An Exploration of Analysis Methods on Predictive Models of Student Success

Alex Beckwith

May 2023

Quick Summary

- Built a system to train, test, & evaluate machine learning models
- Applied to educational data from an online university
- Used system to generate predictions
- Analyzed results

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 had 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 & 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.

Presentation Itinerary

1. Research Questions
2. Motivations
    - Personal/Project
    - Research
3. Previous Research
    - Learning Analytics/Education Data Mining
    - Predictive Models of Student Performance
    - Prior Experiments with Chosen Dataset
    - Model Evaluation Methods
4. Data Analysis
5. Experimental Architecture
6. Model Evaluation
7. Discussion
8. Conclusion

Research Questions

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

Goals/Motivations

- Personal
    - Research personally relevant topic (education)
    - Apply knowledge of SQL/Python/data from job as data analyst
    - Apply interest/knowledge of predictive models learned independently and in data science program
    - Increase knowledge of statistical evaluation methods
- Research
    - Evaluate machine learning models using best practices/methods/tooling
    - Determine if Bayesian or Frequentist methods are better for machine learning problems
    - Test new metric for evaluation of model fairness
    - Apply above goals to case study with education dataset

- Research

- Build database to organize data management

    - Optimize data storage
    - Transform data for analysis
        - Ad hoc
        - Feature engineering
    - Log model params and results
    - SQL / Python ## Project Milestones
- Engineer features

    - Create all features
    - Create all features using first30 data
- Build model pipeline
    - Data preprocessing
    - Feature selection
    - Parameter selection
    - Model training
    - Model testing
    - Logging
        - Settings
        - Metrics
        - Results ## Project Milestones
- Run Model Pipeline
    - Create gridsearch hyperparamter selection
    - Create randomsearch hyperparameter selection
- Evaluate Models
    - Frequentist Methods
    - Bayesian Methods
    - Slicing Analysis (Fairness)

Learning Analytics/Education Data Mining

Predictive Models of Student Performance

Prior Experiments with Chosen Dataset

Main areas related to Educational Data Mining/Learning Analytics

Previous Research

Learning Analytics / Educational Data Mining

Predicting Student Performance

- Most common types of prediction:
    1. Classification
    2. Regression
    3. Clustering

- Most common algorithms:
    1. Tree-based
        - Decision Tree
        - Random Forest
        - Boosted

2. Regression
    - Logistic Regression
    - Linear Regression
3. Support Vector Machine
4. Bayesian
    - Naive Bayes
5. K-Nearest-Neighbor
6. Artificial Neural Networks

- Common predictions:
    1. Final outcome
        - Dropout
        - Pass/Fail
    2. Final grades
    3. Deadline compliance

- Most common student data sources:
    1. Online learning environment
    2. In-person (More data available, data more consistent)

- Most common feature types:

    1. Academic data
        - Assessments
    2. Demographic data
    3. Behavior
        - Virtual learning environment (VLE) interactions
    4. Financial aid data
- Automated vs Expert Engineered

    - Some automated feature engineering, but most not complex
    - More detailed data -> more opportunities for advanced features
    - relationship between interpretability & performance

Dataset

- Open University
    - Exclusively online university
    - Largest university by enrollment in UK
    - Provision one of the largest public learning analytics datasets

- The dataset
    - Open University Learning Analytics Dataset (OULAD)
        - Anonymized using ARX anonymization tool
    - Massively Open Online Courses (MOOC)
        - 2 years

    - 7 courses
    - 23 presentations
    - 32,593 students
  - 7 Tables
    - Course Info
    - Student Info
    - Assessment Info
    - Virtual Learning Environment (VLE) Summaries
      - (clicks per day, per resource, per student)
    - 3 Bridge Tables

- For course to be included in OULAD
  - The number of students in the selected module-presentation is larger than 500.
  - At least two presentations of the module exist.
  - VLE data are available for the module-presentation (since not all the modules are studied via VLE).
  - The module has a significant number of failing students.

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

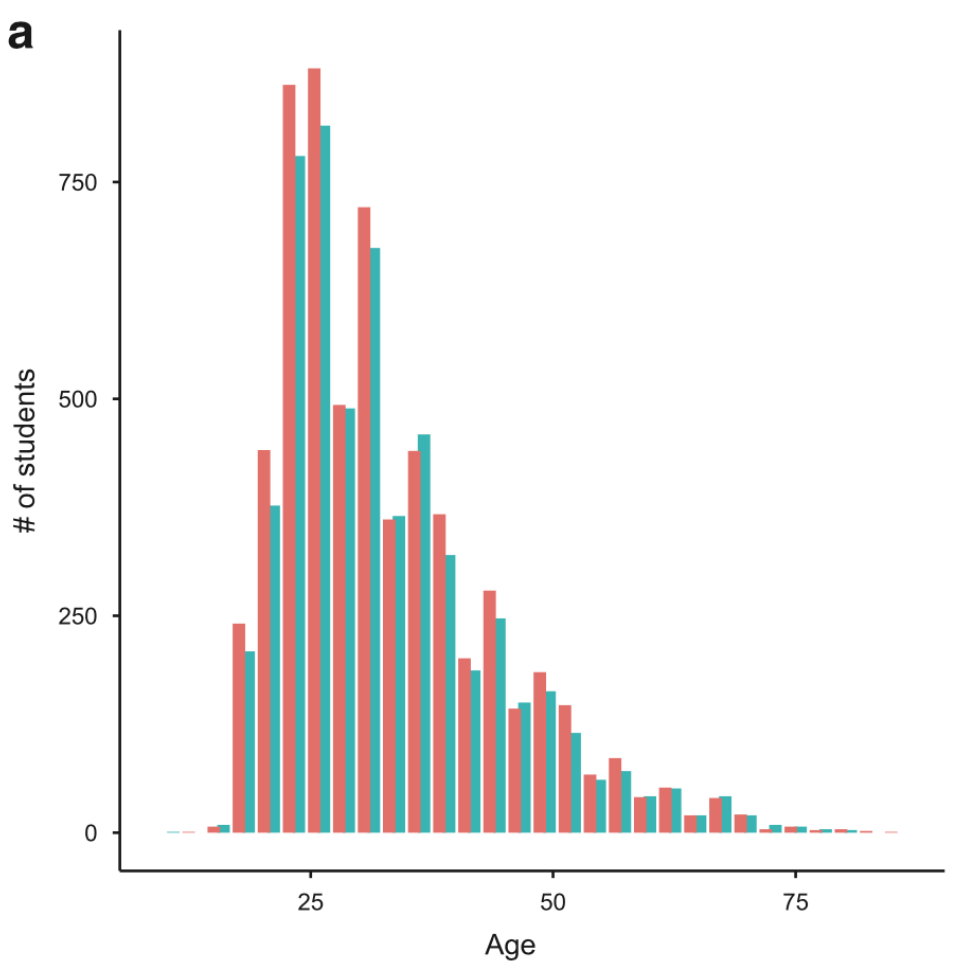
Table 1. Module summary and domain information.

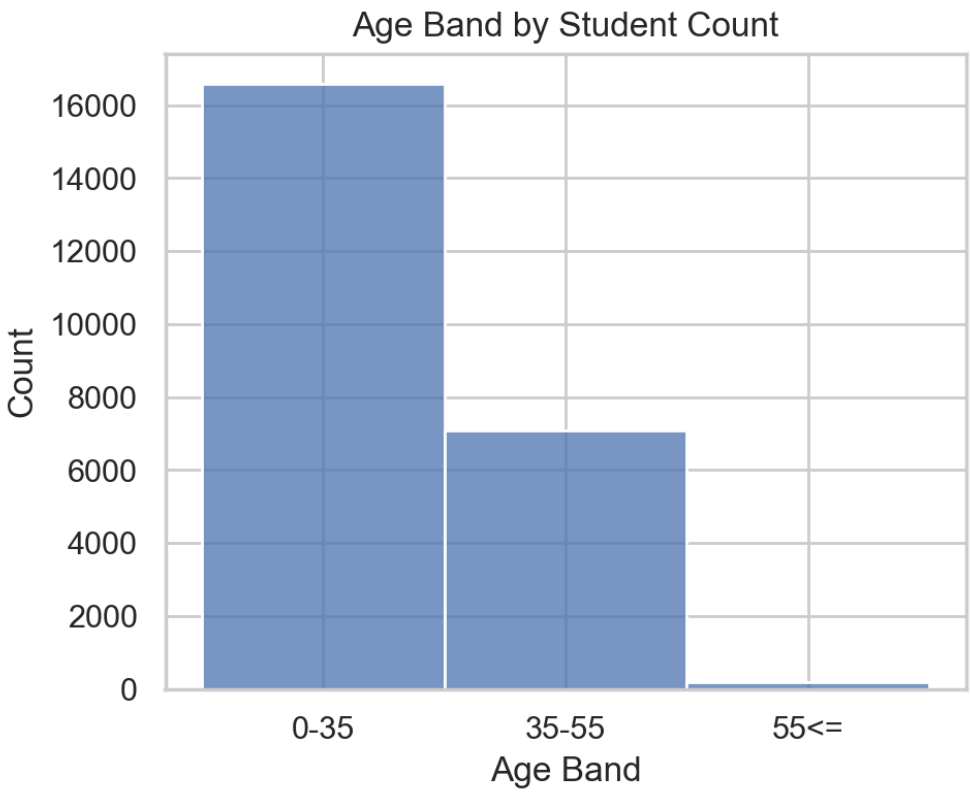
Attribute	Test	Degrees of freedom	Test value	*P*-value
Age	Wilcox	—	17,606,000	0.3558
Disability	χ^2	1	1.0739	0.3001
Education	χ^2	4	0.89201	0.9257
Gender	χ^2	1	2.0537	0.1518
IMD	χ^2	19	15.912	0.6631
Region	χ^2	12	16.325	0.1768

Table 2. Evaluation of similarity between OULAD and 2015 data for CCC module.

