Early Detection of At-Risk Students Using LMS Data

by

Akshay Kumar Dileep

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2021 by the
Graduate Supervisory Committee:

Ajay Bansal, Chair
James Cunningham
Ruben Acuna

ARIZONA STATE UNIVERSITY

May 2021

ABSTRACT

Calculus as a math course is important subject students need to succeed in, in order to venture into STEM majors. This thesis focuses on the early detection of at-risk students in a calculus course which can provide the proper intervention that might help them succeed in the course. Calculus has high failure rates which corroborates with the data collected from Arizona State University that shows that 40% of the 3266 students whose data were used failed in their calculus course.

This thesis proposes to utilize educational big data to detect students at high risk of failure and their eventual early detection and subsequent intervention can be useful. Some existing studies similar to this thesis make use of open-scale data that are lower in data count and perform predictions on low-impact Massive Open Online Courses(MOOC) based courses. In this thesis, an automatic detection method of academically at-risk students by using learning management systems(LMS) activity data along with the student information system(SIS) data from Arizona State University(ASU) for the course calculus for engineers I (MAT 265) is developed. The method will detect students at risk by employing machine learning to identify key features that contribute to the success of a student.

This thesis also proposes a new technique to convert this button click data into a button click sequence which can be used as inputs to classifiers. In addition, the advancements in Natural Language Processing field can be used by adopting methods such as part-of-speech (POS) tagging and tools such as Facebook Fasttext word embeddings to convert these button click sequences into numeric vectors before feeding them into the classifiers. The thesis proposes two preprocessing techniques and evaluates them on 3 different machine learning ensembles to determine their performance across the two modalities of the class.

DEDICATION

This thesis is dedicated to my parents for their unconditional love and support towards my goals, which led to the successful completion of my thesis.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my thesis mentor Dr. Ajay Bansal for his supervision, guidance, and encouragement throughout my thesis. I am thankful to Dr. Ruben Acuna for being supportive of my research and for willing to serve on my thesis committee as a member.

I would like to thank James Cunningham, for providing me with the valuable data and his guidance throughout my thesis. Without this dataset generating and working on this unique problem wouldn't be possible.

Finally, I would like to thank Arizona State University for providing the amenities as well as being kind enough to work with their data which was crucial in the successful completion of my thesis and the department of CIDSE for the constant guidance and assistance throughout my master's program and thesis.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER Page

Chapter 1

INTRODUCTION

1.1 Overview

In education, data mining is the method to explore the various types of unique data that are generated by students so that they can be used to aid their learning. Data mining can determine new patterns in students which will benefit them. Researchers Hooshyar *et al.* (2020), Shin and Shim (2020) have used data mining techniques to model, predict and learn about the various patterns that students usually exhibit. The collection of data by the LMS does not require any administration or intrusion from staff or faculty making it easy to collect also. Thus the study by Villegas-Ch *et al.* (2020) had concluded that LMS in conjunction with data mining techniques can improve student performance in a course.

Macfadyen and Dawson (2010) concluded in their study that time as a feature played a key role in the success of a student in a course. Study patterns, student data engagements, participation criterion, etc are various key factors that can be represented using the data mined from an LMS. The goal of this thesis is to thus identify and determine if there is any pedagogical meaning for a feature such as student activity to detect the success of a student in a calculus course.

1.2 Problem Statement

The number of students entering and completing science, technology, engineering, and mathematics (STEM) related subjects in the U.S. has received increased attention from education researchers. Part of the reason for this increased focus is the high failure/drop rate in these courses (Seymour and Hewitt, 1997). The success of students in fields such as STEM is seen as a key feature for the U.S. to remain strong in the global markets (Chen, 2009). However, many students are unable to succeed. The calculus requirement of STEM courses is often perceived as a 'filter' rather than a 'pump' (Steen, 1988). High failure rates in calculus classes aimed at freshman engineering and science students are not new.

Arizona State University uses LMS as an infrastructure system of the organization which can help keep an attendance record of students, grading the assignment, or broadcasting any messages or information vital to students (Pongpaichet *et al.*, 2020). A study conducted by Maor and Volet (2007), concluded that interactivity with an LMS is an essential environment and constitutes an important factor in online learning. Generally speaking, an LMS is a self-contained system consisting of educational resources to direct the teaching and learning process by retaining, pursuing, and utilizing student interaction records within the LMS. The data generated in educational settings such as LMS, traditional classroom-based interactions, etc. are utilized in understanding the student's academic performance, and this process is commonly known as Educational Data Mining (EDM). It is relatively a new research area where methods are studied, developed, and tested to enhance the standard of teaching and learning. To this effect, in literature, there exist several notable reviews on the application of algorithms on the educational dataset to derive patterns (unsupervised approach) (Dutt *et al.*, 2017) or predictions (supervised approach) (Baker

and Yacef, 2009).

This thesis will utilize the supervised approach to predict student performance from interaction within an LMS for a calculus course. Furthermore, the primary focus will be on the LMS activity data which consists of the button click information within an LMS. Dutt and Ismail (2019) in their research used different classifiers on the LMS dataset to obtain their results. Pongpaichet *et al.* (2020) is the base paper upon which this thesis builds. Instead of limiting ourselves to just the online classes, we delved into the predictive capabilities of LMS data on Face-2-Face classes as well as the course's online counterpart. The use of activity data from LMS and its impact on student success will be another focus of this study. Also, the quality of the dataset being used comes from actual student data sourced from Arizona State University students and the major criteria of the study will focus on identifying At-risk students early on into their Calculus course.

Tran and Sato (2017) in 2017 conducted a study using API sequences to develop malware detecting classifiers using Natural Language Processing techniques. They use API call sequence as their input because it is how a program communicates with the infected OS, thus acting as a language of communication for interaction. The study shows promising results in predicting malware from normal programs and works well as a proof of concept in the application of NLP in the field of classification of natural language sequences. This method of interpreting sequence meaning can thus be extrapolated to other fields where sequences are generated such as Education.

This thesis also aims to make use of the same technique in the field of learning analytics by converting student button click data into student button click sequence and eventually building a classifier that can predict the success of a student in a course. The student button click data generally follows caliper instructions and hence is made up of natural language. Sequencing of button variables can thus create patterns of

student workflow within an LMS. This is the data upon which the thesis strives to build its classifiers.

Chapter 2

BACKGROUND LITERATURE

2.1 Calculus and high failure rates

Pilgrim (2010) in 2010 observed that close to 40% of those who took Calculus for Physical Scientists course I (MATH 160) needed to repeat it. This mirrored the national average of failure rates at that time. The situation is similar even at Arizona State University, this cementing the issue deeply. The effects of failing a course can take its toll financially and with respect to time as the student needs to retake the course in order to achieve their degree. There can also be a mental aspect to the toll due to stress.

Worthley (2013) concluded in their paper that the difference between success and failure of a student in a calculus course lies with the amount of time spent and its quality within a course. The successful students had more study strategies and means rather than spending time just keeping up with their due assignments. Their paper also mentions that if those that need extra guidance can be advised appropriately, then there may be an improvement in the pass rate for the calculus course that they worked with. The main goal thus becomes the detection of students at risk, whose information after which can be used to improve their quality of education.

2.2 Early Warning Systems using LMS

The analysis of LMS data is often described as learning analytics, which is defined as "the measurement, collection, analysis, and reporting of data about learners and their context, for purposes of understanding and optimizing learning and the environments in which it occurs" (Siemens and Long, 2011). There is a large concentration of research within learning analytics where LMS data have been used for the purposes of predicting student success in a class either by gauging their performance to determine whether they would be at risk of failing the course (Wolff *et al.*, 2014; Kondo *et al.*, 2017; Cui *et al.*, 2020). This follows the natural progression of the works in learning analytics which involve the eventual implementation of interventions and personalized feedback. Purdue University (Pistilli and Arnold, 2010) in 2009 developed a Course Management System (CMS) to alert students grade and action plan. This was implemented by the instructors by analyzing their past academic performances spanning two semesters.

Such a system can then be used to identify the impact of new teaching methods, growth and dips in the learning activity, and other various observations that can be derived. Instructors currently are unable to gauge the success of a student as they did previously due to factors such as, increase in class size, slow rate of engagement amongst students and the instructors, student absentees, and the struggle of interacting with the student who might be in need of help as concluded by the studies of Macfadyen and Dawson (2010).

Instructors, therefore, require better tools or strategies that will allow them to identify probable students who are at-risk, in order to help them succeed. Based on some preliminary findings by Wang and Newlin (2002) proposed in 2002 that data on student online activity in a web-based Learning Management System (LMS)

may provide an early indicator of student academic performance. More recently a study by Campbell *et al.* (2007) stated that the application of academic analytics to institutional LMS data can offer broader insights into student success and in identifying students at-risk of failure. For example in the paper by Campbell *et al.* (2006), they conducted a regression analysis of online-based student activity and performance data. They then demonstrated that while student SAT scores are mildly predictive of student success, the inclusion of an additional variable such as LMS login information ended with increasing the predictive power of the model by three. This shows the effect of LMS data on boosting predictive analysis for gauging student success.

This thesis will focus on two approaches in using LMS data to develop an early warning system that can help identify students at risk. Approach A will use a more general approach in handling the LMS activity data whereas approach B will elaborate on a newer means of transforming LMS data to capture more information.

Chapter 3

RELATED WORKS

3.1 Related works to approach A

LMS is the go-to software solution for eLearning these days. A study conducted by Maor and Volet (2007), concluded that interactivity with an LMS is an essential environment and constitutes an important factor in online learning, this validates the usage of LMS data for predictive analysis as they have causation to the susses of a student in a class. The LMS used for this study is Canvas (Instructure, 2016). They have been part of ASU's curriculum since 2018 and have played a key part in transitioning fully online during the time of the pandemic of 2020 (COVID-19). The key dataset that this study will focus on is the Canvas Activity dataset, which records every button lick activity of a student on the Canvas platform.

Macfadyen and Dawson (2010) in 2010 found that time spent on assignments, time spent online, and a number of discussion messages posted were key features that derived the success of a student in a course. Similar research by Kupczynski *et al.* (2011) stated that the time spent online and login frequency have also been key factors that failed to contribute to the success of a student in a course, which frays from the expectations. This provides us with the notion that the quality of time spent learning contributed better to the quantity of time spent to learn. Utilizing LMS data however allows the identification of changes in student interaction patterns, performance in tests, etc. Kadoić and Oreški (2018) in their research performed a correlation analysis between the key LMS logs activity variable and student's final grades. They then conclude by stating that students who often opening the file will

have a higher grade, and students who have a higher grade will do more activities before they study one day. Yet there hasn't been any usage of such activity indicator variables used as features in predictive analysis, which is what this thesis hopes to achieve.

