Benchmarking 50 classification algorithms on 50 gene-expression datasets

Stephen R. Piccolo[1,*], Avery Mecham[1], Nathan P. Golightly[1], Jérémie L. Johnson[1], Dustin B. Miller[1]

1 - Department of Biology, Brigham Young University, Provo, UT, USA

* - Please address correspondence to S.R.P. at stephen_piccolo@byu.edu.

Abstract

By classifying patients into subgroups, clinicians can provide more effective care than using a uniform approach for all patients. Such subgroups might include patients with a particular disease subtype, patients with a good (or poor) prognosis, or patients most (or least) likely to respond to a particular therapy. Diverse types of biomarkers have been proposed for assigning patients to subgroups. For example, DNA variants in tumors show promise as biomarkers; however, tumors exhibit considerable genomic heterogeneity. As an alternative, transcriptomic measurements reflect the downstream effects of genomic and epigenomic variations. However, high-throughput technologies generate thousands of measurements per patient, and complex dependencies exist among genes, so it may be infeasible to classify patients using traditional statistical models. Machine-learning classification algorithms can help with this problem. However, hundreds of classification algorithms exist—and most support diverse hyperparameters—so it is difficult for researchers to know which are optimal for gene-expression biomarkers. We performed a benchmark comparison, applying 50 classification algorithms to 50 gene-expression datasets (143 class variables). We evaluated algorithms that represent diverse machine-learning methodologies and have been implemented in general-purpose, open-source, machine-learning

libraries. When available, we combined clinical predictors with gene-expression data. Additionally, we evaluated the effects of performing hyperparameter optimization and feature selection in nested cross-validation folds. Kernel- and ensemble-based algorithms consistently outperformed other types of classification algorithms; however, even the top-performing algorithms performed poorly in some cases. Hyperparameter optimization and feature selection typically improved predictive performance, and univariate feature-selection algorithms outperformed more sophisticated methods. Together, our findings illustrate that algorithm performance varies considerably when other factors are held constant and thus that algorithm selection is a critical step in biomarker studies.

Author Summary

Keywords: biomarker, gene expression, classification, transcriptome, machine learning, data science, translational bioinformatics, predictive analytics

Introduction

Researchers use observational data to derive categories, or classes, into which patients can be assigned. Such classes might include patients who have a given disease subtype, patients at a particular disease stage, patients who respond to a particular treatment, patients who have poor outcomes, patients who have a particular genomic lesion, etc. Subsequently, a physician may use these classes to tailor patient care, rather than using a one-size-fits-all approach(1–3). However, physicians typically do not know in advance which classes are most relevant for each patient. Thus a key challenge is defining objective and reliable criteria for assigning individual patients to known classes. When such criteria have been identified and sufficiently validated, they can be used in medical “expert systems” for classifying individual patients(4).

Clinical observations are the principle form of data that physicians use to classify patients. Such data are collected through direct observation, interviews, imaging, laboratory tests, and other means. However, molecular observations have also begun to be used for classifying patients. For example, germline mutations in *BRCA1* and *BRCA2* predict responses to poly ADP ribose polymerase inhibitors(5–7). The *BCR-ABL1* fusion gene (also known as the Philadelphia translocation)(8) is a biomarker for Imatinib response in diverse types of leukemias, especially chronic myeloid leukemias(9,10). In another example, patients with metastatic melanoma are candidates for proteasome inhibitors when they have a mutation in the *BRAF* gene(11,12). However, these single-mutation markers have limited utility because the mutations occur in a small subset of patients. Therefore, researchers are developing DNA panels and using "mutation signatures" to account for multiple mutations simultaneously(13–16). Despite the advantages of these approaches, DNA variation does not necessarily predict cellular activity or its downstream effects, nor does it account for epigenetic processes that regulate gene expression(17). As an alternative, protein expression may be used as a molecular biomarker. For example, prostate specific antigen is used in many countries to diagnose prostate cancer and to estimate disease progression; however, its sensitivity and specificity are limited, so it has not become a global standard of care(18). Quantitative proteomics can be used for multimarker panels and may eventually become the preferred medium for molecular signatures because protein levels reflect the downstream effects of genomic, epigenomic, and transcriptional events(19). Proteomics technologies are already being used to guide disease-related classification(20), but these efforts are still in their infancy(21), in part due to the time-consuming and expensive processes required to generate proteomic data(21,22). In contrast, gene-expression profiling technologies are relatively mature and used widely in research(23,24). In addition, gene-expression profiling is now used in clinical settings. For example, physicians use the *PAM50* classifier, based on the expression of 58 genes, to assign breast-cancer patients to "intrinsic subtypes"(25,26). This classifier has received approval from both the US Food and Drug Administration and the European Medicines Agency(27–29), and physicians use it to match patients with treatments and to predict metastasis risk. The success of the *PAM50* classifier has motivated much additional research. In

breast cancer alone, more than 100 gene-expression profiles have been proposed for predicting breast-cancer prognosis(30).

Classification algorithms learn from data much as a physician does—past observations inform decisions about new patients. Thus the first step in developing a gene-expression biomarker is to profile a patient cohort that represents the population of interest. Alternatively, a researcher may use publicly available data for this step. Second, the researcher performs a preliminary evaluation of the potential to assign patients to a particular clinically relevant class based on gene-expression profiles and accompanying clinical information. Furthermore, the researcher may undergo an effort to select a classification algorithm that will perform relatively well for this particular task. Such efforts may be informed by prior experience, a literature review, or trial and error. Using some form of subsampling(31) and a given classification algorithm, the researcher derives a classification model from a subset of the patients' data (training data); to derive this model, the researcher exposes the classification algorithm to the true class labels for each patient. Then, using a disjoint subset of patient observations for which the true class labels have been withheld (test data), the model predicts the label of each patient. Finally, the researcher compares the predictions against the true labels. If the predictive performance approaches or exceeds what can be attained using currently available models, the researcher may continue to refine and test the model. Such steps might include tuning the algorithm, reducing the number of predictor variables, and testing it on multiple, independent cohorts. In this study, we focus on the preliminary processes of selecting algorithm(s).

Modern, high-throughput technologies can produce more than 10,000 gene-expression measurements per biological sample. Thus instead of a traditional approach that uses prior knowledge to determine which genes are included in a predictive model, researchers can use a data-driven approach to infer which genes are most relevant and to identify expression patterns that differ among patient groups(32). These patterns may be highly complex, representing subtle differences in expression that span many genes(33). Due to dependencies among biomolecules and limitations in measurement technologies, high-throughput gene-

92 expression measurements are often redundant and noisy(34). Thus, to be effective at inferring relevant
93 patterns, classification algorithms must be able to overcome these challenges.

94 The machine-learning community has developed hundreds of classification algorithms, spanning diverse
95 methodological approaches(35). Historically, most datasets available for testing had fewer than 100
96 predictor variables, so most algorithms were created and optimized for that use case(36). Consequently,
97 the execution time and predictive performance of many classification algorithms may be unsatisfactory
98 when datasets consist of thousands of predictor variables–the algorithms may have difficulty identifying
99 the most informative features in the data(37,38). When gene-expression microarrays became common in
100 biomedical research in the early 2000s, researchers began exploring the potential to make clinically
101 relevant predictions and overcome these challenges(39–43). As a result of data-sharing policies, gene-
102 expression datasets were increasingly available in the public domain, and researchers performed
103 benchmark studies, comparing the effectiveness of classification algorithms on gene-expression
104 data(32,44–46). Each of these studies evaluated between 5 and 21 algorithmic variants. In addition, the
105 authors typically used at least one method of *feature selection*, a way to reduce the number of predictor
106 variables. The studies used as many as 7 datasets, primarily from tumor cells (and often adjacent normal
107 cells). The authors focused mostly on classical algorithms, including k-Nearest Neighbors(47), linear
108 discriminant analysis(48), and the multi-layer perceptron(49). Pochet, et al. also explored the potential for
109 nonlinear Support Vector Machine (SVM) classifiers to increase predictive performance relative to linear
110 methods(45,50).

111 Later benchmark studies highlighted two types of algorithm—SVM and random forests(51)—that
112 perform relatively well on gene-expression data(38,52–54). Statnikov, et al. examined 22 datasets and
113 specifically compared the predictive capability of these two algorithm types. Overall, they found that
114 SVMs significantly outperformed random forests, although random forests outperformed SVMs in some
115 cases(38). Perhaps in part due to these highly cited studies, SVMs and random forests have been used
116 heavily in diverse types of biomedical research over the past two decades(55).

Community efforts—especially the Sage Bionetworks DREAM Challenges and Critical Assessment of Massive Data Analysis challenges(56–58)—have encouraged the development and refinement of predictive models to address biomedical questions. In these benchmark studies, the priority is to maximize predictive performance and thus increase the potential that the models will have practical use. Accordingly, participants have flexibility to use alternative normalization or summarization methods, to use alternative subsets of the training data, to combine algorithms, etc. These strategies often prove useful; however, this heterogeneity of approaches makes it difficult to deconvolve the relationship between a given solution's performance and the underlying algorithm(s) used.

In recent years, new algorithms and algorithmic variants have been developed and are available in open-source software packages. These include classification algorithms as well as feature-selection algorithms. Gene-expression datasets are more abundant in public repositories, affording opportunities for larger-scale benchmark comparisons. Furthermore, many of these datasets are accompanied by clinically oriented predictor variables. To our knowledge, no benchmark study to date has systematically compared the ability to classify patients using clinical data versus gene-expression data—or combined these two types of data—for a large number of datasets. Moreover, previous benchmarks have not systematically evaluated the benefits of optimizing an algorithm's hyperparameters versus using defaults. Accordingly, we address these gaps with a benchmark study spanning 50 datasets (143 class variables covering diverse phenotypes), 50 classification algorithms (1008 hyperparameter combinations), and 14 feature-selection algorithms. We perform this study in a staged design, allowing us to compare the ability to classify patients using gene-expression data alone, clinical data alone, or both data types together. In addition, we evaluate the effects of performing hyperparameter optimization or feature selection.

Our primary motivation is to provide helpful advice for practitioners. Identifying algorithm(s) and hyperparameter(s) that perform consistently well in this setting may ultimately lead to patient benefits. Accordingly, we questioned whether SVM and random forests algorithms would continue to be the top performers when compared against diverse types of classification algorithms. We also questioned whether

there would be scenarios in which these algorithms would perform poorly. Furthermore, relatively little is known about the extent to which algorithm choice affects predictive success for a given dataset. Thus we questioned how much variance in predictive performance we would see across the algorithms. In addition, we evaluated practical matters such as tradeoffs between predictive performance and execution time, the extent to which algorithm rankings are affected by the performance metric used, and which algorithms behave most similarly—or differently—to each other.

Results

General trends

We evaluated the predictive performance of 50 classification algorithms on 50 gene-expression datasets. Across the 50 datasets, we made predictions for a total of 143 class variables. We divided the analysis into 5 stages as a way to assess benefits that might come from including clinical predictors, optimizing an algorithm's hyperparameters, or performing feature selection (Figure 1).

In Analysis 1, we used only gene-expression data as predictors and used default hyperparameters for each classification algorithm. Figure S1 illustrates the performance of these algorithms using area under the receiver operating characteristic curve (AUROC) as a performance metric. As a method of normalization, we ranked the classification algorithms for each combination of dataset and class variable. Two patterns emerged. Firstly, the top-15 algorithms use linear-decision boundaries, kernel functions, and/or ensembles of decision trees. Secondly, though some algorithms performed consistently well overall, they performed quite poorly in some cases. For example, the `sklearn/logistic_regression` algorithm—which used the LibLinear solver[Fan2008], a *C* value of 1.0, and no class weighting—resulted in the best average rank; yet for 7 (4.9%) of the dataset/class combinations, its performance ranked in the bottom quartile. The `mlr/randomForestSRC` algorithm resulted in the second-best average rank; yet for 8 (5.6%) of dataset/class combinations, its performance ranked in the bottom quartile.

