Supporting Student Success with Machine Learning and Visual Analytics

by

Riley WEAGANT

A thesis submitted to the
School of Graduate and Postdoctoral Studies in partial
fulfillment of the requirements for the degree of

Master of Science in Computer Science

Faculty of Science
Department of Computer Science
Ontario Tech University
Oshawa, Ontario, Canada

August 2019

THESIS EXAMINATION INFORMATION

Submitted by: **Riley Weagant**

Master of Science in **Computer Science**

Thesis title: Supporting Student Success with Machine Learning and Visual Analytics

An oral defense of this thesis took place on August 8, 2019 in front of the following examining committee:

Examining Committee:

Chair of Examining Committee	Dr. Jeremy Bradbury
Research Supervisor	Dr. Christopher Collins
Examining Committee Member	Dr. Faisal Qureshi
Thesis Examiner	Dr. Mark Green, Ontario Tech University

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

Declaration of Authorship

I, Riley WEAGANT, hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I authorize Ontario Tech University to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize Ontario Tech University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

Signed:

_____

Date:

_____

Statement of Contributions

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication. I have used standard referencing practices to acknowledge ideas, research techniques, or other materials that belong to others. Furthermore, I hereby certify that I am the sole source of the creative works and/or inventive knowledge described in this thesis.

Acknowledgements

First, I would like to thank my supervisor Dr. Christopher Collins for his unwavering support and guidance through both personal and academic hurdles. Dr. Collins consistently allowed me to work at my own pace and was always available for help when I needed it.

I would also like to thank Dr. Adam Bradley for his honest feedback and countless pep talks along the way. His positive attitude and candor were always appreciated.

I also need to thank my fellow lab members for their invaluable experience and feedback, and encouragement through my worst days.

Lastly I must express my profound gratitude to my parents and siblings, closest friends, and my partner for their unconditional support. This achievement would not have been possible without them. Thank you.

ONTARIO TECH UNIVERSITY

Abstract

Faculty of Science
Department of Computer Science

Master of Science

Supporting Student Success with Machine Learning and Visual Analytics

by Riley WEAGANT

Post secondary institutions have a wealth of student data at their disposal. This data has recently been used to explore a problem that has been prevalent in the education domain for decades. Student retention is a complex issue that researchers are attempting to address using machine learning. This thesis describes our attempt to use academic data from Ontario Tech University to predict the likelihood of a student withdrawing from the university after their upcoming semester. We used academic data collected between 2007 and 2011 to train a random forest model that predicts whether or not a student will dropout. Finally, we used the confidence level of the model's prediction to represent a students "likelihood of success", which is displayed on a beeswarm plot as part of an application intended for use by academic advisors.

Contents

List of Figures

List of Tables

List of Abbreviations

CSV	**C**omma **S**eparated **V**alues
GPA	**G**rade **P**oint **A**verage
CSRDE	**C**onsortium for **S**tudent **R**etention **D**ata **E**xchange
FTE	**F**ull **T**ime **E**quivalent
OIRA	**O**ffice of **I**nstitutional **R**esearch and **A**nalysis
RO	**R**egistrar's **O**ffice
AI	**A**rtificial **I**ntelligence
CUDO	**C**ommon **U**niversity **D**ata **O**ntario
CGI	**C**ommon **G**ateway **I**nterface

Chapter 1

Introduction

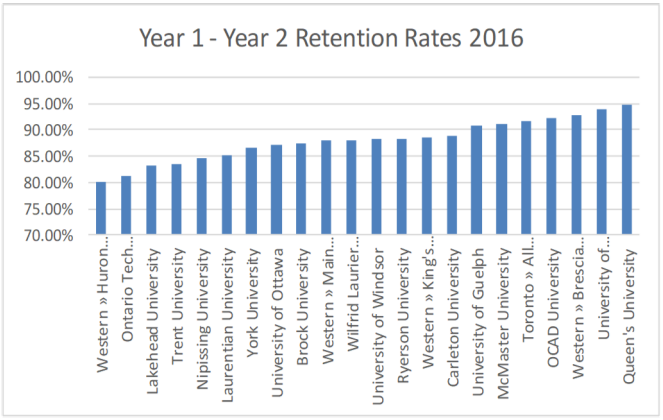

FIGURE 1.1: 2016 Ontario university retention rates for students moving from first to second year. Ontario Tech University is the second bar from the left with a retention rate of just over 80%.

As of 2016, 79% of Ontario Tech University students returned for their second year. This is among the lowest rates in Ontario [5]. High student attrition rates lead to financial pressures on the institution and loss of reputation. The decision to withdraw is complex and varies from student to student [22]. Our curiosity about why students withdraw and the availability of institutional historic data led us to a data-driven approach to assess the likelihood of retention. The Retention Dashboard, as described in Section 2, was the first step to analyzing the problem. The final product, while useful for exploratory analysis and finding surface trends, did not answer our question about why students withdraw. However, we were able to see a significant trend of poor academic performance leading to early withdrawal. Exploring this further, we asked ourselves, can we help students succeed when they would otherwise withdraw? Our hypothesis is that we can train a machine

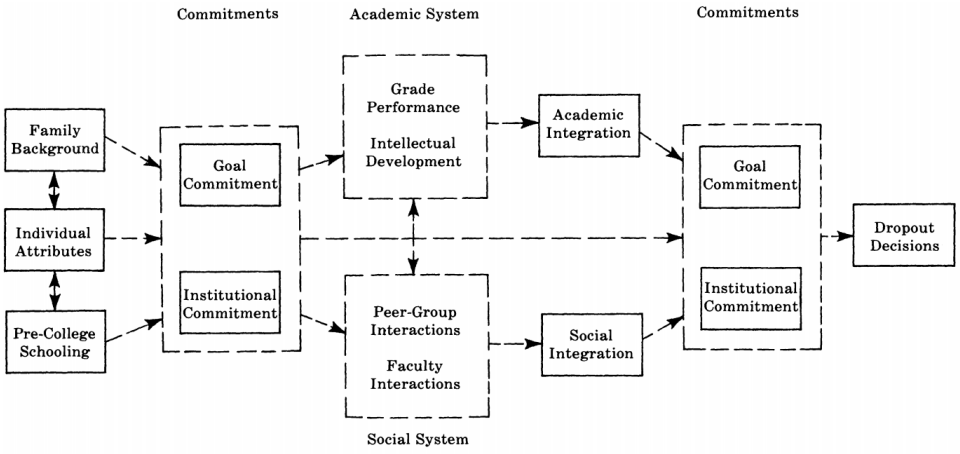

FIGURE 1.2: Conceptual model of dropout behaviour proposed by Vincent Tinto [22]. This model describes the dropout process as a sequence of interactions between the student and the academic and social systems of the institution. The experience of these interactions cause the student to continuously modify their goals and commitments in ways that impact attrition.

learning model using historical course and grade information to predict the likelihood of success for a current student based on the courses they wish to take in the future.

The most recent Common University Data Ontario (CUDO) retention rate data available is from 2016, Figure 1.1. The 2016 statistics includes first year students that were admitted in 2008 and follows them through to graduation. Approximately 19% of students admitted to the university did not return for their second year. When students who withdrew some time after their second year were included, this withdrawal rate increases to 25%. Supporting student success is important at every level of study, admittedly more so in year 1.

1.1 Motivation

Identifying risk factors of student attrition has been well investigated for many years and continues to be an active area of research [18]. Vincent Tinto designed a theoretical model of dropout behaviour (Figure 1.2) that attempts to describe the decision to withdraw as a combination of social integration, academic performance, and level of commitment to personal goals and the

institution [22]. In 2010, Tinto wrote that these theoretical models focus more on describing the problem than defining a course of action [23].

Defining a course of action is a challenge, especially when it comes to factors beyond academic performance. Academic performance is nuanced in that personal circumstances and struggles will impact the grades achieved in a semester. No matter the circumstances, the decision to withdraw is not always voluntary. Students who under perform academically are at risk of dismissal if they do not meet certain GPA thresholds. A student is placed on academic probation if their overall cumulative GPA falls below 2.0. Maintaining a semester GPA above 2.0 will keep the student from being suspended, but they remain on probation until their overall cumulative GPA is above a 2.0. Many retention prediction systems aim to notify the student or advisor that the student is at risk of withdrawing from the institution [6][12]. These "Early Warning Systems" play an important role in student success. In general, early warning systems flag an incoming or first year student as likely to withdraw for some reason.

Our goal was to take the concept of the early warning system and enhance it to describe the likelihood of success for students at any level of study. We present visualizations and interactions to allow analysis beyond the initial flagging that could help the student recover and see some potential positive future paths.

1.2 Contributions

The main contributions of this work are as follows:

- A machine learning model to accurately predict whether or not a student will withdraw from the institution using historical academic data.
- A visual analytics system that uses predictor confidence levels to represent the likelihood of success given the chosen course set for a given semester.

Chapter 2

Background

In this chapter I will review some areas of work that are closely related to my research including student retention and predicting student attrition using a subset of available student data.

2.1 Student Retention

Student retention is a problem at most post-secondary institutions in North America. The complex nature of student retention has been explored in different disciplines since the 1940's [18]. In these cases, both academic and non-academic factors were considered. Living in a data-centric world, we have the opportunity to examine these factors more thoroughly and begin to understand how they relate to each other, and to student retention in general.

Vincent Tinto is well known in the Educational Research community for his work exploring student retention. His work tries to disambiguate the reasons for discontinuing studies noting that there is a significant difference between students that withdraw due to academic failure compared to voluntary withdrawal [22]. The models outlined in this paper aim to differentiate between different types of dropouts and the classification of factors contributing to each type.

Tinto explores the concept of retention and the negative impact of focusing on theoretical concepts that aim to explain this phenomenon rather than defining actionable metrics. These theoretical concepts have led to metrics that can define student engagement or academic and social integration.

However, many of these concepts focus on external factors such as peer integration, and commitment to the institution and personal goals, which cannot be directly influenced by the institution [23].

2.2 Predicting Attrition

Efforts to predict attrition using machine learning have increased over the last few years as AI continues to be more accurate and easy to apply to a wide range of problems. Researchers tend to focus on prediction as a type of analysis tool whereas students with similar predictions are analyzed to find significant similarities and differences that could explain their similar outcome. One such example of this work can be found in Kathleen Pittman's PhD Dissertation [16]. Pittman compares several machine learning techniques with the goal of understanding how this type of analysis will benefit post-secondary institutions developing retention strategies. The analysis included both full-time and part-time students at all year levels. Like the work presented here, Pittman aims to shift the focus from the retention of first-year, full-time students to the entire student body. It is because of this work that we chose to test the machine learning algorithms that we did: linear regression, random forest, Naïve Bayes, and neural networks.

Several different methodologies have been used to try to address the problem of student retention. Barber predicts whether or not a student will be successful in a given course using a logistic regression model [3], and Bayer significantly improved their prediction accuracy by including "social behaviour" which was derived from a social network [4]. Each of these systems added a new dimension to the existing approaches, but all of them relied on information that was either student-declared through a questionnaire or survey, or was personal information collected on admission. The lack of academic information used for these models means they are more suited to analyzing student engagement than predicting academic success.

Sweeney et al. proposed a system that predicts the grades a student will get in the courses taken in the next term [21]. This work is most similar to ours as it uses some historical transcript data to predict an outcome for the following term. A key difference is the amount of non-academic features, and feature engineering of academic, and instructor information. While the

model proposed in this work is quite robust and reports promising results, there is still a lot of work to be done in terms of turning the model into a working application. The amount of data and feature engineering done in this case was something we wanted to avoid. We decided to approach the problem of predicting future performance from a ”less is more” perspective in terms of training data, and to leverage model uncertainty.

Data mining has become a popular approach in the education domain to analyze student data to find causes of attrition. These approaches aim to uncover complex relationships and reveal new insights, rather than to develop a specific algorithm to extract relationships from a specific dataset. Superby et al. identified and grouped factors correlated to withdrawal from the literature. Factors were grouped into three sets including personal history, expression of involvement of the student in their studies, and student perceptions. They created a questionnaire and used data mining approaches to attempt to use these variables to predict whether or not the student will withdraw using decision tree, random forest, neural network, and linear discriminant analysis [20]. The goal was to allocate limited resources to students that need and want it the most. They comment that the prediction results were unremarkable and raise some interesting questions about the stability of non-academic factors from year to year.

Delen’s work is similar in that they used a popular data mining methodology and tested three different prediction models [8]. They conclude by saying that data mining methods can predict attrition with a good level of accuracy. Results obtained in the types of experiments conducted by Pittman [16], Superby et al. [20], and Delen [8] are promising, especially considering the similar data being used, similar models being trained, and an overall increase in model accuracy over time.

2.2.1 Massive Open Online Courses

Massive Open Online Courses (MOOCs) have gained a lot of traction through popular websites like Udacity and Coursera, etc. The number of courses and enrollment have been steadily increasing over the past few years, and so has the amount of data available [19]. A lot of recent work in educational data mining and machine learning has focused on data from these online courses.

Demographic and engagement data from MOOCs is fundamentally different than data collected through institutions that operate in a more traditional campus setting. While the majority of campus populations are made up of students one year removed from secondary school, MOOCs tend to be made up of professionals who want to upgrade their skills, and mature learners. Attrition rates of MOOCs are also significantly higher than that of a traditional campus and approach 90% [15]. Accordingly, results from studies performed on MOOCs cannot be generalized to our student population.

2.2.2 STEM Attrition

Studies that focus on retention in STEM fields are prevalent in the literature, and propose some interesting methods of analysis. Like MOOC's, working specifically with STEM dropouts means the analysis models and results cannot be generalized to the campus community as a whole.

Gerben W. Dekker et. al. attempt to predict whether or not Electrical Engineering students will drop out after their first semester, and identify success factors specific to the Electrical Engineering program [7]. The simple classifier tests resulted in accuracies between 75% and 80% which the authors point out is difficult to beat with more sophisticated models. These findings had an affect on the chosen path for this project.

More recently, Lovenoor Aulck et. al. based their predictions on a single term of academic data. Course features were condensed and represented by department. Department course grades were separated into a variable indicating whether or not students took a course offered by that department, the number of courses taken in that department, and the overall GPA of courses taken by that student in that department. "Gatekeeper courses" refers to first-year courses in physics, chemistry, biology, and math which are typically taken over two semesters (i.e. Physics I and Physics II, Calculus I and Calculus II, etc.). These course variables were included alongside demographic information, and information specific to STEM and gatekeeper courses [1]. Extending this work, Aulck et. al. introduced the concept of "STEM Affinities" [2]. STEM affinities attempt to describe the level of engagement a student has with the course material by taking into account which courses the

student is taking and whether or not they are continuing on a STEM academic path. As a student takes more courses outside of the STEM discipline, their STEM affinity drops. The STEM affinities model correlates to "STEM intention" and can help decision makers see when STEM intention shifts, and when students are more likely to dropout, or switch from a STEM path.

