An Exploration of Analysis Methods on Predictive Models of Student Success

Alex Beckwith

University of New Hampshire

May 2023

Contents

Abstract

Machine learning models are not always evaluated with statistical rigor. This can lead to inferential flaws when assumptions are made about the underlying and performance data, especially when cross-validation is used. In this paper, a Bayesian method of model evaluation is compared to a non-parametric frequentist method. In addition, a metric for analyzing the fairness of a particular algorithm is tested.

The evaluation techniques were applied to a dataset of student and course data made available by the Open University. A system was built to train and test predictive models of student success. The aim was to predict students at risk of failing or withdrawing from a course using the first 30 days of data extracted from the virtual learning environment. In an applied setting, these predictions could be used to direct additional resources to at-risk students.

The project included creating a database to cleanse, transform, and analyze the dataset. Features were engineered to use as predictive inputs using a combination of exploratory analysis and inspiration from research. Four different subsets of input features were applied to nine different classification algorithms. Both randomized and exhaustive hyperparameter tuning procedures were experimented with, which created hundreds of distinct hyperparameter settings.

The Bayesian strategy provided more conclusive results by determining a "region of practical equivalence" as opposed to an inability to reject the null hypothesis. The results were similar to findings from research, which typically found tree-based ensemble methods in the upper-equivalence region.

The proposed metric for predictive fairness is called the Absolute Between Receiver Operating Characteristic Area (ABROCA). This metric was first introduced at the 2019 International Learning Analytics and Knowledge Conference. A significant relationship between ABROCA and the gender ratio of a course as well as between ABROCA and the ratio of students in a course identifying as having a disability. No significant relationship was found between ABROCA and overall model performance.

1 Introduction

When creating a predictive model to solve a problem, many different formulations and algorithms are typically tested. Datasets can be vast and complex, so the "see what sticks" strategy is often a good first step.

Selecting the approriate model to deploy should involve the consideration of many factors. Predictive performance is important, but simply picking the model with the highest mean accuracy is not a valid strategy. Statistical evaluation enables confident decisionmaking, which frees the researcher to address other considerations.

In this project, I applied the best practices from my research on model evaluation to a publicly-available dataset provided by an educational institution. I created a system that generates models that attempt to predict students who are likely to fail or drop out of a course. The practical application of the final model's predictions would be to target vulnerable students with interventions to increase their likelihood of course completion.

Using the results of the model testing, I show why a Bayesian approach is better for the evaluation of predictive models. In addition, I implement a new metric for testing model fairness. I affirm the results of the researchers who proposed the metric, showing that it is strongly correlated with a decrease in predictive fairness while showing scant correlation with predictive performance.

1.1 Research Questions

1. Which models and featuresets are best at predicting student outcomes?
2. How do the results differ when evaluated using naive, frequentist, and Bayesian procedures?
3. Is there an association between model predictive performance and Absolute Between Receiver Operating Characteristic Area (ABROCA)?

2 Previous Research

2.1 Learning Analytics

This project falls squarely within the interdiciplinary field of Educational Data Science (EDS), which itself is the union of Learning Analytics (LA) and Educational Data Mining (EDM). EDM is primarily concerned with the development of methods for exploring the unique types of data that come from education environments [**?**]. LA is focused on the analysis of learners and the optimization of the learning process [**?**]. The first EDM and LA-specific conferences took place in 2008 and 2011, respectively. All three have roots in statistics, education, and computer science, and all three use data to address important educational questions and issues. For a more comprehensive review of these fields, please refer to [**?**, **?**, **?**].

2.2 Modeling Student Performance

2.2.1 Predictions

One of the most common topics in LA and EDM is the creation of machine learning models to predict student behavior and performance. Researchers most often frame problems in the context of classification and regression problems [**?**, **?**, **?**]. Within the subset of research on classification and regression models, the most common features to predict were outcomes and scores [**?**]. In the context of a course, that might be whether the student passes or their final grade. One could also consider the context of a single problem, such as the student's most likely answer or the probability of being correct/incorrect [**?**]. The most effective algorithms for classification problems of these type in my research were tree-based ensemble methods.

2.2.2 Model Types

In my research, the models which best predicted student performance tended to be tree-based ensemble methods [**?**, **?**, **?**]. Tree-based methods were also the most commonly seen type of model, followed by linear and logistic regression [**?**]. Other common models include support vector machines, k-nearest neighbors, and naive Bayes. Research related to the utilization of deep learning is not uncommon [**?**, **?**], though I wasn't able to find a resource which utilized a neural network that significantly outperformed simpler methods.

2.3 Student Data

2.3.1 Sources

The most common sources of student data came from Massively Open Online Courses (MOOCs), Intelligent Tutoring Systems (ITS) and other computer-based platforms, though a great deal of research has also been conducted using data from traditional classrooms [**?**, **?**, **?**, **?**]. Especially since the pandemic of the early 2020s, many students have experienced educational systems which blend the two. Blended sources of learning data would benefit from additional research [**?**, **?**].

2.3.2 Featuresets

The most common featuresets for predicting student outcomes are are academic, demographic, and behavioral [**?**, **?**, **?**, **?**, **?**]. Academic includes information about assessments and assignments [**?**]. Behavioral data is often the logs from student interactions with the Virtual Learning Environment (VLE) [**?**, **?**]. Some research also includes biometric and visual data, such as a student's heartbeat and facial expression as they work through a problem [**?**].

2.3.3 Feature Extraction Strategies

I found three categories of sources that were utilized in research, all with distinct benefits and compromises [**?**, **?**]. Some researches utilize an automated system to extract features.

