

# Clustering Students Using Pre-Midterm Behaviour Data and Predict Their Exam Performance

Anonymous  
Anonymous Institution  
anonymous@anonymous.edu

## ABSTRACT

Student behaviour should correlate to the course performance. This paper explored different types of clustering algorithms using the pre-midterm student behaviour data. We found meaningful and interpretive results when clustering algorithms generate 3 clusters. The clusters can be briefly summarized as potential top performance (PTP) students, potential poor performance (PPP) students, and mixed performance (MP) students. We found that PTP students usually submit early and gain a high score, PPP students usually submit late and gain a low score, and MP students usually make most submissions. MP students are hard to cluster. However, we found a good connection between other students' behaviour and performance if we leave out MP students.

## Keywords

Computer Education, Clustering, Student Behaviour, Auto-grading System, Student Performance

## 1. INTRODUCTION

Students expose different learning behaviours in programming courses. Several studies focused on analysing the students' data in order to understand student behaviours [8, 1, 20, 16]. The clustering technique was one of the common methodologies used in such studies. There are several benefits to identifying students' behaviours. First, groups of students with similar academic and behaviour characteristics would benefit from the same intervention, which can reduce the time for instructors to identify and implement the right intervention for individual students [17]. Second, a better understanding of students' misconceptions can lead to a better support system for novice programmers and to provide adaptive feedback for students [6]. Last but not least, it is also a common technique used to predict student performance [10, 26, 23].

Although clustering techniques can be a valuable tool for

researchers to categorize different student behaviours, several studies only applied a single type of clustering algorithm in their experiments [6, 21, 4, 12]. However, clustering results can be affected by the wrong choice of the clustering algorithm, which can cause the result under the threat of validity. To address this issue, people should experiment with multiple clustering algorithms to confirm that different clustering algorithms produce similar results where the essential characteristics of the resulting clusters are identical. Our experiment confirms that clusters' essential characteristics are the same across multiple clustering algorithms, forming further discussion foundations.

Many studies applied clustering techniques to predict student performances [14, 11, 22]. However, the data used in these studies generally include both the pre-midterm data and the post-midterm data. Because of a strong correlation between the midterm exam grades and the final exam grades [3, 10, 13], which is also true for our data (Pearson correlation of 0.81), it suggests that students who failed the midterm are likely to fail the final exam. Therefore, it is critical to identify at-risk students before the midterm. Our study merged multiple clustering algorithms into a predictive model to predict students' performance using pre-midterm data.

This study applied the clustering techniques to students' behaviours exposed in their pre-midterm submissions in an auto-grading system, Marmoset [24], to categorize them into different clusters. We applied multiple clustering techniques and compared their results to remove any effects caused by the wrong choice of the clustering algorithm. Then we tried to predict students' performance using these clustering techniques. The research questions we want to ask are:

- **RQ1:** Are different clustering algorithms producing different results?
- **RQ2:** What are the characteristics of students in different clusters?
- **RQ3:** What were the exam grades of students in different clusters?

## 2. RELATED WORK

In computing education field, clustering techniques were used in many studies to understand student behaviours. Several studies were targeting to understand students code [21, 15,

6]; while other studies were trying to use forum data [14, 4] or Learning Management System (LMS) data [11, 22, 17, 12] to better understand students.

Auto-grading systems were used in several universities [9, 5, 24]. Sometimes it is also integrated into the LMS [19]. The data collected from an auto-grading system can be considered a combination of LMS and coding data. Because auto-grading systems usually provide instant feedback to students [24, 5, 9], how students interact with such systems may provide new insights into their behaviours. This study used the LMS-like data from the auto-grading system, Marmoset [24].

In addition to understanding student behaviours, it is also important to connect student behaviours with their performance. Although classification techniques are used more frequently than clustering techniques [10], several studies explored the possibility to predict student performance using clustering techniques [14, 11, 22]. For instance, López et al. [14] compared several classification algorithms against clustering algorithms. They put clustering results into classes by finding the minimum-error mapping between clusters to classes. They found that student participation in the course forum was a good predictor of the final marks for the course. Our study also connects students' behaviour to their performance, finding promising results.

