IMPROVING EDUCATIONAL OUTCOMES FOR STEM STUDENTS AT RUTGERS UNIVERSITY-CAMDEN: A MACHINE LEARNING APPROACH

BY

ARTHUR P. PELULLO

A thesis submitted to the

Graduate School—Camden

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of Master of Science

Graduate Program in Scientific Computing

Written under the direction of

Dr. Sunil Shende

and approved by

____

Dr. Sunil Shende

____

Dr. Ravi Rao

____

Dr. Jean-Camille Birget

Camden, New Jersey

October 2017

THESIS ABSTRACT

Improving Educational Outcomes for Stem Students at Rutgers University-Camden: A Machine Learning Approach

By ARTHUR P. PELULLO

Thesis Director:

Dr. Sunil Shende

The goal of this thesis is to demonstrate how machine-learning techniques can be used to improve educational outcomes for STEM students at Rutgers University-Camden. The three main areas of focus are: identifying changes in the academic landscape throughout a 15-year period, identifying predictors of student success, and using these predictors to develop a recommendation system to assist at-risk students. The data in the study consists of student demographic and academic records from 2003-2017. Simple exploratory data analysis is used to highlight changes in student performance over time. Next, a deeper analysis is performed by training three classifiers - logistic regression with L1 penalty, logistic regression with L2 penalty, and a random forest model - to predict the probability that students will graduate. Finally, the predictions of each classifier are calibrated and combined to form a robust recommendation system which can be used to alert advisers when a student is struggling.

Acknowledgements

I would like to thank Dr. Sunil Shende for his continued patience, support, and encouragement throughout the course of this study. His willingness to give his time so generously has been very much appreciated.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Higher education institutions cater to diverse groups of students with a great variety of interests and backgrounds.The complex interplay between student behaviors, societal and economic trends, and other external influences can make it very difficult for faculty and administrators to track measures of student success. Within the last two decades, this difficulty has been compounded by rapid changes in public perception and demand for higher education services, particularly those in science, technology, engineering, and mathematics (STEM) fields. The situation becomes even more confounding in light of recent shifts in student population demographics and an overall decrease in public funding that could be allocated for administrative support. Consequently, faculty and staff have been turning to data analytics to assist with the increasingly daunting task of assessing student performance and, ultimately, guiding students toward academic success.

1.1 Design Summary

This paper proposes a machine learning approach to improving student outcomes, with the goal of developing administrative tools that can quickly and accurately identify predictors of student success, and provide an early warning system for advisers serving large numbers of students. The proposed system employs a suite of machine learning classifiers to predict the probability that a student will graduate, given current demographic and academic records. The probability is reported along with a simple red-orange-yellow-green categorization scheme to facilitate ease of use on the administrative end, and timely interventions for students in need.

The remainder of the paper will systematically break down the construction of this system, spanning the following topics:

- Data processing: including data cleaning, data structuring, feature extraction, and descriptive statistics generation
- Experimental Design: including classifier descriptions, and model selection, evaluation, and reliability
- Results: including interpretations of classifier output and reliability, and combining classifiers for robust recommendations
- Conclusion: including a high-level overview of the results, suggestions for future studies, ethical considerations, and closing remarks

Chapter 2

Data Processing

The data used in this study was provided by Rutgers University-Camden, and contains anonymous demographic and academic records for all students who have attended the university during the 15-year period between 2003 and 2017; the total number of students is 11,834. All processing and subsequent analysis is conducted via Python, version 2.7.13, inside a sequence of Jupyter ipython notebook environments. The interactive and inherently structured nature of ipython notebooks allows for immediate processing validation, which is invaluable when working with complex data, and lends to the creation organized, readable code. The Pandas library is used for data access, management, table generation, and descriptive statistics; it is an excellent tool for intuitive and efficient database-style operations. Finally, the scikit-learn machine learning library is used for classifier training and evaluation.

2.1 Processing Structure

Data processing is split among many modules (ipython notebooks), with each module serving a specific purpose. Modules are organized into several stages, with each stage preparing the data for the next. Referring to figure 2.1, each processing stage is represented as a **row** in the diagram:

- Row 1: Data management for extraction and subsequent exploratory analysis
- Row 2: Data refinement for classifier use
- Row 3: Model selection
- Row 4: Model evaluation

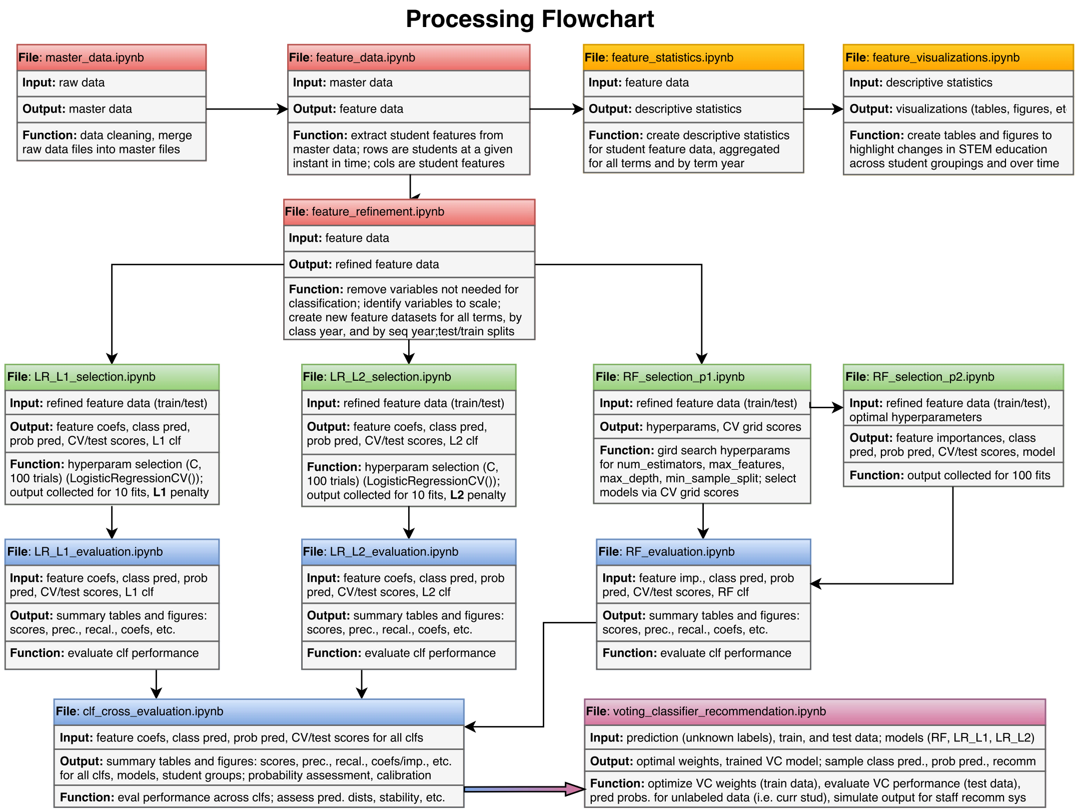

Figure 2.1: Processing Flowchart

- Row 5: Final output

The processing modules shown in figure 2.1 are also grouped by **color** according to their function, with the primary processing functions being data manipulation (red), descriptive statistic generation (yellow), model selection (green), model evaluation (blue), and recommendation system construction (violet).

2.2 Data Descriptions

For the purposes of this study, the data is organized according to two categories: **models** and **student groups**. A **model** refers to a distinct population described by a specialized dataset that is used to train a classifier. More precisely, each model attempts to capture the behavior of students at different temporal stages in their academic career. The models of interest in this study are designated as **aggregate** (all students, all terms), **freshman** (freshman students, any term), **sophomores**, **juniors**, **seniors**,

seqYear1 (students in their first year at Rutgers), **seqYear2**, **seqYear3**, **seqYear4**, **seqYear5**, and **seqYear6**.

A **student group** refers to a sub-population of interest within each model. The student groups of interest in this study are **all students** (which make up the original models) and, of course, **STEM students**. Each combination of model and student group is used to train three different classifiers; this results in 22 models, each with three different classifiers capable of making predictions. The (**model**,**student group**) combinations are highlighted below:

- **aggregate**: (all students, STEM students)
- **freshman**: (all freshman, STEM freshman)
- **sophomores**: (all sophomores, STEM sophomores)
- **juniors**: (all juniors, STEM juniors)
- **seniors**: (all seniors, STEM seniors)
- **seqYear1**: (all seqYear1, STEM seqYear1)
- **seqYear2**: (all seqYear2, STEM seqYear2)
- **seqYear3**: (all seqYear3, STEM seqYear3)
- **seqYear4**: (all seqYear4, STEM seqYear4)
- **seqYear5**: (all seqYear5, STEM seqYear5)
- **seqYear6**: (all seqYear6, STEM seqYear6)

The sections below describe the data at each stage of processing, moving from unfiltered raw data to the (model, student group) specific data that is ready to be used in classification.

2.2.1 Raw Data

Raw data is data that was obtained directly from Rutgers University Camden. There are four raw data files:

- precollege.csv: aggregate data, including ethnicity and high school records
  - key = studyid
- degreedata.csv: aggregate data, including graduation status
  - key = studyid
- termdata.csv: semester based data, including credit counts, GPA, major designation, STEM designation, etc.
  - key = [studyid,semester]
- coursedata.csv: course based data, including course name/number, credits attempted/earned, grades, etc
  - key = [studyid,semester,course]

Operations performed on the raw data include null removal, renaming columns, dummy encoding of categorical variables, time-frame selection, preliminary feature creation (including current age, number of years attended Rutgers, number of major switches total, number of major switches between STEM and non-STEM majors), stratification of graduation status to also include a "current student" designation, and extraction of course codes and department codes. Finally, the four raw datasets are merged into one large "master" set containing all records and all features for every student in the study period (see next section).

Note on graduation status and current students: Current students are defined as those students who have not graduated but completed courses in the most recent semester. These students were **not** included in the train/test sets due to the possible introduction of contradictory information. To be included, these students must be designated as "not graduate" regardless of their actual academic performance. This is likely to negatively impact the ability of the classifier to learn meaningful features.

2.2.2 Master Data

The master data, as mentioned above, contains all records and all features for every student in the study period. The master data is the source of all feature engineering, feature extraction, and partial data partitioning for classification. Operations performed on the master data include defining student and feature identifiers for data partitioning (STEM/non-STEM student ids, first-year/transfer student ids, STEM/non-STEM major codes, course groupings (L100, L200, intro, lab, etc.), time delineations (intervals, class years, sequence years, semesters), demographic feature groups, academic feature groups), and 2-D course grade encoding (see subsection below)), feature engineering (GPA's (major courses, STEM/non-STEM courses, L100 courses, etc), feature extraction (including aggregation and reformatting), and partial data partitioning for classification. Final processing on the master data results in four partially partitioned feature datasets that are ready for descriptive statistical analysis and final refinement for classification.

2-D Course Grade Encoding

Encoding categorical course grades (A,B,C, etc.) numerically forces a decision between a 1-D ordinal system (i.e. A=5, B=4, C=3, etc.) or a dummy encoding (course1-A, course1-B, course1-C, etc.). A 1-D ordinal system imposes a ranking on students who **did not take a course** by forcing a grade to be entered, usually the mean. According to the classifier, this implies that students who did not take a given course are more similar (spatially) to students who earned grades close to the mean than they are to students who earned grades far from the mean; this is not necessarily true and can mislead the classifier. The other conventional option, a dummy encoding of grades for each course, may not mislead the classifier in the same sense, but will introduce thousands of sparse features to the model (there are 3000+ courses total), likely deteriorating classification quality.

As a solution, this study proposes and employs a 2-D course grade encoding system centered on a unit circle as shown in figure 2.2 above. Each course is represented by

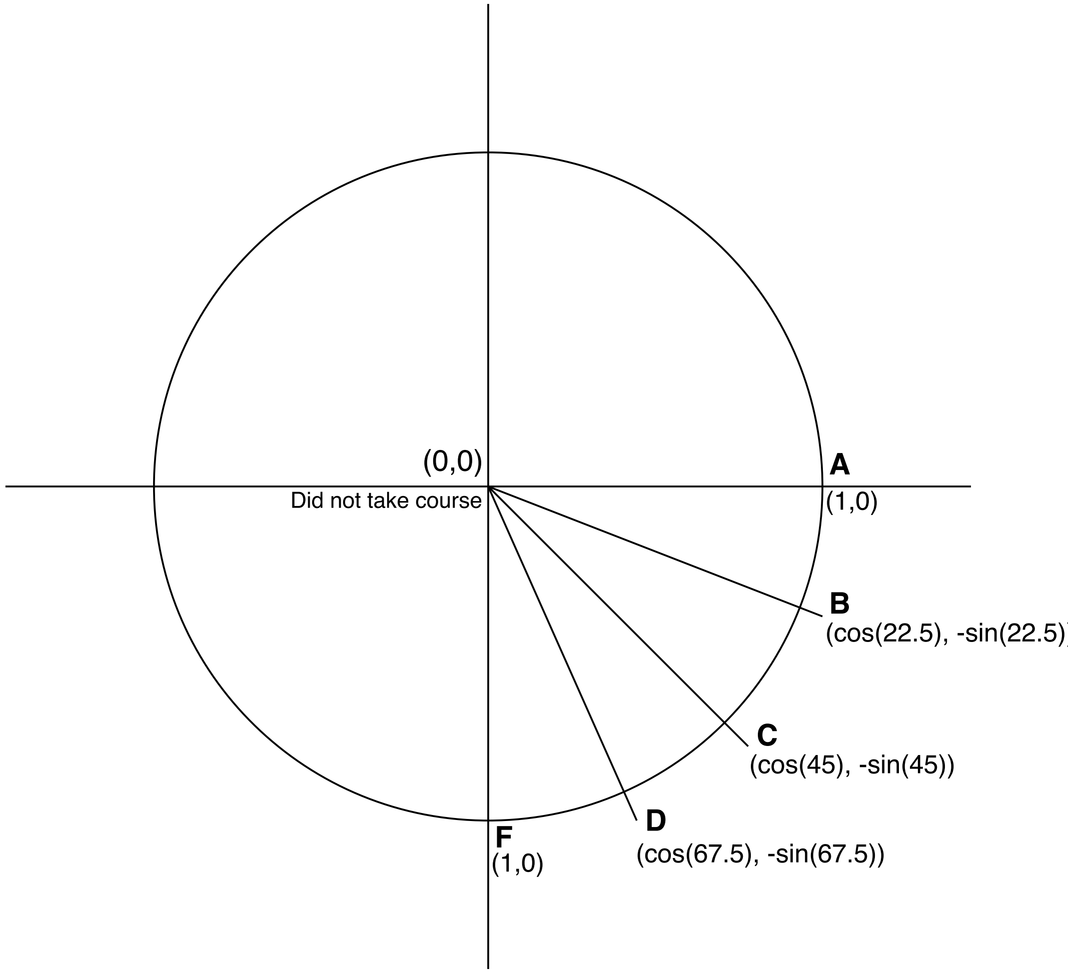

Figure 2.2: Two Dimensional Grade Encoding

two features, *'course-x'* and *'course-y'*, representing coordinated on the unit circle. The coordinate for the center of the circle, *(0,0)*, represent students who did not take the course, and combinations of coordinates on the boundary of the fourth quadrant of the circle represent possible grades. In this scheme, each possible grade has **equal** distance (similarity) from the center of the circle, implying that students who did not take the course are no more or less similar than students who did, regardless of the grade they earned. Furthermore, distances between grades on the boundary of the circle still capture an ordinal grading scale (i.e. 'A' is closer to 'B' than 'C', 'A' is closer to 'C' than 'D', etc.), thus allowing for accurate comparisons among students who did take the course. Further still, by restricting encoded values to the fourth quadrant, we ensure that increases in either of the 'course-x' or 'course-y' feature values always correspond to higher grades. (Note: the original 2-D grade encoding mapped letter grades to quadrants 1 and 4, rather than only quadrant 4, resulting in ambiguous model coefficient interpretations)

2.2.3 Feature Data

As mentioned above, the feature datasets contain all the desired features for descriptive statistical analysis and are appropriately formatted for final refinement and partitioning for classification. There are four feature datasets:

- features-studyid.csv: aggregate data for all students included in the study - each row represents a distinct student and each column represents a student feature (aggregated for all terms)
  - key = studyid
- features-studyidYear.csv: data by term year - each row represents a distinct student in a distinct term year and each column represents a student feature (aggregated for the current term year)
  - key = [studyid,term year]
- features-studyidClass.csv: data by class year - each row represents a distinct student in a distinct class year and each column represents a student feature (aggregated for the current class year)
  - key = [studyid,class year]
- features-studyidSemester.csv: data by semester - each row represents a distinct student in a distinct semester and each column represents a student feature for the current semester
  - key = [studyid,semester] (Note: courses are listed as individual columns in the feature datasets and are not needed as part of the key)

Table 2.1 below shows the dimensions of the feature datasets.

Table 2.1: Feature Data Dimensions

```
features_studyid_master: rows=11834 || columns=6909
features_studyidYear_master: rows=44329 || columns=6943
features_studyidClass_master: rows=27152 || columns=6943
features_studyidSemester_master: rows=85177 || columns=7007
```

2.2.4 Feature Statistics

Feature statistics are derived from 31 demographic and academic features present in the feature data, and organized into a series of indexed tables. The index of each table corresponds to one of **21** distinct **groupings**(coded as g1-g21), each defined over two distinct **time delineations** (coded as t1 and t2), resulting in a total of 42 tables. Groupings and time delineations for descriptive statistics are defined in table 2.2 below:

Table 2.2: Descriptive Statistics Table Format Guide

- **Time delineations:**
  - t1: aggregate data for all terms
  - t2: aggregate data by year (2003-2017)
- **Groupings:** Within each time delineation, tables are grouped according to various combinations of base groups (g1-g6).
  - g1: student group
    - all students, STEM students, non-STEM students, transfer students, first-year students
  - g2: graduation status
    - graduated, current student, did not graduate
  - g3: ethnicity
    - White, Asian, Black/African American, Hispanic, Unknown, Other/Two or More, Native Hawaiian/Pacific Islander, American Indian or Alaskan Native
  - g4: gender
    - male, female
  - g5: class year
    - freshman, sophomore, junior, senior
  - g6: major (STEM only)
    - Biology, Biochemistry, Biology: Computational and Integrative, Chemistry, Computer Science, Mathematics, Physics
  - g7: [student group, graduation status]
  - g8: [student group, ethnicity]
  - g9: [student group, gender]
  - g10: [student group, graduation status, ethnicity]
  - g11: [student group, graduation status, gender]
  - g12: [student group, ethnicity, gender]
  - g13: [student group, graduation status, ethnicity, gender]
  - g14: [student group, class year]
  - g15: [student group, class year, graduation status]
  - g16: [student group, class year, ethnicity]
  - g17: [student group, class year, gender]
  - g18: [major, graduation status]
  - g19: [major, ethnicity]
  - g20: [major, gender]
  - g21: [major, class]

2.2.5 Refined Feature Data

The refined feature data represents the last stage of data processing before classification begins. Operations performed on the refined feature data include renaming columns, imputing missing values (transfer status, credit counts, GPA values), optimizing encoding schemes, removing sparsely populated columns (high school data, GPAs for level 500 and 600 courses, etc.), removing columns that leak future information into the model (see note below), identifying variables to standardize, and final partitioning of the feature data according to the (**model**,**student group**) structure referenced in section 2.2 above. This final processing stage results in eleven refined feature datasets ready to enter the classification pipeline. They are as follows:

- features-studyid-refined
- features-studyidFresh-refined
- features-studyidSoph-refined
- features-studyidJunior-refined
- features-studyidSenior-refined
- features-studyidSeq1-refined
- features-studyidSeq2-refined
- features-studyidSeq3-refined
- features-studyidSeq4-refined
- features-studyidSeq5-refined
- features-studyidSeq6-refined

Chapter 3

Experimental Design

The goal of the experimental design is to develop a method of prediction that is reliable under stable environmental conditions, but also robust to fluctuations in predictor behavior. All classifiers have strengths and vulnerabilities that can be related to their underlying mathematical or procedural foundations, and it is well known that individual classifier performance can vary greatly depending on the nature of the data being used and the sources of variation inherent in the process being modeled. Higher education presents a very complex and dynamic environment that typically produces noisy, high-dimensional, and often sparsely populated data. A single classifier, even a modern ensemble method meant to handle such situations, will struggle to produce consistently accurate predictions under all possible circumstances.

To cope with the complexities of the higher education landscape three distinct classifiers are optimized and trained to predict the probability of student graduation: logistic regression with L1 penalty, logistic regression with L2 penalty, and a random forest ensemble classifier. The classifiers selected attempt to spread the "risk" of poor prediction by minimizing the overlap of vulnerabilities inherent in each classifier, under a variety of unfavorable conditions. Combining the output of each classifier creates a considerably more stable ensemble method of prediction in which the risk of incorrect classification is minimized and often able to be anticipated. This level of stability must be interpreted not only as preferable, but required for the ethical implementation of recommendation systems that may lead to direct student intervention.

The sections that follow outline the general preprocessing, selection, and evaluation procedures for each model (distinguished by classifier when necessary), along with a brief overview of classifier.

3.1 Preprocessing

The datasets corresponding to each model follow the same preprocessing routine. First, a copy of the data is created to prevent corruption of the original set. All rows corresponding to current students are removed and the feature to be classified, graduation status, is extracted for use as the 'dependent variable'. The 'X' and 'Y' data is then jointly divided at random into a training set and a validation set; the training set is used to fit the model and the validation set is used to evaluate performance. Features in the training set are mean centered and scaled by their standard deviations; features in the validation set are also mean-centered and scaled, but by the corresponding **training set** metrics, such that we can evaluate model performance on, theoretically, "unseen" data.

3.2 Classifiers

For each classifier, model selection entails the optimization of the model *hyperparameters*, or those parameters set manually and not determined by the model itself. Different strategies for hyperparameter optimization are employed for each classifier according to classifier behavior

Model evaluation is generally the same for each classifier, and focuses on model accuracy (in both training and validation), the distribution of model predictions, and the identification of top features.

3.2.1 Logistic Regression - L1 Penalty

Logistic regression is an obvious baseline choice in most binary classification problems, especially when we are seeking a 'soft', or probability based classification. Logistic regression estimates the odds outcome of the dependent variable *given exposure to a set of quantitative independent variables*; this is known as the *odds ratio*.

In the context of this study, the dependent variable is student graduation status, and the "set of quantitative independent variables" are the set of features describing each student. The *odds* of a binary dependent variable are defined as (probability of success)/(probability of failure), or, in other words, (probability student graduates)/(probability student does not graduate). The *odds ratio* takes this a step further, and is calculated as (odds student graduates *given the set of student features*)/(odds student does not graduate *given the set of student features*). Coefficients of logistic regression output are interpreted as the log of the odds ratio and can be exponentiated to retrieve the odds ratio itself.

The "L1 Penalty" in logistic regression is a regularization term based on the L1, or "Manhattan distance", that has the property of pushing model coefficients to 0. This is particularly useful in high-dimensional datasets that include many sparse or noisy features and thus is appealing for this application (there are 6000+ features following 2-D course encoding)

Model Selection

Feature elimination is intrinsic to logistic regression with L1 penalty and as such, no feature reduction strategy need be employed. However, the hyperparameter, C, also controls the "freedom" of the model, with smaller values of C constraining the model to fewer non-zero coefficients.

LogisticRegressionCV(), a cross validation strategy built into scikit learn, is employed to select the optimal C via k-fold cross validation on 100 different values for C between 0.001 and 1000. Once the optimal value for C is identified, the model is fit() an additional 10 times, with CVscores (on the training data), validation scores, class predictions, probability predictions, and coefficient values compiled after each iteration for future evaluation.

3.2.2 Logistic Regression - L2 Penalty

The logistic regression with L2 Penalty is interpreted in the same fashion as the L1 variant. However, the L2 regularization term does not have the same "feature eliminating" properties inherent to L1 regularization. As such, a recursive feature elimination strategy is employed with nested cross validation for the optimal C value.

Model Selection

RFE(), a recursive feature elimination strategy built into scikit learn, is employed to recursively eliminate model features according to a user defined step size, 'step', and stopping criteria, 'n-features-to-select'. In each iteration, the model is fit() on the current 'n' features, the 'step' least important features are removed, and the process repeats until n-features-to-select is acheived. This process has been further optimized by conducting a cross validated search for the optimal C value, via LogisticRegressionCV(), *within each iteration of the RFE().* This ensures optimal feature pruning in each step.

3.2.3 Random Forest Classifier

The random forest classifier is itself an ensemble method of classification that combines the predictions of many weak learners, in this case individual decision trees, to increase overall prediction accuracy. Random forest models can perform well with sparse, high-dimensional data, even in the presence of nonlinear relationships between predictors and the dependent variable. Employing a large number of weak learners also reduces the issue of over-fitting, which is common in simple decision tree learning.

However, the over-fitting problem still exists, especially in the presence of noisy data; the random nature of feature selection for each underlying decision tree means that the model can incorrectly identify noise as a signal, attributing importance to meaningless features. On that note, it is also difficult to interpret the meaning of "feature importances" with very high dimensional data due to the low probability that any given feature, significant or not, will be selected frequently enough to strongly influence to model. Furthermore, "feature importances" are strictly positive, limiting the sophistication of their interpretation.

Model Selection

The random forest classifier model selection is split into two stages: one solely for grid search hyperparameter estimation, and one for repeated, cross-validated fitting to gather feature importances and other prediction for analysis. (Note that while overall prediction quality stays relatively constant through each fit() of the model, feature importances can vary widely with each iteration)

The first stage of model selection employs GridSearchCV(), a built in scikit learn function to perform an exhaustive search on a user defined grid of hyperparamters. The random forest classifier has a large number of parameters compared to the single C value for logistic regression, thus necessitating a more thorough search of the parameter space. The hyperparameters of interest are:

- n-estimators: this is the number of trees in the forest
- max-features: the number of features to randomly select from the model
- max-depth: the maximum permitted depth of each decision tree learner in the forest - large depths can improve prediction quality but increase the risk of capturing noise and thus over-fitting
- min-samples-split: the minimum number of samples required for decision tree to branch

The second stage of model selection, as mentioned above, performed repeated, cross-validated fitting of the model (100 trials) to attempt to extract truly important features.

3.2.4 Voting Classifier

The voting classifier is the final stage in the experimental design. It calculates a weighted average of the predicted probabilities for each associated classifier to calculate an overall probability and final classification for each student.

Multiple models are applicable to individual students based on their current class and STEM designation. For example, a freshman STEM student can be used as input for the aggregate model for all students, the aggregate model for STEM students, the freshman model for all students, and the freshman model for STEM students. A custom function was written to identify all applicable models for each student, and report the weighted average of the probabilities from each classifier *for each model*, along with an alert level corresponding to each probability value. Referring again to the example above, the freshman STEM student would have four weighted probabilities and associated alert levels for each applicable model.

Combining the results of multiple ensemble outputs further increases the reliability and robustness of the system.

Weight Optimization

As mentioned above, the probability output of the voting classifier is a weighted average (with default weights of 1 for each classifier). A custom, brute force search was implemented to identify the optimal weights in terms of overall prediction accuracy

3.3 Model Evaluation

The quality of each model is evaluated in context of accuracy, prediction distributions, and top features.

3.3.1 Accuracy

Accuracy is reported by the mean CVscores and test scores calculated during successive fits with the optimal value for C.

3.3.2 Prediction Distributions

Prediction distributions are evaluated using the following methods:

- Confusion matrices: color coded display of the distribution of true positives (tp), false positives (fp), true negatives (tn), and false negatives (fn); the tn and tp values are shown along the main diagonal, with the depth of color indicating the frequency.

- Precision: tp/(tp+fp) - higher values of precision correspond to smaller numbers of false positives

- Recall: tp/(tp+fn) - higher values of recall correspond to smaller numbers of false negatives
- Decision Thresholds: The probability value at which a classifier is forced to make a decision for classification. The default threshold is 0.50. Altering the decision threshold will alter the number of tp,fp,tn, and fn, sometimes in a non-intuitive fashion:
  - a model with many false positives may benefit from a higher threshold by forcing larger numbers of negative classifications. However, this may also reduce the number of true positives and, consequently, increasing the number false negatives
- Precision-Recall Curves: a figure showing the trade-off between precision and recall for different decision thresholds. If a particular metric, say precision, is valued higher than recall, this chart demonstrates the cost in lost recall to increase the model precision.
  - A well performing model displays a precision-recall curve that maintains a precision close to 1.0 as recall increases. Poor performing models will see sharp declines in precision for modest increases in recall.
- ROC curves: a figure showing the trade-off between true positive rates and false positive rates. Again, this can be interpreted as the cost in additional false positives to increase the number of true positives
  - A well performing model displays a ROC curve that shows sharply increases in y (true positives) for small increases in x (false positives), indicating the model yields high numbers of true positives (correct classifications) for low values of false positives. A poor performing model with show gradual increases in y as increases.

3.3.3 Top Features

Top Features are analyzed to attempt to identify strong predictors of student performance that may be used to enhance early warning systems, influence course recommendations, course creation, or university policies.

3.4 Recommendation System

As mentioned in the voting classifier section, multiple classifiers contribute to the weighted probability average for each model, and multiple models can make classification predictions for individual students. Each applicable model reports a weighted probability average and an associated alert level. Alerts are based on a red-orange-yellow-green system and are intended to add a layer of caution to recommendations in the presence of imperfect classification predictions due to false positives and false negatives. The colors in the system are defined as follows:

- GREEN: OK - probability of graduation ¿ 0.75
- Yellow: WATCH - 0.5 ¡ probability of graduation ¡= 0.75
- ORANGE: MEET - 0.25 ¡ probability of graduation ¡= 0.5
- RED: INTERVENE - 0.0 ¡ probability of graduation ¡= 0.25

Chapter 4

Results

Selected results of descriptive statistics analysis and classifier evaluation are shown below. Additional tables and figures can be found in the included appendices.

4.1 Descriptive Statistics

Descriptive statistics are reported in tabular and graphical format, displaying aggregate and yearly summary data.

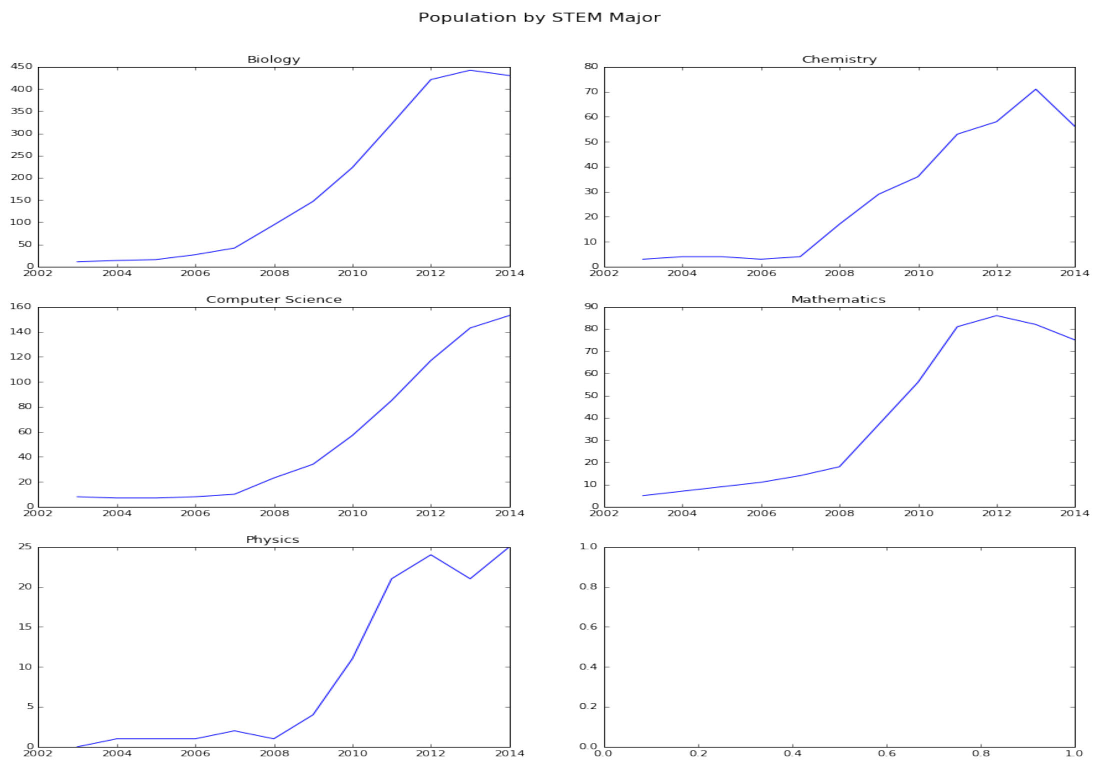

Figure 4.1: STEM Population by Major, by Year

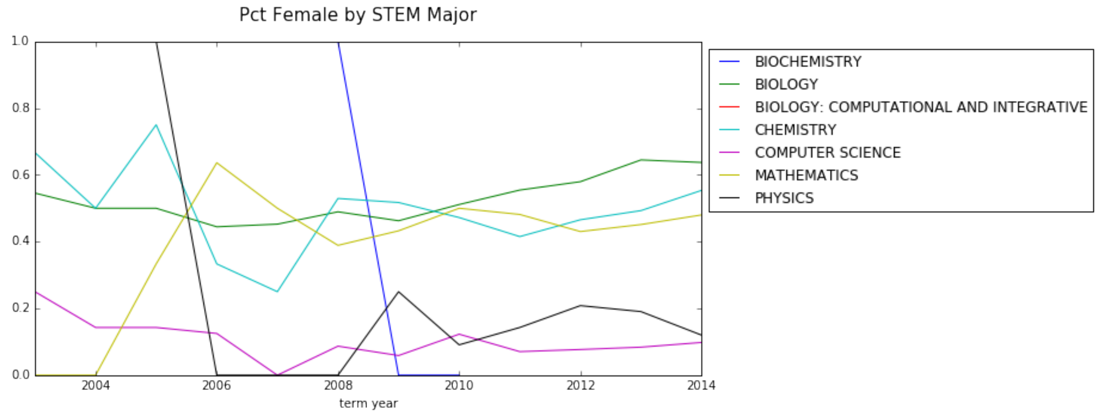

Figure 4.2: Female STEM Population by Major, by Year

Figure 4.3: STEM Graduation Rates by Major, by Year

4.2 Classifiers

Selected results for each classifier, comparisons across classifiers, and sample recommendation system output can be found below. Classifier results are partitioned by accuracy, analysis of prediction distributions, and overview of top features for each model.

4.2.1 Logistic Regression - L1 Penalty

Accuracy

Classifier accuracy is measured in terms of the number of correct classifications on the validation set.

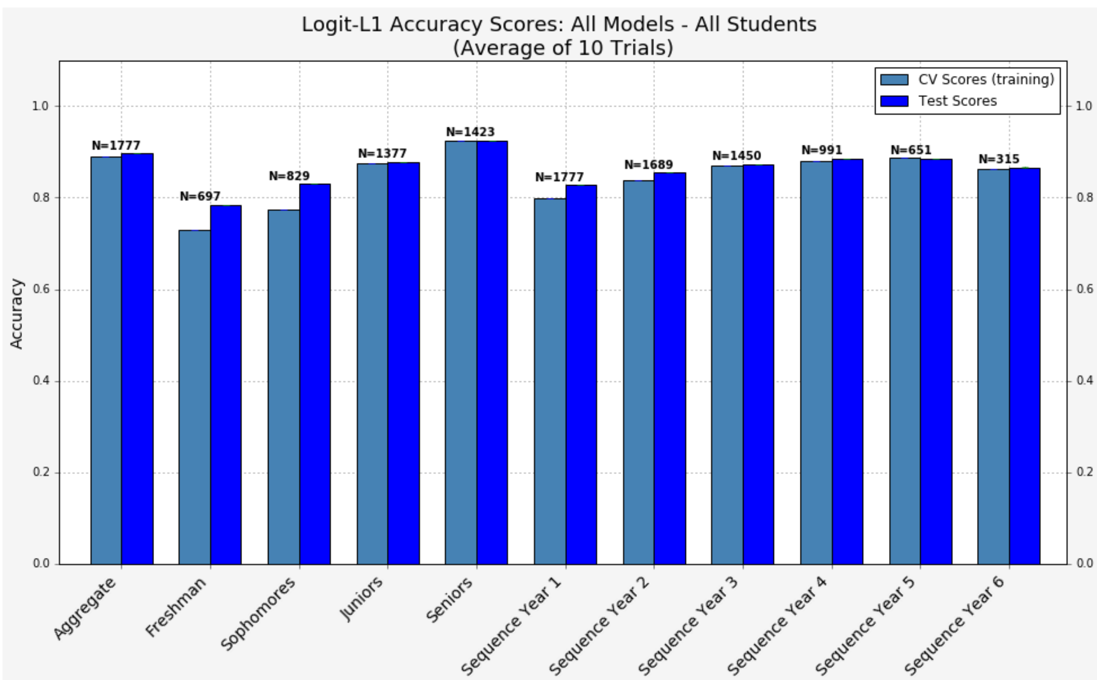

Figure 4.4: Logit-L1 Accuracy Scores: All Models, All Students

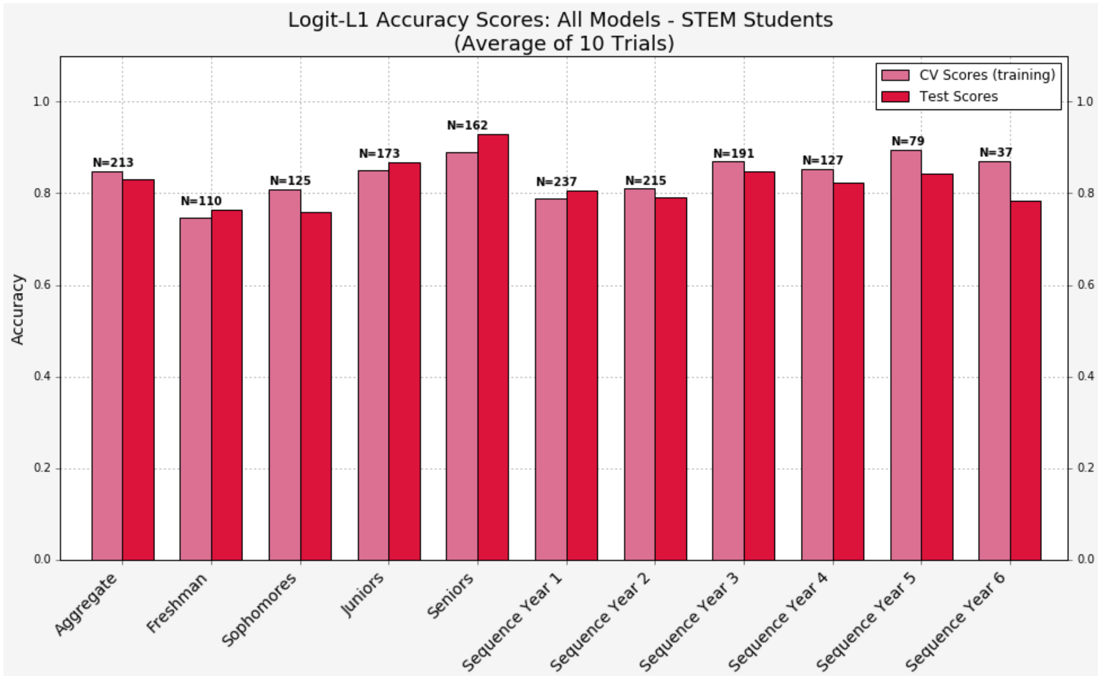

Figure 4.5: Logit-L1 Accuracy Scores: All Models, STEM Students

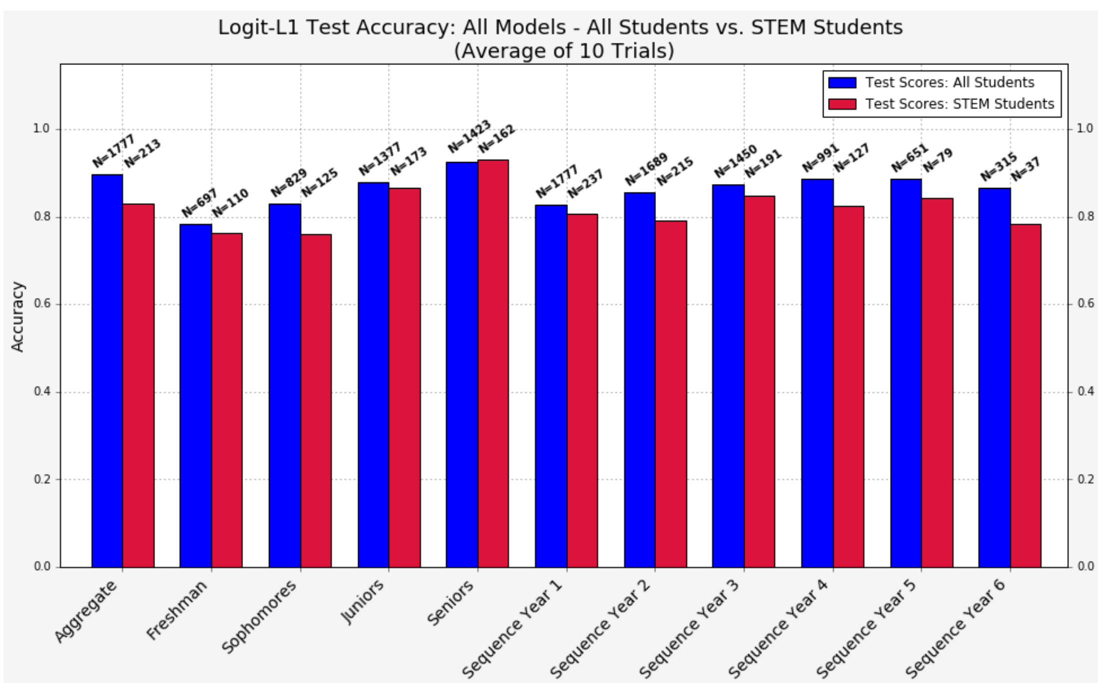

Figure 4.6: Logit-L1 Accuracy Scores: All Models, All Students vs STEM Students

Prediction Distributions

Prediction distributions are analyzed via confusion matrices, precision, recall, F-scores, precision-recal curves, and ROC curves.