The goal of this study is to make use of the Canvas LMS activity dataset and its predictive capabilities on Arizona State university student data for MAT 265 that has both an Online and a Face-2-Face modality. In our study, we want to identify the relationship of student button click activity on the LMS to the success of a student in a class. This type of behavior matching can set arise new ways to improve existing early warning predictive systems by making use of these features.

3.2 Related works to Approach B

There are a majority of institutions that use the tools such as Learning Management Systems (LMSs) in teaching. Some of the most commonly equipped LMS are Moodle (Moodle, 2012), Blackboard (Blackboard, 2015), Canvas (Instructure, 2016), and Absorb (Inc., 2015) LMS. A study conducted by Maor and Volet (2007), concluded that interactivity with an LMS is an essential environment and constitutes an important factor in online learning, this validates the usage of LMS data for predictive analysis as they have causation to the susses of a student in a class.

Morris *et al.* (2005) found that the number of content pages viewed was a significant predictor in three fully online courses in eCore with 354 students. They used eight duration and frequency variables, and no measurements of performance. Further analysis using these predictors with the final grade as the target feature, 284 students who completed showed a variance of 31 percent in their final grades which were accounted for by the number of discussion posts and content pages viewed, and the overall time spent on viewing discussion posts. Moreover, they found that those

who withdrew had a significantly lower frequency of activities and eventually spent less time spent online, compared to those who completed. Macfadyen and Dawson (2010) also found that the number/frequency of links and files viewed had a positive correlation with the final grade.

NLP has been used widely for the purposes of classification in the field of Machine Learning. Razno (2019) demonstrated in their paper the applications of NLP for the purpose of text classification. NLP also shows promise in identifying the sentiment of a text which has proven to be useful in the field of recommendation systems and sentiment analysis. Ghorpade and Ragha (2012) in 2012 produced a study that achieved an accuracy of 96% when identifying the sentiment of hotel reviews. They were also able to overcome the problem of loss of text information by using well-trained training sets.

Tran and Sato (2017), Nagano and Uda (2017) uses natural language processing to transform data and then uses machine learning classifiers to distinguish between malware and benignware. This technique of using NLP to transform data sequence in order to identify patterns and meaning shows promising results in the field of malware detection and could be adopted for use in learning analytics. Ki *et al.* (2015) made use of DNA sequence alignment algorithms for their analysis. These patterns in combination with the critical API sequences were made use to decide whether an examined program is malware or not. They arrived at an accuracy of about 99.8% for this method. This thesis would like to make use of the same technique of prediction using sequence data by generating activity sequence for all students in a course using their event trigger activity data from the LMS. This sequence would be a collection of all the labeled buttons and clicks within an LMS for a course.

The goal of this thesis is to take inspiration from the works of Tran and Sato (2017) and make use of the same sequence prediction techniques for the domain of learning analytics to predict students at risk early in a course such as that proper intervention can be administered. The dataset used for this purpose is a combination of the LMS activity dataset and student information system (SIS) dataset, in the capacity for both the online and offline modality of the same course.

Chapter 4

DATASET

There are two sources of data that will be used in this project. They are the LMS activity dataset from Canvas for ASU for the course MAT 265 and the Student Information System (SIS) dataset. MAT 265 is Calculus for engineers 1. This class has both an online and a F2F modality which will be useful in identifying trends in both areas. For the Online classes, both the teaching and the regular LMS activities such as group discussions exams, etc occur via Canvas. For the F2F classes, the learning materials are hosted on Canvas but general discussions and exams may be conducted F2F also.

4.1 Canvas Activity Dataset

The canvas activity dataset is a collection of every instance of a button click activity recorded on Canvas. Since canvas is still an integral part of the F2F version of the class there are significant records for the activity of students on Canvas even for F2F classed. The dataset used contains the records of students from 2018 fall till 2020 spring, with a total of 3261 student information for both online and F2F combined. Table 4.1 contains the Canvas activity dataset features and their descriptions. These will be the primary features upon which data preprocessing will take place.

Table 4.1: Canvas Activity Dataset Features and Their Description

Features	Description
eventtime	Records the time of the event
eventtype	Unique identifier for event type
actiontype	Unique identifier for action type
objecttype	Unique identifier for object type
detail	Contains the name of the button clicked

4.2 Student Information System (SIS)

The SIS dataset is a general student information dataset maintained by ASU. this contains various demographic and course-related features which will be used by our models. Table 4.2 contains the SIS dataset features and their descriptions. This dataset will further be merged with the preprocessed Canvas activity dataset upon which modeling will take place. The data will be merged on the unique student identifier such that every row contains the student system information and their associated canvas activity data. Some unique features part of this dataset include faculty difficulty index score and Starbucks affiliate feature.

The dataset contains details about 3266 students. The composition is made up of 2440 male students, 822 female students, and 4 undefined students. There are a total of 986 students who transferred. The online student composition contains 1195 students whereas the offline modality contains 2071 which constitutes a ratio of 1:3 ratio of the overall dataset. From table 2, we can observe features such as ethnicity and gender which are controversial features in the applications of machine learning. This is usually due to the biased nature in which the dataset is formed historically. But this absolute use of this tool is to provide students with external

Table 4.2: SIS Dataset Features and Their Description

Features	Description
age_at_class_start	Contains the age of the student at the start of the class
course_load	Sum of credit hours taken in a term by a student
ever_pell	Has the student ever been Pell Grant eligible?
fac_diff_index	Average grade rated by the professor in the previous 2 years
first_gen	Indicates whether a student is a first generation college student
incoming_gpa	Incoming GPA of a student. Transfer first, if null then High School
part_time_offi	Label for if a full-time academic load is not taken
prev_term_gpa	Cumulative GPA of previous term
starbucks	Binary indicating whether a student is a Starbucks affiliate
stud_modality	Indicates whether the student is a face-to-face learner, online, or other
transfer	Binary that indicates whether the student is a transfer student
acad_level	Academic level at the beginning of the term (BOT).
ethnicity	Student's ethnicity
gender	Gender of the student

help such as tutoring or additional support which do not necessarily discriminate against any student.

Table 4.3 and Table 4.4 show the frequency composition of the academic level and the ethnicity features. The dataset contains a majority of students from the Freshman academic level. In terms of ethnicity, the majority of the student falls under the Caucasian category.

Table 4.3: Academic Level Frequency

Academic Levels	Count
Freshman	1436
Sophomore	855
Junior	511
Senior	229
Post-Bacc Undergraduate	188
Graduate	30
Non-degree Undergraduate	17

Table 4.4: Ethnicity Frequency

Ethnicity	Count
White	1751
Hispanic	620
Asian	419
2 or More	163
Black	140
NR	129
American I	36
Haw/Pac	8

Chapter 5

METHODOLOGY AND IMPLEMENTATION

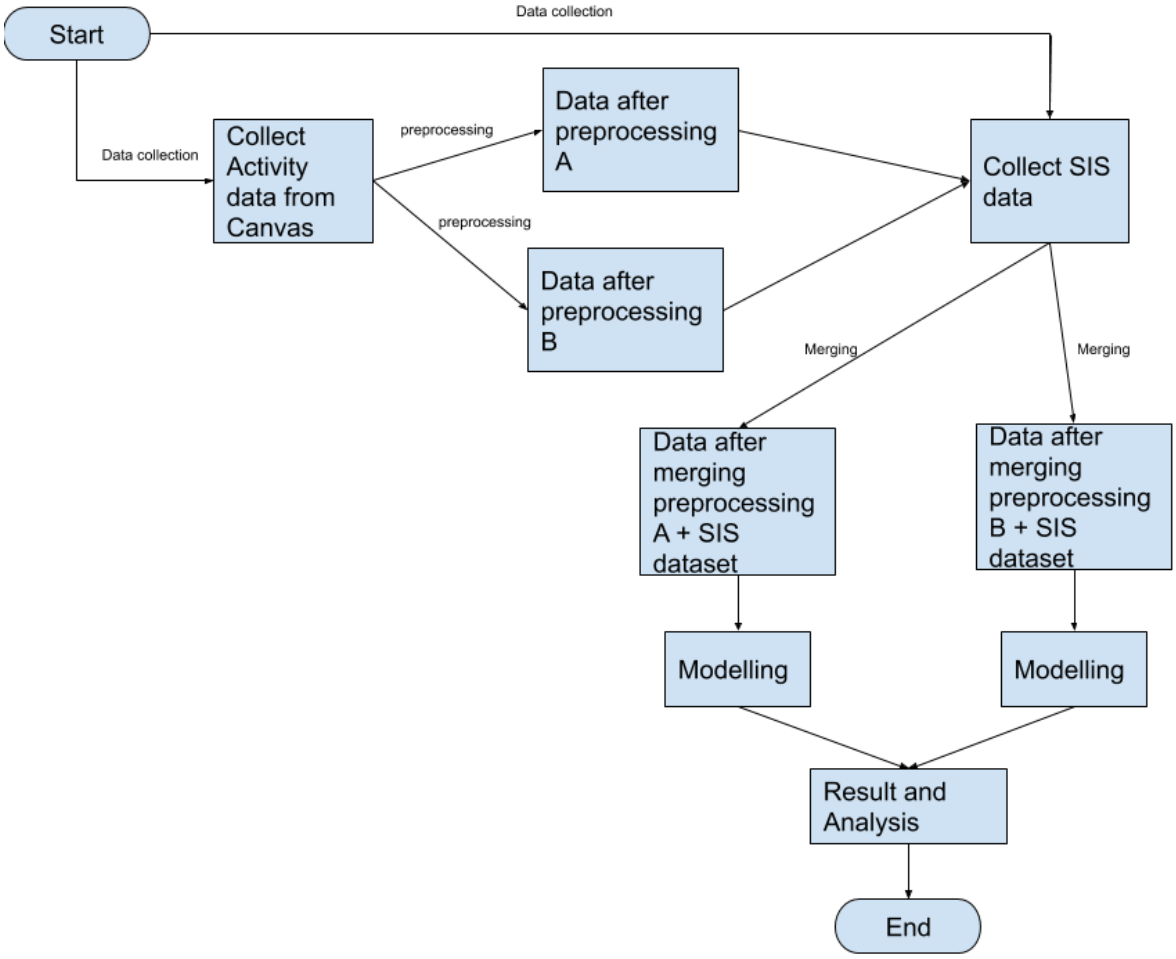

Figure 5.1: Process Workflow

To develop a machine learning model, we have followed the process flow as shown in Fig. 5.1 .

5.1 Approach A

Approach A describes the use of generating frequency count of various button click features for the purpose of modelling.