Performance rankings differed considerably depending on which evaluation metric we used. For example, in Analysis 1, many of the same algorithms that performed well according to AUROC also performed well according to classification accuracy (Figure S2). However, classification accuracy does not account for class imbalance and thus may rank algorithms in a misleading way. For example, the `weka/ZeroR` algorithm is ranked 17th among the algorithms according to classification accuracy, even though the algorithm simply selects the majority class. (Our analysis included two-class and multi-class problems.) Rankings for the Matthews correlation coefficient were relatively similar to AUROC. For example, `sklearn/logistic_regression` had the 2nd-best average rank according to this metric. However, in other cases, the rankings were considerably different. For example, the `mlr/sda` algorithm performed 3rd-best according to MCC but 26th according to AUROC (Figure S3). Figure 2 shows the rankings for each algorithm across all metrics that we evaluated, highlighting the reality that conclusions drawn from benchmark comparisons of classification algorithms depend heavily on which metric(s) are considered important.

Execution times differed substantially across the algorithms. For Analysis 1, Figure 3 categorizes each algorithm according to its ability to make effective predictions in combination with the computer time required to execute the classification tasks. The `sklearn/logistic_regression` algorithm not only outperformed other algorithms in terms of predictive ability but also was one of the fastest algorithms. In contrast, the `mlr/randomForest` algorithm was among the most predictive algorithms but was orders-of-magnitude slower than other top-performing algorithms.

Some classification algorithms are commonly used and thus have been implemented in multiple machine-learning packages. For example, all three open-source libraries that we used in this study have implementations of the SVM and random forests algorithms. However, these implementations differ from each other, often supporting different hyperparameters or using different default values. For example, `mlr/svm` and `weka/LibSVM` both use the LibSVM package(59), a value of 1.0 for the *C* parameter, and the *Radial Basis Function* kernel. However, by default, `mlr/svm` scales numeric values to zero

mean and unit variance, whereas `weka/LibSVM` performs no normalization by default. In Analysis 1, the predictive performance was similar for these different implementations. Their AUROC values were significantly correlated (r = 0.87; CI = 0.82-0.90; p = 2.2e-16). However, in some instances, their performance differed dramatically. For example, when predicting drug responses for dataset GSE20181, `weka/LibSVM` performed 2nd best, but `mlr/svm` performed worst among all algorithms. Figures S4-S5 illustrate for two representative datasets that algorithms with similar methodologies often produced similar predictions; but these predictions were never perfectly correlated. Execution times also differed from one implementation to another; for example, the median execution time for `weka/LibSVM` was 27.9 seconds, while `mlr/svm` was 114.4 seconds. Overall, the median execution times differed significantly across the software packages (Kruskal-Wallis test; p-value = 5.0e-07); the sklearn algorithms executed faster than algorithms from other packages (Figure 3).

Some classification labels were easier to predict than others. Across the dataset/class combinations in Analysis 1, the median AUROC across all algorithms ranged between 0.441 and 0.966 (Additional Data File 1). For a given dataset/class combination, algorithm performance varied considerably, though this variation was influenced partially by the `weka/ZeroR` results, which we used as controls. To gain insight into predictive performance for different types of class labels, we assigned a category to each class variable (Figure S6); the best predictive performance was attained for class variables representing molecular markers, histological statuses, and diagnostic labels. Class variables in the "patient characteristics" category performed worst; these variables represented miscellaneous factors such as the patient's family history of cancer, whether the patient had been diagnosed with multiple tumors, and the patient's physical and cognitive "performance status" at the time of diagnosis.

Effects of using gene-expression predictors, clinical predictors, or both

In Analysis 2, we used only clinical predictors (for the dataset / class-variable combinations with available clinical data). These results differed considerably from Analysis 1, which used only gene-

expression predictors. Three linear-discriminant classifiers performed particularly well: `mlr/sda`, `sklearn/lda`, and `mlr/glmnet` (Figure S7). Two Naïve Bayes algorithms also ranked among the top performers, whereas these algorithms had performed poorly in Analysis 1. Only two kernel-based algorithms were ranked among the top 10: `weka/LibLINEAR` and `sklearn/logistic_regression`. Both of these algorithms used the LibLINEAR solver. Most of the remaining kernel-based algorithms were among the worst performers. As with Analysis 1, most ensemble-based algorithms ranked in the top 25, but none ranked in the top 5.

Additional Data File 2 shows the performance of each combination of dataset and class variable in Analysis 2. As with Analysis 1, we observed considerable variation in our ability to predict particular classes and categories (Figure S8). For approximately two-thirds of the dataset/class combinations, AUROC values decreased—sometimes by more than 0.3 (Figure 4A); however, in a few cases, predictive performance increased. The most dramatic improvement was for GSE58697, in which we predicted progression-free survival for desmoid tumors. The clinical predictors were age at diagnosis, biological sex, and tumor location. Salas, et al. previously found in a univariate analysis that age at diagnosis was significantly correlated with progression-free survival (60). We focused on patients who experienced relatively long or short survival times and used multivariate methods.

In Analysis 3, we combined clinical and gene-expression predictors. We limited this analysis to the 108 dataset / class-variable combinations for which clinical predictors were available (Additional Data File 3; Figure S9). As with Analysis 1, kernel- and ensemble-based algorithms performed best overall (Figure S10). For 90 (83.3%) of the dataset/ class-variable combinations, the AUROC values were identical to Analysis 1 (Figure 4B). Except in three cases, the absolute change in AUROC was smaller than 0.05, including for GSE58697 (0.026 increase). These results suggest that standard classification algorithms (using default parameters) are not well suited for datasets in which gene-expression and clinical predictors have simply been merged. The abundance of gene-expression variables may distract the algorithms and/or

obfuscate signal from the relatively few clinical variables. Additionally, gene-expression and clinical predictors may carry redundant signals.

Effects of performing hyperparameter optimization

In Analysis 4, we performed hyperparameter optimization via nested cross validation. Across all 50 classification algorithms, we employed 1008 distinct hyperparameter combinations under the assumption that the default settings may be suboptimal for the datasets we evaluated. When clinical predictors were available, we included them (as in Analysis 3). When no clinical predictors were available, we used gene-expression data only (as in Analysis 1). Again, kernel- and ensemble-based algorithms performed well overall (Figure S11), although the individual rankings differed modestly from the previous analyses. The `weka/LibLINEAR` algorithm had the best median rank, while algorithms based on random forests were generally ranked lower than in previous analyses. For a majority of dataset / class-variable combinations, the AUROC (median across all classification algorithms) improved with hyperparameter optimization (Figure 5A); however, in some cases, performance decreased.

The best- and worst-performing class variables and categories were similar to the previous analyses (Figure S12; Additional Data File 4). We observed a positive trend in which datasets with larger sample sizes resulted in higher median AUROC values (Figure S13); however, this relationship was not statistically significant (Spearman's rho = 0.12; p = 0.15). We observed a slightly negative trend between the number of genes in a dataset and median AUROC (Figure S14), but again this relationship was not statistically significant (rho = -0.06; p = 0.47).

Evaluating many hyperparameter combinations enabled us to quantify how much the predictive performance varied for different combinations. Some variation is desirable because it enables algorithms to adapt to diverse analysis scenarios; however, large amounts of variation may make it difficult to select hyperparameter combinations that are broadly useful. For some classification algorithms, AUROC values varied widely across hyperparameter combinations when applied to a given dataset / class variable

(Figure S15). These variations were often different for algorithms with similar methodological approaches. For example, the median coefficient of variation was 0.22 for the `sklearn/svm` algorithm but 0.08 for `mlr/svm` and 0.06 for `weka/LibSVM`. In other cases, AUROC varied little across hyperparameter combinations. For example, the four algorithms with the highest median AUROC—`weka/LibLINEAR`, `mlr/glmnet`, `sklearn/logistic_regression`, and `sklearn/extra_trees`—had median coefficients of variation of 0.02, 0.03, 0.01, and 0.03, respectively. For each of these algorithms, we plotted the performance of all hyperparameter combinations across all dataset / class-variable combinations (Figures S16-S19). The default hyperparameter combination failed to perform best for any of these algorithms. Indeed, for two of the four algorithms, the default combination performed *worst*.

Of the 1008 total combinations, 984 were considered best for at least one dataset / class-variable combination (based on average performance in inner cross-validation folds).

Effects of performing feature selection

In Analysis 5, we performed feature selection via nested cross validation. We used 14 feature-selection algorithms in combination with each of the 50 classification algorithms. Due to the computational demands of evaluating these 700 combinations, we used default hyperparameters for both types of algorithm. The feature-selection algorithms differed in their methodological approaches (Table 1). Some were univariate methods, while others were multivariate. Some feature-selection algorithms mirrored the behavior of classification algorithms (e.g., SVMs or random forests); others were based on statistical inference or entropy-based metrics.

Once again, kernel- and ensemble-based classification algorithms performed best overall when feature selection was used (Figure 6). The median improvement per dataset / class-variable combination was slightly larger for feature selection than for hyperparameter optimization, and the maximal gains in predictive performance were larger for feature selection (Figure 5B, Additional Data File 5). Overall,

there was a strong positive correlation between AUROC values for Analyses 4 and 5 (Spearman's rho = 0.75; Figure S20). Among the 10 dataset / class-variable combinations that improved most after feature selection, 8 were associated with prognostic, stage, or patient-characteristic variables—categories that were most difficult to predict overall (Figure S21). The remaining two combinations were molecular markers (HER2-neu and progesterone receptor status).

Across all classification algorithms, the `weka/Correlation` feature-selection algorithm resulted in the best predictive performance (Figure S22), despite being a univariate method. This algorithm calculates the Pearson's correlation coefficient between each feature and the class values, a relatively simple approach that also ranked among the fastest (Figure S23). Other univariate algorithms were among the top performers. To characterize algorithm performance further, we compared the feature ranks between all algorithm pairs for two of the datasets. Some pairs produced highly similar gene rankings, whereas in other cases the similarity was low (Figures S24-S25). The `weka/Correlation` and `mlr/kruskal.test` algorithms produced similar feature ranks; both use statistical inference; the former is a parametric method, while the latter is nonparametric.

Some classification algorithms (e.g., `weka/ZeroR` and `sklearn/decision_tree`) performed poorly irrespective of feature-selection algorithm, whereas other classification algorithms (e.g., `mlr/ranger` and `weka/LibLINEAR`) performed consistently well across feature-selection algorithms (Figure S26). The performance of other algorithms was more variable.

Finally, as a way to provide guidance to practitioners, we examined interactions between individual feature-selection algorithms and classification algorithms (Figure 7). If a researcher had identified a particular classification algorithm to use, they might wish to select a feature-selection algorithm that performs well in combination with that classification algorithm. A feature-selection algorithm that performs well overall may not perform especially well in combination with a given classification algorithm. For example, the `weka/Correlation` feature-selection algorithm performed best overall,

but it was only the 6th-best algorithm on average when `sklearn/logistic_regression` was used for classification. In contrast, a feature-selection algorithm that underperforms in general may perform well in combination with a given classification algorithm. For example, `sklearn/svm_rfe` performed poorly overall but was effective in combination with `mlr/svm`.

Discussion

The overarching purpose of our benchmark study was to provide insights that might inform gene-expression biomarker studies. Such insights could lead to more accurate predictions in future studies and thus benefit patients. In situations where a biomarker is applied to thousands of cancer patients, even modest increases in accuracy can benefit hundreds of patients. We also sought to help bridge the gap between machine-learning researchers who develop general-purpose algorithms and biomedical researchers who seek to apply them in a specific context. When selecting algorithm(s), hyperparameters, and features to use in a gene-expression biomarker study, researchers might base their decisions on what others have reported in the literature for a similar study; or they might consider anecdotal experiences that they or their colleagues have had. However, these decisions may lack an empirical basis and not generalize from one analysis to another. Alternatively, researchers might apply many algorithms to their data to estimate which algorithm(s) will perform best. However, this approach is time- and resource-intensive and may lead to bias if the comparisons are not performed in a rigorous manner. In yet another approach, researchers might develop a custom classification algorithm, perhaps one that is specifically designed for the target data. However, it is difficult to know whether such an algorithm would outperform existing, classical algorithms.

Many factors can affect predictive performance in a biomarker study. These factors include data-generation technologies, data normalization / summarization processes, validation strategies, and evaluation metrics used. Although such factors must be considered, we have shown that when holding

them constant, the choice of algorithm, hyperparameter combination, and features usually affects predictive performance for a given dataset—sometimes dramatically. Despite these variations, we have demonstrated that particular algorithms and algorithm categories consistently outperform others across diverse gene-expression datasets and class variables. However, even the best algorithms performed poorly in some cases. These findings support the theory that no single algorithm is universally optimal(61). But they also suggest that researchers can increase the odds of success in developing accurate biomarkers by focusing on a few top-performing algorithms and by using hyperparameter optimization and/or feature selection, despite the additional computational demands in performing these steps.