Age (age_band)

```
OULAD (Red & 2013-2014), vs 2015 data (Blue)
```





Describing first30.all_features.age_band

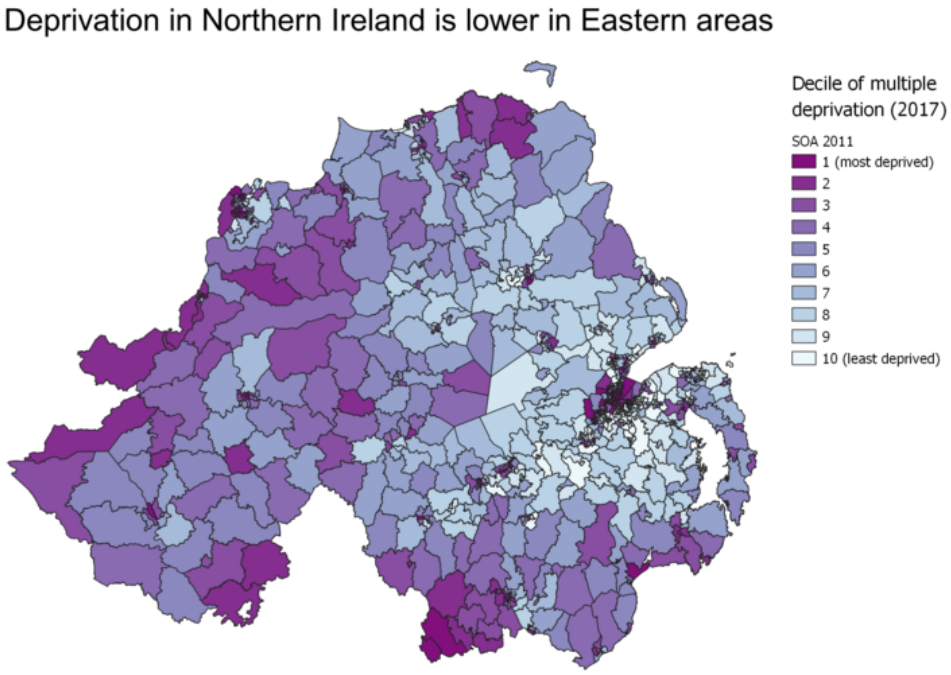
Proportions:

	value	frequency	proportion
0	0-35	16590	0.694956
1	35-55	7091	0.297043
2	55<=	191	0.008001

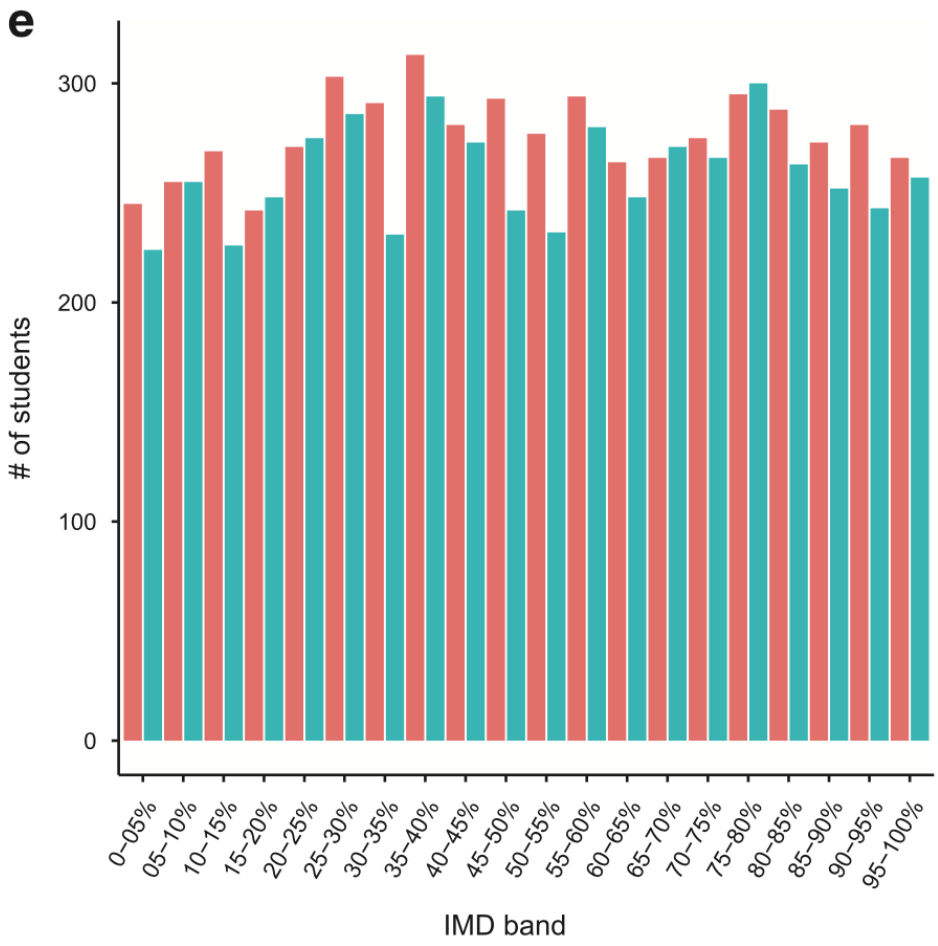
```
Null count: 0
count     23872
unique        3
top        0-35
freq      16590
Name: age_band, dtype: object
```

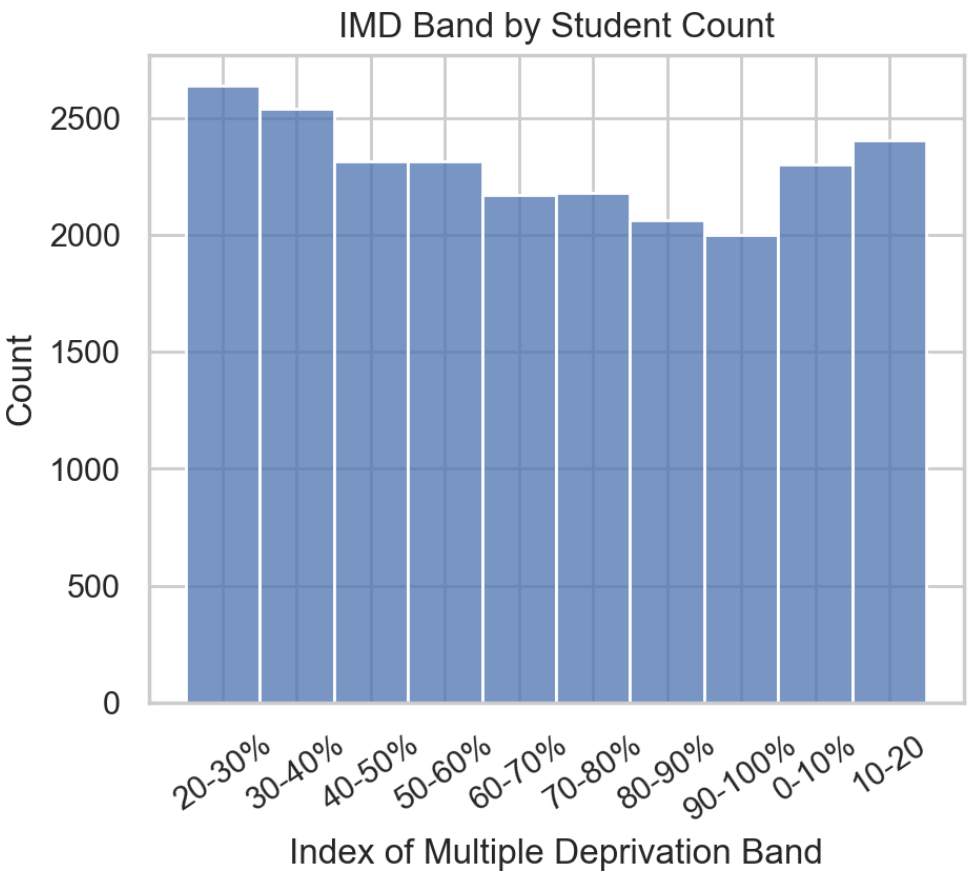
Index of Multiple Deprivation (imd_band)

Example Map of IMD



OULAD (Red & 2013-2014), vs 2015 data (Blue)





In the current English Indices of Deprivation 2019 (IoD2019) seven domains of deprivation are considered and weighted as follows,

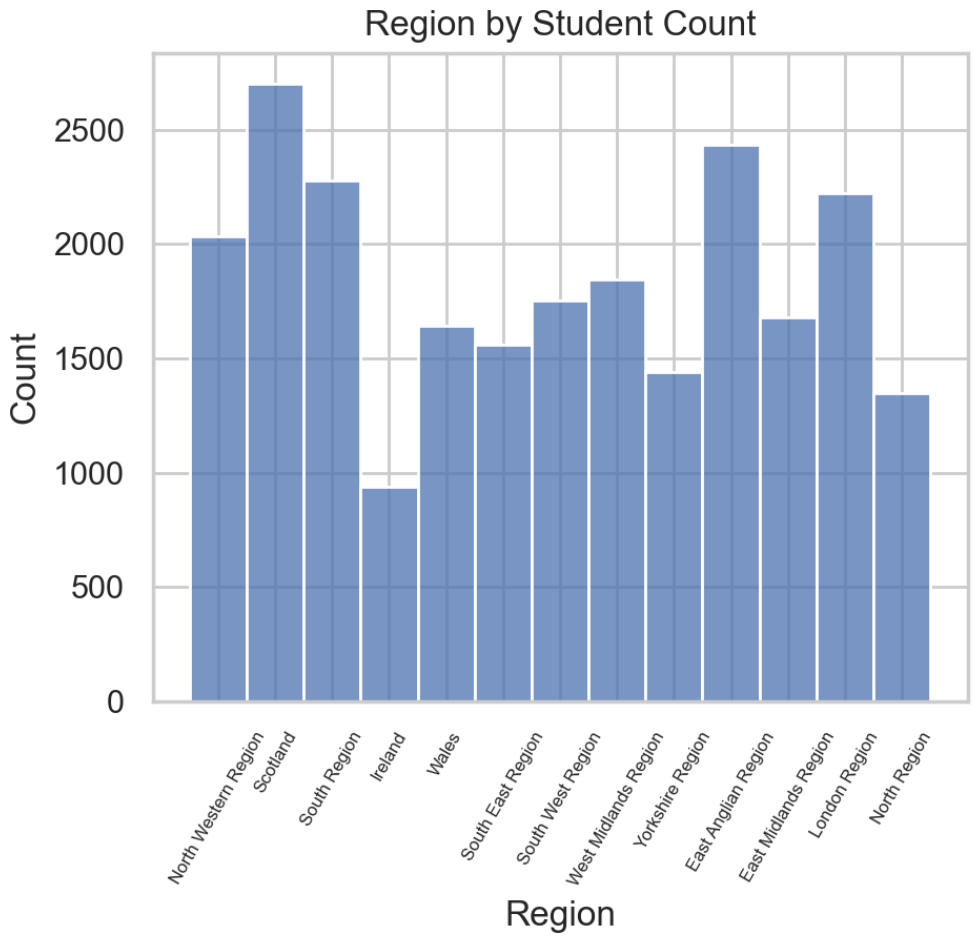
- Income. (22.5%)
- Employment. (22.5%)
- Education. (13.5%)
- Health. (13.5%)
- Crime. (9.3%)
- Barriers to Housing and Services. (9.3%)
- Living Environment. (9.3%)

```
Describing first30.all_features.imd_band

Proportions:
       value  frequency  proportion
0    20-30%       2639    0.110548
1    30-40%       2539    0.106359
2    40-50%       2314    0.096934
3    50-60%       2316    0.097017
4    60-70%       2171    0.090943
5    70-80%       2180    0.091320
6    80-90%       2063    0.086419
7   90-100%       2001    0.083822
8      None          0    0.000000
9     0-10%       2303    0.096473
10    10-20       2406    0.100788

Null count: 940
count      22932
unique        10
top       20-30%
freq        2639
Name: imd_band, dtype: object
```