### 3. COURSE BACKGROUND

The data we used in the study is collected from an introduction-level programming course in an R1 university in Canada. Students were supposed to learn how to carry out operational tasks using the C and C++ languages, perform procedural and object-oriented programming, and other relevant programming knowledge. The preliminary experiments were based on the data of 130 students who attended both the midterm and final exams (they all had grades above zero for both exams) in section 3 of 2018 (but we will share some additional results using the 2017 data later). Programming knowledge was not required to take this course, and students were randomly put into different sections.

In the course, an auto-grading system, Marmoset [24], was used. We refer to a coding/programming question in Marmoset as a *task*. A task may contain multiple test cases. For any task, a student can make multiple submissions against it. Marmoset will automatically test it for each submission and reveal some test results to the student. Students can thus learn some feedback from those test results and then improve or fix their code accordingly.

In the course, there were different types of coding questions.

- i) **Coding Labs:** coding labs took place in the lab room. There was exactly one assigned task for each coding lab, which was to be completed and submitted during the lab period (2 hours in the morning). Before the midterm exam, the coding lab was scheduled weekly. After the midterm exam, the coding lab was scheduled biweekly. There was a corresponding extended deadline (the same date but in the evening). Some students may rely on that deadline rather than finish during the lab.

- ii) **Homework Assignments:** homework assignments were assigned for students to do at home. They were assigned during lab time. Before the midterm, homework problems were due the following week. After the midterm, homework problems were due approximately two weeks later. In every homework assignment, there were multiple tasks, all of which had the same deadline.

- iii) **Coding Examination:** there was an in-lab coding examination during the course. It was similar to a coding lab. However, its grade comprised a portion of the midterm grade. An extended deadline was also allowed for the coding examination.

## 4. EXPERIMENT AND RESULTS

### 4.1 Features

The auto-grading system Marmoset stored every student's submission during the course. In our study, we extracted three features.

- **passrate:** for every Marmoset task, we calculated the best score (the best number of tests passed among the submissions) a student made before the task deadline. Then we divide the total number of tests of that task to form the *passrate* feature. Because every assignment had multiple tasks, so for assignments, we need to sum the tasks' best scores to form the best score for an assignment, and we sum tasks' total number of tests to form the total number of tests for an assignment. For example, assignment 1 had 4 tasks and a total number of 54 tests. If a student's best submissions of that 4 tasks passed 6, 6, 7, and 8 tests separately, then the passrate of that student for assignment 1 will be  $(6 + 6 + 7 + 8)/54 = 0.5$ . This process was not needed for coding labs since there was only one task in a coding lab. Table 1 shows the total number of tests for different assignments and coding labs.
- **lastsub:** we extracted the *lastsub* feature as how many minutes between the last submission a student made and the task deadline. If a student made any submission before the deadline, this feature would contain a non-zero value. Only if a student did not submit before the deadline can the value be zero. Note that this feature will be zero for students whose submissions met the extended deadline but did not meet the original deadline for coding labs.
- **nsub:** the *nsub* feature represents how many submissions a student made before the task deadline for a given task. For assignments, we summed the numbers of submissions of tasks to form the *nsub* of an assignment.

The features used in clustering algorithms is summarized in Table 2.

### 4.2 RQ1: What are the clustering results from different clustering algorithms given the pre-midterm data?

assignment	# total tests	coding lab	# total tests
a1	54	l1	8
a2	85	l2	19
a3	73	l3	19
a4	87	l4	19

**Table 1: Total number of tests for different assignments and coding labs pre-midterm, “a” stands for homework assignment, “l” stands for coding lab**

feature name
a*_passrate
a*_lastsub
a*_nsub
l*_passrate
l*_lastsub
l*_nsub

**Table 2: Features,  $*$   $\in \{1, 2, 3, 4\}$ , “a” stands for homework assignment, “l” stands for coding lab. Only pre-midterm data were used**

For research question 1, we explored all types of the clustering algorithms provided in the scikit-learn python package [18]. It includes: *K-Means*, *Affinity Propagation*, *Spectral Clustering*, *Hierarchical Clustering*, and *Density-based Spatial Clustering*. We standardize every feature by removing the mean and scaling to unit variance before clustering. For a sample  $x$ , the standard score is calculated as:

$$z = \frac{x - u}{s}$$

where  $u$  is the mean of the samples and  $s$  is the standard deviation of the samples.



We tested different options for setting the number of clusters in different algorithms. We found that setting the number to 3 will give us good interpretive results. In this section, we will only compare the labelling. For the characteristics of different clusters, we will discuss them in RQ2.