This benchmark study is considerably larger than any prior study of classification algorithms applied to gene-expression data. We deliberately focused on general-purpose algorithms because they are readily available in well-maintained, open-source packages. Of necessity, we evaluated an inexhaustive list of algorithms and hyperparameter combinations. Other algorithms or hyperparameter combinations may have performed better than those that we used.

Some algorithms had more hyperparameter combinations than others, which may have enabled those algorithms to adapt better in Analysis 4. Additionally, in some cases, our hyperparameter combinations were inconsistent between two algorithms of the same type because different software libraries support different options. Despite these limitations, a key advantage of our benchmarking approach is that we performed these comparisons in an impartial manner, not having developed any of the algorithms that we evaluated nor having any other conflict of interest that might bias our results.

Generally, kernel- and ensemble-based algorithms outperformed other types of algorithms in our analyses. Other algorithm types—such as linear-discriminant and neural-network algorithms—performed well in some scenarios. Deep neural networks have received vast attention in the biomedical literature over the past decade(62); however, the `mlr/h2o.deeplearning` algorithm performed at mediocre levels in all of our analyses. Custom adaptations to this (or any other) deep-learning algorithms may improve predictive performance in future studies. Future efforts to improve predictive ability might also

include optimizing hyperparameters of feature-selection algorithms, combining hyperparameter-optimized classification algorithms with feature selection, and using multiple classifier systems(63). Transfer learning across datasets may also prove fruitful(64).

Our findings are specific to high-throughput gene-expression datasets that have either no clinical predictors or a small set of clinical predictors. However, our conclusions may have relevance to other datasets that include a large number of features and that may include a combination of numeric, discrete, and nominal features.

Finally, we mention additional limitations and caveats. We applied Monte Carlo cross validation to each dataset separately and thus did not evaluate predictive performance in independent datasets. This approach was suitable for our benchmark comparison because our priority was to compare algorithms against each other rather than to optimize their performance for clinical use. On another note, comparisons across machine-learning packages are difficult to make. For example, some *sklearn* algorithms provided the ability to automatically address class imbalance, whereas other software packages often did not provide this functionality. Adapting these weights manually was infeasible for this study. In addition, some classification algorithms are designed to produce probabilistic predictions, whereas other algorithms produce only discrete predictions. The latter algorithms may have been at a disadvantage in our benchmark for the AUROC and other metrics.

Methods

Data preparation

We used 50 datasets spanning diverse diseases and tissue types but focused primarily on cancer-related conditions. We used data from two sources. The first was a resource created by Golightly, et al.(65) that includes 45 datasets from Gene Expression Omnibus(66). For these datasets, the gene-expression data were generated using Affymetrix microarrays, normalized using Single Channel Array

Normalization(67), summarized using BrainArray annotations(68), quality checked using IQRay(69) and DoppelgangR(70), and batch-adjusted (where applicable) using ComBat(71). Depending on the Affymetrix platform used, expression levels were available for 11,832 to 21,614 genes. For the remaining 5 datasets, we used RNA-Sequencing data from The Cancer Genome Atlas (TCGA)(72), representing 5 tumor types: colorectal adenocarcinoma (COAD), bladder urothelial carcinoma (BLCA), kidney renal clear cell carcinoma (KIRC), prostate adenocarcinoma (PRAD), and lung adenocarcinoma (LUAD). These data had been aligned and quantified using the Rsubread and featureCounts packages(73,74), resulting in transcripts-per-million values for 22,833 genes(75). All gene-expression data were labeled using Ensembl gene identifiers(76).

For the microarray datasets, we used the class variables and clinical variables identified by Golightly, et al. (2.8 class variables per dataset)(65). For the RNA-Sequencing datasets, we identified a total of 16 class variables. When a given sample was missing data for a given class variable, we excluded that sample from the analyses. Some class variables were continuous in nature (e.g., overall survival). We discretized these variables to enable classification, taking into account censor status where applicable. To support consistency and human interpretability across datasets, we assigned a standardized name and category to each class variable; the original and standardized names are available in Additional Data File 6.

For most of the Golightly, et al. datasets, at least one clinical variable had been identified as a potential predictor variable. For TCGA datasets, we selected multiple clinical-predictor variables per dataset. Across all datasets, the mean and median number of clinical predictors per dataset were 3.1 and 2.0, respectively (Additional Data File 6). We avoided combinations of clinical-predictor variables and class variables that were potentially confounded. For example, when a dataset included cancer stage as a class variable, we excluded predictor variables such as tumor grade or histological status because oncologists might use those data to determine stage. In some cases, no suitable predictor variable was available for a given class variable, leaving only gene-expression variables as predictors; this was true for 35 class variables.

Algorithms used

We used 50 classification algorithms that were implemented in the ShinyLearner tool, which enables researchers to benchmark algorithms that are included in open-source machine-learning libraries; these libraries are redistributed as software containers(77,78). Via ShinyLearner, we used algorithm implementations from the *mlr* R package (version 2; R version 3.5)(79), *sklearn* Python module (versions 0.18-0.22)(80), and *Weka* Java application (version 3.6)(81). Table 2 lists each algorithm that we used, along with a description and methodological category for each algorithm. Furthermore, it indicates the open-source software package that implemented the algorithm, as well as the number of unique hyperparameter combinations that we evaluated for each algorithm. A full list of these hyperparameter combinations can be found in Additional Data File 7. Among the classification algorithms was Weka's *ZeroR*, which predicts all instances to have the majority class. We included this algorithm in our analysis as a sanity check(82) and a baseline against which all other algorithms could be compared. Beyond the 50 classification algorithms that we used, additional algorithms were available in ShinyLearner. However, we excluded these algorithms from our analysis because they raised exceptions when we used default hyperparameters, required excessive amounts of random access memory (75 gigabytes or more), or were orders of magnitude slower than the other algorithms.

For feature selection, we used 14 algorithms that had been implemented in ShinyLearner(78). Table 1 lists each of the algorithms, along with a description and high-level category for each algorithm.

For all software implementations that supported it, we set the parameters so that the classification algorithms would produce probabilistic predictions and use a single process/thread. Unless otherwise noted, we used default hyperparameter values for each algorithm, as dictated by the respective software implementations. For feature selection, we used *n_features_to_select=5* and *step=0.1* for the *sklearn/random_forest_rfe* and *sklearn/svm_rfe* methods to balance computational efficiency with the size of the datasets we used. For *sklearn/random_forest_rfe*, we specified *n_estimators=50* because execution failed when fewer estimators were used.

To analyze the benchmark results, we wrote scripts for Python (version 3.6)(83) and the R statistical software (version 4.02)(84). We also used the corrplot(85), cowplot(86), ggrepel(87), and tidyverse(88) packages.

Analysis phases

We performed this study in five phases (Figure 1). In each phase, we modulated either the data used or the optimization approach. In Analysis 1, we used gene-expression predictors only and used default hyperparameter values for each classification algorithm. In Analysis 2, we used clinical predictors only and default hyperparameter values for each classification algorithm. In Analysis 3, we used gene-expression and clinical predictors and default hyperparameter values. In Analysis 4, we used both types of predictors and selected hyperparameter values via nested cross-validation. In Analysis 5, we used both types of predictors and selected the most relevant *n* features via nested cross validation before performing classification. Because it would be exponentially more computationally expensive to perform hyperparameter optimization in this phase, we used default hyperparameter values for the feature-selection and classification algorithms.

In each phase, we used Monte Carlo cross validation. For each iteration, we randomly assigned the patient samples to either a training set or test set, stratified by class. We assigned approximately 2/3 of the patient samples to the training set. We then made predictions for the test set and evaluated the predictions using diverse metrics (see below). We repeated this process (an iteration) multiple times and used the iteration number as a random seed when assigning samples to the training or test set (unless otherwise noted). ShinyLearner relays this seed to the underlying algorithms, where applicable.

During Analysis 1, we evaluated the number of Monte Carlo iterations that would be necessary to provide a stable performance estimate. For the *mlr/randomForest*, *sklearn/svm*, and *weka/Bagging* classification algorithms, we executed 100 iterations for datasets GSE10320 (predicting relapse vs. non-relapse for Wilms tumor patients) and GSE46691 (predicting early metastasis following radical prostatectomy). As

the number of iterations increased, we calculated the cumulative average of the AUROC for each algorithm. After performing at most 40 iterations, the cumulative averages did not change more than 0.01 over sequences of 10 iterations (Figures S27-S28). To be conservative, we used 50 iterations in Analysis 1, 2, and 3. In Analysis 4 and Analysis 5, we used 5 iterations because hyperparameter optimization and feature selection are CPU and memory intensive. When optimizing hyperparameters (Analysis 4), we used Monte Carlo cross validation on each training set (5 nested iterations) to estimate which hyperparameter combination was most effective for each classification algorithm; we used AUROC as a metric in these evaluations. When performing feature selection (Analysis 5), we also used nested Monte Carlo cross validation (5 iterations). In each iteration, we ranked the features using each feature-selection algorithm and performed classification using the top-*n* features. We repeated this process for each classification algorithm and used *n* values of 1, 10, 100, 1000, and 10000. For a given combination of feature-selection algorithm and classification algorithm, we identified the *n* value that resulted in the highest AUROC. We used this *n* value in the respective outer fold. Finally, when identifying the most informative features across Monte Carlo iterations, we used the Borda Count method to combine the ranks(63).

While executing each analysis phase, we encountered some situations in which we did not obtain results for all combinations of class variable and algorithms. We describe these exceptions below.

Analysis 1. On iteration 34, the *weka/RBFNetwork* algorithm did not converge after 24 hours of execution time for one of the datasets. We manually changed the random seed from 34 to 134, and it converged in minutes.

Analysis 2. The *mlr/glmnet* algorithm failed three times due to an internal error. We limited the results for this algorithm to the iterations that completed successfully. The total number of classification problems was smaller for this analysis than for Analysis 1 because no clinical predictors were available for some class variables.

Analysis 3. Again, on iteration 34, the *weka/RBFNetwork* algorithm did not converge after 24 hours of execution time for one of the datasets. We manually changed the random seed from 34 to 134, and it converged in minutes. The total number of classification problems was less than for Analysis 1 because no clinical predictors were available for some class variables.

Analysis 4. During nested Monte Carlo cross validation, we specified a time limit of 168 hours under the assumption that some hyperparameter combinations would be especially time intensive. A total of 1022 classification tasks failed either due to this limit or due to small sample sizes. We ignored these hyperparameter combinations when determining the top-performing combinations. Most failures were associated with the `mlr/h2o.gbm` and `mlr/ksvm` classification algorithms.

Analysis 5. During nested Monte Carlo cross validation, we specified a time limit of 168 hours. A total of 574 classification tasks failed either due to this limit or due to small sample sizes. We ignored these tasks when seeking to select an optimal number of features.

Computing resources

We performed these analyses using Linux servers supported by Brigham Young University's Office of Research Computing and Life Sciences Information Technology. In addition, we used virtual servers in Google's Compute Engine environment supported by the Institute for Systems Biology and the United States National Cancer Institute Cancer Research Data Commons. When multiple central-processing cores were available on a given server, we executed tasks in parallel using GNU Parallel(89).

Performance metrics

In outer cross-validation folds, we used diverse metrics to quantify classification performance. These included accuracy (proportion of accurate predictions), AUROC(90), balanced accuracy (proportion of accurate predictions weighted by class-label frequency), Brier score(91), F1 score(92), false discovery rate (false positives divided by total number of positives), false positive rate, Matthews correlation

coefficient(93), mean misclassification error (MMCE), negative predictive value, positive predictive value (precision), and recall (sensitivity). Many of these metrics require discretized predictions; we relied on the machine-learning packages that implemented each algorithm to convert probabilistic predictions to discretized predictions.

Acknowledgements

Results from this study are in part based upon data generated by TCGA and managed by the United States National Cancer Institute and National Human Genome Research Institute (see http://cancergenome.nih.gov). We thank the patients who participated in this study and shared their data publicly. We thank the Simmons Center for Cancer Research for providing resources to facilitate this study. We thank the Fulton Supercomputing Laboratory at Brigham Young University for providing computational facilities.