2.3 Visualizations for Decision Making

Visualizations display information to a user by encoding different attributes with colours, shapes, and positions. They aim to make data easier for the user to understand and interpret. People often need to make decisions based on data which is a difficult task when presented with data tables, and summary information.

Miettinen did a survey on visualization techniques for supporting multi-criteria decision making [14]. They cover several techniques in detail including bar charts, spider web charts, trees, etc. They present a summary table indicating which visualization is appropriate for different types of information. They mention throughout the paper that the data and the visualization should support the specific decision being made, and selecting the wrong visualization or data could hinder rather than help the decision making process.

There is little work that focuses on visualizing and interpreting machine learning output. Most work involving visualization and machine learning attempts to improve model performance by analyzing the input data and trying to interpret the "black box" of a machine learning algorithm. Frank et al. explored visualizing predictor class probabilities. Their approach involves plotting the class probability estimates, and colouring the rectangular background accordingly [11]. The class colours and plotting attributes are chosen by the user. While this method is not new, they provided details on how to generalize it to other classification models that can produce class probability estimates. Our work uses class probability estimates to interpret the classifier results, albeit in a different way. Our approach uses the class probabilities in an attempt to represent the uncertainty of our predictive model as a likelihood of success for a student.

Chapter 3

Retention Dashboard

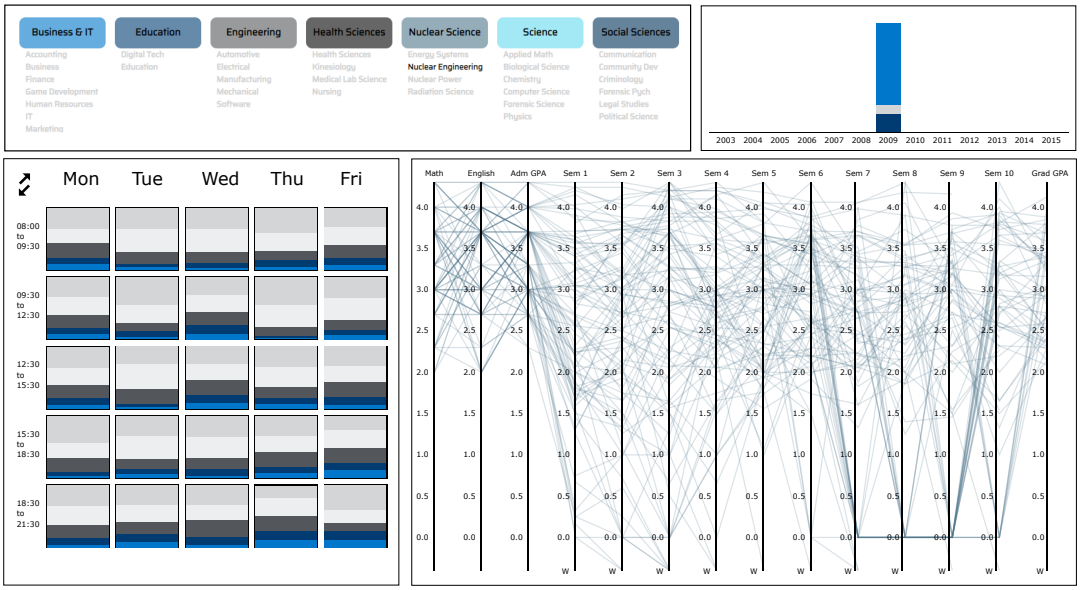

FIGURE 3.1: Retention Dashboard populated with Nuclear Engineering students admitted in 2009. The Dashboard is made up of four components: the faculty/program selector (top left), year selector (top right), timetable (bottom left), and parallel coordinates (bottom right).

3.1 Defining the Problem

Our work exploring student retention at Ontario Tech University began with the design of the Retention Dashboard[1]. In 2015, we were approached by the Registrar's Office (RO) to design a visual analytics tool to assist decision makers with the goal of improving the student retention rate at Ontario Tech

[1]The Retention Dashboard was designed in collaboration with Taylor Smith.

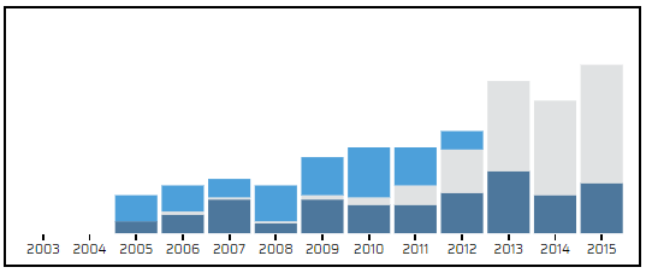

FIGURE 3.2: Year selector populated with students in Computer Science. The bottom segment of the bar (dark blue) represents the number of withdrawn students, the middle segment (light grey) represents students who are still attending the university, and the top segment (light blue) represents students who have graduated. Look at the 2009 bar, we can see that close to half of the students admitted to Computer Science withdrew before graduation, a small portion of the students are still attending the university, and close to half have graduated.

University. A specific goal of the project was to focus on revealing factors that could be influenced by the institution.

3.2 Dashboard Design

We were given access to institutional data gathered between the years 2003 and 2015. This data includes personal student information (i.e. initial city/-postal code, high school, gender, etc.), grade 12 marks in admission courses, course and grade history from the university, and academic standing history. Factors that can be influenced by the institution include admission grade cutoffs, course content, course instructors and class schedules. The visualizations designed for the dashboard draw your attention to these factors without introducing any personal or demographic information which may be distracting and misleading to the user.

The interactive panel uses different filtering techniques to update the full dashboard when a selection is made on a single visualization. As shown in Figure 3.1, the faculty/program selector is used to select Nuclear Engineering as a program, the year selector is used to select 2009 as the year of admission, and the parallel coordinates chart updates accordingly to show grade trends of students admitted to the Nuclear Engineering program in 2009. The timetable visualization also updates to show the grade distributions of courses taken by these students.

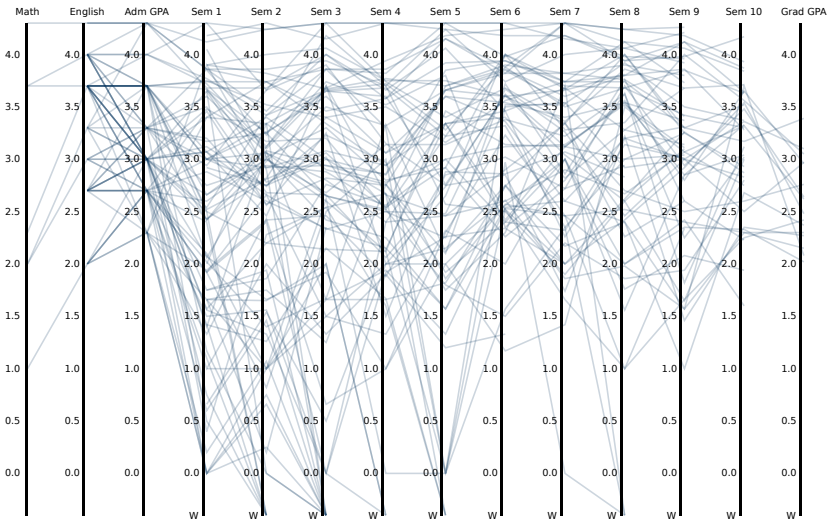

FIGURE 3.3: Parallel coordinates chart populated with Forensic Psychology students admitted in 2011. The first three vertical axes represent high school math (Math), english (English), and admission GPA (Adm GPA) respectively. The following ten axes represent their GPA in each semester that they attended the university (Sem 1 – Sem 10). The final axis represents the student's GPA at graduation (Grad GPA). Each line on the graph represents a single student, and it intersects each axis at the GPA value for that student. The 'W' at the bottom of each semester axis stands for 'Withdraw'. A line will intersect at 'W' the semester after their last reported grades.

3.2.1 Year Selector

The year selector allows the user to view students as a cohort, or group. The term "year" in this case specifically means the year a student was admitted to the university. As shown in Figure 3.2, by visualizing the distribution of students who have withdrawn (bottom, graduated, or are still attending the university, we can see the overall growth of the program and focus on a cohort with a higher withdrawal rate. We can also use these stacked bars as a selector to filter students based on their withdrawal status.

3.2.2 Parallel Coordinates Chart

The parallel coordinates chart shows the trajectory of students from the beginning of their university career to the end. Each line on the chart represents a single student. The first three vertical axes represent high school math grades, high school English grades, and admission GPA respectively. The next ten axes represent each semester at the university, followed by a single

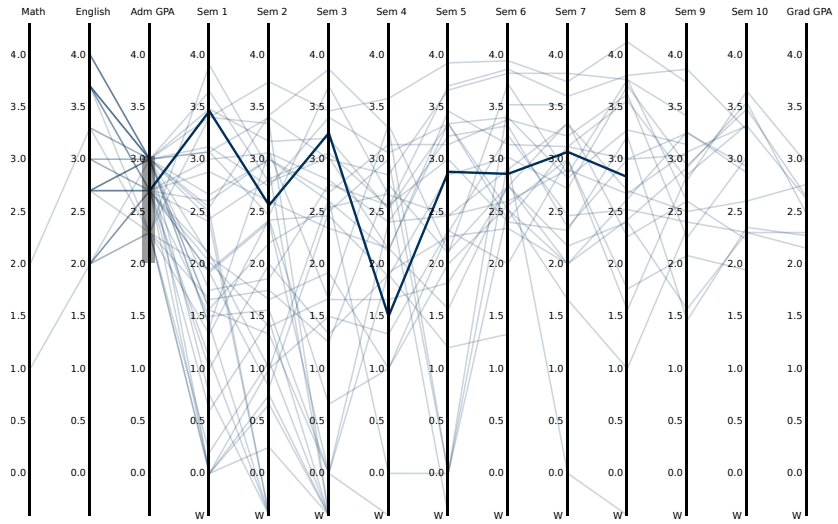

FIGURE 3.4: Parallel coordinates chart populated with Forensic Psychology students admitted in 2011. Students with an admission GPA between 2.0 and 3.0 are selected using the axis brush. A single student line is highlighted by using the hover interaction to view their GPA trajectory more clearly.

axis showing the student's GPA at graduation if applicable. Each student line intersects each axis at the relevant position. Even when filters are applied, the number of students displayed on the parallel coordinates chart can be overwhelming. As shown in Figure 3.3, the chart is cluttered when populated with students admitted to the Forensic Psychology program in 2011. We implemented two interactions to minimize the effect of the clutter:

- Axis brush filtering
- Line hover

As shown in Figure 3.4, the axis brush is used to select a GPA range. The entire dashboard is filtered according to this brush interaction. Drawing attention to a single student can be done by hovering the mouse pointer over their line. This hover interaction does not filter the dashboard since drilling down to a single student could impact anonymity.

3.2.3 Timetable

The timetable visualization shows course time slots organized on a grid. The columns of the grid indicates the day of the week (Monday – Friday) and the

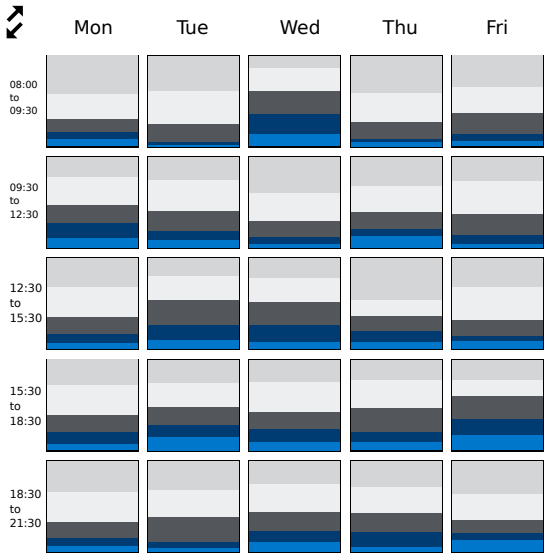

FIGURE 3.5: Timetable visualization populated with students admitted to Nuclear Engineering in 2009. Each square represents a three hour time slot during which students took courses. Inside each square is a distribution of grades received in that time slot. The top (light gray) bar represents A's, the second (white) bar B's, the middle (dark gray) bar C's, the next (dark blue) bar D's, and the bottom (light blue) bar represents F's. We can notice a much larger distribution of D's and F's on Wednesday mornings, and a much larger distribution of A's and B's on Tuesday and Thursday mornings.

rows are divided into 3 hour time slots. Each section is populated to show the distribution of grades obtained by students who took a course during that time slot.

The goal of this visualization was to explore the grade distributions across all time slots to see if there were any problem areas or trends around when courses were offered, the grades obtained, and withdrawal rates. As shown in Figure 3.5, the grade distribution in the Friday time slot from 18:30 – 21:30 is much different than the other slots. Approximately half of the grades received in this time slot are D's and F's. Further analysis is needed, but this sort of information could motivate scheduling changes in the future.

The individual time slots are clickable and filter the whole dashboard to display students who took a course in that time slot. Each individual grade bar is also clickable and filters the dashboard to display students who got the selected grade in the selected time slot.

3.3 Results

Representatives from the RO and the Office of Institutional Research and Analysis (OIRA) have been involved in the iterative design process from the beginning of the project. The usefulness of the dashboard got mixed reviews. While it was useful for exploring the data and generating a broad hypothesis, it didn't reveal any new, specific insights. We came to the dashboard with some preconceived knowledge about why students withdraw:

- Poor academic performance
- Lack of campus engagement
- Problems in personal life

Through the dashboard, we were able to confirm that academic performance is one of the main motivations to withdraw. Whether that poor academic performance is triggered by lack of engagement, or personal issues, we do not know. Although we cannot speculate on the reason for poor academic performance, but we can see that a general downtrend in grades often precedes withdrawal.

Chapter 4

Predicting Student Success

What does it mean to ”predict student success”? We defined student success to mean the completion of an academic term without voluntary or involuntary withdrawal from the institution. Academic advisors are the point of contact for students who are struggling academically, or otherwise. They help students by proposing alternative schedules, course loads, and strategies for achieving their academic goals. We learned through our meetings with advisors that their recommendations often come from knowledge of course statistics and difficulty, experience, and adjusting based on the progress of the student. Students can be hesitant to follow advice to reduce their course load or adjust their schedule to avoid taking multiple ”difficult” courses in the same semester for fear of postponing graduation. The advisors believe that having a tool that could show the potential outcomes of future course scenarios that are grounded in data would be beneficial.