Region

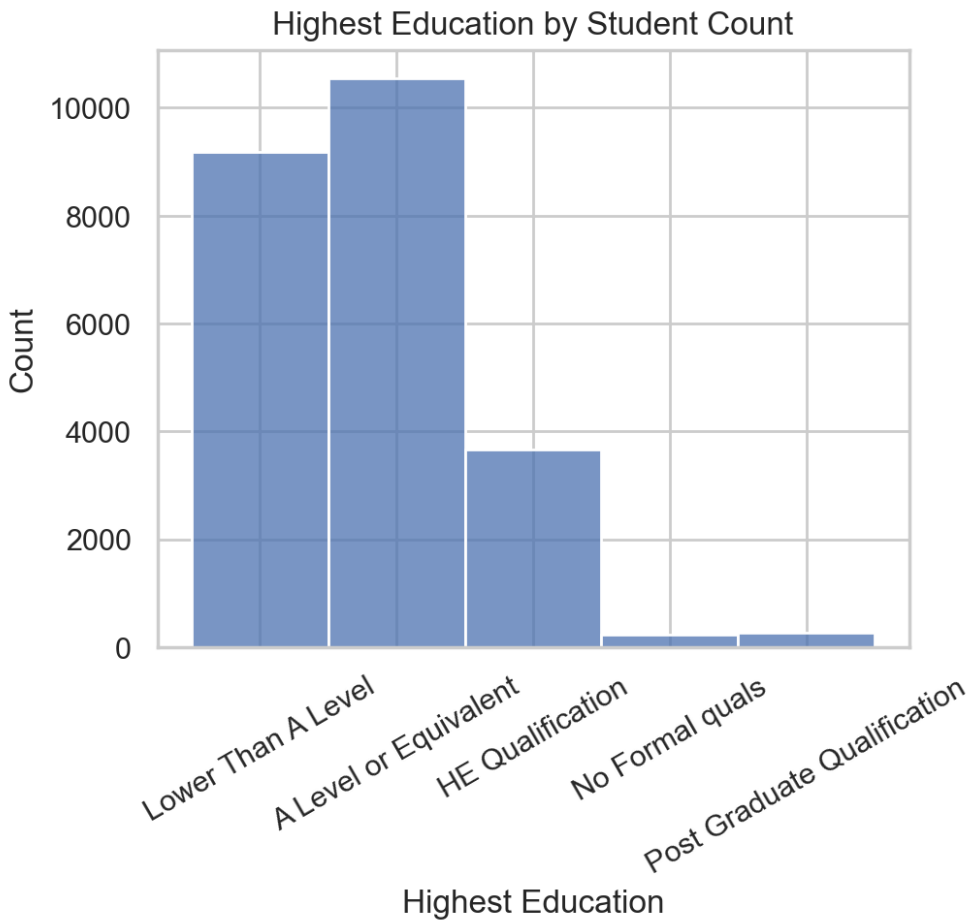



```
Describing first30.all_features.region

Proportions:
                    value  frequency  proportion
0   North Western Region       2032    0.085121
1               Scotland       2701    0.113145
2           South Region       2278    0.095426
3                Ireland        938    0.039293
4                  Wales       1642    0.068784
5      South East Region       1559    0.065307
6      South West Region       1753    0.073433
7   West Midlands Region       1845    0.077287
8       Yorkshire Region       1440    0.060322
9    East Anglian Region       2434    0.101960
10  East Midlands Region       1680    0.070375
11         London Region       2223    0.093122
12          North Region       1347    0.056426

Null count: 0
count        23872
unique          13
top       Scotland
freq          2701
Name: region, dtype: object
```

Highest Education



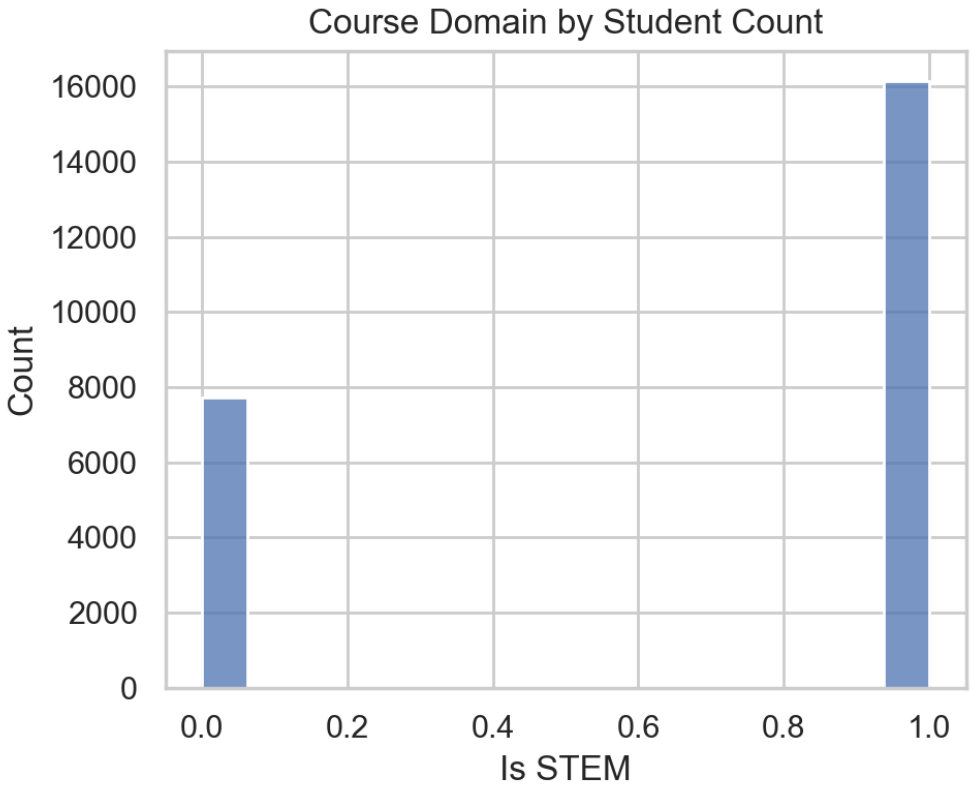

```
Describing first30.all_features.highest_education

Proportions:
                         value  frequency  proportion
0           Lower Than A Level       9187    0.384844
1        A Level or Equivalent      10541    0.441563
2             HE Qualification       3660    0.153318
3              No Formal quals        225    0.009425
4  Post Graduate Qualification        259    0.010850

Null count: 0
count                     23872
unique                        5
top       A Level or Equivalent
freq                      10541
Name: highest_education, dtype: object
```

Course Domain

- STEM or Social Studies

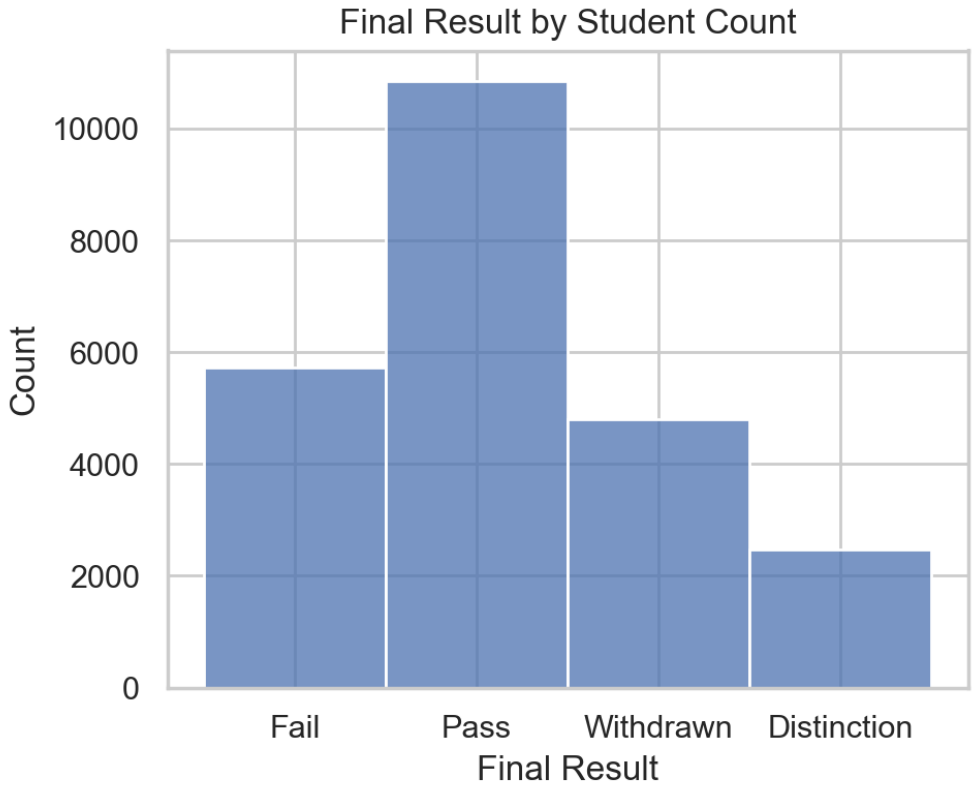



```
Describing first30.all_features.is_stem

Proportions:
    value  frequency  proportion
0       1      16147    0.676399
1       0       7725    0.323601

Null count: 0
count    23872.000000
mean         0.676399
std          0.467860
min          0.000000
25%          0.000000
50%          1.000000
75%          1.000000
max          1.000000
Name: is_stem, dtype: float64
```

Final Result




```
Describing first30.all_features.final_result

Proportions:
          value  frequency  proportion
0          Fail       5728    0.239946
1          Pass      10857    0.454801
2    Withdrawn        4811    0.201533
3  Distinction        2476    0.103720

Null count: 0
count     23872
unique        4
top        Pass
freq      10857
Name: final_result, dtype: object


Describing first30.all_features.region

Proportions:
                    value  frequency  proportion
0   North Western Region       2032    0.085121
1               Scotland       2701    0.113145
2           South Region       2278    0.095426
3                Ireland        938    0.039293
4                  Wales       1642    0.068784
5      South East Region       1559    0.065307
6      South West Region       1753    0.073433
7   West Midlands Region       1845    0.077287
8       Yorkshire Region       1440    0.060322
9    East Anglian Region       2434    0.101960
10  East Midlands Region       1680    0.070375
11         London Region       2223    0.093122
12          North Region       1347    0.056426

Null count: 0
count        23872
unique          13
top       Scotland
```

```
freq           2701
Name: region, dtype: object
```