#### 4.2.1 K-Means

The K-Means algorithm will try to separate samples in groups of equal variance, minimizing a criterion known as the *inertia* or *within-cluster sum-of-squares*.

#### 4.2.2 Affinity Propagation

The affinity propagation algorithm will create clusters by sending messages between pairs of samples until convergence. The advantage of using it is that it does not require the number-of-cluster parameter set in advance. It will group points to the same cluster if they have the same exemplar point.

Although there is no explicit hyper-parameter for setting the number of clusters, a “preference” hyper-parameter affects the number of clusters. We tested it with different settings and found that the result can be well interpreted when the grouped number of clusters is 3.

#### 4.2.3 Hierarchical Clustering

Hierarchical clustering represents a general family of clustering algorithms that build nested clusters by merging or splitting them successively. This hierarchy of clusters is represented as a tree (or dendrogram). The tree’s root is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample. Here we used agglomerative clustering algorithm [28], which will recursively merge the pair of clusters that minimally increases a given linkage distance. The number of clusters we set was 3. The linkage method is “ward”, which minimizes the merged clusters’ variance.

#### 4.2.4 Spectral Clustering

Spectral clustering algorithm [27] performs a low-dimension embedding of the affinity matrix between samples, followed by clustering, e.g., by K-Means, of the components of the eigenvectors in the low dimensional space. We tried different settings for constructing the affinity matrix. The clustering algorithm will give an interpretive result when we use Gaussian (aka RBF) kernel to construct the affinity matrix.

#### 4.2.5 Density-Based Spatial Clustering

The density-based spatial clustering algorithm [7] views clusters as areas of high density separated by areas of low density. The implementation we used in our experiment was *OPTICS* [2]. We used the “cosine” similarity as the distance.

#### 4.2.6 Comparing cluster results

Table 3 presents the size of different clusters using different clustering algorithms. To compare the cluster results across different clustering algorithms, we used *adjusted rand index* [25] for evaluation. Random labelling samples will make the adjusted rand index close to 0.0, and the value will be exactly 1.0 when the clustering results are identical. Table 4 presents the results. To save space, we make abbreviate notations as KM for “K-means”, AP for “Affinity Propagation”, HC for “Hierarchical Clustering”, SC for “Spectral Clustering”, DBSC for “Density-Based Spatial Clustering”. Note that a cluster may be given different labels in different clustering algorithms. We re-numbered them according to the findings in the discussion of RQ2. First, we calculated each student’s passrate mean, lastsub mean, and nsub mean. Then we calculated the mean of each feature’s means in each cluster. We labelled the cluster with largest lastsub and passrate means as cluster 1, cluster with smallest lastsub and passrate means as cluster 3. The rest were labelled as cluster 2.

	KM	AP	HC	SC	DBSC
cluster 1	62	56	48	58	32
cluster 2	59	65	75	63	89
cluster 3	9	9	7	9	9

**Table 3: The size of different clusters in different clustering algorithms**


From table 3 and table 4, we can tell the clustering results from K-Means, Affinity Propagation, and Spectral Clustering are similar to each other (similar cluster sizes and high

first algorithm	second algorithm	adjusted rand index
KM	AP	0.780665
AP	SC	0.730717
KM	SC	0.682500
KM	HC	0.557372
HC	DBSC	0.521486
HC	SC	0.515544
AP	HC	0.475727
AP	DBSC	0.414938
SC	DBSC	0.382402
KM	DBSC	0.322811

**Table 4: The adjusted rand index results (high to low)**

adjusted rand index value). In contrast, Hierarchical Clustering and Density-Based Spatial Clustering produced different clustering results.

### 4.3 RQ2: What are the characteristics of students in different clusters?

We carefully examined students in different clusters. Interestingly, although the cluster results differed in the cluster size and the adjusted rand index  we found that students in different clusters share similar characteristics across different clustering algorithms.