Confusion Matrices:

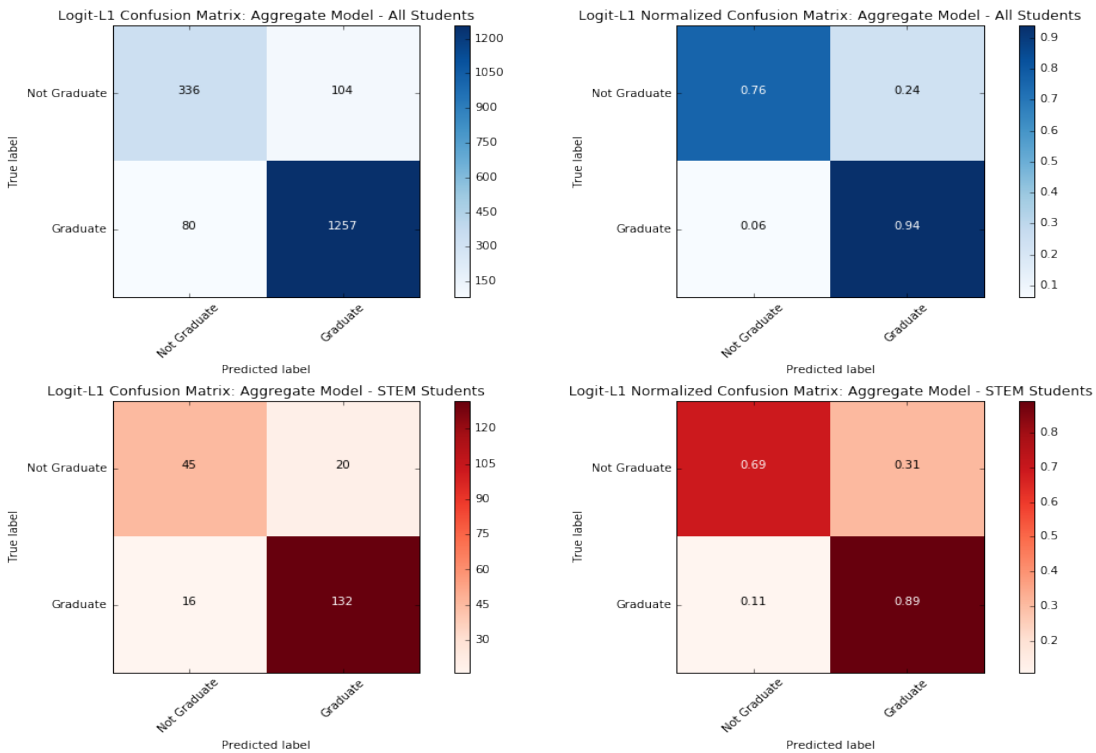

Figure 4.7: Logit-L1 Confusion Matrices: Aggregate Model

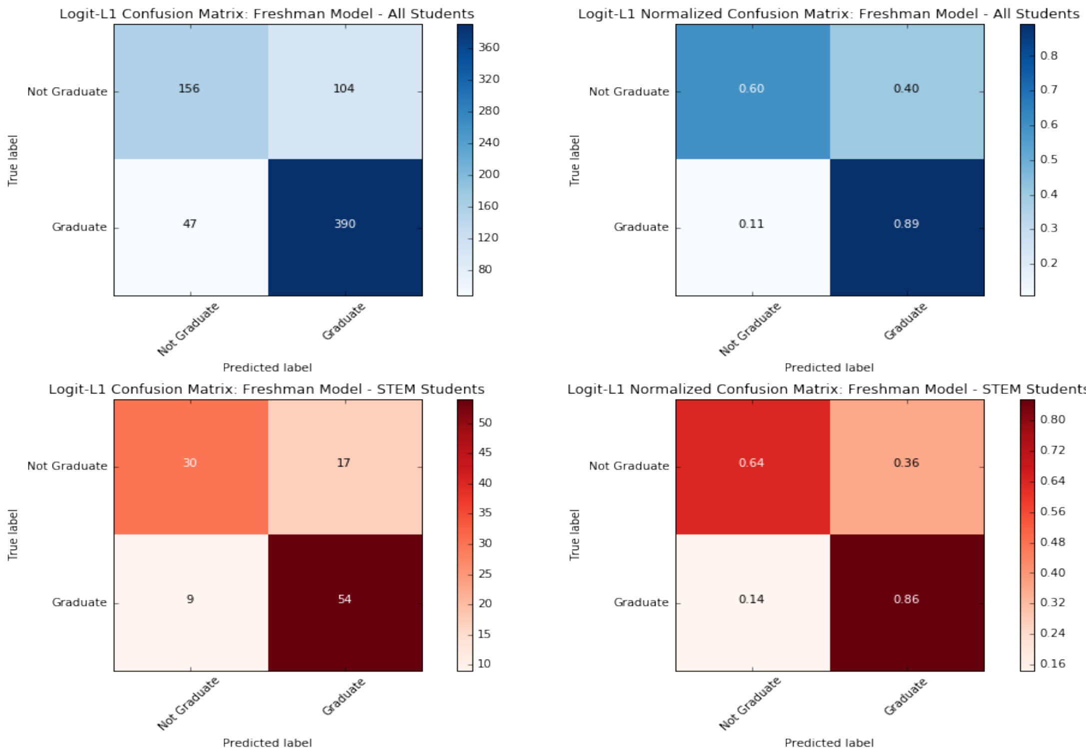

Figure 4.8: Logit-L1 Confusion Matrices: Freshman Model

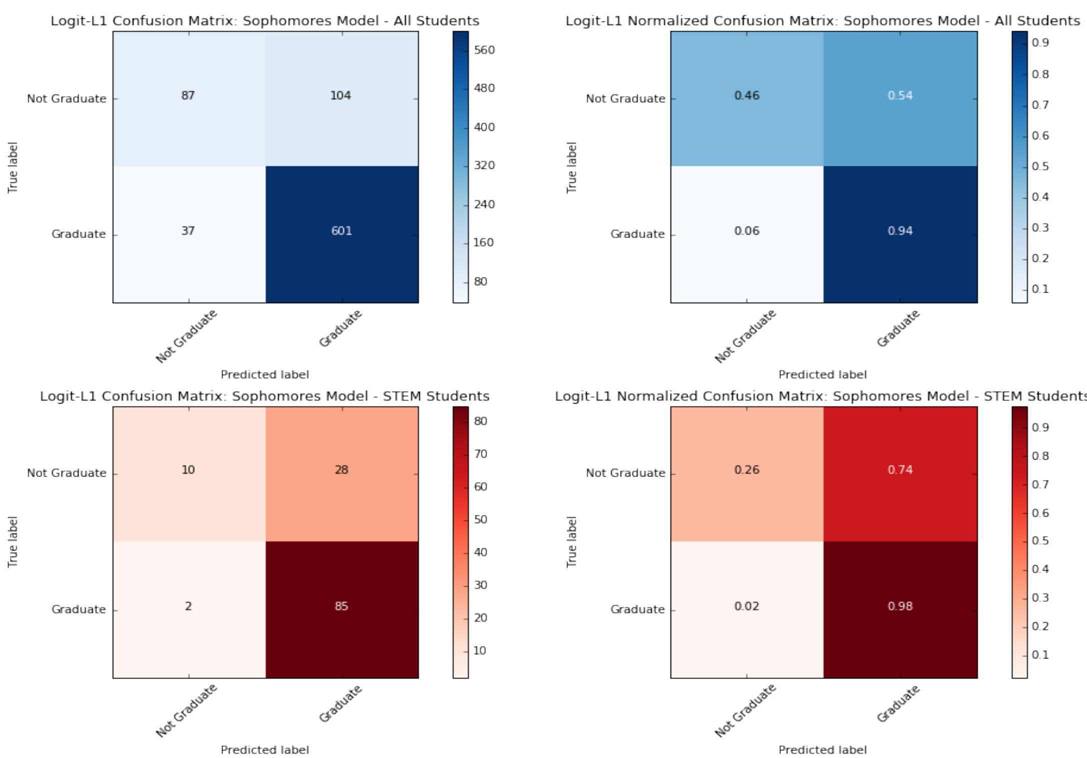

Figure 4.9: Logit-L1 Confusion Matrices: Sophomore Model

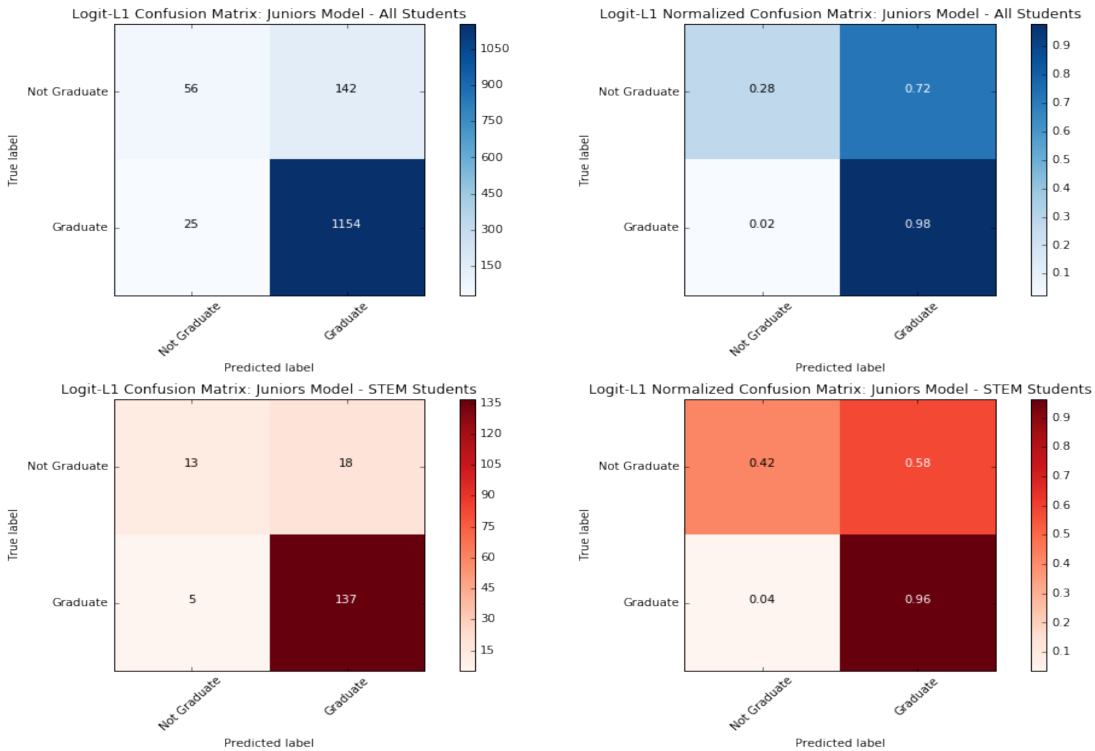

Figure 4.10: Logit-L1 Confusion Matrices: Junior Model

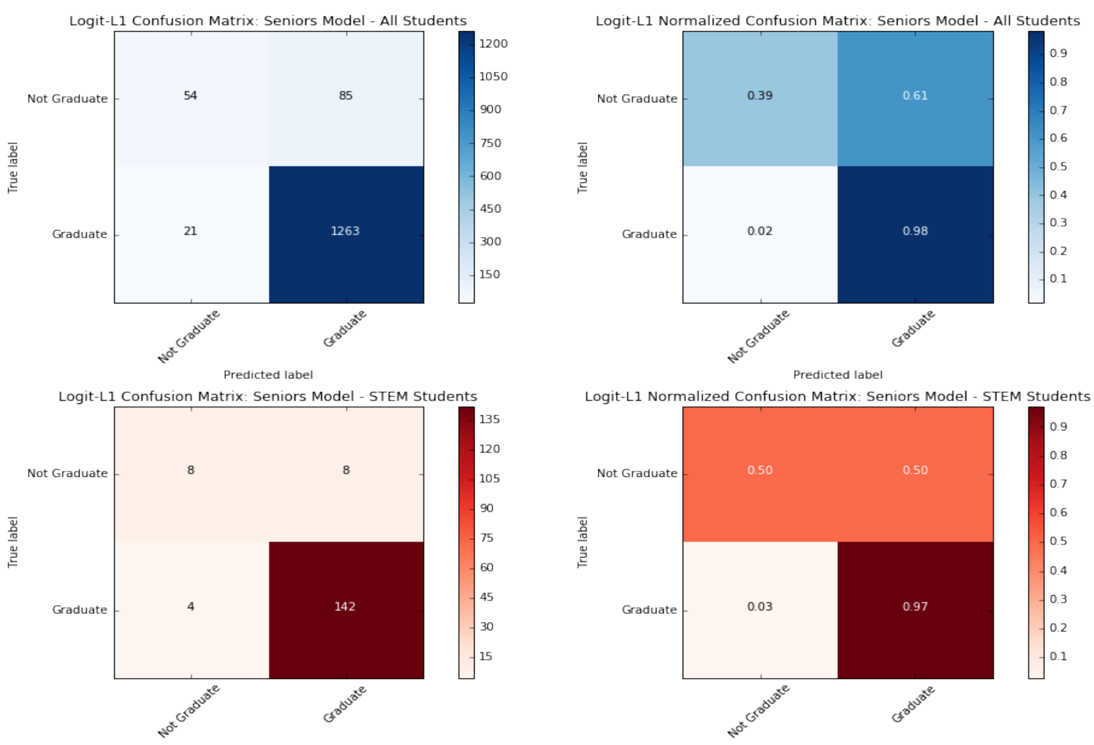

Figure 4.11: Logit-L1 Confusion Matrices: Senior Model

Precision, Recall, F-Scores

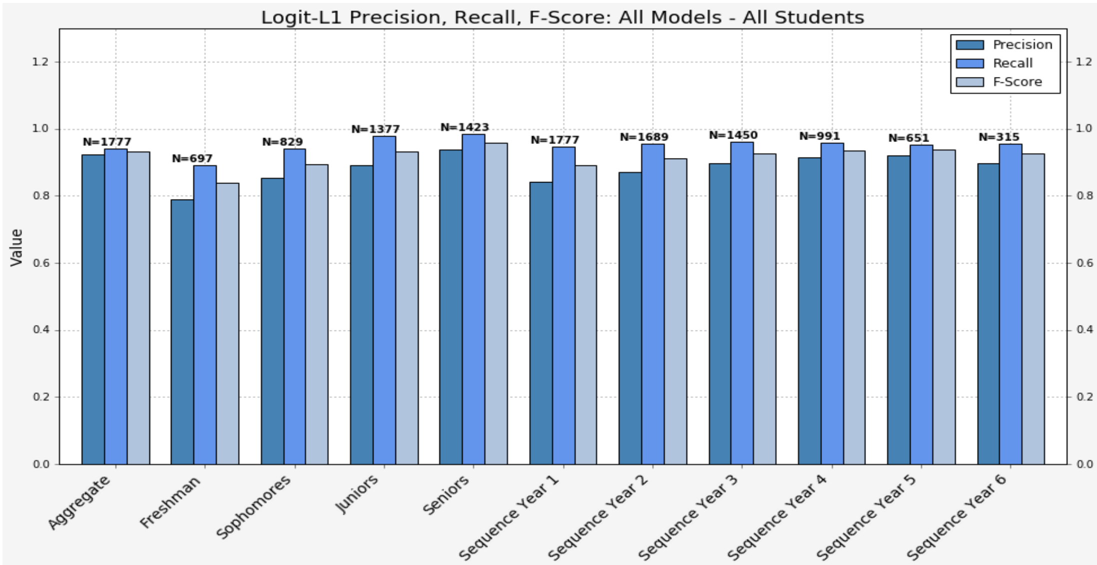

Figure 4.12: Logit-L1 Precision, Recall, and F-scores: All Models, All Students

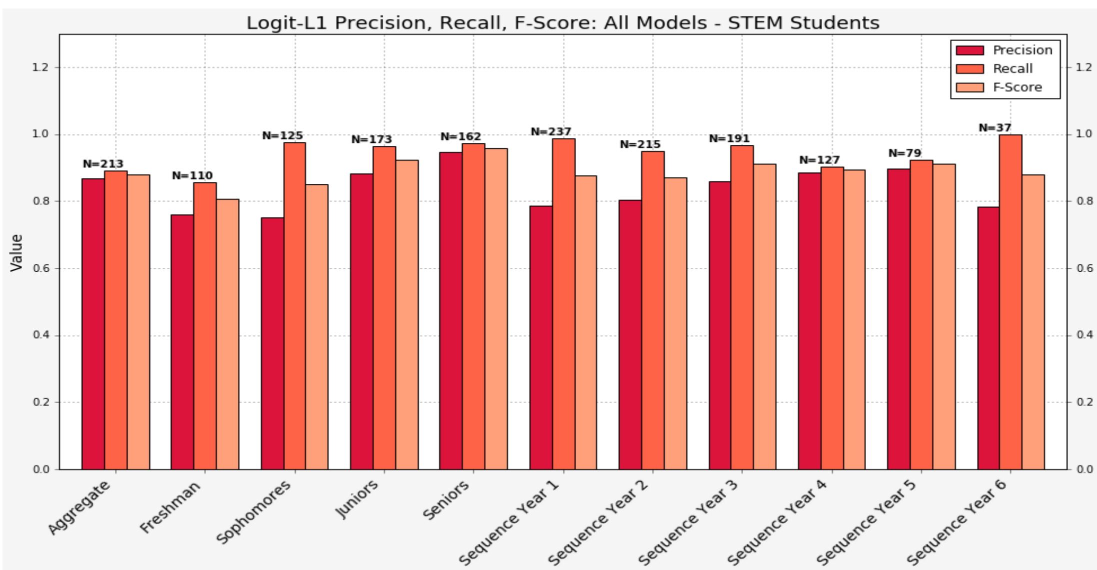

Figure 4.13: Logit-L1 Precision, Recall, and F-scores: All Models, STEM Students

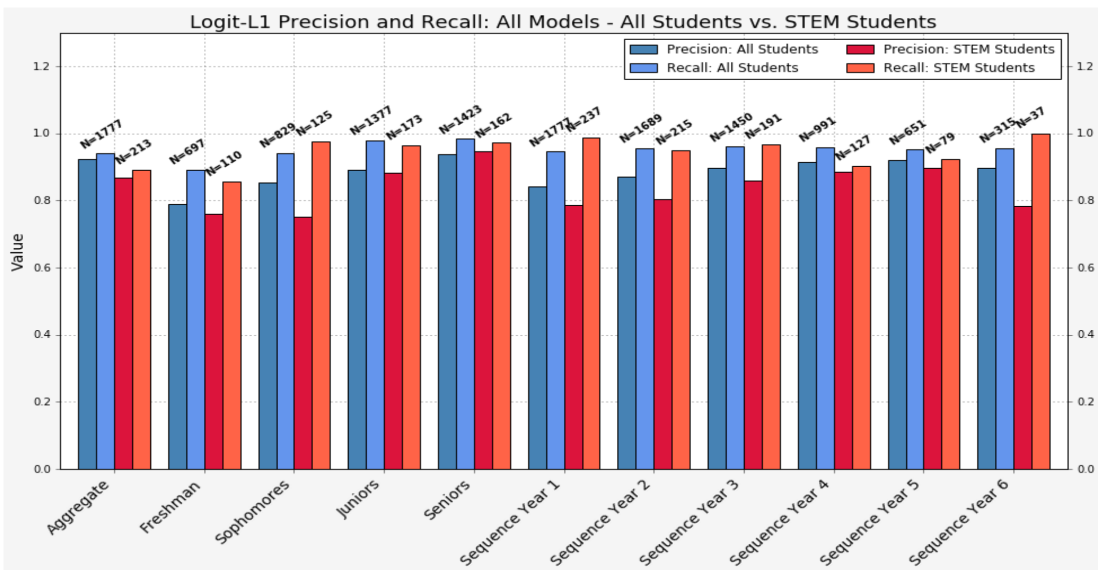

Figure 4.14: Logit-L1 Precision and Recall: All Models, All Students vs STEM Students

Precision-Recall Curves

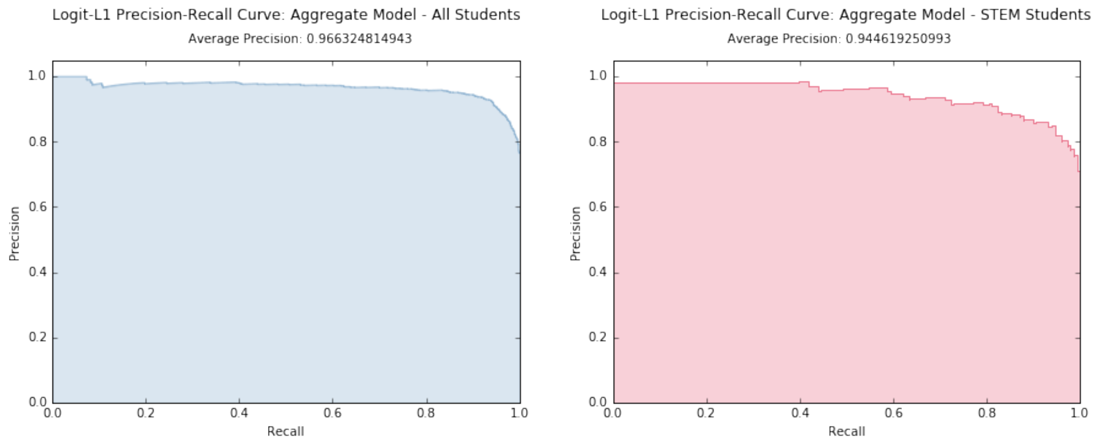

Figure 4.15: Precision-Recall Curve: Aggregate Model

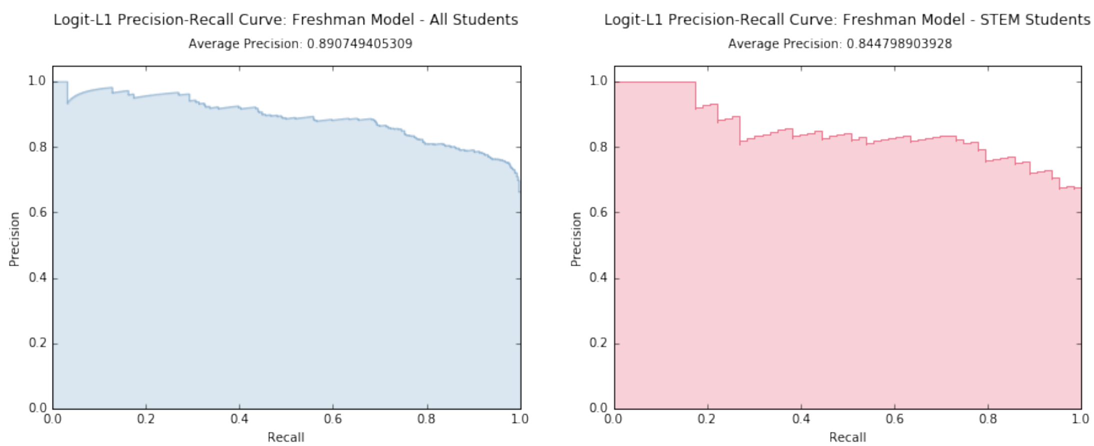

Figure 4.16: Precision-Recall Curve: Freshman Model

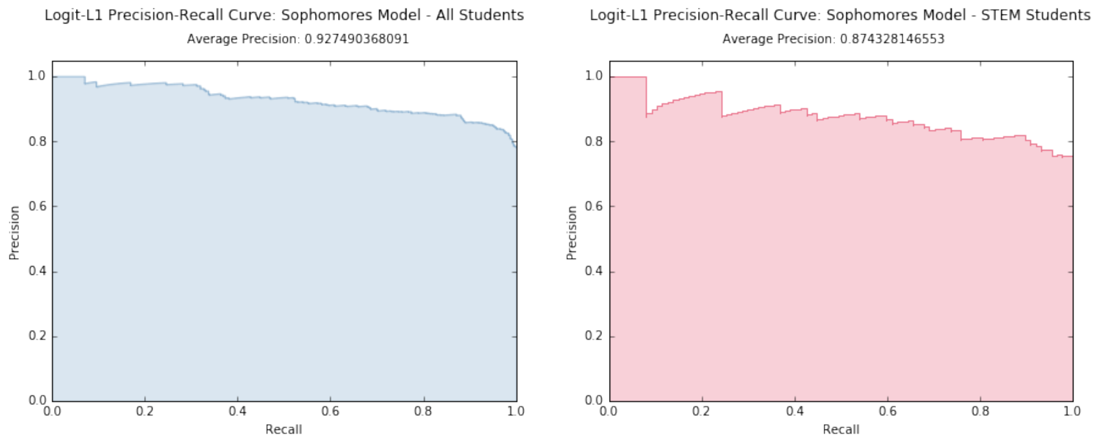

Figure 4.17: Precision-Recall Curve: Sophomore Model

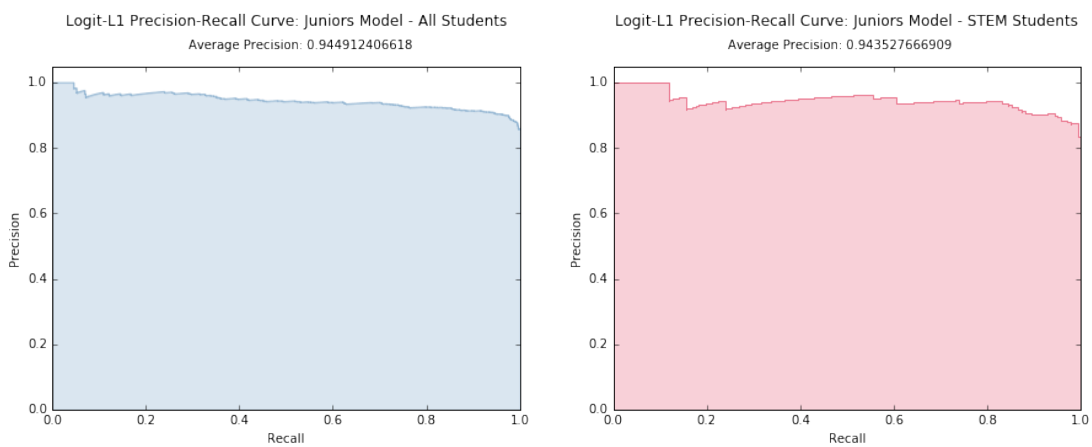

Figure 4.18: Precision-Recall Curve: Junior Model

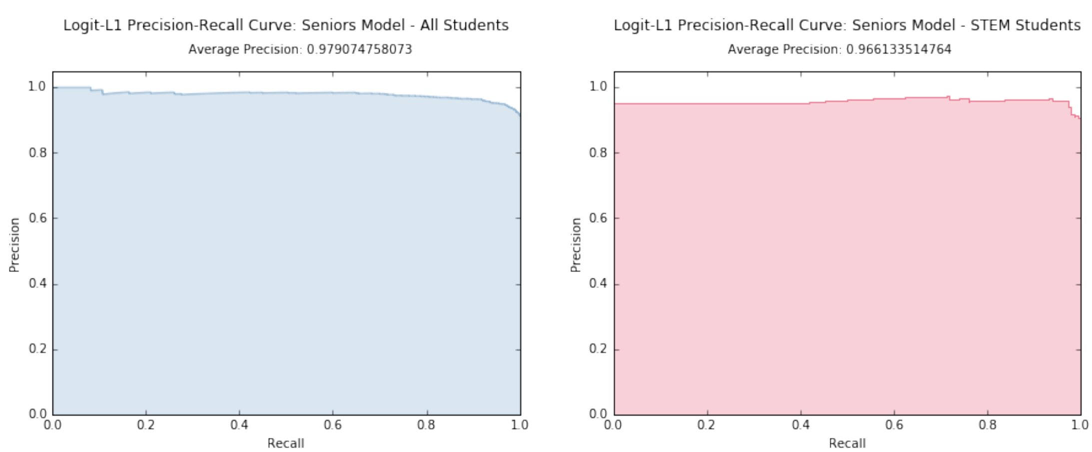

Figure 4.19: Precision-Recall Curve: Senior Model

ROC Curves

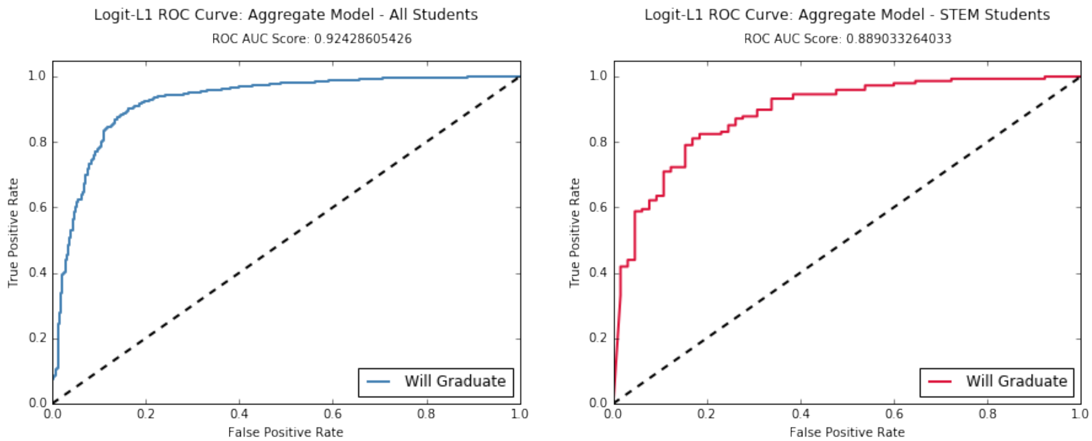

Figure 4.20: ROC Curve: Aggregate Model

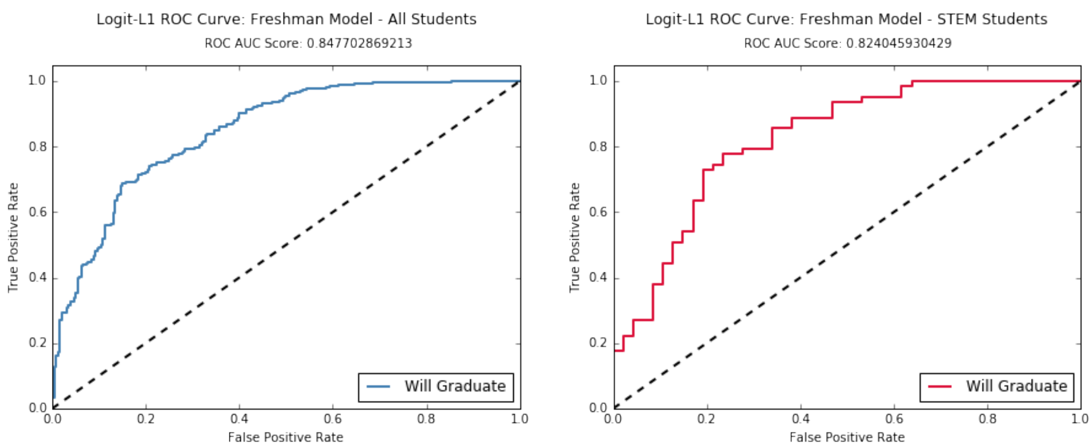

Figure 4.21: ROC Curve: Freshman Model

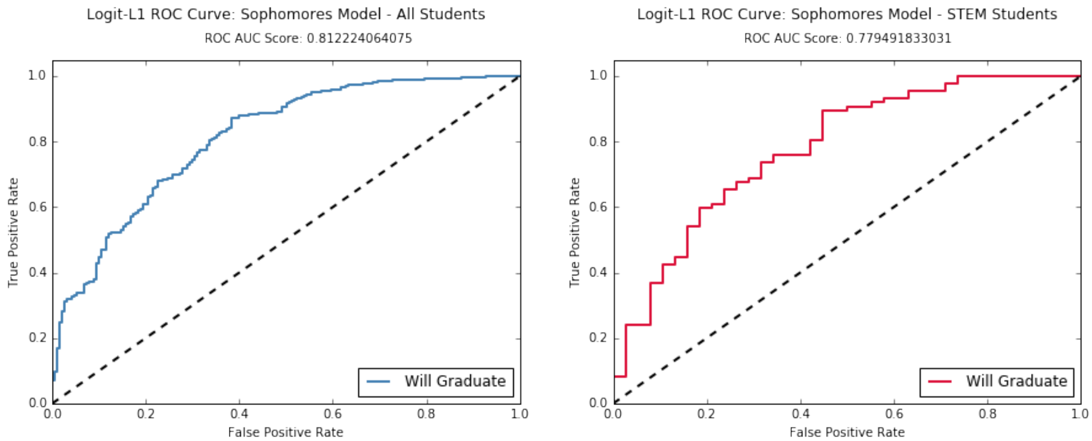

Figure 4.22: ROC Curve: Sophomore Model

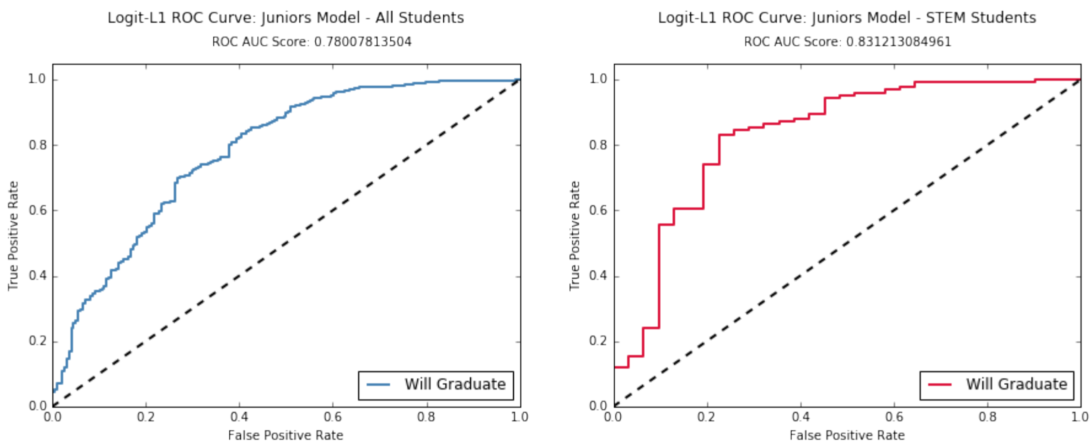

Figure 4.23: ROC Curve: Junior Model

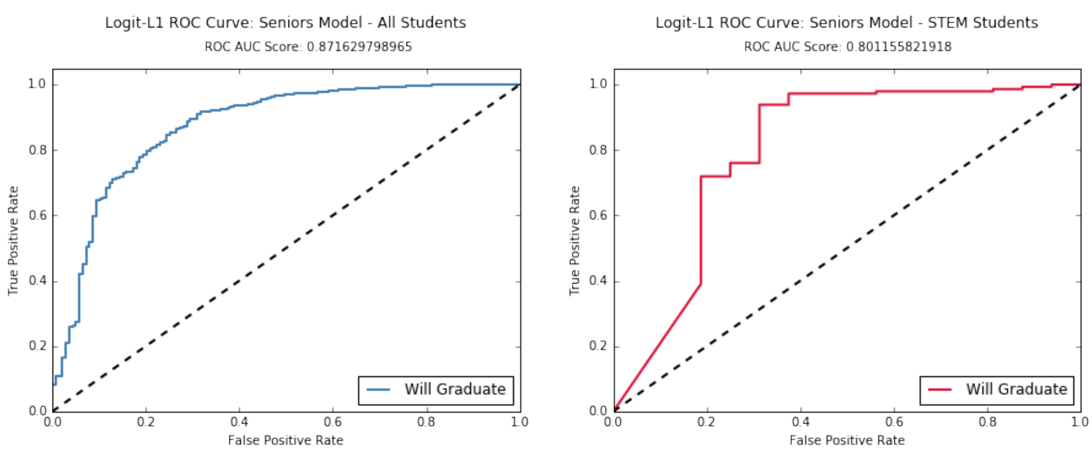

Figure 4.24: ROC Curve: Senior Model

Top Features

The top twenty-five features for each model are shown below; top features are determined by the absolute value of the model coefficients, indicating features with the strongest influence on final classification.

