MACHINE LEARNING METHODS FOR GRADE PREDICTION

A PREPRINT

Charlie Hannum, Mark Leadingham II
Jerome Roehm, Dominick Sinopoli
Department of Mathematical Sciences
University of Delaware
channum@udel.edu mpltwo@udel.edu
jroehm@udel.edu domsinop@udel.edu

May 21, 2020

ABSTRACT

Students often have difficulty determining how they will ultimately perform in a course at the add/drop deadline. Instructors may benefit from predictive tools that can be used in the advising process. To assist in this process, we obtained the anonymized gradebook data for over 1,600 MATH221 students over eight previous semesters. We applied machine learning methods to predict the final grade of students using quiz and exam scores that occur before the change of status deadline. When it came to determining pass or fail at the add/drop deadline an altered gradebook scoring and a nearest neighbors adaptation performed just as well as support vector machines and decision trees. However, the nearest neighbors algorithm was determined to be the best predictive tool overall and was able to predict student performance to within half of a letter grade. These results are promising for providing instructors and students with a new predictive tool to aid in withdrawal advisement.

Keywords Machine Learning · Educational Data · Nearest Neighbors · Regression · Decision Trees

1 Introduction

MATH221 (Business Calculus) is a standardized course offered at the University of Delaware. Calculus is a required course for students in multiple majors and students typically enroll in this course during their first year to fulfill requirements for some academic programs. There are a varying number of lecture sections offered for these courses each fall and spring, but the usual amount is between 8-12 lecture sections to accommodate the number of students wishing to take the course (around 700-1100 per semester). This paper covers 1672 instances of anonymous scores from 8 semesters at the university. To alleviate the differences between instructors, a common curriculum was developed to standardize the course. In a given semester there are the same amount of quizzes, homeworks, and exam scores, with the exception of one semester which included additional "discussion quiz" scores. These are shorter quizzes administered during a Discussion Section, which is an additional meeting time with the Teaching Assistant of the course. The Discussion Section meets once a week for MATH221 to work on non-assessment questions to better the students' understanding of the course material. Additional details about the data are given in the next section.

The Drop/Fail/Withdrawal rates of a course are important details used by higher learning institutions for adjusting future course assessment. From an instructor perspective, it is also useful to accurately advise a student as to whether they can succeed in their course at a specified time within the semester, ideally before a withdraw deadline. Additionally, predictions can be used to help provide interventions and additional support to students who are on a trajectory to fail the course. This is a binary classification problem as to whether the student will pass or fail, which can also be refined as determining further subsets of letter grades accurately. Instructors and students alike would benefit from knowing the outcomes of "students like me", historical data that predicts the range of future (and most importantly final) scores a student is likely achieve given their current scores. Machine Learning covers a wide variety of methods useful for classification models. In this paper we cover three main goals for our algorithms as follows:

1. Achieve the highest possible accuracy for C- Pass/Fail classification.
2. Achieve the lowest possible MAE for predicting the students letter grade (grade points).
3. Identify methods that can be used as intuitive prediction tools for undergraduates to use.

2 Literature Review

Educational Data Mining (EDM) is a field with growing interest. The ability to project future performance based on a current state or a record of gradebook data is of great importance to higher education. A recent survey details the most relevant methods to the EDM field and the importance of stable algorithms [1]. We implemented many of the techniques mentioned in the article using the well-known Scikit-learn library [4] and several other Python packages [6]. Most of the literature deals with factors that are inherent to the individual student (gender, race, socioeconomic class, first-generation, etc) as well as prior performance in other courses [7]. Others focus on degree programs and the dropout rates associated with the students in differing subjects [5].

However, there is substantial interest in developing support for students that struggle in introductory level or further “gateway" courses. The University of Michigan has implemented several programs to study the effect of additional resources for students [2]. They detail prediction accuracy of within half of a letter grade by using previous course performance and prior grade point average of over 48,000 students. UM then uses this information to identify students that are may need additional resources early on in the course. A slightly different approach follows 700 students enrolled in a signal processing course at the University of California, Los Angeles. The authors study the optimal intervention time to within 4 weeks of the beginning of the course [3]. This paper gives the most support for our study, as it is unique in using only course data for prediction. We hope to apply similar ideology in our approach to predicting final grades and pass/fail rates using self-contained information.

3 Data Requirements

The data was obtained from instructors of the course after they anonymized the data in the interest of protecting student privacy. Therefore we have no knowledge of the semester nor year in which the student was enrolled in the course. However, we know the grading scheme is largely the same because the course is standardized. In fact, this will be verified with linear regression later on in this report. Below are the detailed descriptions of the features and target values of the dataset.

- `Semester` is an integer from 1-8 and denotes an anonymous semester in which the grades occurred.
- `Identifier` denotes a student. i.e., Student 1, Student 2, etc. These are randomized before importing.
- `Attendance and Participation Score` is the grade received for attending lecture and discussion throughout the course. This is usually curved upwards to a max of 100.
- `Assignments Score` is the averaged grade amongst all homework assignments, with the lowest score dropped. Information about individual assignments was not obtained.
- `Quizzes Score` is the averaged grade amongst all five quizzes, with the lowest of the scores dropped.
- `Mid-Term Exams Average` is the average between `Exam 1` and `Exam 2`.
- `Final Exam Score` is the grade received on the final exam.
- `Final Score` is the grade received for the course overall.
- `Final Grade` is the letter grade received for the course overall.
- `Quiz 1`, `Quiz 2`, `Quiz 3`, `Quiz 4`, `Quiz 5` are the grades received for the 5 "Lecture Quizzes" (quizzes taken during regular lectures).
- `Exam 1` and `Exam 2` are the grades received on mid-term exams.
- `DQ_Y_N` is a Yes/No binary that states whether or not a semester incorporated "Discussion Quizzes" (quizzes taken during regular discussion section meetings).
- `DQ#` or `Discussion Quiz Score` is the grade received for the 5 individual “Discussion Quizzes" (if a semester includes them, blank otherwise).
- `GPA` indicates the “Grade Points" that a student receives for the course, according to the grading policies of the University of Delaware.
- `C Pass` is a binary value denoting whether a final score is passing (with an A, B, or C) or failing (D and F), i.e., a score is greater than or equal to 67%, the cutoff grade to receive a letter grade of C-. In this report, we will focus on this measure of passing and failing.
- `PF` is a binary value denoting whether a final score is passing (A, B, C, D) or failing (F).
- `Deadline Grade` is a percentage grade at the change of status deadline as calculated by `Quiz 1`, `Quiz 2`, `Exam 1`, `Quiz 3`, and `Attendance and Participation Score`, using the weights in the gradebook. This score is very close to the score that students would see when they checked the online gradebook. It is not identical as the `Assignments Score` is not included. More details to follow.

Additional decisions by the instructors vary. Some semesters include “Discussion Quizzes" while other do not. These scores are typically negligible in terms of grade determination, and largely align with the attendance and participation scores. Another more recent policy allows either the `Exam 1` or `Exam 2` score to be replaced by the `Final Exam Score` provided the latter is greater than 70% and greater than one of the exam scores. However, only the final values entered into the gradebook are accessible, so we cannot determine whether or not a student took advantage of the above policy. However, after a preliminary search, only a handful of student could have taken advantage of this policy. These considerations may provide insight into anomalies in the classification of grades (detailed in the Results Section). The assessments in the MATH221 course follow the same schedule each semester:

`Quiz 1, Quiz 2, Exam 1, Quiz 3, Quiz 4, Exam 2, Quiz 5, Final Exam`

with the withdrawal deadline falling between third and fourth Quiz, making the ideal subset of features:

`Quiz 1, Quiz 2, Exam 1, Quiz 3`

in addition to the `Assignments Score` and `Attendance and Participation Score`, which are are determined throughout the entire semester. For classification purposes we will focus on the latter subset. We justify including `Attendance and Participation Score` as this score is almost entirely in the students control. We also assume that it remains consistent throughout the semester. Additionally, including this data allows us to use our conclusions as motivation for the students to attend their lectures. This application is seen specifically in the Nearest Neighbors section.

3.1 Grading Scheme

The grading scheme for the MATH221 course remains mostly consistent between semesters. The two possible scenarios we encounter are laid out in the following tables.

Assignment Name	Percentage (%)
Exam 1	20
Exam 2	20
Final Exam	25
Best 4 of 5 Quizzes	20
Homework	10
Attendance	5

Table 1: Grading Scheme 1

Assignment Name	Percentage (%)
Exam 1	20
Exam 2	20
Final Exam	25
Best 4 of 5 Quizzes	20
Homework	10
Section Grade*	5

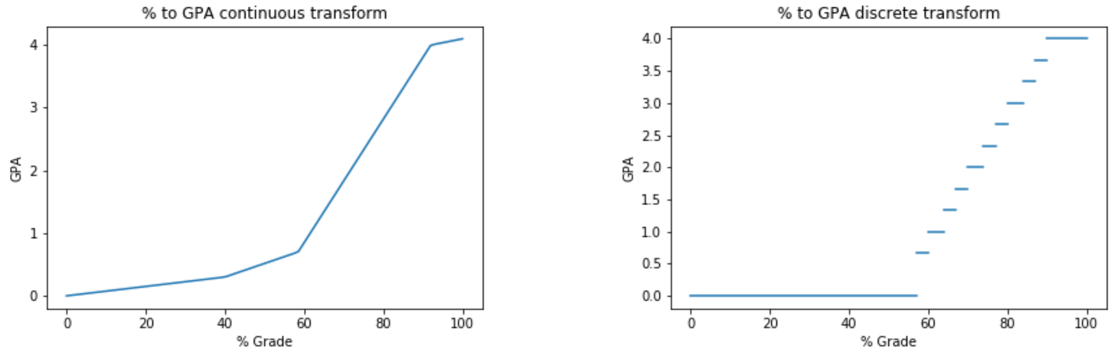
Table 2: Grading Scheme 2

Grade	**Percentage Range**	**GP**
A	90 - 100.0	4.000
A-	87 - 89.99	3.667
B+	84 - 86.99	3.333
B	80 - 83.99	3.000
B-	77 - 79.99	2.667
C+	74 - 76.99	2.333
C	70 - 73.99	2.000
C-	67 - 69.99	1.667
D+	64 - 66.99	1.333
D	60 - 63.99	1.000
D-	57 - 59.99	0.667
F	< 57	0.000