#### 4.3.1 Potential-Top-Performance (PTP) students

Students who are clustered into this cluster (cluster 1) are those students whose best scores were very high for pre-midterm assignments (the median passrates of them are 96%-98%, the median is a range because different clustering algorithms may produce different median), and they tend to submit their last submission early for those assignments (median: 2-3 days early). The number of submissions they made for assignments was roughly 25-29 (median). Students in this cluster also achieved a very high score for almost all coding labs (median: 95%-100% passrates). The number of submissions they made was 3-6 (median). For their last submissions, because the first coding lab was due the next day and was used to familiarize students with Marmoset, we need to consider it separately from other coding labs, which have a 2-hour time limit. Students in this cluster generally finished it on the same day when it was assigned (median: 14-16 hours early). For other regular coding labs, the time varied. For lab 2 and lab 3, the last submission was made mostly half-hour early (median: 30-58 minutes). For lab 4, the last submission was made mostly 1 hour early (median: 70-77 minutes). Currently, it is not clear why this situation happens. However, we noticed that lab 2 and lab 3 asked students to use a loop to read in command line inputs; meanwhile, they needed to validate the inputs and return an error if there was an invalid value. We guess the loop usage forced students to spend more time thinking about the potential complex edge cases. Therefore, their last submission may not be as early as other coding labs.

#### 4.3.2 Potential-Poor-Performance (PPP) students

Students that are clustered into this cluster (cluster 3) are those students whose best scores passed only 34%-66% tests (median) for pre-midterm assignments, and they tend to

submit their last submission late for those assignments (median: 2-6 hours early). The number of submissions they made for assignments was 8-21 (median). Students in this cluster achieved an even lower score for coding labs, passing 25% tests on average and 0% as the median. For the first coding lab, the last submissions made by students in this cluster were mainly 7 minutes ahead of the deadline (median: 7-14 minutes). It was even less (median: 0-2 minutes) for other regular coding labs. The number of submissions they made for coding labs was 0-4 (median).

#### 4.3.3 Mixed-Performance (MP) students

Students clustered into this cluster (cluster 2) are those whose essential characteristics can be summarized as “Middle-top performance on assignments, not so stable performance on labs; the last submission was made early but not that early”. Their best scores passed 85%-95% tests as the median for pre-midterm assignments, and they tend to submit their last submission early for those assignments (median: 16-27 hours early). The number of submissions they made for assignments was 26-42 (median). For coding labs, the number of submissions they made for most coding labs was 4-6 (median), with an exception for lab 3, for which the number was 1-2. Students in this cluster achieved good scores on lab 1 and lab 4 (around 95% as averages, 100% as the median). However, they only passed 52%-74% tests (median) for lab2. For lab 3, they only passed 0%-53% tests as medians. For the first coding lab, the last submissions made by students in this cluster were 3-4 hours (medians) ahead of the deadline. By looking at the actual number, we found the majority of them finished it on the second day, which is also the due day, before the deadline. For lab 2 and lab 3, the last submission was primarily made no more than 15 minutes early. For lab 4, the last submission was made 42-54 minutes early (medians). We think students in this cluster can get middle-top performance on almost all pre-midterm assignments because there was much time before an assignment was due. They could solve any difficulty by spending more time working on it. However, when the time was limited, and the question became difficult, their performance decreased, observed from lab 2 and lab 3.

Table 5 summarizes the characteristics of different students.

We also examined the students that were put into different clusters from different algorithms.

- *PTP vs PPP*: We found that no student that was clustered into the PTP cluster in one algorithm was clustered into the PPP cluster in another algorithm, which means the difference between the two clusters were huge enough that all clustering algorithms we explored in this paper were able to capture it.
- *PTP vs MP*: There were 36 students who were clustered into the PTP cluster in one algorithm and then clustered into the MP cluster in another algorithm. By looking at the average, we found these 36 students made the last submission early, but not as early as PTP students, while not as late as most MP students. They achieved satisfying assignment performance and lab performance, which was not as good as PTP students, while not as poor as most MP students. The



	PTP students	MP students	PPP students
assignment passrate	96% – 98%	85% – 95%	34% – 66%
assignment lastsub	2 – 3 days early	16 – 27 hours early	2 – 6 hours
assignment nsub	25 – 29	26 – 42	8 – 21
lab 1-4 passrate	95% – 100%	lab 1,4: 100%, lab 2: 52% – 74%, lab 3: 0% – 53%	0%
lab 1 lastsub	14 – 16 hours early	3 – 4 hours early	7 – 14 minutes early
lab 2,3 lastsub	30 – 58 minutes early	0 – 15 minutes early	0 – 2 minutes early
lab 4 lastsub	70 – 77 minutes early	42 – 54 minutes early	0 – 2 minutes early
lab 1-4 nsub	3 – 6	lab 1,2,4: 4 – 6, lab 3: 1 – 2	0 – 4

**Table 5: Characteristics of different clusters.** Because there are multiple clustering algorithms, we calculated the medians of students of different clustering algorithms and then combined them into ranges. The lastsub feature of the coding lab 1 was treated specially due to the deadline difference, while lab 4 was due to the significant time differences observed.

number of submissions they made for assignments was between the number of submissions made by PTP students and that of most MP students. However, their number of submissions for coding labs was more than both MP students and PTP students.