Our proposed system attempts to predict the likelihood of a student succeeding in the following semester given the courses they want to take. We use a statistical sampling method to generate the sets of grades that a student is most likely to get given their past grade history. With this grade sample, we generate a set of student vectors that include all of their grade history and the generated grade sets. The random forest binary classifier uses this information to predict whether the student will be successful or not. Along with the binary classification of each student vector, the model returns a confidence score that tells us how confident the model is in its prediction. Our system uses this confidence score to represent the predicted likelihood of success.

Table Name	No. Rows	No. Columns	Size (Mb)
course_history_a_b	126924	252	380
course_history_c	131624	188	296
course_history_e	123400	303	444
course_history_f_i	123400	267	444
course_history_j_n	129570	230	333
course_history_p_w	123400	270	444
retention_raw	21086	65	7.5
studentdata_view	148075	1503	N/A

TABLE 4.1: The size and shape of our database tables, and the final view that is queried by our system. Due to the large number of columns, we were forced to split the course history into groups. course_history_a_b contains records for course codes beginning with the letter A or B. course_history_c contains records for course codes beginning with the letter C. course_history_f_i contains records for course codes beginning with the letter F, G, H, or I. The rest follow in the same fashion. Note that there is no table course_history_d because there are no course codes beginning with the letter D. A series of join operations are performed on the course_history and retention_raw tables to generate the query view studentdata_view.

4.1 Student Dataset

We were given access to student data from 2003 to 2015. We acquired the data in comma separated value (CSV) format which was directly uploaded to a MySQL database. The final database tables are described in Table 4.1.

4.1.1 Data Wrangling

The Ontario Tech course history dataset is what we used to pull course grades as features for the training data. The current format of the grade data is one row for every course each student takes. The information being fed into the predictive model is a *student vector*. We decided to use every course offered at the university as vector features. For each student, the feature value is the grade received in that course. If the student didn't take a certain course, the feature value is 0. Our predictive model only accepts numeric values. For

Letter Grade	GPA Equivalent
A+	4.3
A	4.0
A-	3.7
B+	3.3
B	3.0
B-	2.7
C+	2.3
C	2.0
D	1.0
F	0.0

TABLE 4.2: Ontario Tech letter grades and equivalent GPA value. The letter grades were replaced with their GPA equivalent in our dataset.

this reason, all of the letter grades in the dataset were converted to their numerical GPA value. As shown in Table 4.2, the GPA equivalent of the letter grade F is 0. This posed a problem for vectorizing the data. 0 will represent both failing a course, and not taking a course. We needed a way to differentiate between the two cases. We came up with three possible solutions:

1. Course indicator column
2. Change the numerical representation of 'did not take'
3. Change the numerical representation of F

The course indicator column indicates whether or not a student took a course. We implemented an indicator column for each course with value 1 if the student took the course, and 0 otherwise. There were several problems resulting from this approach. The first problem was that the indicator columns doubled the size of the dataset which led to longer model training times, and data portability issues. The second problem was that the predictive model preferred the indicator columns. It became insensitive to changes in GPA value and relied heavily on whether or not a student took a course instead of what grade they earned.

R_ID	S_ID	Adm GPA	Sem #	Per.	C1	C2	C3	C4	C5
0	234	3.0	1	1	0	0	3.3	0	0
1	234	3.0	2	1	2.7	0	3.3	0	0
2	234	3.0	3	1	2.7	0	3.3	1.0	0
3	234	3.0	4	1	2.7	3.3	3.3	1.0	0
4	234	3.0	5	1	2.7	3.3	3.3	1.0	2.3

TABLE 4.3: A student with five semesters of data separated by semester. The columns are Row ID (R_ID), Student ID (S_ID), Admission GPA (Adm GPA), Semester Number (Sem #), Persister (Per.), and Course 1–Course 5 (C1–C5). In semester 1, this student took one course, C3. In semester 2, this student took one course, C1. Note that courses taken in the previous semester have grade values in subsequent semesters, and the S_ID, Adm GPA, and Per. values are consistent. The training data does not include the student ID or semester number.

Our next option was to replace every 0 that represents a course not taken with a different value. Every student has taken significantly fewer courses than are offered at the institution which means that the majority of values in each student vector are 0s. Replacing every 'did not take' value would be time consuming, and visually confusing since 0 tends to represent a lack of information, especially when using sparse matrices and vectors.

Changing the numerical representation of F means replacing the letter grade F with a number other than its GPA equivalent of 0. We chose to use -4.3 to represent F in our data instead of 0. Theoretically, we could use any value instead of 0 since the predictive model is, at its core, a rule-based decision tree that doesn't make assumptions about the scale and magnitude of incoming data. We chose -4.3 because it represents the opposite of A+ (4.3 GPA).

The academic year is divided into three four-month sections. At Ontario Tech University the three semesters are referred to as the Fall semester (September–December), Winter semester (January–April), and Summer semester (May–August). Since our application is designed to predict the success of the student based on a hypothetical next semester, we decided to divide the dataset by semester. The first step in this process was to assign a semester

number to each course. From the dataset, we have a term code that is associated with each course instance. The term code is made up of a year and a semester code, which is the month number in which the semester starts. For example, the term code for a course taken in the Fall semester in 2009 is 200909. We converted this date into an integer by ordering the term codes sequentially. At this point, we have all of the semesters numbered from 1 to 38. For every student, we convert this overall semester number into a relative semester number. The relative semester number starts at 1 and increases sequentially for each semester the student has attended the university. This semester number value is not used to train the model. It is used for querying and to group students for model evaluation.

To create the student vectors, the course history dataset including semester numbers was transformed from long format, where the course codes and grades have their own columns, to wide format, where the course codes are the column names and the grade is the value. The student vectors also include a high school admission GPA column and a *persister* column. According to a linear correlation analysis performed on the dataset, admission GPA has a high linear correlation to retention. The persister column is a binary column that indicates whether or not the student in the dataset withdrew from the university, 0 if the student withdrew, 1 otherwise. This column is only used for training the predictive model.

In our dataset, the samples where the persister column indicated success tended to have more semesters of data. We wanted to make sure that the model wouldn't connect taking more courses with being successful. While it is true that upper year students are less likely to withdraw, it is also true that successful upper year students were once successful first and second year students. We wanted to include as many samples of success as possible in the dataset used to train the model. To achieve this we decided to take each student and remove semesters one-by-one to create snapshots of each student after each semester. For example, a student that graduated after eight semesters will have one row with semester 1 grades, one row with semester 1 and semester 2 grades, one row with semester 1 and semester 2 and semester 3 grades, etc. The value of the persister column remains the same across a single student. The resulting vectors for a single student are shown in Table 4.3. Using this approach, we can increase the number of samples with

Algorithm	Accuracy	Precision	Recall
Logistic Regression	81.6%	89.6%	88.1%
Random Forest	89.8%	91.7%	96.6%
MLP Neural Network	84%	84.6%	98.8%
Naïve Bayes	16%	16%	100%

TABLE 4.4: Performance metrics of each of our tested algorithms: logistic regression, random forest, MLP neural network, and Naïve Bayes. Overall, the random forest performed the best with an accuracy of 89.8%, and precision of 91.7%.

fewer semesters of data with a label of success.

4.2 Algorithm Selection

We tested four different predictive models with our data early in the project. The goal was to see which machine learning model would give us the best results given the dataset. The algorithms we tested were Logistic Regression, Random Forest, Multilayer Perceptron (MLP) Neural Network, and Naïve Bayes Classification. To test these algorithms, we used a visual analytics platform called KNIME. Using KNIME allowed us to rapidly compare algorithm performance and have as much control over the data consistency as possible. As shown in Table 4.4, most of the algorithms performed reasonably well with the exception of Naïve Bayes, and random forest performing the best.

4.2.1 Performance Metrics

As part of our model evaluation, we generated a confusion matrix. The confusion matrix shows the number of correctly and incorrectly classified samples labelled as ”True Positive”, ”False Positive”, ”True Negative”, and ”False Negative”. In this case, our data had a label of 1 (Retained) and 0 (Withdraw). A true positive is a sample with an original label of 1, and a predicted label of 1. A true negative is a sample with an original label of 0, and a predicted label of 0. A false positive and false negative is a sample labelled as 0 and predicted as 1, or labelled as 1 and predicted as 0 respectively.

Semester Model	Training Accuracy	Testing Accuracy	Precision	Recall
Semester 1	70.2%	82.9%	92.1%	97.4%
Semester 2	77.3%	84.9%	94.2%	92.4%
Semester 3	80.8%	91.2%	97.4%	96.8%
Semester 4	82.7%	94.3%	98.3%	98.9%
Semester 5	86.6%	96.8%	99.2%	99.9%
Semester 6	89.5%	98.0%	99.6%	99.9%
Semester 7	92.3%	98.4%	99.6%	100%
Semester 8	94.1%	98.8%	99.7%	100%

TABLE 4.5: Performance metrics of predictive models trained on individual semesters of data. Each model was trained using data from 2007–2011, and tested using data from after 2011. Training Accuracy was calculated using 10-fold cross validation. Testing accuracy was calculated by scoring the trained model with our testing data. Precision and recall scores were calculated using testing data.

In terms of the student data, a student labelled as 0 and predicted as 1 (false positive) is our worst case scenario and should be minimized. This would reduce the number of at-risk students being shown results that may lead them to believe they are not at-risk. A student that was predicted to withdraw (0), with a real label of 1 is not ideal in terms of model accuracy, but preferable to telling a student that they will be successful when they will likely voluntarily or involuntarily withdraw. Precision is a metric that directly relates to the false positive rate, $Precision = TruePositive/(TruePositive + FalsePositive)$. A high precision score correlates to a low false positive rate.

Naïve Bayes did not perform well on our dataset. The model predicted all samples as "0", or withdraw. Perhaps with some parameter tuning, this model could have better results. Since the other models performed much better, we did not investigate this further.

Logistic regression is a common technique used to predict attrition using student data. We wanted to see if logistic regression could hold up to more modern techniques. It performed reasonably well with an accuracy of 82%.This number becomes less impressive when you consider that the overall attrition rate of the dataset is 14%, meaning if the classifier predicted every student to persist (1) it would have an accuracy of 86%. The precision score

Testing Data	Accuracy	Precision	Recall
Overall	85%	85.7%	98.9%
Semester 1	77.8%	89.1%	93.4%
Semester 2	81.3%	91.6%	95.7%
Semester 3	86.9%	96.5%	97.8%
Semester 4	90.1%	98%	99%
Semester 5	94.1%	99.1%	99.7%
Semester 6	96.2%	99.5%	99.8%
Semester 7	97.2%	99.7%	99.9%
Semester 8	98%	99.8%	100%

TABLE 4.6: Metrics of the overall model tested on individual semesters. Overall accuracy was calculated using 10-fold cross validation. Accuracy, precision, and recall for each semester were calculated using testing data.

of 90% improves this model slightly since the false positive rate remains low.

The MLP neural network performed reasonably well with our dataset. The accuracy was 84%, with a precision of 85%. The accuracy slightly better than the logistic regression model but the precision score suffered.

Random forests are known as a good "general purpose" machine learning algorithm, and for their performance on relatively small datasets with a high number of features. The random forest model performed the best with an accuracy of 90%, and precision of 92%. With both of these values being the highest among the tested algorithms, we decided to use a random forest classifier as our predictive model. Facing model portability issues with KNIME, we went on to implement the random forest model in Python using the SciKit-Learn library.

Choosing a Training Set

Based on the literature, previous student prediction models have been more successful when trained on the last 5 years of academic data as opposed to the entire set of historical data [16]. This can be explained by simply change over time. Instructors, course content, schedules, and students all change over time. These changes lead to shifts in grade averages and distributions,

Semester Model	Percent Withdraw	Training Samples	Testing Samples
Semester 1	27.1%	8770	7124
Semester 2	24.5%	8454	6796
Semester 3	17.5%	7590	5860
Semester 4	14.0%	7174	5221
Semester 5	11.1%	6745	3804
Semester 6	8.9%	6462	3013
Semester 7	7.1%	6158	2130
Semester 8	5.8%	5908	1424

TABLE 4.7: The class split of the data used to train each semester model, and the number of samples used to train and test each model.

and administrative changes like new courses, and changing course codes. Limiting the training data to the last 5 years minimizes the amount of change in the data, while maintaining a sufficient number of samples.

We were curious about model performance given different subsets of the data. Furthermore, we wanted to see how our model was impacted by using a dimensionality reduction technique called Principal Component Analysis (PCA). Dimensionality reduction came up simply because of the number of features in our training data. Working with the full feature set was cumbersome to work with in terms of storage and data manipulation. While manageable, we wanted to explore our options. PCA is a well known feature projection method that aims to project the original features into a lower dimensional space with the most variance between components. The goal is to maintain model accuracy and precision with fewer features.

We considered training a few different predictive models to use in our system. Because of previous work, we know that predictive models perform well when trained on academic data of students in similar groups [1]. We felt that grouping the students by faculty would open a door to exploring grouping students by program, and focusing on training the most accurate classifier. With respect to the project goal being to train an accurate, broad-spectrum classifier to predict success in the upcoming term, we opted to group the dataset by semester number.

The amount of data each student has varies throughout the dataset. It would make sense that students with 1 semester of data have less information than a student with 8 semesters who has taken more courses. However, because we separated each student into their cumulative semester rows, a student with 8 semesters also has a row with 7 semesters of information, 6 semesters, and so on. This means there were more samples used to train the semester 1 model compared to the semester 7 or 8 models. The number of training and testing samples, and the class split of the data are reported in Table 4.7. Each semester model was trained using 10-fold cross validation and the accuracy, recall, and precision are reported in Table 4.5. We initially thought that since the number of students that withdraw after semester 1 or 2 is significantly higher than the number of students that withdraw after semester 6 or 7, that the accuracy of the higher semester models would be much higher. As shown in Table 4.5, we were correct in our assumptions. We wanted to know if the overall model would show a similar trend in accuracy when tested on semester groups. The accuracy, precision and recall of these tests can be found in Table 4.6. As you can see, not much accuracy was lost when compared to a model trained specifically on that semester group.

While PCA reduced the size of our dataset, and accuracy remained consistent, there were significant problems with using PCA. First of all, dimensionality reduction removed a level of explainability of our model that we rely on in our system. With reduced dimensions, it is impossible to tell which individual features (courses) contributed to the trained model. The features with the highest model importance rating can be interpreted as the courses most predictive of success. We also lost a level of precision. Even with a small reduction in the number of features (i.e. 1501 features to 1450 features) the model simply predicted the class split. In other words, it predicted everyone to be successful, ensuring it would be correct 85% of the time. 85% accuracy is good, but only when the classifier is making an informed decision which is demonstrated by the precision and recall scores.