Table 3: Letter Grade to GPA Conversion

Table 2 incorporates `Discussion Quiz Score` into the `Attendance and Participation Score` to form the Section Grade, where the weekly attendance over 15 weeks and the sum of the 5 discussion quizzes at 3 points each comprises 30 total points, or 5% of the total points possible in the course. There is a high correlation between `Discussion Quiz Score` and `Attendance and Participation Score` in the single semester they occur, so we focus on attendance. Moving forward we assume there is no significant difference between these two grading schemes and we drop the `DQ` columns for the entirety of our analysis. In order to discretize this classification problem we can transform our data according to the `GP` scale detailed in Table 3. We will also be using the continuous transform to scale the percentage data in order to accentuate the differences in grade points. As detailed in Section 4, scaling the percentage data improves the performance of various machine learning algorithms. The equation of the continuous function is shown below, and the discrete case from Table 3 is visualized in Figure 1b. This formula was motivated by the desire to accentuate the differences between the grade points, under the constraints of injectivity and continuity. The function was obtained by a simple rough fitting of a piecewise linear model to the discrete model, subject to the aforementioned constraints.

$$g(p) = \begin{cases} \frac{3}{400}p & p < 40 \\ \frac{4}{185}p - \frac{209}{370} & 40 \leq p < 58.5 \\ \frac{33}{335}p - \frac{1696}{335} & 58.5 \leq p < 92 \\ \frac{1}{80}p + \frac{57}{20} & 92 \leq p \end{cases} \tag{1}$$



(a) Piecewise Linear transform of grade percentage to grade points. (b) Discrete transform of grade percentage to grade points.

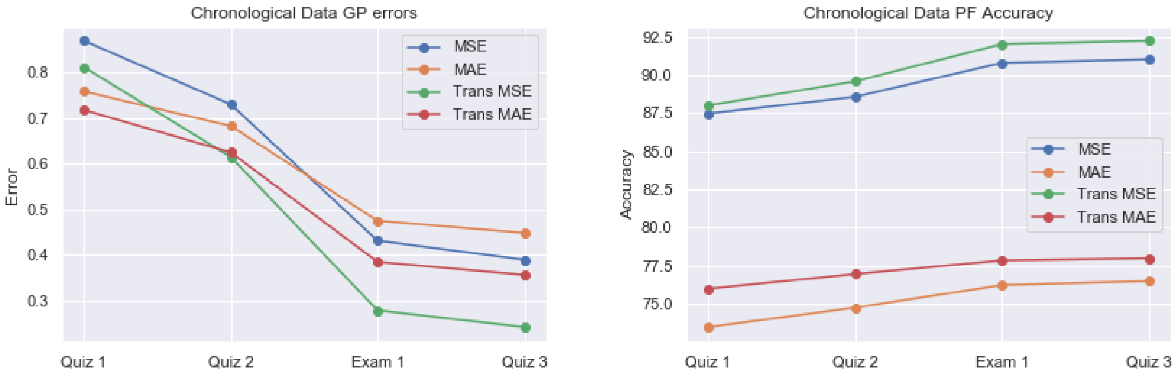
Figure 1: Two scaling methodologies for prediction in continuous or discrete setting.

4 Various Approaches

Below we detail the algorithms applied to our classification problem and the necessary assumptions for each method. The subsections provide motivation for the chosen method, appropriate figures, and a brief summary of their results. Some methods performed well under the default parameters, whereas others were optimized to improve performance. These optimal parameter sets are detailed below. A more detailed summary is given in the Results Section.

4.1 Linear Regression

Given our data, $X \in \mathbb{R}^{n \times p}$, and the target column `Final Score`, $y \in \mathbb{R}^{n \times 1}$, linear regression can be used to approximate the grading scheme of a course, even if this information is not known. For $p = 6, n = 1672$ we fit the data to the features provided in Table 1 and minimize the residual sum of squares. As a verification, the weights $\beta_1, \dots, \beta_p$ determined with `scikitlearn`'s `LinearRegression` [4] are $[0.1998, 0.2077, 0.2459, 0.1979, 0.0988, 0.0532]$, ordered according to the features in Table 1. Rewritten as percentages these values approximate the correct grading scheme, and this method holds true for each semester considered.
Now consider the chronological data before the Withdrawal Deadline, denoted $X_c \in \mathbb{R}^{n \times r}$, with target columns $y_{PF} \in \mathbb{R}^{n \times 1}$ and $y_{GP} \in \mathbb{R}^{n \times 1}$ where $r = 5, n = 1672$. The `Attendance and Participation Score` can be reasonably estimated at any point in time, so we include this feature as detailed at the end of the Data Requirements Section. For each feature the mean squared error (MSE) and mean absolute error (MAE) can be computed for both the `GP` and `PF` predictions. The numerical values demonstrate the benefit of knowing more features as the semester progresses in time. This is illustrated in Figure 2. We can see that basic linear regression on the full subset will result in an accurate prediction of y_{GP} and y_{PF}. Additionally, the transformed data is now on the same scale as the `GP` column, so it is expected the prediction accuracy will increase.



(a) MSE and MAE predictions of GP using Linear Regression. (b) MSE and MAE accuracy score of PF using Linear Regression.

Figure 2: Linear Regression predictions of `GP`/`PF` and the effect of transformation on the chronological data. The features are each combined with the `Attendance and Participation Score`.

4.2 Nearest Neighbors

When discussing K-Nearest Neighbors, an important consideration is the metric involved to determine the relative “closeness" of data points. With our data, we intuitively suspect that some of the dimensions, namely the exams, are more important and more predictive of the outcome than others. Basic linear regression confirms this. The following figure shows the results of performing linear regression with all data in raw percentages. This verifies our intuition that some dimensions (assignments) are more predictive than others. We choose to use a variant of the usual Euclidean metric in $\mathbb{R}^p$ by stretching the axis of the more predictive dimensions. Conceptually, when comparing students to find their closest neighbors, we value similarity on the exams more highly than similarity on the other assignments. The following figures depict the effect of stretching an axis. Suppose *Student A* scores 95% on `Quiz 2`, and 60% on `Exam 1`. In Figure 4, the Euclidean metric is altered so that the `Exam 1` and `Quiz 2` axes are stretched in proportion to their weights in the gradebook. This means that traveling in the `Exam 1` direction one unit is 4 times as far as traveling one unit in the `Quiz 2` direction. With no stretch, the nearest neighbor method would base the prediction of *Student A* on the final grades of the students in red. With stretched axes, the method would base the prediction on the final grades of the students in yellow. Observe that stretching the axes deforms a circle (or p-sphere in higher dimensions) into an ellipse (or ellipsoid in higher dimensions). Now this stretching can affect the predictions that the model makes, regardless of whether the model predicts by the median or mean of the neighbors. Figure 5 displays the grade distribution of the neighbors with the unstretched and stretched metric. Recalling that *Student A* scored 95% (A) on `Quiz 2`, but a 60% (D) on `Exam 1`, in determining the neighbors to make the prediction, it is intuitive that when we increase the importance of `Exam 1` the prediction decreases. With stretching, the median of the neighbors decreases from 2.0 to 1.7, and the mean decreasing from 2.1 to 1.7. By stretching the predicted grade has changed from a C to a C-. Note also that with stretching, *Student A* has none of the 125 neighbors earning an A of A- in the course.

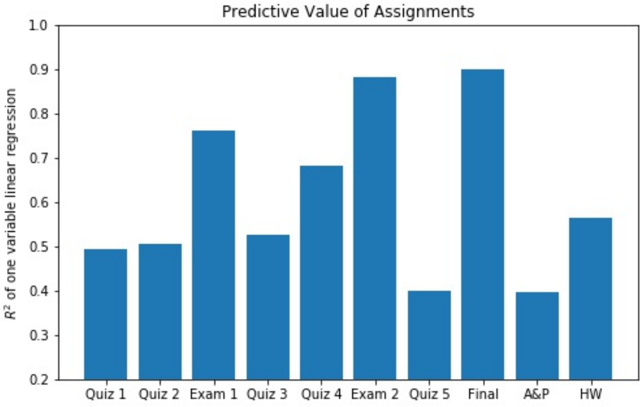


Figure 3: R^2 value for Linear Regression using each feature individually.

The question then becomes, what is the optimal stretch for each of the assignments, how many neighbors to select, and whether to assign the prediction by the median or mean? We choose mean absolute error (MAE) as our loss function as it is more intuitive than mean squared error. For example, if a method achieves a 0.4 MAE, we can say clearly that our prediction is off by less than half a letter grade, on average. Now to optimize the parameters we perform a grid search using N-fold cross validation. That is, for each student we make a prediction based on all the other students in the dataset, then measure the MAE on our predictions for each student. We focus here on making a prediction at the deadline, using the subset of pre-deadline assignments, including the `Attendance and Participation Score`. After running code overnight, we find the approximate optimum parameters to minimize MAE as:

- 25 neighbors
- Assign prediction by *median* of the neighbors
- Stretch assignments as follows:
  1. `Quiz 1`: $\times 1.1$
  2. `Quiz 2`: $\times 1.0$
  3. `Exam 1`: $\times 2.0$
  4. `Attendance and Participation Score`: $\times 1.0$

Note that the theory (as discussed in MATH637) supports assigning the prediction by the median in order to minimize MAE. If we were to optimize in order to minimize mean square error, MSE, the model would assign the prediction by