Model Evaluation

Naive Averaging

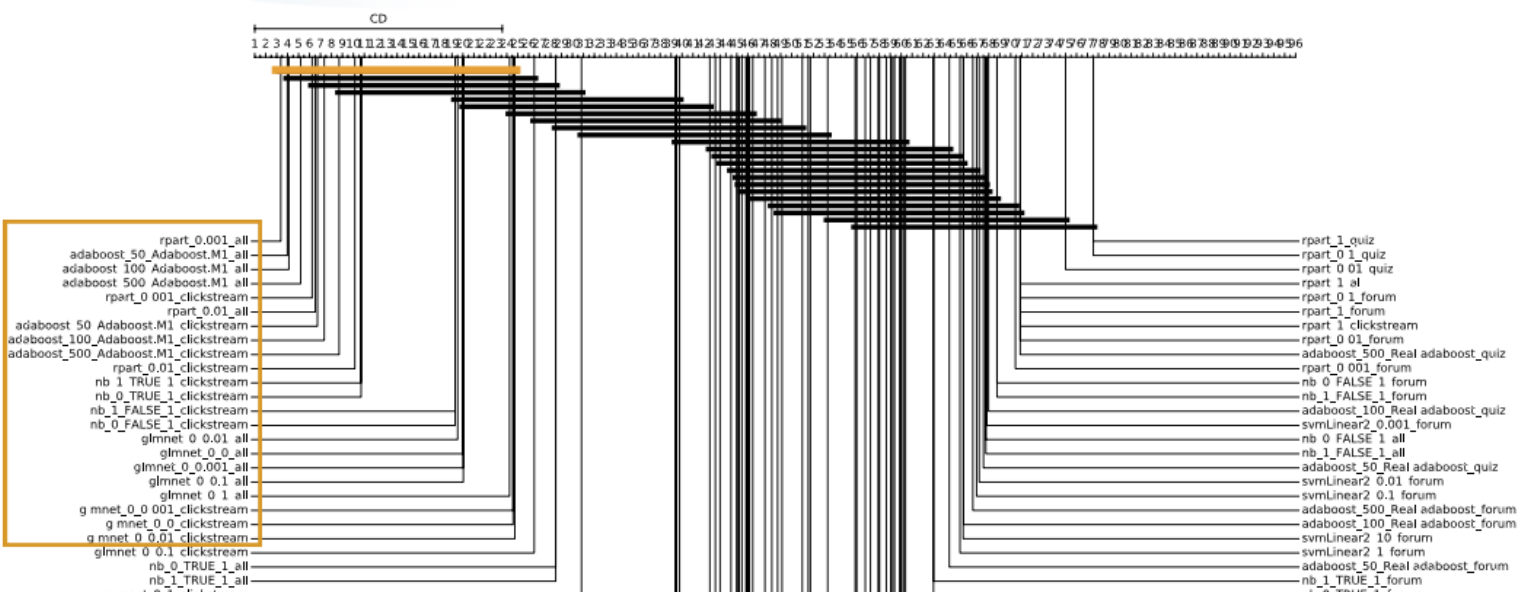
- Simply sorting by metric and picking top value
- Difficult to discern differences between models
- No sense of variability between model types/settings
- Tough to weigh other factors like interpretability & fit time in justified way

	model_type	mean_fit_time	std_fit_time	mean_score_time	std_score_time	mean_test_roc_auc	std_test_roc_auc
0	rforest	43.546772	6.621612	0.308331	0.079489	0.773876	0.006709
1	etree	22.712446	2.100217	0.221903	0.027837	0.771116	0.004962
2	hxg_boost	8.673935	0.686707	0.297604	0.038754	0.770126	0.004659
3	mlp	13.990430	0.143472	0.077275	0.011556	0.769375	0.007105
4	hxg_boost	6.469628	1.027840	0.263747	0.042775	0.769049	0.006249
5	etree	14.824491	3.315720	0.251914	0.041687	0.767965	0.004917
6	hxg_boost	1.569689	0.177629	0.086444	0.013947	0.767947	0.005761
7	mlp	1.304684	0.044562	0.029652	0.002250	0.767719	0.008325
8	rforest	21.928703	4.551270	0.763771	0.412899	0.767640	0.005182
9	hxg_boost	1.494005	0.194275	0.075681	0.012553	0.767497	0.004604

Frequentist (Null Hypothesis Significance Testing)

- Used non-parametric tests to minimize assumptions of distributions of model data
- Friedman Test indicated that a significant difference exists in top model results
- Post-hoc Nemenyi Test to indicate if significant difference exists between two models
  - Model uniquely identified by [model type, distinct parameter settings, and featureset]
- Tough to compare large set of models in this way

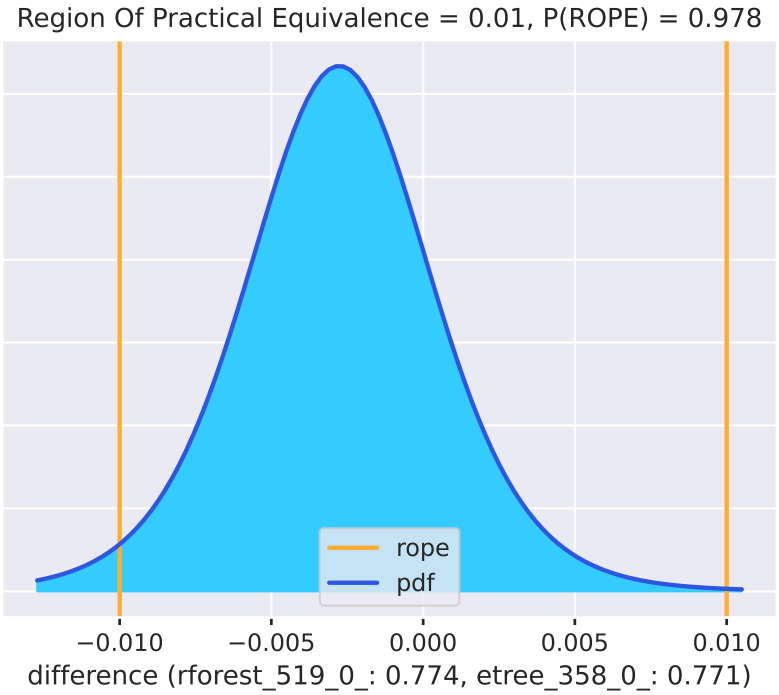
```
Output from test on results from single dataset
```



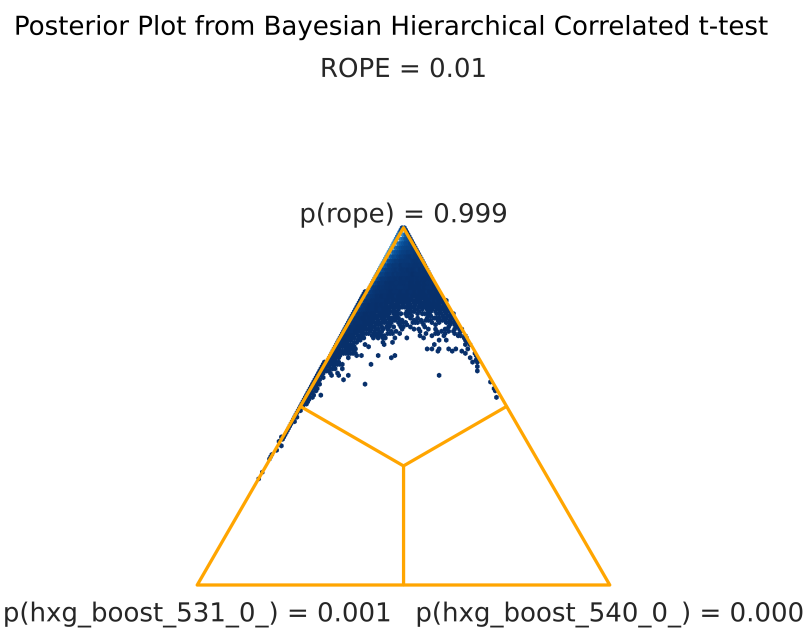
Bayesian

- Uses test based on Bayesian hierarcical modeling to estimate probability of means being in a prespecified "Region of Practical Equivalence" (ROPE)
- This test used a ROPE vaue of 0.01, indicating that a 1% difference in means is a wide enough band to consider the performance of two models equivalent for all practical purposes

`Output from test on results from single dataset | Ranked Sign Test`



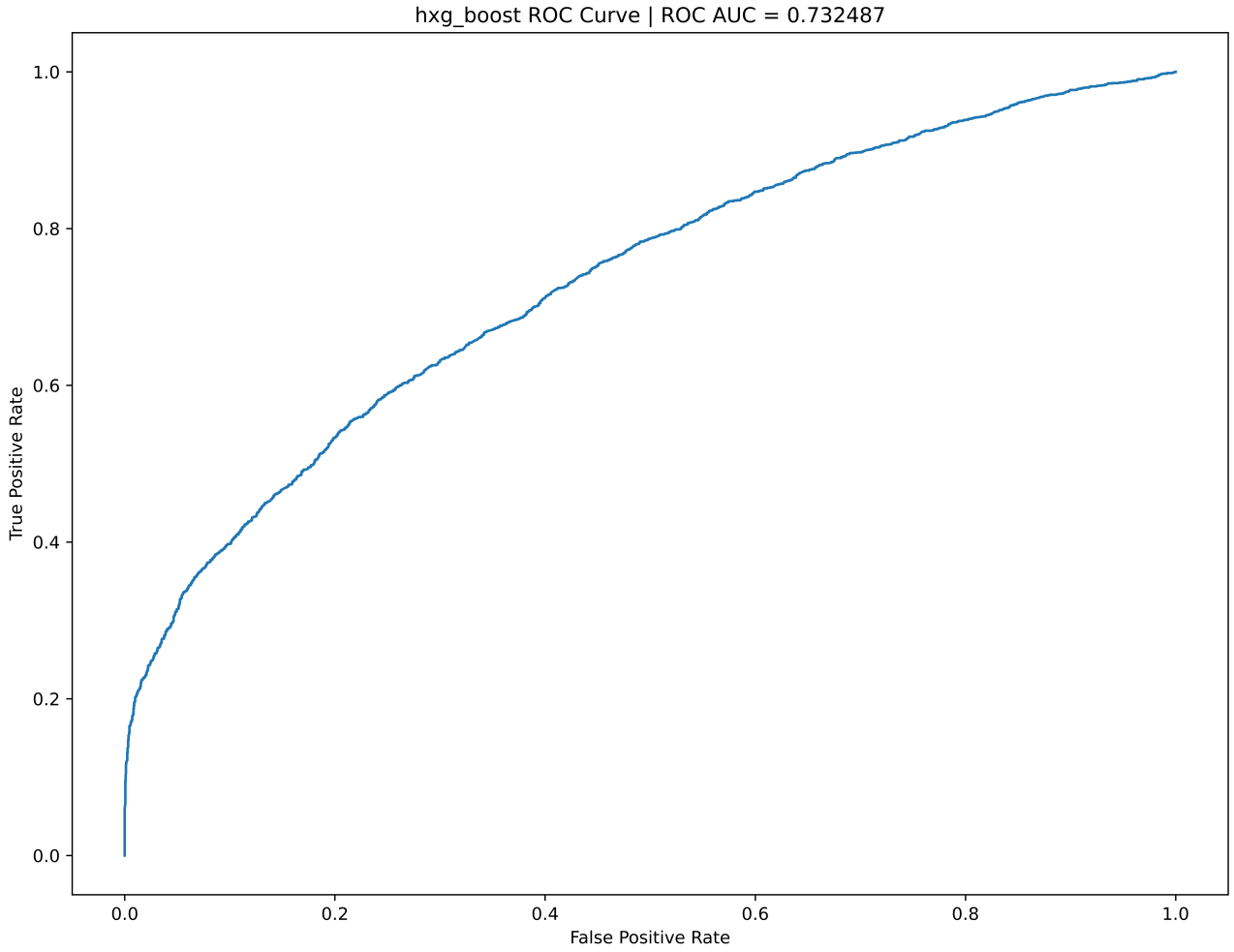
`Output from test on multiple datasets:`