5.1.1 Data Processing

Table 4.1 shows us the data format for the Canvas activity dataset. the eventtype, actiontype, objecttype, and the detail features are caliper data and are interpreted in the following manner.

$$eventtype + actiontype + objecttype + detail$$

But the detail column is a feature that is highly inconsistent in nature. This is because its content is named by the user who sets up the canvas course shell, either the professor or his teaching assistant, etc. Due to the inconsistencies, we decided to drop the details column and use the following setup.

$$eventtype + actiontype + objecttype$$

The above formula generated 17 unique strings as shown in table 5.1, which is the intermediate form of the data. Table 5.1 also shows the count of the available 17 unique strings. The ideal data form would be to generate one row per student and to capture all their button click activities for a select period. Thus these generated 17 unique strings were then converted into features for each student and a frequency of use counter was set up to capture these event activation across multiple weeks.

This dataset was then merged on unique ID and class start date with the SIS dataset, which contains the student demographic information. The merged dataset will be the actual data upon which modeling occurs. This concludes the preprocessing for approach A.

Table 5.1: Caliper Features From Canvas Activity Dataset

Features	count
NavigationEvent NavigatedTo Entity	1127896
NavigationEvent NavigatedTo Page	229906
NavigationEvent NavigatedTo Thread	125339
NavigationEvent NavigatedTo AssignableDigitalResource	81961
Event Modified Attempt	32325
AssessmentEvent Submitted Attempt	12633
AssignableEvent Submitted Attempt	5792
MessageEvent Posted Message	684
MessageEvent Posted Message	684
Event Modified AssignableDigitalResource	72
Event Modified Entity	55
Event Created Document	37
Event Created AssignableDigitalResource	35
Event Modified Page	8
Event Created Page	4
Event Deleted Page	4
Event Created Entity	2
ThreadEvent Created Thread	2

Figures A.1 - A.3 shows us the selective code for data preprocessing Approach A.

Due to the unavailability of the details columns as a result of data inconsistencies, a new approach is proposed which utilizes 100% of the data available by generating a student activity sequence and using advanced natural language processing techniques

to perform classification upon.

5.2 Approach B

Approach B capitalizes on the downfalls present in Approach A. It makes use of feature generation by converting the button click activity into student activity sequence, which is then transformed using various NLP techniques.

5.2.1 Data Processing

Because of the exclusion of the details feature from the Canvas activity dataset, we are losing valuable information that could be extracted. To solve this ordeal we will be employing two Natural language processing techniques to help make use of the details column, namely Part of speech tagging (POS), and Facebook fasttext word embedding.

To make use of both the NLP techniques, we need to generate an intermediate state for the data. Since the goal for the dataset generation is to form one row per student entry, we decided to concatenate all the button click caliper information onto a single string and save it for each student based on their timely activation. The formula below shows us the composition of each button click entry.

$$eventtype + actiontype + objecttype + detail$$

The intermediate data will have one row per student and a feature that holds a string with all the activity for a given time frame within the same sting. This string will hence represent the activity sequence that the student follows within the time frame. This helps with the usage of POS tagging and Facebook fasttext embedding.

5.2.2 Parts-of-speech Tagging

Parts-of-Speech (POS) Tagging is a process of labeling words in a sentence. The labels are part of speech tags such as nouns, verbs, adjectives, etc. As per our understanding, POS is the first step in any Natural Language Processing (NLP) system or experimentation(NLTK, 2019). For our experimentation, we used the NLTK POS Tagger. The POS tagger uses the preceding as well as the next tag contexts to tag an individual word. It uses Penn Treebank tagset for Parts of speech tags. An example of one of the tagged questions is shown below:

Navigation/NNP Event/NNP Navigated/NNP To/TO Entity/NNP Home/NNP
The tags are then converted into a series of one-hot vector encodings.

5.2.3 FastText

FastText is a way of representing vectors (Facebook, 2017). We use FastText representation because it handles vocabulary words better. FastText uses an approach based on the skip-gram model. Here, each word is represented as a bag of character n-grams. The dataset had over 32000 unique tokens which were compressed into word vectors of dimensions 300 x 1 using the FastText word embedding model.

This dataset was then merged on unique ID and class start date with the SIS dataset, which contains the student demographic information. The merged dataset will be the actual data upon which modeling occurs. This concludes the preprocessing for approach B.

Table 5.2 list the different pre-processing techniques compared in our study along with their dimensionality and description.

The reason why the 300 length word embedding array is flattened as features for the approach is due to the input nature of the models being used. No three-

Table 5.2: Post Processing Dataset Dimension

Pre-Processing technique	Dimensionality	Description
Approach A (A)	47X1	Merge between SIS and Canvas button frequency count
Approach B (B)	369X1	Both POS tagging with Facebook Fasttext embedding merged with the SIS dataset

dimensional data is allowed as an input for the models and hence the need for flattening the word embedding array into 300 features.

Figures A.4 - A.6 shows us the selective code for data preprocessing Approach B.

5.3 Oversampling of Minority Class

Due to the nature of the problem, the class which categorizes student success as pass or fail is highly imbalanced as most students tend to pass a class. As a result, Oversampling of the minority class was equipped to avoid this problem. The most widely used approach to synthesizing new examples is called the Synthetic Minority Oversampling Technique or SMOTE for short Chawla *et al.* (2002). This procedure can be used to create as many synthetic examples for the minority class as are required. It first uses random under-sampling to trim the number of examples in the majority class, then uses SMOTE to over-sample the minority class to balance the class distribution. After Oversampling of the minority class the number of rows for

the ASUO dataset becomes 1282 and for the F2F dataset becomes 2598.

5.4 Modelling

The two preprocessing techniques are then subjected to various machine learning ensembles upon which our analysis will be based. The three machine learning techniques are Random Forest, XGBoost, and CatBoost.

5.4.1 Random Forest Classifier:

Random forest, like its name implies, consists of a large number of individual decision trees that operate as a collection (Kho, 2018).To classify any object based on newer attributes each tree gives a classification and then maps the vote of the tree to that class. The forest then chooses the classifications having the most votes of all the other trees. Each individual tree in the random forest generates a class prediction and the class with the most votes becomes the model's prediction (Lee *et al.*, 2010).

The advantages of using a random forest classifier are that it is versatile, has quick prediction/training speed, handles unbalanced data, and has low bias and moderate variance. All decision trees in Random Forrest have a low bias and a high variance since all trees are averaged out the resultant bias is low and the variance is moderate for the model.

The random forest model being used has the following parameters as depicted in table 5.3. This model was then run with 2 different pre-processing approaches on 2 different datasets (ASUO and F2F). The pre-processed dataset was split into train and test datasets with a split of 75% training and 25% test data.

Table 5.3: Random Forest Classifier Parameters

Parameters	Value
n_estimators	1000
max_features	'auto'
max_depth	140
criterion	"gini"
random_state	(42)

5.4.2 XGBoost Classifier:

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. XGBoost stands for eXtreme Gradient Boosting. It is a software library that you can download and install on your machine, then access it from a variety of interfaces. The implementation of the algorithm was engineered for the efficiency of computing time and memory resources (Brownlee, 2021). A design goal was to make the best use of available resources to train the model. Some key algorithm implementation features include: (i) sparse aware implementation with automatic handling of missing data values; (ii) block structure to support the parallelization of tree construction; (iii) continued training so that you can further boost an already fitted model on new data.

XGBoost and Gradient Boosting Machines (GBMs) are both ensemble tree methods that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture. However, XGBoost improves upon the base GBM framework through systems optimization and algorithmic enhancements.

The XGBoost model being used has the following parameters as depicted in table 5.4. This model was then run with 2 different pre-processing approaches on 2 different

datasets (ASUO and F2F). The pre-processed dataset was split into train and test datasets with a split of 75% training and 25% test data.

Table 5.4: XGBoost Classifier Parameters

Parameters	Value
base_score	0.5
booster	'gbtree'
colsample_bylevel	1
colsample_bynode	1
colsample_bytree	1
gamma	0
gpu_id	-1
importance_type	'gain'
interaction_constraints	''
learning_rate	0.300000012
max_delta_step	0
max_depth	6
min_child_weight	1
missing	nan
monotone_constraints	'()'
n_estimators	100
n_jobs	()
num_parallel_tree	1
objective	'multi:softprob'
random_state	()
reg_alpha	1
reg_lambda	None
scale_pos_weight	None
subsample	-1
tree_method	'exact'
validate_parameters	1
verbosity	None

5.4.3 CatBoost Classifier:

CatBoost is a machine learning algorithm that uses gradient boosting on decision trees. It is available as an open-source library. "CatBoost" name comes from two words "Category" and "Boosting". The advantages of CatBoost (Ray, 2017) are as follows: (i) Performance: CatBoost provides state-of-the-art results and it is competitive with any leading machine learning algorithm on the performance front. (ii) Handling Categorical features automatically: We can use CatBoost without any explicit pre-processing to convert categories into numbers. CatBoost converts categorical values into numbers using various statistics on combinations of categorical features and combinations of categorical and numerical features. (iii) Robust: It reduces the need for extensive hyper-parameter tuning and lowers the chances of overfitting also which leads to more generalized models. (iv) Easy-to-use: You can use CatBoost from the command line, using a user-friendly API for both Python and R.

Data is generally of two types; homogeneous or Heterogeneous. Data that is ambiguous with uncertainty, poor quality, and high data redundancy are called Heterogeneous data. They usually have a high data type and format variability. Homogeneous data is a dataset that is made up of the same data type and format.

Credit card information is heterogeneous whereas image or text dataset is homogeneous.

CatBoost Works well on Heterogeneous data. Thus for a classification problem with Heterogeneous data using CatBoost will outperform the majority of the Boosting algorithms.

The CatBoost model being used has the following parameters as depicted in table 5.5 and table 5.6. This model was then run with 2 different pre-processing approaches

on 2 different datasets (ASUO and F2F). The pre-processed dataset was split into train and test datasets with a split of 75% training and 25% test data.

Table 5.5: CatBoost Classifier Parameters - 1

Parameters	Value
nan_mode	'Min'
eval_metric	'Logloss'
iterations	1000
sampling_frequency	'PerTree'
leaf_estimation_method	'Newton'
grow_policy	'SymmetricTree'
penalties_coefficient	1
boosting_type	'Plain'
model_shrink_mode	'Constant'
feature_border_type	'GreedyLogSum'
bayesian_matrix_reg	0.10000000149011612
l2_leaf_reg	3
random_strength	1
rsm	1
boost_from_average	False
model_size_reg	0.5
subsample	0.800000011920929
use_best_model	True
class_names	[0,1]
random_seed	0

Table 5.6: CatBoost Classifier Parameters - 2

Parameters	Value
depth	6
posterior_sampling	False
border_count	254
classes_count	0
auto_class_weights	'None'
sparse_features_conflict_fraction	0
custom_metric	['Logloss', 'AUC:hints=skip_train false']
leaf_estimation_backtracking	'AnyImprovement'
best_model_min_trees	1
model_shrink_rate	0
min_data_in_leaf	1
loss_function	'Logloss'
learning_rate	0.029999999329447743
score_function	'Cosine'
task_type	'CPU'
leaf_estimation_iterations	10
bootstrap_type	'MVS'
max_leaves	64

Figures A.7 - A.9 shows us the selective code for data modelling.