In [**?**], two automated feature-engineering systems were compared against a set of results produced by experts in EDM. One of the systems outperformed the other two sets, but at the risk of being significantly less interpertable. In many cases, the ability to explain why a particular feature is predictive of a result is paramount to producing actionable conclusions from the experiment. Another source suggested that the crowdsourcing of features in addition to exploration and expert recommendation produced useful results [**?**].

2.4 Model Evaluation Methods

2.4.1 Naive Averaging

Naive averaging is the evaluation tactic by which models are evaluated by ranking by the average of the target metric between runs an that alone. This strategy is both common and problemtatic because it doesn't take into account the magnitude and variability of differences in performance [**?**].

2.4.2 Frequentist

Null-hypothesis significance testing (NHST) is a common tool for differentiating the results of experiments. While useful to many problems, evaluating predictive models using NHST does not produce satisfying, actionable result. The t-test is a popular test to compare groups of returned values. Because most machine learning models utilize cross validation, the t-test is invalidated. One typically wants to be able to conclusively say which model is better on a variety of metrics in order to deploy said model. The result of NHST is either the rejecting of the null hypothesis or an inability to reject the null hypothesis. Typically involves running a global test for grouped mean differences, followed by a pairwise comparison test pending the results of the global test. In my research, the recommended NHST strategy was to use non-parametric tests as they work without working on the assumption that the means are normally distributed. Therefore, the recommended global test was a Friedman test for the global test and a Nemenyi test for pairwise differences.

2.4.3 Bayesian

The recommended approach to compare classification models is Bayesian in nature [**?**, **?**, **?**]. Similar to the frequentist approach, pairwise differences are compared using a test. The primary difference between the output of these strategies is that the Bayesian test can produce what might be considered a "three-tailed" test. That is, there are three regions produced by the modelled probability density function (PDF). There's the probability that either model is

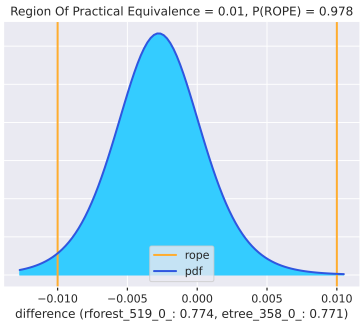

Figure 1: A probability density function resulting from a Bayesian Signed Rank Test

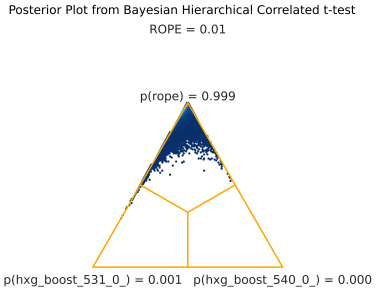

Figure 2: A Bayesian posterior plot resulting from a Bayesian hierarchical correlated t-test

better to a significantly discernable degree. In addition, there's the Region Of Practical Equivalence (ROPE), the probability that for all practical purposes, the two models produce equivalent results. The ROPE is specified as the difference in proportion between the two models being compared. For example, a ROPE value of 0.01 is assessing the probability that the mean of the two classifiers' results will be within 1 percent. eg for all practical purposes, they can be considered equivalent. Another benefit of the Bayesian method is its robustness to multiple comparisons and its accounting for cross-validation.

The package baycomp [**?**] is a Python package that implements these Bayesian tests. It uses hierarchical for multiple datasets, signed rank for single. The package appears to have been created and refined while this paper was being written [**?**]

2.4.4 ABROCA — Slicing Analysis

To understand ABROCA, it is necessary to understand the Receiver Operating Characteristic (ROC). The ROC is a function of the false positive rate to true positive rate over the range of threshold values for a predictor. originally used to measure performance of radar equipment For a more formal exploration of the mathematical properties of the ROC, see [**?**].

By integrating over the range of threshold values $[0, 1]$, one finds the area under ROC curve (ROC AUC), a commonly used as a metric to optimize performance of machine learning models. By maximizing the ROC AUC, one is maximizing the density of correct predictions while making the threshold value arbitrary. It follows that a perfect predictor would reach the theoretical maximum of ROC AUC = 1.0, having correct prediction at all threshold values. The lowest practical ROC AUC is that of a random predictor, ROC AUC = 0.5. In this case, the model is equally likely to pick correctly or incorrectly at all threshold values.

ABROCA is a metric used to analyze model fairness. It was first introduced at 2019 International Learning Analytics and Knowledge Conference [**?**]. To calculate ABROCA, split dataset by feature of interest and obtain the ROC for the model's predictions on each partion of the split dataset. Sum absolute values of between-curve area.

How does this relate to fairness? A model that produces equivalent prediction probabilities for the subgroups of the split dataset equally would have ABROCA = 0. Producing the same prediction probabilities wouldproduce the same ROC curves, so there would be no area to sum. Hypothesis - Higher ABROCA associated with lower predictive fairness [**?**] found significant differences in fairness when testing different models on a large collection of Massively Open Online Course (MOOC) datasets.