Ethics approval and consent to participate

Brigham Young University's Institutional Review Board approved this study under exemption status. This study uses data collected from public repositories only. We played no part in patient recruiting or in obtaining consent.

Data Availability Statement

The data and code that we used for this analysis are available from https://osf.io/fv8td/. This repository contains raw and summarized versions of the analysis results, as well as code that we used to generate the figures and tables for this manuscript. The repository is freely available under the Creative Commons Universal 1.0 license.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

The contributions listed below correspond to the CRediT Taxonomy(94).

SRP: conceptualization, data curation, formal analysis, investigation, methodology, project administration, resources, software, supervision, visualization, writing – original draft preparation.

AM: data curation, visualization, writing – review & editing.

NPG: data curation, formal analysis, methodology, writing – review & editing.

JLJ: writing – original draft preparation, writing – review & editing.

DBM: visualization, writing – review & editing.

Tables

Table 1: Summary of feature-selection algorithms. We evaluated 14 feature-selection algorithms that were available in ShinyLearner and had been implemented across 3 open-source machine-learning libraries. The abbreviation for each algorithm contains a prefix that indicates which machine-learning library implemented the algorithm (mlr = Machine learning in R, sklearn = scikit-learn, weka = WEKA: The workbench for machine learning). For each algorithm, we provide a brief description of the algorithmic approach; we extracted these descriptions from the libraries that implemented the algorithms. In addition, we assigned high-level categories that indicate whether the algorithms evaluate a single feature (univariate) or multiple features (multivariate) at a time. In some cases, the individual machine-learning libraries aggregrated algorithm implementations from third-party packages. In these cases, we cite the machine-learning library and the third-party package. When available, we also cite papers that describe the algorithmic methodologies used.

Abbreviation	Description	Category
mlr/cforest.importance	Uses the permutation principle (based on Random Forests) to calculate standard and conditional importance of features(79,95,96)	Multivariate
mlr/kruskal.test	Uses the Kruskal-Wallis rank sum test(79,97)	Univariate
mlr/randomForestSRC.rfsrc	Uses the error rate for trees grown with and without a given feature(79,98,99)	Multivariate
mlr/randomForestSRC.var.select	Variable selection using minimal depth (Random Forests)(79,98,99)	Multivariate
sklearn/mutual_info	Calculates the mutual Information between two feature clusterings(80,100)	Univariate
sklearn/random_forest_rfe	Recursively eliminates features based on Random Forests classification(51,80)	Multivariate
sklearn/svm_rfe	Recursively eliminates features based on support vector classification(80,101)	Multivariate
weka/Correlation	Calculates Pearson's correlation coefficient between each feature and the class(81,102)	Univariate
weka/GainRatio	Measures the gain ratio of a feature with respect to the class(81,103)	Univariate

weka/InfoGain	Measures the information gain of a feature with respect to the class(81,103)	Univariate
weka/OneR	Evaluates the worth of a feature using the OneR classifier(81,104)	Univariate
weka/ReliefF	Repeatedly samples an instance and considers the value of a given attribute for the nearest instance of the same and different class(81,105)	Multivariate
weka/SVMRFE	Recursively eliminates features based on support vector classification(81,101)	Multivariate
weka/SymmetricalUncertainty	Measures the symmetrical uncertainty of a feature with respect to the class(81,106)	Univariate

Table 2: Summary of classification algorithms. We compared the predictive ability of 50 classification algorithms that were available in ShinyLearner and had been implemented across 3 open-source machine-learning libraries. The abbreviation for each algorithm contains a prefix indicating which machine-learning library implemented the algorithm (mlr = Machine learning in R, sklearn = scikit-learn, weka = WEKA: The workbench for machine learning). For each algorithm, we provide a brief description of the algorithmic approach; we extracted these descriptions from the libraries that implemented the algorithms. In addition, we assigned high-level categories that characterize the algorithmic methodology used by each algorithm. In some cases, the individual machine-learning libraries aggregrated algorithm implementations from third-party packages. In these cases, we cite the machine-learning library and the third-party package. When available, we also cite papers that describe the algorithmic methodologies used. Finally, for each algorithm, we indicate the number of unique hyperparameter combinations that we evaluated in Analysis 4.

Abbreviation	Description	Category	Combos
mlr/C50	C5.0 Decision Trees(79,107)	Tree- or rule-based	32
mlr/ctree	Conditional Inference Trees(79,108)	Tree- or rule-based	4
mlr/earth	Multivariate Adaptive Regression Splines(79,109)	Linear discriminant	36
mlr/gausspr	Gaussian Processes(79,110)	Kernel-based	3
mlr/glmnet	Generalized Linear Models with Lasso or Elasticnet Regularization(79,111)	Linear discriminant	3
mlr/h2o.deeplearning	Deep Neural Networks(79,112,113)	Artificial neural network	32
mlr/h2o.gbm	Gradient Boosting Machines(79,112,114)	Ensemble	16
mlr/h2o.randomForest	Random Forests(51,79,112)	Ensemble	12
mlr/kknn	k-Nearest Neighbor(79,115)	Miscellaneous	6

mlr/ksvm	Support Vector Machines(50,79,110)	Kernel-based	40
mlr/mlp	Multi-Layer Perceptron(49,79,116)	Artificial neural network	14
mlr/naiveBayes	Naive Bayes(79,117)	Miscellaneous	2
mlr/randomForest	Breiman and Cutler's Random Forests(79,118)	Ensemble	12
mlr/randomForestSRC	Fast Unified Random Forests for Survival, Regression, and Classification(79,98,99)	Ensemble	108
mlr/ranger	A Fast Implementation of Random Forests(79,119)	Ensemble	12
mlr/rpart	Recursive Partitioning and Regression Trees(79,120,121)	Tree- or rule-based	1
mlr/RRF	Regularized Random Forests(79,122)	Ensemble	24
mlr/sda	Shrinkage Discriminant Analysis(79,123)	Linear discriminant	2
mlr/svm	Support Vector Machines(59,79,117)	Kernel-based	28
mlr/xgboost	eXtreme Gradient Boosting(124)	Ensemble	3
sklearn/adaboost	AdaBoost(80,125)	Ensemble	8
sklearn/decision_tree	A decision tree classifier(80)	Tree- or rule-based	96
sklearn/extra_trees	An extra-trees classifier(80)	Ensemble	24
sklearn/gradient_boosting	Gradient Boosting for classification(80,114)	Ensemble	6
sklearn/knn	k-nearest neighbors vote(47,80)	Miscellaneous	12
sklearn/lda	Linear Discriminant Analysis(80)	Linear discriminant	3
sklearn/logistic_regression	Logistic Regression(80,126)	Kernel-based	32
sklearn/multilayer_perceptron	Multi-layer Perceptron(49,80)	Artificial neural network	24
sklearn/random_forest	Random Forests(51,80)	Ensemble	24

sklearn/sgd	Linear classifiers with stochastic gradient descent training(80,127)	Linear discriminant	36
sklearn/svm	C-Support Vector Classification(50,80)	Kernel-based	32
weka/Bagging	Bagging a classifier to reduce variance(81,128)	Ensemble	32
weka/BayesNet	Bayes Network learning using various search algorithms and quality measures(81,129)	Miscellaneous	2
weka/DecisionTable	Simple decision table majority classifier(81,130)	Tree- or rule-based	6
weka/HoeffdingTree	Hoeffding tree(81,131)	Tree- or rule-based	32
weka/HyperPipes	HyperPipe classifier(81)	Miscellaneous	1
weka/J48	Pruned or unpruned C4.5 decision tree(81,132)	Tree- or rule-based	96
weka/JRip	Repeated Incremental Pruning to Produce Error Reduction(81,133)	Tree- or rule-based	12
weka/LibLINEAR	LIBLINEAR - A Library for Large Linear Classification(81,134)	Kernel-based	16
weka/LibSVM	Support vector machines(59,81)	Kernel-based	32
weka/NaiveBayes	A Naive Bayes classifier using estimator classes(81,135)	Miscellaneous	3
weka/OneR	1R (1 rule) classifier(81,104)	Tree- or rule-based	3
weka/RandomForest	Forest of random trees(51,81)	Ensemble	18
weka/RandomTree	Tree that considers K randomly chosen attributes at each node(81)	Tree- or rule-based	2
weka/RBFNetwork	Normalized Gaussian radial basis function network(81)	Miscellaneous	18
weka/REPTree	Fast decision tree learner (reduced-error pruning with backfitting)(81)	Tree- or rule-based	16
weka/SimpleLogistic	Linear logistic regression models(81,136,137)	Linear discriminant	5
weka/SMO	Sequential minimal optimization for a support vector	Kernel-based	20

	classifier(81,138–140)		
weka/VFI	Voting feature intervals(81,141)	Miscellaneous	6
weka/ZeroR	0-R classifier (predicts the mean for a numeric class or the mode for a nominal class)(81)	Baseline	1

559

560

Figures

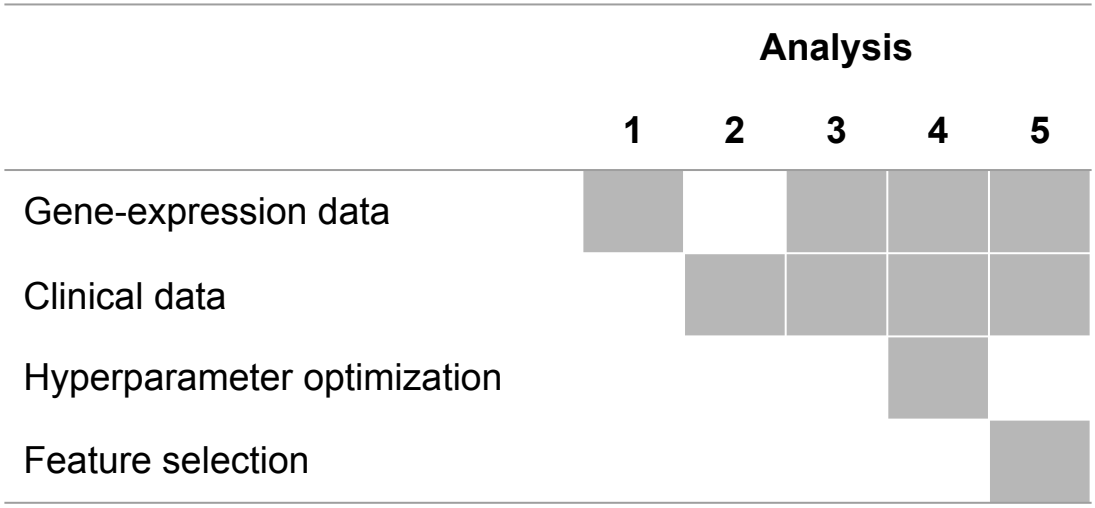


Figure 1: Overview of analysis scenarios. This study consisted of five separate but related analyses. This diagram indicates which data type(s) was/were used and whether we attempted to improve predictive performance via hyperparameter optimization or feature selection in each analysis.