Chapter 6

EXPERIMENTAL RESULTS

This chapter consolidates all the models generated for this research.

6.1 ASUO - Online Dataset

This section holds all the models on the ASUO Dataset.

6.1.1 Approach A

This section holds all the models that make use of Approach A using ASUO dataset.

Random Forest

Figure 6.1 represents the classification report of the Random forest model whose data uses approach A on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 5th and whose AUC score ranks 5th amongst all models using the ASUO dataset.

```
RandomForestClassifier(max_depth=140, n_estimators=1000, random_state=42)

 Classification report :
               precision    recall  f1-score   support

        fail       0.76      0.67      0.71       161
        pass       0.70      0.78      0.74       160

    accuracy                           0.73       321
   macro avg       0.73      0.73      0.73       321
weighted avg       0.73      0.73      0.73       321

Accuracy   Score :  0.7258566978193146
Area under curve :  0.8138781055900621
```

Figure 6.1: Random Forest Model on Approach A(ASUO modality)

XGBoost

Figure 6.2 represents the classification report of the XGBoost model whose data uses approach A on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 4th and whose AUC score ranks 6th amongst all models using the ASUO dataset.

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)

 Classification report :
               precision    recall  f1-score   support

        fail       0.77      0.66      0.71       161
        pass       0.70      0.80      0.75       160

    accuracy                           0.73       321
   macro avg       0.73      0.73      0.73       321
weighted avg       0.73      0.73      0.73       321

Accuracy   Score :  0.7289719626168224
Area under curve :  0.8118788819875776
```

Figure 6.2: XGBoost Model on Approach A(ASUO modality)

CatBoost

Figure 6.3 represents the classification report of the CatBoost model whose data uses approach A on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 3rd and whose AUC score ranks 3rd amongst all models using the ASUO dataset.

```
<catboost.core.CatBoostClassifier object at 0x7fb345c96e50>

 Classification report :
               precision    recall  f1-score   support

        fail       0.76      0.69      0.72       161
        pass       0.71      0.78      0.75       160

    accuracy                           0.74       321
   macro avg       0.74      0.74      0.73       321
weighted avg       0.74      0.74      0.73       321

Accuracy   Score :  0.735202492211838
Area under curve :  0.8272903726708075
```

Figure 6.3: CatBoost Model on Approach A(ASUO modality)

6.1.2 Approach B

This section holds all the models that make use of Approach B using ASUO dataset.

Random Forest

Figure 6.4 represents the classification report of the Random forest model whose data uses approach B on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 1st and whose AUC score ranks 2nd amongst all models using the ASUO dataset.

```
RandomForestClassifier(max_depth=140, n_estimators=1000, random_state=42)

 Classification report :
              precision    recall  f1-score   support

           0       0.82      0.70      0.76       161
           1       0.74      0.85      0.79       160

    accuracy                           0.78       321
   macro avg       0.78      0.78      0.77       321
weighted avg       0.78      0.78      0.77       321

Accuracy   Score :  0.7757009345794392
Area under curve :  0.8364130434782608
```

Figure 6.4: Random Forest Model on Approach B(ASUO modality)

XGBoost

Figure 6.5 represents the classification report of the XGBoost model whose data uses approach B on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 6th and whose AUC score ranks 4th amongst all models using the ASUO dataset.

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)

 Classification report :
               precision    recall  f1-score   support

           0       0.73      0.67      0.70       161
           1       0.70      0.76      0.72       160

    accuracy                           0.71       321
   macro avg       0.72      0.71      0.71       321
weighted avg       0.72      0.71      0.71       321

Accuracy   Score :  0.7133956386292835
Area under curve :  0.823835403726708
```

Figure 6.5: XGBoost Model on Approach B(ASUO modality)

CatBoost

Figure 6.6 represents the classification report of the CatBoost model whose data uses approach B on the ASUO dataset. There are a total of 1282 students in the dataset post oversampling using SMOTE, who take the online modality of the calculus course.

This model generates a report whose accuracy ranks 2nd and whose AUC score ranks 1st amongst all models using the ASUO dataset.

```
<catboost.core.CatBoostClassifier object at 0x7f810ebbf050>

 Classification report :
               precision    recall  f1-score   support

           0       0.79      0.70      0.74       161
           1       0.73      0.81      0.77       160

    accuracy                           0.76       321
   macro avg       0.76      0.76      0.76       321
weighted avg       0.76      0.76      0.76       321

Accuracy   Score :  0.7570093457943925
Area under curve :  0.8495729813664595
```

Figure 6.6: CatBoost Model on Approach B(ASUO modality)

6.2 F2F - Offline Dataset

This section holds all the models on the F2F Dataset.

6.2.1 Approach A

This section holds all the models that make use of Approach A using F2F dataset.

Random Forest

Figure 6.7 represents the classification report of the Random Forest model whose data uses approach A on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 3rd and whose AUC score ranks 3rd amongst all models using the F2F dataset.

```
RandomForestClassifier(max_depth=140, n_estimators=1000, random_state=42)

 Classification report :
              precision    recall  f1-score   support

        fail       0.88      0.82      0.85       325
        pass       0.84      0.89      0.86       325

    accuracy                           0.86       650
   macro avg       0.86      0.86      0.86       650
weighted avg       0.86      0.86      0.86       650

Accuracy   Score :  0.8584615384615385
Area under curve :  0.9297041420118344
```

Figure 6.7: Random Forest Model on Approach A(F2F modality)

XGBoost

Figure 6.8 represents the classification report of the XGBoost model whose data uses approach A on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 2nd and whose AUC score ranks 2nd amongst all models using the F2F dataset.

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)

 Classification report :
               precision    recall  f1-score   support

        fail       0.88      0.85      0.87       325
        pass       0.86      0.88      0.87       325

    accuracy                           0.87       650
   macro avg       0.87      0.87      0.87       650
weighted avg       0.87      0.87      0.87       650

Accuracy   Score :  0.8676923076923077
Area under curve :  0.9330840236686391
```

Figure 6.8: XGBoost Model on Approach A(F2F modality)

CatBoost

Figure 6.9 represents the classification report of the XGBoost model whose data uses approach A on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 1st and whose AUC score ranks 1st amongst all models using the F2F dataset.

```
<catboost.core.CatBoostClassifier object at 0x7fb33f1dfd90>

 Classification report :
              precision    recall  f1-score   support

        fail       0.89      0.86      0.88       325
        pass       0.87      0.90      0.88       325

    accuracy                           0.88       650
   macro avg       0.88      0.88      0.88       650
weighted avg       0.88      0.88      0.88       650

Accuracy   Score :  0.8784615384615385
Area under curve :  0.9406674556213017
```

Figure 6.9: CatBoost Model on Approach A(F2F modality)

6.2.2 Approach B

This section holds all the models that make use of Approach B using F2F dataset.

Random Forest

Figure 6.10 represents the classification report of the Random Forest model whose data uses approach B on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 6th and whose AUC score ranks 6th amongst all models using the F2F dataset.

```
RandomForestClassifier(max_depth=140, n_estimators=1000, random_state=42)

 Classification report :
              precision    recall  f1-score   support

           0       0.83      0.78      0.80       325
           1       0.79      0.85      0.82       325

    accuracy                           0.81       650
   macro avg       0.81      0.81      0.81       650
weighted avg       0.81      0.81      0.81       650

Accuracy   Score :  0.8107692307692308
Area under curve :  0.8925112426035503
```

Figure 6.10: Random Forest Model on Approach B(F2F modality)

XGBoost

Figure 6.11 represents the classification report of the XGBoost model whose data uses approach B on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 5th and whose AUC score ranks 5th amongst all models using the F2F dataset.

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)

 Classification report :
               precision    recall  f1-score   support

           0       0.85      0.84      0.84       325
           1       0.84      0.85      0.85       325

    accuracy                           0.84       650
   macro avg       0.84      0.84      0.84       650
weighted avg       0.84      0.84      0.84       650

Accuracy   Score :  0.8446153846153847
Area under curve :  0.921192899408284
```

Figure 6.11: XGBoost Model on Approach B(F2F modality)

CatBoost

Figure 6.12 represents the classification report of the XGBoost model whose data uses approach B on the F2F dataset. There are a total of 2598 students in the dataset post oversampling using SMOTE, who take the offline modality of the calculus course.

This model generates a report whose accuracy ranks 4th and whose AUC score ranks 4th amongst all models using the F2F dataset.

```
 Classification report :
              precision    recall  f1-score   support

           0       0.86      0.84      0.85       325
           1       0.84      0.86      0.85       325

    accuracy                           0.85       650
   macro avg       0.85      0.85      0.85       650
weighted avg       0.85      0.85      0.85       650

Accuracy   Score :  0.8476923076923077
Area under curve :  0.9256899408284023
```

Figure 6.12: CatBoost Model on Approach B(F2F modality)

Chapter 7

ANALYSIS OF RESULTS

The final dataset contains, 3266 students out of which 1195 were ASU Online(ASUO) and 2071 were Face-2-Face(F2F) students. A total of 12 models were developed, 6 for each class of students (ASUO and F2F), using Random Forest, XGBoost, and CatBoost models using the two approaches mentioned in the method section. The week in which the dataset was accumulated and the models were build was determined using the course length for ASUO and F2F classes. ASUO classes were typically 7.5 weeks long which meant an ideal week for early predictions would be week 3, which is slightly under the halfway course, whereas for F2F classes the ideal week chosen was week 6.

7.1 Evaluation Metrics

The metrics mentioned below are the main metrics upon which the overall analysis was made.

7.1.1 Precision:

Precision is the ratio of correctly predicted positive resources to the total predicted positive resources. High precision relates to the low false-positive rate.

$$Precision = TP/TP + FP$$

where TP is True Positive and FP is False Positive

7.1.2 Recall:

Recall is the ratio of correctly predicted positive resources to all resources in the actual class.

$$Recall = TP/TP + FN$$

7.1.3 F1 Score:

F1 Score is the weighted average of Precision and Recall. This score takes both false positives and false negatives into account. F1 score is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar costs. If this is not the case it's better to look at both Precision and Recall.

$$F1 = 2 * (Recall * Precision)/(Recall + Precision)$$

7.1.4 Area under curve(AUC) score:

AUC - Receiver Operating Characteristics(ROC) curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting the classes as is. By analogy, the higher the AUC, the better the model is at distinguishing between students who pass or fail their calculus course.