1. Significant Differences
   (a) Statistical Algorithm
   (b) Featureset Used
2. Relationship to ABROCA
   (a) Gender Imbalance Ratio
   (b) Curricular Area
3. Specific Course
   (a) Performance and Fairness
   (b) Not a strict tradeoff with performance and fairness

Module	Domain	Presentations	Students
AAA	Social Sciences	2	748
BBB	Social Sciences	4	7,909
CCC	STEM	2	4,434
DDD	STEM	4	6,272
EEE	STEM	3	2,934
FFF	STEM	4	7,762
GGG	Social Sciences	3	2,534

Figure 3: Module summary and domain information [**?**]

3 Experimental Architecture

3.1 Dataset

3.1.1 Background

The dataset I used in this experiment was made public by the Open University (OU), an online institution based in the United Kingdom. The OU provisions one of the largest publicly available LA datasets, known as the Open University Learning Analytics Dataset (OULAD) [**?**]. The data comes from 23 presentations (semesters) of 7 different Massive Open Online Courses (MOOCs) (called "modules" by the university) over 2 years and includes demographic, assessment, Virtual Learning Environment (VLE), and course information. VLE data is summarized by the number of clicks per course activity per student per day, which amounts to 10,655,280 records associated with 32,593 students. In order to be included in the OULAD, a module-presentation had to have over 500 students, at least two different presentations, and utilize the VLE. The module also had to have a significant amount of failing students.

3.1.2 Description of Tables

The Open University Learning Analytics Dataset (OULAD) consists of seven tables. One table contains student demographic information. A trio of tables describe the courses. There's a table containing the distinct module and presentation codes and a table each for the assessments and Virtual Learning Environment (VLE) activities for each course. The three remaining tables bridge the student demographic table with each of the course tables. One table contains the inforamiton about course registration. One summarizes students' interactions with individual VLE activities on a daily timescale. One contains records on students' assessment completion.

3.1.3 Comparison to 2015 Data

The publishers of the dataset performed an analysis on the distribution of demographic characteristics over time. Specifically, they compared a sample of student data from 2015 to the dataset distributions of module CCC, which are from 2013 and 2014. The authors performed chi-squared and Wilcox tests to evaluate the null hypothesis that the mean values of these distributions are equivalent. At typical significance levels, the null hypothesis cannot be rejected.

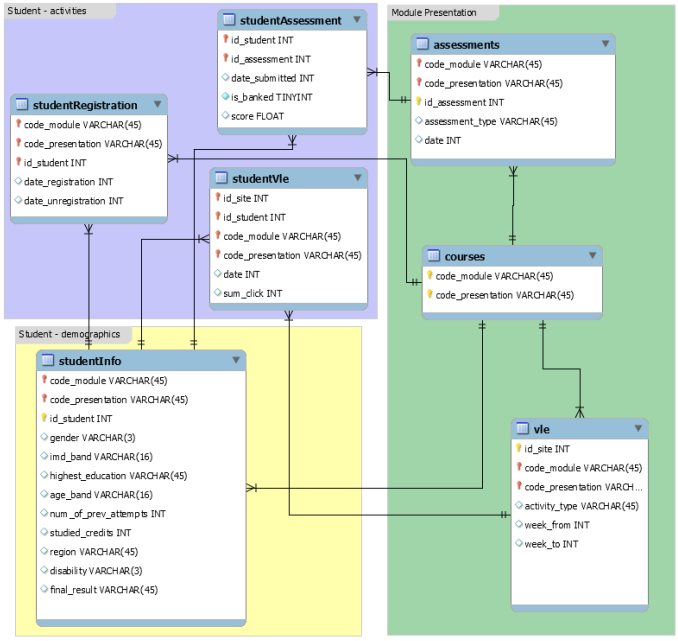

Figure 4: An entity-relationship diagram describing the published dataset [**?**].

Attribute	Test	Degrees of freedom	Test value	*P*-value
Age	Wilcox	—	17,606,000	0.3558
Disability	χ^2	1	1.0739	0.3001
Education	χ^2	4	0.89201	0.9237
Gender	χ^2	1	2.0537	0.1518
IMD	χ^2	19	15.912	0.6631
Region	χ^2	12	16.525	0.1768

Figure 5: Evaluation of similarity between OULAD and 2015 data for CCC module [**?**].

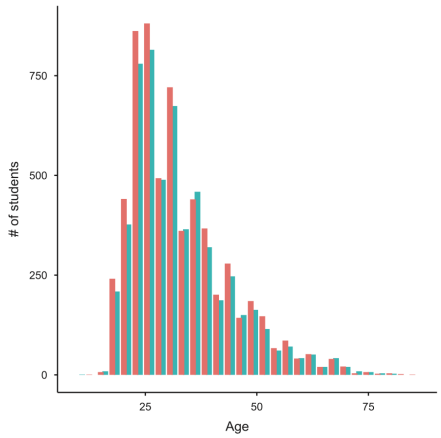

Figure 6: The distribution of student ages for OULAD data (blue) and 2015 data (red) for CCC module [**?**].

A difference between the data from the authors' analysis and the published dataset is the granularity of the information available. An anonymization tool was used to obfuscate identifying details in the data while still including specific student demographic information. To illustrate the impact on the data, please see figure xyz.

3.1.4 Other Student Characteristics

3.2 Feature Extraction

3.2.1 Database System

I chose to create a PostgreSQL relational database to organize the cleansing pipeline for this project [**?**]. PostgreSQL is an open source, performant, and reliable relational database system. The clear syntax of Structured Query Language (SQL) for scripting transformations and database operations enabled me to wrangle millions of rows efficiently.