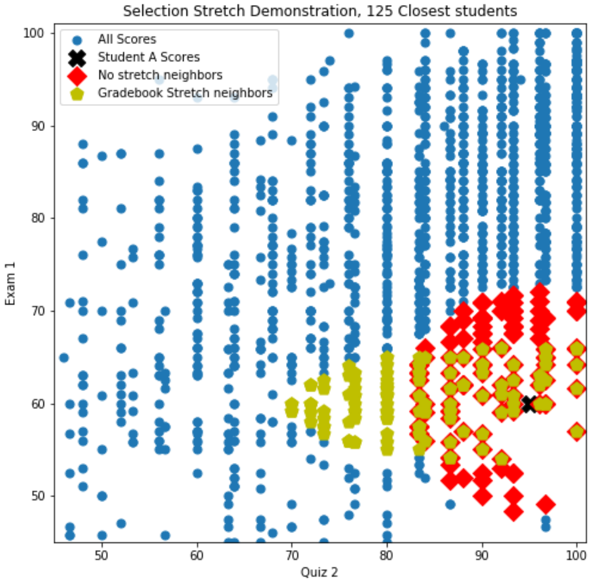
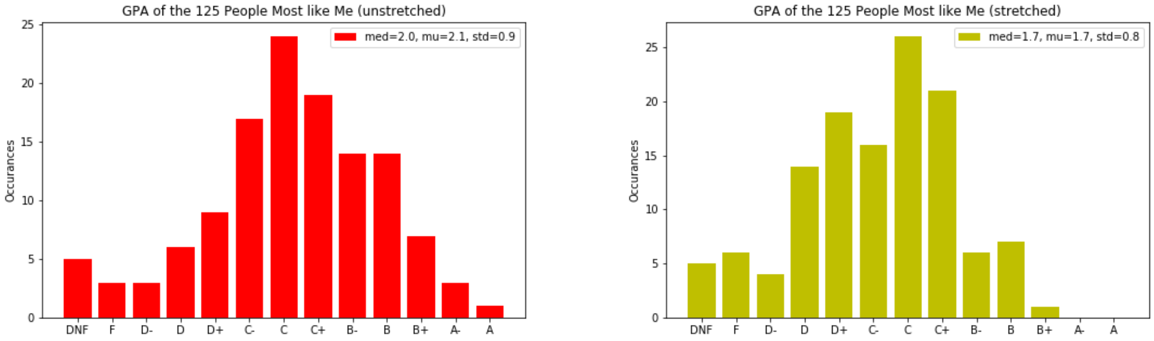


Figure 4: Gradebook Stretch Demonstration of 125-NN for the `Quiz 2`, `Exam 1` projection.



(a) Barchart of 125 similar students without stretch.

(b) Barchart of 125 similar students with stretch.

Figure 5: Demonstration of the effect on the grade distribution of similar students when stretching the 125-NN.

the *mean* of the neighbors. Interestingly, the stretching factors are not equivalent to the gradebook ratios. For example, `Exam 1` is worth 4 times as much in the gradebook as `Attendance and Participation`, but is only stretched by a factor of two.

Using these parameters we obtain the following histogram of errors for the nearest neighbor method. Note that this data is obtained through the one vs. rest procedure described as N-fold cross validation. Observe the approximately normal distribution of the errors. This method achieves a test MAE of 0.314. Additionally, 68% of predictions are within a single classification unit (ex. C+ vs. B-) and 96% of predictions are within a single letter grade. Finally, when it comes to the key tipping point, this method predicts whether a student will fall on the right side of a C- with just over 92% accuracy.

We now present a few interesting case studies of students, using variants of our nearest neighbor method. First consider Student 1642. This student's grades are shown in the table below. This student had a score that would earn a C- at the change of status deadline. However, our nearest neighbor algorithm notices the downward trend in Table and predicts the student will earn below a C-. Our algorithm sees this because there are many other students who have exhibited this downhill slide. The neighbors of Student 1642 are shown to the right. Note that the median is 1.3, so the algorithm predicts a D+.

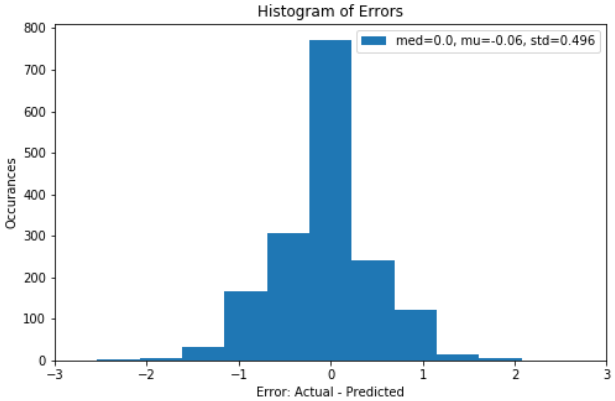
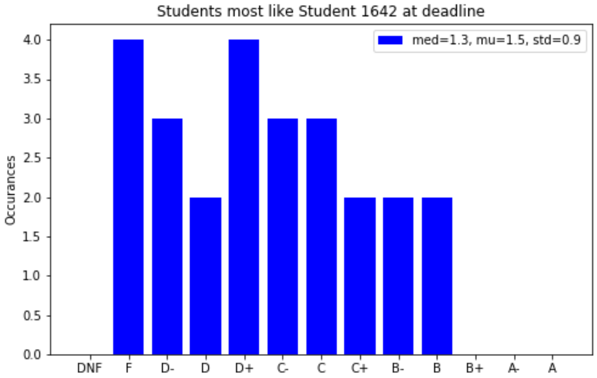


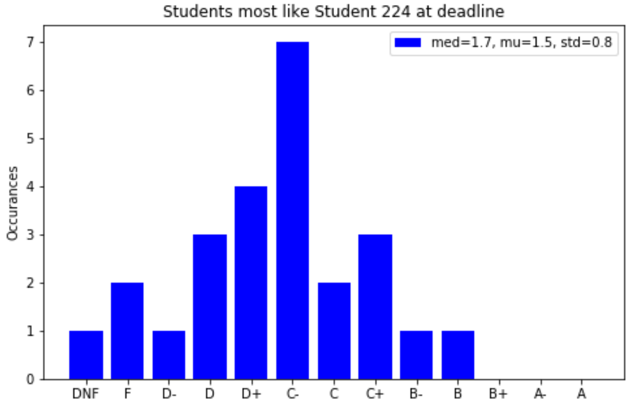
Figure 6: GP Prediction Error Distribution associated with the stretch parameters above.

Assignment Name	Percentage (%)
Quiz 1	100
Quiz 2	73
Exam 1	65
Quiz 3	43
Attendance	62
Deadline Grade	*67*
Quiz 4	87
Exam 2	48
Quiz 5	80
Final Exam	53
Homework	64
Final Grade	*62* (D)

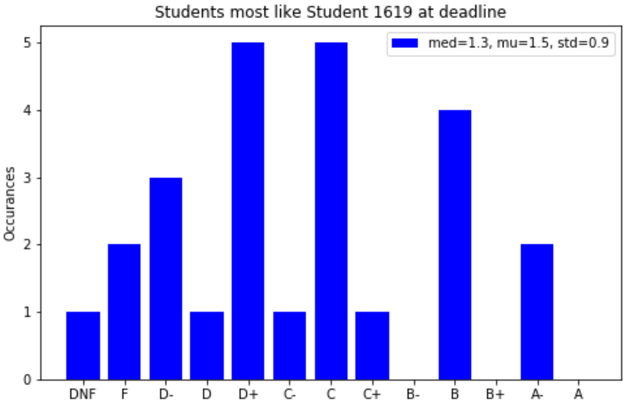


Our NN algorithm can also identify the opposite trend, as in the case of Student 224. This student is on an upward trend, with a relatively strong performance on `Exam 1`. This student takes a 0 on `Quiz 3`, likely because the student did not attend class. This causes the student's deadline grade to be considered failing. However, our algorithm correctly predicts that Student 224 will pass. The positive prediction also reveals another interesting characteristic of the NN method. Many students obtain a 0% on a Quiz Score, for one reason or another. In the final grade, the best 4 of 5 quizzes are used, so a student can recover from a 0%, particularly if it is only one quiz. With this understanding of the historical data, the NN method is able to correctly predict the outcome of Student 224, based on the upward trajectory and the quiz forgiveness policy. The neighbors are shown to the right. Note that the median is 1.7, so the algorithm predicts a C-, which is the student's actual grade.

Assignment Name	Percentage (%)
Quiz 1	50
Quiz 2	68
Exam 1	78
Quiz 3	0
Attendance	85
Deadline Grade	*64*
Quiz 4	64
Exam 2	74
Quiz 5	8
Final Exam	60
Homework	80
Final Grade	*67* (C-)



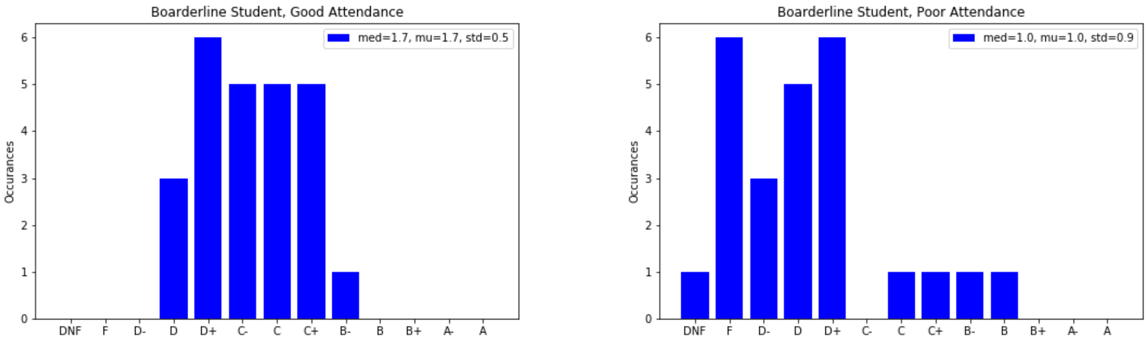
Assignment Name	Percentage (%)
Quiz 1	97
Quiz 2	77
Exam 1	59
Quiz 3	77
Attendance	65
Deadline Grade	*69*
Quiz 4	93
Exam 2	68
Quiz 5	63
Final Exam	81
Homework	100
Final Grade	*76* (C+)



This is not to say, of course, that the algorithm is perfect. It is still makes incorrect predictions on pass/fail about 8% of the time. This is only slightly better than if one just predicts the final grade by the deadline grade alone. This prediction task is a difficult one. Consider student 1619. This student is passing at the deadline, but appears to be on a downward trajectory, similar to student 1642. However, student 1619 turns this trend around to earn a C+. Our NN algorithm predicts this student will earn a D+.

