**Statistical Models**

**Model Descriptions**

We used four statistical models to predict student outcomes. Summary information is below.

|  |  |  |
| --- | --- | --- |
| **Name** | **Algorithm** | **Number of Predictors** |
| GBM | Gradient Boosting Model | 10 |
| GBM Plus | Gradient Boosting Model | 48 |
| Mixed Model | Generalized Additive Mixed Model | 5 |
| Mixed Model Plus | Generalized Additive Mixed Model | 8 |

**Gradient Boosting Models.** GBM is a powerful machine learning technique that builds an ensemble of decision trees in a sequential manner, where each tree attempts to correct the errors of its predecessor. The model was implemented using the *h2o* library in R.

***Case weighting.*** To address the imbalance in our data across different institutions, where some institutions have a significantly higher amount of student data than others, we employed weighting techniques. Specifically, we applied inverse probability weighting while also implementing a trimming procedure to handle extreme values. Our goal was to ensure that the maximum weighting assigned to an institution was no more than 10 times higher than the lowest weighting assigned to any institution.

***Hyper-parameter optimization.*** Hyperparameter optimization is a crucial step in optimizing Gradient Boosting Models (GBMs). To ensure a comprehensive search for the best hyperparameters, we followed a multiphase optimization plan based on expert recommendations[[1]](#footnote-1). Our approach involved two stages of hyperparameter optimization.

In the first stage, we focused on tuning key parameters, starting with the maximum depth of decision trees. We performed a grid search, systematically exploring various depths ranging from 2 to 16. The performance of each model was evaluated, and the range of maximum tree depth associated with the top-performing models was identified. This range served as a guide for the subsequent stage of hyperparameter optimization.

In the second stage, we aimed to optimize a set of additional hyperparameters, including row sample rates per tree, column sample rates per split, column sampling rates per tree, and minimum rows per terminal node, among others. To efficiently explore this space, we utilized a random grid search approach, allowing a maximum of 1000 models to be evaluated. The model training was conducted with 10,000 trees and evaluated using 5-fold cross-validation.For further reference, detailed information and code regarding the hyperparameter optimization process can be found in the Appendix.

***Evaluating model performance.*** Our models were evaluated according to logloss, which measures how close a model’s predicted values (uncalibrated probability estimates) are to the actual target value[[2]](#footnote-2).

***Using ensemble models.*** To improve the performance of our gradient boosting algorithms, we employed ensemble learning techniques - specifically, the process of stacking or super learning. This approach amalgamates multiple models to refine the prediction accuracy, thereby surpassing the capabilities of a single model[[3]](#footnote-3).

For our implementation, we chose the Stacked Ensemble method provided by the H2O machine learning platform. This method is a supervised ensemble machine learning algorithm designed to find the optimal combination of a collection of prediction algorithms via a stacking process. We used the default settings provided by H2O.

Subsequently, we compared the predictive performance of the single best gradient boosting model against that of the ensemble models. This comparison was executed based on log loss, a typical measure for evaluating the accuracy of classifiers. Although ensemble models are generally expected to surpass individual models in terms of performance, there exist instances where they may lead to a degradation in performance.

***Predicting by institution*.** To predict individual student retention at various universities, we used an “individual conditional expectations” approach: we replace the student’s actual institution with other institutions, while keeping the student’s features (e.g. HS GPA) the same[[4]](#footnote-4).

**Generalized Additive Mixed Models.** Generalized Additive Mixed Models (GAMMs) are an extension of generalized linear models, possessing at least two advantages that are well suited to this research question. First, they allow for flexible modeling of nonlinear effects using smoothing functions such as splines, which can lead to greater predictive accuracy for numeric variables that may not have linear effects. Second, GAMMs are able to handle hierarchical or clustered data (e.g. students nested within institutions). They can capture the unobserved variability associated with these groupings, providing a more comprehensive and accurate analysis of hierarchical data structures.

***Variable selection***. We

***Evaluating model performance.*** Identical to our gradient boosting models, our GAMM models were evaluated according to logloss. We also provide additional performance metrics in Appendix.

**Appendix**

**Gradient Boosting Models (Parameter Tuning)**

|  |  |
| --- | --- |
| **Parameter/Setting** | **Value(s)** |
| Number of Folds | 5 |
| Number of Trees | 10,000 |
| Learning Rate | 0.05 |
| Learning Rate Annealing | 0.99 |
| Max Runtime (seconds) | 3600 |
| Stopping Rounds | 5 |
| Stopping Tolerance | 0.0001 |
| Stopping Metric | logloss |
| Score Tree Interval | 10 |
| Seed | 1234 |
| Max Depth | From 2 to 8 |
| Column Sample Rate | From 0.2 to 1, in 0.01 increments |
| Column Sample Rate Per Tree | From 0.2 to 1, in 0.01 increments |
| Column Sample Rate Change Per Level | From 0.9 to 1.1, in 0.01 increments |
| Min Rows in Terminal Node | 2^seq(0,log2(total rows)-1,1) |
| Number of Bins (Continuous & Integer) | 2^seq(4, 10, 1) |
| Number of Bins (Categorical) | 2^seq(4, 12, 1) |
| Min Split Improvement | 0, 1e-8, 1e-6, & 1e-4 |
| Histogram Type | "UniformAdaptive", "QuantilesGlobal", & "RoundRobin" |
| Search Strategy | RandomDiscrete |

**R Code**

**Gradient Boosting Model (GBM)**

*Adopted from h2o’s GBM Tuning Tutorial:* [*https://github.com/h2oai/h2o-3/blob/master/h2o-docs/src/product/tutorials/gbm/gbmTuning.Rmd*](https://github.com/h2oai/h2o-3/blob/master/h2o-docs/src/product/tutorials/gbm/gbmTuning.Rmd)

library**(**tidymodels**)**

library**(**h2o**)**

library**(**rstudioapi**)**

library**(**openxlsx**)**

library**(**janitor**)**

options**(**scipen**=**999**)** #Remove scientific notation

#Get the path of your current open file

current\_path **=** rstudioapi**::**getActiveDocumentContext**()$**path

setwd**(**dirname**(**current\_path **))**

print**(** getwd**()** **)**

#Select location of analysis-ready data file

file\_path **<-** file.choose**()**

#Read data file

df **<-** read.xlsx**(**file\_path**)**

#Clean column names

df **<-** clean\_names**(**df**)**

#Convert character columns to factor

df **<-** df %>%

mutate\_if**(**base**::**is.character, as.factor**)**

###############################################################

#Create predictive models

###############################################################

h2o.init**()** #initialize H2O

# Create dataframe with ONLY your outcome variable ("nsc\_fall\_to\_fall" or similar) and all of your predictor variables.

#df\_model <- #ADD HERE

# Define the name of the outcome variable (whether student was retained from 1st to 2nd year)

y **<-** "nsc\_fall\_to\_fall"

# Define the names of the predictor variables by finding the set difference between all column names in df\_model and the outcome variable name

x **<-** setdiff**(**names**(**df\_model**)**, y**)**

1. E.g. https://github.com/h2oai/h2o-3/blob/master/h2o-docs/src/product/tutorials/gbm/gbmTuning.Rmd [↑](#footnote-ref-1)
2. See <https://docs.h2o.ai/h2o/latest-stable/h2o-docs/performance-and-prediction.html#logloss> [↑](#footnote-ref-2)
3. <https://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/stacked-ensembles.html> [↑](#footnote-ref-3)
4. For introduction to individual conditional expectations, see <https://christophm.github.io/interpretable-ml-book/ice.html> [↑](#footnote-ref-4)