Top Features: Aggregate Model

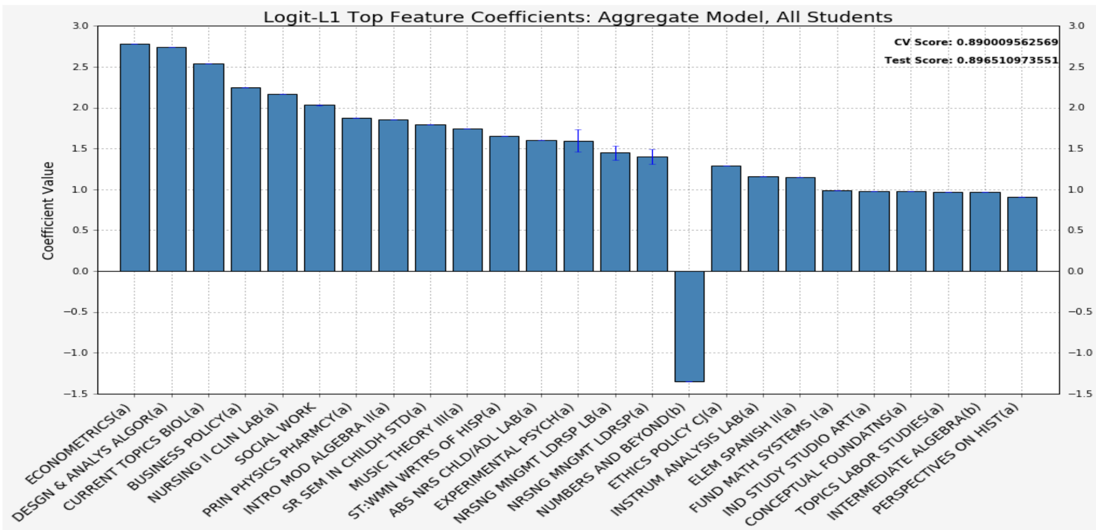

Figure 4.25: Top Features: Aggregate Model, All Students

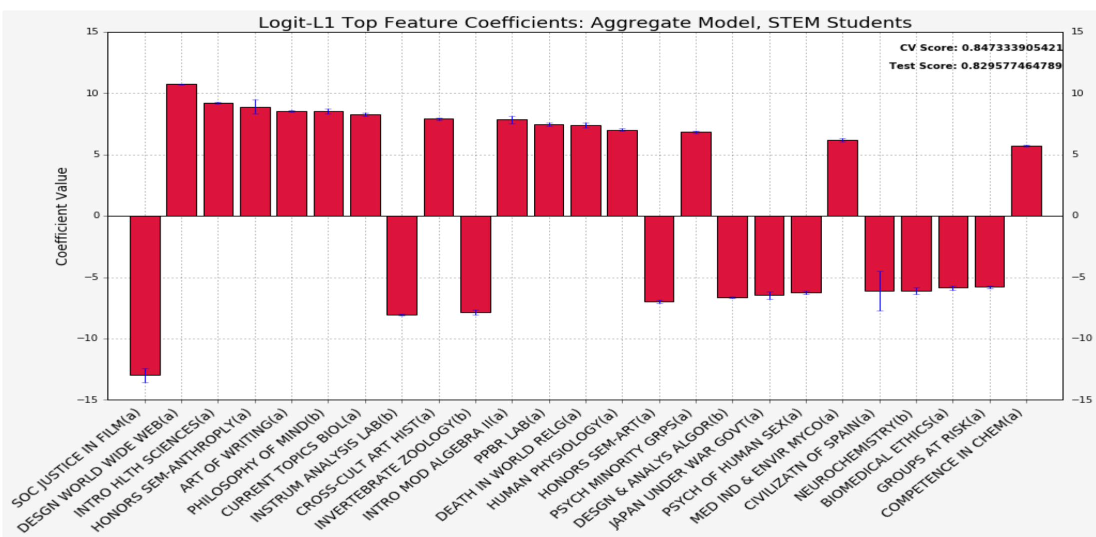

Figure 4.26: Top Features: Aggregate Model, STEM Students

Top Features: Freshman Model

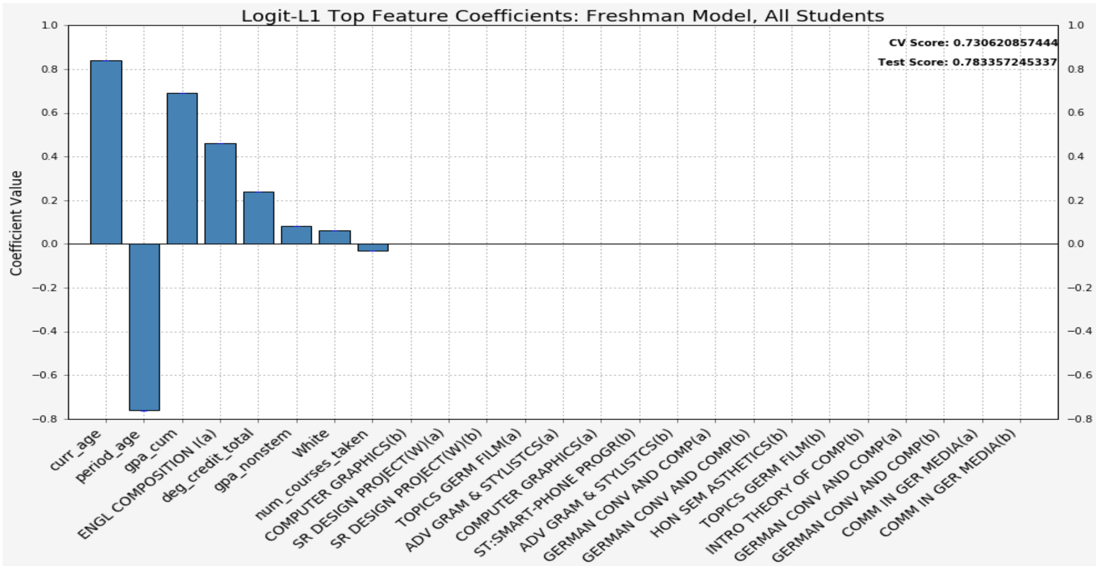

Figure 4.27: Top Features: Freshman Model, All Students

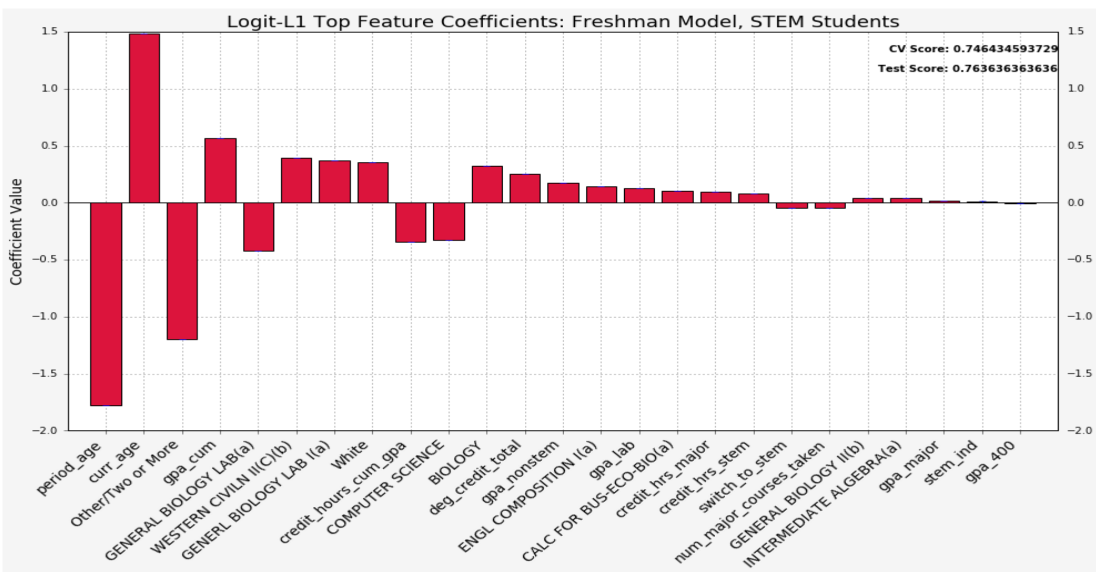

Figure 4.28: Top Features: Freshman Model, STEM Students

Top Features: Sophomores Model

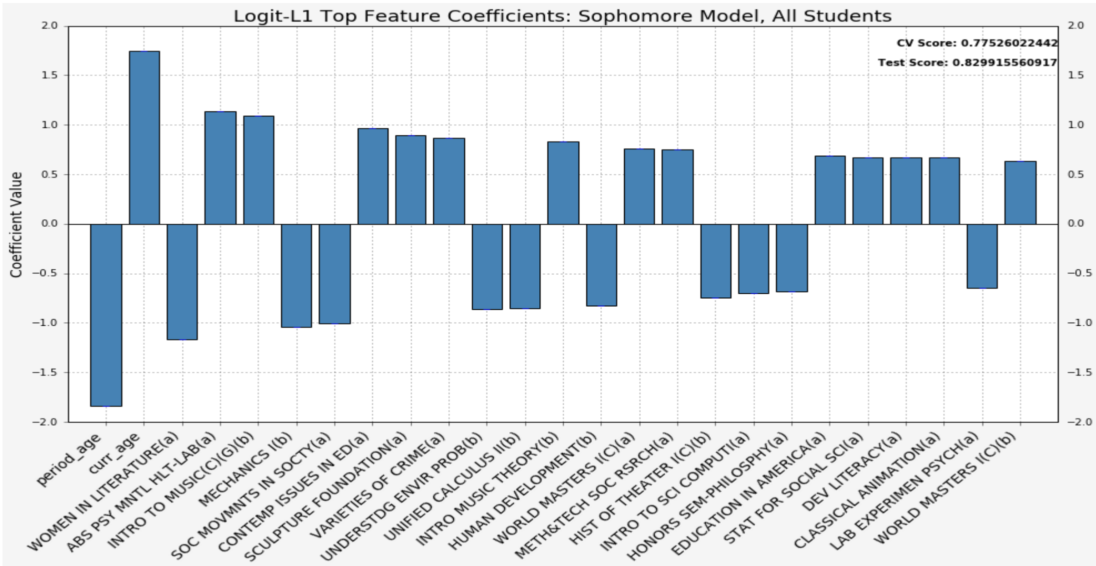

Figure 4.29: Top Features: Sophomores Model, All Students

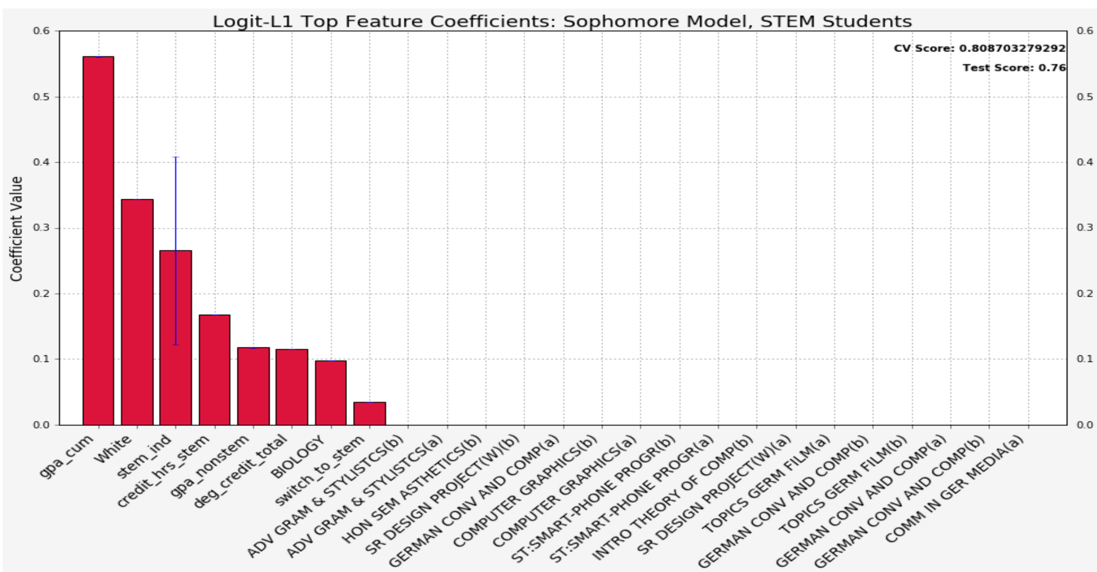

Figure 4.30: Top Features: Sophomores Model, STEM Students

Top Features: Juniors Model

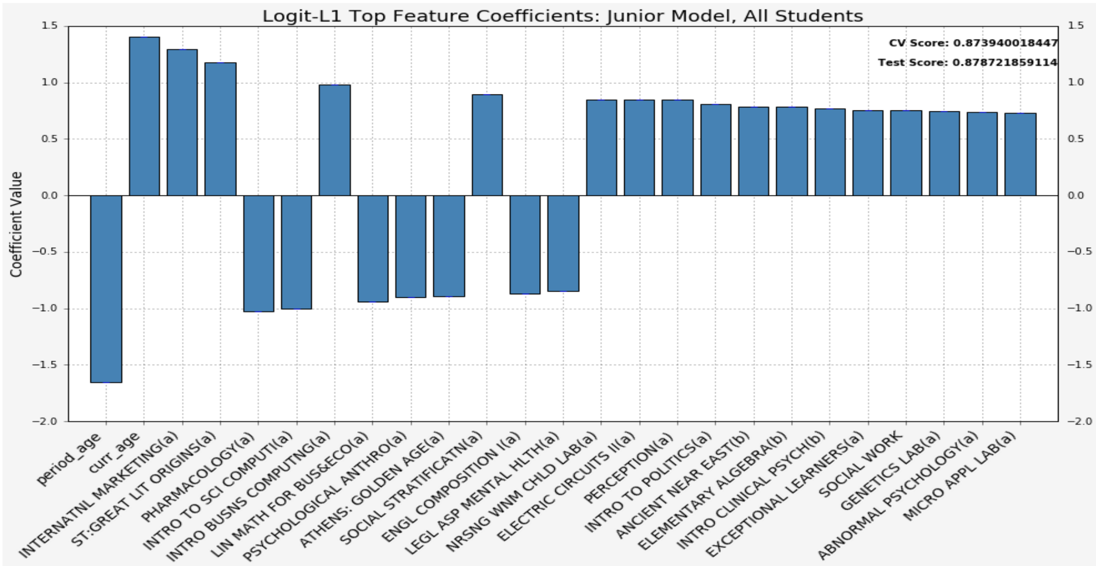

Figure 4.31: Top Features: Juniors Model, All Students

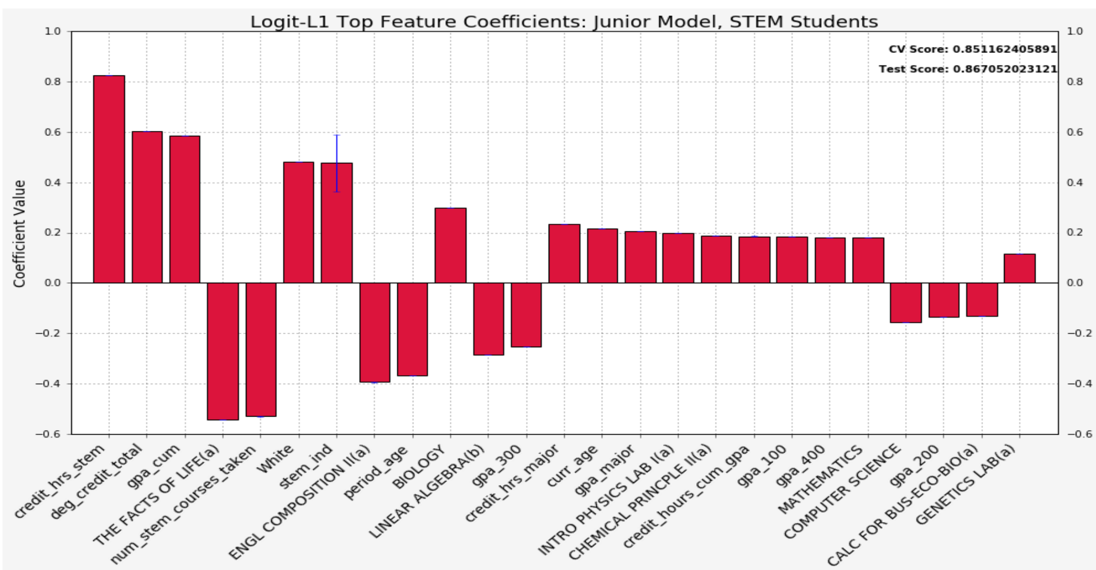

Figure 4.32: Top Features: Juniors Model, STEM Students

Top Features: Seniors Model

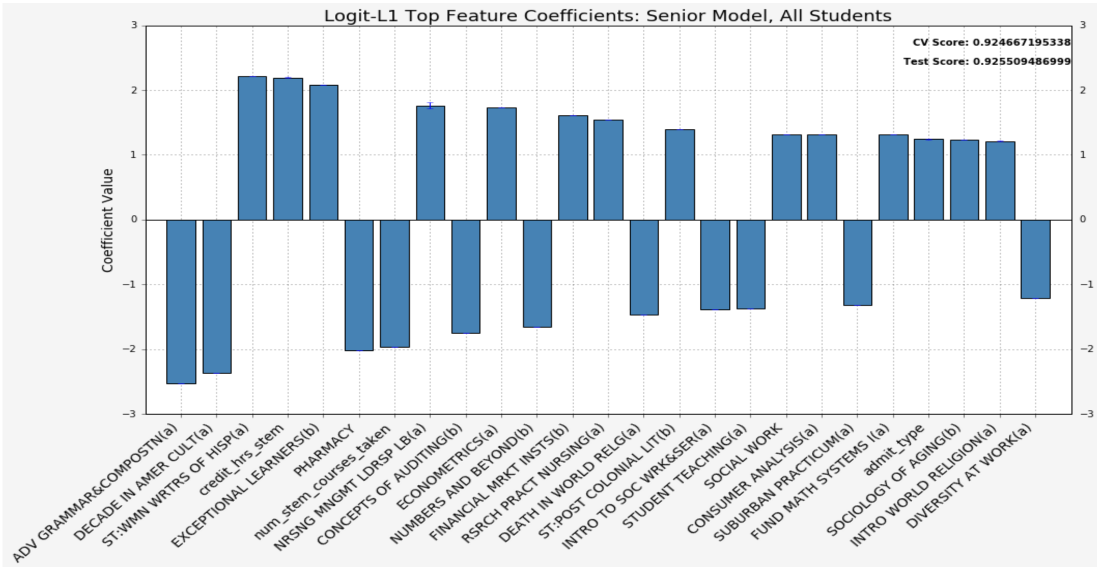

Figure 4.33: Top Features: Seniors Model, All Students

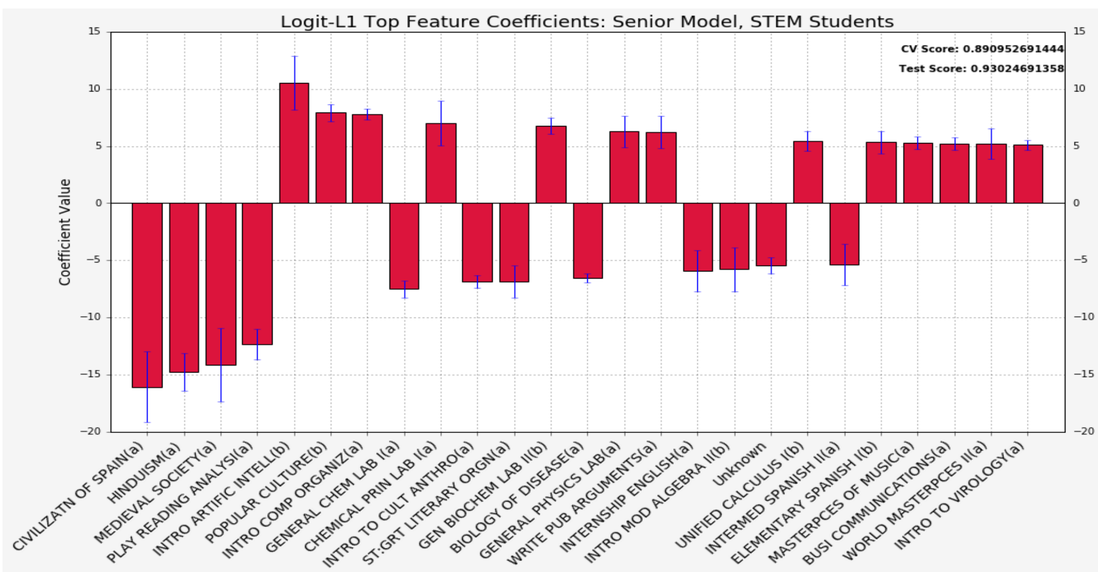

Figure 4.34: Top Features: Seniors Model, STEM Students

4.2.2 Logistic Regression - L2 Penalty

Accuracy

Classifier accuracy is measured in terms of the number of correct classifications on the validation set.

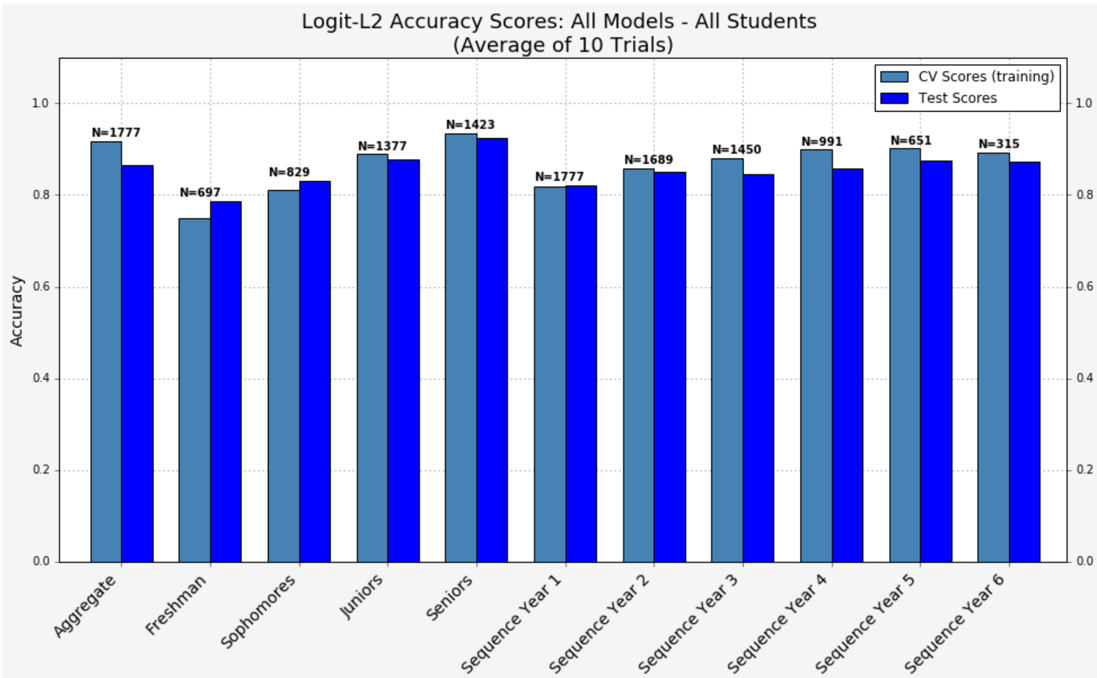

Figure 4.35: Logit-L2 Accuracy Scores: All Models, All Students

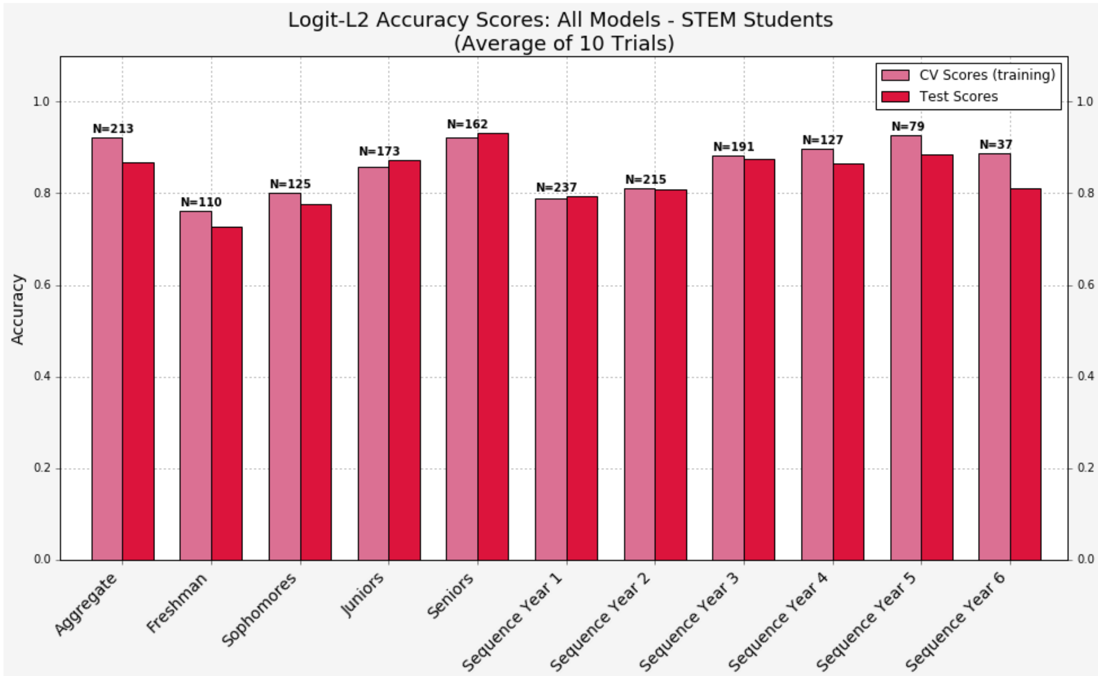

Figure 4.36: Logit-L2 Accuracy Scores: All Models, STEM Students

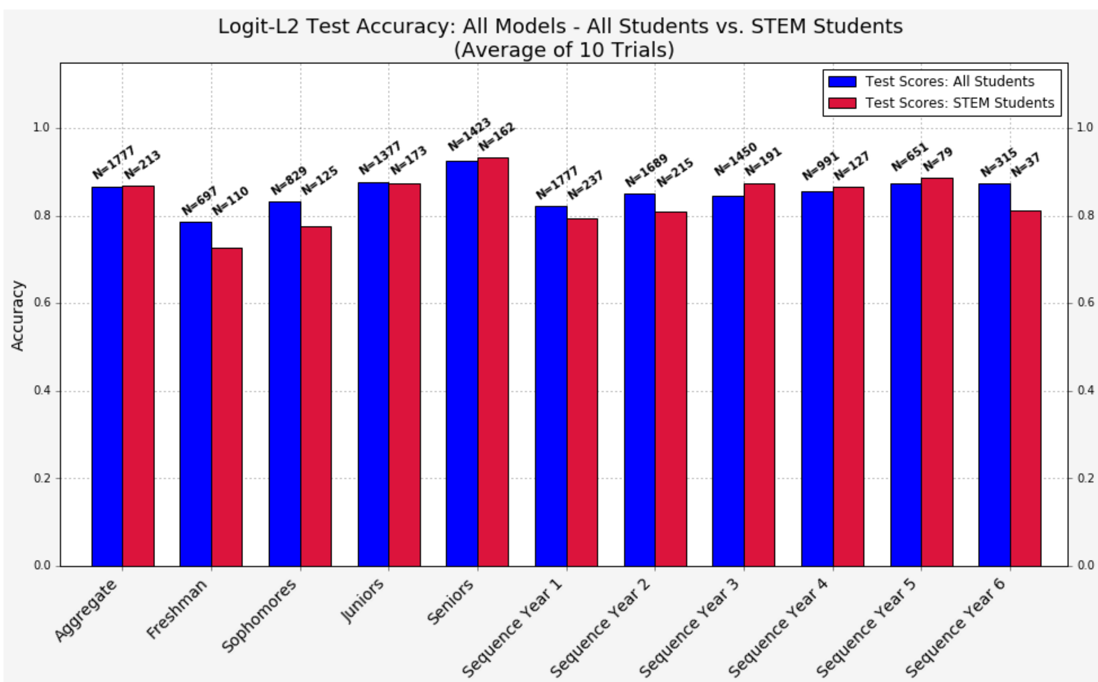

Figure 4.37: Logit-L2 Accuracy Scores: All Models, All Students vs STEM Students

Prediction Distributions

Prediction distributions are analyzed via confusion matrices, precision, recall, F-scores, precision-recal curves, and ROC curves.

Confusion Matrices:

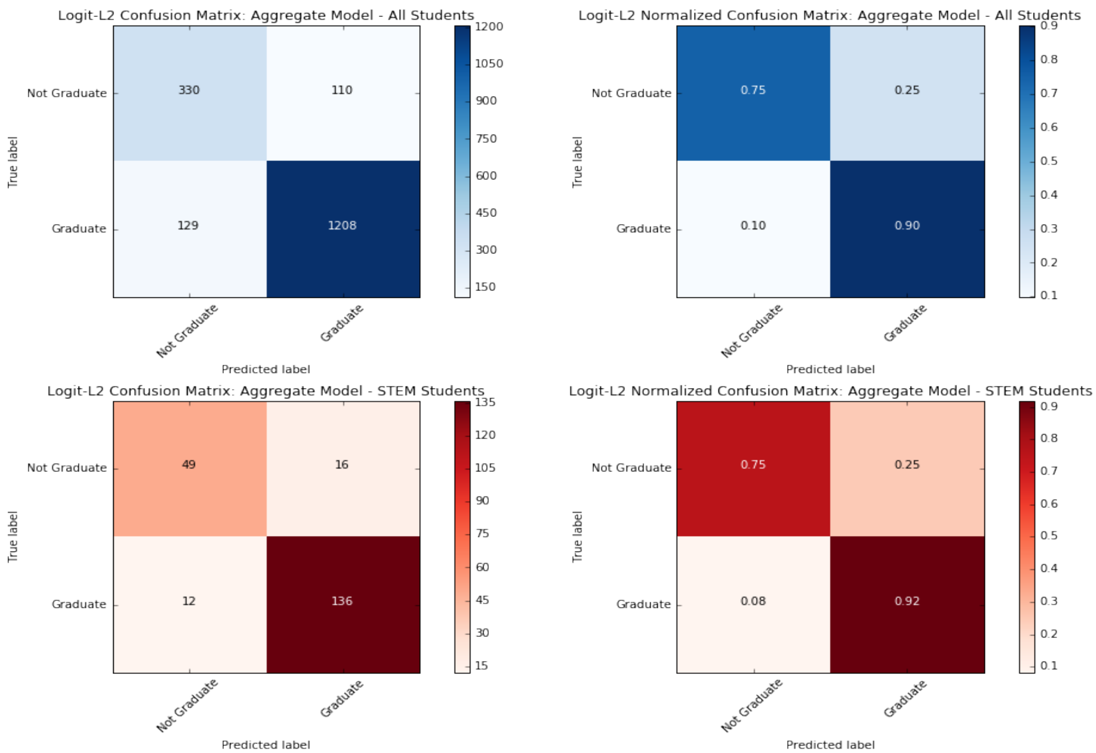

Figure 4.38: Logit-L2 Confusion Matrices: Aggregate Model

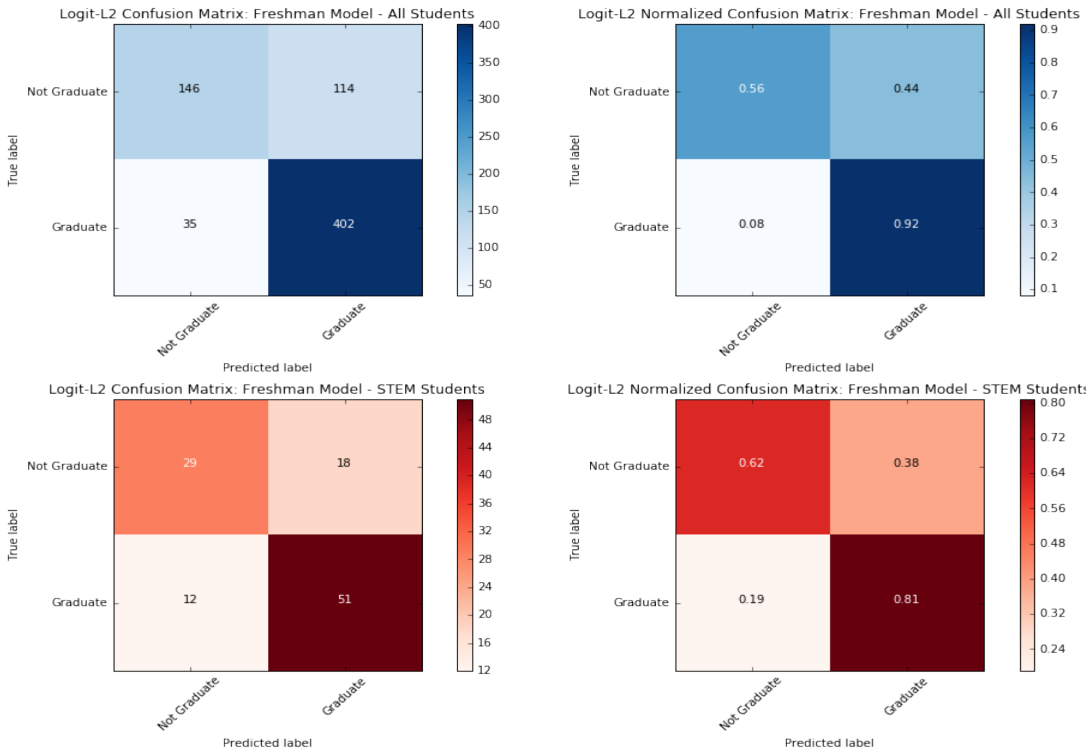

Figure 4.39: Logit-L2 Confusion Matrices: Freshman Model

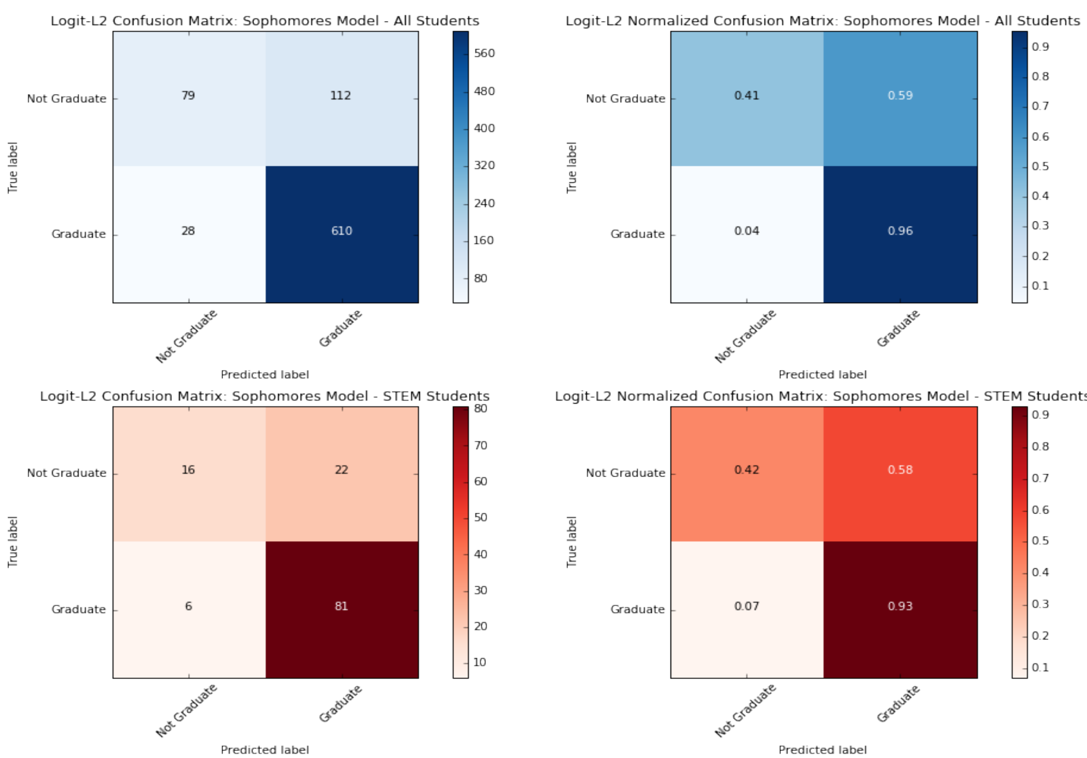

Figure 4.40: Logit-L2 Confusion Matrices: Sophomore Model

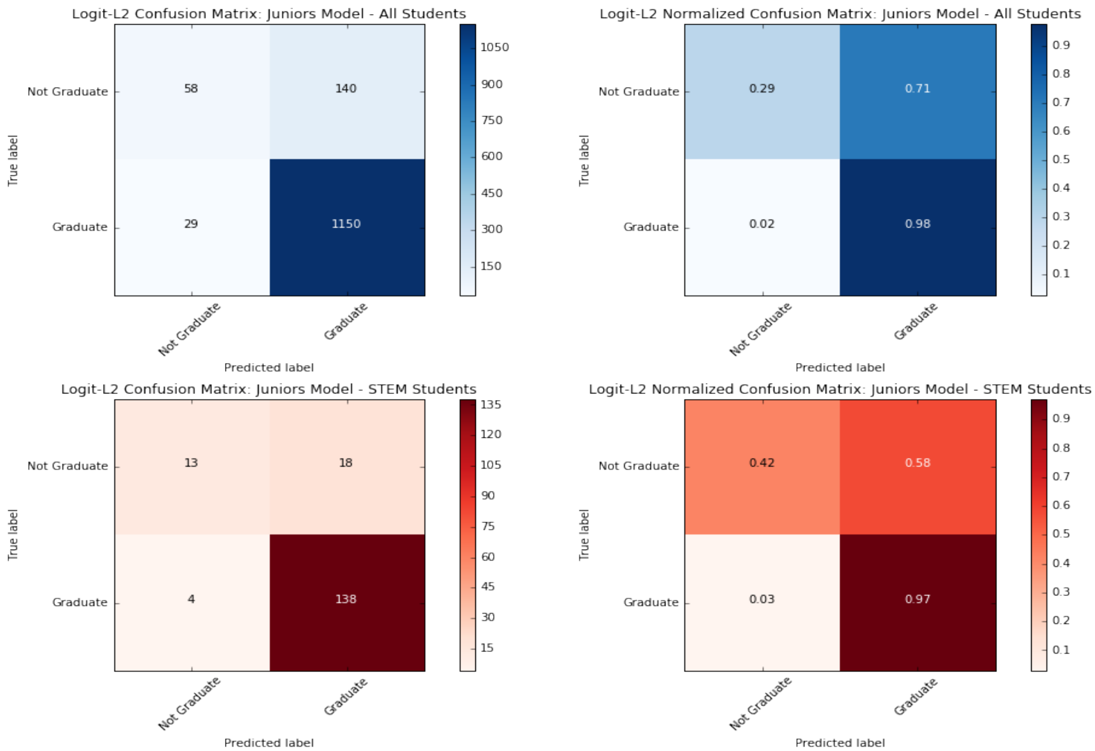

Figure 4.41: Logit-L2 Confusion Matrices: Junior Model

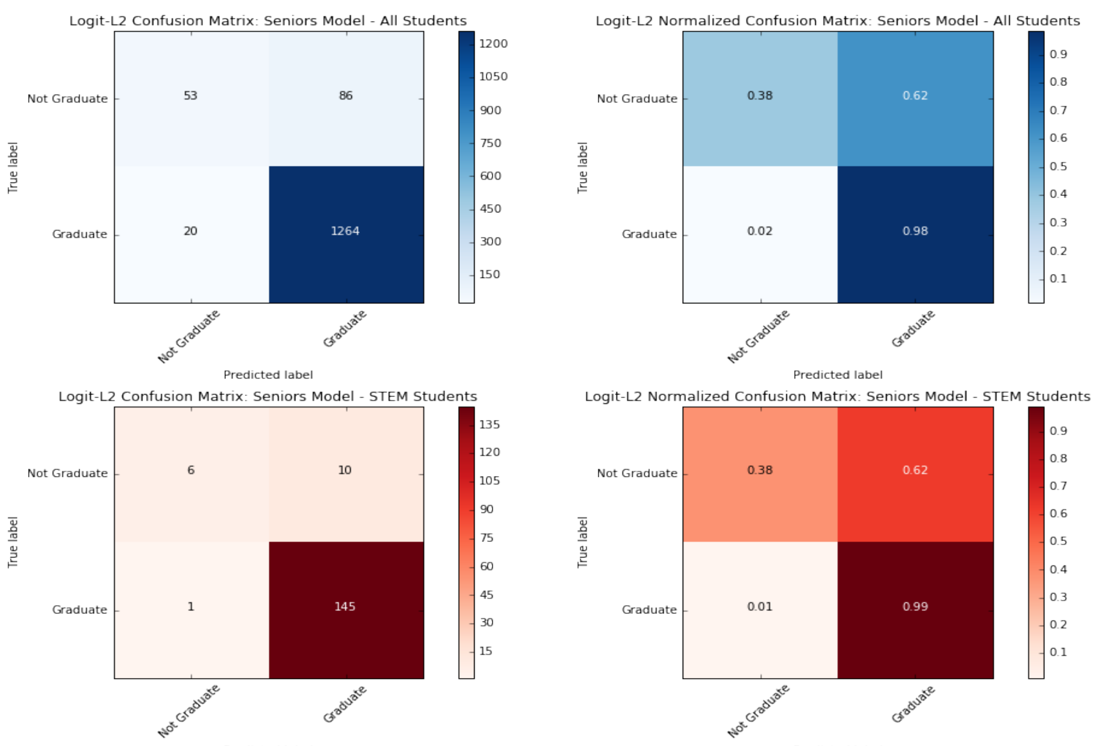

Figure 4.42: Logit-L2 Confusion Matrices: Senior Model

Precision, Recall, F-Scores

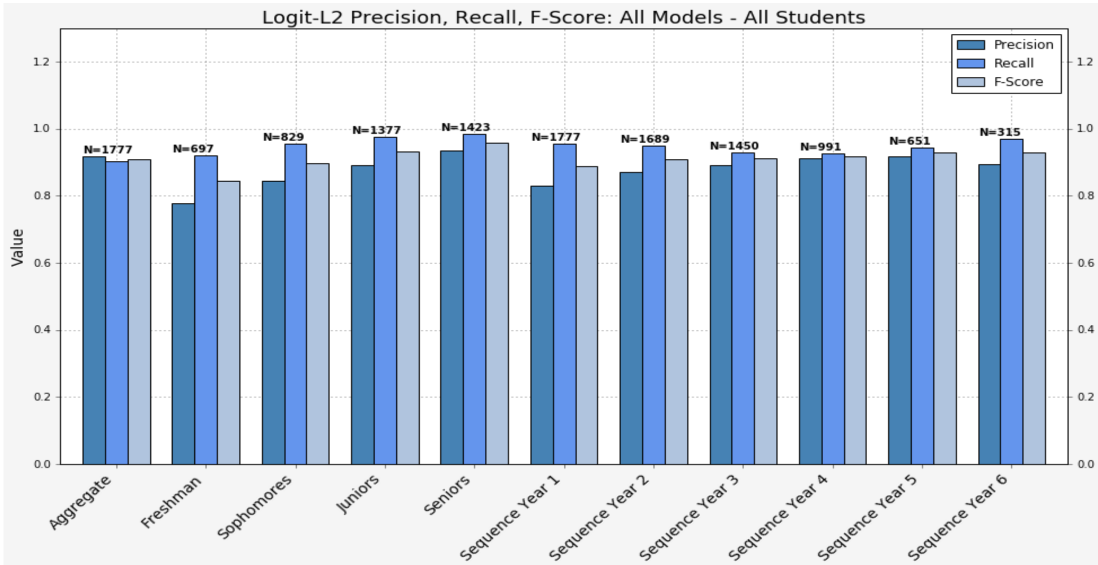

Figure 4.43: Logit-L2 Precision, Recall, and F-scores: All Models, All Students

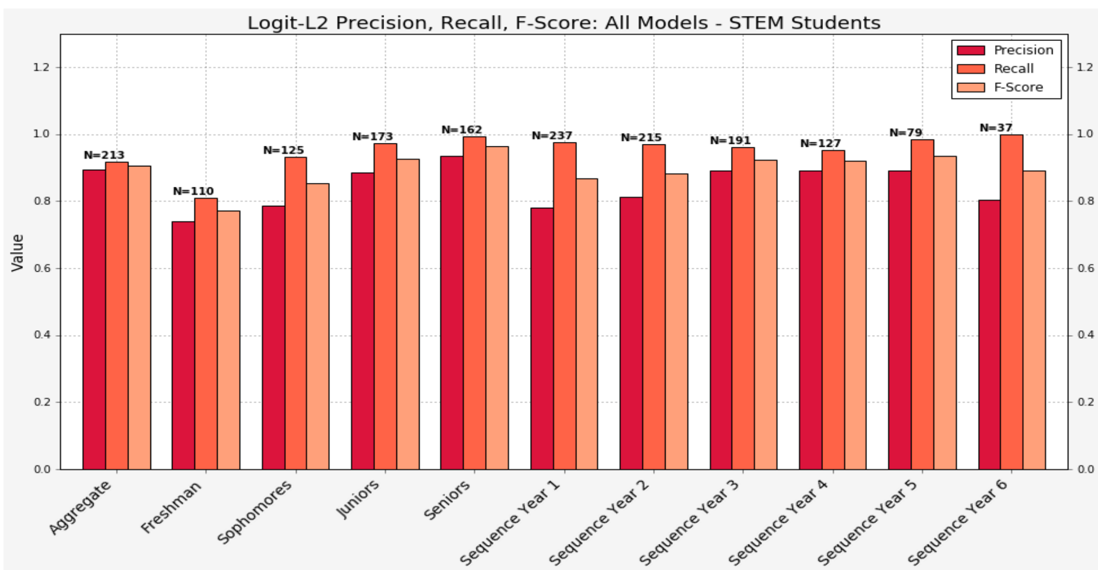

Figure 4.44: Logit-L2 Precision, Recall, and F-scores: All Models, STEM Students

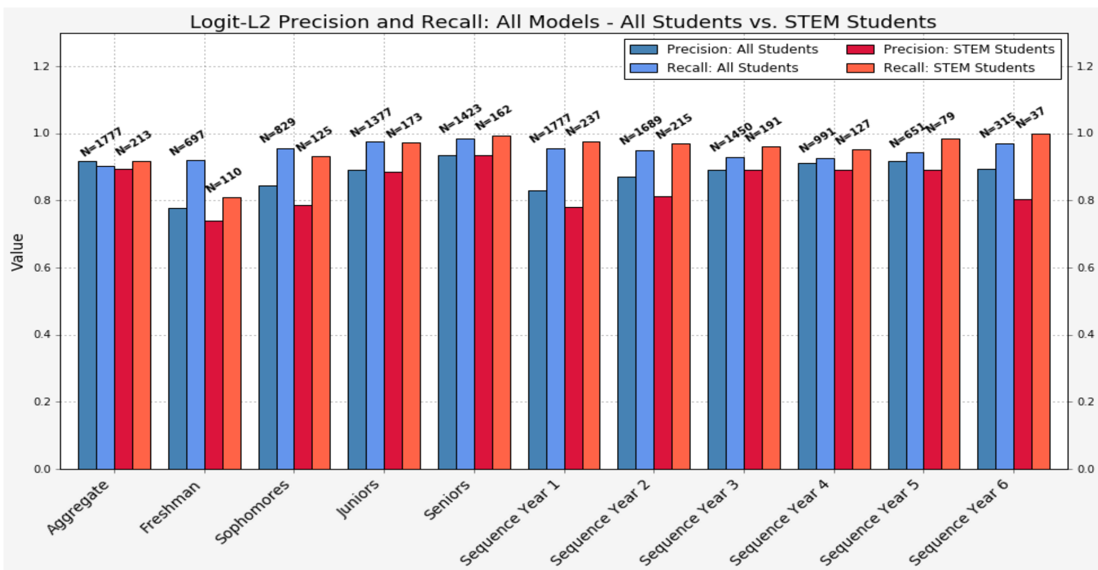

Figure 4.45: Logit-L2 Precision and Recall: All Models, All Students vs STEM Students

Precision-Recall Curves

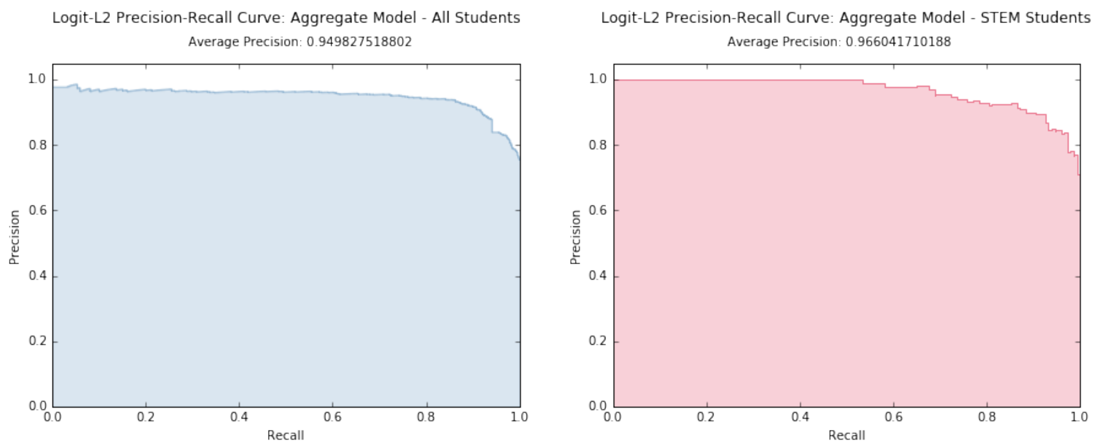

Figure 4.46: Precision-Recall Curve: Aggregate Model

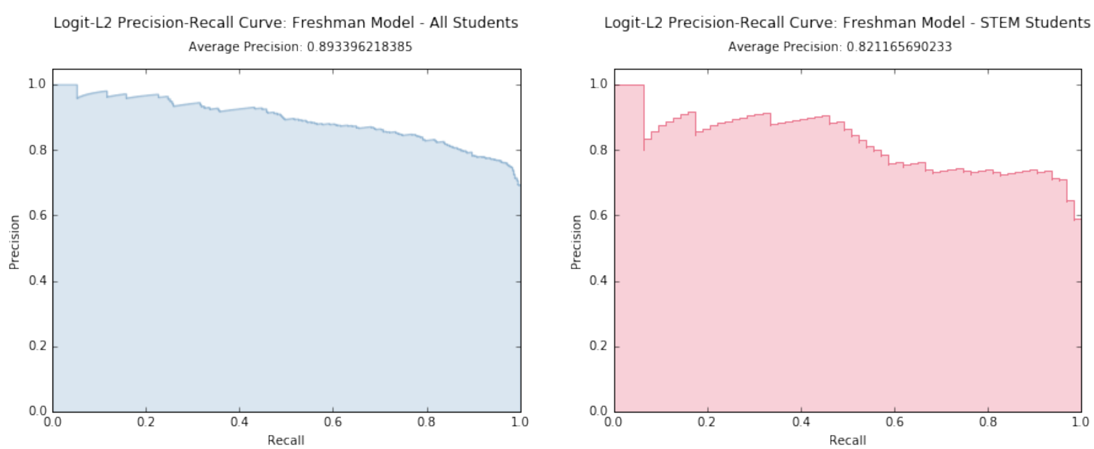

Figure 4.47: Precision-Recall Curve: Freshman Model

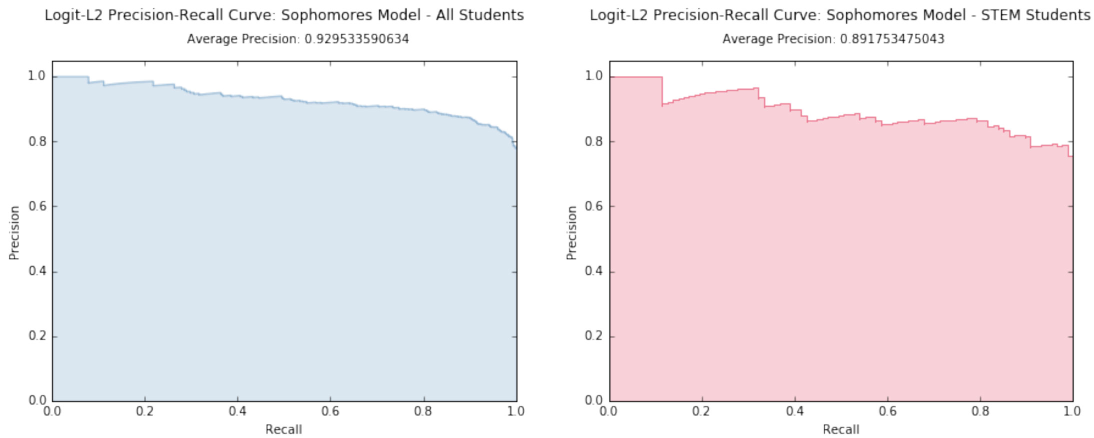

Figure 4.48: Precision-Recall Curve: Sophomore Model

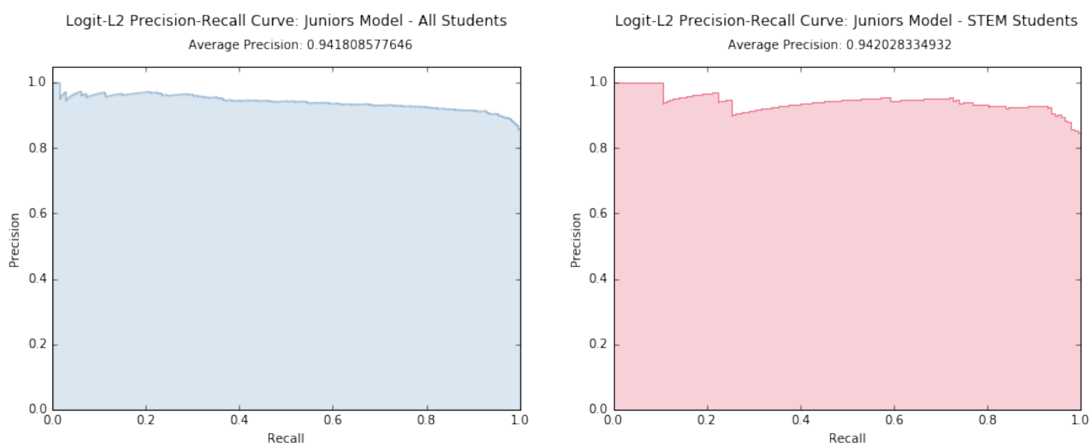

Figure 4.49: Precision-Recall Curve: Junior Model

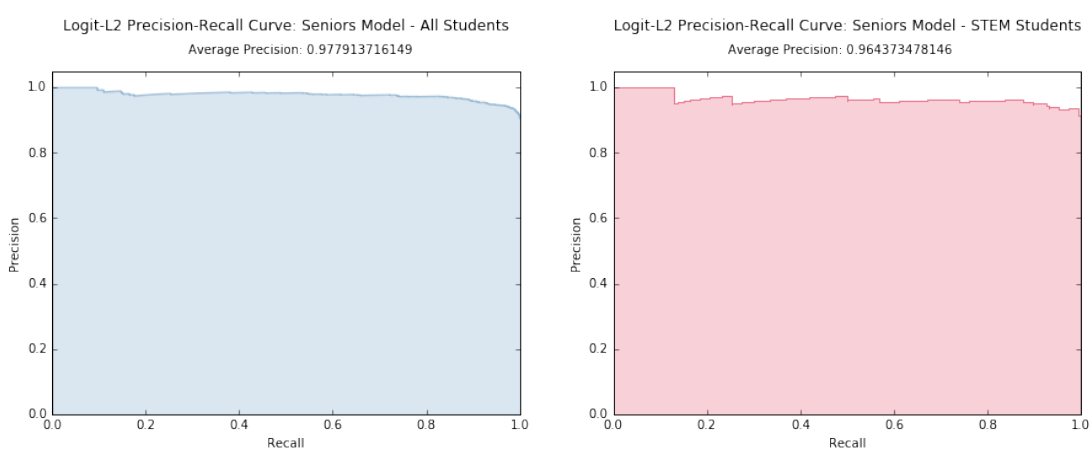

Figure 4.50: Precision-Recall Curve: Senior Model

ROC Curves

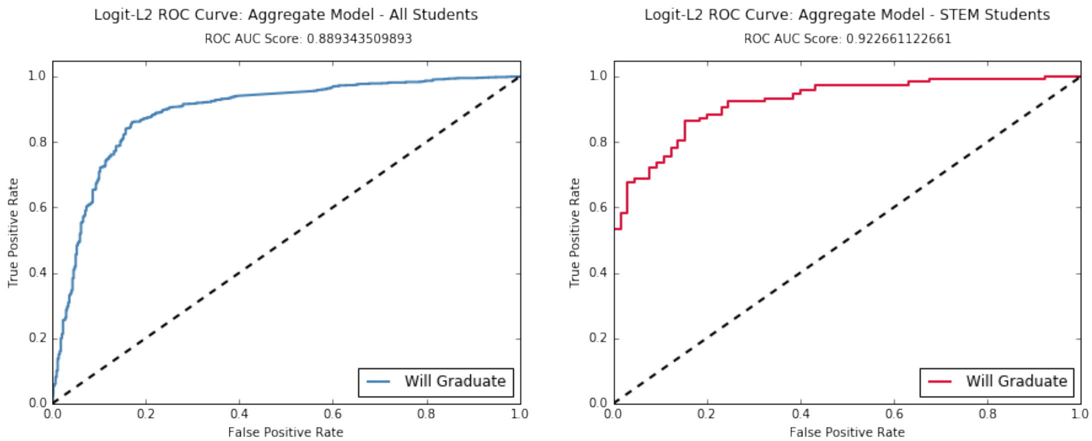

Figure 4.51: ROC Curve: Aggregate Model

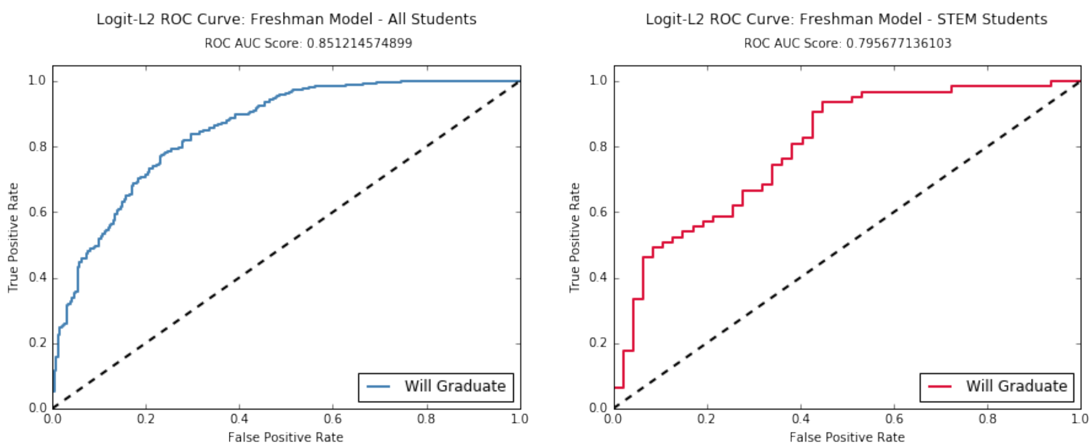

Figure 4.52: ROC Curve: Freshman Model

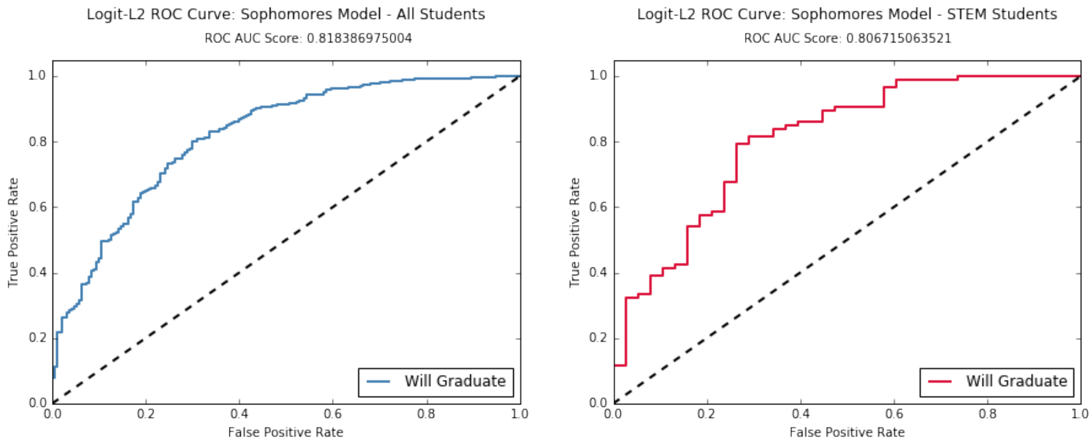

Figure 4.53: ROC Curve: Sophomore Model

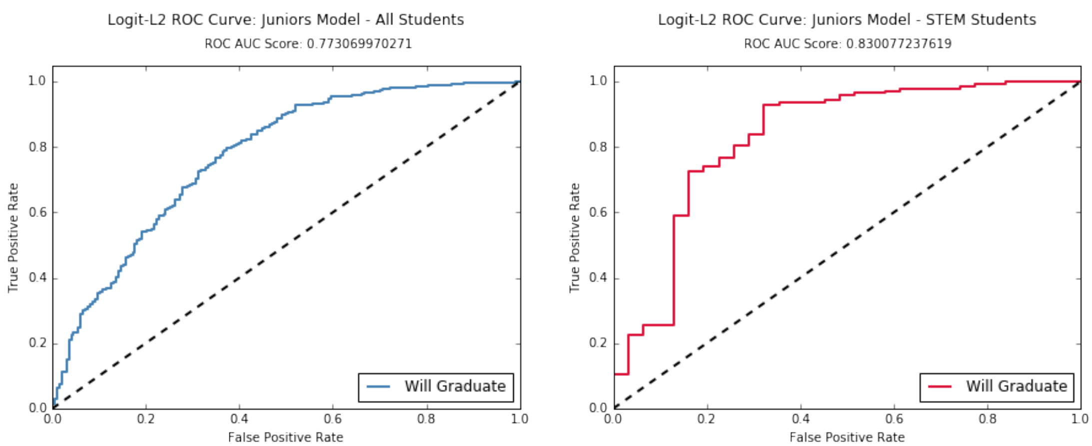

Figure 4.54: ROC Curve: Junior Model

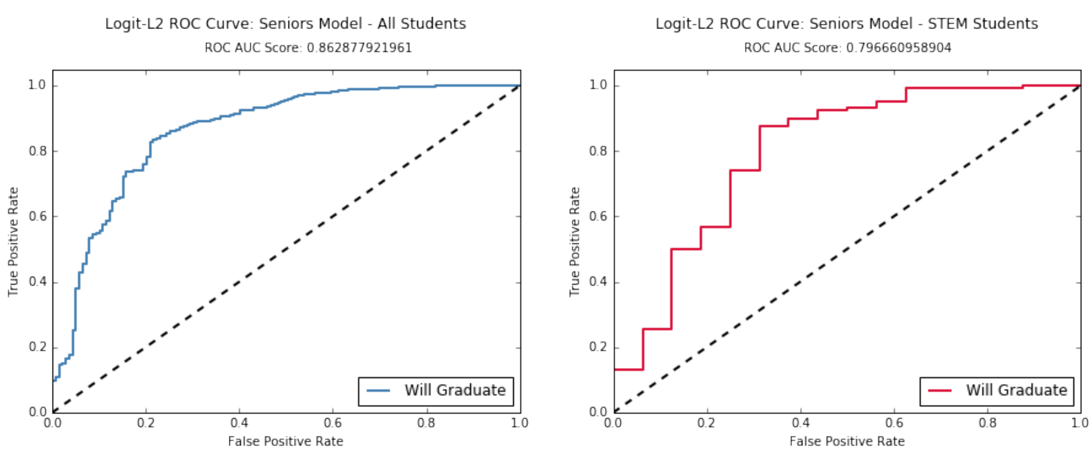

Figure 4.55: ROC Curve: Senior Model

Top Features

The top twenty-five features for each model are shown below; top features are determined by the absolute value of the model coefficients, indicating features with the strongest influence on final classification.