Lastly, we elaborate on the strength of our Nearest Neighbors prediction algorithm as a student (and instructor) facing tool. It would be quite easy to allow students and instructors access to a simple version of this tool. Students could enter their current scores and then the algorithm would produce a bar chart, similar to those above, detailing how the 25 students most like them have performed in the past. This could aid students and advisors in the decision of whether or not to change status to listening in the course. It could also help instructors provide interventions and extra support to students who are on track to fail the course.

Using the Nearest Neighbors algorithm as a student facing tool also allows the algorithm to act as motivation for attendance and participation. Although `Attendance and Participation Score` is only 5% of the overall score in the gradebook, as intuition suggests, it has a much larger impact on the success of the students. For example, consider a scenario of a student who is on the boarder at the change of status deadline. For example, let this student's scores be as follows: `Quiz 1`: 70%, `Quiz 2`: 60%, `Exam 1`: 65%, `Quiz 3`: 70%. This amounts to a deadline grade of 66%. Suppose this student has attended most, but not all classes. Using the nearest neighbors prediction algorithm, the student could manipulate their `Attendance and Participation Score` and watch the results change. Consider the two bar graphs in Figure 7. If the student attends the remainder of the classes and earns a 90% `Attendance and Participation Score`, the algorithm predicts the student will pass with a C-. However, if the student skips some, but not all of the upcoming classes and earns a 60% `Attendance and Participation Score`, the algorithm predicts, that the student will fail with a D grade. Note that all but 4 of the 25 neighbors failed the course, indicating the the model is in some sense confident about the prediction of failure for this boarderline student with poor attendance.



(a) Barchart of similar students with 90% Attendance

(b) Barchart of similar students with 60% Attendance

Figure 7: Demonstration of the effect on the grade distribution of similar students when attendance is manipulated.

Even though `Attendance and Participation Score` is a very small portion of the grade, students who attend class fair much better than those that don't, as evidenced above. We believe that students who see that people like them who attend class perform much better than people like them who don't attend class is a powerful motivational tool. Overall, in the task of predicting the final letter grade of the students we believe that our nearest neighbor method is the strongest for the following reasons:

- From a student perspective, seeing "People Like Me" is a powerful comparison, which is intuitive to students.
- This method provides a distribution, as well as a single prediction.
- It has a motivating effect on students concerning A&P. Students can alter the input and watch the predicted outcome shift.
- This method is roughly as accurate as all other methods we have found. See the Results section for a full summary. This method has the added bonus of transparency and intuitiveness that other methods, such as SVM and Neural Networks, lack.

4.3 Support Vector Machines

Support Vector Machines (SVM) are one of the most useful machine learning techniques for binary classification by maximizing the margin between two separate classes with a separating hyperplane of lesser dimension than the data. The binary classification for this data set was the pass or fail case. Using the grade of a C- for a pass, a gradebook score of 67% was determined to be the cut off. The binary classifier 0 or 1 was used to classify the data. A 0 represented a failing grading less than 67% while 1 represented a passing grade greater than 67%. An in-depth analysis was performed on the normal data set using `scikitlearn`'s `SVC` package.[4]

Kernel Name	R^2
Rbf	91.897
Linear	91.689
Poly	91.545
Sigmoid	72.433

(a) Kernel analysis using gamma = 'scale'

Kernel Name	R^2
Rbf	71.291
Linear	90.431
Poly	91.866
Sigmoid	72.488

(b) Kernel analysis using gamma = 'auto'

Figure 8: Examining the basic SVM models while varying the gamma parameter

For each of the various kernels, `RBF`, `Linear`, `Poly`, and `Sigmoid`, a test-train-split approach was taken to generate R^2 values. Selecting a `RBF` kernel will result in a nonlinear boundary, `Linear` will result in a linear boundary, a `Poly` kernel will generate a boundary of a specified degree (default=3), and a `Sigmoid` kernel will generate a hyperbolic tangent boundary. A test-train-split approach was taken to train the model with each of the varying kernels and generate an R^2 value. This was repeated over 100 trials and the average R^2 from the 100 trials are shown in Figure 8. The kernel review was done to help figure out which kernel best separated the data set. From this evaluation the chosen model to further improve upon was using a linear kernel and gamma = 'scale'. The replicability of the SVM model and ease of interpreting the coefficients made it an excellent choice for analysis.

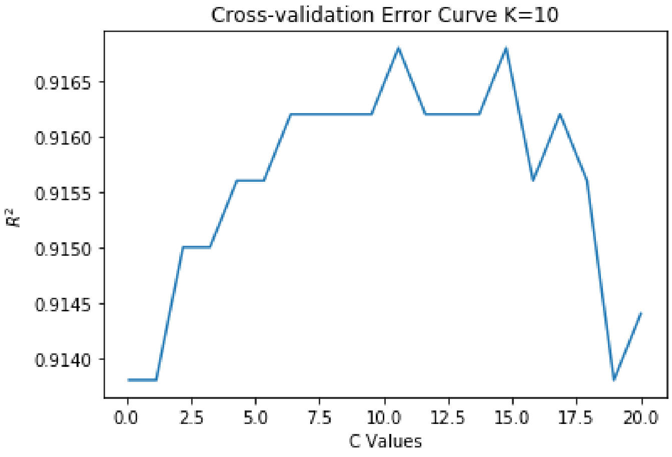


Figure 9: SVM regularization parameter using K=10 fold cross validation

The parameter C can be used to adjust the margin of the hyperplane and is used as a regularization parameter for the SVM model. For smaller values of C a larger margin is used for the separating hyperplane while conversely larger values of C will result in a smaller margin. The default value for C in the SVM model is 1. Using $K = 10$ fold cross validation and values of C ranging between 0 and 20, it was determined the value of C had very little effect on the model. In Figure 9 above, the error only ranged from 91.40% to 91.65% for a difference of .25%.

Next we apply a bootstrapping approach to try and further improve the model. By using a `Linear` kernel we can extract the intercept and coefficients from the model which are the weights assigned to each parameter. 100 models were constructed using the default value for C and the coefficients and intercepts from each were averaged together to create one model. This boosted model was created to help mitigate the variance between each model and thus produce a stronger model with better predictive power. Using the variables `Quiz 1`, `Quiz 2`, `Quiz 3`, `Exam 1`, and `Attendance and Participation` the weights $\beta_1, \ldots, \beta_5$ from the bootstrapped model are $[0.0244, 0.0165, 0.0218, 0.1163, 0.0323]$ with an intercept average of -14.59. The averaged coefficients weights being positive and the intercept being negative aligns with the data. From this it is observed that the variable `Attendance and Participation` had greater weight assigned to it that any of the quizzes and the greatest weight was assigned to `Exam 1`. From Figure 10 the boosted linear model on average (using 100 trials) and a test-train-split approach had a R^2 value of 92.035% which is roughly .5% better than the default methods.

Method	R^2
Boosted Linear kernel	92.035
Default Linear kernel	91.607
Default SVM (RBF kernel)	91.696
Linear Regression	54.959

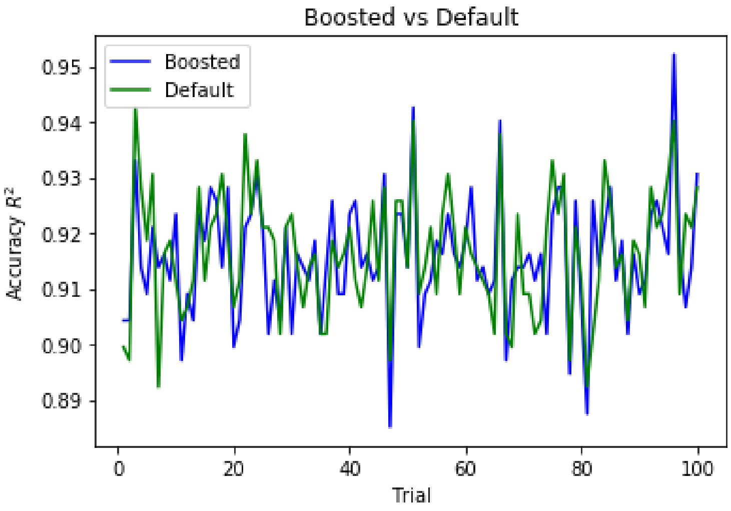
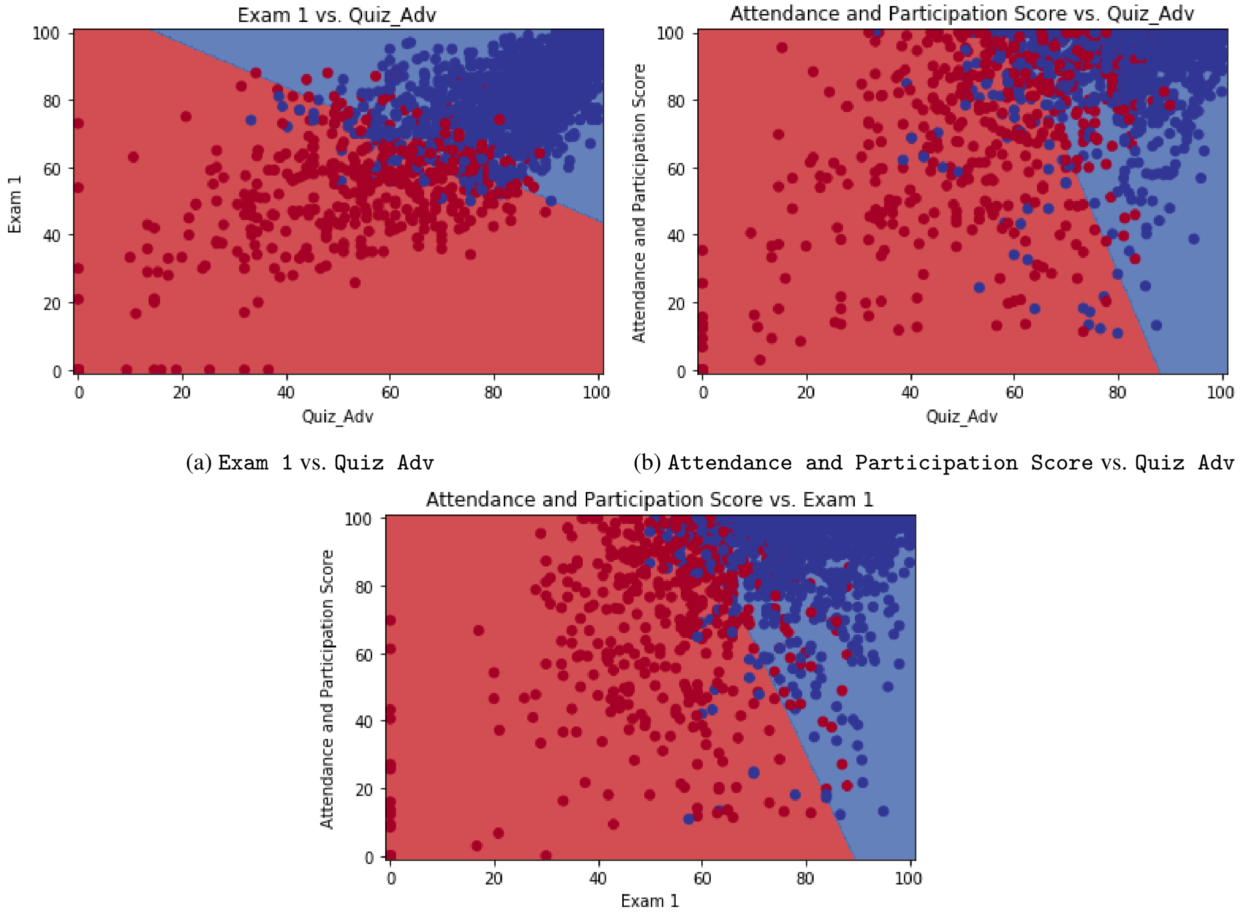