7.1.5 Accuracy:

Accuracy is the most understandable performance measure and it is simply a ratio of correctly predicted resources to the total resources.

$$Accuracy = TP + TN/TP + FP + FN + TN$$

7.2 Approach A

The results of approach A can be viewed in table 7.1. This table provides an overview of the scores obtained by each model based on the evaluation metrics. Based on the results it can be determined that the CatBoost model for pre-processing A on F2F and ASUO provides the best predictions. They have the highest Precision, recall, F1 score, and accuracy of their bunch, which falls within our evaluation metrics.

Table 7.1: Evaluation Metrics for Class Fail on Models with Approach A

Class Fail				
Pre-Processing A Models	Precision	Recall	F1 Score	Overall Accuracy
ASUO -Random Forest	0.76	0.67	0.71	0.725
ASUO -XGBoost	0.77	0.66	0.71	0.729
ASUO -CatBoost	0.76	0.69	0.72	0.735
F2F -Random Forest	0.88	0.82	0.85	0.858
F2F -XGBoost	0.88	0.85	0.87	0.867
F2F -CatBoost	0.80	0.86	0.88	0.878

Figure 7.1 and figure 7.2 represent the model fit using the Area Under the Curve Graph for ASUO and F2F respectively. With respect to AUC scores, CatBoost

performs the best amongst all the other models for both ASUO and F2F. Thus the CatBoost models achieve the best model fit as well on approach A. Figure 7.3 and figure 7.4 show us the most important features out of the best performing models. Transfer, prev_term_gpa, and incoming_gpa are the top three most common features that contribute to prediction making for the best performing CatBoost model.

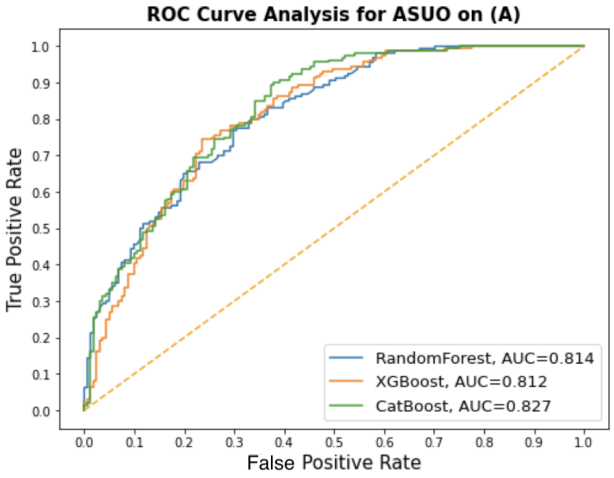

Figure 7.1: ROC Curve Best Model on Approach A(ASUO modality)

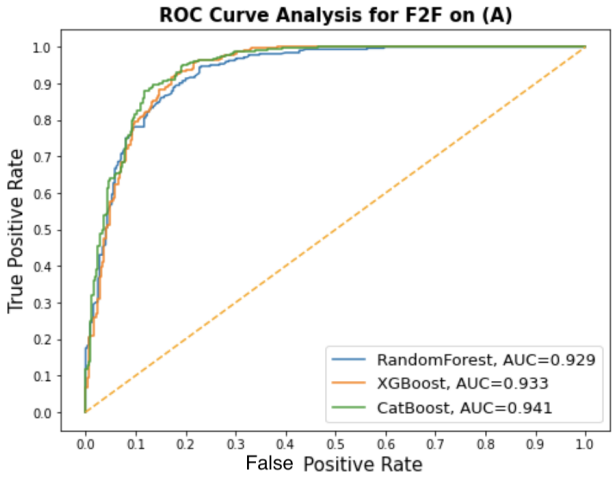

Figure 7.2: ROC Curve Best Model on Approach A(F2F modality)

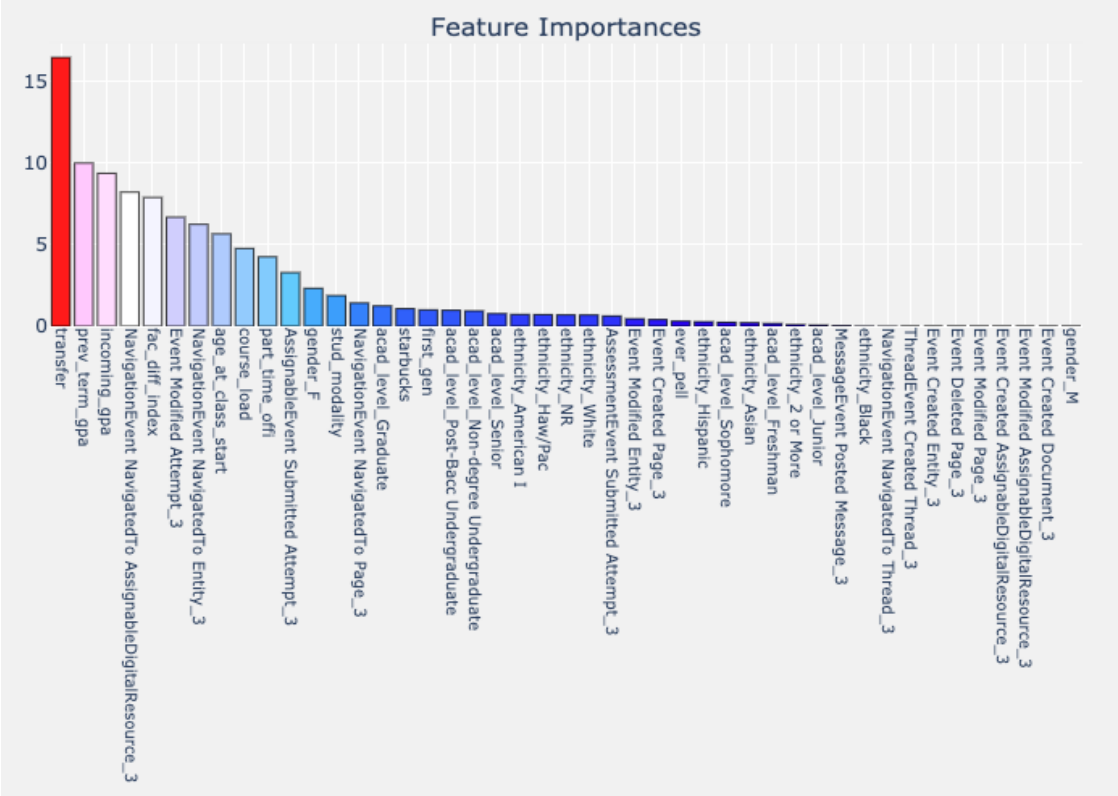

Figure 7.3: Feature Importance for CatBoost Model on ASUO for Approach A

7.3 Approach B

The results of approach B can be viewed in table 7.2. This table provides an overview of the scores obtained by each model based on the evaluation metrics. Based on the results it can be determined that the CatBoost model for approach A on F2F provides the best predictions. They have the highest Precision, recall, F1 score, and accuracy of their bunch, which falls within our evaluation metrics. In the case of ASUO, the best results were observed from the random forest model with the best scores on precision, recall, F1 scores as well as their accuracies. Figure 7.5 represents the model fit using the Area Under the Curve Graph for ASUO, where the random forest model falls short in the AUC score whereas the CatBoost model on approach B for the ASUO dataset shines.

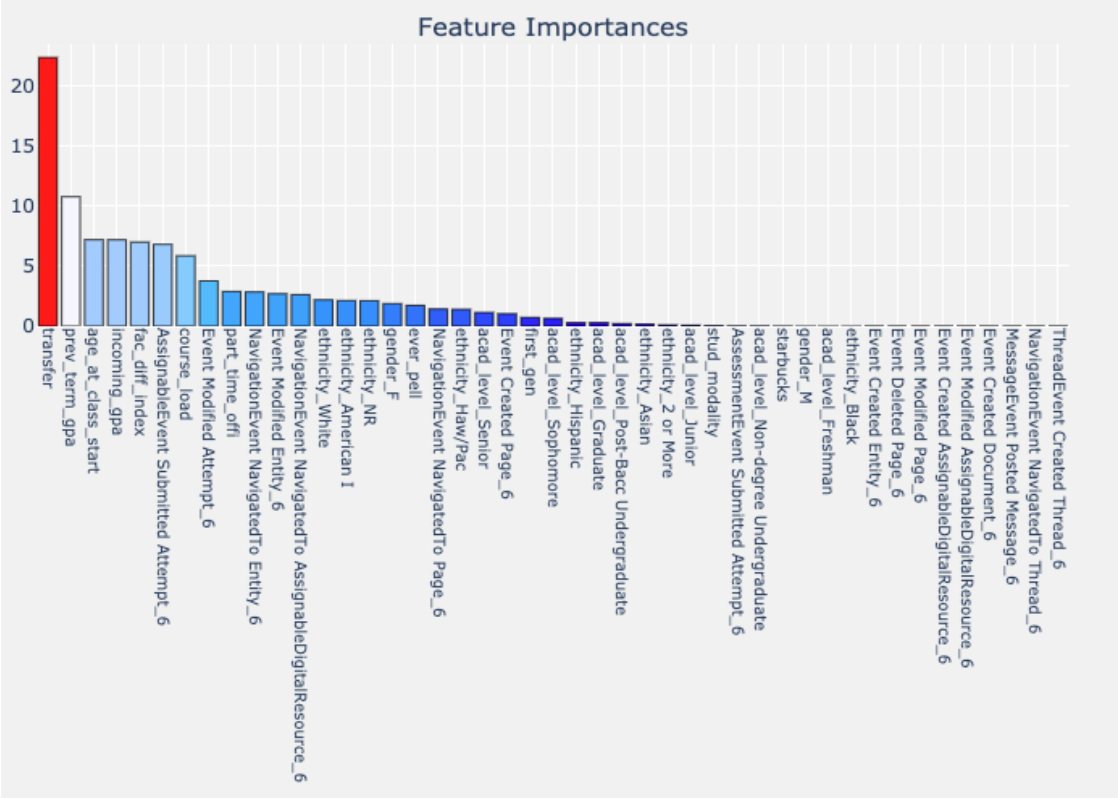

Figure 7.4: Feature importance for CatBoost Model on F2F for Approach A

Table 7.2: Evaluation metrics for class fail on models with Approach B

Class Fail				
Pre-Processing B Models	Precision	Recall	F1 Score	Overall Accuracy
ASUO -Random Forest	0.82	0.70	0.76	0.775
ASUO -XGBoost	0.73	0.67	0.70	0.713
ASUO -CatBoost	0.79	0.70	0.74	0.757
F2F -Random Forest	0.83	0.78	0.80	0.811
F2F -XGBoost	0.85	0.84	0.84	0.845
F2F -CatBoost	0.86	0.84	0.85	0.848

Figure 7.6 represents the model fit using the Area Under the Curve Graph for the F2F dataset. With respect to the AUC scores, CatBoost performs the best amongst all the other models for both ASUO and F2F. Thus the CatBoost model achieves the best model fit as well. As are 369 features that will be tough to represent with a feature importance graph.