- *PPP vs MP:* There are 3 students who were clustered into the PPP cluster in one algorithm and then clustered into the MP cluster in another algorithm. We notice they made their last submission earlier than the average of PPP students for some assignments and labs, but not as early as most MP students. They had at least one assignment in which their performance fell below the average performance of PPP students. They performed well on lab 1 and 4, but not lab 2 and 3, which were more difficult since PTP students also needed to spend more time on these two labs. For assignments, the average number of submissions they made was fewer than the average number of submissions made by PPP students and most MP students. For labs, that number is between the average number of submissions made by PPP students and most MP students.

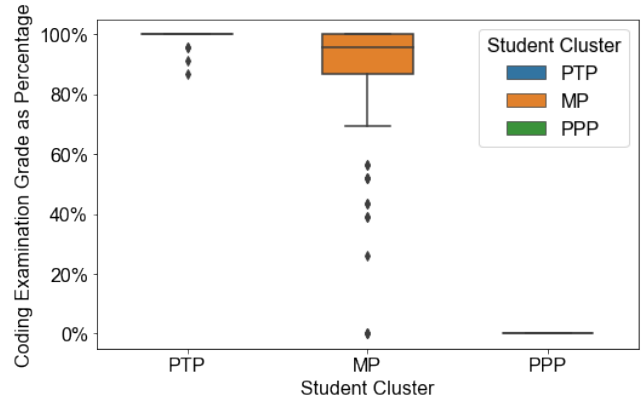
In general, we can consider those students put into different clusters from different algorithms as the students whose behaviour was in the middle of PTP students and MP students or the middle of PPP students and MP students. Different clustering algorithms may consider the weight of each feature differently. Therefore, some inconsistencies may exist between algorithms. However, since there were not any students put into the PTP cluster in one algorithm while being put into the PPP cluster in another algorithm, we can tell that students from these two clusters share no typical behaviour from any aspect.

#### 4.4 RQ3: What were the exam grades of students in different clusters?

Because some students were put into different clusters from different clustering algorithms, we consider students in the PTP cluster only if they were put into the PTP cluster by all clustering algorithms. Similarly, students in the PPP cluster were only put into the PPP cluster by all clustering algorithms. The remaining students will be MP students.

In addition to the midterm grades and final exam grades, there was a coding examination grade, which will comprise

a portion of the midterm grade. Figure 1 shows us the relation between the coding examination grades and different clusters. Figure 2 shows us the relation between midterm grades and final grades of different clusters.