Figure 10: Varying SVM models

In order to better visualize the SVM classifiers the dimensions of the data set had to be reduced from $X \in \mathbb{R}^{n\times 5}$ to $X \in \mathbb{R}^{n\times 2}$ where n is the number of observations in this case $n = 1672$. The variables were reduced by averaging all three of the quiz scores into one variable called `Quiz Adv`. Each combination of remaining variables was observed in 2D, `Exam 1` vs. `Quiz Adv` (a), `Attendance and Participation` vs. `Quiz Adv` (b), and `Attendance and Participation` vs. `Exam 1` (c). For each of the plots below passing is represented by the blue region and failing is represented by the red region. The SVM classifier can be observed as the line separating the two regions. In plot (a), there is a strong positive correlation between `Exam 1` and `Quiz Adv`. While in plot (b) there appears to be no correlation between `Attendance and Participation` and `Quiz Adv`. In plots (a) and (c) the passing and failure allocating becomes very muddy around the SVM classifier as the model has a hard time separating out these borderline cases.



(a) `Exam 1` vs. `Quiz Adv`

(b) `Attendance and Participation Score` vs. `Quiz Adv`

(c) `Attendance and Participation Score` vs. `Exam 1`

Figure 11: Boosted Linear Kernel classifiers in 2D

By averaging all three quiz scores into one variable `Quiz Adv` we can also construct a data set matrix $X \in \mathbb{R}^{n\times 3}$ and visualize the 3D hyperplane separating the data. By solving the equation $f(x_1, x_2, x_3) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$ for x_3 where x_1, x_2, x_3 are the variables `Quiz Adv`, `Exam 1`, `Attendance and Participation` one can plot the hyperplane in a 3D graph. As shown below from two different angles the hyperplane tries to slice between the blue region which are the passing grades and red region which are the failing grades.

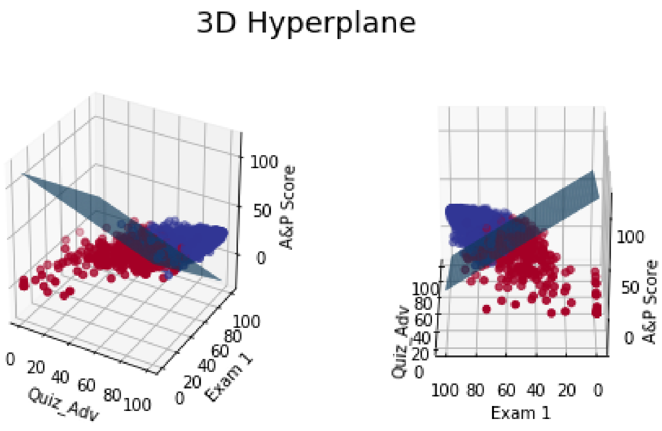


Figure 12: Boosted Linear Kernel classifier in 3D

As mentioned above when talking about the 2D graphs the passing and failure prediction becomes very difficult around the hyperplane which is further shown by Figure 13 on the misclassified observations. One can calculate the distance from the hyperplane by the formula:

$$D(X, W, B_0) = \frac{XW + B_0}{\|W\|} \tag{2}$$

where $X \in \mathbb{R}^{n \times p}$ is the data set matrix, $W \in \mathbb{R}^{p \times 1}$ is the vector of the weights, and $B_0 \in \mathbb{R}^{n \times 1}$ is the vector of the intercepts. On the x axis is the distance from the hyperplane using the boosted SVM model and on the y axis is the number of misclassified observations that had that distance. From Figure 13 the greatest number of misclassified observations are between [-5,10] distance away from the hyperplane which coincides with the difficulty of predicting pass or fail around the hyperplane. From the misclassified data 59.83% were a false positive and 40.17% were a false negative. The histogram also supports this by being right skewed meaning more people where passing at the drop add deadline and failed than failing at the drop add deadline and passing.

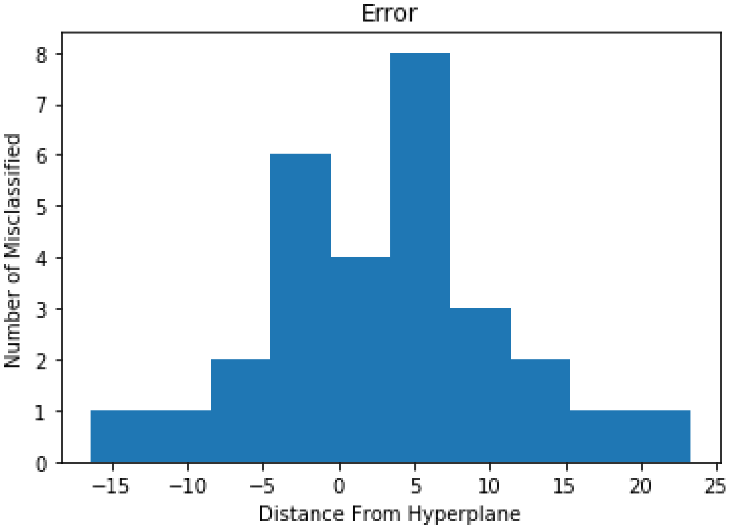


Figure 13: Error Graph of misclassified observations

Overall the SVM model does a very good job of separating the data between passing grades and failing grades. But if the scores are on the borderline of passing and failing it may not come down to a model prediction on whether or not a passing grade will be achieved but ultimately on the student's determination to pass the class.

4.4 Decision Trees

Decision trees are promising because they output a binary tree that determines a route to classification. This is a very interpretable approach and could be presented to a student to provide a quick glimpse of their course grade. The two big parameters that determine the produced tree are the depth of the tree and the criterion used for determining the best partition to make. The depth of the tree is the true factor of the accuracy of the model. While a deep tree will lead to higher accuracy, it also restricts the model and is the leading contributor to over-fitting the data. For instance, when attempting to classify GPA, a tree depth of 17 will result in a perfect fit for the entire dataset. However, there are 2^{17} result nodes which mainly contain 1 student data point. With a simpler tree of depth 3, 90% accuracy can still be achieved and still allow for error with unseen data. Now for the secondary factors, the gini index and entropy. The gini index is the probability a random sample of data is misclassified if a random label is picked, while entropy is the measure of information gained. In the end, both methods provide similar results but small improvements are beneficial.

One disadvantage to the decision trees is that training is best done using the entire data set instead of utilizing a train-test data split. While this will provide a better model for the training data, it does not guarantee that it will be optimal on unseen data. Using the decision trees to classify grade provided a challenge as there are 13 potential output grades including an incomplete. Below is a decision tree produced with a depth of 3 and using entropy.