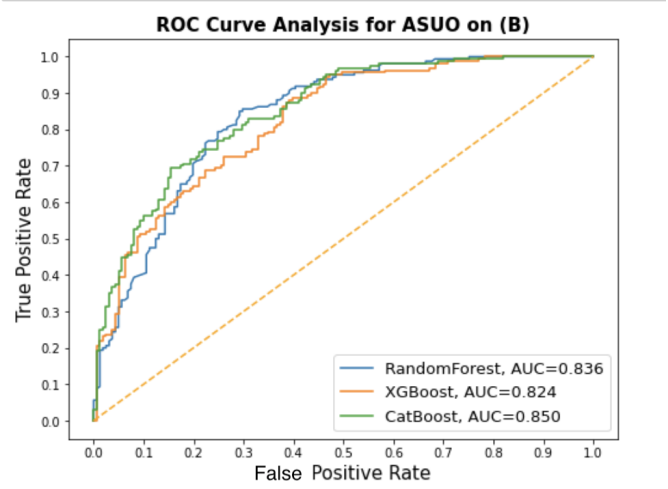

Figure 7.5: ROC Curve Best Model on Approach B(ASUO modality)

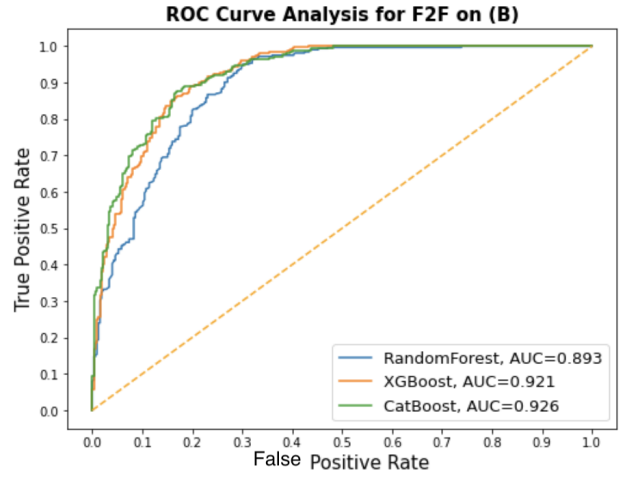

Figure 7.6: ROC Curve Best Model on Approach B(F2F modality)

7.4 Best Model Analysis

Table 7.3 depicts the model accuracies for all the models generated. From the results of table 7.3, we can infer that the CatBoost model with approach A performs the best on the F2F dataset. Similarly, the Random Forest model on approach A performs the best on the ASUO dataset. The results were arrived at by focusing on our evaluation metrics for the class Fail for each model as the use-case for this thesis is to identify AT-Risk students. Approach A on the CatBoost model provided the best score for the Face-2-Face Model with an accuracy of 87.8% and an AUC score of 0.941. Further analysis with respect to the precision, recall, and F1 scores also corroborate with the findings. Approach B on the Random Forest model provided the best score for the ASU Online Model with an accuracy of 77.5%. Further analysis with respect to the precision, recall, and F1 scores also corroborate with the findings.

Table 7.3: Accuracy for All Models

Pre-Processing Model	Random Forest	XGBoost	CatBoost
Approach A on ASUO(A)	72.5%	72.8%	73.5%
Approach A on F2F(A)	85.8%	86.7%	87.8%
Approach B on ASUO(B)	77.5%	71.3%	74.1%
Approach B on F2F(B)	81.1%	84.4%	84.8%

Chapter 8

CONCLUSIONS AND FUTURE WORK

8.1 Conclusion

This Thesis compares 2 datasets(ASUO and F2F) with 2 pre-processing techniques(A and B) on 3 machine learning models(Random Forest, XGBoost, CatBoost) to determine which is the best model for their respective dataset. The thesis concludes that the CatBoost model with approach A performed best on the F2F dataset, and the Random Forest model with approach B performed best on the ASUO dataset. They each contributed the best evaluation metrics to solve the problem statement which is to provide an early warning system to detect students at risk. Furthermore due to the inconsistency of the details feature present in the Canvas Activity dataset approach B was developed to handle it.

Pre-processing A avoids the usage of the inconsistent details column and generates a frequency count for each student in a weekly fashion. Pre-processing B makes use of the details column by subjecting the entire dataset to two Natural Language base processing which are: (i) Part of speech tagging (ii) Facebook FastText word embeddings. The dataset is further split into two types: (i) Asu Online dataset where the entire class span is 7.5 weeks (ii) Face-2-Face classes where the class span is 15 weeks. Due to the nature of the problem, there is a class imbalance with the target columns for each dataset ie. students who pass and students who fail. In order to overcome this issue oversampling of the minority class as a technique was used. This was employed using SMOTE which oversamples the minority class. Further, it was decided that for early detection week 3 for ASUO and week 6 for the F2F dataset

provided the ideal outcomes for the problem statement. These datasets were then subjected to modeling to generate 12 models.

The thesis was not able to arrive at a single approach as the best to model both the ASUO and F2F datasets on. The reason for this lies in the nature of the datasets. ASUO dataset is rich in canvas activities due to the online modality of the class. But F2F dataset which is the offline modality of the course has much lower counts for canvas activity when compared to the ASUO dataset. Also, the size of the ASUO dataset half when compared to the size of the F2F dataset.

The nature of approach A is to determine the classes based on the gap in the activity frequency. This gap is much more pronounced in the F2F dataset due to its large size. This claim can be corroborated by analyzing the AUC scores for the models built on approach A for the F2F datasets. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the higher the AUC, the better the model is at distinguishing between students who pass or fail their calculus course. Thus the best model for predicting the F2F dataset was the CatBoost model on approach A with an accuracy of 87.8% and an AUC score of 0.941.

The nature of approach B is to analyze the activity sequence being generated by the student. The output of the Natural Language preprocessing on the activity sequence will always be a word embedding vector of size 300. Hence approach B concentrates on the pattern for student success whereas approach A concentrates on the gap between successful and unsuccessful students. Thus the best model for predicting the F2F dataset was the CatBoost model on approach A with an accuracy of 77.5% and an AUC score of 0.836.

Furthermore, the Canvas activity dataset shows us the validity and its applicability

in the area of predicting student success. This thesis has presented sample findings investigating the potential for LMS (Canvas) activity data variables to be used as predictors and lead indicators of student academic success early in a course. We are mindful that correlations do not necessarily indicate causality but the significance of this study lies not in causation but in correlation.

8.2 Future Direction

There are a couple of enhancements that will be explored in the future. Some of them are described below:

- Hyper-parameter Tuning: After deciding to fix on a single machine learning ensemble, a proper K-fold cross validation can be evaluated on that ensemble. This will ensure that the model chosen will be hyper-parameter tuned to achieve the best result possible.
- Additional LMS Data: Combinations of various other LMS datasets along with the activity dataset used in this research can be explored for the creation of more robust models. Data points such as gradebook data as well as the login frequency may provide additional support from which the model may adapt better.
- One Model for All Weeks: The current preprocessing technique makes use of data from the start till the week of prediction. Therefore based on this form on data pruning a separate model will be required to run predictions for every week throughout the duration of the course. Converting multiple weekly model into a single model to run all weeks might be production viable and may improve the overall accuracy of the system

REFERENCES

Baker, R. S. and K. Yacef, "The state of educational data mining in 2009: A review and future visions", JEDM— Journal of Educational Data Mining **1**, 1, 3–17 (2009).

Blackboard, "Blackboard app: Classroom and learning application.", URL `https://www.blackboard.com/teaching-learning/learning-management` (2015).

Brownlee, J., "A gentle introduction to xgboost for applied machine learning", URL `https://rb.gy/a6ukr6` (2021).

Campbell, J., C. Finnegan and B. Collins, "Academic analytics: Using the cms as an early warning system", in "WebCT impact conference", (2006).

Campbell, J. P., P. B. DeBlois and D. G. Oblinger, "Academic analytics: A new tool for a new era", EDUCAUSE review **42**, 4, 40 (2007).

Chawla, N. V., K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique", Journal of artificial intelligence research **16**, 321–357 (2002).

Chen, X., "Students who study science, technology, engineering, and mathematics (stem) in postsecondary education. stats in brief. nces 2009-161.", National Center for Education Statistics (2009).

Cui, Y., F. Chen and A. Shiri, "Scale up predictive models for early detection of at-risk students: a feasibility study", Information and Learning Sciences (2020).

Dutt, A. and M. A. Ismail, "Can we predict student learning performance from lms data? a classification approach", in "3rd International Conference on Current Issues in Education (ICCIE 2018)", pp. 24–29 (Atlantis Press, 2019).

Dutt, A., M. A. Ismail and T. Herawan, "A systematic review on educational data mining", Ieee Access **5**, 15991–16005 (2017).

Facebook, "Facebook - fasttext", URL `https://fasttext.cc/docs/en/faqs.html` (2017).

Ghorpade, T. and L. Ragha, "Featured based sentiment classification for hotel reviews using nlp and bayesian classification", in "2012 International Conference on Communication, Information & Computing Technology (ICCICT)", pp. 1–5 (IEEE, 2012).

Hooshyar, D., M. Pedaste and Y. Yang, "Mining educational data to predict students' performance through procrastination behavior", Entropy **22**, 1, 12 (2020).

Inc., L. S., "Learning management system - learning platform - absorb lms software.", URL `https://www.absorblms.com/` (2015).

Instructure, "A learning platform that's loved by everyone.", URL https://www.instructure.com/canvas (2016).

Kadoić, N. and D. Oreški, "Analysis of student behavior and success based on logs in moodle", in "2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)", pp. 0654–0659 (IEEE, 2018).

Kho, J., "Why random forest is my favorite machine learning model", URL https://rb.gy/ikf95c (2018).

Ki, Y., E. Kim and H. K. Kim, "A novel approach to detect malware based on api call sequence analysis", International Journal of Distributed Sensor Networks **11**, 6, 659101 (2015).

