessed: Oct. 06, 2021. [Online]. Available: http://papers.nips.cc/paper/4522-practical-bayesian -optimization-of-machine-learning-algorithms.pdf.

[12] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA," *J. Mach. Learn. Res.*, vol. 18, pp. 1–5, Mar. 2017, doi: 10.1007/978-3-030-05318-5_4/TABLES/2.

[13] R. S. Olson, R. J. Urbanowicz, P. C. Andrews, N. A. Lavender, L. C. Kidd, and J. H. Moore, "Automating biomedical data science through tree-based pipeline optimization," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9597, doi: 10.1007/978-3-319-31204-0_9.

[14] E. R. Sparks, A. Talwalkar, D. Haas, M. J. Franklin, M. I. Jordan, and T. Kraska, "Automating model search for large scale machine learning," 2015, doi: 10.1145/2806777.2806945.

[15] H. Wickham, "Tidy data," *J. Stat. Softw.*, vol. 59, no. 10, 2014, doi: 10.18637/jss.v059.i10.

[16] F. Clemente, G. M. Ribeiro, A. Quemy, M. S. Santos, R. C. Pereira, and A. Barros, "ydata-profiling: Accelerating data-centric AI with high-quality data," *Neurocomputing*, vol. 554, 2023, doi: 10.1016/j.neucom.2023.126585.

[17] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Sci. Data*, vol. 6, no. 1, 2019, doi:

10.1038/s41597-019-0103-9.

[18] Y. Song *et al.*, “Prediction of depression onset risk among middle-aged and elderly adults using machine learning and Canadian Longitudinal Study on Aging cohort,” *J. Affect. Disord.*, vol. 339, 2023, doi: 10.1016/j.jad.2023.06.031.