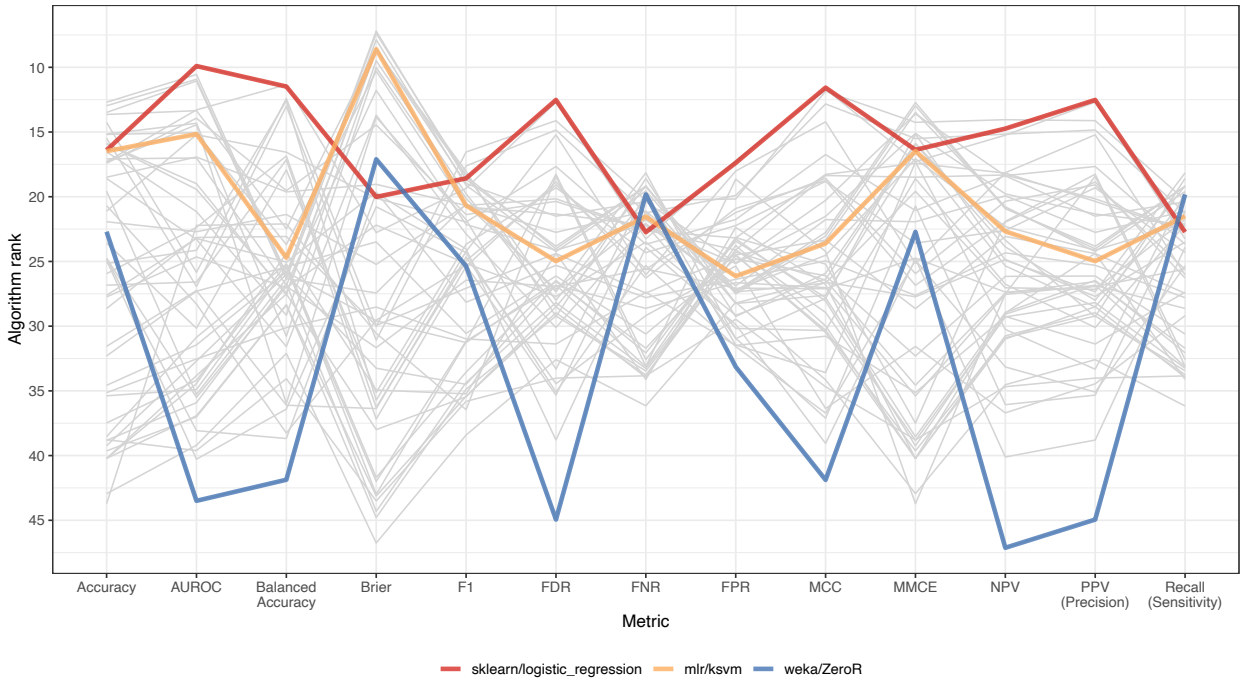
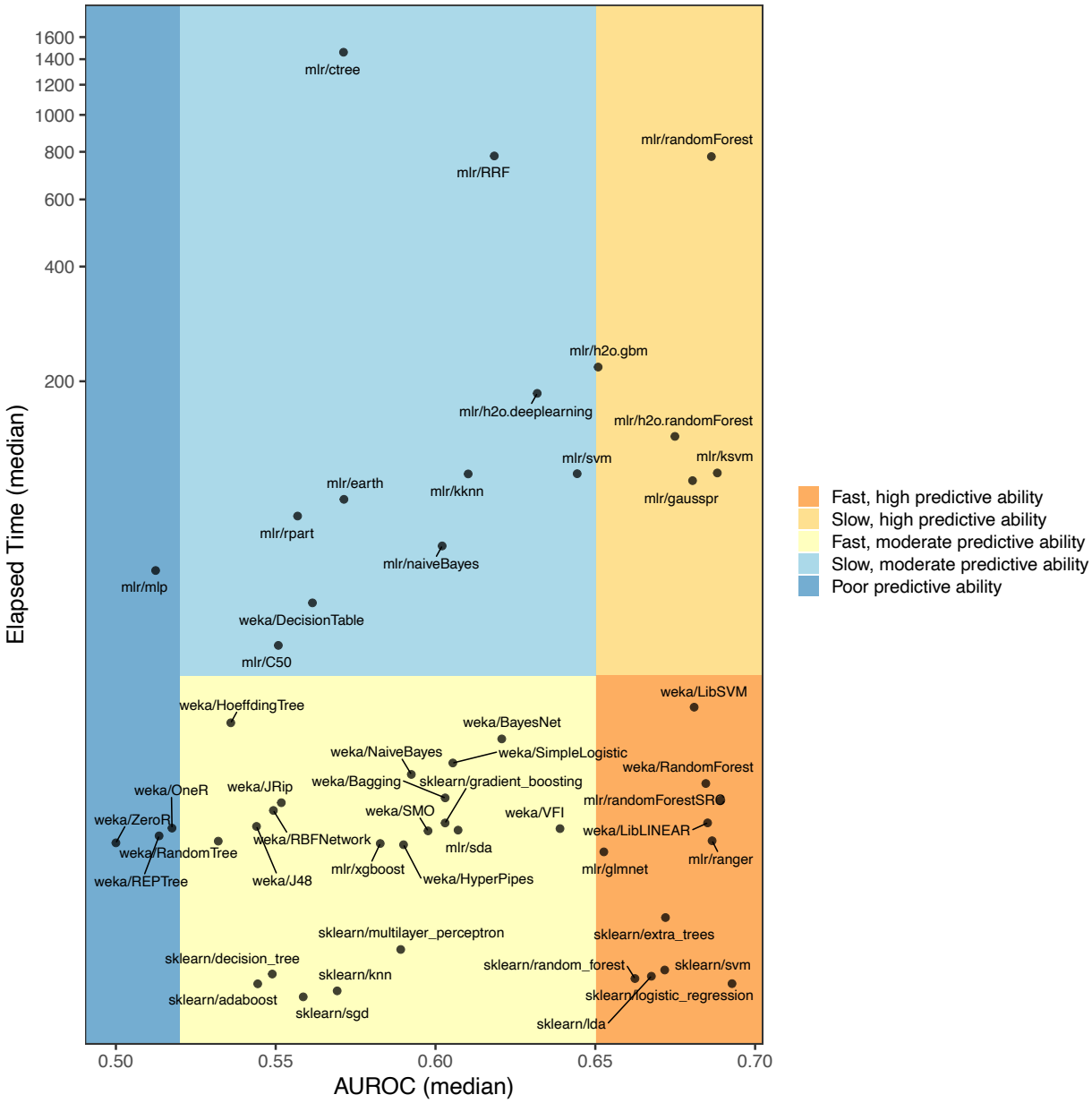


Figure 2: Comparison of ranks for classification algorithms across performance metrics. We calculated 13 performance metrics for each classification task. This graph shows results for Analysis 1 (using only gene-expression predictors). For each combination of dataset and class variable, we averaged the metric scores across all Monte Carlo cross-validation iterations. For some metrics (such as Accuracy), a relatively high value is desirable, whereas the opposite is true for other metrics (such as FDR). We ranked the classification algorithms such that relatively low ranks indicated more desirable performance for metrics and averaged these ranks across the dataset/class combinations. This graph illustrates that the best-performing algorithms for some metrics do not necessarily perform optimally according to other metrics. AUROC = area under the receiver operating characteristic curve. FDR = false discovery rate. FNR = false negative rate. FPR = false positive rate. MCC = Matthews correlation coefficient. MMCE = mean misclassification error. NPV = negative predictive value. PPV = positive predictive value.



579

580 **Figure 3: Tradeoff between execution time and predictive performance for classification**

581 **algorithms.** When using gene-expression predictors only (Analysis 1), we calculated the median area

582 under the receiver operating characteristic curve (AUROC) across 50 iterations of Monte Carlo cross

583 validation for each combination of dataset, class variable, and classification algorithm. Simultaneously,

584 we measured the median execution time (in seconds) for each algorithm across these scenarios.

585 `sklearn/logistic_regression` attained the top predictive performance and was the 4th fastest

586 algorithm (median = 5.3 seconds). Values on the y-axis have been log-transformed (base 10). We used
587 arbitrary AUROC thresholds to categorize the algorithms based on low, moderate, and high predictive
588 ability.

589

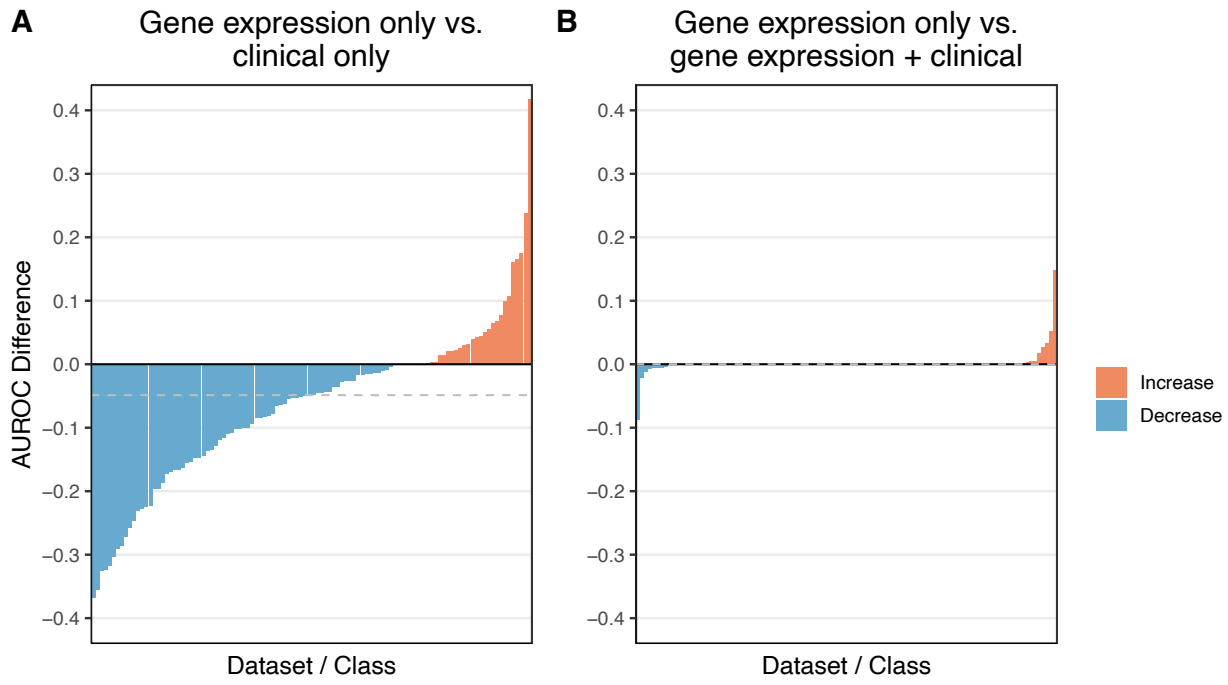
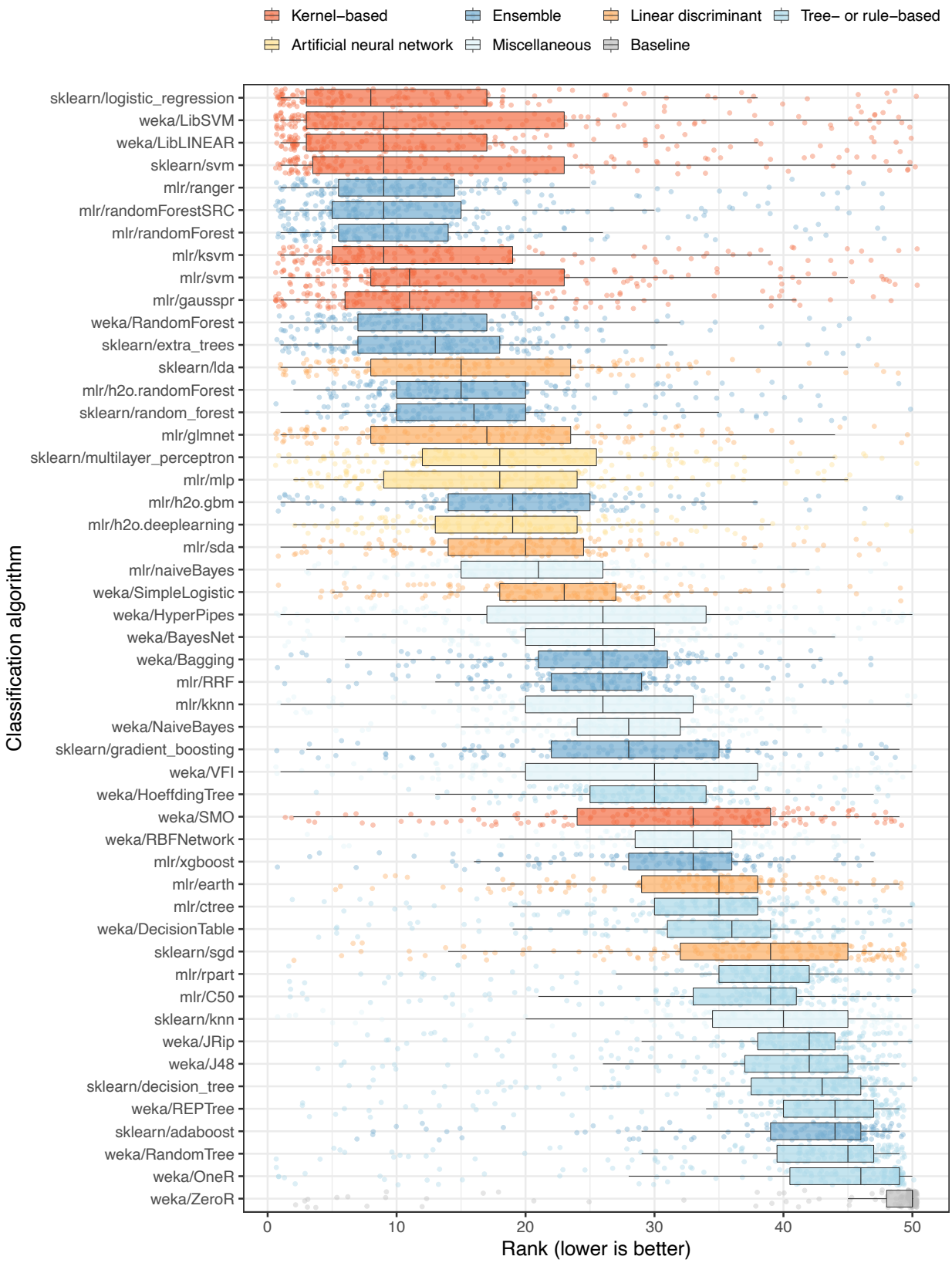