3.2.2 Flat Files to Main

The first step in utilizing the database was injesting the data. I used the Python package pandas to load the Comma-Separated Values (CSV) files into a schema called [landing]. I then created a schema called [staging] and used SQL to correct and optimize the datatypes of the landing tables. I created a schema called [main] to hold an alternative structure of the data.
The categorical text fields from the staging tables were each given a table to store their distinct values with an integer id as a primary key. In the original tables, distinct courses required two text fields for identification. In the original

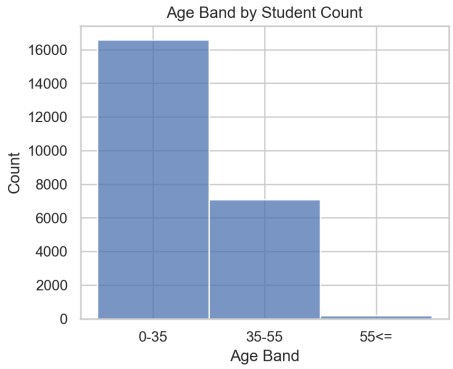

Figure 7: The distribution of published OULAD age groups [**?**].

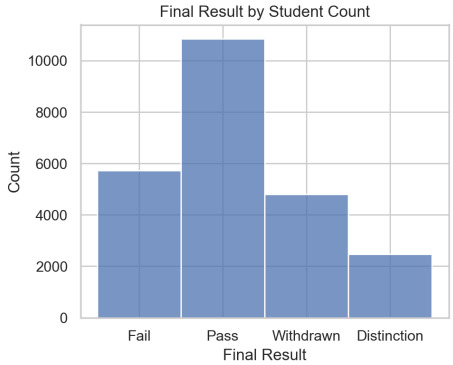

Figure 8: The distribution of published OULAD final module results. [**?**].

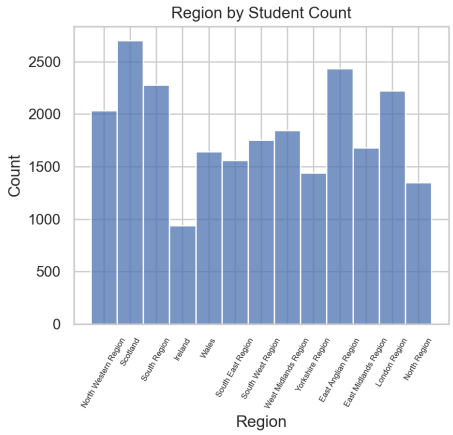

Figure 9: The distribution of published OULAD final module results. [**?**].

student table, the same two fields were necessary, with the addition of an integer student_id. During the process of transforming the tables into [main], I created a [course_id] field to identify each module/presentation combination with a single integer. I also created a [student_id] field to identify each individual student record with a distinct integer. Using a single integer field instead of multiple text fields to identify distinct records leads to significant efficiency improvements in storage, memory, and computation. While creating new [student_info] and [course_info] tables in the [main] schema, I used the new categorical tables to substitute the text values for integer ids. This was advantageous not only for the efficiency of utilizing the database, but also because some machine learning models only accept numeric values as input.

3.2.3 Main to First30

A great deal of research exists which utilizes predictive modeling to enhance the learning experience. This information could be used to intervene positively impact students' lives, increasing graduation rates. For my project, I decided to predict course outcome. This information is most useful when it can be predicted early, which is why I limited the data used for predictions to the first 30 days from the start of the course. I also excluded data relating to students who dropped out before the 30 day mark so that their lack of data for certain features wouldn't impact the training process.
I therefore decided to Difference in proportion from Main to First30 for Shown demographics Final Outcomes LAnding to Main Database schema decisions Main to First30 Database schema decisions

3.2.4 Feature Engineering

I built features to use as inputs for the predictive models with inspiration from research and by following intuition [**?**, **?**]. The breadth of feature extraction strategies in research spanned from subject-matter expertise to automation and crowdsources, all of which have their merits.

Some of the features were built in, requiring only the effort to collect to utilize, such as gender and region. The major categories of features fall into these four groups: student demographic information, course information, assessment data, and VLE interaction data. The assessment and interaction categories are where most of the "derived" features sit.

Here's an example of a feature in the VLE activity category that I created when following intuition:

Here's an example of a feature in the VLE activity category that I built based on on of the expert-created fields in this paper [**?**]:

3.3 Algorithms and Hyperparameters

Hyperparameters are the input parameters for the operation of a machine learning model. All of the algortihms and parameter selection components I used are

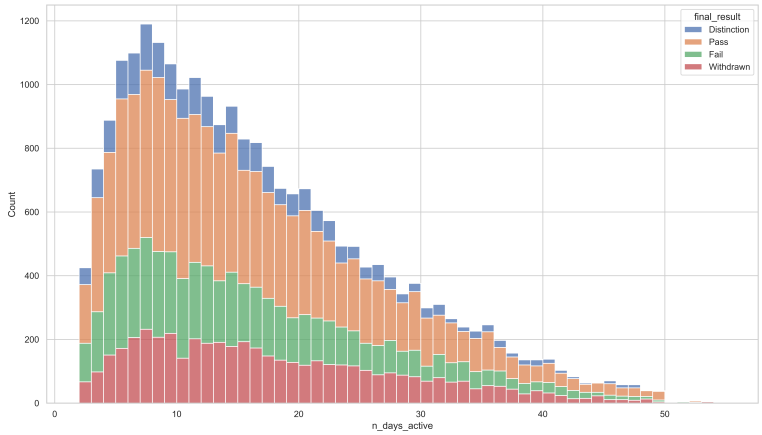

Figure 10: The number of distinct days that students interacted with the VLE in the first 30 days