Absolute Between Receiver Operating Characteristic Area | Slicing Analysis

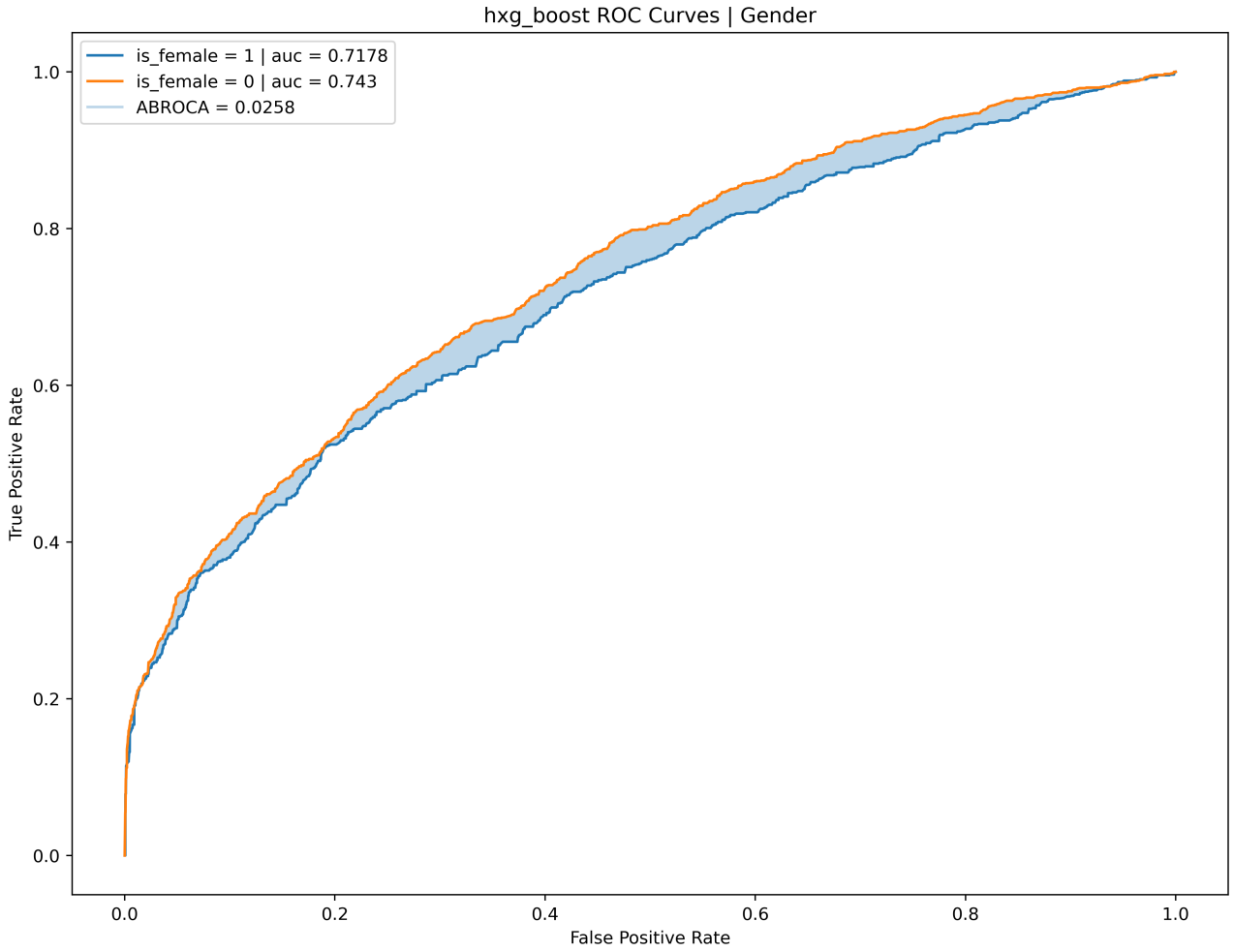
ROC Curve

- The Receiver Operating Characteristic (ROC) is a plot of the false positive rate to true positive rate over the range of threshold values
  - $x \in [0, 1]$
- Area under ROC curve (ROC AUC) commonly used as metric to optimize performance of machine learning models.
  - Perfect predictor -> ROC AUC = 1.0 (correct prediction at all threshold values)
  - Random predictor -> ROC AUC = 0.5 (equally likely to pick correctly or incorrectly at all threshold values)



ABROCA | Slicing Analysis

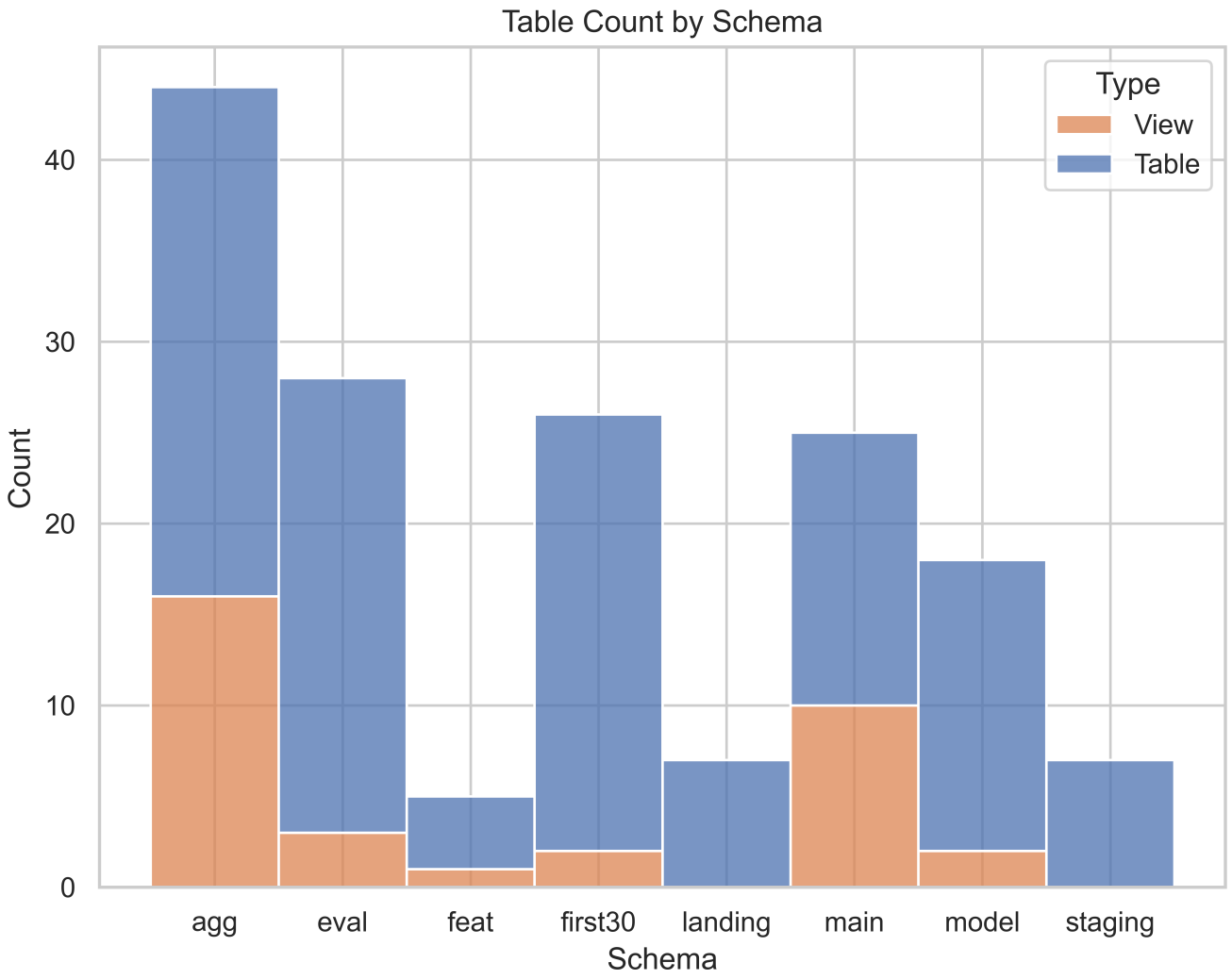
- Proposed as metric with which to compare predictive model fairness
  - First introduced at 2019 International Learning Analytics and Knowledge Conference
- To calculate:
  - Split dataset by feature of interest
  - Calculate ROC curves for the model on each part of split dataset
  - Sum absolute values of between-curve area
- How does this relate to fairness?
  - A model that predicts subgroups of split dataset equally would have ABROCA = 0 (Same ROC curves, so no area between)
  - Hypothesis - Higher ABROCA associated with lower predictive fairness



Data Processing/Analysis

Initial Database Schemas

- Landing
  - Raw CSV load
- Staging
  - Datatype and naming standardization
- Main [Maybe ERD]
  - Data architecture optimization
  - Categorical/text columns stored in tables linked with integer foreign keys
  - Joined data saved in views