We did more tests combining the strategies of training on the most recent 5 years of data, dimensionality reduction, and grouping by semester. All of the tests using dimensionality reduction yielded the same results and precision issues as previous tests using PCA. Furthermore, grouping by semester once again proved unnecessary. The only grouping that proved effective

Parameter	Description	Possible Value(s)	Default Value
n_estimators	Number of trees in the forest	1+	10
max_leaf_nodes	Set number of leaf nodes for leaf-wise tree growth	1+, None	None
max_depth	Maximum depth of the tree	1+, None	None
min_samples_split	Minimum number of samples required to split an internal node	0+, None	2
min_samples_leaf	Minimum number of samples required to be a leaf node	0+, None	1
oob_score	Whether to use out-of-bag samples to estimate the generalization accuracy	True/False	False
criterion	Function to measure the quality of a split	Gini impurity (gini), Information gain (entropy)	gini
max_features	Number of features to consider when looking for the best split	1+, sqrt, log2	sqrt
class_weight	Weights associated with each class	balanced, None, custom	None

TABLE 4.8: Adjustable parameters for SciKit-Learn's `RandomForestClassfier`. All of the parameters are optional with the exception of n_estimators.

was training on 5 consecutive years of data, and testing on the following 2 years. Even though this method presents some challenges with implementation such as the need to retrain every year with a new dataset, it was this iteration that produced the best results.

4.3 Parameter Tuning

We used the default parameter values for initial model comparison. However some of the parameters or settings of the random forest model can be adjusted to affect the accuracy of the trained model. Table 4.8 shows the possible values of those parameters. Tweaking our model to squeeze out every possible point of accuracy was not the goal of these tests. The default values for the model parameters yielded acceptable results and met the needs of our system. That being said, we wanted to see how much of a difference a change in certain parameter values would make, and see if there was a much better model we could be using.

In order to select the best parameters for our data, we wrote a python script to test different combinations of different values for each parameter. The range of values used to test each parameter is outlined in Table 4.9. This

Parameter	Values/Range	Increment	Final Value
n_estimators	50–1000	50	50
criterion	gini/entropy	N/A	gini
max_leaf_nodes	10–960	50	860
max_depth	3–48	5	33
min_samples_split	2–10	1	9
min_samples_leaf	2–10	1	9
max_features	sqrt/None	N/A	sqrt
oob_score	0/1	N/A	0
class_weight	balanced/None	N/A	None

TABLE 4.9: Tested and final values for our predictive model parameters. We decided on the final values based on the model with the highest testing accuracy among those with the highest training accuracy.

Feature	Description	Rank
SOCI1000U	Introductory Sociology	1
BUSI2000U	Collaborative Leadership	2
PSYC1000U	Introductory Psychology	3
MATH2860U	Differential Equations for Engineers	4
EDUC1050U	Technical Communications	5
BUSI1450U	Statistics	6
HLSC2460U	Pathophysiology I	7
STAT2800U	Statistics and Probability for Engineers	8
adm_gpa	Admission GPA	9
BIOL1020U	Biology II	10

TABLE 4.10: Top ten most important features as ranked by our predictive model. Among the top courses are popular electives (Sociology and Psychology), and courses taken by most engineering and science students (Differential Equations, Technical Communications and Statistics). We knew that Admission GPA was correlated with retention, so it is not surprising to see that it is ranked number 9.

combination of parameter values resulted in over 10000 possible models. For the sake of time, we randomly selected 1000 models to train. We deployed these tests on a Shared Hierarchical Academic Research Computing Network (SHARCNET) cluster. The model was trained using data from 2008–2012 and tested using data from 2013 and 2014. These tests returned the training accuracy using 10-fold cross validation and testing accuracy of each model. We selected the parameter values based on the model with the highest training and testing accuracy.

It is important to note that these models were trained on an early version of our dataset. We redefined our training and testing sets by admission year, and 11 of the 1510 training features used to test the parameter values were removed after the feature selection process. These features represented demographic, and first-year summary information.

4.4 Feature Selection

The purpose of the final application is to predict a student's success given a set of future courses. We knew that we needed to include courses in the feature set, however there was more than one way to achieve this. The first 3 or 4 characters of an Ontario Tech course code denote the department that offers the course. We considered grouping courses by this department code to decrease the overall number of features. However, we would lose an important level of information. If we group by department code, the feature value could be the student's average grade in those courses, or the number of courses taken in that department, or some other computed value. This means we lose information at the individual grade level, and even information like the year level of the course, or whether or not it is an elective course.

Another option was to group by department code and year level. In Ontario Tech course codes, the year level of a course is denoted by the first digit following the department code. Grouping courses by department code and year level gives us an extra level of information when compared to just grouping by department code, but we still need to use some sort of average or summary as the value. Since we were interested in course performance and we want the model to be sensitive to different grade values, we decided not to group the courses and use each individual course as a feature.

We decided to include one additional feature with the courses, admission GPA. Admission GPA is the average GPA of the student's grade 12 courses used for admission to the university. Based on a linear correlation analysis performed on the `retention_raw` dataset (Table 4.1), we know that along with first semester GPA and first year GPA, admission GPA is correlated to retention. We decided not to include first semester GPA and first year GPA as features because those values can be calculated using the existing features, making them redundant.

The trained random forest model assigns as level of importance to each of the features used in the training set. The top ten features are shown in Table 4.10.

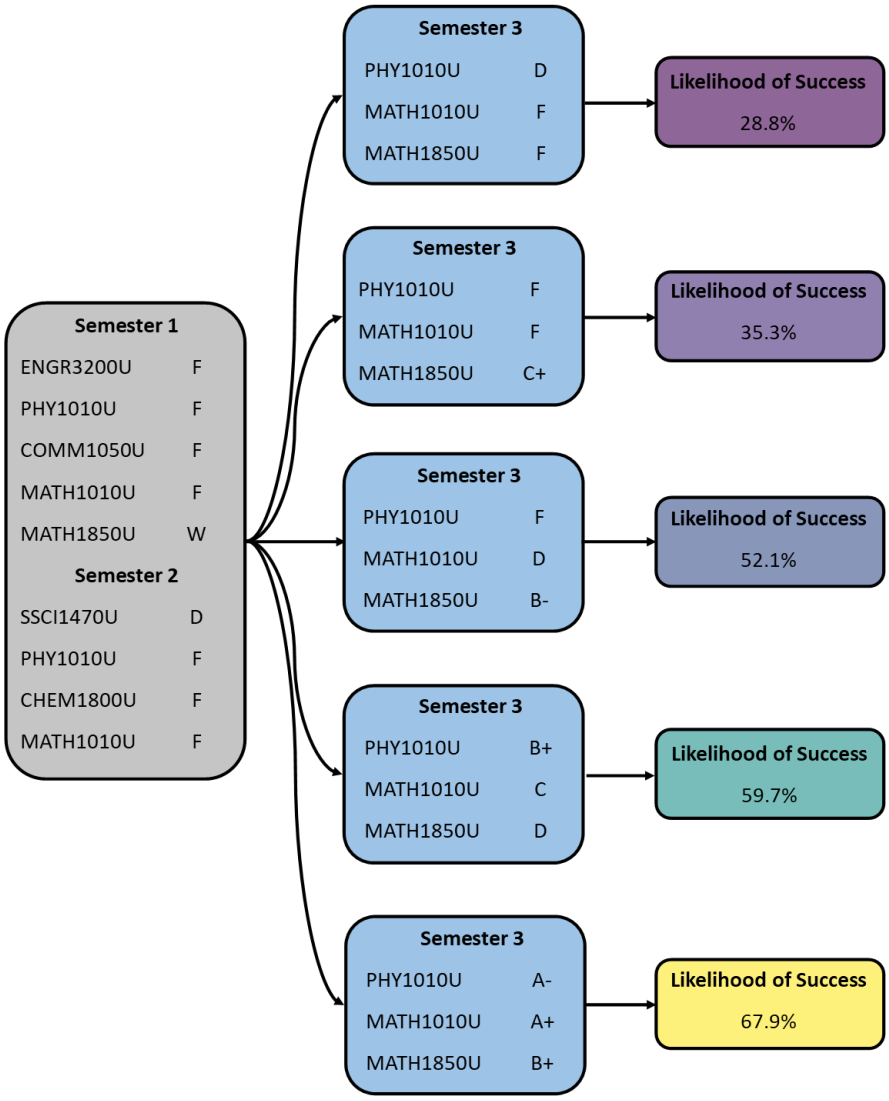

FIGURE 4.1: Each student vector is made up of the student's existing academic information, and a generated scenario. In this case, the student has two semesters of data. They chose three courses to take in their third semester: INFR1310U (Graphic Design I), CSCI1200U (Computers and Media), and CSCI2010U (Principles of Computer Science). Sets of likely grades were selected and assigned to the three courses using the algorithm described in Section 4.5.1 to create five hypothetical third semesters. These third semesters are combined with the semester 1 and semester 2 information to create five student vectors, which when fed into the model return a likelihood of success value.

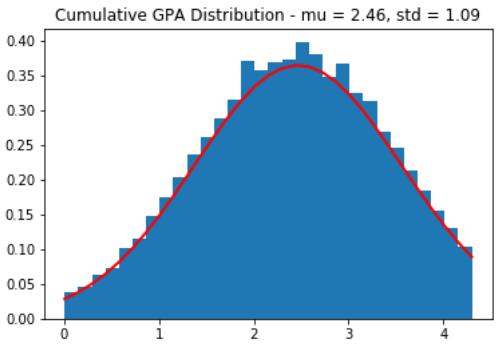

FIGURE 4.2: Distribution of semester cumulative GPAs.

4.5 Generating Student Vectors

The *student vector* that will get passed through the predictive model contains all of a student's course and grade information since admission, and a hypothetical next semester as shown in Figure 4.1. The hypothetical next semester is made up of a set of 1–7 courses chosen by the student, and a set of generated grades that will be assigned to the chosen courses. For example, if a student wanted to take 5 courses, the system would generate a set of 5 grades that the student is likely to achieve and assign each grade to one of the 5 chosen courses. This process of generating and assigning grades is repeated a number of times. A set of courses with assigned grades is called a *scenario*. A student interested in taking 5 courses would generate 90,000 possible scenarios assuming 5 different courses, and 10 possible grades. This would be considered the "brute-force" approach. There are two notable problems with the brute-force approach:

- Too much information
- Irrelevant scenarios

Displaying all possible scenarios has a high potential to overwhelm the user and cause the chart to become over-saturated. Over-saturating the chart could stop the user from interpreting the chart correctly as the different colours become less apparent, and the visual space gets distorted. One possible solution to this is to random sample from the full set. The problem with random sampling is that the sample wouldn't be tailored to individual student

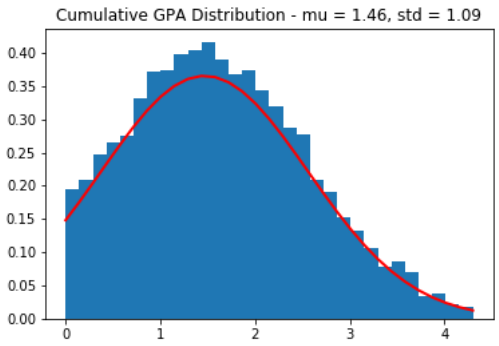

FIGURE 4.3: Sampling distribution for a student with a cumulative GPA of 1.46.

performance. A student with a cumulative GPA of 4.0 would get a similar sample as a student with a cumulative GPA of 1.0, introducing irrelevant scenarios.

Displaying irrelevant scenarios has the possibility of misleading users by displaying scenarios that are unlikely for the individual student to achieve. In order to mitigate these effects, we needed to implement a sampling technique that selected scenarios that were representative of the student's past academic performance, and covered the likely grade spectrum. The sampling technique involves a few steps which are described in Section 4.5.1:

1. Set up a normal distribution across all cumulative GPAs in the dataset and store the standard deviation
2. Calculate the cumulative GPA of the student
3. Set up a normal distribution using the standard deviation calculated in Step 1, and the mean calculated in Step 2
4. Sample a cumulative GPA from the distribution from Step 3
5. Select a scenario from the scenario probabilities table using the sampled GPA
6. Assign each grade in the scenario to each of the proposed courses using the course/grade probabilities table
7. Repeat Steps 4–6 for the desired number of samples

ALGORITHM 4.1: Assigning grades to courses. This algorithm assigns grades to courses by taking into account the probability of receiving each grade in each course. It takes an array of grades (G), an array of courses (C), and a probability table (P) as input, and returns an object of grades assigned to courses. This algorithm is repeated for every scenario generated in the previous step.

```
input:  G = [g_1, g_2, ... , g_i],
        C = [c_1,c_2, ... , c_j],
        P = [p(g_1,c_1), p(g_1,c_2), ... , p(g_i,c_j)]
output: Assigned grades object
begin
    foreach g_i in G
        S ← sum of probabilities of receiving g_i
        rnd ← random number between 0 and S
        foreach c_j in C
            check ← rnd − p(g_i,c_j)
            rnd ← check
            if check ≤ 0
                assign grade g_i to course c_j
                remove g_i from G
                remove all c_j probabilities from P
    return assigned grades
end
```

4.5.1 Generating Scenarios

The first step to generate the scenarios was to set up a normal distribution across all semester cumulative GPAs in the dataset. We fit the curve to our data and calculated the standard deviation and mean of the distribution (Figure 4.2). This standard deviation value will be used as the standard deviation for our sampling distribution. Next we calculate the cumulative GPA of the student in question. This cumulative GPA value will be used as the mean (`mu`) for our sampling distribution.

At this point, we can set up our sampling distribution using the standard deviation of semester GPAs and student cumulative GPA (mu) calculated earlier. This approach allows us to set up a sampling distribution that is generalizable, and adjusts to the performance of an individual student. A sample distribution is shown in Figure 4.3. From this distribution, we sample 400 cumulative GPA values that will be used to select scenarios from our *scenario probabilities table*. The scenario probabilities table is described later in this section.