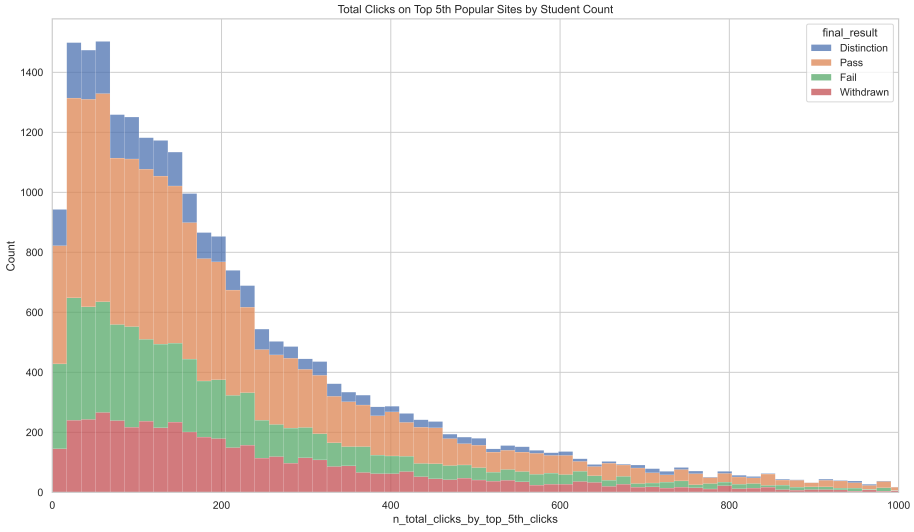

Figure 11: The total number of clicks on activities ranked in the top 5th percentile of activities as measured by student clicks

part the Python package scikit-learn [**?**]

3.3.1 Hyperparameter Selection

I utlized two different strategies for parameter selection: GridSearch and RandomSearch. GridSearch is a brute force strategy where one can specify collections of parameter selections and the cross-validating object will walk through them all. RandomSearch is similar to GridSearch in the way that parameters are mapped out, but selects random combinations from the available ranges. Interestingly, one can even set scipy [**?**] random variables as distributions for this cross-validation strategy to pick from during testing.

My logic for using both strategies was to first apply wide ranges of parameter combinations for the GridSearch strategy. I then used the results from runs to restrict the parameter ranges to those yielding better results. Better results in this case refers to higher average ROC AUC while also running in a reasonable timespan. If models were taking longer than 30 minutes to run, I moved the limits of the parameter selection ranges to avoid that when possible. The optimization algorithms used to train these models can sometimes not converge to a local (ideally global) minimum, meaning that the predictions will not be the ebst, which is why restricting the paramter ranges is necessary.

Once the paramter ranges were sufficiently restricted, I ran my model-generating pipeline using the RandomSearch strategy. This can be advantegous because certain combinations of parameter values will never be tried on a reasonable timescale while using a brute force strategy.

3.3.2 Algorithms

Here is a list of the classification algorithms used in this project along with abbreviations that will appear in later figures. If curious about a particular model or its available parameters, please see [**?**]. The parameterizations of the top two models will be included in the results section of this paper.

- Decision Tree
- Ada Boosting with decision tree as submodel
- Histogram Gradient Boosting
- Random Forest
- Extra Trees
- K-Nearest Neighbors
- Logistic Regression
- Multi-Layer Perceptron
- Support Vector Machines

3.4 Model Pipeline

The model generating, training, and testing pipeline was created using scikit-learn components [**?**] I created separate Python classes that worked together to handle the feature selection, data preprocessing, and parameter selection. A class I called ModelEngine orchestrated the process from generation to the logging of results.

3.4.1 Data Preprocessing

In the data preprocessing pipeline, features were transformed in order to improve their compatability with making predictions. Categorical data was encoded using the "one hot" strategy where each value was pivoted into a Boolean feature. All Boolean features were made numeric. Once all features are numeric, the pipeline standardizes each field so that the mean is zero and the variance is one. This improves the algorithm's predictive performance by not allowing any of the features to artificially outweigh any other.

If values were missing or null, they were imputed as zeros. I decided to impute them instead of exclude them in order to conserve as much data as possible for training.

Any fields with zero-variance are excluded in order to increase computational efficiency. With no variance, a field won't be predictive of any result, and will instead just add overhead. Another strategy implemented to increase efficiency is Principle Component Analysis (PCA) using a Maximum-Likelihood Estimator (MLE) strategy. Similar to the variance threshold, PCA works by reducing model dimensionality in orer to reduce model training time, though through a more complex process.

3.4.2 Model Training

Each model was crossvalidated using a repeated stratified k folds algorithm. Each run included five splits and two repeats, totalling ten runs for each model parameterization. During training, each model was refit by attempting to maximize ROC AUC. The feature that was predicted for the experiment was is_withdrawn_or_fail, simple binary classification. The training and testing proportions varied between 0.25 and 0.35.

4 Results Analysis

4.1 Naive Averaging

By simply averaging the results and sorting by performance, we'd miss key details. In the below figure, we can see that there is quite a degree of variability on key metrics like predictive performance and fit time.