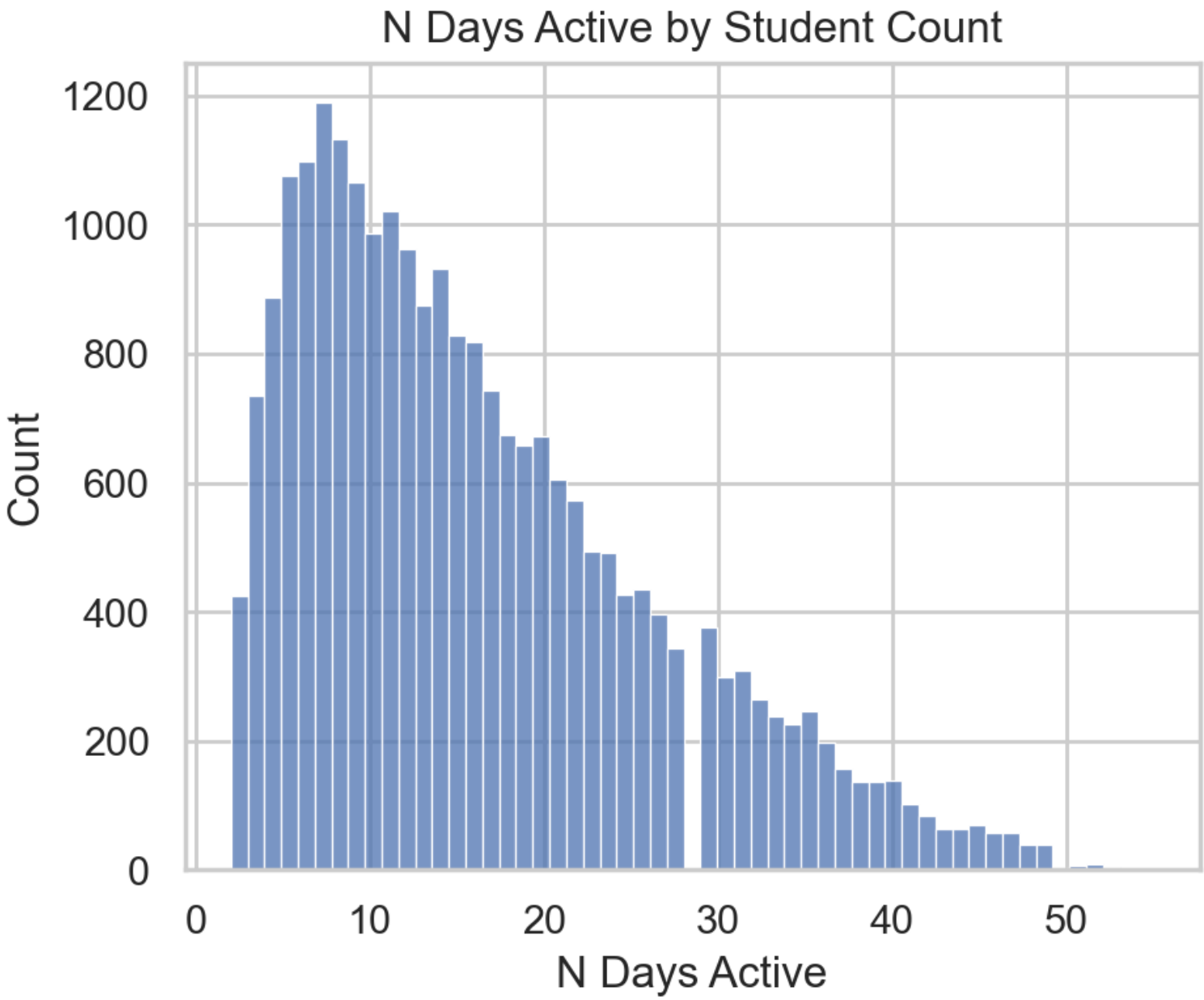


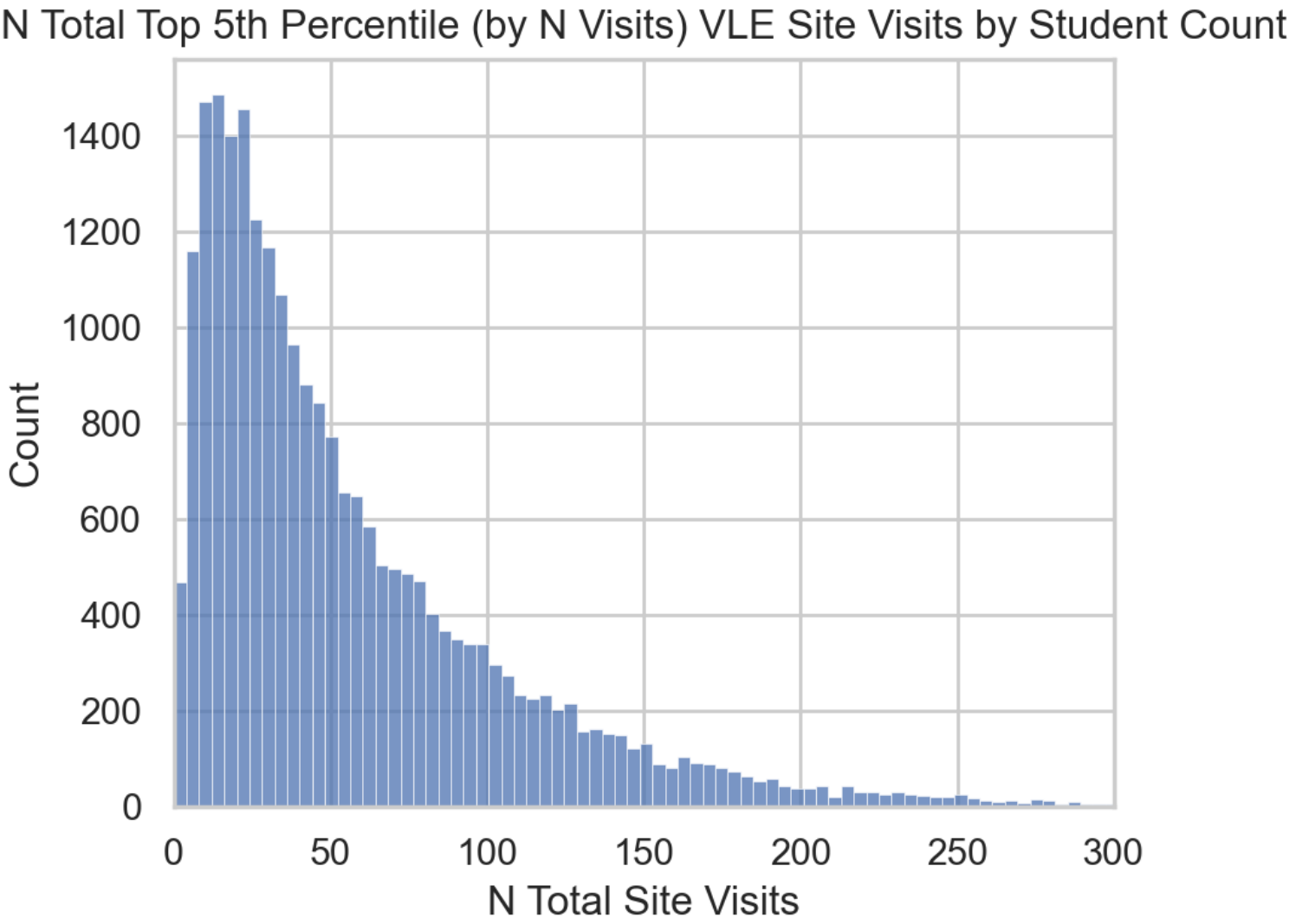
- Agg
  - Aggregations and calculations
  - [Avg Assignment Days Early by N Days Active]
- Feat
  - First pass at organizing features/calculations for predictive models
  - [N Days Active]
  - [N Distinct Top 5th by Visits]
- First30
  - Version of Feat created using first 30 days of class data
  - Captures 49.60% of all withdrawn students
  - Captures 72.22% of students who withdrew after class started
  - Soon enough to make actionable difference to most withdrawing/failing students
  - [Final Result]
- Model
- Eval

Experimental Architecture

Feature Engineering

- Categories
  - Demographic Info
  - Course Info
  - VLE Interaction Data
  - Assignment Data





Classification Algorithms & HyperParameters

- Grid Search
  - Created large arrays of available hyperparameters
  - Brute-force search through combination of available hyperparameters
- Random Search
  - Used GridSearch to limit the bounds of hyperparameter settings
  - Created random variables to represent distribution of particular hyperparamters, limited by results from GridSearch
  - Ran models where each iteration would pick from a model's available parameter combinations and distributions

```
Example of GridSearch Cross-Validation for hxg_boost:
clf__learning_rate [0.1]
clf__random_state [None]


clf__learning_rate [0.01]
clf__random_state [None]


clf__learning_rate [0.001]
clf__random_state [None]
```

Decision Tree (dtree/DT)

- Simple decision rules are optimized from features to sort data

```
Example of RandomizedSearch Cross-Validation for dtree:
clf__splitter ['best']
clf__random_state [None]
clf__min_samples_split [84]
clf__min_samples_leaf [2]
clf__max_features [None]
clf__max_depth [52]
clf__criterion ['log_loss']
```

Ada Boost (ada_boost/ADA)

- Ensemble Method
- Fits on original dataset, then creates copies which weight incorrectly classified instances more heavily in sequential cycles
- Used Decision Tree as base estimator, but can use many

```
Example of RandomizedSearch Cross-Validation for ada_boost:
clf__learning_rate [0.016322649720355895]
clf__random_state [None]
```

Histogram Gradient-Boosting (hxg_boost/HGB)

- Similar to Ada Boost, but correction based on gradient of loss function from residuals (gradient descent)
- Dataset large enough that Histogram Gradient-Boosting Classifier much faster than Regular Gradient-Boosting Classifier
- Histograms increase training efficiency by bucketing continuous features

```
Example of RandomizedSearch Cross-Validation for hxg_boost:
clf__interaction_cst ['no_interactions']
clf__l2_regularization [0.24841032861067147]
clf__learning_rate [0.0180795552628978]
clf__max_bins [203]
clf__max_depth [32]
clf__max_iter [30]
clf__min_samples_leaf [6]
clf__random_state [None]
clf__warm_start [False]
```

Random Forest (rforest/RF)

- Ensemble Method
- Fits many decision trees on sub-samples of dataset, then uses averaging to boost accuracy and control over-fitting

```
Example of RandomizedSearch Cross-Validation for rforest:
clf__bootstrap [True]
clf__criterion ['log_loss']
clf__max_features ['log2']
clf__max_samples [0.36012871046936756]
```

```
clf__min_samples_leaf [6]
clf__min_samples_split [7]
clf__n_estimators [150]
clf__n_jobs [-1]
clf__oob_score [True]
clf__random_state [None]
```

Extra Trees (etree/ET)

- Ensemble Method
- Fits many decision trees on sub-samples of dataset, then uses averaging to boost accuracy and control over-fitting

```
Example of RandomizedSearch Cross-Validation for etree:
clf__bootstrap [True]
clf__criterion ['gini']
clf__max_features ['sqrt']
clf__max_samples [0.4409163442397114]
clf__min_samples_leaf [5]
clf__min_samples_split [8]
clf__n_estimators [43]
clf__n_jobs [-1]
clf__oob_score [True]
clf__random_state [None]
```

Extra Trees vs Random Forest

- Both construct many decision trees during execution & avg for classification/regression
- RF uses bootstrapping to sample subsets, ET by default does not
- RF looks for best split, ET randomly selects split
- ET typically will have faster fit times & lower variance, higher bias
- Performance of ET vs RF is often conditional upon feature selection/noisiness

K-Nearest Neighbor (knn/KNN)

- Calculates most likely value based on closest points in discrete numeric space

```
Example of RandomizedSearch Cross-Validation for knn:
clf__weights ['distance']
clf__p [1]
clf__n_neighbors [5]
clf__n_jobs [-1]
clf__leaf_size [72]
clf__algorithm ['ball_tree']
```

Logistic Regression (logreg/LOG)

- Calculates most likely value based on weighted contribution of feature values

```
Example of RandomizedSearch Cross-Validation for logreg:
clf__C [0.050188292655831426]
clf__max_iter [40]
clf__n_jobs [-1]
clf__penalty [None]
```

```
clf__random_state [None]
clf__solver ['lbfgs']
```

Multi-Layer Perceptron (mlp/MLP)

- Simple (vanilla) neural network
- Consists of layers of connected nodes with activation functions
- Optimizes weights of nodes in each layer using backpropogation during training
- Last layer is output layer, which produces most likely result given trained inputs

```
Example of RandomizedSearch Cross-Validation for mlp:
clf__activation ['identity']
clf__alpha [0.024419617927526942]
clf__early_stopping [True]
clf__hidden_layer_sizes [187]
clf__learning_rate ['invscaling']
clf__learning_rate_init [0.008063494205721687]
clf__max_iter [46]
clf__power_t [0.05170481714548528]
clf__random_state [None]
clf__solver ['adam']
```

Support Vector Machines (svc/SVC)

- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- A hyperplane is optimized to best split the data into different spatial regions

```
Example of RandomizedSearch Cross-Validation for svc:
clf__C [0.20456796576588798]
clf__degree [4]
clf__gamma ['auto']
clf__kernel ['poly']
clf__probability [True]
clf__random_state [None]
```