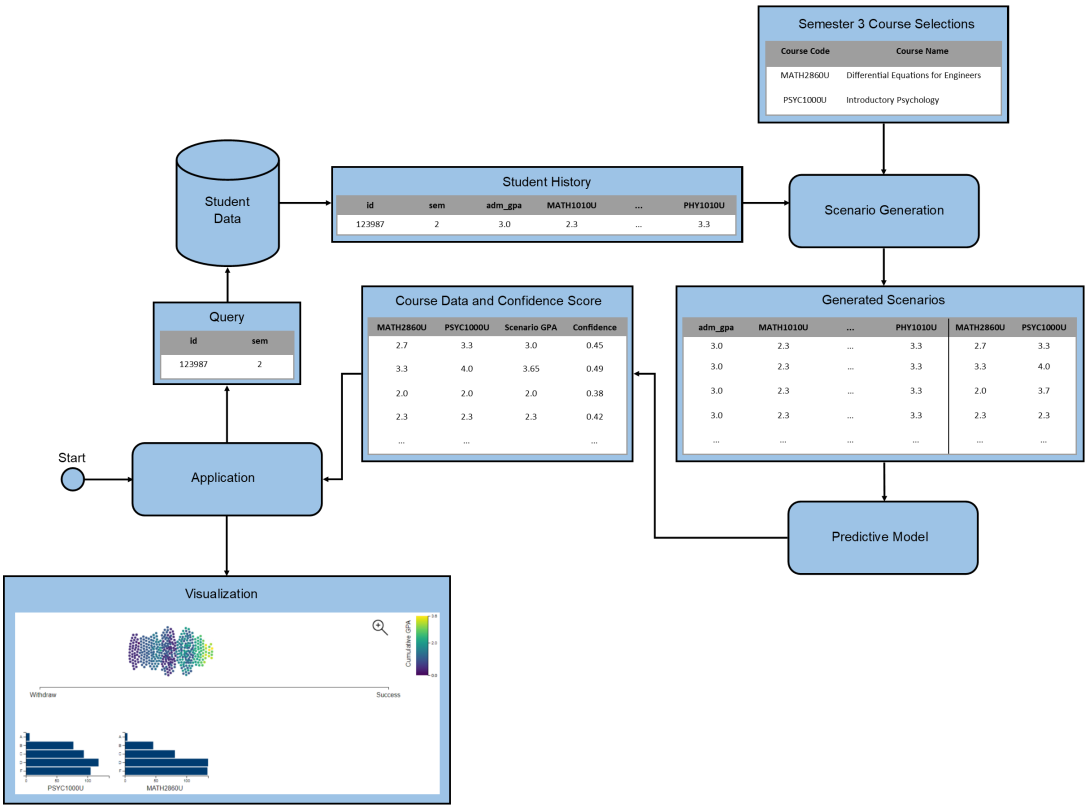

FIGURE 4.4: This flowchart describes our system from start to finish. We start by using our application to query the database for the course history of a single student. The result of this query (Student History) is passed to the Scenario Generation step along with the course selections for the following semester. The Generated Scenarios are combined with the Student History to represent the completion of the following semester and are fed into the Predictive Model. The Predictive Model returns the Confidence Score and Generated Scenarios which are passed to the Application to populate the Visualization.

Assigning Grades

The final step of the sampling technique is assigning the grades in each selected grade scenario to one of the courses chosen by the student. To decide which grades are assigned to which course, we need three pieces of information; the grade scenario (G), list of chosen courses (C), and the conditional probability of getting each grade in each course (P). This process is described in Algorithm 4.1.

Once we have a sample of scenarios, we can generate student vectors to feed into the predictive model. Figure 4.4 describes how this sampling technique is implemented within the overall system.

Semester GPA	Scenario	Probability
0.00	[F, F, F, F]	1.000000
0.25	[D, F, F, F]	0.705426
0.25	[F, F, F, D]	0.108527
0.25	[F, D, F, F]	0.096899
0.25	[F, F, D, F]	0.089147
4.15	[A, A+, A, A+]	0.126506
4.15	[A+, A+, A-, A+]	0.126506
4.15	[A, A+, A+, A]	0.114458
4.15	[A+, A+, A, A]	0.108434

TABLE 4.11: A snapshot of the *scenario probabilities table* for four courses. This table contains 6020 rows. We generated a separate probability table for each number of courses taken in a semester ranging from 1–7.

Course Code	Final Grade	Probability
MATH1020U	D	0.227899
MATH1020U	F	0.214592
MATH1020U	C	0.167670
MATH1020U	B	0.070586
MATH1020U	B-	0.066941
MATH1020U	A-	0.066362
MATH1020U	C+	0.059246
MATH1020U	A+	0.052939
MATH1020U	A	0.040616
MATH1020U	B+	0.033152

TABLE 4.12: A snapshot of the *course/grade probabilities table*. Probabilities are shown for MATH1020U (Calculus II). The full *course/grade probabilities table* includes probability values for every course that has been offered at the university and contains 11733 rows.

Probability Tables

The scenarios used for the probabilities are sampled directly from the database. For every full-time student at the university, we isolated each semester and stored the grade combination (e.g. [A,A+,A+,B,B-] would be a combination for a 5-course semester). For each of these combinations, we calculated the semester GPA. With this information we were able to calculate the probability of a specific scenario given a semester GPA as shown in Table 4.11. Another method to generate sets of grades would be to generate individual probable grades for each of the proposed courses and combine them as a set. However, we know that semester grades are not mutually exclusive. This means that the probability of receiving a particular grade in a course is impacted by the other courses being taken and the grade received in those courses. For this reason, we chose to pull grade sets directly from the data.

The grades selected from the *scenario probabilities table* are assigned based on the probability of getting a certain grade in each course. We pulled all of the grades received in each course at the university from the database. For every course, we calculated the probability of getting each grade and stored the values in our *course probabilities table* as shown in Table 4.12.

4.5.2 Prediction Confidence Score

The prediction probability (confidence) score of each generated scenario can be interpreted as the likelihood or confidence that the prediction is true. This is the value that will ultimately be shown to users in our tool to be interpreted as a student's likelihood of success. A Random Forest classifier is a collection of decision trees that work together to generate a single prediction. Each tree in the forest uses different features to make a decision which is counted as a vote for the final prediction. For example, a classifier made up of five trees may return three 0 predictions, and two 1 predictions. Each prediction counts as a vote and the majority wins. In this case, the classifier would return an overall prediction of 0. Along with this overall prediction, the classifier will also return a value between 0 and 1 which is the likelihood that this prediction is true, what we refer to as the prediction confidence score.

Chapter 5

Application Design

Our proposed application aims to present the output from our predictive model to Academic Advisors to help students make decisions that will make them more likely to succeed. The system takes a student ID, and a list of courses as input and generates likely scenarios based on the student's past academic performance. These scenarios are fed into the predictive model as *student vectors* and the confidence level of each prediction is displayed on an exploratory analytics interface for interpretation by the user.

5.1 Iterative Design

Our iterative design process began with the initial design of the Retention Dashboard, described in Chapter 3. We met with a representative from the Office of Institutional Research and Analysis (OIRA) and the Registrar's Office (RO) three times during the design of the dashboard.

The initial meeting was a project brainstorming session, where after some discussion we chose to focus on student retention. Some of the ideas brought up during our discussion include coming up with user friendly way to visualize and interact with the student dataset, finding similar problems among student and providing advice according to those similarities, and generally finding a better way to provide human intervention. Moving forward with the idea of an interactive dashboard to assist human intervention, we went on to meet with the Dean of Science, program directors, and academic advising staff to help us understand the kind of data that could be most helpful in terms of analyzing retention. From these meetings, we learned that it was

important to include information that allow users to draw actionable conclusions from their analysis.

From here we designed the dashboard piece by piece. We started with the timetable visualization and the parallel coordinates chart. Scheduling, and grade trends were both brought up as actionable information in our previous meetings. Once we had working versions of both visualizations, we met with a representative from the OIRA and the RO again to discuss the current visualizations, and future steps. From here we included more filters on the parallel coordinates chart in the form of axis brushing and line hovering, and on the timetable as individual grade filters and column/row selectors. We added new filters to the dashboard including individual program selection, and the admission year bar chart selector. All of these components together allow the user to explore relationships between grade trends, scheduling, and attrition over time.

We met with OIRA and an academic advisor at the beginning of the thesis project to discuss moving away from the dashboard, and towards tool to assist academic advising. They provided positive feedback to the idea of representing student success, and an early prototype of the beeswarm plot populated with simulated data. They emphasized that it could help to convince students of a poor course path, or encourage students who are not as confident in their performance. From here we began the process of training and testing different models with different datasets, and creating the application interface.

5.2 Interface

We had some initial criteria for designing our interface based on our meetings with OIRA and academic advisors. Because student-advisor meetings only last approximately 15 minutes, simplicity and intuitive design were key. We needed a fast way for users to enter the student information and courses, and an intuitive way to display and interact with the information. We went with a single-page design with three components: input panel, visualization component, and a summary component. The design and selection of each component are described below.

FIGURE 5.1: An example of a beeswarm plot. Points are spread out along a horizontal axis, clustering in swarms when multiple points have the same value.

5.2.1 Visualization Selection

We wanted to represent the concept of 'likelihood of success'. The confidence level output from the predictive model represents the level of confidence that the model has, given the information in the student vector, that the student will not dropout. We decided that this was a good representation of likelihood of success. The confidence level is a decimal number between 0 and 1 ranging from less likely to succeed to more likely to succeed.

Choosing the correct visualization to display the confidence level presented a challenge. We needed to choose a visualization that was easy to read for visualization novices, could be part of a larger dashboard type of application, and allowed for some simple analysis. A beeswarm plot, shown in Figure 5.1, displays points distributed across a horizontal scaled axis, stretching the axis in places where more points need to be displayed. When many points have the same value, the points cluster together to form a swarm. The predicted confidence levels tend to cluster around similar values for a single student which is represented well by the beeswarm plot. We can also add filters and colour to the points being displayed on this type of chart which support analysis.

Scaling the Plot

The confidence level ranges for a single student can vary anywhere from 2 or 3 percentage points to more than 20 percentage points. Displaying the points on a 0 - 1 scale provides a level of information useful for students with a wide range of confidence levels, especially when the predicted confidence levels cross the 0.5 mark. A confidence level below 0.5 means that the model predicts that the student will likely dropout, where a confidence level above

(A) Beeswarm plot scaled from 0–1.

(B) Beeswarm plot scaled from minimum to maximum confidence.

FIGURE 5.2: The same beeswarm plot scaled (A) from 0–1 and labelled Withdraw–Success and (B) from minimum confidence–maximum confidence prediction.

0.5 means a student will likely be retained. Being able to see which scenarios cross this threshold will highlight the course-grade combinations more likely to lead to success. Instead of labelling the end points of the axis as 0 and 1 we decided to use labels of Withdraw and Success as shown in Figure 5.2a. This helps the user to interpret the axis as a spectrum of Withdraw to Success rather than focusing on the raw numbers.

Scaling the axis from the minimum confidence level and maximum confidence level allows for another level of information more useful to students whose plots look like the one shown in Figure 5.2a. Seeing all of the points spread out between their minimum and maximum value, as shown in Figure 5.2b, will help students find the courses to focus on to maximize their likelihood of success.

The previously mentioned scaling techniques complement each other but neither of them are sufficient on their own. We decided to allow the user to switch back and forth between the two scales. As shown in Figure 5.3, clicking the *zoom-in* icon in the top right corner of the chart will change the scale from Withdrawn–Success to Minimum Confidence–Maximum Confidence. Clicking the *zoom-out* icon on the zoomed-in view will switch the scale back to Withdrawn–Success.

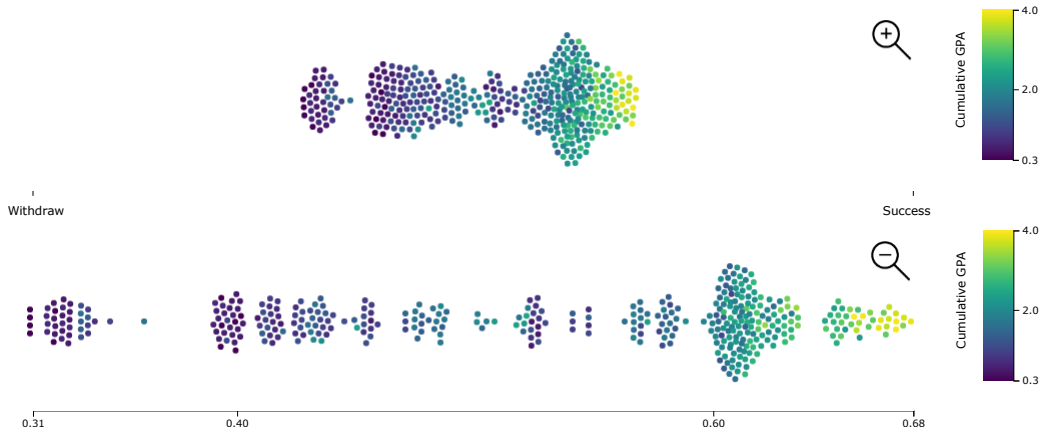

FIGURE 5.3: Top: Beeswarm visualization component displayed on a scale from Withdraw(0)–Success(1). Bottom: The same data displayed on a scale from minimum confidence–maximum confidence.

FIGURE 5.4: The Viridis colour scale. In our visualization the darker purple represents lower cumulative GPA values of the scenarios, while the lighter yellow represents higher cumulative GPA values of the scenarios.

Colouring the Points

As previously discussed, each point represents a set of courses and grades, called a scenario. The likelihood of success is encoded as the horizontal position of the point on the axis. The points are coloured based on the semester GPA of the scenario. The Viridis colour palette, Figure 5.4, is known for creating plots which are more accurately perceived, are accessible, and are visually appealing [9]. Using the colour scale, we can visually see where the scenarios with a higher semester GPA lie on the axis. The darkest colour on the colour scale represents the lowest semester GPA of the scenarios, and the lightest colour represents the highest semester GPA of the scenarios.