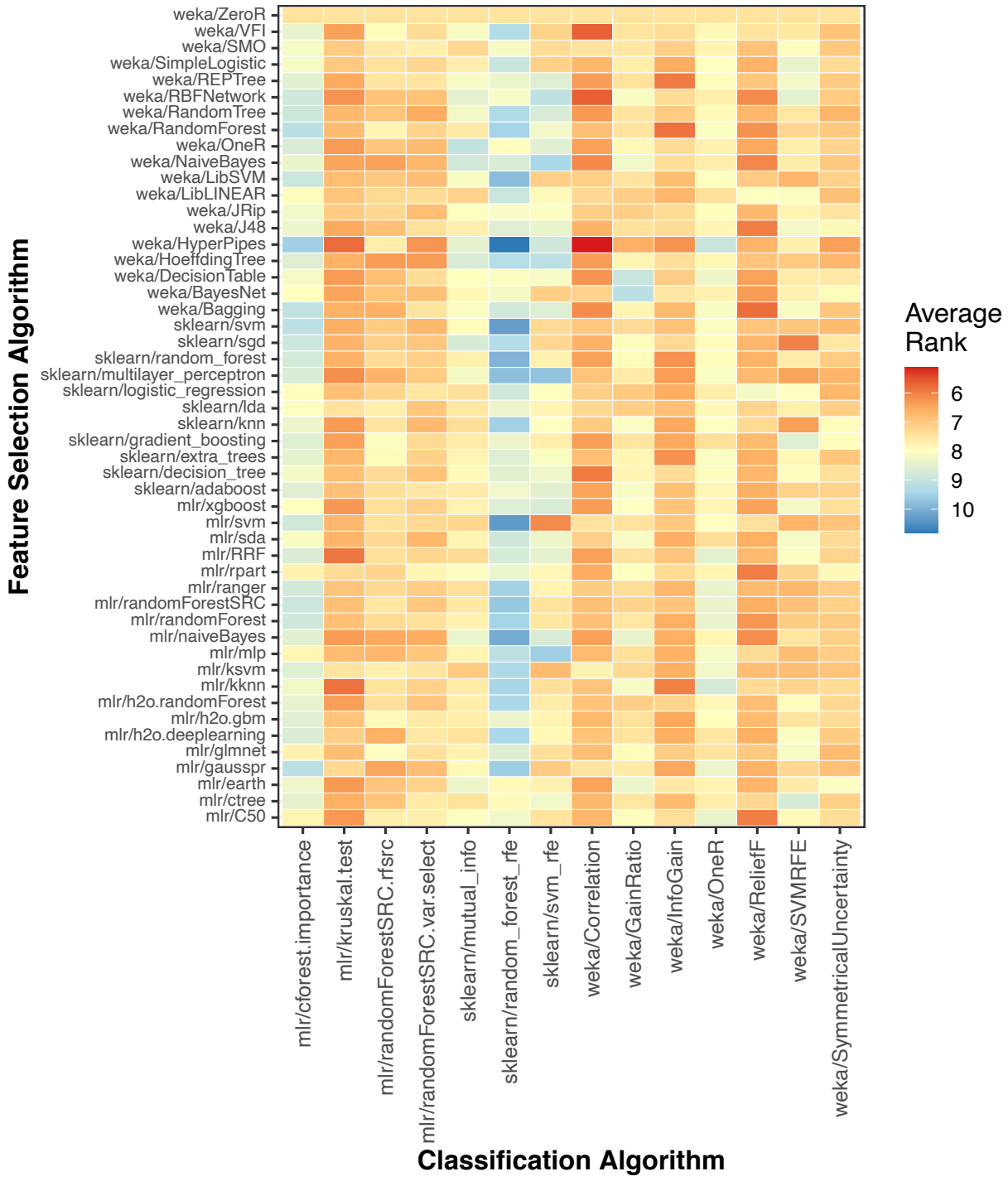
Figure 4: Relative predictive performance when training on gene-expression predictors alone vs. using clinical predictors alone or gene-expression predictors in combination with clinical predictors. In both **A** and **B**, we used as a baseline the predictive performance that we attained using gene-expression predictors alone (Analysis 1). We quantified predictive performance using the area under the receiver operating characteristic curve (AUROC). In **A**, we show the relative increase or decrease in performance when using clinical predictors alone (Analysis 2). In most cases, AUROC values decreased; however, in a few cases, AUROC values increased (by as much as 0.42). In **B**, we show the relative change in performance when using gene-expression predictors in combination with clinical predictors (Analysis 3). For 82/109 (75%) of dataset/class combinations, include clinical predictors had no effect on performance. However, for the remaining 27 combinations, the AUROC improved by as much as 0.15 and decreased by as much as 0.09.

Figure 5: Relative predictive performance when using default algorithm hyperparameters and all features vs. tuning hyperparameters or selecting features. In both **A** and **B**, we use as a baseline the predictive performance that we attained using default hyperparameters for the classification algorithms (Analysis 3). We quantified predictive performance using the area under the receiver operating characteristic curve (AUROC). In **A**, we show the relative increase or decrease in performance when tuning hyperparameters within each training set (Analysis 4). In most cases, AUROC values increased. In **B**, we show the relative change in performance when performing feature selection within each training set (Analysis 5). Predictive increased for most dataset / class-variable combinations. The horizontal dashed lines indicate the median improvement across all dataset / class-variable combinations.



612

Figure 6: Relative performance of classification algorithms using gene-expression and clinical predictors and performing feature selection. We predicted patient states using gene-expression and clinical predictors with feature selection (Analysis 5). We used nested cross validation to estimate which features would be optimal for each algorithm in each training set. For each combination of dataset, class variable, and classification algorithm, we calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values across 5 iterations of Monte Carlo cross-validation. Next we sorted the algorithms based on the average rank across all dataset/class combinations. Each data point that overlays the box plots represents a particular dataset/class combination. The algorithm rankings followed similar trends as Analyses 3 and 4 (Figures S10-S11).



622

623 **Figure 7: Relative classification performance per combinations of feature-selection and**

624 **classification algorithm.** For each combination of dataset and class variable, we averaged the area under

625 receiver operating characteristic curve (AUROC) values across all Monte Carlo cross-validation

626 iterations. Then for each classification algorithm, we ranked the feature-selection algorithms based on
627 AUROC scores across all datasets and class variables. Lower ranks indicate better performance. Dark-red
628 boxes indicate cases where a particular feature-selection algorithm was especially effective for a
629 particular classification algorithm. The opposite was true for dark-blue boxes.

630

Additional Data Files

Additional Data File 1: Summary of predictive performance per dataset when using gene-expression predictors. We predicted patient states using gene-expression predictors only (Analysis 1). For each combination of dataset, class variable, and classification algorithm, we calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values across 50 iterations of Monte Carlo cross-validation. Next we calculated the minimum, first quartile (Q1), median, third quartile (Q3), and maximum for these values across the algorithms. Finally, we sorted the algorithms in descending order based on median values. Each row represents a particular dataset/class combination.

Additional Data File 2: Summary of predictive performance per dataset when using clinical predictors. We predicted patient states using clinical predictors only (Analysis 2). For each combination of dataset, class variable, and classification algorithm, we calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values across 50 iterations of Monte Carlo cross-validation. Next we calculated the minimum, first quartile (Q1), median, third quartile (Q3), and maximum for these values across the algorithms. Finally, we sorted the algorithms in descending order based on median values. Each row represents a particular dataset/class combination. For some dataset/class combinations, no clinical predictors were available; these combinations are excluded from this file.

Additional Data File 3: Summary of predictive performance per dataset when using gene-expression and clinical predictors. We predicted patient states using gene-expression and clinical predictors (Analysis 3). For each combination of dataset, class variable, and classification algorithm, we calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values across 50 iterations of Monte Carlo cross-validation. Next we calculated the minimum, first quartile (Q1), median, third quartile (Q3), and maximum for these values across the algorithms. Finally, we sorted the algorithms in descending order based on median values. Each row represents a particular dataset/class

655 combination. For some dataset/class combinations, no clinical predictors were available; these
656 combinations are excluded from this file.

657 **Additional Data File 4: Summary of predictive performance per dataset when using gene-**
658 **expression and clinical predictors and performing hyperparameter optimization.** We predicted
659 patient states using gene-expression and clinical predictors (Analysis 4). For classification algorithms (n =
660 47) that evaluated multiple hyperparameter combinations, we selected based on performance in each
661 respective training set. For each combination of dataset, class variable, and classification algorithm, we
662 calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values
663 across 5 (outer) iterations of Monte Carlo cross-validation. Next we calculated the minimum, first quartile
664 (Q1), median, third quartile (Q3), and maximum for these values across the algorithms. Finally, we sorted
665 the algorithms in descending order based on median values. Each row represents a particular dataset/class
666 combination.

667 **Additional Data File 5: Summary of predictive performance per dataset when using gene-**
668 **expression and clinical predictors and performing feature selection.** We predicted patient states using
669 gene-expression and clinical predictors (Analysis 5). Using each respective training set, we performed
670 feature selection for each of 14 feature-selection algorithms and performed classification using *n* top-
671 ranked features. For each combination of dataset, class variable, and classification algorithm, we
672 calculated the arithmetic mean of area under the receiver operating characteristic curve (AUROC) values
673 across 5 (outer) iterations of Monte Carlo cross-validation. Next we calculated the minimum, first quartile
674 (Q1), median, third quartile (Q3), and maximum for these values across the algorithms. Finally, we sorted
675 the algorithms in descending order based on median values. Each row represents a particular dataset/class
676 combination.

677 **Additional Data File 6: Summary of datasets used.** This file contains a unique identifier for each
678 dataset, indicates whether gene-expression microarrays or RNA-Sequencing were used to generate the
679 data, and indicates the name of the class variable from the original dataset. In addition, we assigned

680 standardized names and categories as a way to support consistency across datasets. The file lists any
681 clinical predictors that were used in the analyses as well as the number of samples and genes per dataset.

682 **Additional Data File 7: Classification algorithm hyperparameter combinations** This file indicates all
683 hyperparameter combinations that we evaluated via nested cross-validation in Analysis 4.