Not Implemented

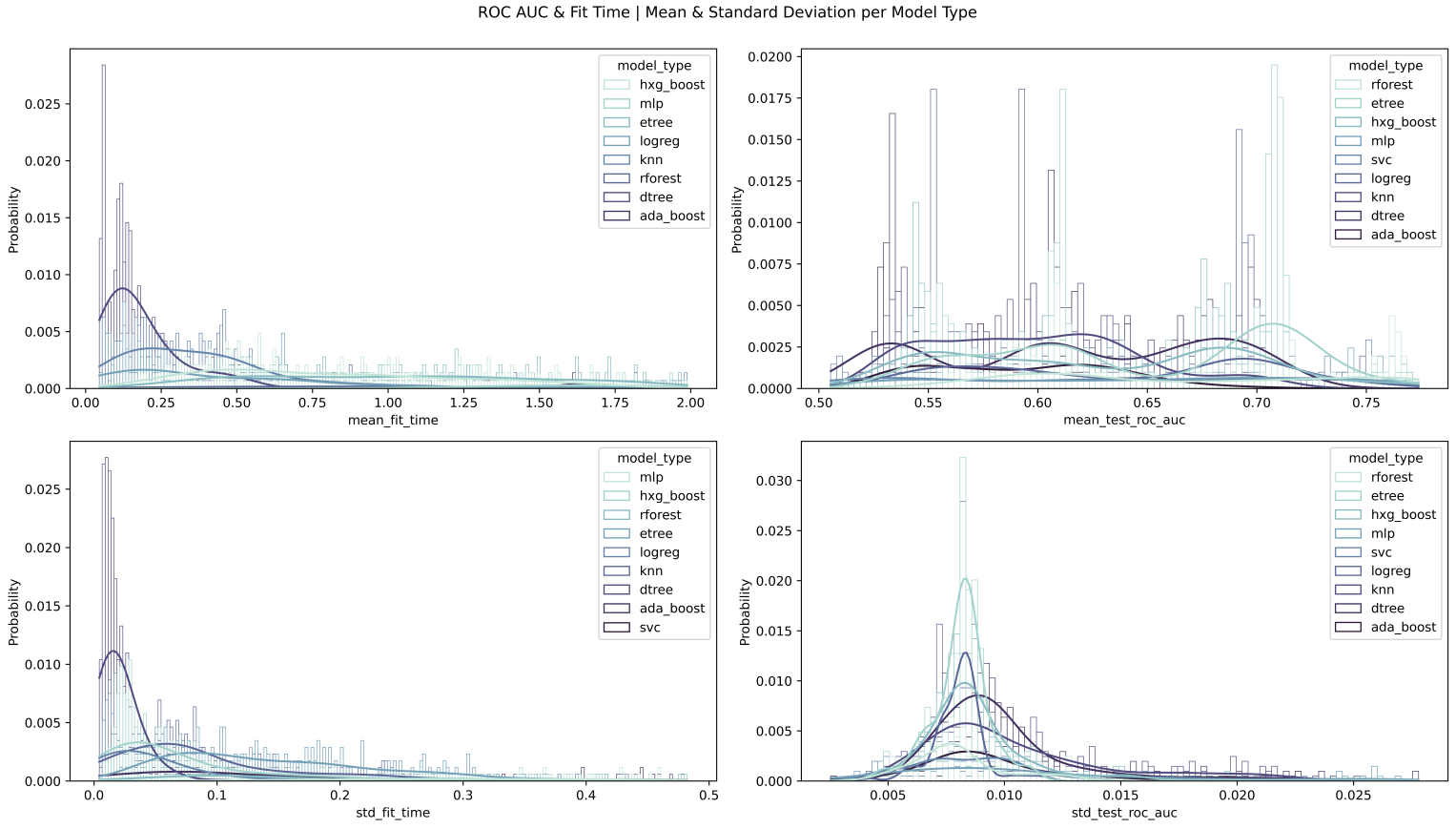
- Others considered but not implemented due to data preprocessing changes necessary/compute/memory overhead
- Gaussian
  - (Blew up RAM)
- Naive Bayes
  - (Would need to preprocess data differently)
- Gradient Boosting
  - (Histogram-Based Algorithm more efficient at this scale)

```
Example of RandomizedSearch Cross-Validation for compnb:
clf__alpha [0.010936544356329807]
clf__norm [True]
```

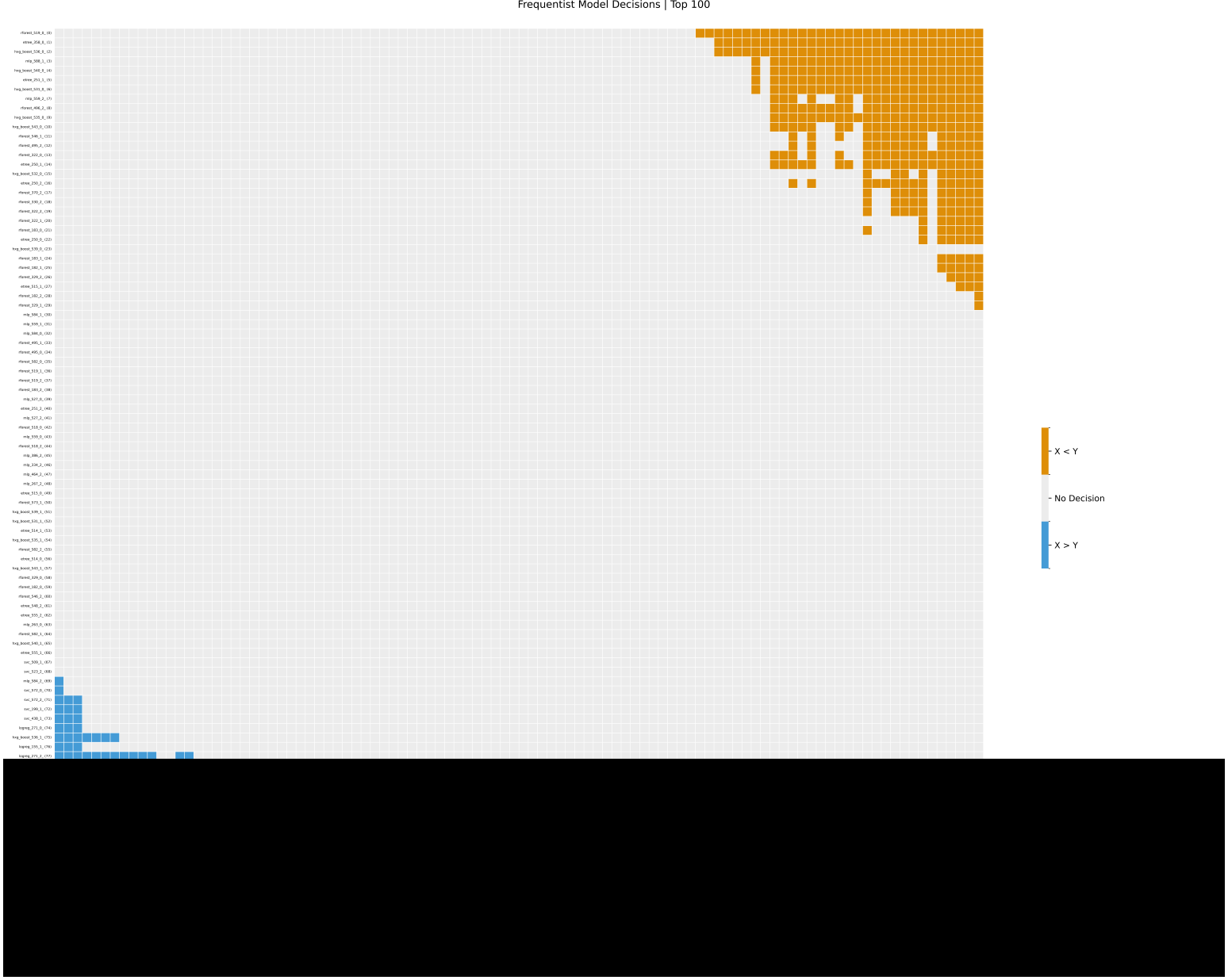
Model Evaluation

Naive Averaging

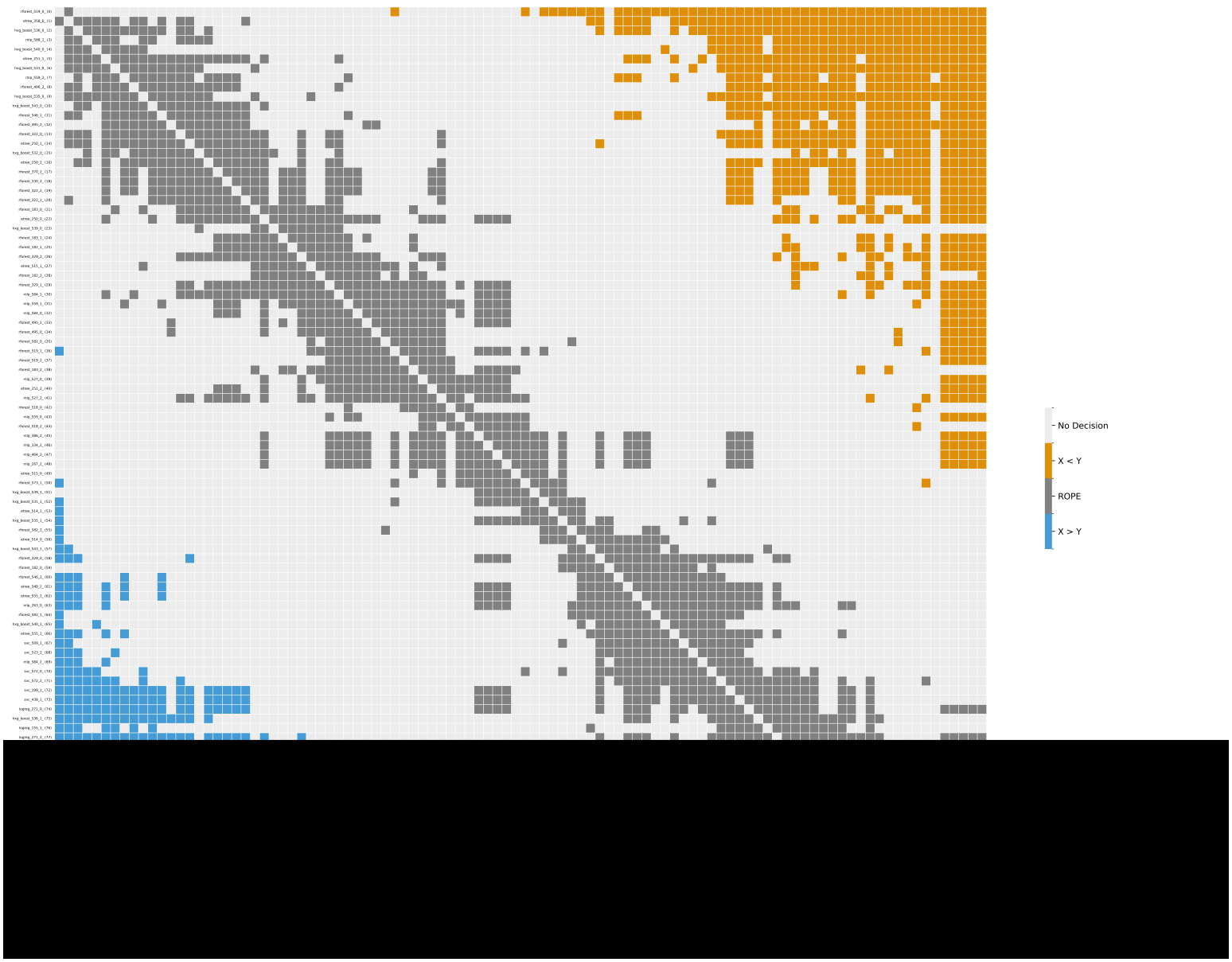
	model_type	mean_fit_time	std_fit_time	mean_score_time	std_score_time	mean_test_roc_auc	std_test_roc_auc
0	rforest	43.546772	6.621612	0.308331	0.079489	0.773876	0.006709
1	etree	22.712446	2.100217	0.221903	0.027837	0.771116	0.004962
2	hxg_boost	8.673935	0.686707	0.297604	0.038754	0.770126	0.004659
3	mlp	13.990430	0.143472	0.077275	0.011556	0.769375	0.007105
4	hxg_boost	6.469628	1.027840	0.263747	0.042775	0.769049	0.006249
5	etree	14.824491	3.315720	0.251914	0.041687	0.767965	0.004917
6	hxg_boost	1.569689	0.177629	0.086444	0.013947	0.767947	0.005761
7	mlp	1.304684	0.044562	0.029652	0.002250	0.767719	0.008325
8	rforest	21.928703	4.551270	0.763771	0.412899	0.767640	0.005182
9	hxg_boost	1.494005	0.194275	0.075681	0.012553	0.767497	0.004604