5.2.2 Interaction Design

The final application will be used by academic advisors as students come in for a meeting. Some of these meetings are booked in advance, but many are walk-ins. According to one of the academic advisors we interviewed, advisors often have approximately 15 minutes to meet with a student. Our

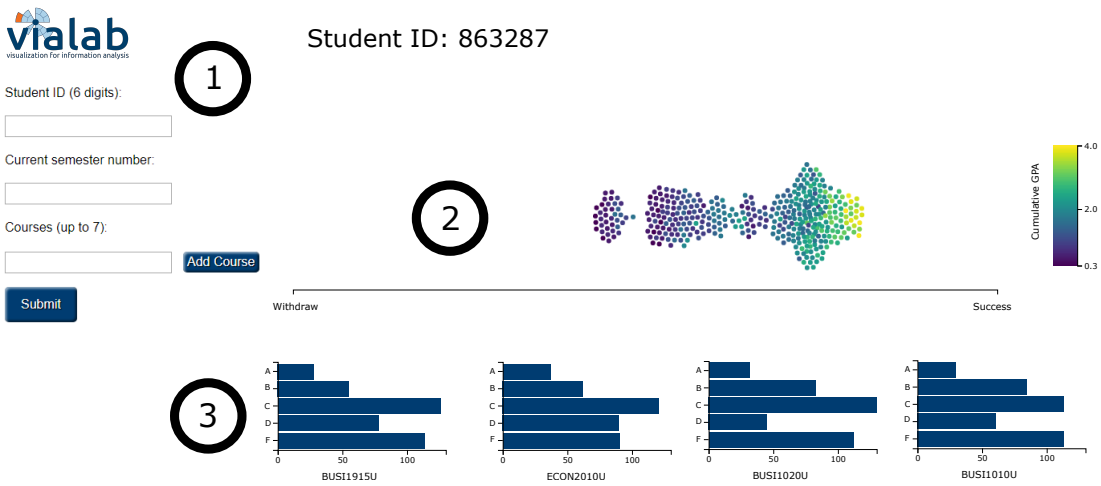

FIGURE 5.5: The application interface showing a Business student going into their third semester, and a cumulative GPA of 1.47. (1) Form input panel component where the user enters a student ID, current semester number, and up to 7 courses to take in the upcoming semester. (2) Beeswarm component that displays the prediction confidence level. This component includes a "zoom" button to toggle the axis scale between 0 and 1, and the minimum and maximum confidence level. (3) Summary bar chart component that shows the number of occurrences of each grade in each course of the selected portion of the chart.

application needs to allow them to get the system up and running as quickly as possible when a student comes in for a meeting. We decided on a simple, single-page design with classic input types including input text fields, and buttons. The interface is made up of 3 components, shown in Figure 5.5:

1. Form input
2. Beeswarm visualization
3. Summary bar charts

The form input component, component 1 in Figure 5.5, is made up of static text input fields for the student ID number, and the student's current semester number. The student ID is used to query the database for the current user, and the semester number is used to filter the query results after some calculations are performed on the full result set. Course selections are entered using dynamic text input fields. Depending on the number of courses the student wants to take in the following semester, the user can add that number of fields. Any of the added fields can also be removed. Clicking

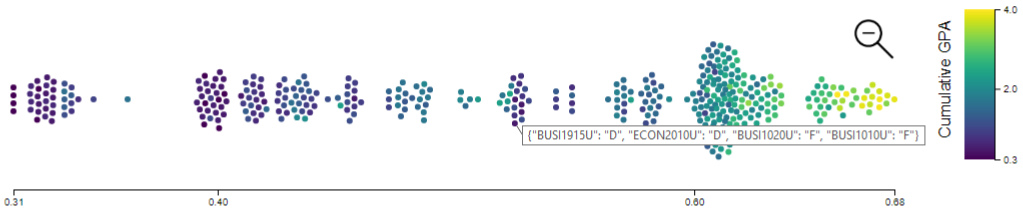

FIGURE 5.6: Beeswarm plot showing scenario details on hover. In this particular scenario, the student got a D in BUSI1915U, D in ECON2010U, F in BUSI1020U, and F in BUSI1010U.

the "Submit" button invokes the Common Gateway Interface (CGI) Python script which uses the information entered into the HTML form to generate student vectors and predict the likelihood of success.

The beeswarm visualization component is made up of a static beeswarm chart, colour scale, and dropdown menu to toggle the axis scale. The beeswarm plot is populated using the CSV file generated from the CGI script. As described in Section 5.2.1, the user can toggle the scale of the beeswarm plot by clicking on the *zoom* icon. Hovering the mouse over an individual point on the chart shows the course/grade combination that led to that confidence scores in a tooltip (Figure 5.6). Clicking and dragging horizontally (brushing) on the x-axis selects points on the chart that fall within the brushed range (Figure 5.7).

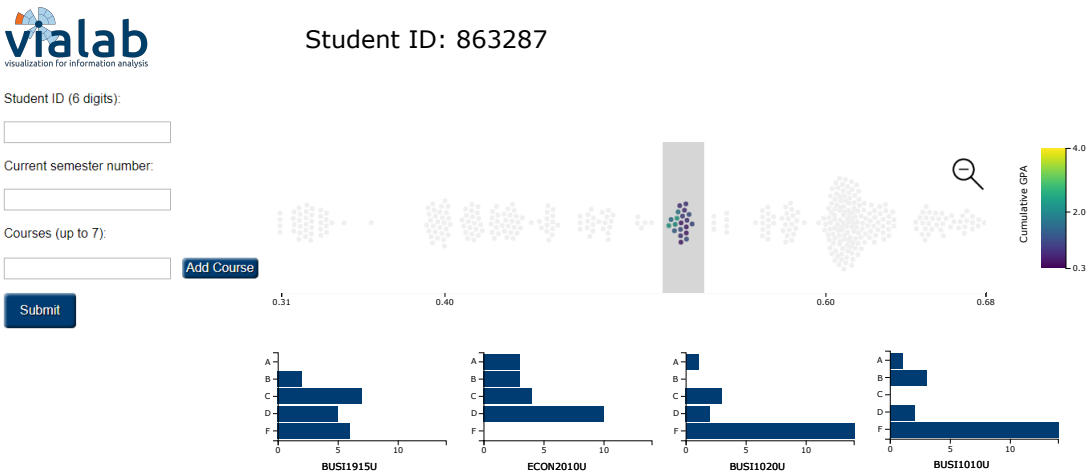

FIGURE 5.7: Brushing interaction used to select a set of scenarios on the beeswarm plot. From the bar charts, we can see that the selected scenarios contain a lot of F's in the BUSI courses but no F's in the ECON course.

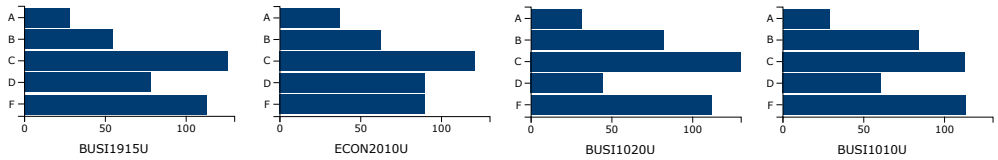

FIGURE 5.8: Bar charts showing the grade distribution of the scenarios for each proposed course. These bar charts update when the user selects a portion of the chart using the brush interaction.

The summary bar chart component, Figure 5.8, attempts to summarize the course and grade information embedded in the beeswarm chart. A horizontal bar chart is drawn for each course that the student selected. When the page loads, the bar charts show the number of occurrences of each letter grade in each course for all scenarios. The "brushing" interaction described earlier updates these bar charts to show the number of occurrences of each letter grade in each course within the brushed range. The goal of these summary charts was to allow users to not only see the likelihood of success given a set of courses, but also to see which courses are more indicative of success. For example, let's say a student would like to see their likelihood of success if they take Physics, Calculus, Psychology, and Programming. When this information is fed into the system, the predicted confidence scores range from 0.4 to 0.7. When the user brushes over the 0.6–0.7 range, the bar charts update to show that there is a high occurrence of A's and B's for Physics, Psychology and Programming within this range, and a high occurrence of D's for Calculus. One possible takeaway is that the student in question needs to do well (A's or B's) in Physics, Psychology and Programming, but doesn't need to do as well in Calculus to be successful. This type of information could be useful when thinking about which courses to take at the same time, which courses may require more time and effort, and future scheduling.

5.3 Backend

The web application uses CGI to execute a Python script which generates the data to populate the beeswarm chart. The script writes the student id, confidence level, cumulative GPA, and courses to a CSV file. Finally, the CGI script returns HTML code that tells the browser to refresh the current

page, which loads the updated CSV file. Files are served using Python CGI-HTTPServer.

Chapter 6

Conclusion and Discussion

We were able to train a machine learning model to predict whether or not a student will dropout after the given semester with 89% accuracy. Our proposed application uses a robust sampling algorithm to interactively display the prediction results to the user. By using our likelihood of success measure, we can convey to students the probable outcome of their next semester given their selected courses, and probable grades.

In the following sections, we will discuss the potential use of our system through fictional case studies, the limitations of our system and possible future work.

6.1 Discussion

In this section, we are going to present some fictional case studies that demonstrate the applicability of our system in different use case scenarios, and the interpretation of results in these different cases.

6.1.1 Case Study 1

Holly is a software engineering student who was admitted in 2011 and just finished her first year. Academically, she performed well in most of her five first semester courses, finishing with a cumulative GPA of 2.88. She struggled in both Calculus I, and Physics I but was optimistic knowing that these were difficult courses and many of her peers were also struggling. Holly was also engaged in the campus community as a member of the Varsity Curling team, and a member of the Board Games Club. Unfortunately, semester 2 took its toll on Holly and her semester GPA fell to 1.67, leaving her with an

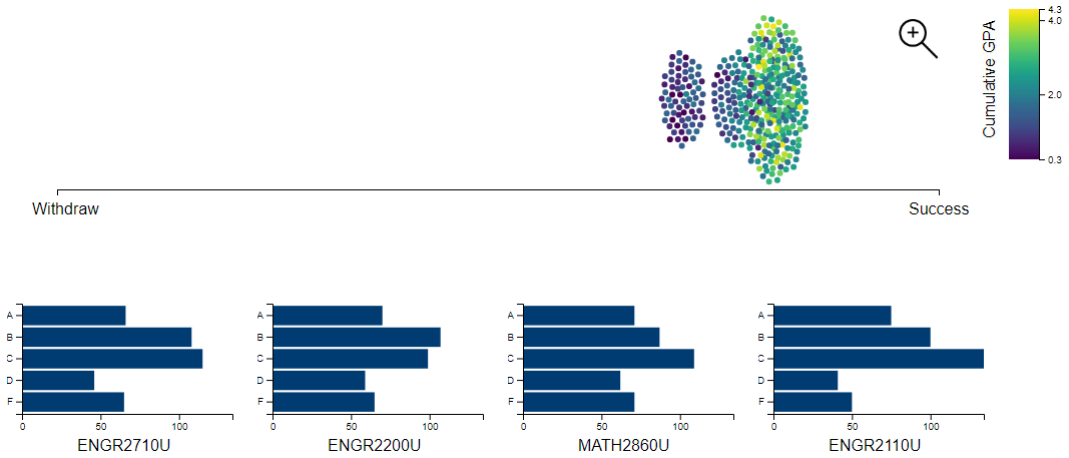

FIGURE 6.1: Beeswarm plot showing the likelihood of success for Holly given the four required courses. All of the scenarios lie above the mid-point indicating an overall high likelihood of success.

overall cumulative GPA of 2.27. While she did not fail any courses, she was concerned about her falling GPA and possibly ending up on academic probation. Holly was able to identify 3 main concerns about moving forward and her poor academic performance:

1. Course load increase from 5 to 6 courses
2. Increased demand from the curling team during competition season
3. Lack of interest in course material

Before registering for second year courses, Holly decided to book an appointment with an Academic Advisor to discuss her future. The Software Engineering program map lists 5 courses to be taken in the first semester of year 2; Discrete Mathematics (ENGR2110U), Electrical Engineering Fundamentals (ENGR2200U), Object Oriented Programming (ENGR2710U), Differential Equations (MATH2860U), and a liberal studies elective. Due to her poor performance in previous math courses, Holly is particularly concerned about taking Differential Equations, and Electrical Engineering Fundamentals.

During the first 5 minutes of Holly's meeting with her advisor, they discuss her reason for coming in, and she brings up her concerns listed above. During this discussion, the advisor can enter the required courses into our

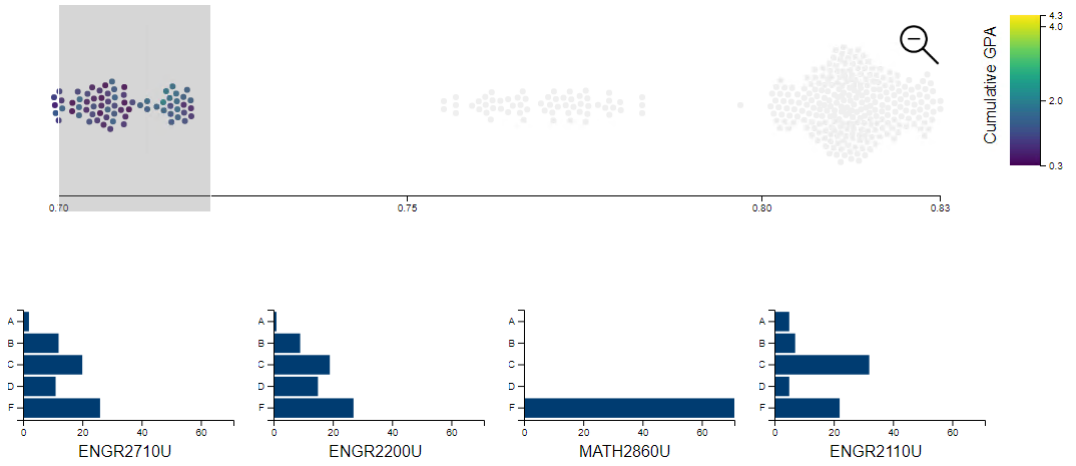

FIGURE 6.2: Lower likelihood scenarios selected on the zoomed in beeswarm plot. The grade distributions are similar for the three engineering courses, but the math course indicates that receiving an F results in a lower likelihood of success.

Student Success Prediction System. Holly is still unsure about which elective to take, so the advisor simply enters the 4 required courses. The resulting plot is shown in Figure 6.1. As you can see, all of the scenario predictions lie to the right of the mid-point meaning that in all generated cases, Holly is likely to succeed.

Drawing this conclusion without further exploration could be misleading. These results mean one thing: that past students with similar academic history have been successful. This leaves a lot of room for interpretation by the advisor, and it is important for the advisor to communicate that these results do not mean guaranteed success.

To further analyze these results, the advisor selects the scenarios with a lower likelihood of success (Figure 6.2). The summary bar charts update to show which grades are present for each course in the selected scenarios. Right away we can see the high frequency of F's in MATH2860U. Sliding the selection window to the far right of the chart, Figure 6.3, we can see that the most successful predictions have a minimum grade of C in MATH2860U. Given this information, the advisor would communicate to Holly that she needs to focus on doing well in MATH2860U, and a grade below a C in this course could lower her likelihood of success.

This visual analysis could allow for further discussion about reducing

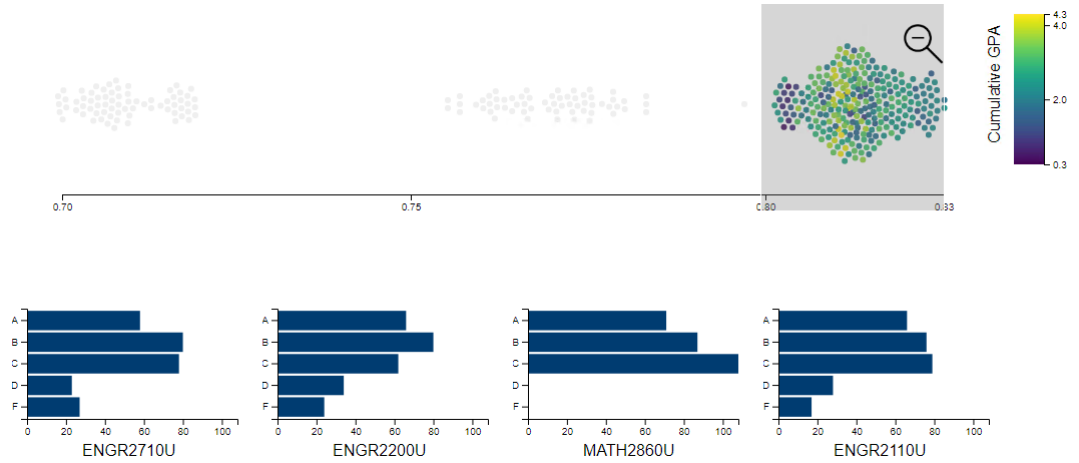

FIGURE 6.3: Higher likelihood scenarios selected on the zoomed in beeswarm plot. The grade distributions are similar for the three engineering courses, and the math course indicates that receiving a C grade or higher leads to a higher likelihood of success.

course load to spend more time on critical courses, and extracurricular activities. After her meeting with the advisor, Holly feels more confident about her academic future and time management in the upcoming semester.