This tree provides a 41.2% accuracy. The reason for the low accuracy can be seen with the terminal nodes of the tree. Since we only have 8 terminal nodes when there are 12 potential grade outcomes, there can only decision outcomes for

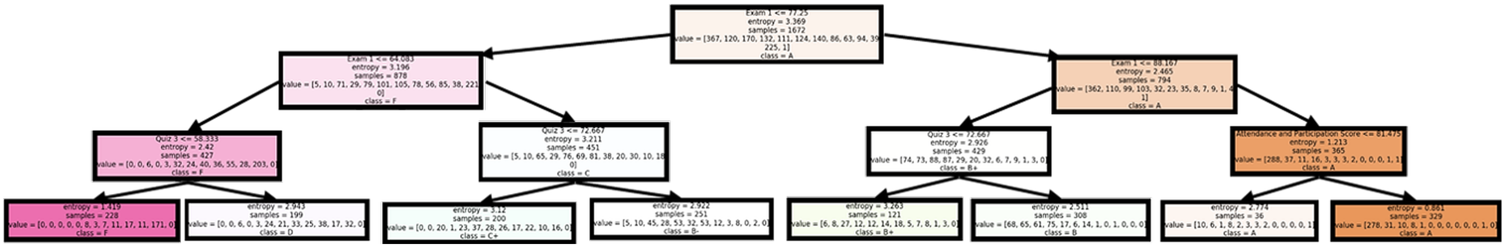

Exam 1 <= 77.25
entropy = 3.369
samples = 1672
value = [367, 120, 170, 132, 111, 124, 140, 86, 63, 94, 39
225, 1]
class = A
Exam 1 <= 64.083
entropy = 3.196
samples = 878
value = [5, 10, 71, 29, 79, 101, 105, 78, 56, 85, 38, 221
0]
class = F
Exam 1 <= 88.167
entropy = 2.465
samples = 794
value = [362, 110, 99, 103, 32, 23, 35, 8, 7, 9, 1, 4
1]
class = A
Quiz 3 <= 58.333
entropy = 2.42
samples = 427
value = [0, 0, 6, 0, 3, 32, 24, 40, 36, 55, 28, 203, 0]
class = F
Quiz 3 <= 72.667
entropy = 3.211
samples = 451
value = [5, 10, 65, 29, 76, 69, 81, 38, 20, 30, 10, 18
0]
class = C
Quiz 3 <= 72.667
entropy = 2.926
samples = 429
value = [74, 73, 88, 87, 29, 20, 32, 6, 7, 9, 1, 3, 0]
class = B+
Attendance and Participation Score <= 81.475
entropy = 1.213
samples = 365
value = [288, 37, 11, 16, 3, 3, 3, 2, 0, 0, 0, 1, 1]
class = A
entropy = 1.419
samples = 228
value = [0, 0, 0, 0, 0, 8, 3, 7, 11, 17, 11, 171, 0]
class = F
entropy = 2.943
samples = 199
value = [0, 0, 6, 0, 3, 24, 21, 33, 25, 38, 17, 32, 0]
class = D
entropy = 3.12
samples = 200
value = [0, 0, 20, 1, 23, 37, 28, 26, 17, 22, 10, 16, 0]
class = C+
entropy = 2.922
samples = 251
value = [5, 10, 45, 28, 53, 32, 53, 12, 3, 8, 0, 2, 0]
class = B-
entropy = 3.263
samples = 121
value = [6, 8, 27, 12, 12, 14, 18, 5, 7, 8, 1, 3, 0]
class = B+
entropy = 2.511
samples = 308
value = [68, 65, 61, 75, 17, 6, 14, 1, 0, 1, 0, 0, 0]
class = B
entropy = 2.774
samples = 36
value = [10, 6, 1, 8, 2, 3, 3, 2, 0, 0, 0, 0, 1]
class = A
entropy = 0.861
samples = 329
value = [278, 31, 10, 8, 1, 0, 0, 0, 0, 0, 0, 1, 0]
class = A

Figure 14: GP Decision Tree with depth 3

a subset of the grades. However, we are given counts of how much of each grade is classified into the terminal node. This can be used when making a prediction. Below shows the percentages when predicting the final grade for a student who scored 80% on all assignments before the deadline.

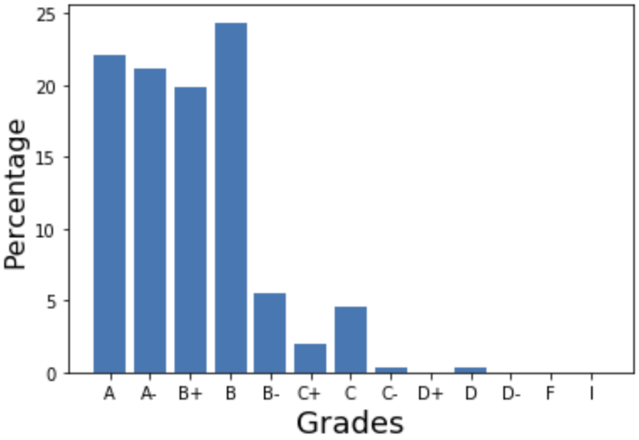


Figure 15: Grade Distribution according to decision tree.

This output provides an output that is similar to the nearest neighbors as it shows which grade groups you would be most likely to be a part of. While the tree does not provide a direct solution, it can provide a helpful result to students.

Improvements to the decision tree come when looking at a C- Pass/Fail. In this case, the tree only needs to label output as either pass or fail which removes the issue of too many output classifications as seen with the grades. Now, it is only either correct or not. Starting with a similar tree of depth 3 and using entropy yields the below result.

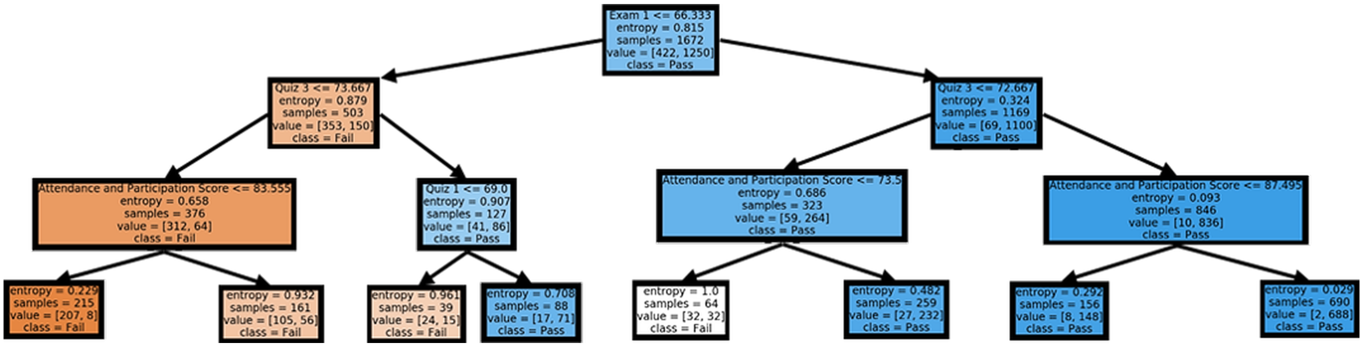


Figure 16: PF Decision Tree with depth 3

With this new model, there is 90.13% accuracy. This is a massive out of the box prediction when compared to the grade tree. This result can be slightly improved through changing the depth and criterion parameters. It was found that the optimum performance was a tree of depth 4 using gini which achieves 92% accuracy. But with this increased depth comes the trade-off of a more intricate graphic for students to look at. Incredibly, even a simple tree with depth 1 or 2 can provide 86% and 89% accuracy respectively. This shows that even a very simple tree could be presented to students to aid students and still be near as useful as a more complex model.

An interesting case study is to analyze some of the misclassifications made by the Pass/Fail tree to understand potential flaws in the model. As a whole, there were 165 misclassifications. Of that, 54 were false positives and 111 were false negatives. When broken down into letter grade achieved, it can be shown that a majority of the errors occur in the C+ to D- range which is as expected. Students close to the pass/fail division may be misclassified due to the rigid structure of the tree. However, there are students in the A, B, and F range that are misclassified and will be interesting to analyze.

Assignment Name	Percentage (%)
Quiz 1	87
Quiz 2	97
Exam 1	90
Quiz 3	57
Attendance	28
Prediction	*Fail*
Final Grade	*Pass* (A-)

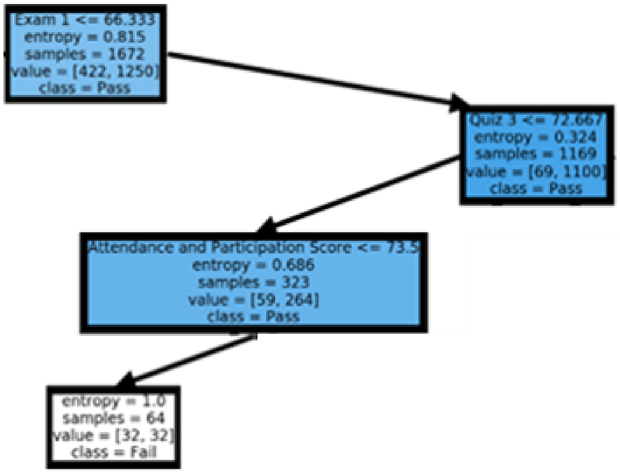


Figure 17: Student 1643 pre-deadline grades with decision tree path

For the A- students, they are misclassified because of a combination of a low `Quiz 3` score and a low `Attendance and Participation Score`. When looking back at the tree created above, it can be seen how this path is outlined on the tree. In Figure 17, Student 1643 is shown as an example of the decisions the tree makes that leads to a false negative. In this case, it returns fail but the classification distribution is split 50/50 between passing and failing but is labeled fail.

Assignment Name	Percentage (%)
Quiz 1	12
Quiz 2	48
Exam 1	70
Quiz 3	100
Attendance	61
Prediction	*Pass*
Final Grade	*Fail* (F)

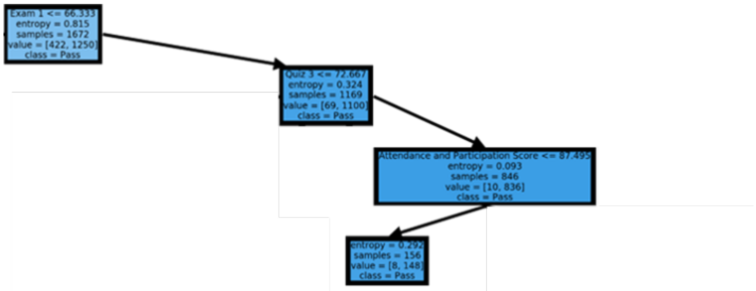


Figure 18: Student 930 pre-deadline grades with decision tree path

When looking at the misclassified students whose final grades were F, they had pretty good starts to the semester and would have in been in a position to pass. However, they ended up with poor later quiz scores and exam scores, or even scored 0 on exams, which led to their eventual fail. In Figure 18 above, Student 930 is shown as an example of the decisions the tree makes that leads to a false positive. In this case, it returns pass but because Student 930 performed well on the metrics the tree favors for a pass like `Exam 1` and `Quiz 3` despite their poor performance on other assignments. In the end, this student performed poorly on remaining assignments. From these, we can see that these cases are hard to fix as they are anomalies, but helps to illustrate some of the issues that arise with decision trees and the rigid structure they provide. Overall, decision trees provide a simple model and illustration to students of their performance and have fairly high accuracy which allows them to be a useful model to provide to students.