Kondo, N., M. Okubo and T. Hatanaka, "Early detection of at-risk students using machine learning based on lms log data", in "2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)", pp. 198–201 (IEEE, 2017).

Kupczynski, L., A. M. Gibson, P. Ice, J. Richardson and L. Challoo, "The impact of frequency on achievement in online courses: A study from a south texas university.", Journal of Interactive Online Learning **10**, 3 (2011).

Lee, S. L. A., A. Z. Kouzani and E. J. Hu, "Random forest based lung nodule classification aided by clustering", Computerized medical imaging and graphics **34**, 7, 535–542 (2010).

Macfadyen, L. P. and S. Dawson, "Mining lms data to develop an "early warning system" for educators: A proof of concept", Computers & education **54**, 2, 588–599 (2010).

Maor, D. and S. Volet, "Interactivity in professional online learning: A review of research based studies", Australasian journal of educational technology **23**, 2 (2007).

Moodle, "Moodle - open-source learning platform.", URL https://moodle.org/ (2012).

Morris, L. V., C. Finnegan and S.-S. Wu, "Tracking student behavior, persistence, and achievement in online courses", The Internet and Higher Education **8**, 3, 221–231 (2005).

Nagano, Y. and R. Uda, "Static analysis with paragraph vector for malware detection", in "Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication", pp. 1–7 (2017).

NLTK, "Categorizing and tagging words", URL http://www.nltk.org/book/ch05.html (2019).

Pilgrim, M., *Concepts for calculus intervention: measuring student attitudes toward mathematics and achievement in calculus, A*, Ph.D. thesis, Colorado State University (2010).

Pistilli, M. D. and K. E. Arnold, "Purdue signals: Mining real-time academic data to enhance student success", About Campus **15**, 3, 22–24 (2010).

Pongpaichet, S., S. Jankapor, S. Janchai and T. Tongsanit, "Early detection at-risk students using machine learning", in "2020 International Conference on Information and Communication Technology Convergence (ICTC)", pp. 283–287 (IEEE, 2020).

Ray, S., "Catboost: Catboost categorical features", URL `https://rb.gy/3401mc` (2017).

Razno, M., "Machine learning text classification model with nlp approach", Computational Linguistics and Intelligent Systems **2**, 71–73 (2019).

Seymour, E. and N. M. Hewitt, *Talking about leaving* (Westview Press, Boulder, CO, 1997).

Shin, D. and J. Shim, "A systematic review on data mining for mathematics and science education", International Journal of Science and Mathematics Education pp. 1–21 (2020).

Siemens, G. and P. Long, "Penetrating the fog: Analytics in learning and education.", EDUCAUSE review **46**, 5, 30 (2011).

Steen, L. A., *Calculus for a New Century: A Pump, Not a Filter. Papers Presented at a Colloquium (Washington, DC, October 28-29, 1987). MAA Notes Number 8.* (ERIC, 1988).

Tran, T. K. and H. Sato, "Nlp-based approaches for malware classification from api sequences", in "2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES)", pp. 101–105 (IEEE, 2017).

Villegas-Ch, W., M. Román-Cañizares and X. Palacios-Pacheco, "Improvement of an online education model with the integration of machine learning and data analysis in an lms", Applied Sciences **10**, 15, 5371 (2020).

Wang, A. Y. and M. H. Newlin, "Predictors of performance in the virtual classroom: Identifying and helping at-risk cyber-students", THE Journal (Technological Horizons In Education) **29**, 10, 21 (2002).

Wolff, A., Z. Zdrahal, D. Herrmannova, J. Kuzilek and M. Hlosta, "Developing predictive models for early detection of at-risk students on distance learning modules", (2014).

Worthley, M., *Mixed methods explanatory study of the failure/drop rate for freshman STEM calculus students, A*, Ph.D. thesis, Colorado State University (2013).

APPENDIX A

CODE

```
frame2 = frame2.fillna(0)
frame2['emplid'] = pd.to_numeric(frame2['emplid'], errors='coerce').dropna()
frame2['emplid'].fillna(0)


result2 = pd.merge(frame2, res, on='emplid')
result2['eventtime'] = pd.to_datetime(result2['eventtime'], format='%Y/%m/%d %H:%M:%S').dt.date
result2['start_dt'] = pd.to_datetime(result2['start_dt'], format='%Y-%m-%d')



df2 = result2['eventtype'].map(str) + ' ' + result2['action'].map(str) + ' ' + result2['object_type'].map(str)
result2 =  pd.concat([result2, df2], axis=1).reindex(result2.index)



keyarr2 = ['NavigationEvent NavigatedTo Entity',
       'NavigationEvent NavigatedTo Thread', 'Event Modified Entity',
       'AssignableEvent Submitted Attempt', 'Event Modified Attempt',
       'NavigationEvent NavigatedTo AssignableDigitalResource',
       'NavigationEvent NavigatedTo Page',
       'AssessmentEvent Submitted Attempt', 'MessageEvent Posted Message',
       'Event Created Document',
       'Event Modified AssignableDigitalResource',
       'ThreadEvent Created Thread',
       'Event Created AssignableDigitalResource', 'Event Modified Page',
       'Event Deleted Page', 'Event Created Entity', 'Event Created Page']
emparray2 = result2['emplid'].unique()
cols2 = keyarr2
```

Figure A.1: Data Preprocessing for Approach A - 1

```
limit = len(keyarr2)
for j in range(0,11):
    colmn = ['emplid','start_dt']
    for q in range (0,limit):
        string = cols2[q] + '_'+str(j+1)
        colmn.append(string)
    df_new = pd.DataFrame(columns = colmn)
    for i in emparray2:
        emplid_grade = result2[result2['emplid'] == i]
        All = df_All[df_All['emplid'] == i]
        apparr = []
        dayzs = (j+1)*7
        apparr.append(i)
        apparr.append(All['start_dt'].iloc[0])
        for p in range(0,limit):
                apparr.append(0)
        if i in emparray2:
            basedate = emplid_grade['start_dt'] + timedelta(days= 0)
            date1 = emplid_grade['start_dt'] + timedelta(days= dayzs)
            date2 = emplid_grade['eventtime']

            basedated = basedate.iloc[0]
            value1 = date1.iloc[0]
            value2 = date2.iloc[0]

            for index, value in emplid_grade.iterrows():
                if(basedated <= value['eventtime'] <= value1):
                    action = emplid_grade.at[index, 0]
                    stringg = str(action) + '_'+str(j+1)
                    if(stringg in colmn):
                        ind = colmn.index(stringg)
                    else:
                        continue
                    apparr[ind] = apparr[ind] + 1
        df_yolo = pd.DataFrame(data=[apparr], columns = colmn )
        df_new = pd.concat([df_new,df_yolo])
    df_new['start_dt'] = pd.to_datetime(df_new['start_dt'], format='%Y-%m-%d')
    namer = 'PreprocessingA-Click' +'_'+str(j+1)+'.csv'
    df_new.to_csv(namer)
```

Figure A.2: Data Preprocessing for Approach A - 2

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
#merging and other preprocessing with SIS dataset
def runner():
    df_All = pd.read_csv('../Data/Mat265_SIS_Trim.csv', sep=',')
    keyarr = [
        'NavigationEvent NavigatedTo Entity',
       'NavigationEvent NavigatedTo Thread', 'Event Modified Entity',
       'AssignableEvent Submitted Attempt', 'Event Modified Attempt',
       'NavigationEvent NavigatedTo AssignableDigitalResource',
       'NavigationEvent NavigatedTo Page',
       'AssessmentEvent Submitted Attempt', 'MessageEvent Posted Message',
       'Event Created Document',
       'Event Modified AssignableDigitalResource',
       'ThreadEvent Created Thread',
       'Event Created AssignableDigitalResource', 'Event Modified Page',
       'Event Deleted Page', 'Event Created Entity', 'Event Created Page']

    emparr = df_All['emplid'].unique()
    arr = {

    }
    for i in emparr:

        if i in arr:
            arr[i] = arr[i] + 1
        else:
            arr[i] = 1

    Xer = []
    yer = []
    flist = []

    for i in range (1,12):

        namer2 = 'FitNewweeksClick' +'_'+str(i)+'.csv'
        df_new2 = pd.read_csv(namer2, sep=',')
        pd.set_option('display.max_columns', None)
        df_new2['start_dt'] = pd.to_datetime(df_new2['start_dt'], format='%Y-%m-%d')
        df_All['start_dt'] = pd.to_datetime(df_All['start_dt'], format='%Y-%m-%d')
        df = pd.merge(df_All, df_new2, on = ['emplid', 'start_dt'])

        for j in arr1:
            df = df[df['strm'] != j]
        df = df.drop(['acad_plan', 'act_eng_max','act_math_max','admit_term' ,'admit_type', 'aid_year', 'asu_new_underg


        df = df.fillna(0)
        for index, value in df.iterrows():
            if(df.at[index,'cgrade'] < 2):
                df.at[index,'fail'] = 0
            else:
                df.at[index,'fail'] = 1
        categorical_cols = ['acad_level','ethnicity','gender']
        df = df.fillna(0)
        numerical_cols = ['age_at_class_start','course_load','fac_diff_index','incoming_gpa','prev_term_gpa']

        binary_cols = ['ever_pell','first_gen','part_time_offi','starbucks','transfer','fail']
        target_col = ["fail"]
        df = df.drop(['Unnamed: 0', 'cgrade'], axis=1)
        df = df.drop(['emplid',"start_dt"], axis = 1)

        le = LabelEncoder()

        for col in binary_cols :
            df[col] = le.fit_transform(df[col])
        scaler = StandardScaler()

        df[numerical_cols] = scaler.fit_transform(df[numerical_cols])
        df = pd.get_dummies(data=df, columns=['acad_level','ethnicity','gender'])


        feature_list = list(df.columns)
        feature_list.remove("fail")
        flist.append(feature_list)
        X = df[[col for col in df.columns if col not in target_col]]
        y = df['fail']

        Xer.append(X)
        yer.append(y)
        d.append(df)
    return Xer, yer, flist,d
```

Figure A.3: Data Preprocessing for Approach A - 3