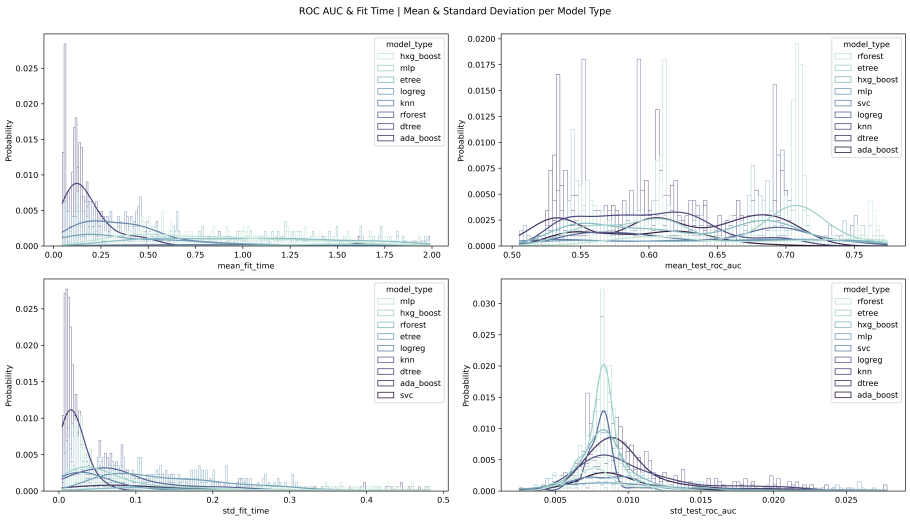

Figure 12: The mean and standard deviation of fit time and ROC AUC by model type

4.2 Null Hypothesis Significance Testing

For the frequentist model comparison, I used non-parametric tests to evaluate model differences. I wanted to make as few assumptions as possible about the distribution of the underlying data. Additionally, since cross-validation was used, we shouldn't use a ttest [**?**, **?**].

A Friedman test [**?**, **?**] to check for global differences returned a p-value of less than 10e-6, so I turned to the Nemenyi signed rank test [**?**, **?**] to look at pairwise differences. The results can be seen on the below figure, which illustrate one of the major pitfalls of utilizing frequentist methods for model evaluation. The wide band of gray shows where we have an inability to reject the null hypothesis at a significance level of 0.05 for the top 69 models. That the performance of the models is approximately equivalent is not particularly conclusive.

4.3 Bayesian Region of Practical Equivalence

When evaluating model performance using the Bayesian strategy, we find much more conclusive results. A Bayesian signed rank test [**?**, **?**, **?**] was used to evaluate each pairwise comparison using a ROPE of 0.002. I attempted to do all pairwise comparisons with full hierarchical model tests, but the computational overhead for the number of comparisons was too high.

The same top 100 results are shown, but with only two models in the uppermost ROPE, an Extra Tree and a Random Forest model. Even looking at the top two approximately-equivalent regions, the top six of seven are tree-based ensemble methods, similar to research. Surprisingly there ain't an Ada

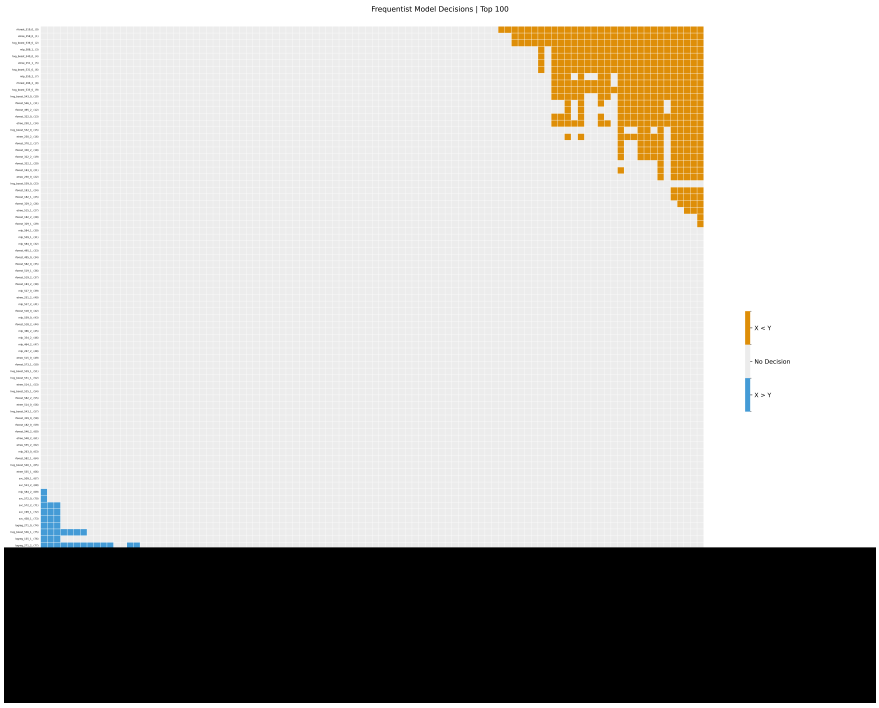

Figure 13: A windowpane plot showing the results of the non-parametric Nemenyi test