4.5 Random Forests

Random forests are a popular machine learning classification method that builds on decision trees. Instead just one tree being constructed and used for classification, many trees are constructed and a prediction is made base on the composite of all the trees. A failure of decision trees is that the simplicity of the model can sometimes cause bizarre misclassifications, such as one outlines above where a tree misclassifies a student who earns a A- as failing. The random forest method is considered more robust. However, this method loses the transparency and intuitiveness of a decision tree. After brief experimentation, we found a random forest with a maximum depth of three to be most accurate with acceptable efficiency. As summarized in the results section, this method did not provide great results, compared to the other methods. On the binary task of C-Pass, the method performed fair. However in predicting GPA, this method was the worst of our top methods. We believe that this is due in part to the forest treating the task as a strict classification task, without the understanding that the grades are ordered in a near continuous fashion.

4.6 Principal Component Analysis

Despite Occam's Razor, it would be insightful to see if more abstract methodologies perform similarly or even better than the simple methods. Unfortunately this is not the case. We will still describe the principal component analysis (PCA) of our data and note potential flaws in the representation of the features in component space. We seek to identify a low-dimensional space in which the data can be represented with little loss of information. Let $X_i \in \mathbb{R}^{1672\times i}$ with $i = 2, \ldots 5$, be the chronological data set with target columns y_{PF} and $y_G P$ as before. Computing the eigenvectors $\{v_1, \ldots, v_i\}$ of $X_i^T X_i$ we obtain a linear combination $v_1 X_1 + \cdots + v_i X_i$ with maximum variance. Projecting onto the component space $v_1 X_1, \ldots, v_i X_i$ provides a transformation of the data. Selecting fewer than the maximum number of components loses some information, but we can measure the remaining variance described by each component. This transformation is plotted below in Figure 19 along with the projected data.

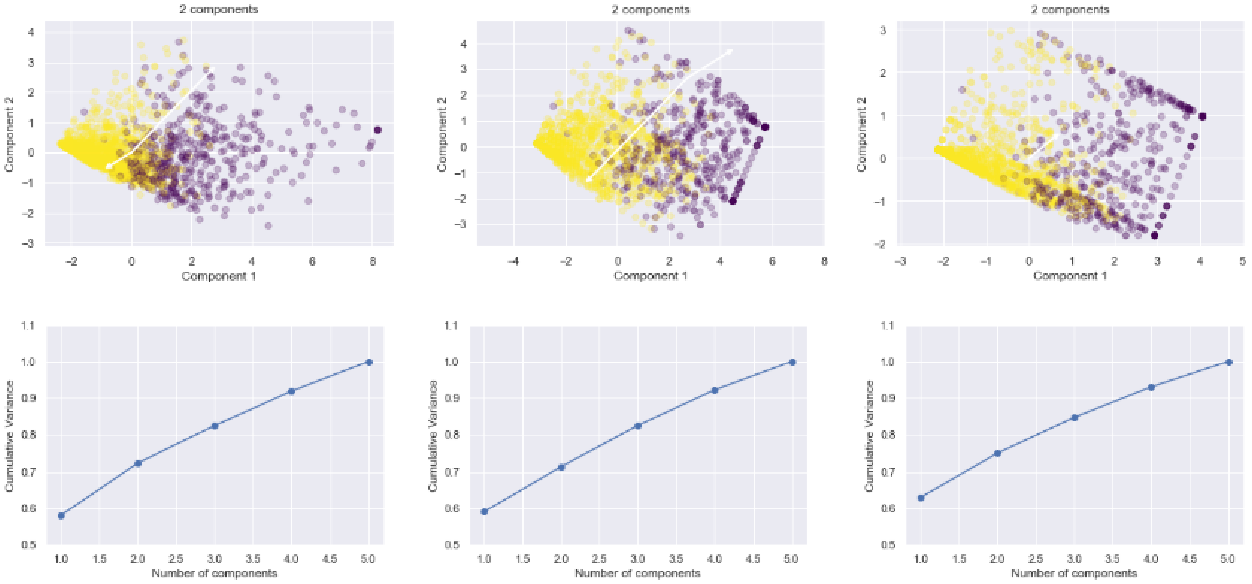


Figure 19: (From left to right) PCA across the standard scaled chronological data, continuously transformed GP data, and the scaled continuously transformed GP data. The following row is the corresponding explained variance ratio vs. the number of components. Note the difference in starting position for component 1.

To accompany this visualization of the data we can now pipeline this result into SVM to differentiate between passing and failing cases while varying the number of components. The prediction results in Figure 20 are averaged over 10-fold cross validation. However, PCA actually performs worse than regular SVM as the linear combination maximizes over variance but not the actual shape of our data. This is due to passing instances having very small variance, but borderline cases having much larger variance in scores. Additionally, the data is not linearly separable over the chronological features so a linear kernel cannot achieve absolute accuracy. An RBF kernel is then applied and produced only minor differences on the prediction error. The results from PCA are still dismal compared to the other methods, and are omitted from the final results plot.

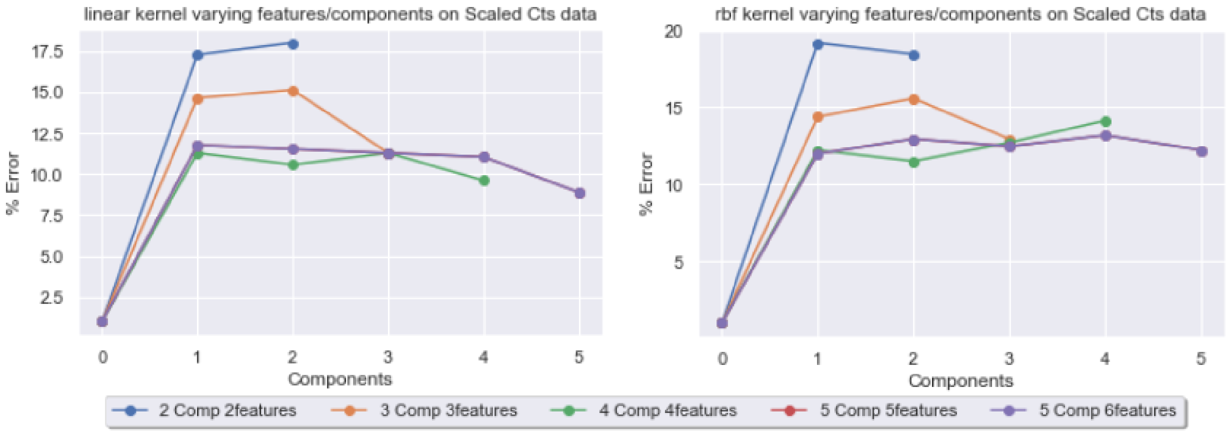


Figure 20: Linear and RBF kernel applied to the PCA of the chronological continuously transformed GP data. The PF prediction result is averaged over 10 trials.

4.7 Gradebook Function

In predicting the final letter grade and whether or not a student will pass the course, perhaps the most obvious way, as the way that is naturally used by many students and teachers, is to simply look at what a student's grade is at the time, and predict the final grade as that grade. We call this method the gradebook method.

In order to be consistent with the other models we use, the gradebook method uses only Quiz 1, Quiz 2, Exam 1, and Quiz 3. Additionally, as in the other methods, Attendance and Participation Score is used at all the different timestamps.

For $t \in \{1, 2, 3, 4\}$ corresponding to the times where Quiz 1, Quiz 2, Exam 1, Quiz 3, are recorded, the gradebook prediction grade at time t is calculated as

$$g_t = \frac{.05 * \texttt{Attendance and Participation Score} + \sum_{i=1}^{t} w_i s_i}{.05 + \sum_{i=1}^{t} w_i} \tag{3}$$

where w_i is the weight of the assignment at time i and s_i is the score on the assignment at time i. Then the percentage grade g_t is converted to a letter grade, or pass/fail based on the scale in the syllabus. Using only the assignments used in the other methods, this is the grade that would appear to the instructor and student online.

The results are summarized in Results section, but this method performs remarkably well. At nearly every timestamp, for both the GPA prediction and the C-Pass prediction, this method is nearly as good as any other method we try, including a MAE of 0.3 in predicting GPA at the deadline and an accuracy rate of just above 92% in predicting C-Pass at the deadline. This shows empirically that on the whole, student's grades do not fluctuate very much after the deadline. In fact, in early experimentation we found that after the deadline, in particular after Exam 2, the gradebook predictions became so accurate that we abandoned other prediction methods after the deadline.

4.8 Altered Gradebook Function

The gradebook method can be summarized as taking a linear combination of the grades available, then converting (non-linearly) to a prediction of GPA of C-Pass. The weights of the linear combination are determined by weights of the assignments in the gradebook. The success of the gradebook method inspires a generalization of the gradebook method, which we call the altered gradebook method. A quickly trained model is fit as follows. Starting with the weights in the gradebook, perturb each weight independently in increments of 1% by up to 2% for A&P and quizzes, and in increments of 2% and up to 4% for the exam. For each new weighting scheme, find percentage prediction and conversion to GPA or Pass/Fail as in the gradebook method. Then select the weighting scheme that gave the best MAE or accuracy rate, as desired. This weighting scheme can then be tested on a test set in the same way.

As summarized in Results section, this method is a top performer. In repeated cross validation, with training and testing, this method edges out all the other methods at the deadline in predicting both GPA and C-Pass. It achieves a MAE of under 0.3 in predicting GPA and has an accuracy rate of over 93% at the deadline.

When using the entire data set as a training set and allowing more relaxation on the weights, this method was able to achieve a 94% accuracy rate for C-Pass at the deadline. To achieve this rate, the data was first preprocessed by replacing one quiz where a student scored less than 10% with the next lowest pre-deadline quiz score, in order to reflect the policy of taking the best four of five quizzes for the final grade. The weights used to achieve that score were: `Quiz 1`: 8%, `Quiz 2`: 3%, `Exam 1`: 19%, `Quiz 3`: 7%, `Attendance and Participation Score`: 8%. Additionally, the cutoff point for passing on this altered method was shifted to 68%, rather than 67%.