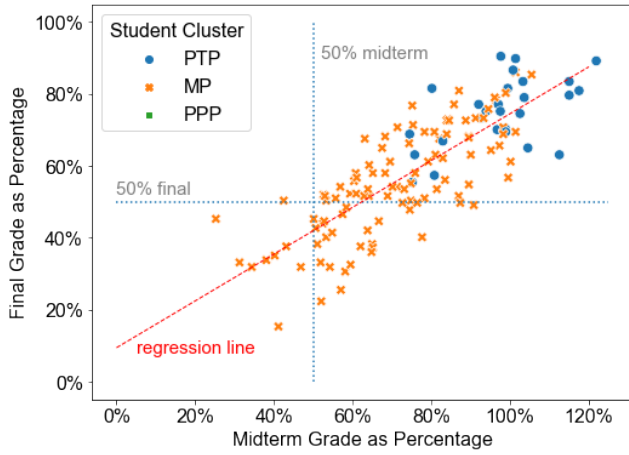


**Figure 1: The box-plot of coding examination grades.** PPP students all got zero while PTP students mostly got 100%.

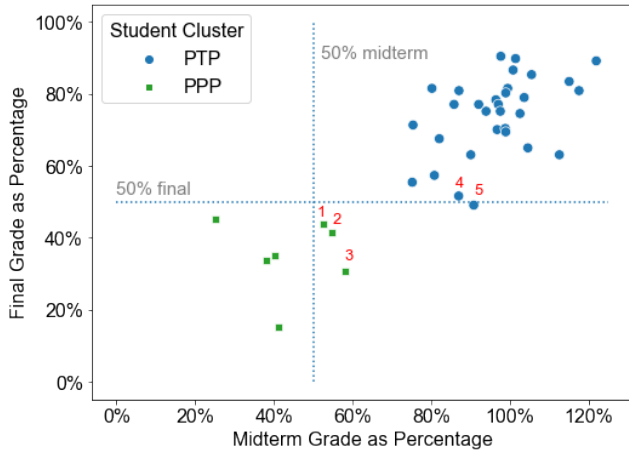
From Figure 1, we can see PTP students achieved a very high score on the coding examination, while PPP students achieved 0% on the coding examination. The result is expected since PTP students mostly performed well on assignments and labs, of which they were programming questions. It is reasonable that they achieved a high score on the coding examination. In contrast, for PPP students, since they performed poorly on those questions, it is not surprising that they got a significantly low score.

From 2, we can see the performances of MP students messed up with other clusters of students. However, if we exclude them, as shown in Figure 3, we can see the performances of PTP students and PPP students were completely different.

It is essential to consider how many students genuinely need help from the identified PPP students. In other words, precision is important [3]. It is the higher, the better. Suppose we consider students' midterm grades and use 50% as the threshold. Then we can calculate the precision of the PPP students, which is  $4/7 = 0.57$  (4 students out of the total 7 PPP students were below 50% in the midterm). Similarly, the precision of the final grades is 1.0, since no PPP students



**Figure 2: Midterm Grades and Final Exam Grades (as percentage).** The total grade for midterm was 120% because there were bonus questions.



**Figure 3: Midterm Grades and Final Exam Grades (as a percentage).** The total grade for the midterm was 120% because there were bonus questions.

had a final grade above 50%. These results are promising, implying that the clustering technique can predict student performance.

In addition, we looked into the students who had midterm performance between 50% and 60% (labelled as 1, 2, and 3 in Figure 3). We found that the student 1 and student 2 made early submissions for some assignments before the midterm, which may explain why they got a slightly higher than 50% grade in the midterm. We did not see anything special from the student 3 that makes him/her different from other PPP students. It might explain why he got roughly 60% on the midterm, but later only roughly 30% grade on the final.

We also looked into the two students who got roughly 85% on the midterm while later only got roughly 50% on the final (labelled as 4 and 5 in Figure 3). We found they had a significant performance drop on assignments and labs when

the concept of object-orient programming was involved. It might indicate they faced many difficulties understanding those concepts.

## 4.5 Additional Results

We want to discuss some additional insights in this section.

### 4.5.1 Behaviour consistency before and after the midterm

It is natural to think about whether students behave consistently before and after the midterm.

We applied Welch's t-test [29] against the pre-midterm data (using the absolute feature values) and the post-midterm data cluster by cluster and algorithm by an algorithm (in total 5 algorithms, 3 clusters each). Then we checked the  $p$  values (significant level as 0.05) to see if the data was consistent (equal means). A  $p$  value larger than or equal 0.05 indicates the data was consistent, while a  $p$  value less than 0.05 indicates the data was not consistent. Table 6 shows us the Welch's t-test results of assignments and Table 7 shows use the Welch's t-test results of coding labs.

From Table 6, we can see that for PTP students and MP students, none of their behaviour was consistent for assignments. Since the assignments were assigned bi-weekly after midterm and the difficulties and the number of tests of assignments increased, it is reasonable that PTP students' and MP students' behaviours on assignments became not consistent. By looking at the median and the mean values, we think the trend is *the last submission submitted by them was not as early as they did before the midterm, even though they had one more week to work on it; the number of submissions increased; their assignment performances decreased*.

From Table 7, we can see that for coding labs, both PTP students and MP students tended to make more submissions for post-midterm labs, which causes the number of submissions to be inconsistent with the number before the midterm. Also, they both tended to submit not as early as they did before the midterm. However, for how much percentage of tests was passed, it is interesting that MP students had similar performances as before the midterm, while PTP students' performance decreased.