684 References

685 1. National Research Council (US) Committee on A Framework for Developing a New Taxonomy
686 of Disease. Toward Precision Medicine: Building a Knowledge Network for Biomedical Research and a
687 New Taxonomy of Disease. Washington (DC): National Academies Press (US); 2011. (The National
688 Academies Collection: Reports funded by National Institutes of Health).
689 2. Collins FS, Varmus H. A New Initiative on Precision Medicine. N Engl J Med. 2015
690 Feb;372(9):793–5.
691 3. Krumholz HM. Big Data And New Knowledge In Medicine: The Thinking, Training, And Tools
692 Needed For A Learning Health System. Health Aff (Millwood). 2014 Jul;33(7):1163–70.
693 4. Obermeyer Z, Emanuel EJ. Predicting the Future - Big Data, Machine Learning, and Clinical
694 Medicine. N Engl J Med. 2016 Sep;375(13):1216–9.
695 5. Moore K, Colombo N, Scambia G, Kim B-G, Oaknin A, Friedlander M, et al. Maintenance
696 Olaparib in Patients with Newly Diagnosed Advanced Ovarian Cancer. N Engl J Med. 2018
697 Dec;379(26):2495–505.
698 6. Robson M, Im S-A, Senkus E, Xu B, Domchek SM, Masuda N, et al. Olaparib for Metastatic
699 Breast Cancer in Patients with a Germline BRCA Mutation. N Engl J Med. 2017 Aug;377(6):523–33.
700 7. Litton JK, Rugo HS, Ettl J, Hurvitz SA, Gonçalves A, Lee K-H, et al. Talazoparib in Patients
701 with Advanced Breast Cancer and a Germline BRCA Mutation. N Engl J Med. 2018 Aug;379(8):753–63.
702 8. Kurzrock R, Kantarjian HM, Druker BJ, Talpaz M. Philadelphia chromosome-positive leukemias:
703 From basic mechanisms to molecular therapeutics. Ann Intern Med. 2003 May;138(10):819–30.

9. Druker BJ, Talpaz M, Resta DJ, Peng B, Buchdunger E, Ford JM, et al. Efficacy and Safety of a Specific Inhibitor of the BCR-ABL Tyrosine Kinase in Chronic Myeloid Leukemia. N Engl J Med. 2001 Apr;344(14):1031–7.

10. Talpaz M, Shah NP, Kantarjian H, Donato N, Nicoll J, Paquette R, et al. Dasatinib in imatinib-resistant Philadelphia chromosome-positive leukemias. N Engl J Med. 2006 Jun;354(24):2531–41.

11. Rubinstein JC, Sznol M, Pavlick AC, Ariyan S, Cheng E, Bacchiocchi A, et al. Incidence of the V600K mutation among melanoma patients with BRAF mutations, and potential therapeutic response to the specific BRAF inhibitor PLX4032. J Transl Med. 2010 Jul;8(1):67.

12. Flaherty KT, Puzanov I, Kim KB, Ribas A, McArthur GA, Sosman JA, et al. Inhibition of Mutated, Activated BRAF in Metastatic Melanoma. N Engl J Med. 2010 Aug;363(9):809–19.

13. Marrone M, Filipski KK, Gillanders EM, Schully SD, Freedman AN. Multi-marker Solid Tumor Panels Using Next-generation Sequencing to Direct Molecularly Targeted Therapies. PLoS Curr. 2014 May;6.

14. Martincorena I, Campbell PJ. Somatic mutation in cancer and normal cells. Science. 2015 Sep;349(6255):1483–9.

15. Davies H, Glodzik D, Morganella S, Yates LR, Staaf J, Zou X, et al. HRDetect is a predictor of BRCA1 and BRCA2 deficiency based on mutational signatures. Nat Med. 2017 Apr;23(4):517–25.

16. Bodily WR, Shirts BH, Walsh T, Gulsuner S, King M-C, Parker A, et al. Effects of germline and somatic events in candidate BRCA-like genes on breast-tumor signatures. PLoS One. 2020;15(9):e0239197.

17. Jaenisch R, Bird A. Epigenetic regulation of gene expression: How the genome integrates intrinsic and environmental signals. Nat Genet. 2003 Mar;33 Suppl(march):245–54.

18. Gomella LG, Liu XS, Trabulsi EJ, Kelly WK, Myers R, Showalter T, et al. Screening for prostate cancer: The current evidence and guidelines controversy. Can J Urol. 2011 Oct;18(5):5875–83.

19. Tyers M, Mann M. From genomics to proteomics. Nature. 2003 Mar;422(6928):193–7.

20. Hanash S. Disease proteomics. Nature. 2003 Mar;422(6928):226–32.

730 21. Borrebaeck CAK. Precision diagnostics: Moving towards protein biomarker signatures of clinical
731 utility in cancer. Nat Rev Cancer. 2017 Mar;17(3):199–204.
732 22. Schubert OT, Röst HL, Collins BC, Rosenberger G, Aebersold R. Quantitative proteomics:
733 Challenges and opportunities in basic and applied research. Nat Protoc. 2017 Jul;12(7):1289–94.
734 23. Butte A. The use and analysis of microarray data. Nat Rev Drug Discov. 2002 Dec;1(12):951–60.
735 24. Mortazavi A, Williams BA, McCue K, Schaeffer L, Wold B. Mapping and quantifying
736 mammalian transcriptomes by RNA-Seq. Nat Methods. 2008 Jul;5(7):621–8.
737 25. Parker JS, Mullins M, Cheang MCU, Leung S, Voduc D, Vickery T, et al. Supervised Risk
738 Predictor of Breast Cancer Based on Intrinsic Subtypes. JCO. 2009 Feb;27(8):1160–7.
739 26. Wallden B, Storhoff J, Nielsen T, Dowidar N, Schaper C, Ferree S, et al. Development and
740 verification of the PAM50-based Prosigna breast cancer gene signature assay. BMC Med Genomics. 2015
741 Aug;8.
742 27. Gnant M, Filipits M, Greil R, Stoeger H, Rudas M, Bago-Horvath Z, et al. Predicting distant
743 recurrence in receptor-positive breast cancer patients with limited clinicopathological risk: Using the
744 PAM50 Risk of Recurrence score in 1478 postmenopausal patients of the ABCSG-8 trial treated with
745 adjuvant endocrine therapy alone. Ann Oncol. 2014 Feb;25(2):339–45.
746 28. Dowsett M, Sestak I, Lopez-knowles E, Sidhu K, Dunbier A, Cowens J, et al. Comparison of
747 PAM50 Risk of Recurrence Score With Oncotype DX and IHC4 for Predicting Risk of Distant
748 Recurrence After Endocrine Therapy. Journal of clinical oncology : official journal of the American
749 Society of Clinical Oncology. 2013 Jul;31.
750 29. Nielsen T, Wallden B, Schaper C, Ferree S, Liu S, Gao D, et al. Analytical validation of the
751 PAM50-based Prosigna Breast Cancer Prognostic Gene Signature Assay and nCounter Analysis System
752 using formalin-fixed paraffin-embedded breast tumor specimens. BMC Cancer. 2014 Mar;14(1):177.
753 30. Tofigh A, Suderman M, Paquet ER, Livingstone J, Bertos N, Saleh SM, et al. The Prognostic
754 Ease and Difficulty of Invasive Breast Carcinoma. Cell Reports. 2014 Oct;9(1):129–42.

755 31. Stone M. Cross-validatory choice and assessment of statistical predictions. J R Stat Soc Ser B
756 Methodol. 1974;36(2):111–33.
757 32. Dudoit S, Fridlyand J. Classification in microarray experiments. In: Speed T, editor. Statistical
758 Analysis of Gene Expression Microarray Data. Chapman and Hall/CRC; 2003.
759 33. Fielden MR, Zacharewski TR. Challenges and Limitations of Gene Expression Profiling in
760 Mechanistic and Predictive Toxicology. Toxicol Sci. 2001 Mar;60(1):6–10.
761 34. Eling N, Morgan MD, Marioni JC. Challenges in measuring and understanding biological noise.
762 Nat Rev Genet. 2019 Sep;20(9):536–48.
763 35. Fernández-Delgado M, Cernadas E, Barro S, Amorim D. Do we Need Hundreds of Classifiers to
764 Solve Real World Classification Problems? J Mach Learn Res. 2014;15:3133–81.
765 36. Bay SD, Kibler D, Pazzani MJ, Smyth P. The UCI KDD archive of large data sets for data mining
766 research and experimentation. ACM SIGKDD Explor Newsl. 2000;2(2):81–5.
767 37. Domingos P. A Few Useful Things to Know about Machine Learning. :9.
768 38. Statnikov A, Wang L, Aliferis CF. A comprehensive comparison of random forests and support
769 vector machines for microarray-based cancer classification. BMC Bioinformatics. 2008;9(1):319.
770 39. Golub T, Slonim D, Tamayo P, Huard C, Gaasenbeek M, Mesirov J, et al. Molecular
771 classification of cancer: Class discovery and class prediction by gene expression monitoring. Science.
772 1999 Oct;286(5439):531–7.
773 40. Alizadeh AA, Eisen MB, Davis RE, Ma C, Lossos IS, Rosenwald A, et al. Distinct types of
774 diffuse large B-cell lymphoma identified by gene expression profiling. Nature. 2000 Feb;403(6769):503–
775 11.
776 41. Sørlie T, Perou CM, Tibshirani R, Aas T, Geisler S, Johnsen H, et al. Gene expression patterns of
777 breast carcinomas distinguish tumor subclasses with clinical implications. Proc Natl Acad Sci U S A.
778 2001 Sep;98(19):10869–74.
779 42. van ’t Veer LJ, Dai H, van de Vijver MJ, He YD, Hart AAM, Mao M, et al. Gene expression
780 profiling predicts clinical outcome of breast cancer. Nature. 2002 Jan;415(6871):530–6.

781 43. Ramaswamy S, Ross KN, Lander ES, Golub TR. A molecular signature of metastasis in primary
782 solid tumors. Nat Genet. 2003 Jan;33(1):49–54.
783 44. Cho S-B, Won H-H. Machine learning in DNA microarray analysis for cancer classification. In:
784 Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003-Volume 19.
785 2003. p. 189–98.
786 45. Pochet N, De Smet F, Suykens JA, De Moor BL. Systematic benchmarking of microarray data
787 classification: Assessing the role of non-linearity and dimensionality reduction. Bioinformatics.
788 2004;20(17):3185–95.
789 46. Lee JW, Lee JB, Park M, Song SH. An extensive comparison of recent classification tools
790 applied to microarray data. Comput Stat Data Anal. 2005;48(4):869–85.
791 47. Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat.
792 1992;46(3):175–85.
793 48. Fisher RA. The use of multiple measurements in taxonomic problems. Ann Eugen.
794 1936;7(2):179–88.
795 49. Rosenblatt F. Principles of neurodynamics. Perceptrons and the theory of brain mechanisms.
796 Cornell Aeronautical Lab Inc Buffalo NY; 1961.
797 50. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97.
798 51. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
799 52. Statnikov A, Aliferis CF, Tsamardinos I, Hardin D, Levy S. A comprehensive evaluation of
800 multicategory classification methods for microarray gene expression cancer diagnosis. Bioinformatics.
801 2005 Mar;21(5):631–43.
802 53. Pirooznia M, Yang JY, Yang MQ, Deng Y. A comparative study of different machine learning
803 methods on microarray gene expression data. BMC Genomics. 2008;9 Suppl 1:S13.
804 54. Díaz-Uriarte R, Alvarez de Andrés S. Gene selection and classification of microarray data using
805 random forest. BMC Bioinformatics. 2006 Jan;7:3.

806 55. Koohy H. The rise and fall of machine learning methods in biomedical research. F1000Research.
807 2018 Jan;6:2012.
808 56. Jarchum I, Jones S. DREAMing of benchmarks. Nat Biotechnol. 2015 Jan;33(1):49–50.
809 57. Saez-Rodriguez J, Costello JC, Friend SH, Kellen MR, Mangravite L, Meyer P, et al.
810 Crowdsourcing biomedical research: Leveraging communities as innovation engines. Nat Rev Genet.
811 2016;17(8):470.
812 58. Sumsion GR, Bradshaw MS, Beales JT, Ford E, Caryotakis GRG, Garrett DJ, et al. Diverse
813 approaches to predicting drug-induced liver injury using gene-expression profiles. Biol Direct. 2020
814 Jan;15(1):1.
815 59. Chang C-C, Lin C-J. LIBSVM: A library for support vector machines. ACM Trans Intell Syst
816 Technol TIST. 2011;2(3):1–27.
817 60. Salas S, Brulard C, Terrier P, Ranchere-Vince D, Neuville A, Guillou L, et al. Gene Expression
818 Profiling of Desmoid Tumors by cDNA Microarrays and Correlation with Progression-Free Survival.
819 Clin Cancer Res. 2015 Sep;21(18):4194–200.
820 61. Ho YC, Pepyne DL. Simple Explanation of the No-Free-Lunch Theorem and Its Implications.
821 Journal of Optimization Theory and Applications. 2002 Dec;115(3):549–70.
822 62. Ching T, Himmelstein DS, Beaulieu-Jones BK, Kalinin AA, Do BT, Way GP, et al.
823 Opportunities and obstacles for deep learning in biology and medicine. Journal of The Royal Society
824 Interface. 2018 Apr;15(141):20170387.
825 63. Ho TK, Hull JJ, Srihari SN. Decision combination in multiple classifier systems. IEEE Trans
826 Pattern Anal Mach Intell. 1994;16(1):66–75.
827 64. López-García G, Jerez JM, Franco L, Veredas FJ. Transfer learning with convolutional neural
828 networks for cancer survival prediction using gene-expression data. PLOS ONE. 2020
829 Mar;15(3):e0230536.
830 65. Golightly NP, Bell A, Bischoff AI, Hollingsworth PD, Piccolo SR. Curated compendium of
831 human transcriptional biomarker data. Sci Data. 2018 Apr;5:180066.