4.9 Neural Network

Neural Networks, or multi-layer perceptrons, are a popular tool for classification tasks like ours. Data from the input is fed forward through a system of neurons. At each layer, neurons take in data from the last layer, then form a linear combination to feed to the next layer, until the data reaches the output layer, at which point a prediction is made. We dipped only lightly into this very deep pool for this project, so future work using neural networks may be rewarding. After some experimentation with the parameters, we were unable to significantly improve on the "out of the box" implementation from `sklearn`. The parameters can be found on the `sklearn` documentation. As summarized in the Results section, this method was among the top performers, but did not separate itself from the other methods. Neural networks have the additional drawback of being very slow to fit and test. In fact, due to lack of convergence in an acceptable time frame, when fitting the neural network to the GPA data, we found it necessary to round the data to the nearest integer (i.e. A- rounded to A, C+ rounded to C). Additionally, this methods lacks the transparency and intuitiveness of other methods, such as nearest neighbors and decision trees.

4.10 Logistic Regression

Logistic regression is another common data science tool used for classification. Similar to the neural networks, after experimentation, we were unable to improve on the default parameters found in the `sklearn` documentation. In the binary setting of predicting C-Pass, every time step, this method was a top performer, as summarized in the results section. In predicting GPA, logistic regression was less successful. We believe that this is due to the near continuous nature of the grade points. From the perspective of logistic regression, it is predicting the grade points as separate categories, with no knowledge or understanding that they are ordered. One preprocessing method that we found helpful was to scale the percentage data non-linearly as in the continuous transform of Figure 1. This scaling helps to accentuate the differences between the grade point categories. Logistic regression is also not an entirely intuitive method for a student-facing tool.

4.11 Some Ensemble Methods

To predict C-Pass at the deadline, we made an ensemble method that combined the results of our top 5 methods, other than the standard gradebook. We used nearest neighbors, SVM, altered gradebook, neural network, and logistic regression. This method simply took the consensus of the methods to predict if a student will pass. This method did not perform any better than the top method, altered gradebook, performed on its own.

5 Results

5.1 Grade Points Prediction

We summarize the results of the different methods predicting GPA with the following graph. The data set for this comparison of methods is obtained as follows. First, anomalies were removed. An anomaly here is defined as student whose grade at the deadline is a C+ or higher that scored below a 40% on the final exam. We suggest that the handful of students in this category had a life event outside of class, which no algorithm could predict. At this point, the gradebook method is tested. As the model does not need to be fit, the whole set is used as a test set, and the error is computed.

Next, for the altered gradebook model, if a student scored less than 10% on a quiz, that score is replaced with the next lowest score, reflecting the policy of taking the best 4 of 5 quizzes in the final grade. Then the altered gradebook model is fit and tested on that dataset as follows. The average test error is recorded over 50 repetitions of training and testing (with replacement) using a 20% testing size.

Next, the percentage grades are scaled using the continuous function of Figure 1. Then the data is normalized. We found this scaling and normalization helps all the remaining methods perform better and converge quicker.

Finally, all the remaining methods are fit and tested using the same process. The average error is recorded over 50 repetitions of training and testing (with replacement) using a 20% testing size.

This process results in scores that are "apples to apples" comparisons between the methods, where each method performs optimally without over fitting.

On the x-axis, detailing assignment completed, the data used for that time step are `Attendance and Participation Score` in addition to all assignments coming before it.

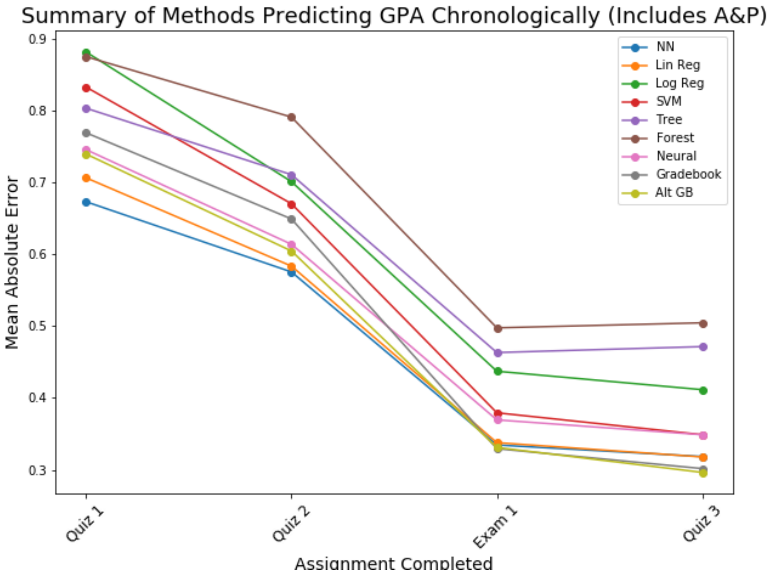


Figure 21: Summary of Methods Predicting GPA

5.2 Pass/Fail Prediction

We summarize the results of the different methods predicting C-Pass with the following graph. The data sets used and testing procedures are identical to those detailed above in the Grade Points Prediction section. These procedures allow for fair comparisons between the methods, where each method performs optimally without over fitting.

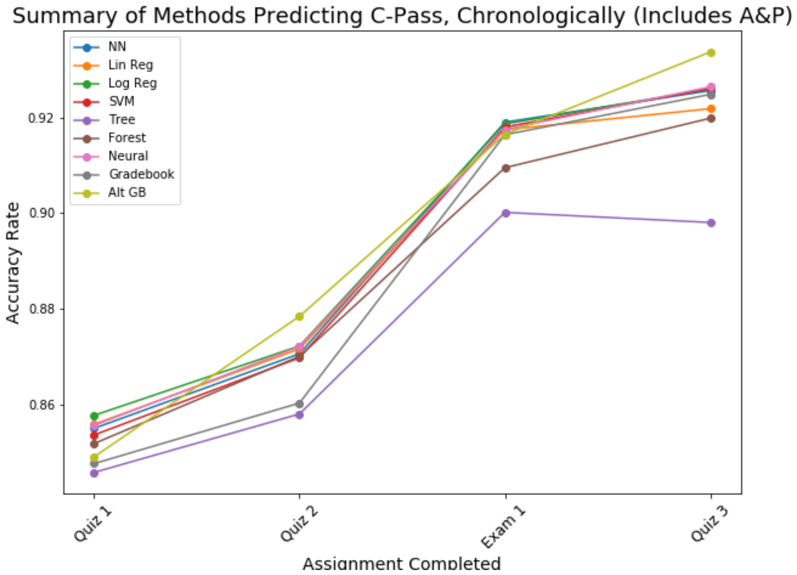


Figure 22: Summary of Methods Predicting C-Pass

Finally, we restrict our focus to perhaps the most significant prediction at the most significant time. That is, the C-Pass prediction at the change of status deadline. In order to obtain predictions at the individual student level, we perform one vs. rest training a testing with the data at the deadline. That is, for each student, we train the top performing methods on all other students, then record the predictions of those methods on that individual student. The results are in the file labeled `c_pass_summary.csv`. The accuracy measures may but differ very slightly from those in the C-Pass summary figure as the testing procedures are slightly different.

	Accuracy (%)	**False -** (%)	**False +** (%)
Nearest Neighbors	93.0	2.4	4.6
SVM	92.6	2.7	4.7
Gradebook	92.5	3.1	4.4
Altered Gradebook	**93.9**	**2.1**	**4.0**
Neural Network	92.5	2.9	4.6
Logistic Regression	92.5	2.7	4.8
Ensemble	93.0	2.7	4.3

Here we define a false negative for a method as follows. The method predicts the student will fail, but the student passes in the end. A false positive is defined as the reverse. The method predicts the student will pass, but the student fails in the end.

The first striking observation from the table is that for each of the methods of interest, the rate of false positives is nearly double the rate of false negatives. Those familiar with the context of the class will not be surprised. There are more students that fade after the halfway point than students who become stronger. After manually examining the misclassifications for each method, the misclassifications seemed to be with students who were barely failing at the deadline, then raised their scores after the deadline just enough to pass, or students who were barely passing at the deadline then their scores started to fall. We could not find any extreme outliers.

6 Future Work

As alluded to in the neural network, random forests and logistic regression sections, it is possible that more investigation and alteration of input parameters could produce marginally better results, particularly with neural network as this is a very deep field. The aforementioned literature considers convolutional neural networks, Naive Bayes, deeper decision trees, and custom methodologies. The authors hope to revisit these methods to investigate further when time permits.

All of our methods were built and measured by a single output prediction, either the grade points or a binary prediction on C-Pass. We found that the majority of the cases that our methods misclassify are very close to the decision boundary and with small perturbations could change sides. This motivates creating methods that predict probability of passing or a probability distribution on grade points. We would then have to alter our loss function to accommodate. Predicting probability of passing would also be more helpful to students. In the future, we would like to further develop and optimize our methods to make probabilistic predictions, in addition to discrete predictions.

Finally, we hope to apply all of our methods to other data sets. We used data from the MATH221 course, and hope to use our methods on MATH241, which follows a similar structure. Additionally, our research could be employed to study MATH115 (an introductory pre-calculus course) as well as data from courses outside the mathematics department.

References

[1] Ginika Mahajan and Bhavna Saini. Educational data mining: A state-of-the-art survey on tools and techniques used in edm. *International Journal of Computer Applications Information Technology*, 2020.

[2] Tim McKay, Kate Miller, and Jared Tritz. What to do with actionable intelligence: E2coach as an intervention engine. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 88–91, 2012.

[3] Yannick Meier, Jie Xu, Onur Atan, and Mihaela Van der Schaar. Predicting grades. *IEEE Transactions on Signal Processing*, 64(4):959–972, 2015.

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

[5] Agoritsa Polyzou and George Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, 2(3-4):159–171, 2016.

[6] Jake VanderPlas. *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media, Inc., 1st edition, 2016.

[7] SN Vivek Raj and SK Manivannan. Predicting student failure in university examination using machine learning algorithms. *forest*, 84(66.14):0–24.