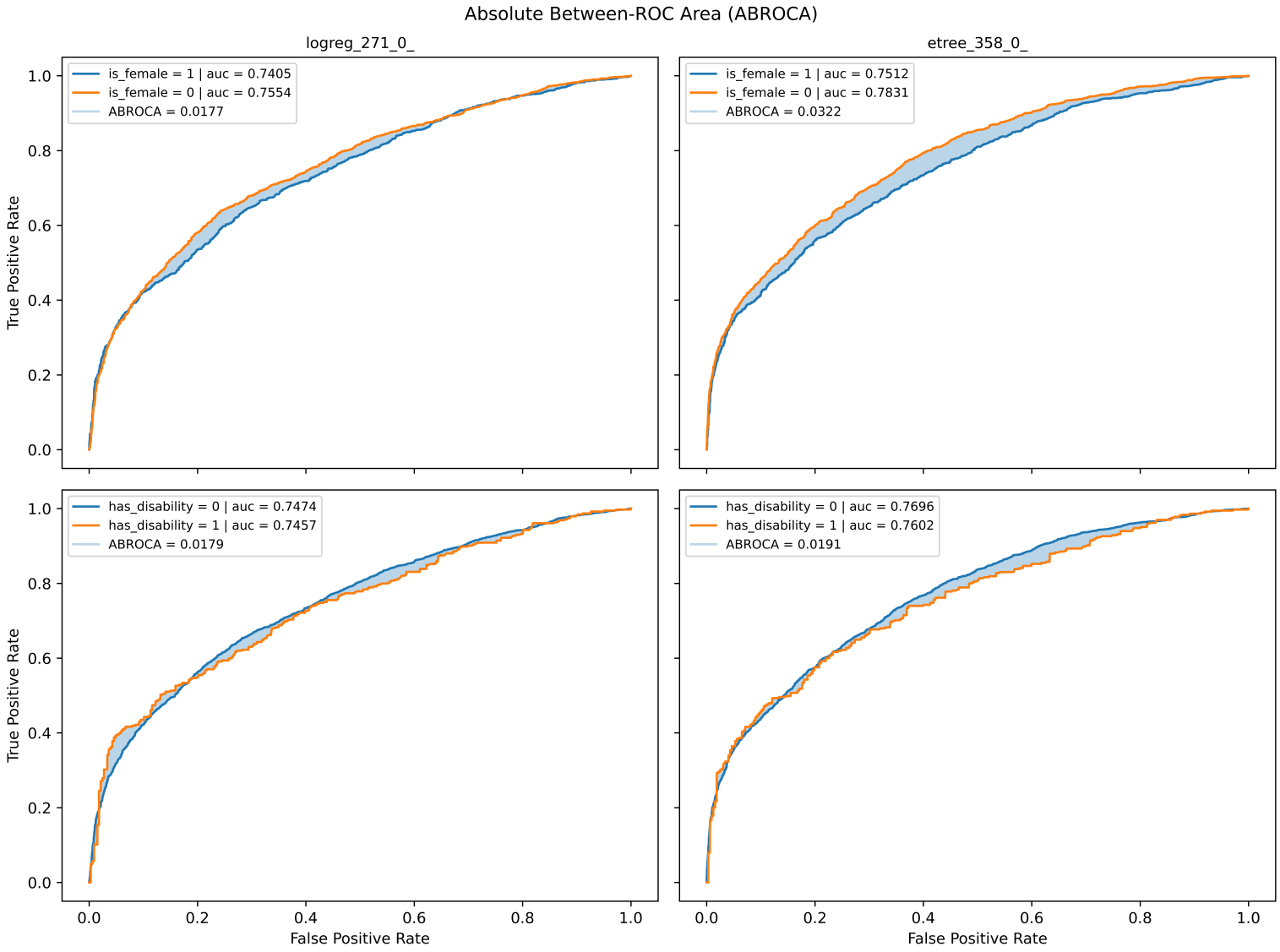
Frequentist

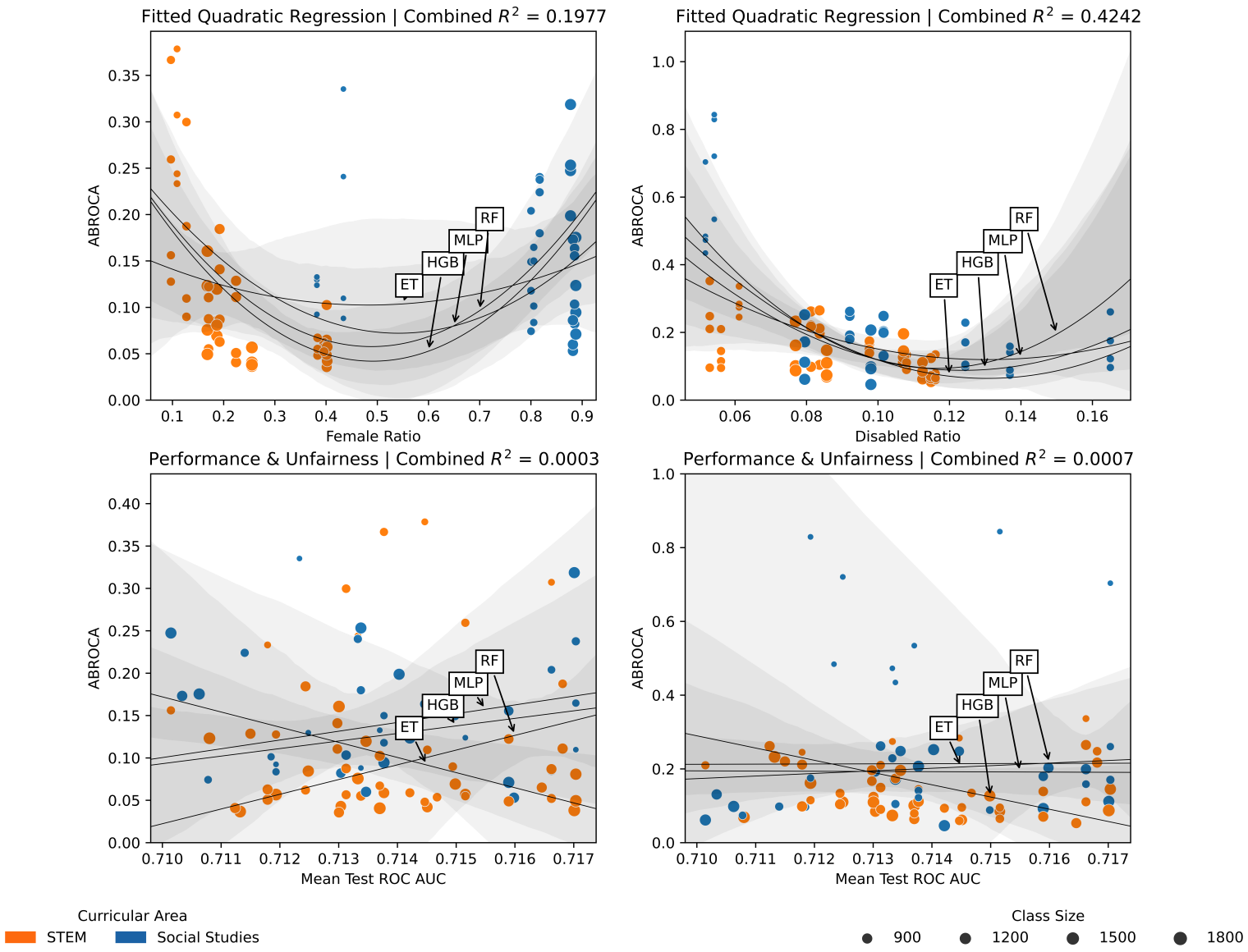


Bayesian



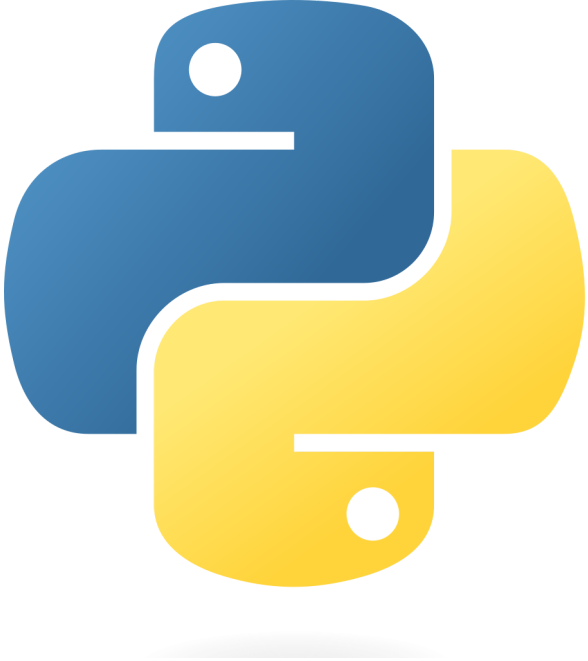
ABROCA


Absolute Between-ROC Area (ABROCA)
logreg_271_0_
etree_358_0_
is_female = 1 | auc = 0.7405
is_female = 0 | auc = 0.7554
ABROCA = 0.0177
is_female = 1 | auc = 0.7512
is_female = 0 | auc = 0.7831
ABROCA = 0.0322
has_disability = 0 | auc = 0.7474
has_disability = 1 | auc = 0.7457
ABROCA = 0.0179
has_disability = 0 | auc = 0.7696
has_disability = 1 | auc = 0.7602
ABROCA = 0.0191
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0



Tools Used

Scripting Language of Choice



Database System

Database Communication

Database Communication & Data Manipulation

Visualizations

Visualizations

Array Computation

Bayesian Statistical Tests

baycomp

by:
- Janez Demsar
- Alessio Benavoli
- Giorgio Corani

Random Variables & Statistical Tests

Machine Learning Models & Components