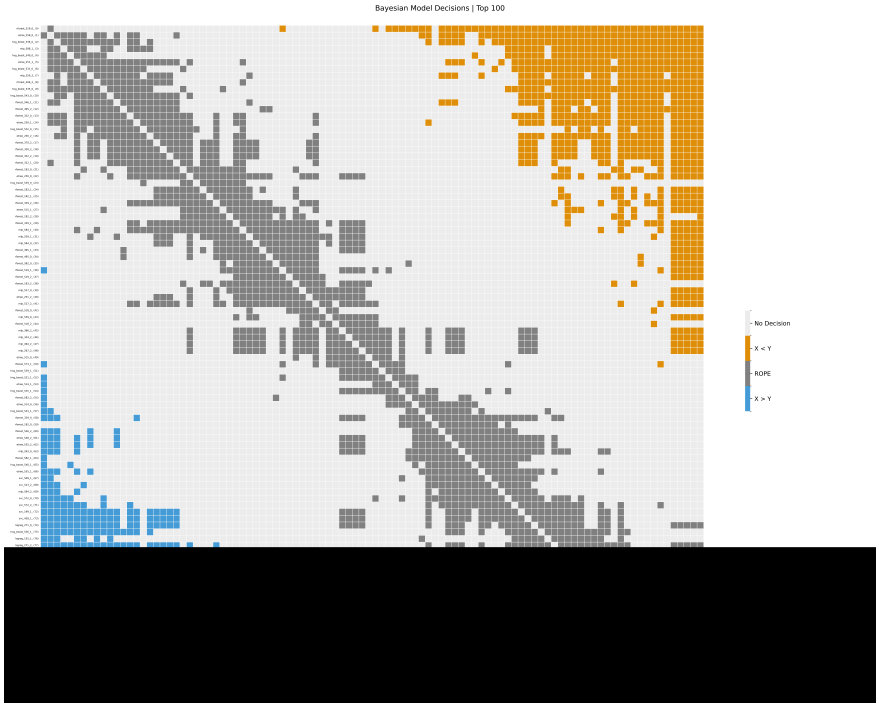

Figure 14: A windowpane plot showing the results of a Bayesian signed rank test using a ROPE of 0.002

Boostingup there.

4.4 Fairness Though Slicing Analysis

I also evaluated the ABROCA statistic and found similar results to [**?**].

First, we have a visualization of two models, each being split two ways. In both cases, the extra tree model appears to have better average predictive performance than the logistic regression. Both models have a similar ABROCA when it comes to predictions for disabled students, but the extra tree model appears to make predictions less fairly when it comes to gender.

The top four models happened to be different model types, so I retrained and evaluated the ABROCA for each model on each distict course's data. The points on each chart represent an individual model's ABROCA score for an individual course, differentiate by color for each course domain. The top two plots show two quadratic regressions that seem to indicate a strong correlation between ABROCA and both the course disabled ratio and course female ratio. The bottom two plots appear to show a rather weak correlation between the models' predictive performance and ABROCA.

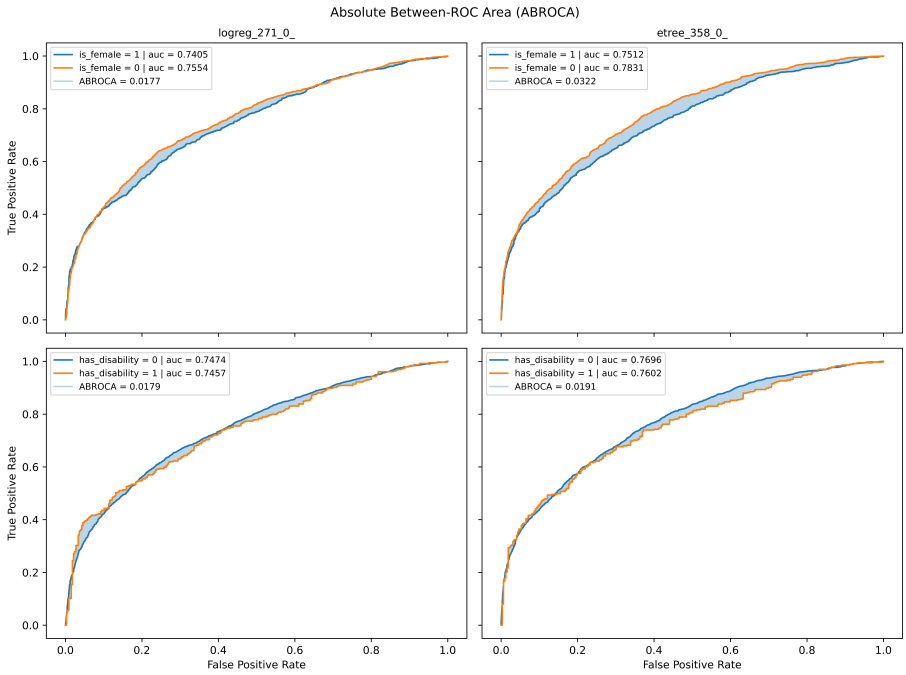

Figure 15: The ABROCA statistic for two models split by $\text{is}_f emale and has_d isability$

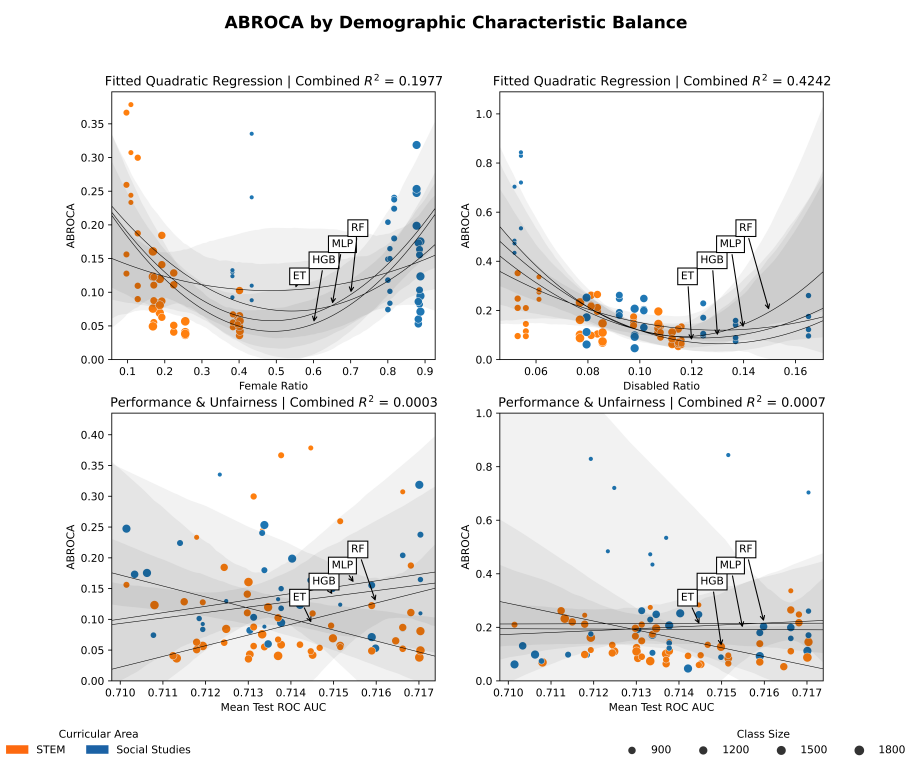

Figure 16: The relationship between ABROCA and demographic balance and mean test ROC AUC for the top four models