Top Features: Aggregate Model

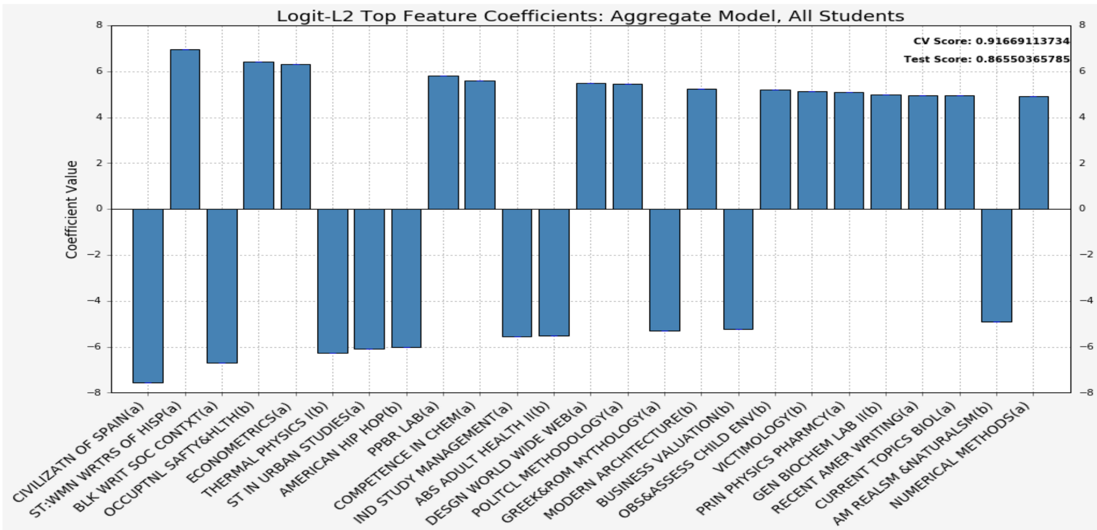

Figure 4.56: Top Features: Aggregate Model, All Students

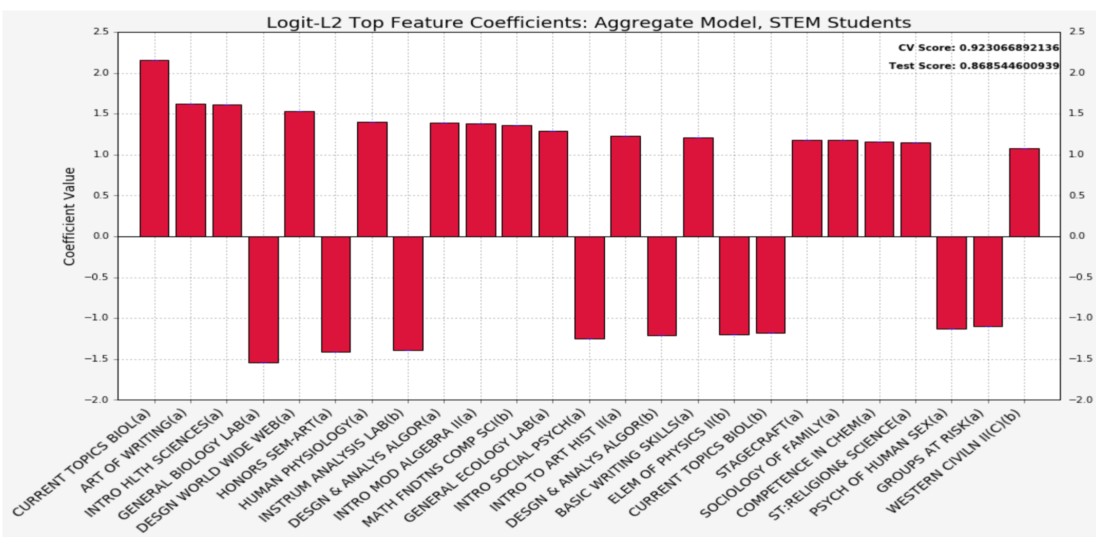

Figure 4.57: Top Features: Aggregate Model, STEM Students

Top Features: Freshman Model

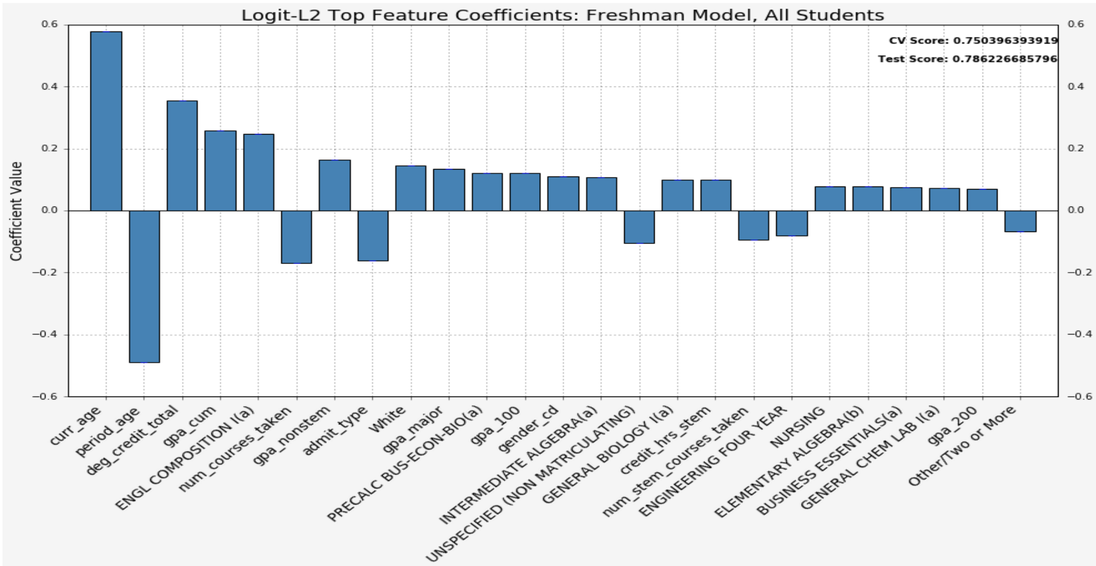

Figure 4.58: Top Features: Freshman Model, All Students

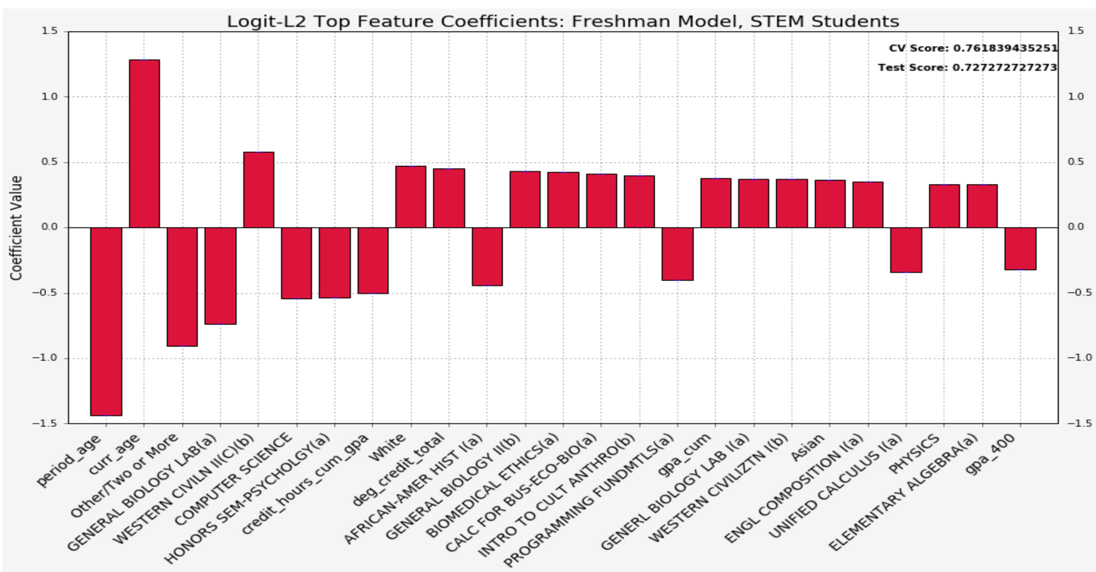

Figure 4.59: Top Features: Freshman Model, STEM Students

Top Features: Sophomores Model

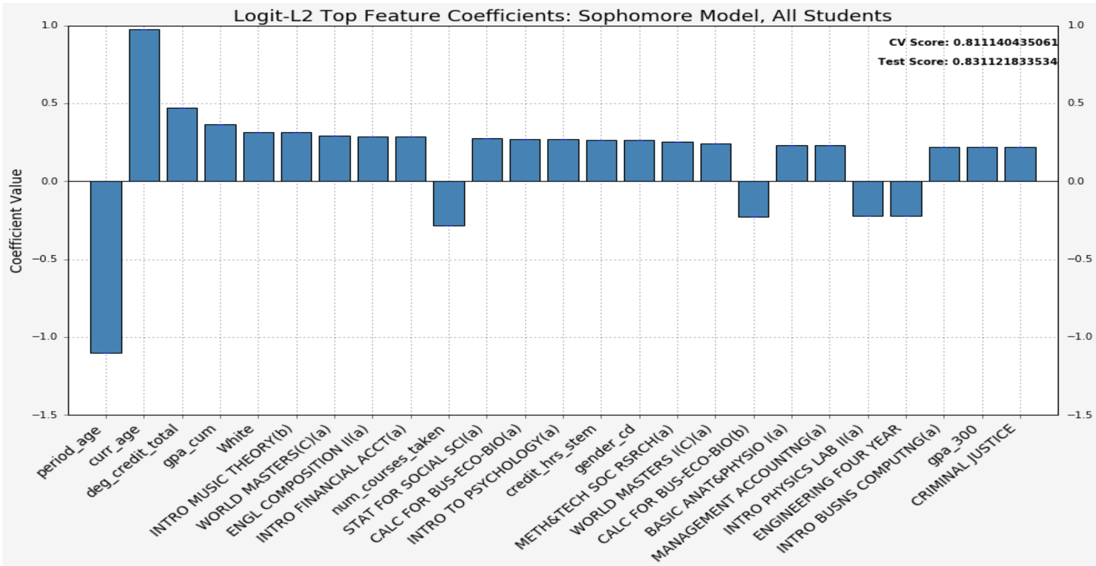

Figure 4.60: Top Features: Sophomores Model, All Students

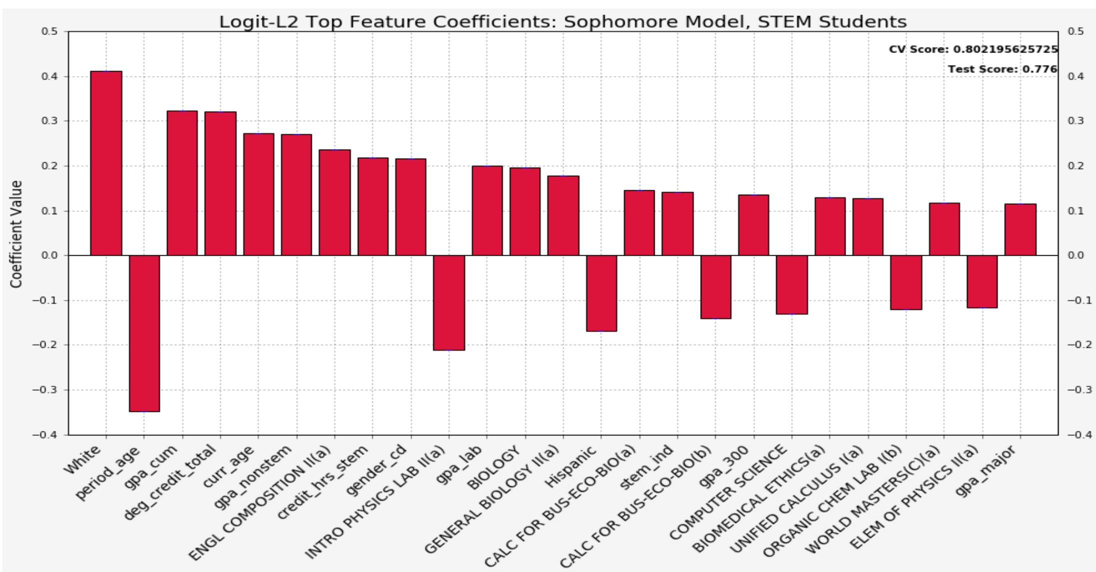

Figure 4.61: Top Features: Sophomores Model, STEM Students

Top Features: Juniors Model

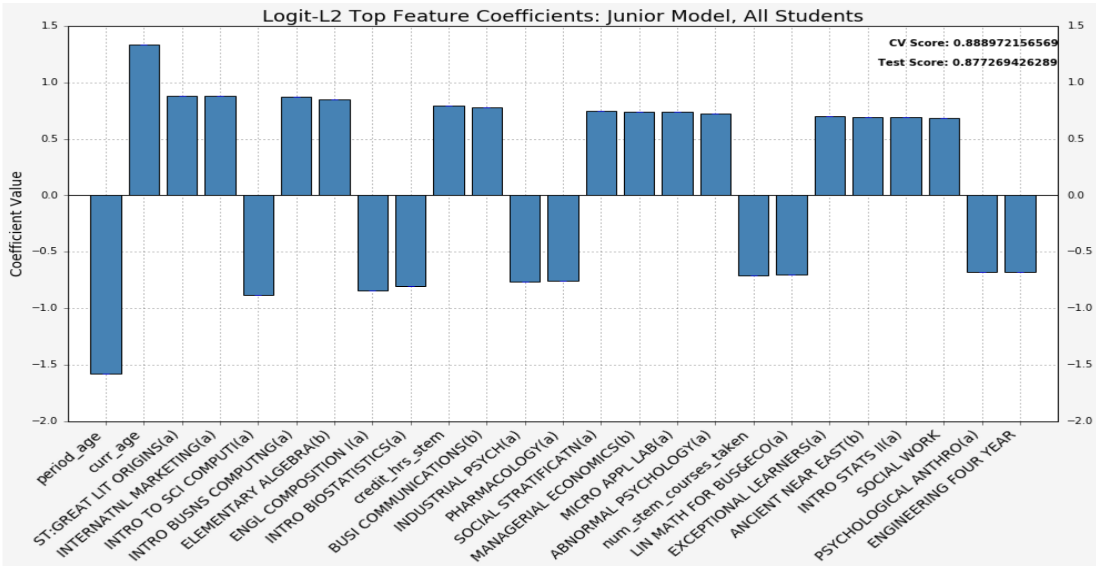

Figure 4.62: Top Features: Juniors Model, All Students

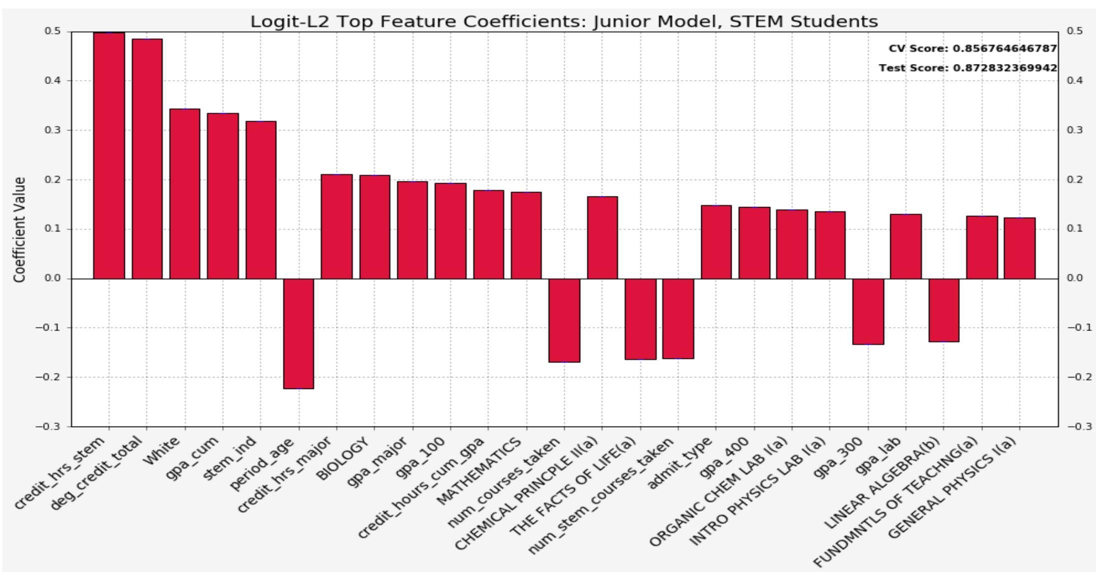

Figure 4.63: Top Features: Juniors Model, STEM Students

Top Features: Seniors Model

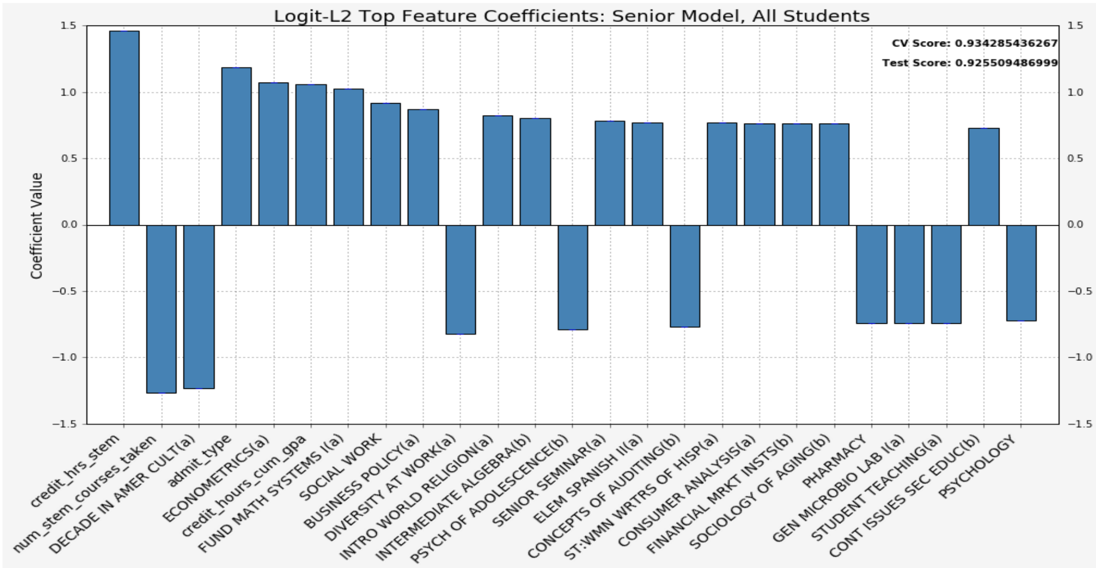

Figure 4.64: Top Features: Seniors Model, All Students

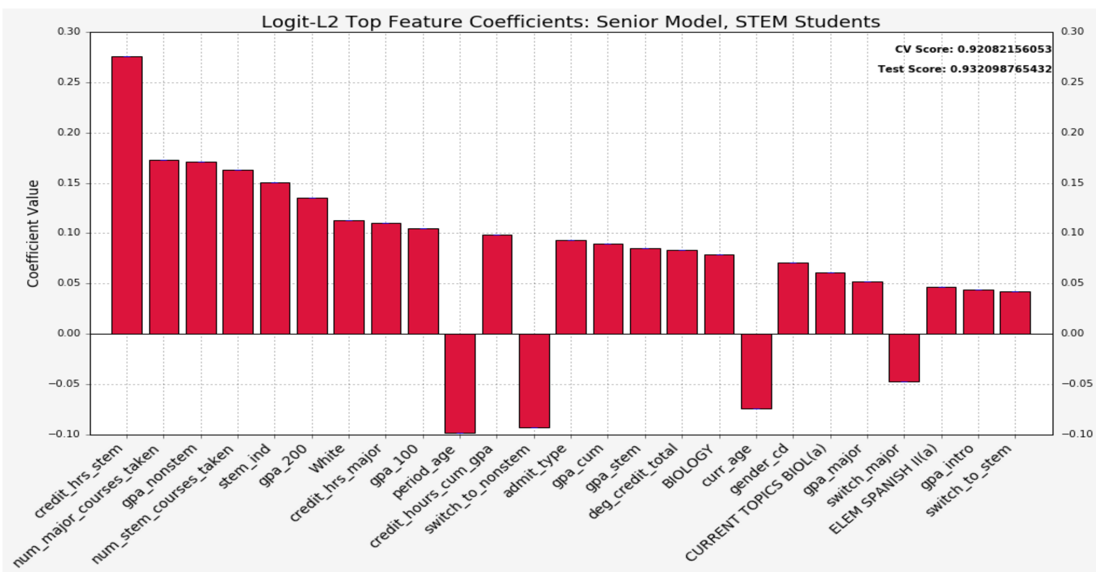

Figure 4.65: Top Features: Seniors Model, STEM Students

4.2.3 Random Forest Classifier

Accuracy

Classifier accuracy is measured in terms of the number of correct classifications on the validation set.

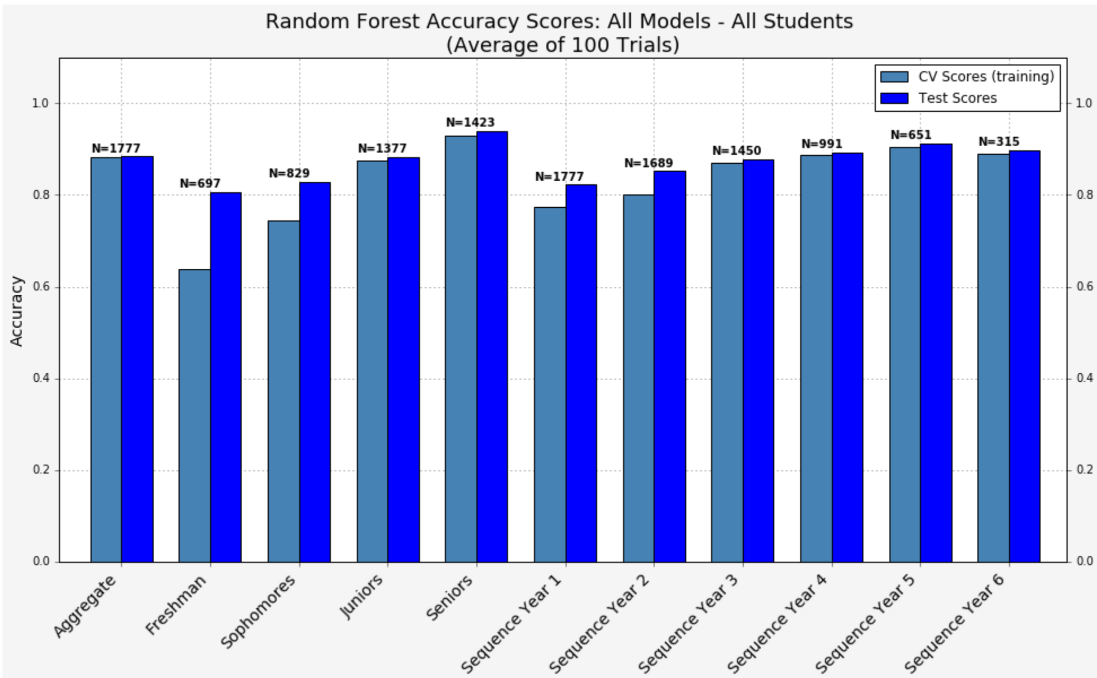

Figure 4.66: RF Accuracy Scores: All Models, All Students

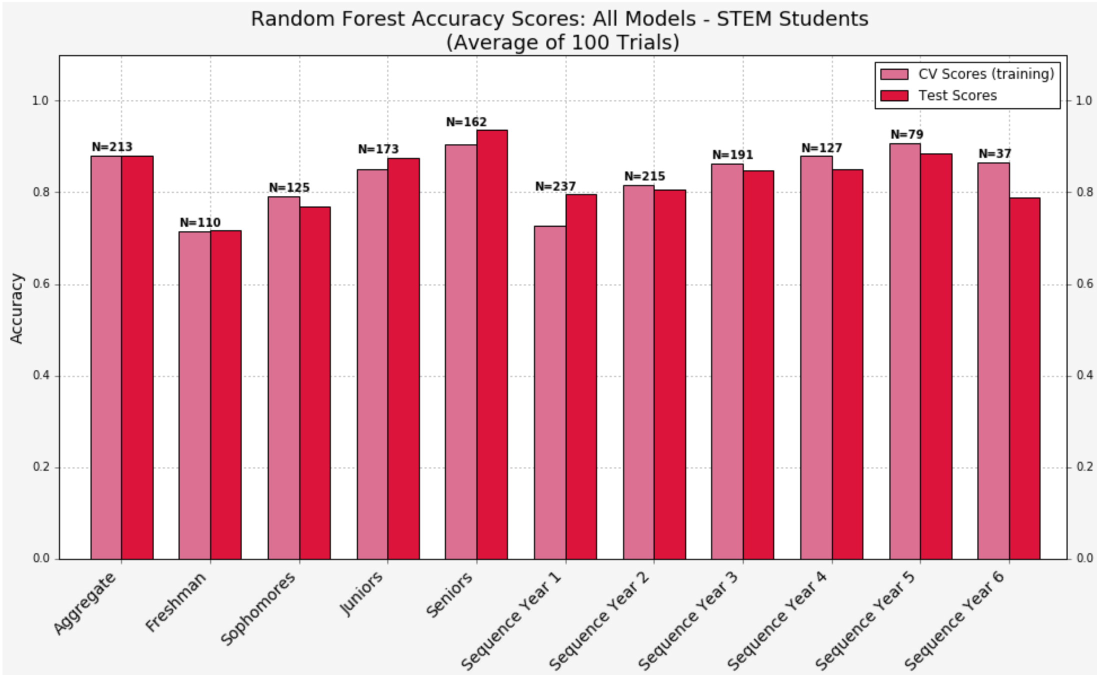

Figure 4.67: RF Accuracy Scores: All Models, STEM Students

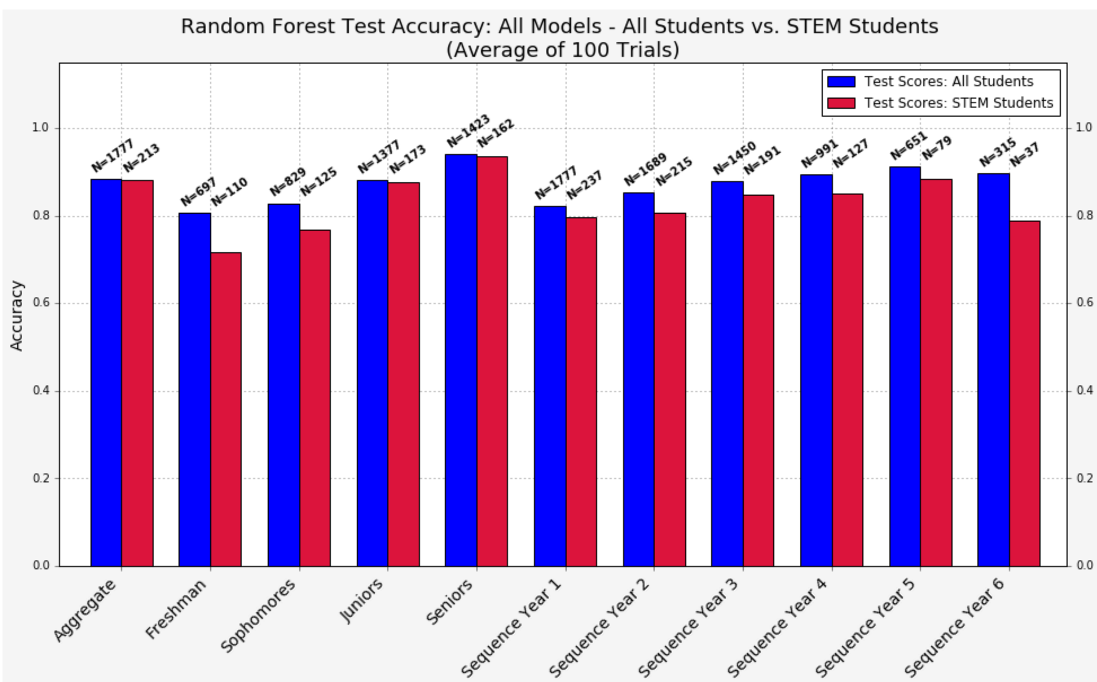

Figure 4.68: RF Accuracy Scores: All Models, All Students vs STEM Students

Prediction Distributions

Prediction distributions are analyzed via confusion matrices, precision, recall, F-scores, precision-recal curves, and ROC curves.

Confusion Matrices:

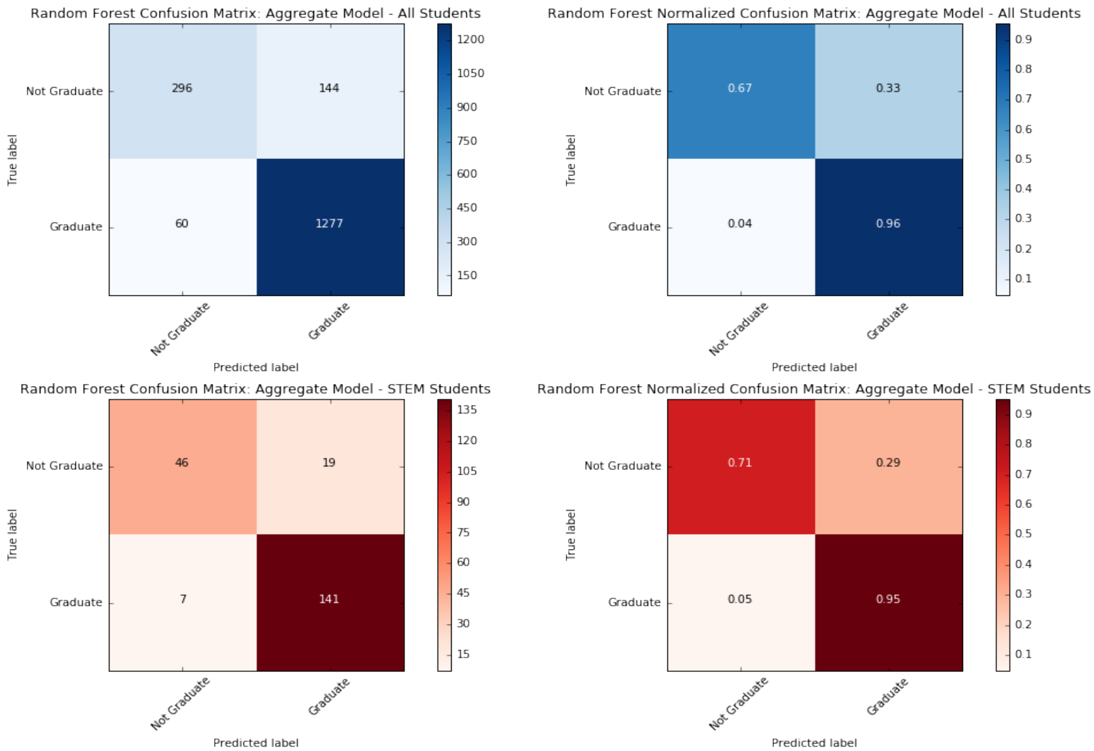

Figure 4.69: RF Confusion Matrices: Aggregate Model

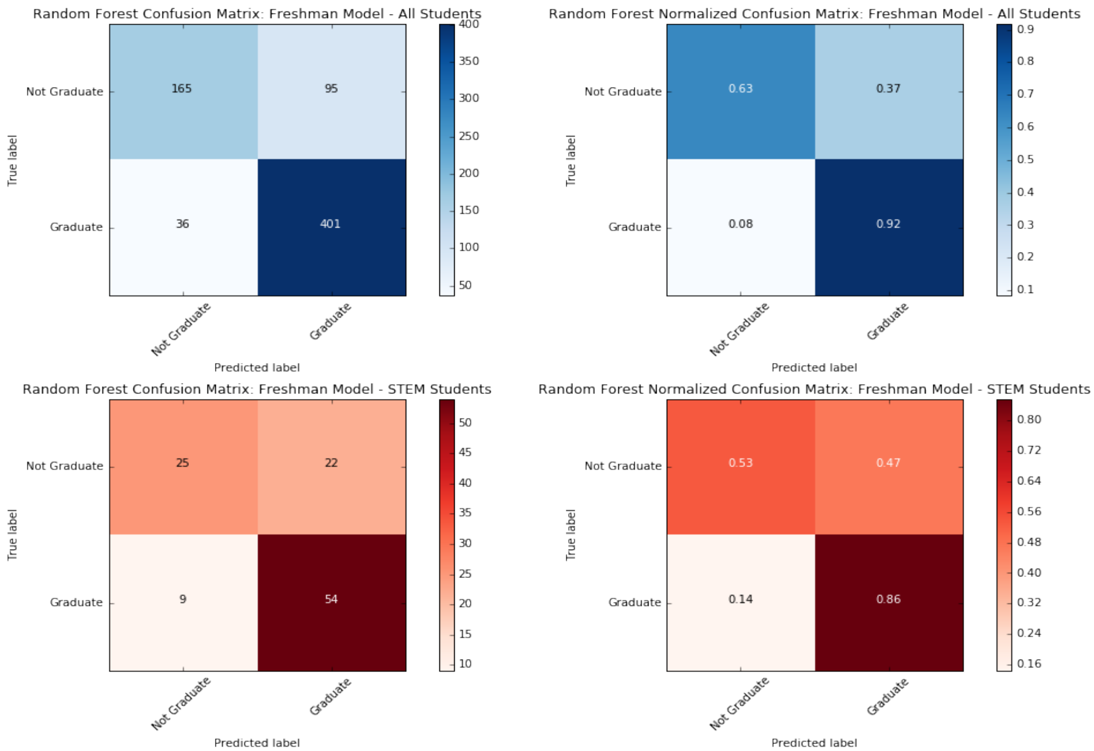

Figure 4.70: RF Confusion Matrices: Freshman Model

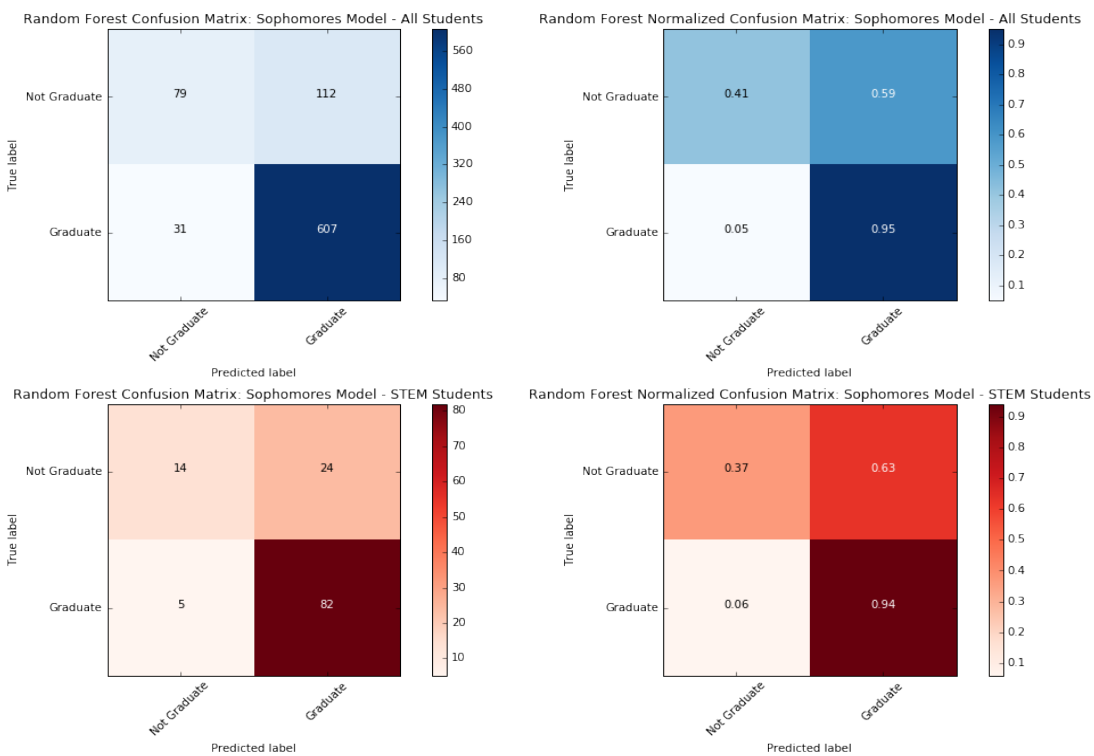

Figure 4.71: RF Confusion Matrices: Sophomore Model

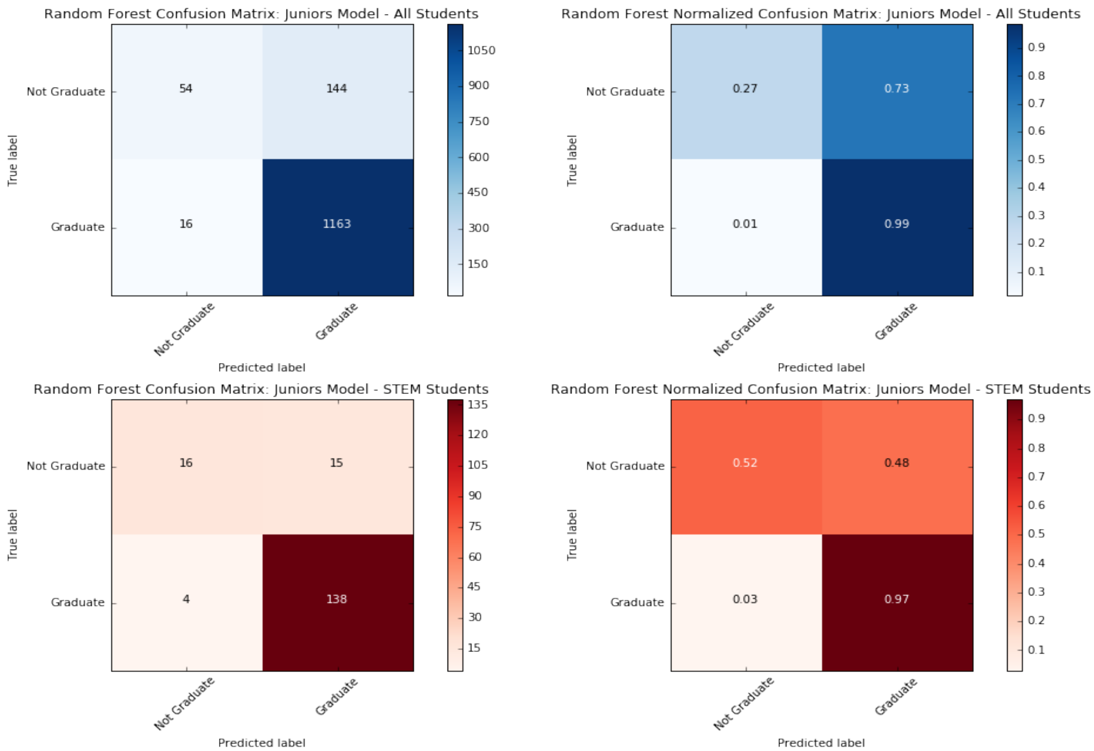

Figure 4.72: RF Confusion Matrices: Junior Model

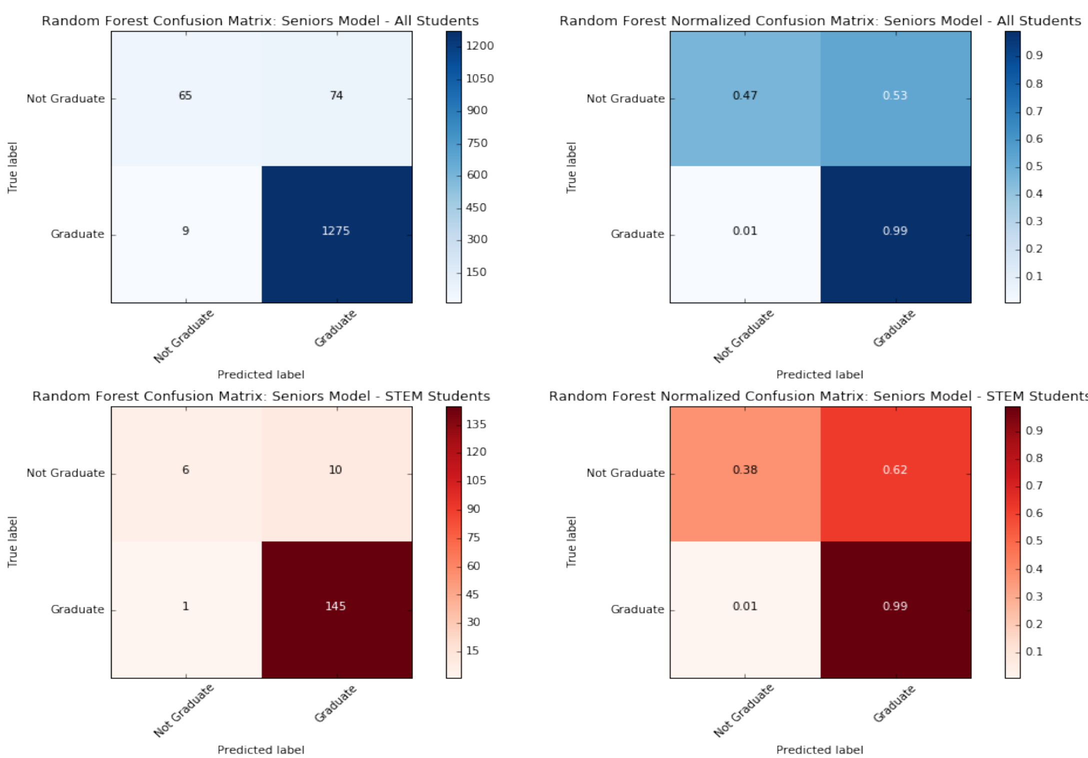

Figure 4.73: RF Confusion Matrices: Senior Model

Precision, Recall, F-Scores

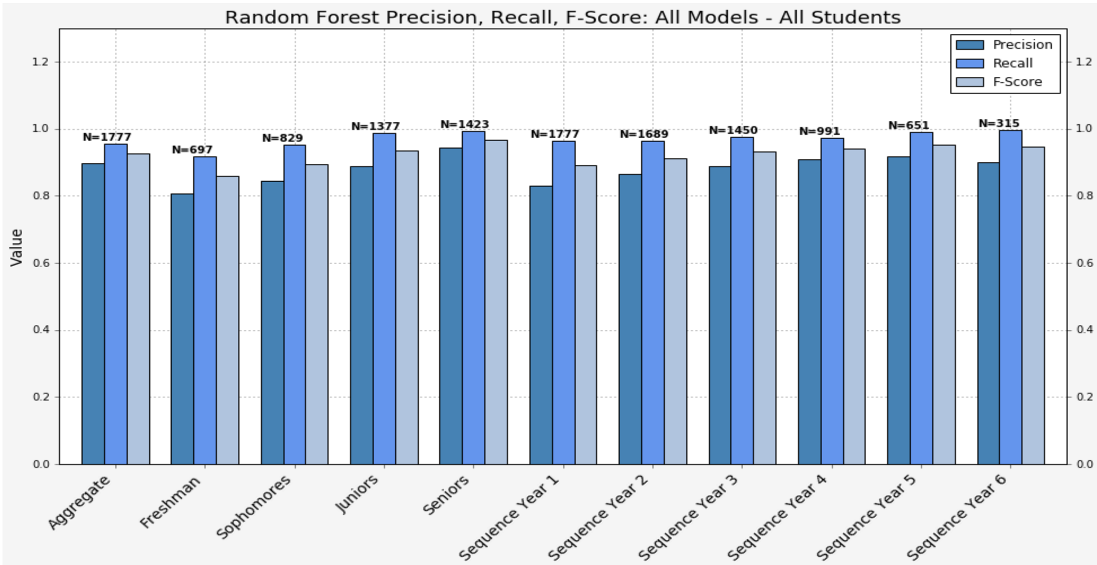

Figure 4.74: RF Precision, Recall, and F-scores: All Models, All Students

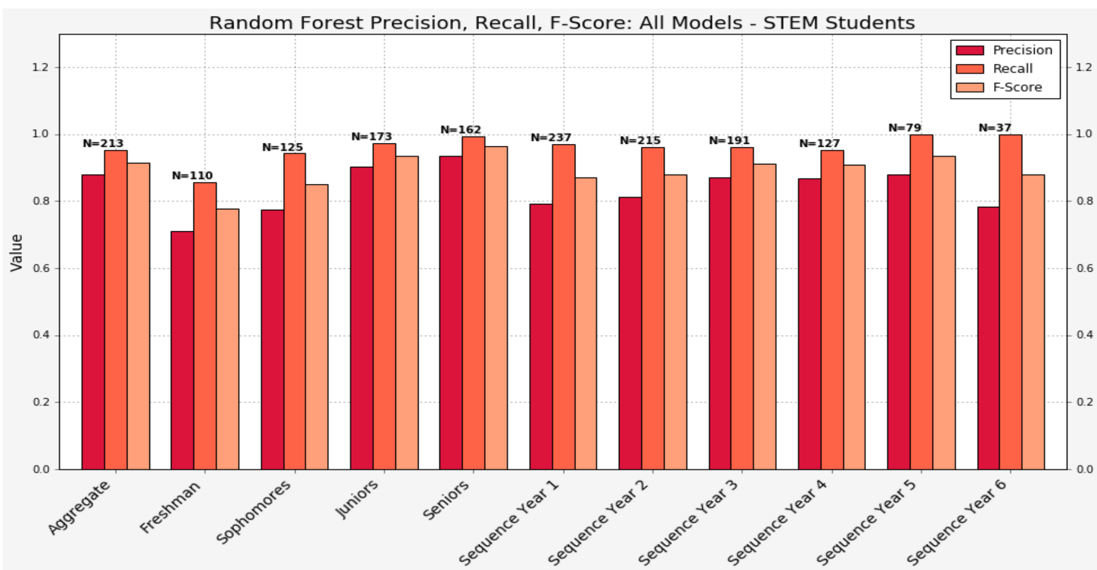

Figure 4.75: RF Precision, Recall, and F-scores: All Models, STEM Students

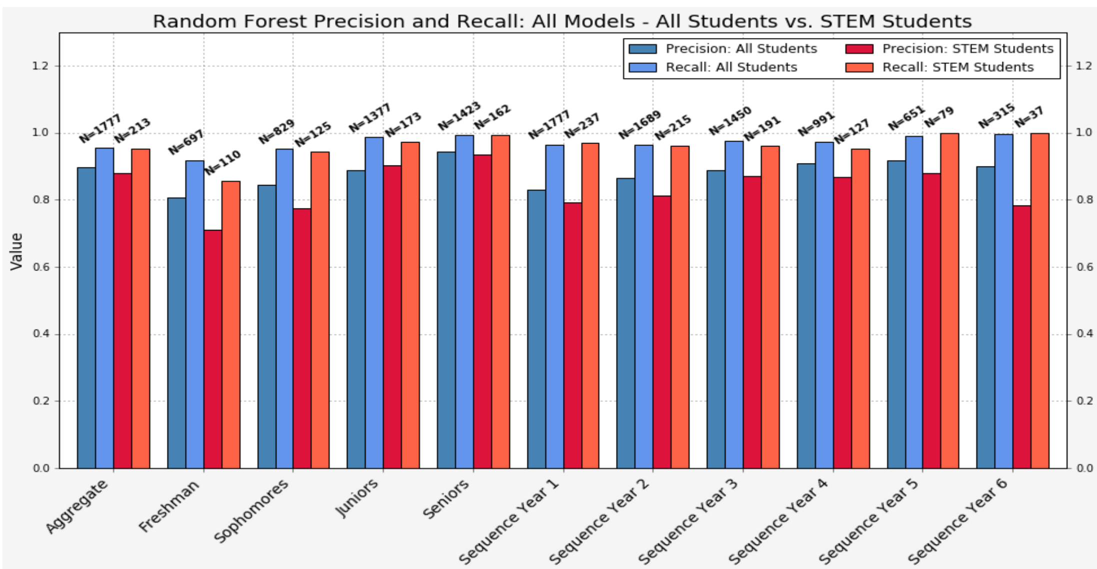

Figure 4.76: RF Precision and Recall: All Models, All Students vs STEM Students

Precision-Recall Curves

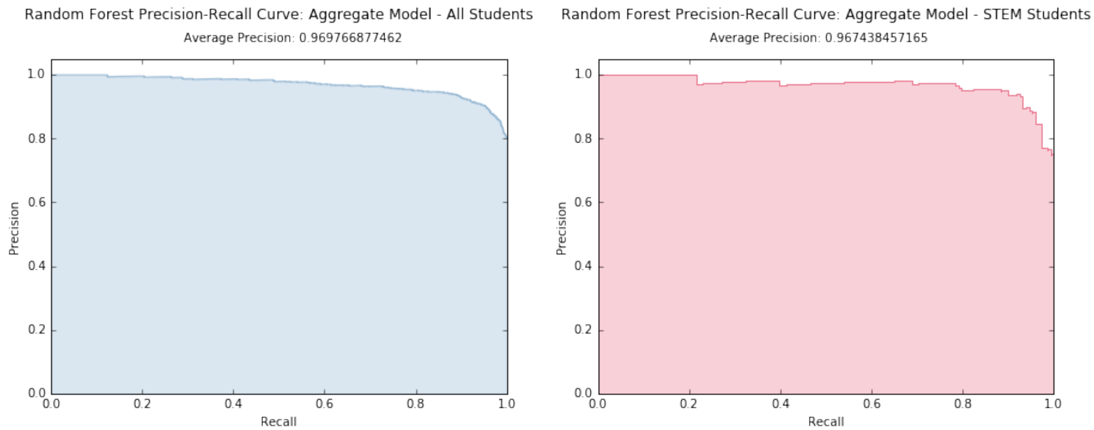

Figure 4.77: Precision-Recall Curve: Aggregate Model

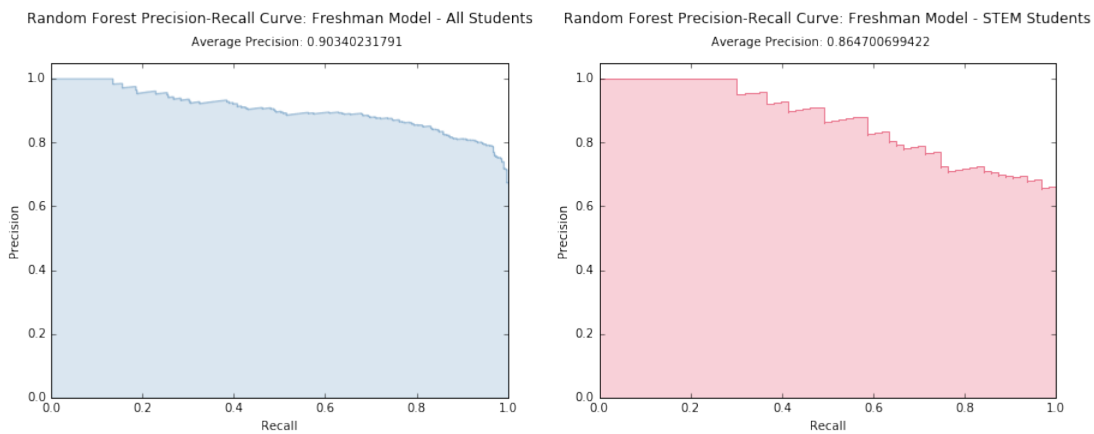

Figure 4.78: Precision-Recall Curve: Freshman Model

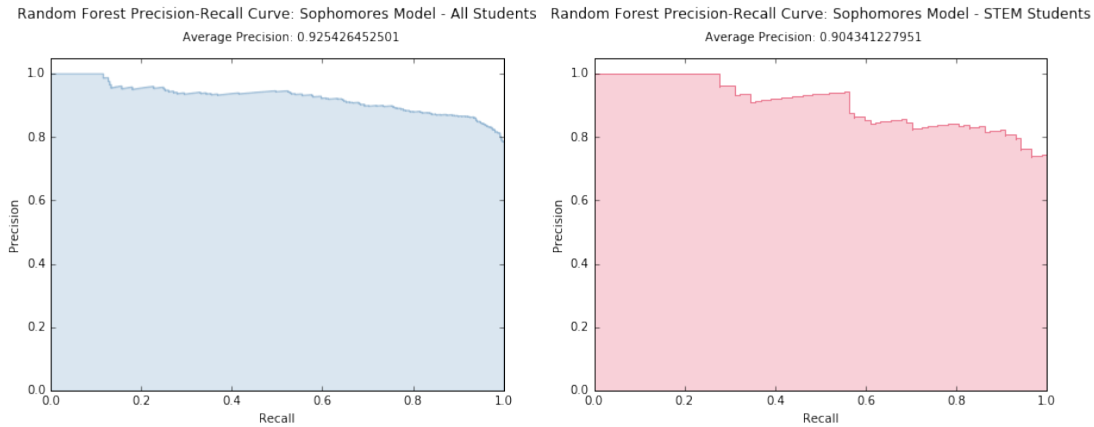

Figure 4.79: Precision-Recall Curve: Sophomore Model

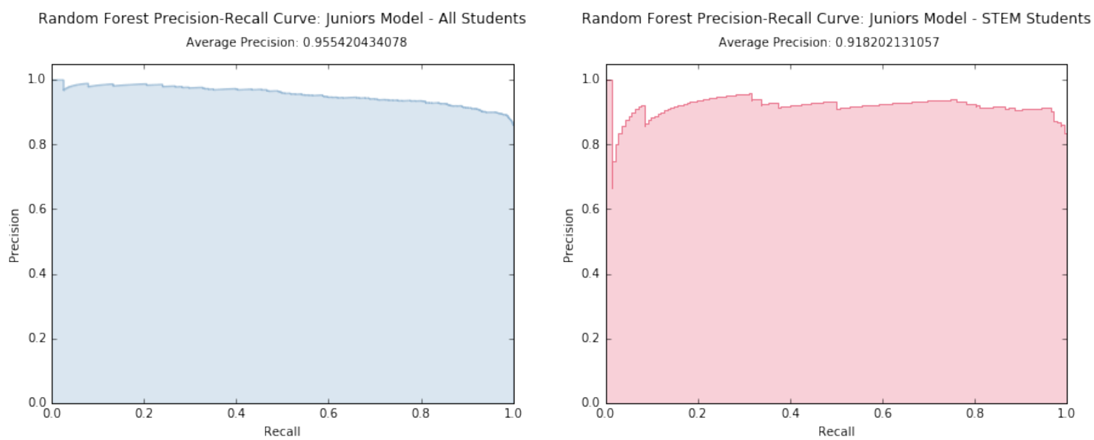

Figure 4.80: Precision-Recall Curve: Junior Model

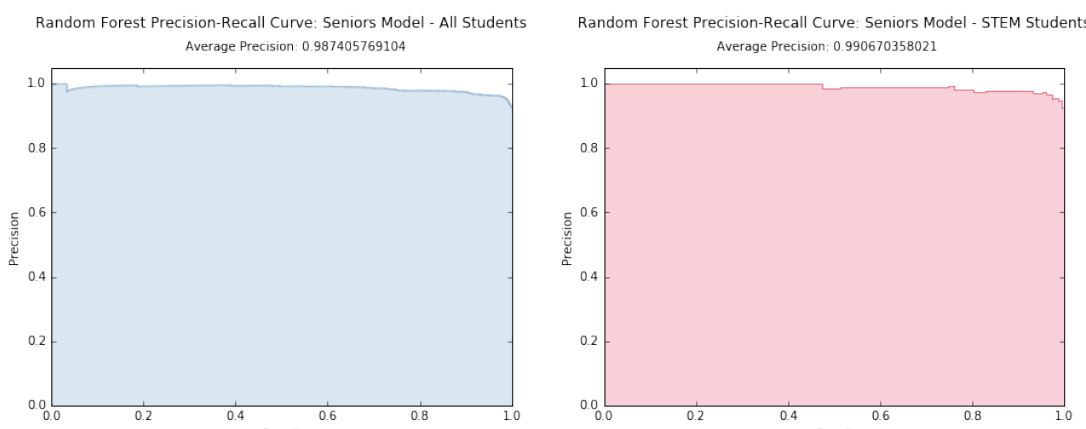

Figure 4.81: Precision-Recall Curve: Senior Model

ROC Curves

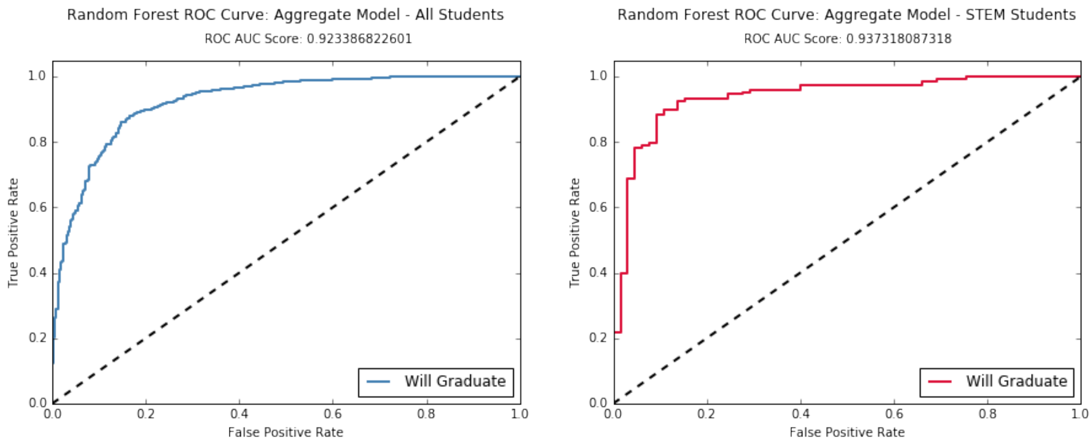

Figure 4.82: ROC Curve: Aggregate Model

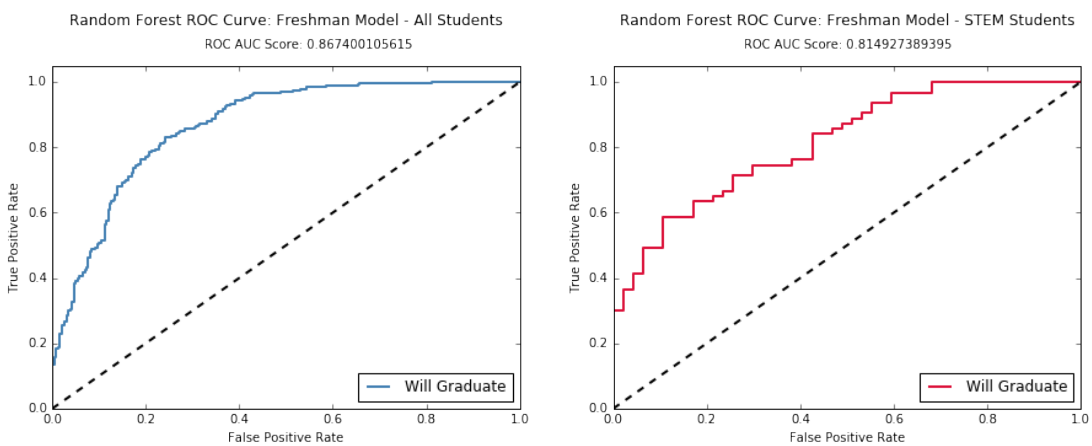

Figure 4.83: ROC Curve: Freshman Model

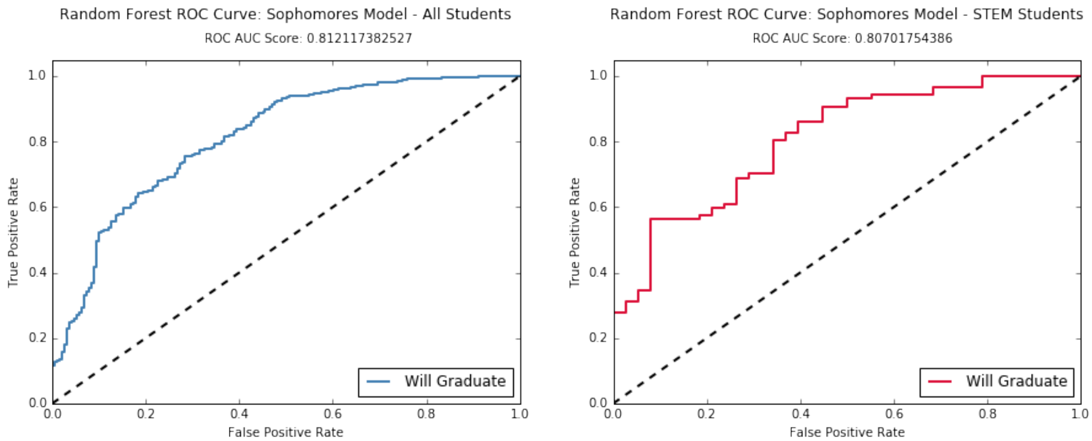

Figure 4.84: ROC Curve: Sophomore Model

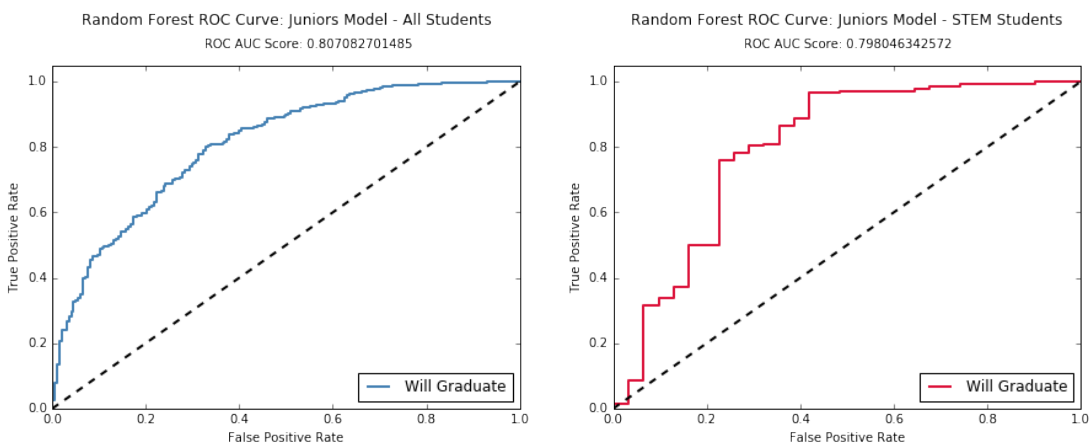

Figure 4.85: ROC Curve: Junior Model

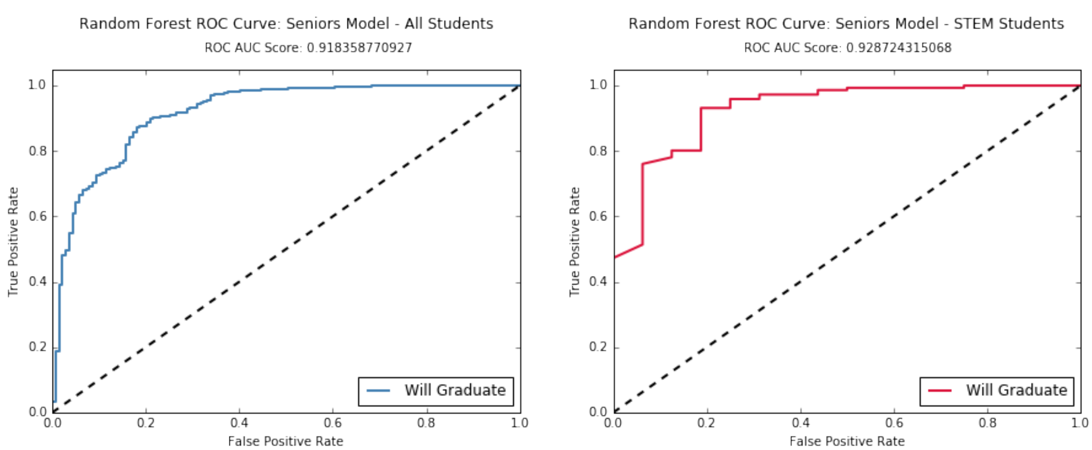

Figure 4.86: ROC Curve: Senior Model

Top Features

The top twenty-five features for each model are shown below; top features are determined by the absolute value of the model coefficients, indicating features with the strongest influence on final classification.