6.1.2 Case Study 2

Travis is a business student who was admitted in 2012 and is going into his second year. Travis grew up in a community approximately 20 minutes from the university and decided to live at home to save some money. He has a group of peers that he socializes with while on campus, but is expected to be home in the evenings to contribute to the family home. He finished his first semester of first year with a GPA of 0.5. This placed Travis on academic probation, meaning that as long as his overall cumulative GPA remains below 2.0, he must achieve a semester GPA of 2.0 or risk being suspended. The added stress of avoiding academic suspension motivated Travis in the short-term, but saw him withdraw from 4 out of 5 courses later in the semester. He achieved a C in his remaining semester 2 course (2.0 semester GPA), which was enough to avoid suspension but not enough to be taken off academic probation (0.75 overall GPA). At this point, Travis decides to book a meeting with an Academic Advisor to discuss his options.

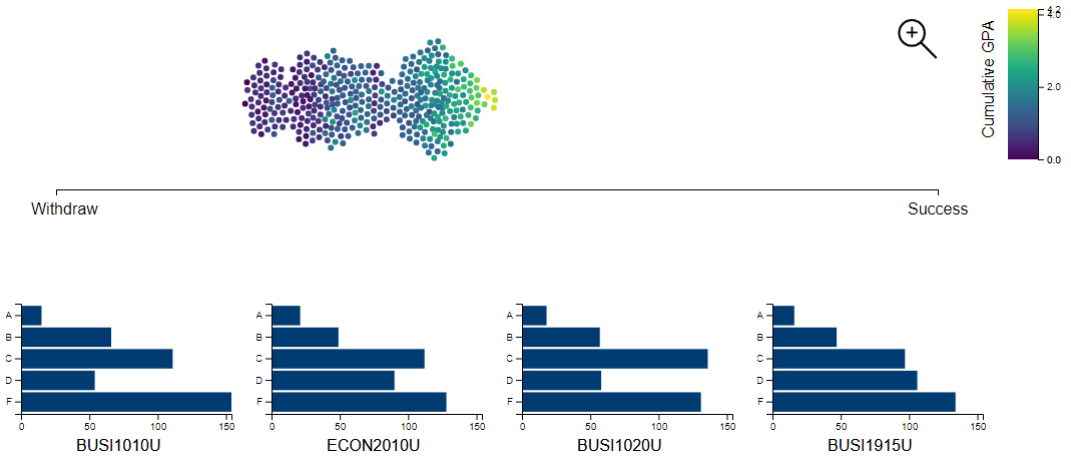

FIGURE 6.4: Beeswarm plot showing Travis' likelihood of success with his four chosen courses. The scenarios lie near the middle of the chart and cross the mid-point.

At the beginning of the meeting, Travis and the Advisor come up with a set of courses to take using the relevant program map, and previously failed courses. Because of his history of withdrawing from courses before the end of the semester, they decide to look at a reduced course load of 4 courses. The courses Travis needs to take are Business Communications (BUSI1020U), Microeconomics (ECON2010U), Business Math I (BUSI1915U), and Critical Thinking and Ethics (BUSI1010U). The Advisor generates the plot and begins their cursory analysis.

As you can see in Figure 6.4, the predictions fall above and below the mid-point of the plot. This means that the predictor is not highly confident that Travis will succeed, or that he will withdraw. At this point, the Advisor should point out that the lighter coloured scenarios (which represent higher grades in a scenario) lie to the right of the mid-point, while the darker colour scenarios lie to the left. They should communicate that unless Travis performs well academically, he is likely to withdraw. This information should spark further discussion about Travis' priorities, and changes he could make to improve his grades. For example, they could discuss moving into campus housing, further reducing his course load, or switching to part-time status.

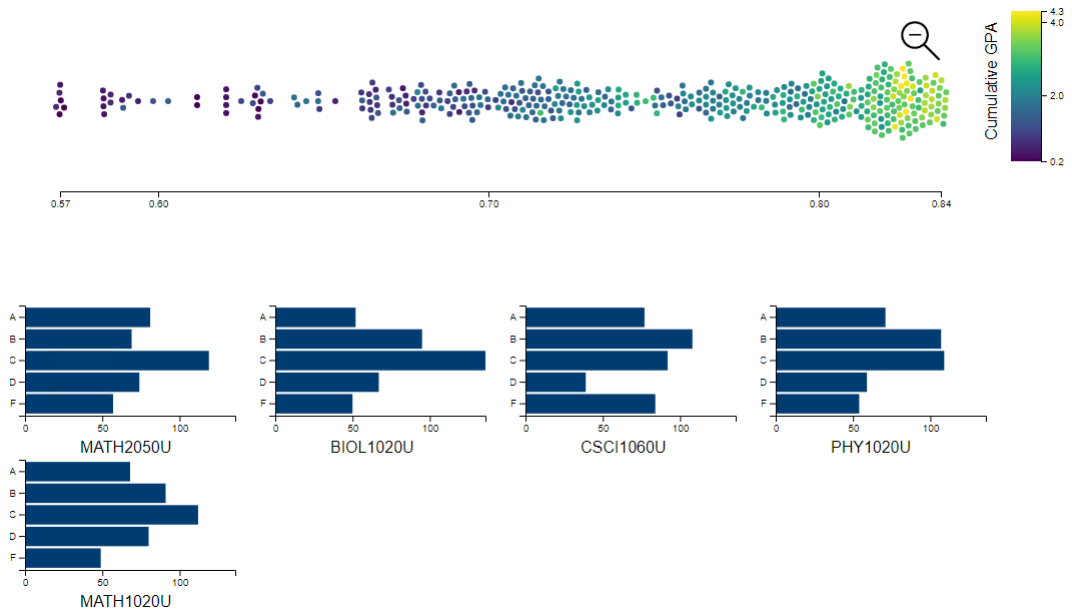

FIGURE 6.5: Zoomed in beeswarm plot showing Abby's likelihood of success if she takes Biology as an elective. All of the confidence levels lie between 0.59 and 0.85.

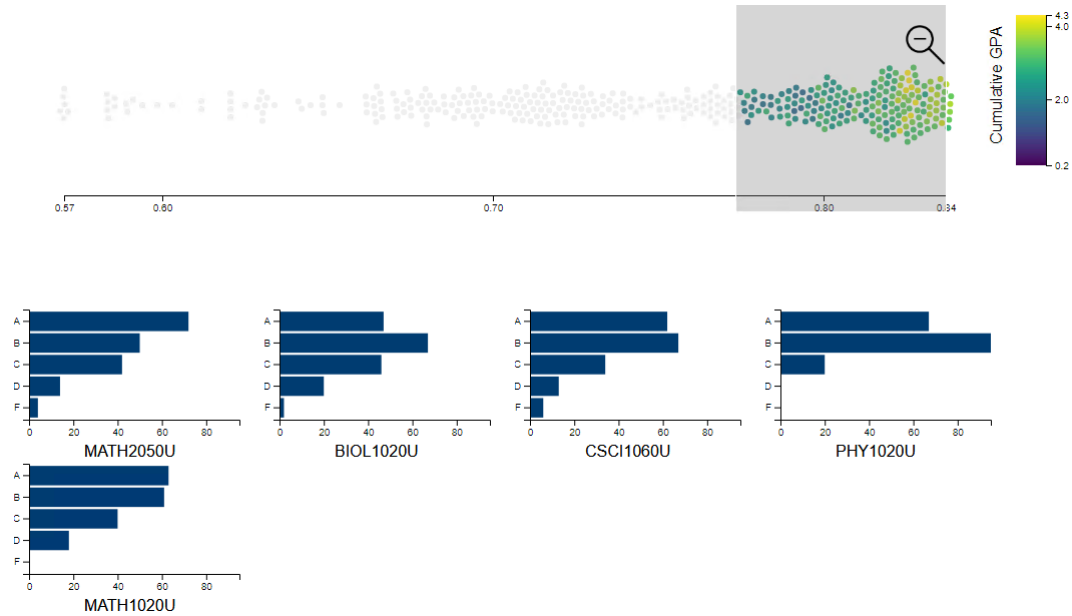

FIGURE 6.6: Zoomed in beeswarm plot with the right portion filtered. The summary bar charts update accordingly to show the grade distributions of the filtered scenarios.

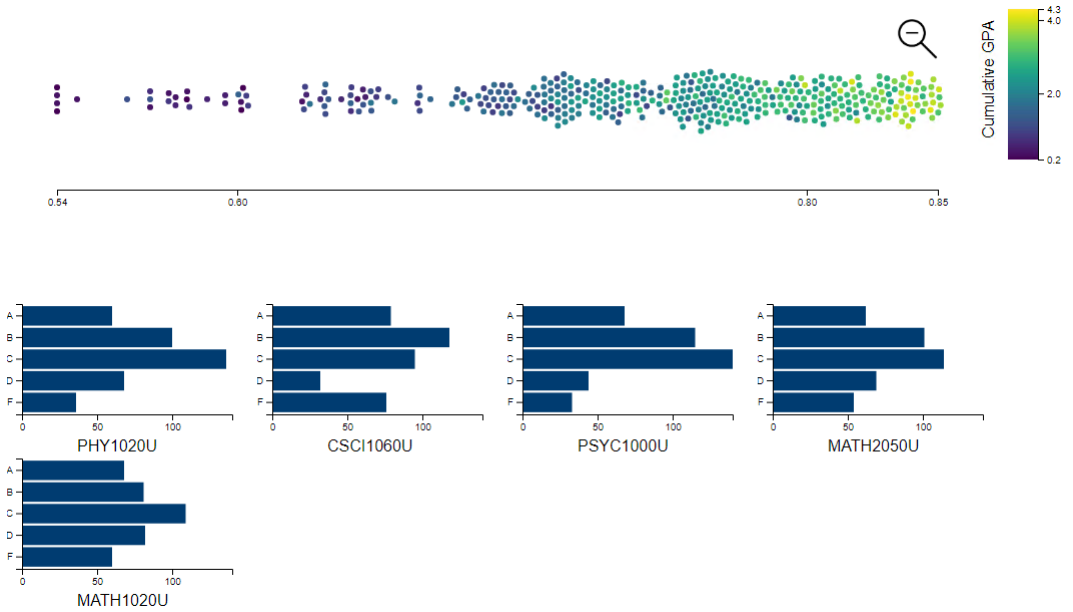

FIGURE 6.7: Zoomed in beeswarm plot showing Abby's likelihood of success if she takes Psychology as an elective. The confidence levels lie between 0.55 and 0.86 which is similar to the Biology results.

6.1.3 Case Study 3

Abby is a first year computer science student going into her second semester. Abby made some close friends in her program in semester one and felt she was adjusting well to university life. She fell slightly below average in her academic performance, finishing with an A-, four C's and a semester GPA of 2.34. Despite her underwhelming semester GPA, Abby was proud that she achieved her best grade in her programming course. Although she did not do particularly well in her elective, Biology I, she enjoyed it and was planning to take Biology II as her semester 2 elective. Abby wanted to take this opportunity to improve her GPA and decided to schedule and appointment with an Academic Advisor to discuss which elective to take.

Since Abby would like to take Biology II, the Academic Advisor enters this elective into the Student Success Prediction System alongside the 4 required courses: Calculus II (MATH1020U), Physics II (PHY1020U), Programming Workshops (CSCI1060U), and Linear Algebra (MATH2050U). Abby expresses that she is nervous about taking Biology II with an already science-heavy course load. Looking at the zoomed in view of the chart, Figure 6.5, we can see that Abby's likelihood of success values lie between 0.59 and 0.85.

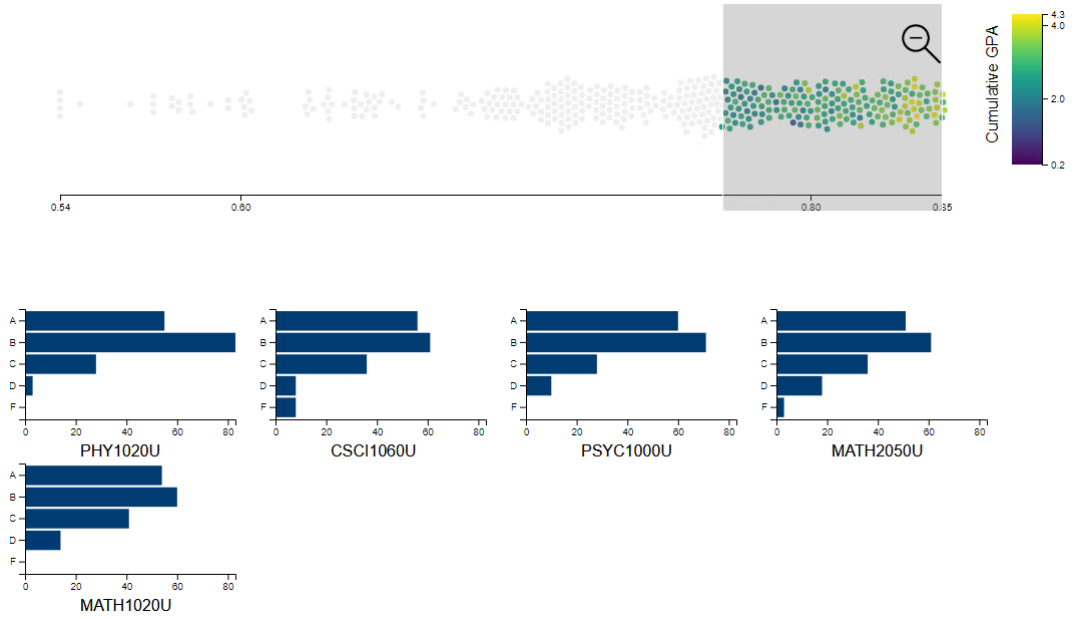

FIGURE 6.8: Zoomed in beeswarm plot with the right section filtered. The summary bar charts update to show the grade distributions of the selected scenarios.