5 Conclusion

5.1 Best Models and Features

In this experiment, I found that tree-based ensemble methods tended to be the best-performing models on this dataset, with these features. The models which performed the best were those that used the entire set of features, not just a subset.

5.2 Best Evaluation Method

Similar to expectations, we found that Bayesian evaluation methods produced the most conclusive ansnswer to the "which model is better question" and suffers from the fewest assumptions or invalidations.

5.3 Fairness vs Performance

We did not see a strong relationship between model predictive performance (as measured by ROC AUC) and ABROCA. There did appear to be a strong relationhip between demographic ratio and ABROCA. This result seems to indicate the ABROCA may be a useful metric in the evaluation of model fairness. In addition, this usefulness does not appear to come at the cost of predictive performance.

5.4 Future Work

To extend this work, additional development of the model generation system could allow for automated hyperparameter distribution optimization. More time could be spent refining the feature database and extracting additional predictive features. Expend more computing resources on hierarchical comparisons rather than different parameterizations

A future researcher might evaluate the ABROCA statistic on additional datasets and different categorical splits. An attempt could be made to integrate it into the model selection process. One might explore if a higher-dimension expression of the calculation holds water (get it, volume). More of the mathetmatical properties of ABROCA could be explored.

References

[1] E. Aguiar and Et al. Engagement vs performance: Using electronic portfolios to predict first semester engineering student persistence. *Journal of Learning Analytics*, 1(3), 2014.

[2] A. Benavoli and Et al. Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. *Journal of Machine Learning Research*, 18(1), 2017.

[3] N. Bosch. Automl feature engineering for student modeling yields high accuracy, but limited interpretability. *Jornal of Educational Data Mining*, 13(2), 2021.

[4] C. Calì and M. Longobardi. Some mathematical properties of the roc curve and their applications. *Archivio della Ricerca*, 2014.

[5] W. Chango, R. Cerezo, M. Sanchez-Santillan, R. Azevedo, and C. Romero. Improving prediction of students' performance in intelligent tutoring systems using attribute selection and ensembles of different multimodal data sources. *Journal of Computing in Higher Education*, 1(33):614–634, 2021.

[6] W. Chen and Et al. Early detection prediction of learning outcomes in online short-courses via learning behaviors. *Transactions on Learning Technologies*, 12(1), 2019.

[7] G. Corani and Et al. Statistical comparison of classifiers through bayesian hierarchical modeling. *Machine Learning*, 1(106), 2017.

[8] J. Demsar, A. Benavoli, and G. Corani. Bayesian comparison of classifiers in Python. *Journal of Machine Learning Research*, 12, 2018.

[9] A. Esteban, C. Romero, and A. Zafra. Assignments as influential factor to improve the prediction of student performance in online courses. *Applied Sciences*, 11(21), 2021.

[10] J. Gardner and C. Brooks. Evaluating predictive models of student success: Closing the methodological gap. *Journal of Learning Analytics*, 5(2), 2018.

[11] J. Gardner, C. Brooks, and R. Baker. Evaluating the fairness of predictive student models through slicing analysis. Association for Computing Machinery, 2019.

[12] A. Gelman, J. Hill, and M. Yajima. Why we usually don't have to worry about multiple comparisons. *Journal of Research on Educational Effectiveness*, 5(2), 2009.

[13] The PostgreSQL global development team. Postgresql. 2023.

[14] J. Kuzilek, M. Hlosta, and Z. Zdrahal. Open university learning analytics dataset, 2017.

[15] S. Lowes, P. Lin, and B. Kinghorn. Exploring the link between online behaviours and course performance in asynchronous online high school courses. *Journal of Learning Analytics*, 2(2), 2015.

[16] J. López-Zambrano, J. Torralbo, and C. Romero. Early prediction of student learning performance through data mining: A systematic review. *Psicothema*, 33(3), 2021.

[17] L. Miller and Et al. A comparison of educational statistics and data mining approaches to identify characteristics that impact online learning. *Journal of Educational Data Mining*, 7(3), 2015.

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

[19] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33, 2007.

[20] C. Romero and S. Ventura. Educational data mining: A review of the state of the art. *Transactions on Systems: Man and Cybernetics*, 20, 2010.

[21] C. Romero and S. Ventura. Educational data mining and learning analytics: An updated survey. *Data Mining and Knowledge Discovery*, 10, 2019.

[22] K. Veeramachaneni, U. O'Reilly, and C. Taylor. Towards feature engineering at scale for data from massive open online courses. *1*, 1(1), 2014.

[23] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. Scipy 1.0: Fundamental algorithms for scientific computing in pythonn. *Nature Methods*, 17:261–272, 2020.

[24] I. Zualkernan. Exploring predicting performance of engineering students using deep learning. *International Conference on Cognition and Exploratory Learning in Digital Age*, 1(18), 2021.