Top Features: Aggregate Model

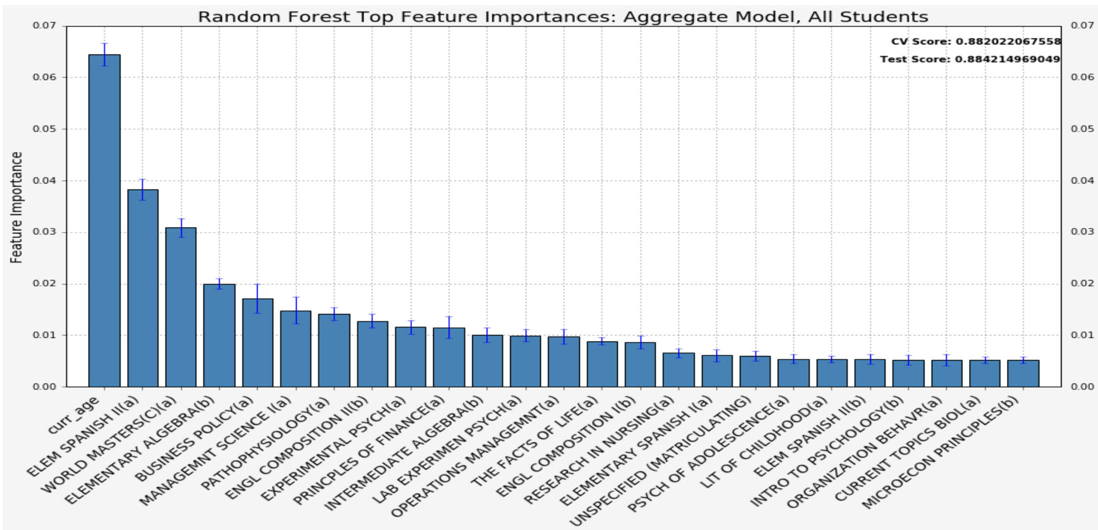

Figure 4.87: Top Features: Aggregate Model, All Students

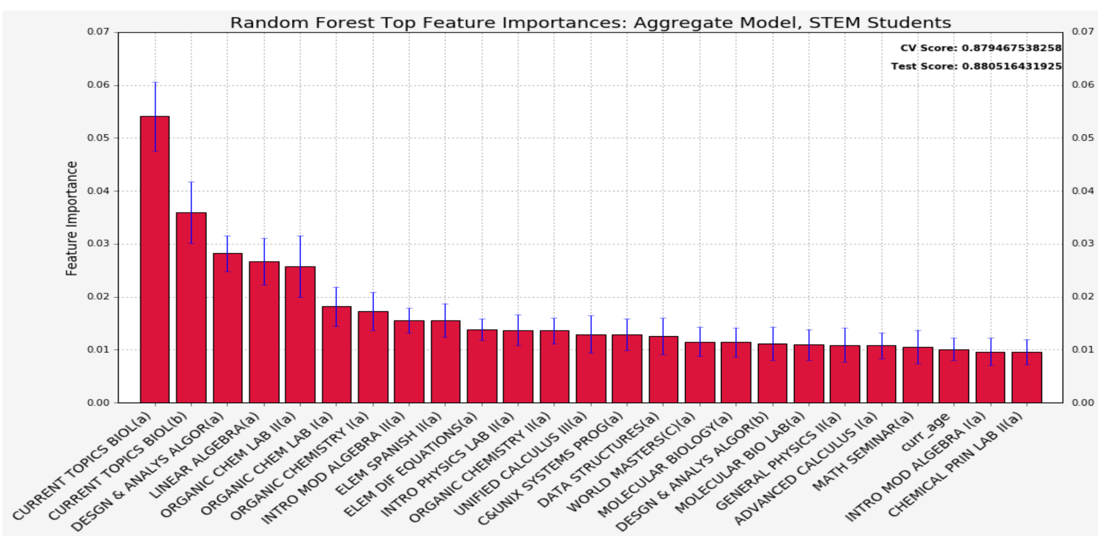

Figure 4.88: Top Features: Aggregate Model, STEM Students

Top Features: Freshman Model

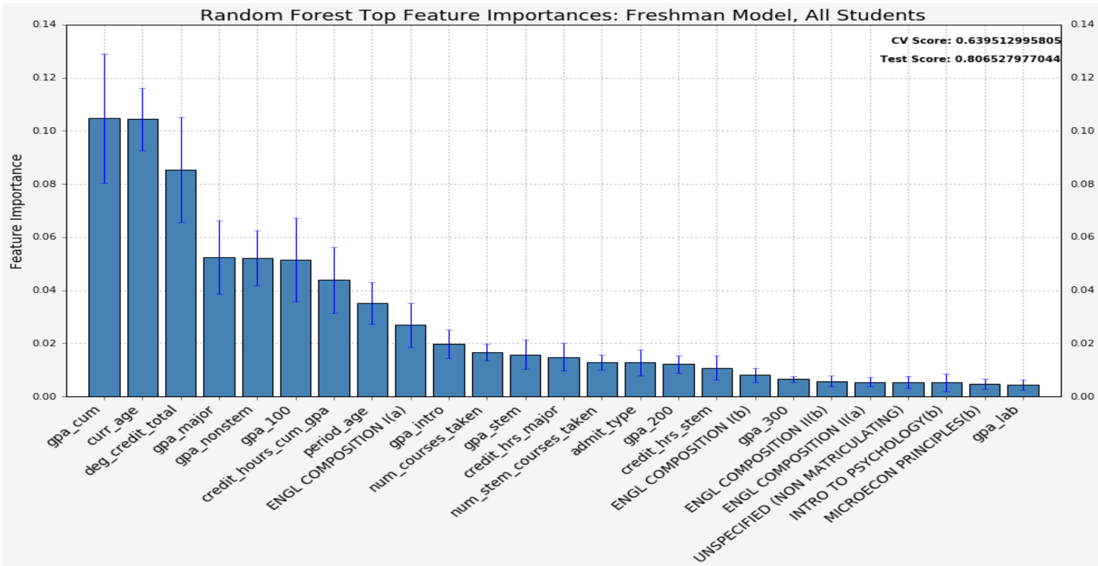

Figure 4.89: Top Features: Freshman Model, All Students

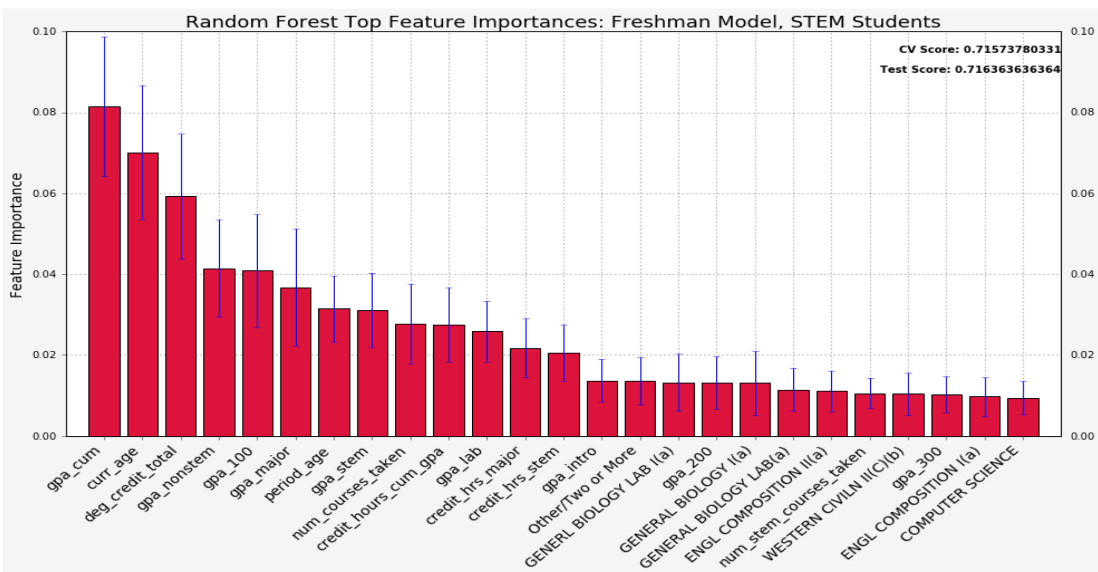

Figure 4.90: Top Features: Freshman Model, STEM Students

Top Features: Sophomores Model

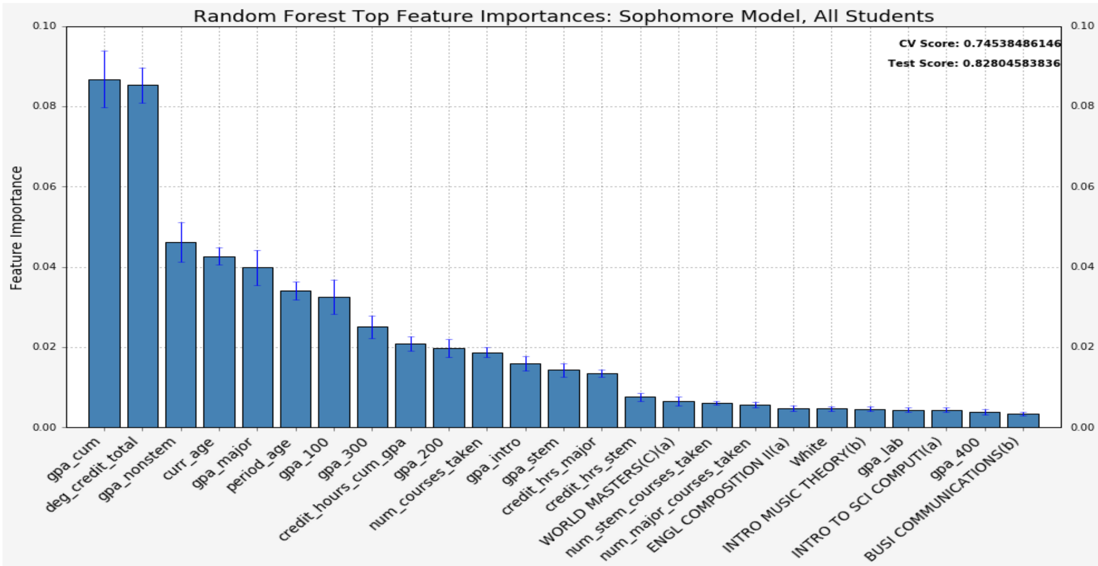

Figure 4.91: Top Features: Sophomores Model, All Students

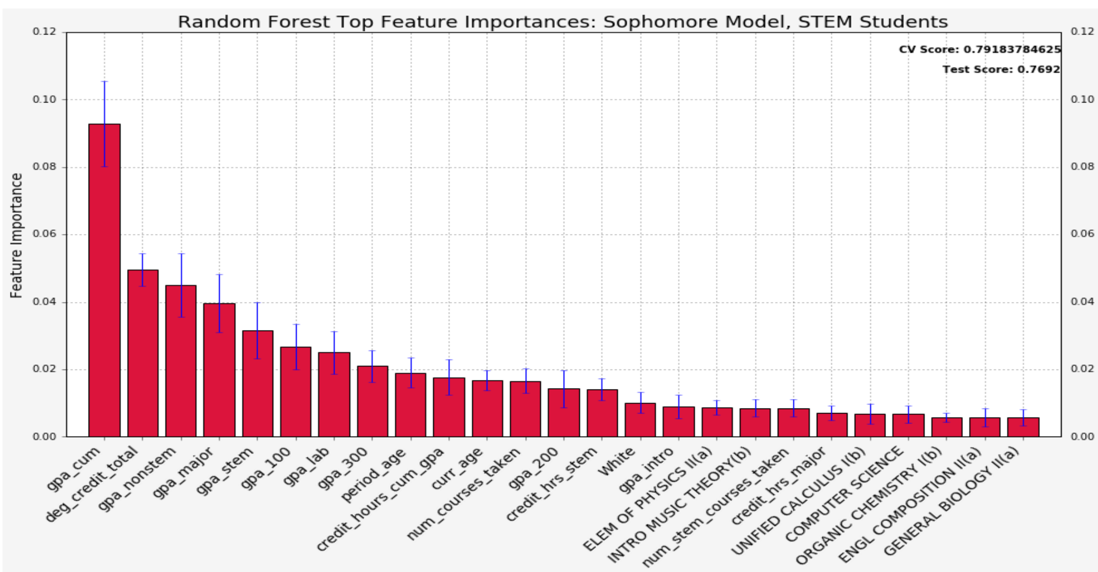

Figure 4.92: Top Features: Sophomores Model, STEM Students

Top Features: Juniors Model

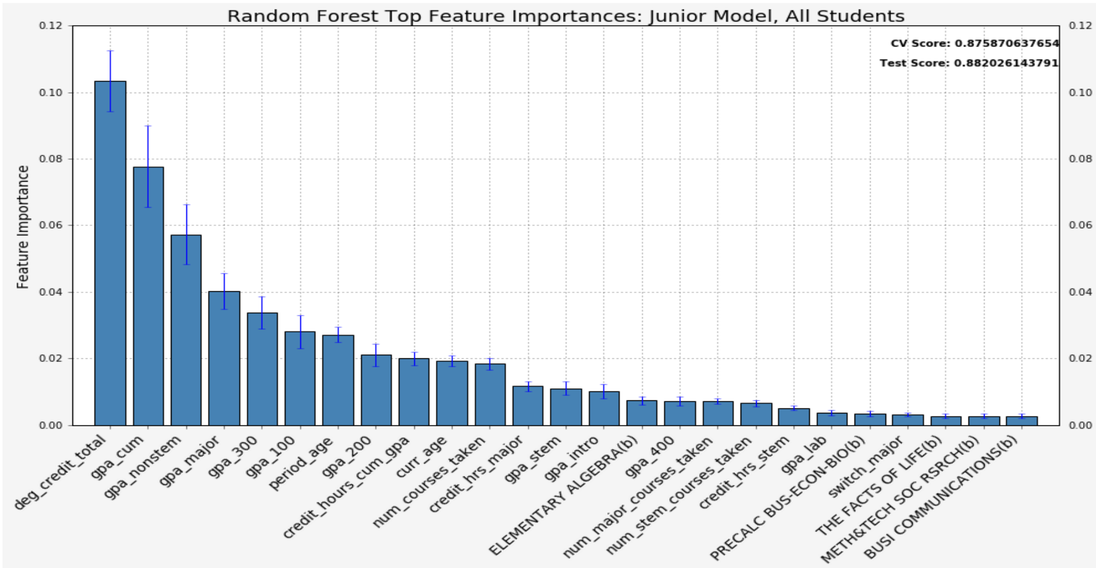

Figure 4.93: Top Features: Juniors Model, All Students

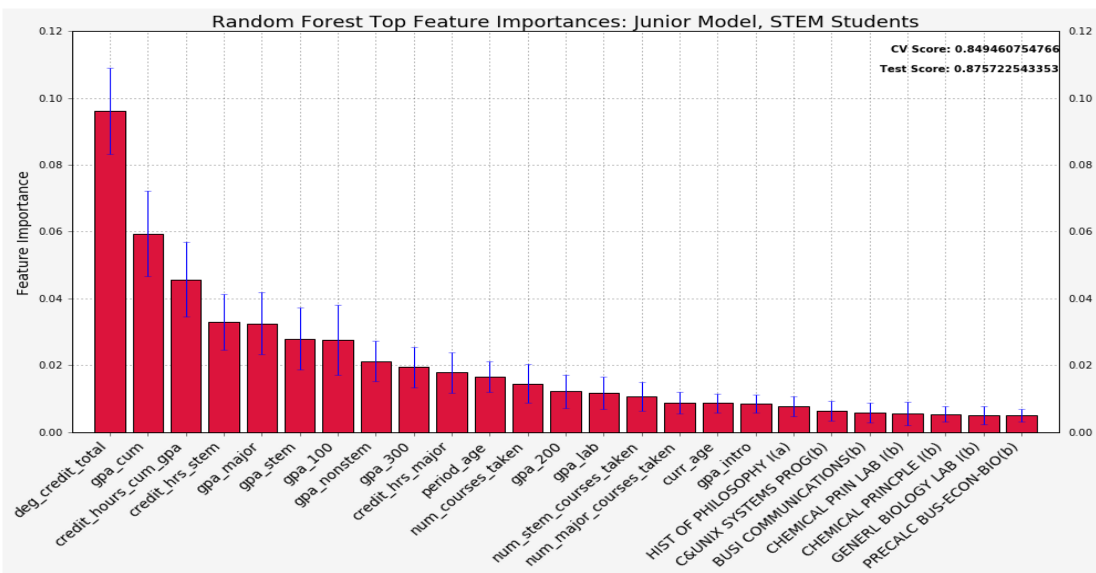

Figure 4.94: Top Features: Juniors Model, STEM Students

Top Features: Seniors Model

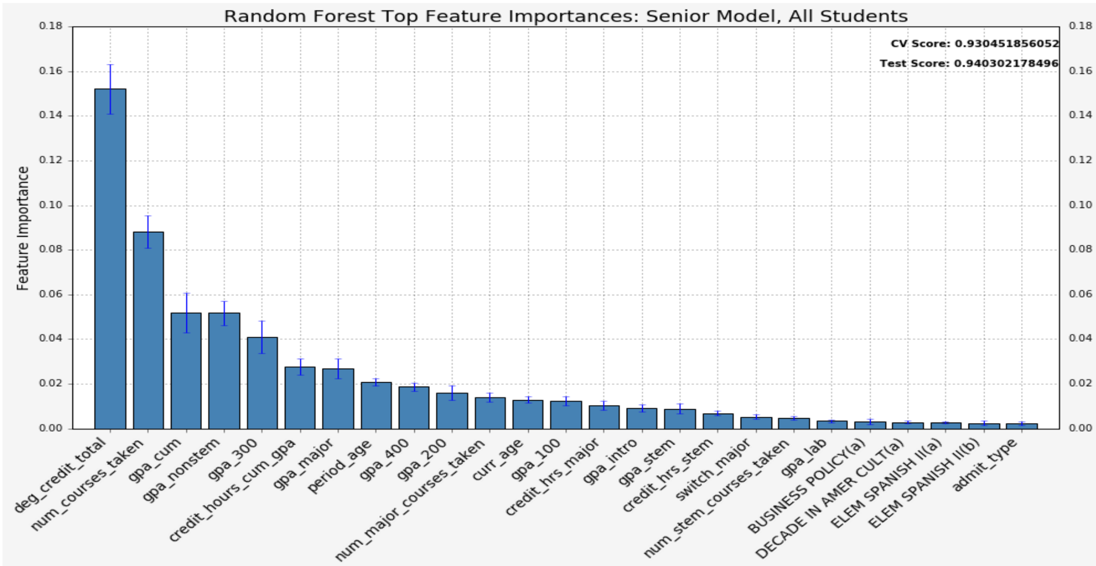

Figure 4.95: Top Features: Seniors Model, All Students

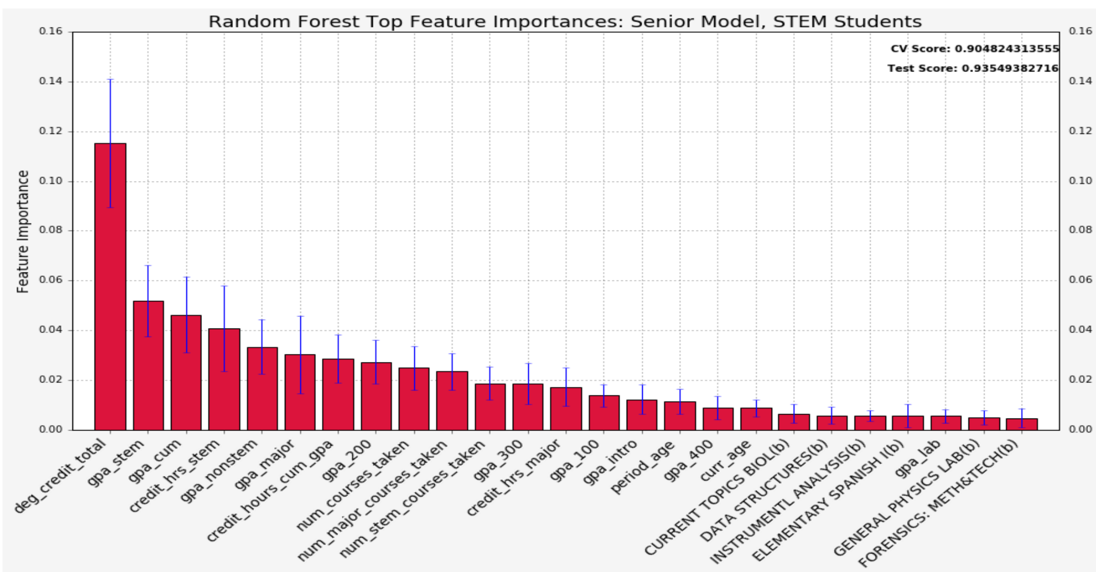

Figure 4.96: Top Features: Seniors Model, STEM Students

4.2.4 Comparison Across Classifiers

Accuracy

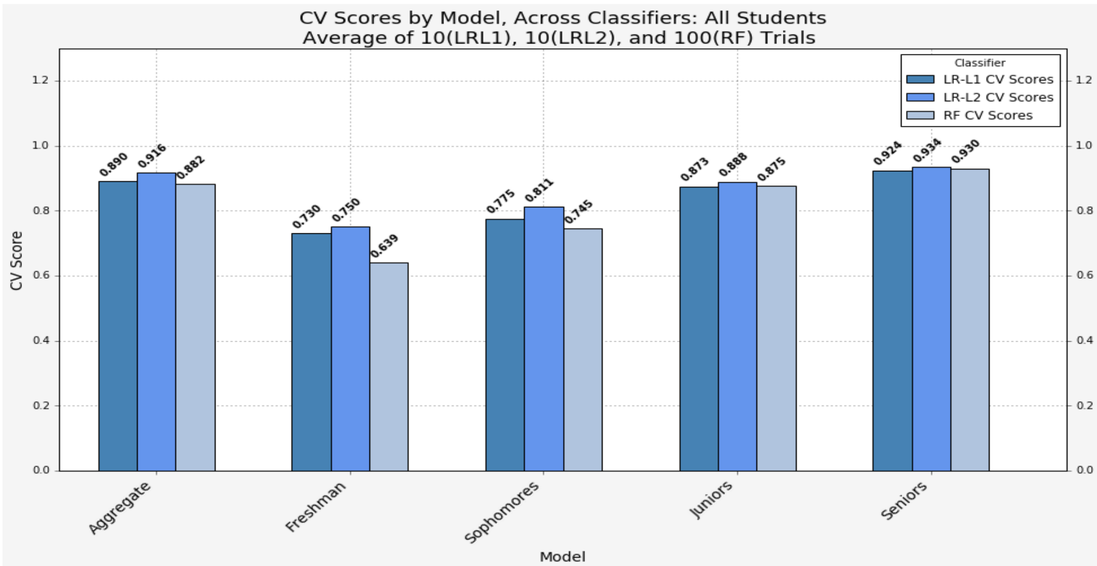

Figure 4.97: CV Scores: Across Classifiers, All Students

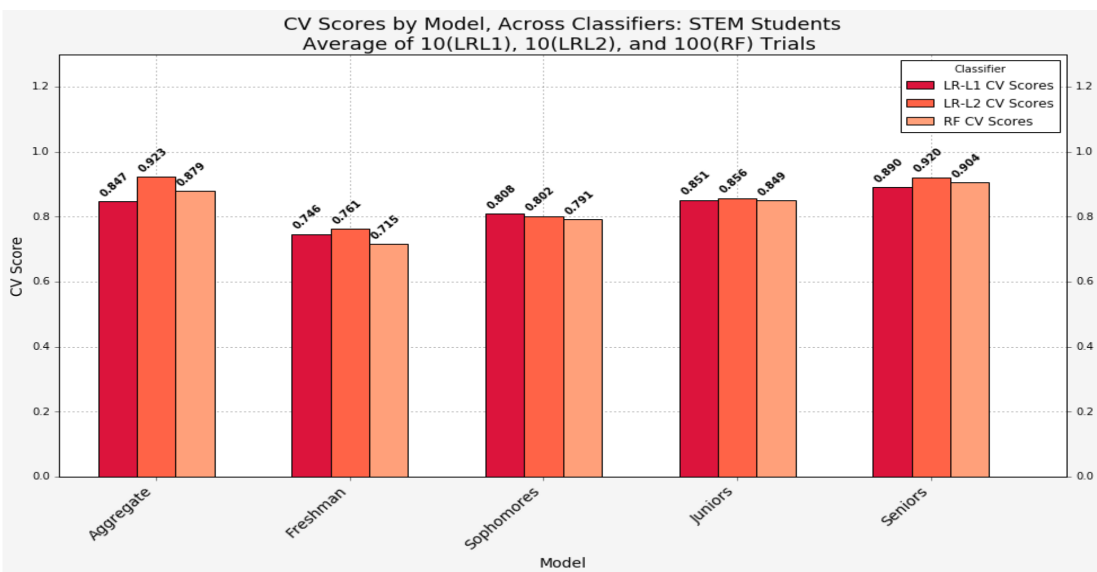

Figure 4.98: CV Scores: Across Classifiers, STEM Students

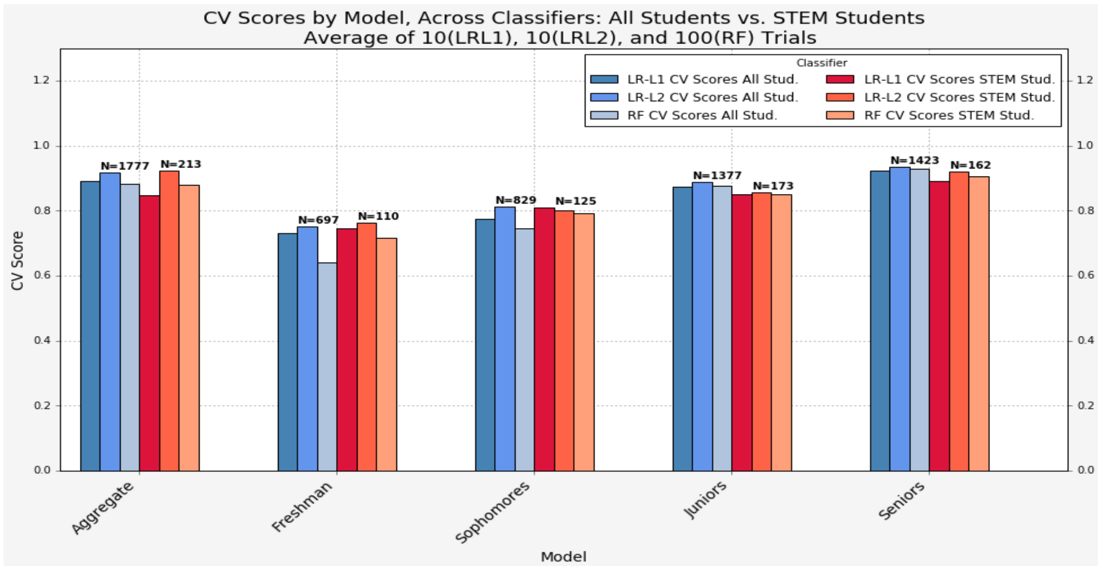

Figure 4.99: CV Scores: Across Classifiers, All Students vs STEM Students

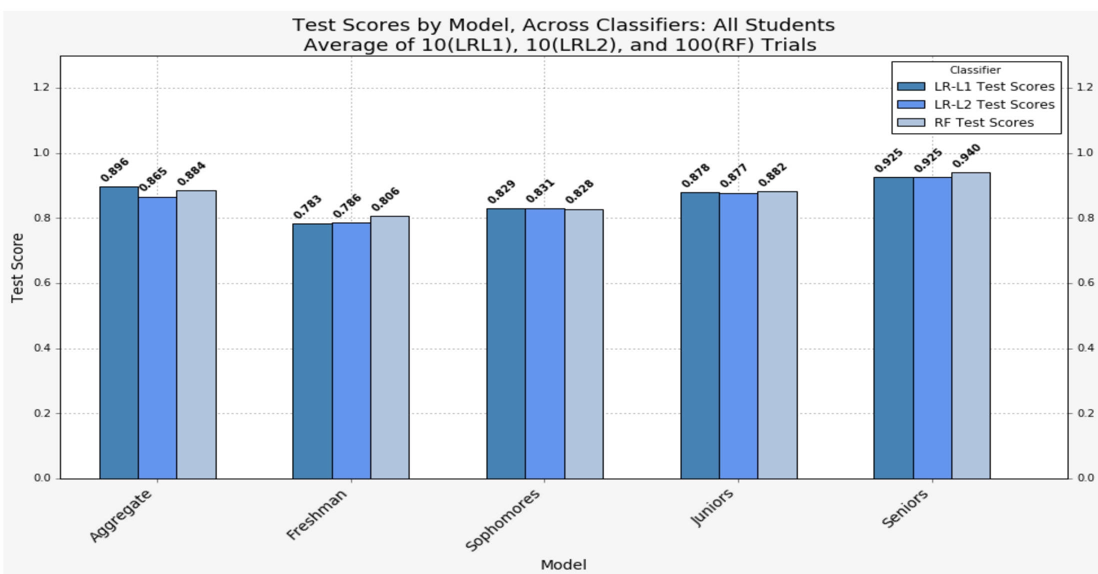

Figure 4.100: Test Scores: Across Classifiers, All Students

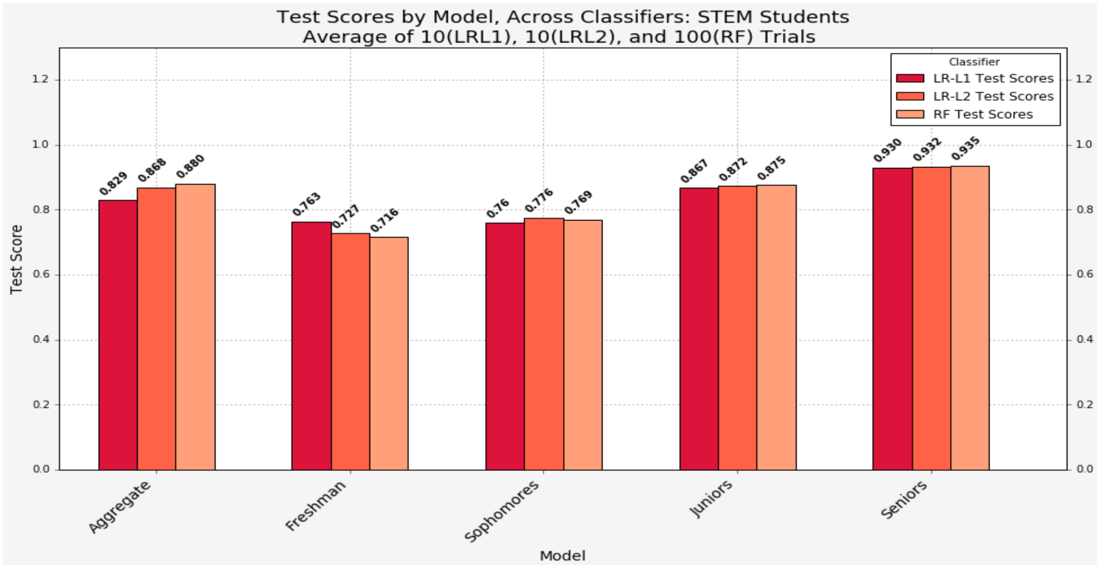

Figure 4.101: Test Scores: Across Classifiers, STEM Students

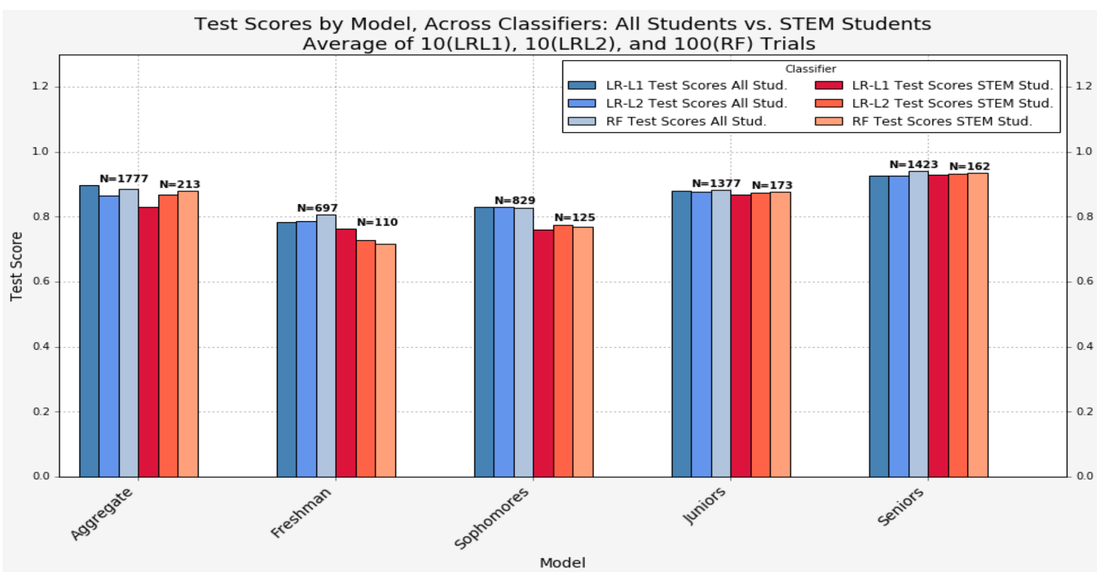

Figure 4.102: Test Scores: Across Classifiers, All Students vs STEM Students

Prediction Distributions

Confusion Matrices:

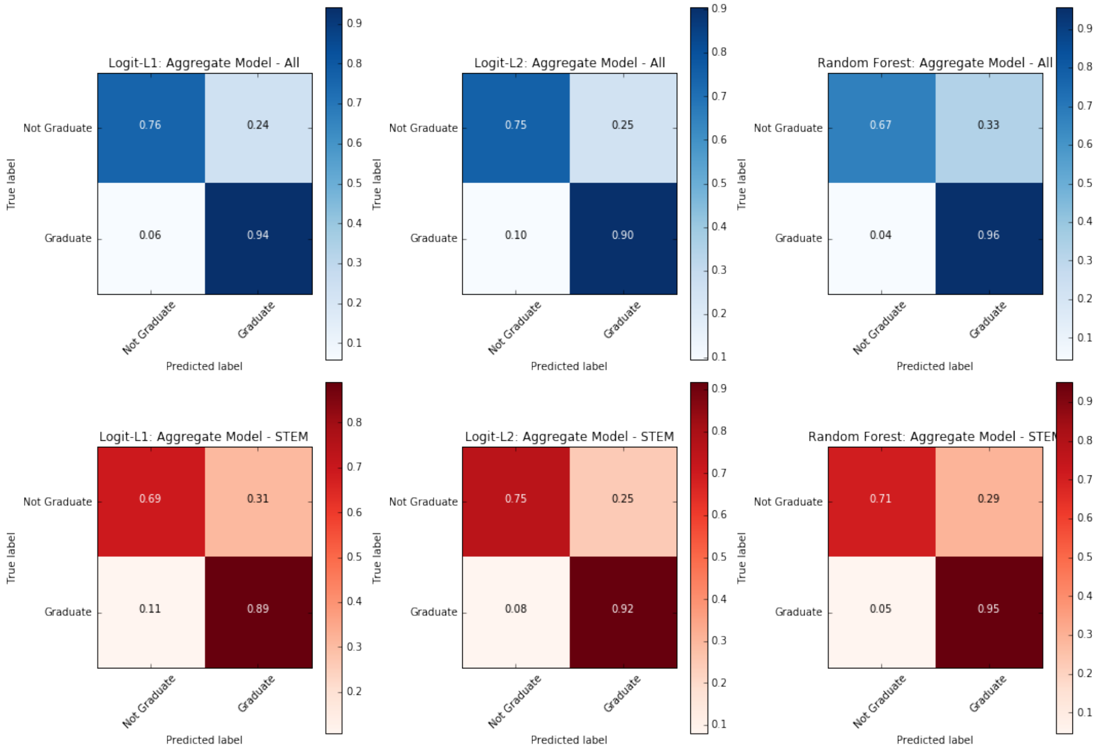

Figure 4.103: Confusion Matrices: Aggregate Model

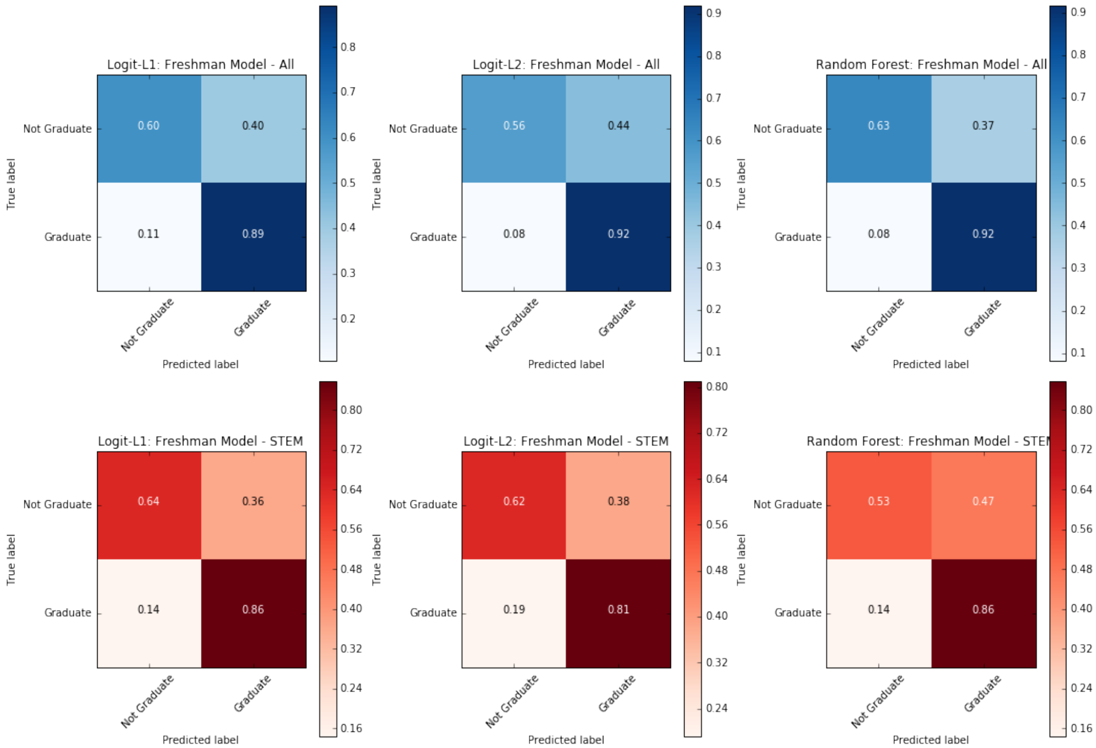

Figure 4.104: Confusion Matrices: Freshman Model

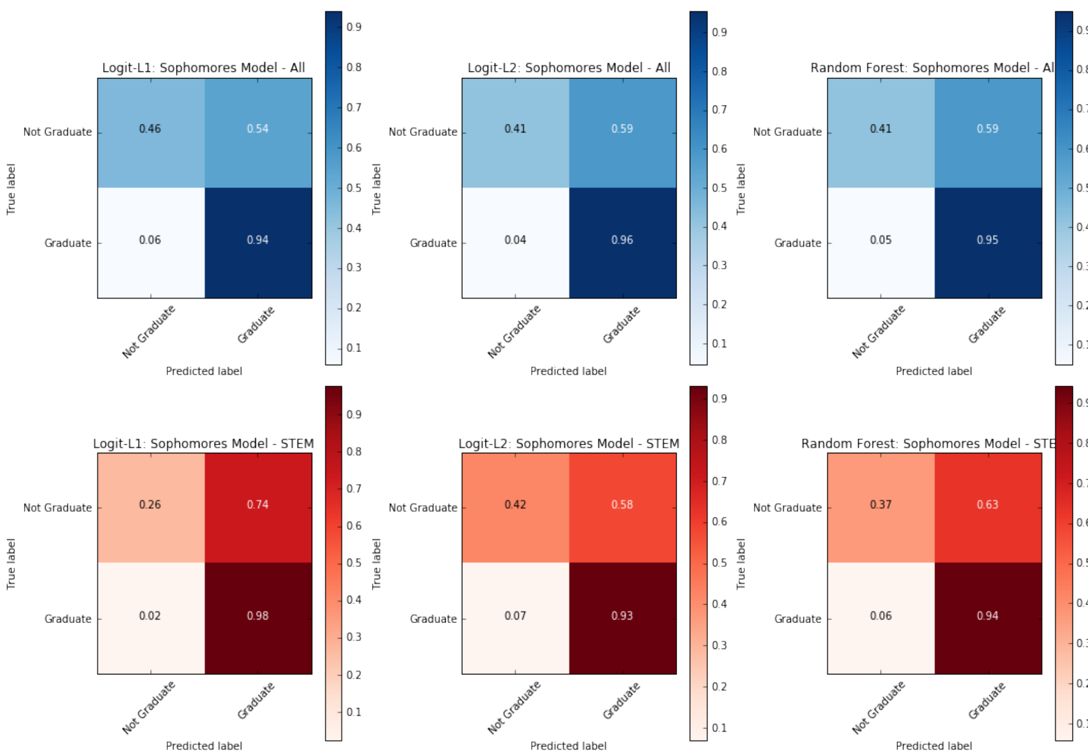

Figure 4.105: Confusion Matrices: Sophomore Model

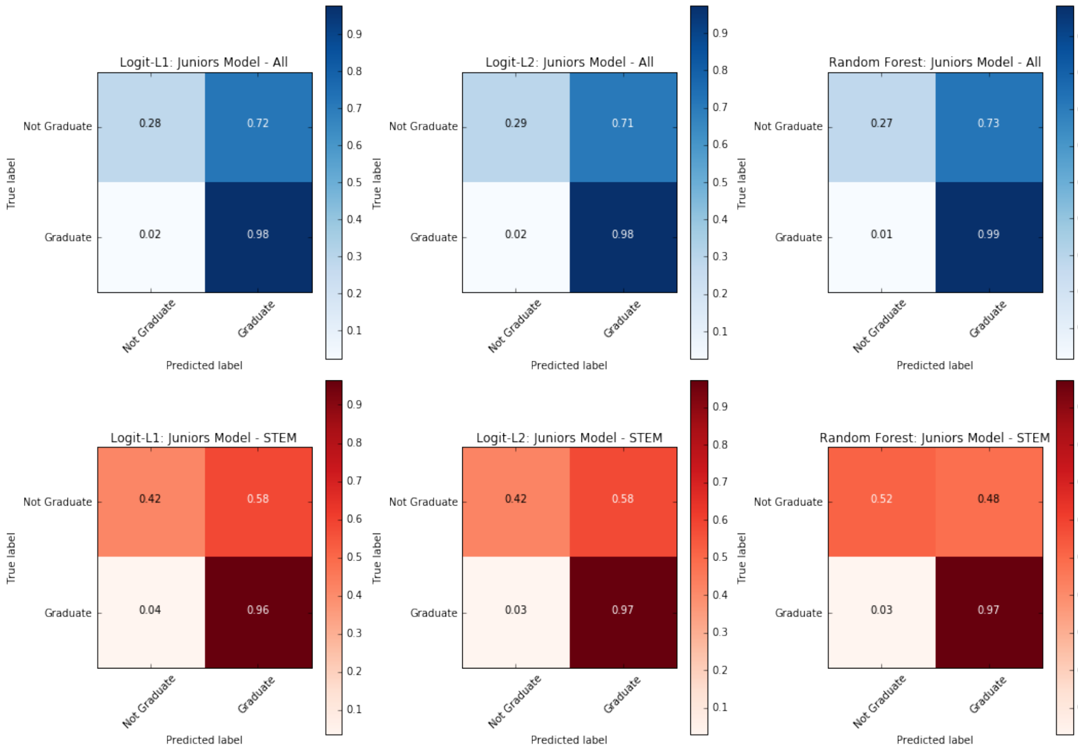

Figure 4.106: Confusion Matrices: Junior Model

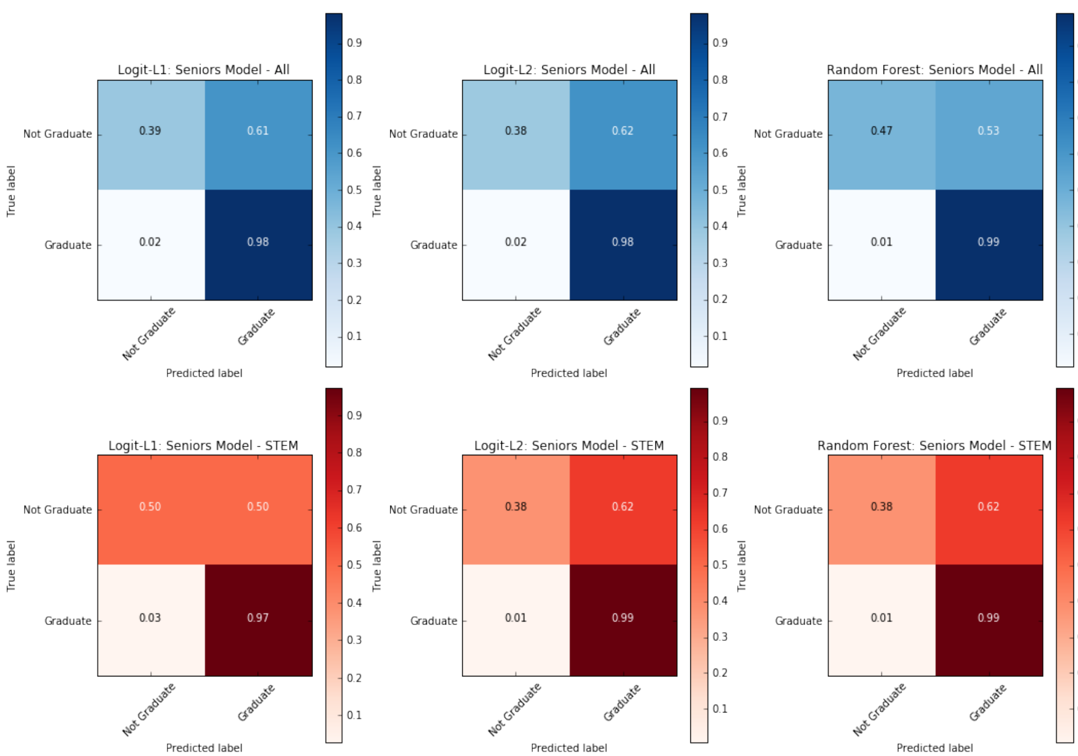

Figure 4.107: Confusion Matrices: Senior Model

Precision and Recall

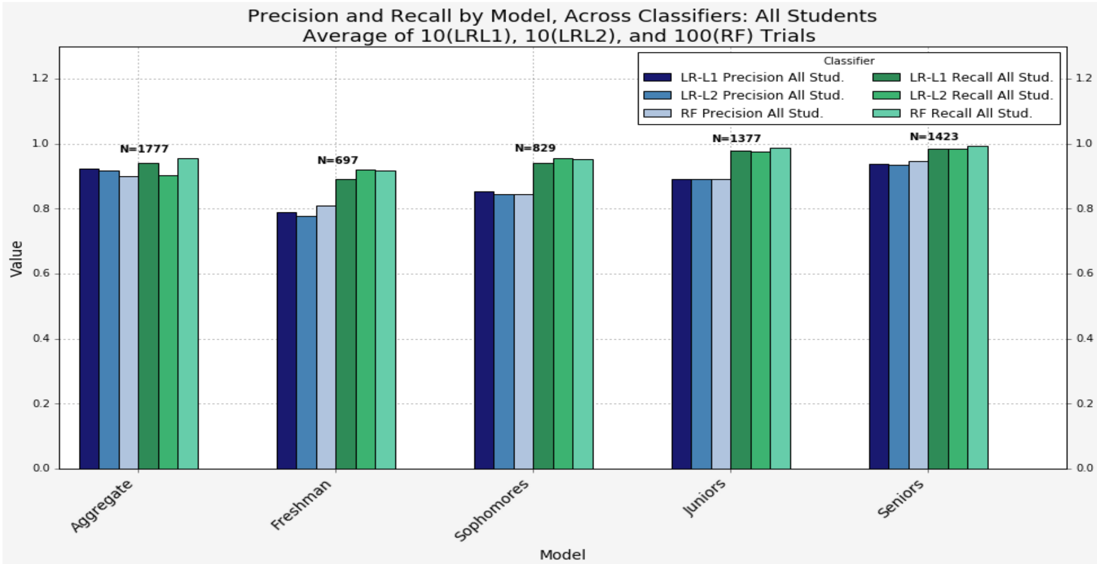

Figure 4.108: Precision and Recall: All Models, All Students

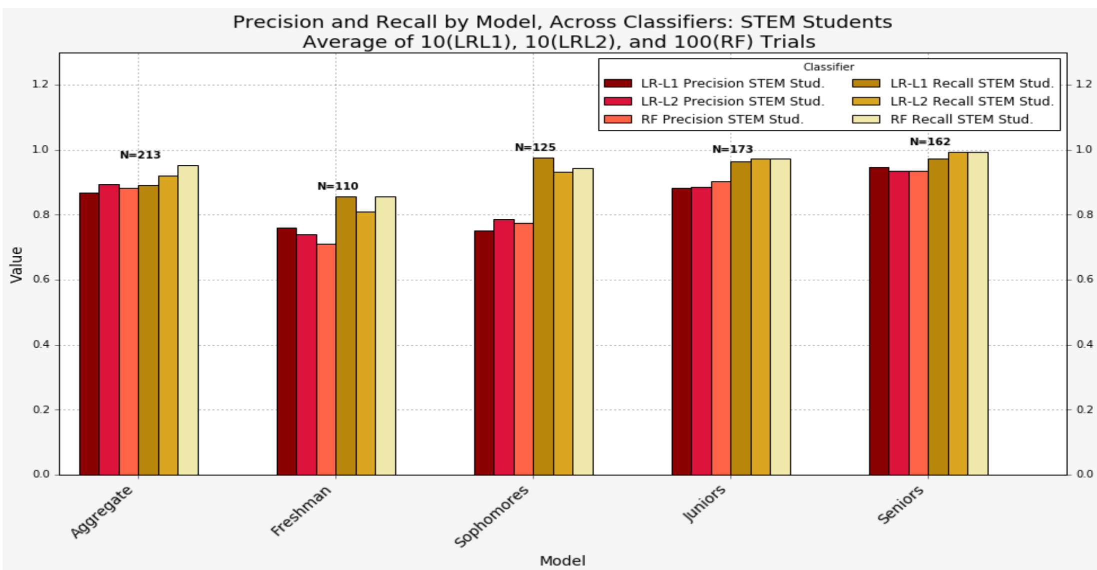

Figure 4.109: Precision and Recall: All Models, STEM Students

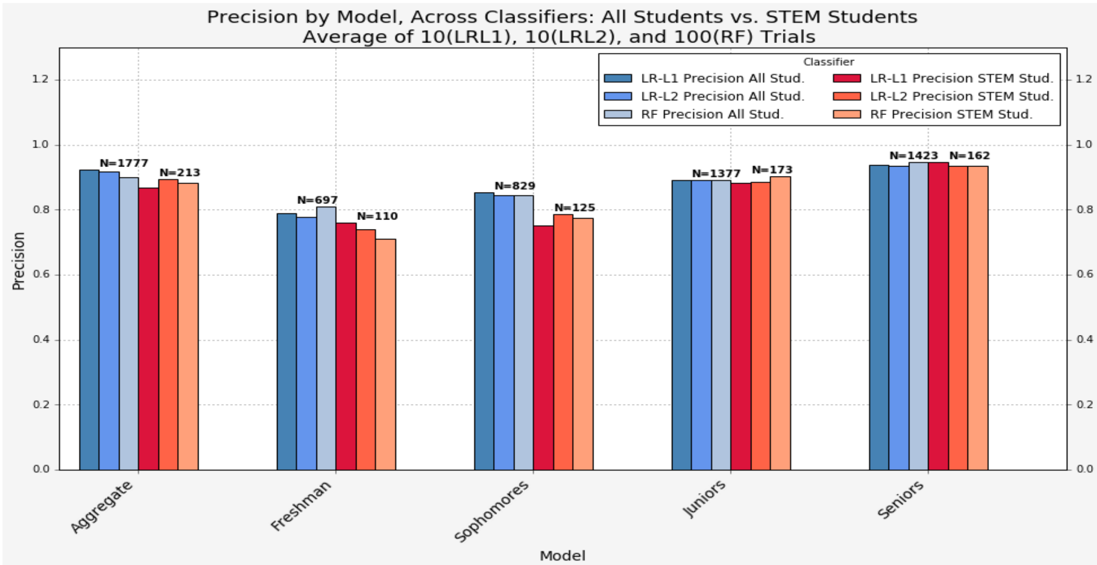

Figure 4.110: Precision: All Models, All Students vs STEM Students

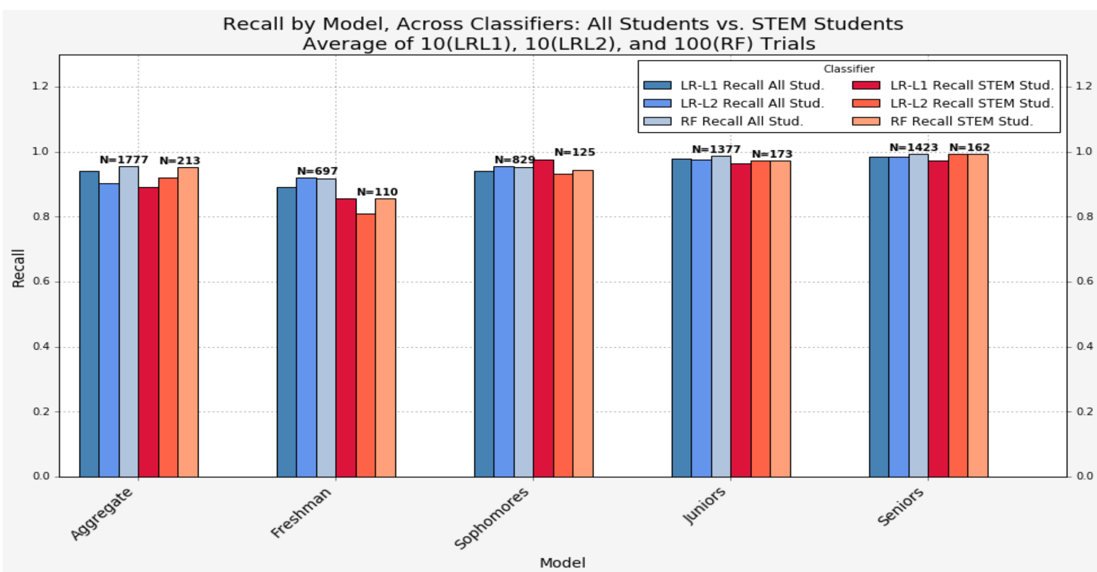

Figure 4.111: Recall: All Models, All Students vs STEM Students

Precision-Recall Curves

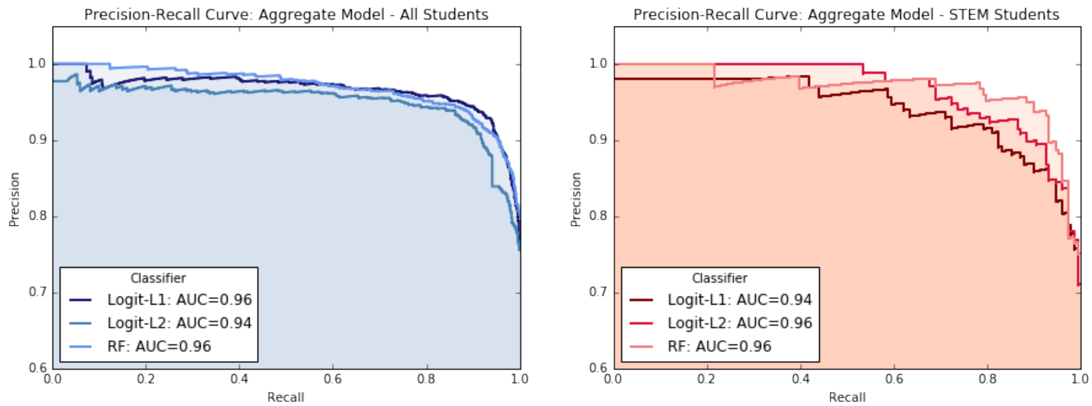

Figure 4.112: Precision-Recall Curve: Aggregate Model

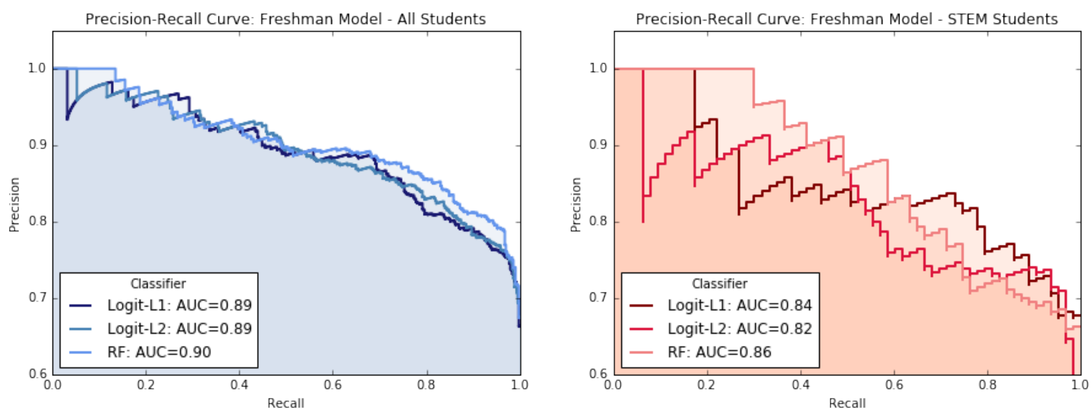

Figure 4.113: Precision-Recall Curve: Freshman Model

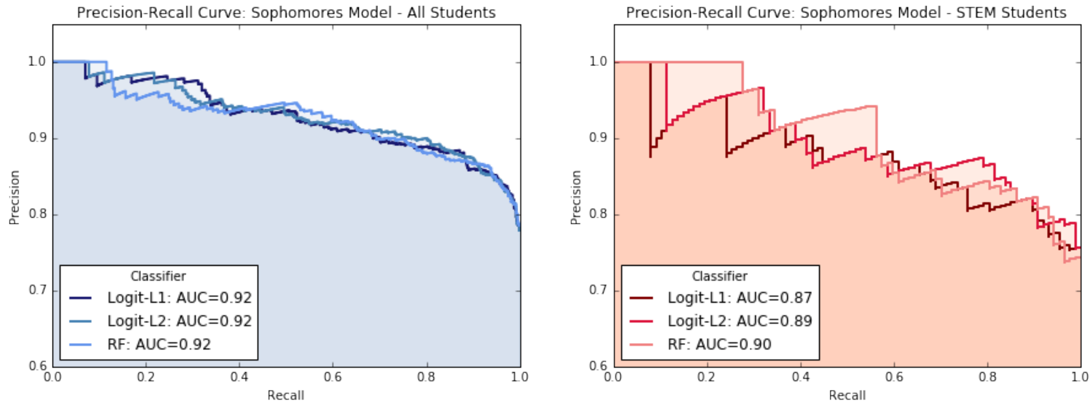

Figure 4.114: Precision-Recall Curve: Sophomore Model

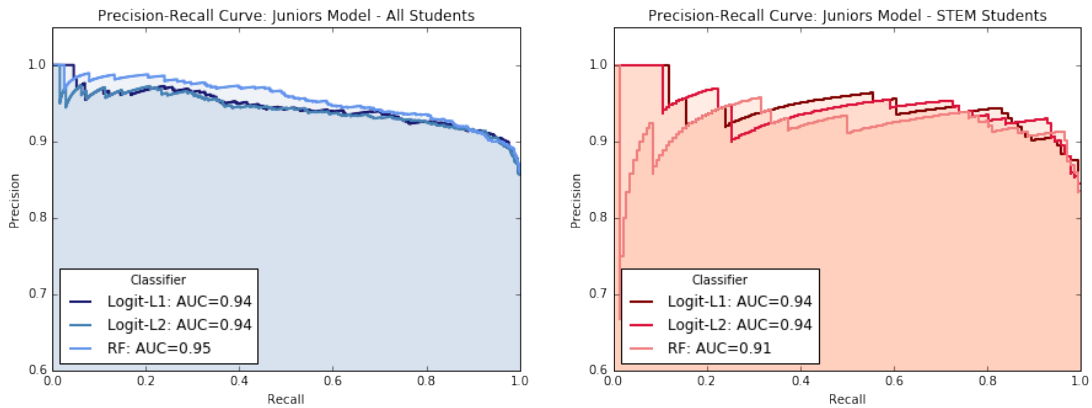

Figure 4.115: Precision-Recall Curve: Junior Model

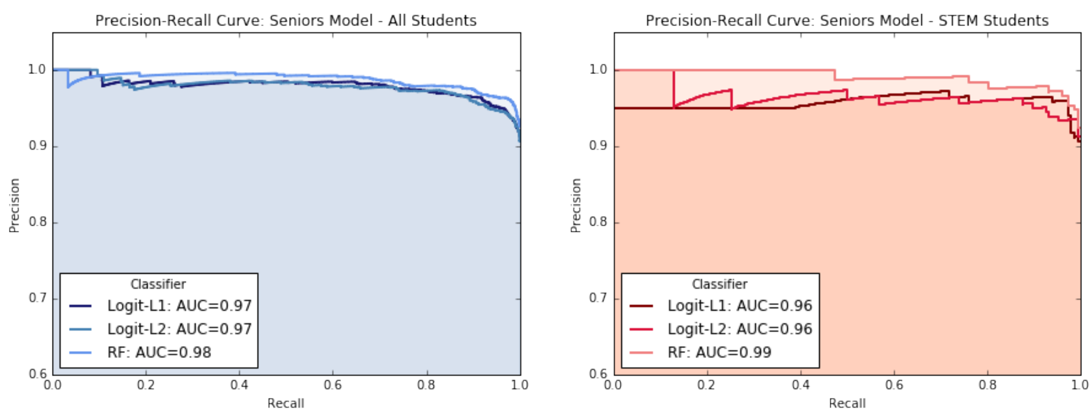

Figure 4.116: Precision-Recall Curve: Senior Model

ROC Curves

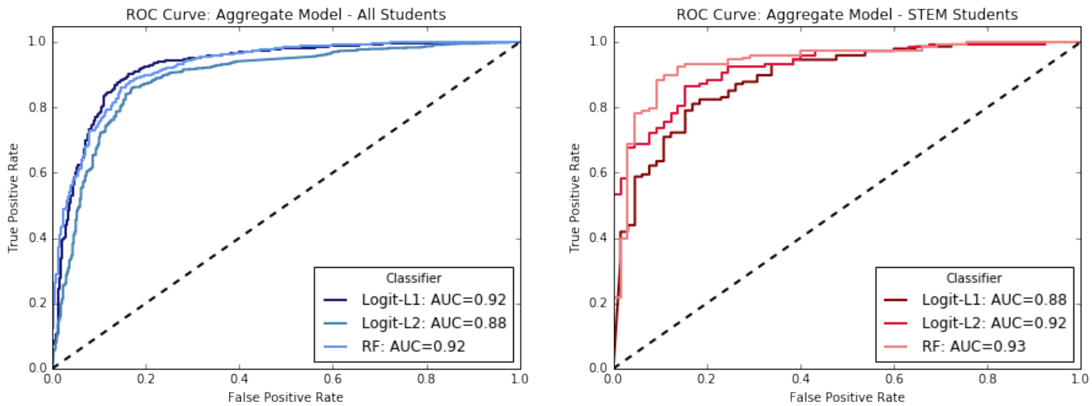

Figure 4.117: ROC Curve: Aggregate Model

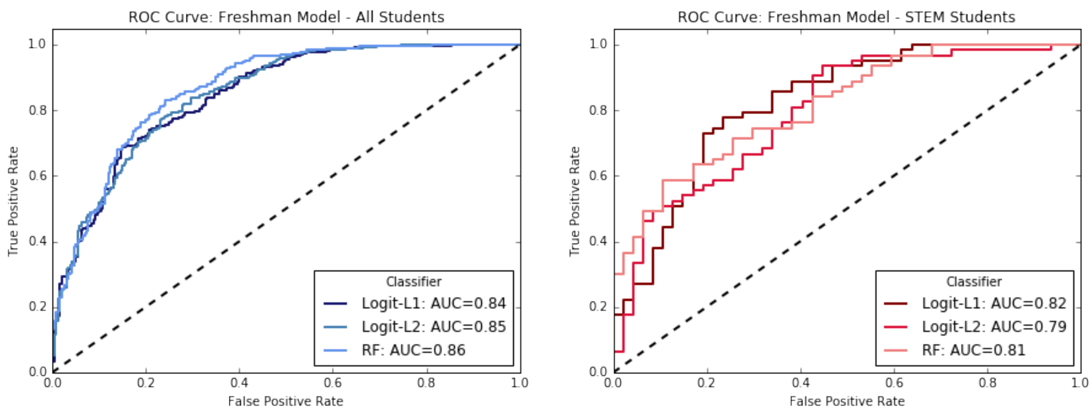

Figure 4.118: ROC Curve: Freshman Model

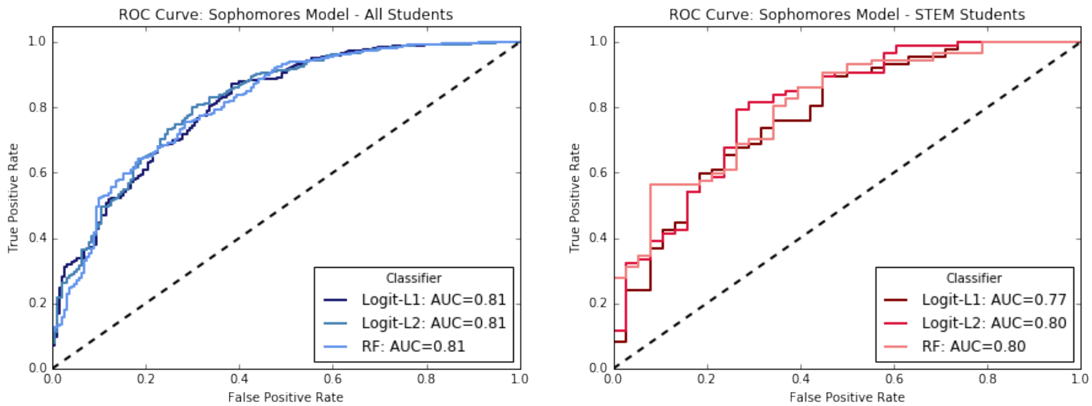

Figure 4.119: ROC Curve: Sophomore Model

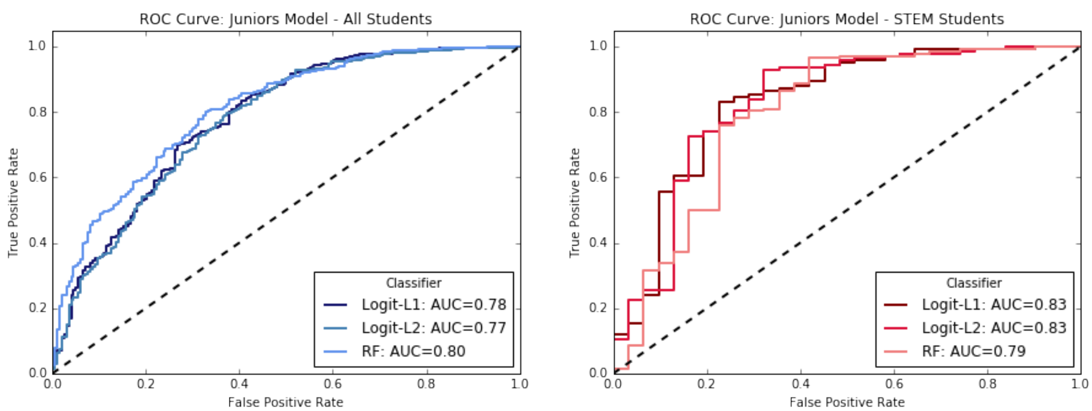

Figure 4.120: ROC Curve: Junior Model

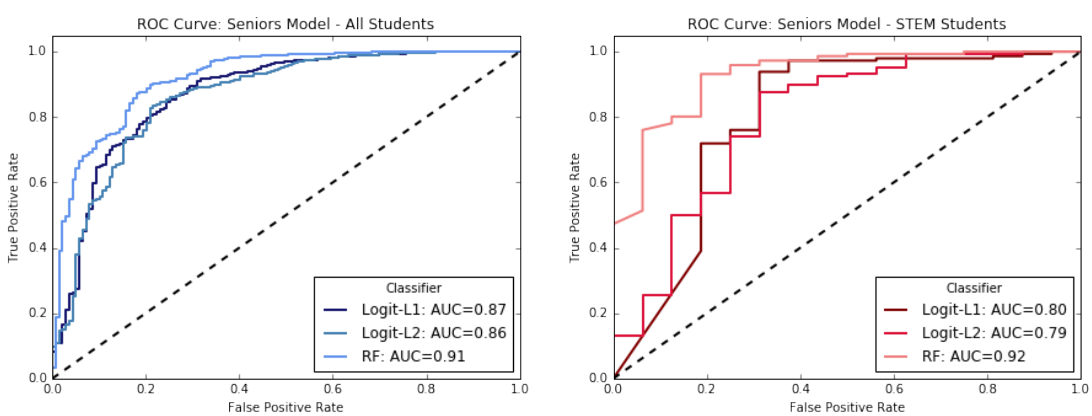

Figure 4.121: ROC Curve: Senior Model

Probability Calibration

Probability Calibration: Aggregate Model

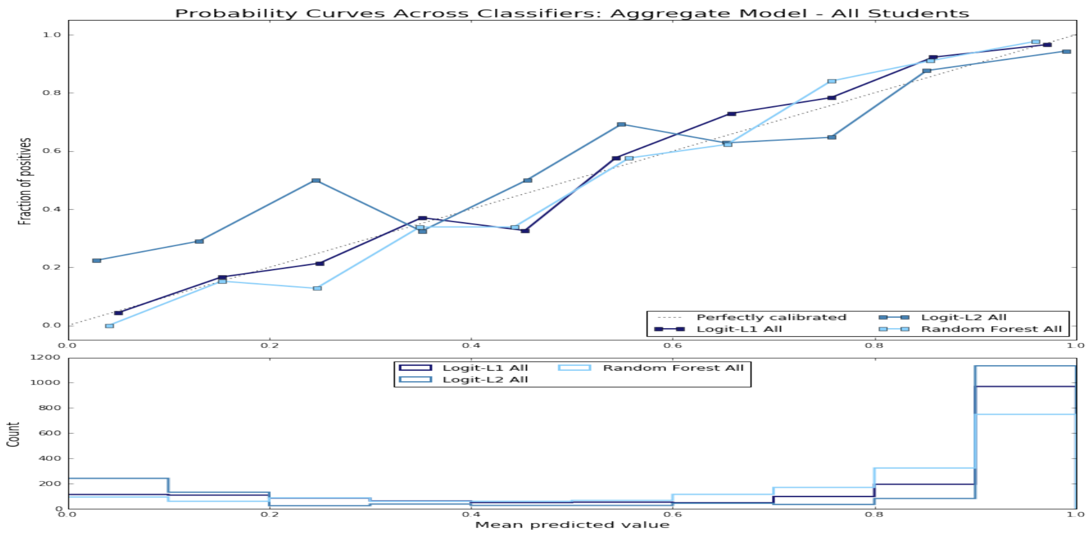

Figure 4.122: Probability Calibration: Aggregate Model, All Students

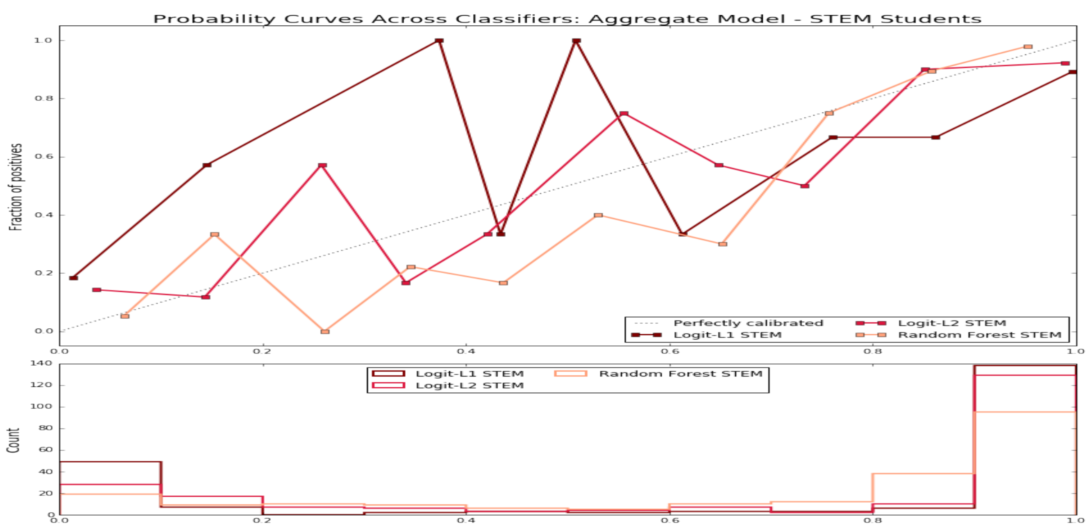

Figure 4.123: Probability Calibration: Aggregate Model, STEM Students

Probability Calibration: Freshman Model

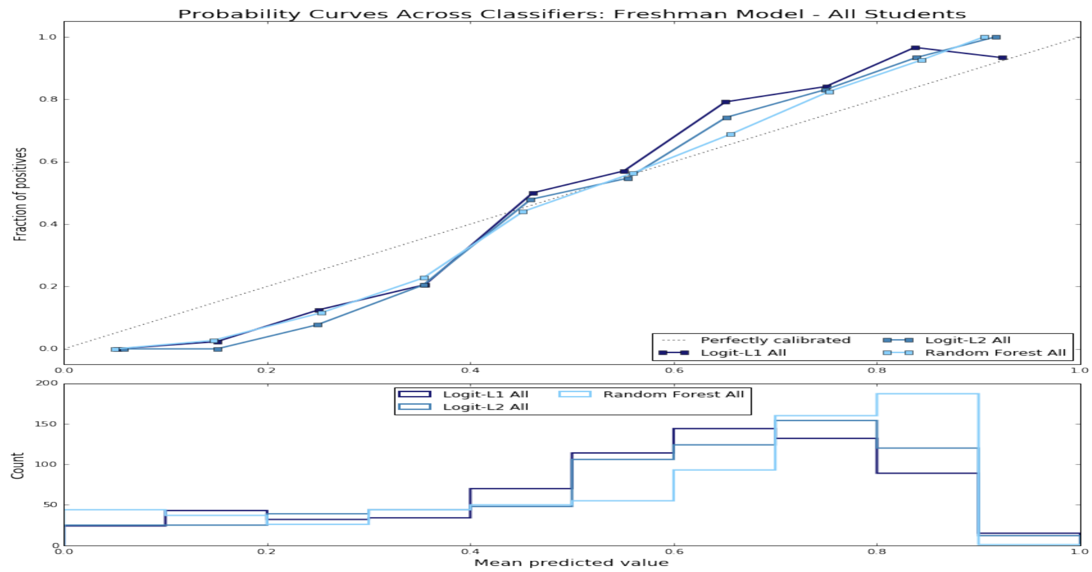

Figure 4.124: Probability Calibration: Freshman Model, All Students

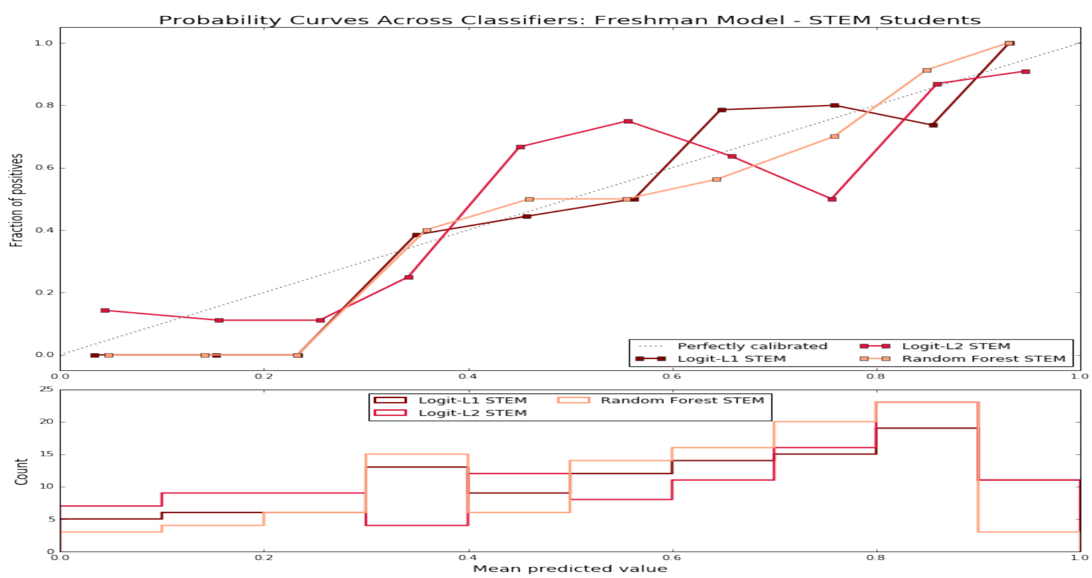

Figure 4.125: Probability Calibration: Freshman Model, STEM Students

Probability Calibration: Sophomores Model

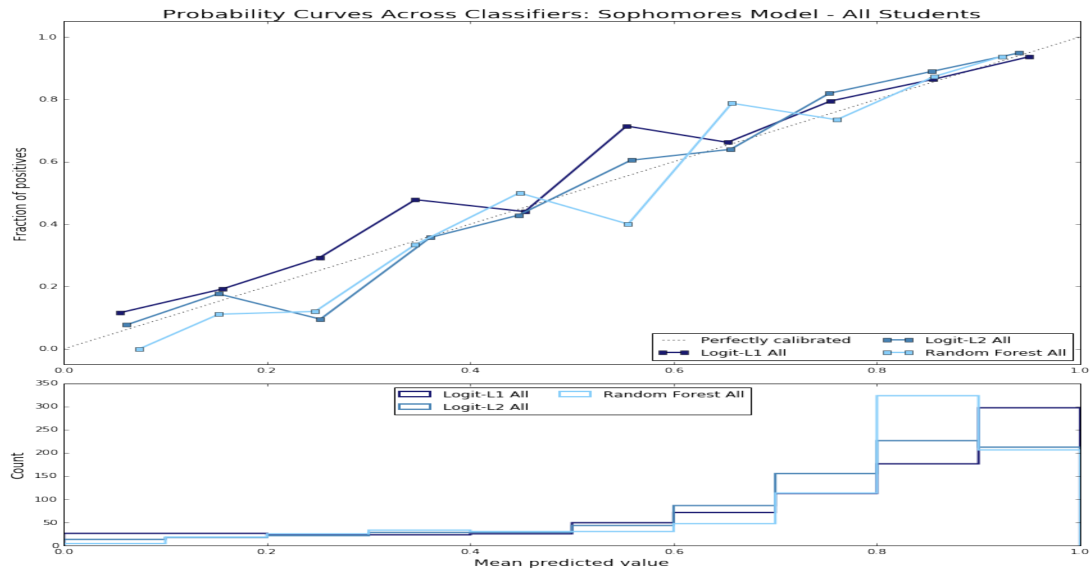

Figure 4.126: Probability Calibration: Sophomores Model, All Students

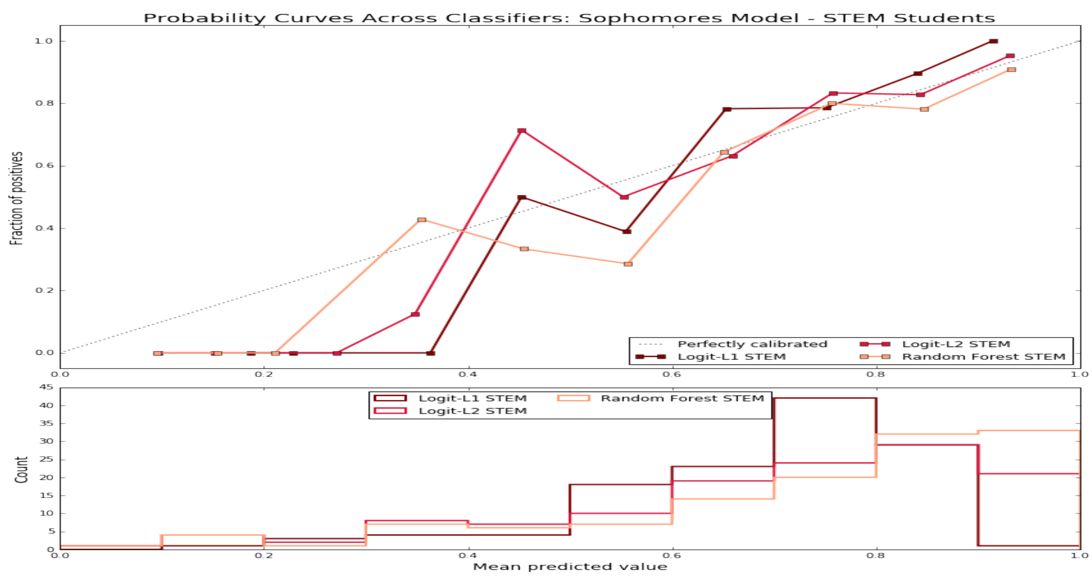

Figure 4.127: Probability Calibration: Sophomores Model, STEM Students

Probability Calibration: Juniors Model

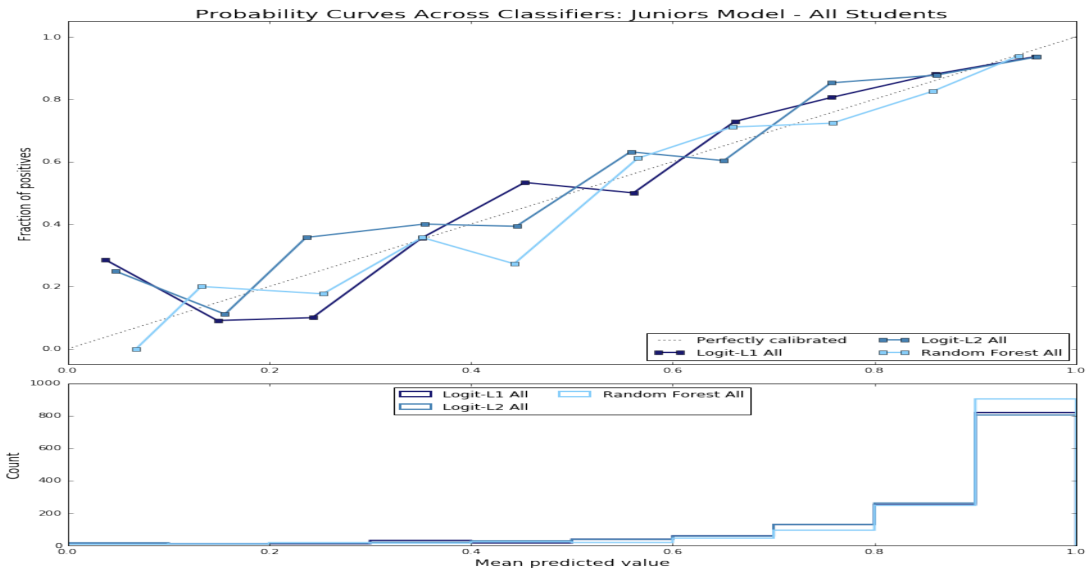

Figure 4.128: Probability Calibration: Juniors Model, All Students

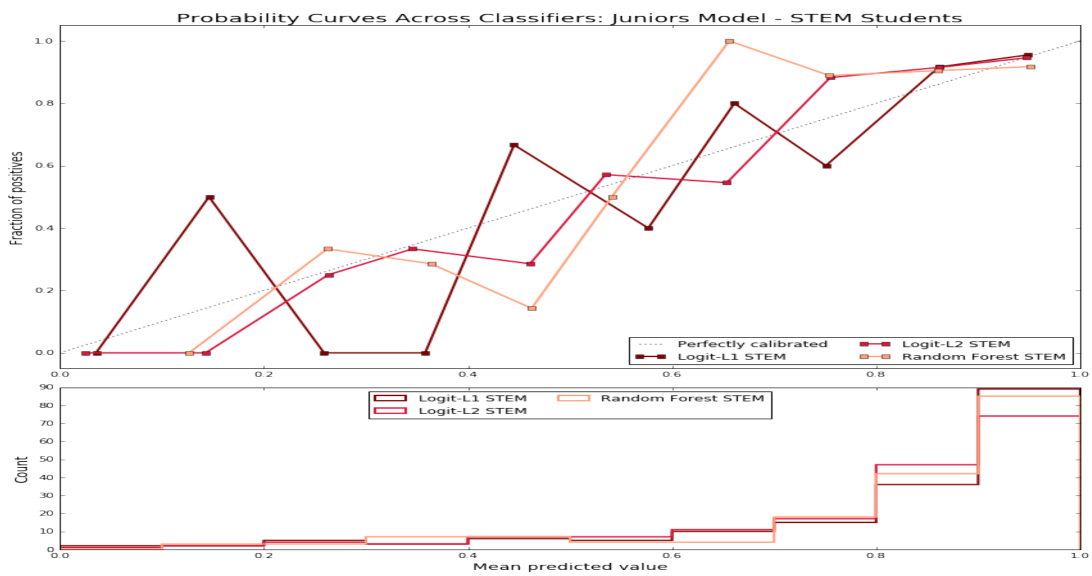

Figure 4.129: Probability Calibration: Juniors Model, STEM Students

Probability Calibration: Seniors Model

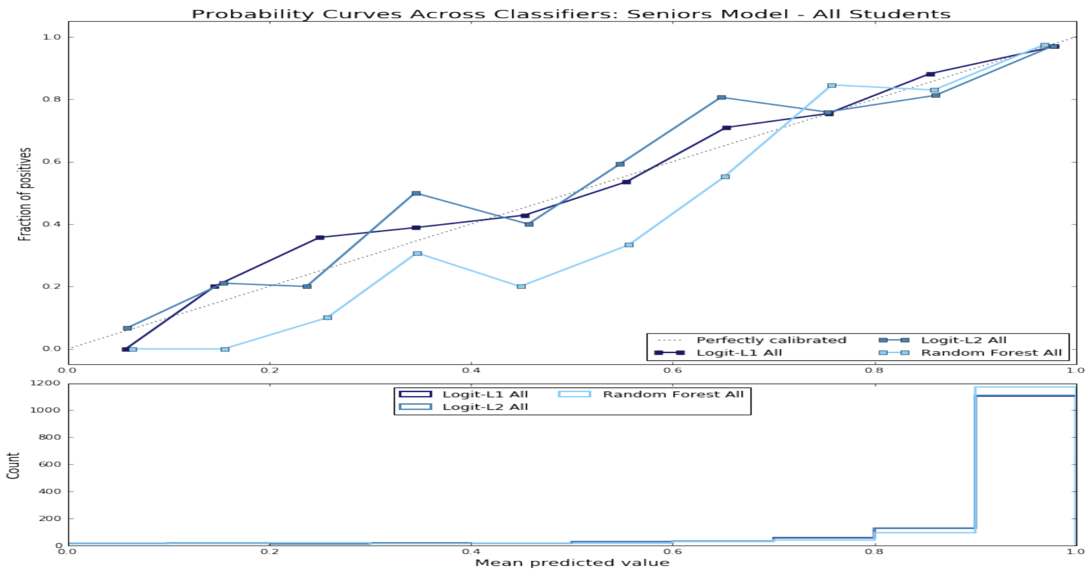

Figure 4.130: Probability Calibration: Seniors Model, All Students

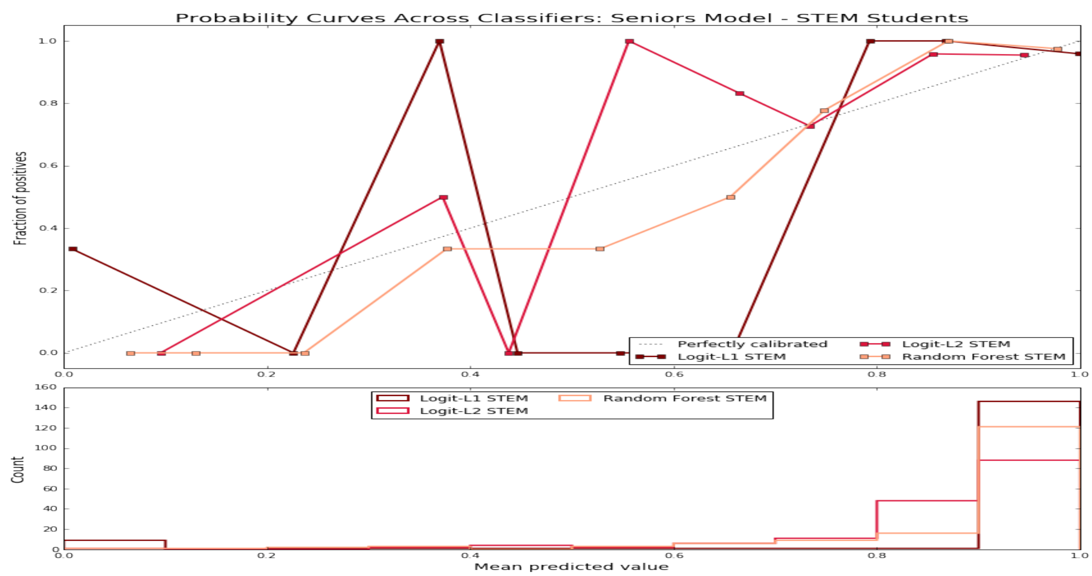

Figure 4.131: Probability Calibration: Seniors Model, STEM Students

4.2.5 Voting Classifier

Weights

A table displaying the optimal weights to be applied to each classifier is shown below, where w1 corresponds to logit-L1, w2 corresponds to logit-L2, w3 corresponds to the ranom forest, and mean is the mean accuracy score of the voting classifier after the given weights are applied.

Table 4.1: Voting Classifier: Optimal Weights and Mean Accuracy

		w1	w2	w3	mean	std
Model	**Students**					
Aggregate	**All Students**	2	1	2	0.892767	0.00158867
	STEM Stuents	1	3	1	0.881645	0.00900851
Freshman	**All Students**	2	3	1	0.736234	0.0471065
	STEM Stuents	2	3	1	0.736234	0.0471065
Sophomore	**All Students**	3	3	1	0.800263	0.0148444
	STEM Stuents	3	3	1	0.800263	0.0148444
Junior	**All Students**	2	1	3	0.881851	0.00806995
	STEM Stuents	2	1	3	0.881851	0.00806995
Senior	**All Students**	2	1	3	0.932503	0.0053264
	STEM Stuents	2	1	3	0.932503	0.0053264

4.3 Recommendation System

Sample output from a proposed recommendation system designed for administrators is shown below. A single non-STEM student and a single STEM student are chosen at random, and their probabilities of graduating are assessed by all applicable models. Tabular output shows the probability of graduating, and the associated alert level, generated by each applicable model; average probability and alert level for the table are displayed above the table itself.

Freshman: Non-STEM Student

```
recommend_intervention(studentFresh,cls='fresh')
```

```
('Average Probability: ', '0.319647749921')
('Average Alert Level: ', 'ORANGE')
('Actual Graduation Status: ', -1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.133445	RED
Freshman	**All Students**	0.50585	YELLOW

Freshman: STEM Student

```
recommend_intervention(stem_studentFresh,stem='stem',cls='fresh')
```

```
('Average Probability: ', '0.784511903694')
('Average Alert Level: ', 'GREEN')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.659436	YELLOW
	STEM Students	0.896311	GREEN
Freshman	**All Students**	0.757127	GREEN
	STEM Students	0.825173	GREEN

Table 4.2: Recommendation Sample Output: Freshman Non-STEM and STEM Students

Sophomores: Non-STEM Student

```
recommend_intervention(studentSoph,cls='soph')
```

```
('Average Probability: ', '0.51648546935')
('Average Alert Level: ', 'YELLOW')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.266045	ORANGE
Sophomore	**All Students**	0.766926	GREEN

Sophomores: STEM Student

```
recommend_intervention(stem_studentSoph,stem='stem',cls='soph')
```

```
('Average Probability: ', '0.463853690209')
('Average Alert Level: ', 'ORANGE')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.150294	RED
	STEM Students	0.0988062	RED
Sophomore	**All Students**	0.795438	GREEN
	STEM Students	0.810876	GREEN

Table 4.3: Recommendation Sample Output: Sophomore Non-STEM and STEM Students

Juniors: Non-STEM Student

```
recommend_intervention(studentJunior,cls='junior')
```

```
('Average Probability: ', '0.811725049138')
('Average Alert Level: ', 'GREEN')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.655773	YELLOW
Junior	**All Students**	0.967677	GREEN

Juniors: STEM Student

```
recommend_intervention(stem_studentJunior,stem='stem',cls='junior')
```

```
('Average Probability: ', '0.458657539435')
('Average Alert Level: ', 'ORANGE')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.345231	ORANGE
	STEM Students	0.209499	RED
Junior	**All Students**	0.51311	YELLOW
	STEM Students	0.76679	GREEN

Table 4.4: Recommendation Sample Output: Junior Non-STEM and STEM Students

Seniors: Non-STEM Student

```
recommend_intervention(studentSenior,cls='senior')
```

```
('Average Probability: ', '0.957341094062')
('Average Alert Level: ', 'GREEN')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.928574	GREEN
Senior	**All Students**	0.986108	GREEN

Seniors: STEM Student

```
recommend_intervention(stem_studentSenior,stem='stem',cls='senior')
```

```
('Average Probability: ', '0.7423869624')
('Average Alert Level: ', 'YELLOW')
('Actual Graduation Status: ', 1)
```

		Probability of Graduation	Alert Level
Model	**Students**		
Aggregate	**All Students**	0.393552	ORANGE
	STEM Students	0.726091	YELLOW
Senior	**All Students**	0.985784	GREEN
	STEM Students	0.864122	GREEN

Table 4.5: Recommendation Sample Output: Senior Non-STEM and STEM Students

4.3.1 Model Performance by Student Group

The tables below compare the accuracy of each model for different groups of students (i.e. the accuracy of the **aggregate model** for **freshman students** vs the accuracy of the **freshman model** for **freshman students**). Model performance varies greatly by student group, with specialized models designed for specific sub-populations almost always outperforming their more general counterparts. For example, for the sub-population of freshman students, the freshman models (both freshman-all and freshman-STEM) greatly outperform the aggregate models when predicting the probability of graduation. This indicates that, from a recommendation system design standpoint, specialized models are preferable to generalized models in terms of prediction accuracy.

Freshman

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.43759
Freshman	**All Students**	0.774749

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.454545
	STEM Students	0.545455
Freshman	**All Students**	0.690909
	STEM Students	0.681818

Table 4.6: Overall Model Accuracy: Freshman

Sophomores

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.483715
Sophomores	**All Students**	0.814234

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.48
	STEM Students	0.424
Sophomores	**All Students**	0.792
	STEM Students	0.784

Table 4.7: Overall Model Accuracy: Sophomores

Juniors

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.513435
Juniors	**All Students**	0.884532

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.427746
	STEM Students	0.410405
Juniors	**All Students**	0.890173
	STEM Students	0.872832

Table 4.8: Overall Model Accuracy: Juniors

Seniors

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.875615
Seniors	**All Students**	0.936051

		Accuracy
Model	**Students**	
Aggregate	**All Students**	0.820988
	STEM Students	0.864198
Seniors	**All Students**	0.932099
	STEM Students	0.925926

Table 4.9: Overall Model Accuracy: Seniors

Chapter 5

Conclusion

Individual classifier accuracy is enough to warrant continued study and eventual implementation of such systems at Rutgers University-Camden and, hopefully, the greater Rutgers community. Combining classifiers into an ensemble for prediction further improves both reliability and robustness by spreading the ”risk” associated with each classifier's unique vulnerabilities, and thus reducing the chance of incorrect classification. Class-specific ensembles perform particularly well, returning an accuracy of roughly 77 percent for freshmen, 81 percent for sophomores, 88 percent for juniors, and 93 percent for seniors. These values are well above random chance (50 percent) and present an opportunity to significantly improve student retention when combined with the experience and intuition of dedicated advisers.

5.1 Future Studies

5.1.1 Extending the Current Models

The current models, while performing exceptionally well, should be seen as more of a proof of concept than a rigorously tested system ready for implementation. Additional classifiers should be analyzed and, if performance warrants, added to the ensemble to further improve the reliability of predictions under stable environmental conditions and robustness of predictions to (inevitable) environmental fluctuations. Furthermore, a dynamic system that can automatically detect and respond to unusual behavior, whether in the inputs or the outputs, is preferable to a static system. For example the following two adaptations could improve the current system:

- a function to detect unusual input and temporarily exclude classifiers that are particularly vulnerable from participating in the ensemble

- a function to detect unusual prediction output from an individual classifier in the ensemble (assuming normal inputs), such as low accuracy or high numbers of false positives, and exclude that classifier from the communal decision making process.

Selection Metrics

Prediction accuracy, measured as the correct number of classifications, is not (and should not) be the only metric used during model calibration and eventual selection. There are many other selection metrics suitable to the application of this recommendation system that, in combination with prediction accuracy, can be used to fine-tune the selection of the "best" performing model/parameters. For example, precision, a measure of the frequency of false positives, is an excellent candidate to be added to the scoring process. In the context of predicting student graduation, a false positive represents a student who is predicted to graduate but, in reality, does not. In terms of the design objective, false positive are extremely "costly" as they represent the model's failure to detect and assist a student in need.

Recall, which is a measure of the frequency of false negatives, should also be considered as a scoring metric, but not necessarily with the same weight as precision. A false negative represents a student who is predicted not to graduate, but in reality, does graduate; the "cost" associated with a false negative relates to the unnecessary stress placed on a student due to misinformation. This is indeed a potential negative impact of incorrect classification, but in context of the design objective not nearly as detrimental as missing a struggling student completely.

It is recommended that the selection and weighting of scoring metrics be carefully considered by individuals with both computational and pedagogical experience, as certain interpretations and "cost" evaluations can quickly enter the realm of subjectivity.

Enhancing Data Collection

Collecting additional data on student behavior has the potential to further improve model performance by providing a more complete picture of a student's experience at the university. Some examples of useful student information include:

- historic and financial data: parent's education, average income, financial aid, tuition reimbursement, scholarship information, etc.
- tutoring center data: frequency of visits, length of stay, subject/class material covered, etc.
- clubs and extracurricular activities: clubs attended, frequency of attendance, club events, etc.
- fine grain course data: assignments given, assignments graded, ratio of number of grades to number of assignments, class attendance, class meeting time, etc.

5.2 Ethical Considerations

Recommendation systems must be assessed and categorized by the degree to which they influence human behavior and well-being; a system that recommends a song or movie is not nearly as influential in this regard as one that recommends direct action or intervention. In the context of this study, special attention must be paid to the proportions of false positives and false negatives among the incorrectly classified samples, as these situations could lead to action or inaction that has negative consequences for the students involved. As stated above, it is highly recommended that these systems be applied in combination with the experience and intuition of dedicated advisers to ensure that their potential benefit is realized without causing unnecessary or avoidable harm.

References

[1] National Science Foundation: Science and Engineering Indicators 2016, https://www.nsf.gov/statistics/2016/nsb20161/

[2] New York Times: Will You Graduate? Ask Big Data, https://www.nytimes.com/2017/02/02/education/edlife/will-you-graduate-ask-big-data.html

[3] New York Times: When a Few Bucks Can Get Students to the Finish Line, https://www.nytimes.com/2017/03/14/opinion/when-a-few-bucks-can-get-students-to-the-finish-line.html

[4] New York Times: What Can Stop Kids From Dropping Out, https://www.nytimes.com/2016/05/01/opinion/sunday/what-can-stop-kids-from-dropping-out.html

[5] New York Times: At College, A Guided Path on Which to Find Oneslef, https://www.nytimes.com/2017/03/28/opinion/at-college-a-guided-path-on-which-to-find-oneself.html

[6] Andrew Y. NG. *Feature Selection, L1 vs L2 Regularization, and Rotation Invariance* http://www.andrewng.org/portfolio/feature-selection-l1-vs-l2-regularization-and-rotational-invariance/