```
frame2 = frame2.fillna(0)
frame2['emplid'] = pd.to_numeric(frame2['emplid'], errors='coerce').dropna()
frame2['emplid'].fillna(0)


result2 = pd.merge(frame2, res, on='emplid')
result2['eventtime'] = pd.to_datetime(result2['eventtime'], format='%Y/%m/%d %H:%M:%S').dt.date
result2['start_dt'] = pd.to_datetime(result2['start_dt'], format='%Y-%m-%d')



df2 = result2['eventtype'].map(str) + ' ' + result2['action'].map(str) + ' ' + result2['object_type'].map(str)+" "+resu
result2 =  pd.concat([result2, df2], axis=1).reindex(result2.index)



keyarr2 = ['NavigationEvent NavigatedTo Entity',
       'NavigationEvent NavigatedTo Thread', 'Event Modified Entity',
       'AssignableEvent Submitted Attempt', 'Event Modified Attempt',
       'NavigationEvent NavigatedTo AssignableDigitalResource',
       'NavigationEvent NavigatedTo Page',
       'AssessmentEvent Submitted Attempt', 'MessageEvent Posted Message',
       'Event Created Document',
       'Event Modified AssignableDigitalResource',
       'ThreadEvent Created Thread',
       'Event Created AssignableDigitalResource', 'Event Modified Page',
       'Event Deleted Page', 'Event Created Entity', 'Event Created Page']
emparray2 = result2['emplid'].unique()
cols2 = keyarr2

import re
for index,value in result2.iterrows():
    string = value[0]
    strarr = re.findall('[A-Z][^A-Z]*', string)
    strr = " ".join(strarr)
#     print(strr)
    result2.at[index,"result"] = strr
```

Figure A.4: Data Preprocessing for Approach B - 1

```
limit = len(keyarr2)
for j in range(0,11):
    colmn = ['emplid','start_dt',"result"]

    df_new = pd.DataFrame(columns = colmn)
    for i in emparray2:
        emplid_grade = result2[result2['emplid'] == i]
        All = df_All[df_All['emplid'] == i]
        apparr = []
        dayzs = (j+1)*7
        apparr.append(i)
        apparr.append(All['start_dt'].iloc[0])
        apparr.append("")
        if i in emparray2:
            basedate = emplid_grade['start_dt'] + timedelta(days= 0)
            date1 = emplid_grade['start_dt'] + timedelta(days= dayzs)
            date2 = emplid_grade['eventtime']

            basedated = basedate.iloc[0]
            value1 = date1.iloc[0]
            value2 = date2.iloc[0]

            for index, value in emplid_grade.iterrows():
                if(basedated <= value['eventtime'] <= value1):
                    action = emplid_grade.at[index, "result"]
                    apparr[2] = apparr[2] +" " +str(action)
        df_yolo = pd.DataFrame(data=[apparr], columns = colmn )
        df_new = pd.concat([df_new,df_yolo])
    df_new['start_dt'] = pd.to_datetime(df_new['start_dt'], format='%Y-%m-%d')
    namer = 'ResultweeksClick' +'_'+str(j+1)+'.csv'
    df_new.to_csv(namer)
```

Figure A.5: Data Preprocessing for Approach B - 2

```
from collections import Counter
from itertools import chain
import sister
embedder = sister.MeanEmbedding(lang="en")
import pandas as pd

from nltk import word_tokenize, pos_tag


Xer = []
yer = []
flist = []
random = []
xgb = []

d = []
for i in range (1,12):
    namer2 = 'ResultweeksClick' +'_'+str(i)+'.csv'
    #         df_new = pd.read_csv(namer, sep=',')
    df_new2 = pd.read_csv(namer2, sep=',')
    pd.set_option('display.max_columns', None)

    #         df_new['start_dt'] = pd.to_datetime(df_new['start_dt'], format='%Y-%m-%d')
    df_new2['start_dt'] = pd.to_datetime(df_new2['start_dt'], format='%Y-%m-%d')
    df_new2 = df_new2.fillna("nothing")

    tok_and_tag = lambda x: pos_tag(word_tokenize(x))
    print("hello")
    df_new2['lower_sent'] = df_new2['result'].apply(str.lower)
    df_new2['tagged_sent'] = df_new2['lower_sent'].apply(tok_and_tag)

    possible_tags = sorted(set(list(zip(*chain(*df_new2['tagged_sent'])))[1]))

    def add_pos_with_zero_counts(counter, keys_to_add):
        for k in keys_to_add:
            counter[k] = counter.get(k, 0)
        return counter

    print("judo")
    # Detailed steps.
    df_new2['pos_counts'] = df_new2['tagged_sent'].apply(lambda x: Counter(list(zip(*x))[1]))
    df_new2['pos_counts_with_zero'] = df_new2['pos_counts'].apply(lambda x: add_pos_with_zero_counts(x, possible_tags))
    df_new2['sent_vector'] = df_new2['pos_counts_with_zero'].apply(lambda x: [count for tag, count in sorted(x.most_com

    # All in one for pos tagging and one hot encoding
    df_new2['sent_vector'] = df_new2['tagged_sent'].apply(lambda x:
        [count for tag, count in sorted(
            add_pos_with_zero_counts(
                Counter(list(zip(*x))[1]),
                        possible_tags).most_common()
             )
        ]
    )
    print('bulbul')
    df2 = pd.DataFrame(df_new2['sent_vector'].tolist())
    df2.columns = possible_tags

    df_new2 = df_new2.join(df2, how='outer')
    df_new2 = df_new2.drop(['lower_sent','tagged_sent','pos_counts',"pos_counts_with_zero","sent_vector"], axis=1)
    #for fasttext embedding
    for index,value in df_new2.iterrows():
        sentence = df_new2.at[index,"result"]
        vector = embedder(sentence)
        for ind,i in enumerate(vector):
            df_new2.at[index,str(ind)] = i
    df_All = pd.read_csv('../Data/Mat265_SIS_Trim.csv', sep=',')
    df_new2['start_dt'] = pd.to_datetime(df_new2['start_dt'], format='%Y-%m-%d')
    df_All['start_dt'] = pd.to_datetime(df_All['start_dt'], format='%Y-%m-%d')
    # #         df1 = pd.merge(df_new, df_new2, on = ['emplid', 'start_dt'])
    df = pd.merge(df_All, df_new2, on = ['emplid', 'start_dt']) #merge with SIS dataset
    for j in arr1:
        df = df[df['strm'] != j]
    df = df.drop(['acad_plan', 'act_eng_max','act_math_max','admit_term' ,'admit_type', 'aid_year', 'asu_new_undergrad'


    df = df.fillna(0)
    for index, value in df.iterrows():
        if(df.at[index,'cgrade'] < 2):
            df.at[index,'fail'] = 0
        else:
            df.at[index,'fail'] = 1
    categorical_cols = ['acad_level','ethnicity','gender']
    df = df.fillna(0)
    numerical_cols = ['age_at_class_start',
         'course_load',

         'fac_diff_index',
         'incoming_gpa',
         'prev_term_gpa']

    binary_cols = [
         'ever_pell',
         'first_gen',
         'part_time_offi',
         'starbucks',
         'transfer',
         'fail']
    target_col = ["fail"]
    df = df.drop(['Unnamed: 0', 'cgrade'], axis=1)
    df = df.drop(['emplid',"start_dt","result"], axis = 1)

    le = LabelEncoder()

    for col in binary_cols :
        df[col] = le.fit_transform(df[col])
    scaler = StandardScaler()

    df[numerical_cols] = scaler.fit_transform(df[numerical_cols])
    df = pd.get_dummies(data=df, columns=['acad_level','ethnicity','gender'])


    X = df[[col for col in df.columns if col not in target_col]]
    y = df['fail']

    Xer.append(X)
    yer.append(y)
    d.append(df)
```

Figure A.6: Data Preprocessing for Approach B - 3

```
def model(X,y,feature_list, index):
    # Using Skicit-learn to split data into training and testing sets
    randomforest = []
    xgboost = []
    result_table = pd.DataFrame(columns=['classifiers', 'fpr','tpr','auc'])
    from sklearn.model_selection import train_test_split

    # Split the data into training and testing sets
    train_features, test_features, train_labels, test_labels = train_test_split(X, y, test_size = 0.25,stratify = y,
                                                                                random_state = 42)

    # Import the model we are using
    from sklearn.ensemble import RandomForestClassifier

    # Instantiate model
    rf = RandomForestClassifier(n_estimators= 1000, max_features= 'auto', max_depth= 140 ,random_state = 42,criterion =

    # Train the model on training data
    rf.fit(train_features,train_labels)
    e,f,g = prediction(rf,train_features, test_features, train_labels, test_labels, feature_list,"features",threshold_p
    randomforest.append(rf.score(test_features,test_labels))

    result_table = result_table.append({'classifiers':"RandomForest",
                                        'fpr':e,
                                        'tpr':f,
                                        'auc':g}, ignore_index=True)

    from xgboost import XGBClassifier
    xgc = XGBClassifier()
    a,b,c = prediction(xgc,train_features, test_features, train_labels, test_labels, feature_list,"features",threshold_
    xgboost.append(xgc.score(test_features,test_labels))
    result_table = result_table.append({'classifiers':"XGBoost",
                                        'fpr':a,
                                        'tpr':b,
                                        'auc':c}, ignore_index=True)
    model1 = CatBoostClassifier(learning_rate=0.03,
                           custom_metric=['Logloss',
                                          'AUC:hints=skip_train~false'])

    model1.fit(train_features,
              train_labels,
              eval_set=(test_features, test_labels),
              verbose=False)
    h, i, j = prediction(model1,train_features, test_features, train_labels, test_labels, feature_list,"features",thres
    result_table = result_table.append({'classifiers':"CatBoost",
                                        'fpr':h,
                                        'tpr':i,
                                        'auc':j}, ignore_index=True)

    return rf.score(test_features,test_labels),xgc.score(test_features,test_labels),result_table
```

Figure A.7: All Models

```
#FINAL ASUO

index = 2


#         print(feature_list)
dff = d[index]
check = dff[dff["stud_modality"] == "ASUO"]
X = check[[col for col in check.columns if col not in target_col]]
y = check['fail']
feature_list =  list(X.columns)
New = X.drop(["stud_modality"], axis = 1)
oversample = SMOTE(random_state = 42)
New, y = oversample.fit_resample(New, y)
print(y.shape[0])
#     print(New)
r, x,dfa = model(New,y,feature_list,index)
```

Figure A.8: Runner for All Models on ASUO dataset at Week 3

```
#FINAL F2F

index = 5


#         print(feature_list)
dff = d[index]
check = dff[dff["stud_modality"] == "F2F"]
X = check[[col for col in check.columns if col not in target_col]]
y = check['fail']
feature_list =  list(X.columns)
print(y.shape[0])
New = X.drop(["stud_modality"], axis = 1)
oversample = SMOTE(random_state = 42)
New, y = oversample.fit_resample(New, y)
print(y.shape[0])
#     print(New)
r, x,dfg = model(New,y,feature_list,index)
```

Figure A.9: Runner for All Models on F2F dataset at Week 6