With such a wide range of confidence values, the Advisor filters the right section of the plot to see the grade distribution at the higher end of the confidence range. Shown in Figure 6.6, the summary bar charts indicate Calculus and Physics to be the most important courses with zero F's, and the other three courses with similar distributions. With a good understanding of her likelihood of success should she choose to take Biology, Abby decides she would like to see the possibilities with a different elective. Most of Abby's friends have decided to take Introductory Psychology (PSYC1000U) so she asks the Advisor to load a chart using this course as an elective. The resulting zoomed in view is shown in Figure 6.7. The likelihood of success values lie between 0.55 and 0.86, which is very similar to the previous chart. We can also see a similar colour pattern in that the lighter colours cluster to the right of the chart, and the darker colours trail off to the left. From here, the Advisor filters the chart to examine the grade distributions on the right side of the chart, Figure 6.8. Once again, the summary bar charts look very similar. At this point, the Advisor could communicate that taking Biology or Psychology lead to a high likelihood of success with good academic performance. The advisor would encourage Abby to decide whether she is comfortable with a science-heavy schedule, or wants to take something that students tend to

find less difficult, like Psychology.

6.2 Contributions

The main contribution of this thesis was to train a model to predict whether or not a student would withdraw from the university given their grade history and admission GPA at a reasonable level of accuracy. The random forest model was trained using student vectors made up of 1500 course features with GPA grade values, and 1 high school admission grade feature with a GPA grade value. The model had an accuracy of 89%, and precision of 98%.

Furthermore, we designed and implemented a proof-of-concept system that predicts whether or not a student will dropout given a set of courses and likely grades. The confidence level of these predictions are presented as a likelihood of success measure. Since each likelihood of success value is tied to a set of grades, we can also analyze the grade distributions with respect to a range of confidence levels. This type of interaction allows the user to see which courses are possibly going to present more of a challenge, and inspire time management strategies.

6.3 Assumptions and Limitations

In this section we will outline some assumptions that we made and the limitations of our work.

6.3.1 Sparsity of Data

Machine learning models are only as good as the data they are trained on. We were fortunate that the university has consistent, digitized records since the school opened in 2003. However, since the school is still relatively new, our population is small. Our decision to train the model on only 5 years of data also significantly reduced the sample size. As the student population grows, so will the amount of training data. The results of our model are promising considering the significantly lower sample sizes used when training different models at small liberal arts colleges in the United States, and other small sample sizes [7, 13, 17, 20].

6.3.2 Homogeneity of our Sample

A further limitation imposed by our training data is the lack of homogeneity of our sample. The students in our sample come from seven different Faculties and 43 different Programs. While we acknowledge that fitting a different model for each Faculty or Program would make sense from a homogeneity point of view, our already small and sparse data sample would become even smaller and prone to over-fitting.

6.3.3 Server-Side Implementation

We were limited by our decision to use CGI instead of PHP or Node.js. Due to time constraints we chose to use a simpler CGI implementation where the CGI Python script generates a CSV file and reloads the page. This implementation led to problems with saving sessions, and adding new beeswarm plots with different course combinations on the same page.

6.3.4 External Factors

We need to acknowledge the fact that there are factors outside of our dataset that have a strong correlation to student success including the student's family and social situation, and ethnicity. For ethical reasons we decided not to include ethnicity as a training feature. Further, a student's family and social situation cannot be represented numerically or otherwise with the data that we have access to. However, we assume that these factors impact the academic performance of the student and are implicitly captured by the predictive model.

6.4 Future Work

This project presents a step towards supporting student success with machine learning. The beeswarm plot is a good way to visualize the prediction confidence levels, and the summary bar charts add another layer of useful information. Moving forward, we would like to explore different ways to visualize the summary information. For example, it could be interesting to

see the distribution of grades across the entire axis, or the grade density at each cluster.

The random forest model proved that we could accurately predict withdrawal using course and grade information using an out-of-the-box predictor. It would be interesting to see the performance of a more sophisticated model.

The predictive model could also be used to implement an early warning system. Moving forward in this direction, there a few different options. For example, using the current model, if advisors ran all students through the predictive model, the system could flag all students who are predicted to withdraw (or some other confidence threshold).

We have also discussed including other sources of data in the training set. These other sources could include data collected through Blackboard, Ontario Tech's Learning Management System (LMS). LMSs collect "student activity" information including the number of system logins, files accessed, assignments and files submitted, etc. This information could be used to generate a feature to describe a student's course engagement level. Another source could be campus Wi-Fi access points. This information could allow us to see how often a student comes to campus, and even whether or not they are attending their scheduled lecture, lab, or tutorial. Another interesting data source would be the financial aid office. Many students withdraw from school for financial reasons, though it is not widely researched [10]. Using information about who was able to access financial aid, how much they qualified for, etc. would add another factor that influences attrition alongside the academic history and enhance the predictive model. Using any of these data sources could be interpreted as invasive, and would require an extra level of security to ensure privacy.

In an attempt to address the lack of homogeneity in our training sample, we would consider including Faculty and/or Program as a training feature. Currently, we expect that the model can loosely differentiate between Faculties and Programs based on the courses a student takes. Including these training features would allow the model to explicitly differentiate between these defined groups of students. To further address homogeneity, we would consider including semester number as a training feature. We expect that the current model can loosely determine the semester or year level of the student

based on the number of courses taken and the specific course codes. Including the semester number could allow the model to explicitly differentiate between first-year students taking first-year courses, and third-year students taking first-year courses.

Finally, this work would benefit from a formal evaluation. We would like to run a user study to assess the usability of our application. This would help us understand how potential users interpret the visualizations and ways to communicate the results to students. With the development of the application ongoing, user feedback from a study could help us to enhance the interface with different filters and visualizations.

6.5 Conclusion

We can conclude that our system is a promising step towards ensuring student success from a machine learning, and visualization perspective. We were able to predict whether or not a student would dropout with a reasonable level of accuracy, and use the prediction confidence level to convey a likelihood of success. Without a formal evaluation of our system, it is difficult to say that the current interface is successful in terms of interacting with the underlying model, and displaying information in a meaningful way. However, through our iterative design method we were provided with feedback from academic advisors that leads us to believe the current interface is adequate.

In summary, we were able to train a machine learning algorithm and design and implement a system that supports student success by presenting a likelihood of success score.

Appendix A

Student Data

The following tables describe the data attributes that we were provided by Ontario Tech University.

Attribute	Description
ID	Anonymized student ID number
Term Code	Indicates the term that the course was taken
Course Code	3 – 4 letter subject code followed by 4 digit course number. Ontario Tech University undergraduate course codes end with a U, graduate course codes end with a G, and Trent course codes end with a T
Course Title	Full title of course
CRN	The Course Reference Number is a 5 digit number to uniquely identify course sections
Schedule Type	Whether the section is a lecture, lab, tutorial, web, etc.
Section Enrolment	Number of students enrolled in a section
Repeat Course Indicator	Whether or not the student has taken the course more than once

Continued on next page

Table A.1 – *Continued from previous page*

Attribute	Description
Final Grade	The final letter grade the student received in the course
Instructor ID	Anonymized ID number of the instructor that taught the course
Monday	Indicates if the course was offered on a Monday
Tuesday	Indicates if the course was offered on a Tuesday
Wednesday	Indicates if the course was offered on a Wednesday
Thursday	Indicates if the course was offered on a Thursday
Friday	Indicates if the course was offered on a Friday
Begin Time	Course start time
End Time	Course finish time

TABLE A.1: `course_history` data attributes and descriptions. This data was used to compile our training data using the Course Code, Final Grade, and Term Code columns.

Attribute	Description
ID	Anonymized student ID number
Year	Year the student was admitted
Year Level	Student year level when admitted
Faculty	Faculty student was accepted to

Continued on next page

Table A.2 – *Continued from previous page*

Attribute	**Description**
Program	Program student was accepted to
Initial City	City where the student lived at time of application
Postal Code	Postal code where the student lived at time of application
County Code	Four digit code for the county where the student lived at time of application
County Name	Name of the county where the student lived at time of application
Age	Age of the student at time of first registration
IMSTAT	Immigration status (Canadian Citizen, Permanent Resident, VISA)
SESTOT	Number of semester to complete the degree
SESLEV	Semester the student is enrolled in
FTE	Full-time equivalent (number of hours currently enrolled/number of credit hours of program per year)
CSRDE Time Status	CSRDE time status is 80% course load
Time Status	Ontario Tech time status is 9+ credit hours per term
Admit Code	Type of admission (High school, mature student, college transfer, etc.)
REZ	Indicates whether or not the student live on campus
Gender	

Continued on next page

Table A.2 – *Continued from previous page*

Attribute	**Description**
Ethnicity Code	First Nation status
adm avg	High school admission average
adm GPA	GPA equivalent of high school admission average
Return +1yr – +10yr	Separate columns indicating if the student returned after the 1st year, 2nd year, 3rd year, etc.
Grad 4yr – 10yr	Separate columns indicating if the student graduated after 4 years, 5 years, etc.
Grad	Indicates whether the student graduated
Grad Program	Program the student graduated from
Grad Faculty	Faculty the student graduated from
Grad Year	Year the student graduated in
Program +1Yr	Program the student was in after 1st year
GPA 1st sem	Semester GPA for semester 1
GPA 1st yr	Cumulative GPA after year 1
cont 2nd sem	Indicates if the student continued to second semester
Suspend +1yr	Indicates if the student was suspended after year 1
Withdraw	Did the student withdraw after year 1
cr_tkn_1t	Number of credit hours attempted in semester 1
cr_pass_1t	Number of credit hours passed in semester 1

Continued on next page

Table A.2 – *Continued from previous page*

Attribute	**Description**
credits F or W	Number of credit hours failed or withdrawn
School Board	High school board the student came from
High School	High school the student came from
School Board Code	Five digit code for the school board the student came from
Dropout	Indicates if the student dropped out
Stopout	Indicates if the student stopped out (left and returned)
Persister	Indicates if the student persists
# years stop out	Number of years stopped out
time to degree	Number of years to earn degree from first registration

TABLE A.2: `retention_raw` data attributes and descriptions. The *adm GPA* and *Persister* columns were used in the final training set.

Bibliography

[1] Lovenoor Aulck et al. "Predicting Student Dropout in Higher Education". In: *International Conference on Machine Learning* (June 2016), pp. 16–20. ISSN: 0018-8158.

[2] Lovenoor Aulck et al. *STEM-ming the Tide: Predicting STEM attrition using student transcript data*. Tech. rep. 2017.

[3] Rebecca Barber and Mike Sharkey. "Course correction: Using analytics to predict course success". In: *International Conference on Learning Analytics and Knowledge*. ACM Press, 2012, pp. 259–262. ISBN: 9781450311113.

[4] Jaroslav Bayer et al. "Predicting Drop-Out from Social Behaviour of Students." In: *International Conference on Educational Data Mining* (June 2012).

[5] *Common University Data Ontario*. 2015. URL: https://cudo.ouac.on.ca/page.php?id=7%7B%5C&%7Dtable=23%7B%5C#%7Duniv=1,2,3,8,9,11,12,14,16,17,21,22,23,24,25,27,28,29,30,31,32,33,34,42%7B%5C&%7Dy=2016 (visited on 06/11/2019).

[6] Shane Dawson et al. "Current state and future trends: A citation network analysis of the learning analytics field". In: *International Conference on Learning Analytics And Knowledge*. New York, New York, USA: ACM Press, 2014, pp. 231–240. ISBN: 9781450326643.

[7] Mykola Dekker Gerben W.and Pechenizkiy and Jan M. Vleeshouwers. "Predicting Students Drop Out: A Case Study". In: *International Conference on Educational Data Mining* (July 2009).

[8] Dursun Delen. "Predicting Student Attrition with Data Mining Methods". In: *Journal of College Student Retention: Research, Theory & Practice* 13.1 (May 2011), pp. 17–35. ISSN: 1521-0251.

[9] Colin Fay. *[ggplot2] Welcome viridis ! - (en) The R Task Force*. 2018. URL: https://rtask.thinkr.fr/blog/ggplot2-welcome-viridis/ (visited on 07/05/2019).

[10] David S. Fike and Renea Fike. "Predictors of First-Year Student Retention in the Community College". In: *Community College Review* 36.2 (Oct. 2008), pp. 68–88. ISSN: 0091-5521.

[11] Eibe Frank and Mark Hall. *Visualizing Class Probability Estimators*. Tech. rep. 2003.

[12] Sandeep M. Jayaprakash et al. "Early Alert of Academically At-Risk Students: An Open Source Analytics Initiative". In: *Journal of Learning Analytics* 1.1 (May 2014), pp. 6–47.

[13] Zlatko J. Kovačić. "Early prediction of student success: Mining student enrollment data". In: *Informing Science & IT Education Conference*. 2010.

[14] Kaisa Miettinen. "Survey of methods to visualize alternatives in multiple criteria decision making problems". In: *OR Spectrum* 36.1 (Jan. 2014), pp. 3–37. ISSN: 0171-6468.

[15] D.F.O Onah, J Sinclair, and Boyatt. "Dropout Rates of Massive Open Online Courses". In: *International Conference on Education and New Learning Technologies)* (2014), pp. 1–10. ISSN: 2340-1117.

[16] Kathleen Pittman. "Comparison of data mining techniques used to predict student retention." PhD thesis. 2008, p. 416. ISBN: 0549474684.

[17] Kevin Rask. "Attrition in STEM fields at a liberal arts college: The importance of grades and pre-collegiate preferences". In: *Economics of Education Review* 29.6 (Dec. 2010), pp. 892–900. ISSN: 02727757.

[18] Virginia Staudt Sexton. "Factors Contributing to Attrition in College Populations: Twenty-Five Years of Research". In: *The Journal of General Psychology* 72.2 (Apr. 1965), pp. 301–326. ISSN: 0022-1309.

[19] Dhawal Shah. *By The Numbers: MOOCS in 2018 — Class Central*. 2018. (Visited on 04/17/2019).

[20] JF Superby, J. P. Vandamme, and N Meskens. "Determination of factors influencing the achievement of the first-year university students using data mining methods". In: *International Conference on Intelligent Tutoring Systems*. 2006, pp. 1–8.

[21] Mack Sweeney et al. "Next-Term Student Performance Prediction: A Recommender Systems Approach". In: (2016).

[22] Vincent Tinto. "Dropout from Higher Education: A Theoretical Synthesis of Recent Research". In: *Review of Educational Research* 45.1 (1975), pp. 89–125.

[23] Vincent Tinto. "From Theory to Action: Exploring the Institutional Conditions for Student Retention". In: *From Theory to Action: Exploring the Institutional Conditions for Student Retention*. Review of Educational Research, 2010, pp. 51–89. ISBN: 9780874216561.