66. Barrett T, Troup DB, Wilhite SE, Ledoux P, Evangelista C, Kim IF, et al. NCBI GEO: Archive for functional genomics data sets years on. Nucleic Acids Res. 2011 Jan;39(Database issue):D1005–10.

67. Piccolo SR, Sun Y, Campbell JD, Lenburg ME, Bild AH, Johnson WE. A single-sample microarray normalization method to facilitate personalized-medicine workflows. Genomics. 2012;100(6):337–44.

68. Dai M, Wang P, Boyd AD, Kostov G, Athey B, Jones EG, et al. Evolving gene/transcript definitions significantly alter the interpretation of GeneChip data. Nucleic Acids Res. 2005 Jan;33(20):e175.

69. Rosikiewicz M, Robinson-Rechavi M. IQRray, a new method for Affymetrix microarray quality control, and the homologous organ conservation score, a new benchmark method for quality control metrics. Bioinformatics. 2014;30(10):1392–9.

70. Waldron L, Riester M, Ramos M, Parmigiani G, Birrer M. The Doppelgänger effect: Hidden duplicates in databases of transcriptome profiles. JNCI J Natl Cancer Inst. 2016;108(11).

71. Johnson WE, Li C, Rabinovic A. Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics. 2007;8(1):118–27.

72. The Cancer Genome Atlas Research Network. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. Nature. 2008 Oct;455(7216):1061–8.

73. Liao Y, Smyth GK, Shi W. The Subread aligner: Fast, accurate and scalable read mapping by seed-and-vote. Nucleic Acids Res. 2013 May;41(10):e108.

74. Liao Y, Smyth GK, Shi W. FeatureCounts: An efficient general purpose program for assigning sequence reads to genomic features. Bioinformatics. 2014;30(7):923–30.

75. Rahman M, Jackson LK, Johnson WE, Li DY, Bild AH, Piccolo SR. Alternative preprocessing of RNA-Sequencing data in The Cancer Genome Atlas leads to improved analysis results. Bioinformatics. 2015 Nov;31(22):3666–72.

76. Yates AD, Achuthan P, Akanni W, Allen J, Allen J, Alvarez-Jarreta J, et al. Ensembl 2020. Nucleic Acids Research. 2020 Jan;48(D1):D682–8.

858 77. Piccolo SR, Frampton MB. Tools and techniques for computational reproducibility. Gigascience.
859 2016 Jul;5(1):30.
860 78. Piccolo SR, Lee TJ, Suh E, Hill K. ShinyLearner: A containerized benchmarking tool for
861 machine-learning classification of tabular data. Gigascience. 2020 Apr;9(4).
862 79. Bischl B, Lang M, Kotthoff L, Schiffner J, Richter J, Studerus E, et al. Mlr: Machine learning in
863 r. J Mach Learn Res. 2016;17(1):5938–42.
864 80. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn:
865 Machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
866 81. Hall M, National H, Frank E, Holmes G, Pfahringer B, Reutemann P, et al. The WEKA data
867 mining software. ACM SIGKDD Explor Newsl. 2009 Nov;11(1):10.
868 82. Sculley D, Snoek J, Wiltschko A, Rahimi A. Winner's Curse? On Pace, Progress, and Empirical
869 Rigor. 2018 Feb;
870 83. Van Rossum G, others. Python Programming Language. In: USENIX Annual Technical
871 Conference. 2007. p. 36.
872 84. R Core Team. R: A language and environment for statistical computing. Vienna, Austria: R
873 Foundation for Statistical Computing; 2020.
874 85. Wei T, Simko V. R package "corrplot": Visualization of a correlation matrix. 2017.
875 86. Wilke CO. Cowplot: Streamlined Plot Theme and Plot Annotations for 'Ggplot2'. 2017.
876 87. Slowikowski K. Ggrepel: Automatically Position Non-Overlapping Text Labels with 'Ggplot2'.
877 2018.
878 88. Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, et al. Welcome to the
879 tidyverse. J Open Source Softw. 2019;4(43):1686.
880 89. Tange O. GNU Parallel - The Command-Line Power Tool. Login USENIX Mag. 2011
881 Feb;36(1):42–7.
882 90. Green DM, Swets JA, others. Signal detection theory and psychophysics. Vol. 1. Wiley New
883 York; 1966.

884 91. Brier GW. Verification of forecasts expressed in terms of probability. Mon Wea Rev. 1950
885 Jan;78(1):1–3.
886 92. Vickery BC. Techniques of Information Retrieval. London: Butterworths; 1970.
887 93. Matthews BW. Comparison of the predicted and observed secondary structure of T4 phage
888 lysozyme. Biochim Biophys Acta BBA-Protein Struct. 1975;405(2):442–51.
889 94. Brand A, Allen L, Altman M, Hlava M, Scott J. Beyond authorship: Attribution, contribution,
890 collaboration, and credit. Learn Publ. 2015;28(2):151–5.
891 95. Strobl C, Boulesteix A-L, Zeileis A, Hothorn T. Bias in random forest variable importance
892 measures: Illustrations, sources and a solution. BMC Bioinformatics. 2007 Jan;8(1):25.
893 96. Strobl C, Boulesteix A-L, Kneib T, Augustin T, Zeileis A. Conditional variable importance for
894 random forests. BMC Bioinformatics. 2008 Jul;9(1):307.
895 97. Kruskal WH, W. Allen Wallis. Use of ranks in one-criterion variance analysis. J Am Stat Assoc.
896 1952;47(260):583–621.
897 98. Ishwaran H, Kogalur UB, Kogalur MUB. Package 'randomForestSRC.' 2020;
898 99. Ishwaran H, Kogalur UB, Blackstone EH, Lauer MS, others. Random survival forests. Ann Appl
899 Stat. 2008;2(3):841–60.
900 100. Shannon CE. A mathematical theory of communication. Bell Syst Tech J. 1948;27(3):379–423.
901 101. Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support
902 vector machines. Mach Learn. 2002;46(1-3):389–422.
903 102. Pearson K. In: Proceedings of the Royal Society of London. Taylor & Francis; 1895. p. 240–2.
904 103. Quinlan JR. Induction of decision trees. Mach Learn. 1986;1(1):81–106.
905 104. Holte RC. Very simple classification rules perform well on most commonly used datasets. Mach
906 Learn. 1993;11:63–91.
907 105. Kononenko I. Estimating attributes: Analysis and extensions of RELIEF. In: Bergadano F, Raedt
908 LD, editors. European conference on machine learning. Springer; 1994. p. 171–82.

106. Witten IH, Frank E. Data mining: Practical machine learning tools and techniques with Java implementations. Acm Sigmod Rec. 2002;31(1):76–7.

107. Kuhn M, Quinlan R. C50: C5.0 decision trees and rule-based models. 2020.

108. Hothorn T, Hornik K, Zeileis A. Unbiased recursive partitioning: A conditional inference framework. J Comput Graph Stat. 2006;15(3):651–74.

109. Hastie SMilborrowD from mda:mars by T, wrapper. RTibshiraniUAMF utilities with TL leaps. Earth: Multivariate adaptive regression splines. 2020.

110. Karatzoglou A, Smola A, Hornik K, Zeileis A. Kernlab an S4 package for kernel methods in R. J Stat Softw. 2004;11(9):1–20.

111. Friedman J, Hastie T, Tibshirani R. Regularization paths for generalized linear models via coordinate descent. J Stat Softw. 2010;33(1):1.

112. LeDell E, Gill N, Aiello S, Fu A, Candel A, Click C, et al. H2o: R interface for the 'H2O' scalable machine learning platform. 2020.

113. Bengio Y. Learning deep architectures for AI. Now Publishers Inc; 2009.

114. Natekin A, Knoll A. Gradient boosting machines, a tutorial. Front Neurorobotics. 2013;7:21.

115. Schliep K, Hechenbichler K. Kknn: Weighted k-Nearest neighbors. 2016.

116. Bergmeir C, Benítez JM. Neural networks in R using the stuttgart neural network simulator: RSNNS. J Stat Softw. 2012;46(7):1–26.

117. Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F. E1071: Misc functions of the department of statistics, probability theory group (formerly: E1071), TU wien. 2019.

118. Liaw A, Wiener M. Classification and regression by randomForest. R News. 2002;2(3):18–22.

119. Wright MN, Ziegler A. ranger: A fast implementation of random forests for high dimensional data in C++ and R. J Stat Softw. 2017;77(1):1–7.

120. Therneau T, Atkinson B. Rpart: Recursive partitioning and regression trees. 2019.

121. Therneau TM, Atkinson EJ, others. An introduction to recursive partitioning using the RPART routines. Technical report Mayo Foundation; 1997.

935 122. Deng H, Runger G. Gene selection with guided regularized random forest. Pattern Recognit.
936 2013;46(12):3483–9.
937 123. Ahdesmaki M, Zuber V, Gibb S, Strimmer K. Sda: Shrinkage discriminant analysis and CAT
938 score variable selection. 2015.
939 124. Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, et al. Xgboost: Extreme gradient
940 boosting. 2020.
941 125. Freund Y, Schapire R, Abe N. A short introduction to boosting. J-Jpn Soc Artif Intell.
942 1999;14(771-780):1612.
943 126. Berkson J. Application of the logistic function to bio-assay. J Am Stat Assoc. 1944;39(227):357–
944 65.
945 127. Saad D. Online algorithms and stochastic approximations. Online Learn. 1998;5:6–3.
946 128. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–40.
947 129. Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. Mach Learn. 1997;29(2-
948 3):131–63.
949 130. Kohavi R. The power of decision tables. In: 8th european conference on machine learning.
950 Springer; 1995. p. 174–89.
951 131. Hulten G, Spencer L, Domingos P. Mining time-changing data streams. In: ACM SIGKDD intl
952 Conf On knowledge discovery and data mining. ACM Press; 2001. p. 97–106.
953 132. Quinlan R. C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann Publishers;
954 1993.
955 133. Cohen WW. Fast effective rule induction. In: Twelfth international conference on machine
956 learning. Morgan Kaufmann; 1995. p. 115–23.
957 134. Fan R-E, Chang K-W, Hsieh C-J, Wang X-R, Lin C-J. LIBLINEAR - a library for large linear
958 classification. 2008;
959 135. John GH, Langley P. Estimating continuous distributions in bayesian classifiers. In: Eleventh
960 conference on uncertainty in artificial intelligence. San Mateo: Morgan Kaufmann; 1995. p. 338–45.

961 136. Landwehr N, Hall M, Frank E. Logistic model trees. Machine learning. 2005;95(1-2):161–205.

962 137. Sumner M, Frank E, Hall M. Speeding up logistic model tree induction. In: 9th european

963 conference on principles and practice of knowledge discovery in databases. Springer; 2005. p. 675–83.

964 138. Platt J. Fast training of support vector machines using sequential minimal optimization. In:

965 Schoelkopf B, Burges C, Smola A, editors. Advances in kernel methods - support vector learning. MIT

966 Press; 1998.

967 139. Keerthi SS, Shevade SK, Bhattacharyya C, Murthy KRK. Improvements to platt's SMO

968 algorithm for SVM classifier design. Neural Comput. 2001;13(3):637–49.

969 140. Hastie T, Tibshirani R. Classification by pairwise coupling. In: Jordan MI, Kearns MJ, Solla SA,

970 editors. Advances in neural information processing systems. MIT Press; 1998.

971 141. Demiroz G, Guvenir A. Classification by voting feature intervals. In: 9th european conference on

972 machine learning. Springer; 1997. p. 85–92.