Again from Table 6 and Table 7, we can see for PPP students, their performances on both assignments and labs were consistent pre-midterm and post-midterm. They also submitted late consistently for both assignments and labs (Except for PPP students from the DBSC clustering algorithm, which has a value of 0.04. However, it is still close to 0.05). They made a consistent number of submissions for assignments, but they slightly ( $p = 0.02$ ) tended to make more submissions for labs post-midterm than pre-midterm (Except for PPP students from the HC clustering algorithm, which has a value of 0.07. However, it is still close to 0.05). It seems students clustered into this cluster were not trying to change their behaviour much.

### 4.5.2 Applying the same clustering technique in a different year

We want to share our results about how this clustering process performs using the year 2017 data. Especially, we want

algorithm	feature	PTP	MP	PPP
AP	a*_percentage	***5.32e - 18	***1.47e - 23	0.11
	a*_lastsub	***2.32e - 15	***1.15e - 05	0.30
	a*_nsub	***4.97e - 10	**7.48e - 03	0.92
DBSC	a*_percentage	***9.86e - 12	***5.43e - 28	0.13
	a*_lastsub	***1.64e - 10	***4.03e - 10	*0.04
	a*_nsub	***7.56e - 13	***1.99e - 05	0.92
HC	a*_percentage	***1.01e - 14	***3.10e - 22	0.37
	a*_lastsub	***4.31e - 14	***4.07e - 07	0.24
	a*_nsub	***2.50e - 09	***7.85e - 05	0.99
KM	a*_percentage	***4.25e - 19	***2.42e - 20	0.11
	a*_lastsub	***6.45e - 16	***5.85e - 05	0.30
	a*_nsub	***8.05e - 09	**1.24e - 03	0.92
SC	a*_percentage	***4.81e - 17	***3.72e - 23	0.11
	a*_lastsub	***1.82e - 14	***3.38e - 06	0.30
	a*_nsub	***9.11e - 12	**1.51e - 03	0.92

Table 6: The  $p$  values of the Welch’s t-test results of pre-midterm and post-midterm student behaviour data (assignments)

algorithm	feature	PTP	MP	PPP
AP	l*_percentage	*3.13e - 02	4.38e - 01	0.23
	l*_lastsub	***2.16e - 08	***1.75e - 06	0.61
	l*_nsub	***1.77e - 07	***5.69e - 07	*0.02
DBSC	l*_percentage	**7.82e - 03	8.34e - 01	0.14
	l*_lastsub	***9.49e - 06	***3.53e - 09	0.42
	l*_nsub	***1.36e - 05	***7.72e - 10	*0.02
HC	l*_percentage	*1.68e - 02	5.18e - 01	0.17
	l*_lastsub	***2.14e - 07	***1.07e - 07	0.71
	l*_nsub	***5.45e - 06	***8.94e - 09	0.07
KM	l*_percentage	*1.76e - 02	3.12e - 01	0.23
	l*_lastsub	***1.04e - 09	***4.86e - 05	0.61
	l*_nsub	***6.21e - 08	***1.62e - 06	*0.02
SC	l*_percentage	*2.30e - 02	3.57e - 01	0.23
	l*_lastsub	***1.92e - 08	***2.38e - 06	0.61
	l*_nsub	***1.37e - 07	***4.49e - 07	*0.02

Table 7: The  $p$  values of the Welch’s t-test results of pre-midterm and post-midterm student behaviour data (labs)

to see the connections between students’ behaviour and their performances. In other words, we primarily focused on the RQ3 for the year 2017 data.

In 2017, students were split into different sections based on their programming experience. Section 1 students (235 students) were students with prior programming experience, while section 2 and 3 students (131 students) were without prior programming experience.

The setup in Marmoset was greatly improved in the year 2018 compared to the year 2017. Figure 4 shows us the total number of tests for assignments before the midterm. Also, there were no coding labs in the year 2017.

We used 5 assignments for section 1 students and 4 for section 2&3 students. They were all pre-midterm assignments.

Figure 5 and Figure 6 show us the results of the RQ3 for year 2017 students.

Table 8 shows the precision of the PPP students. It indicates

year	section	midterm	final
2017	1	0.33	0.66
2017	2&3	0	<b>1.0</b>
2018	3	<b>0.57</b>	<b>1.0</b>

Table 8: The precisions of potential-poor-performance students in different years (the higher the better)

the precision will be higher in final grades than midterm grades for both years. The observation might not be compelling due to the small number of PPP students in 2017. However, the consistency of the observation for both years indicates that our clustering technique can predict students’ final exam performance using the pre-midterm data.

In addition, Table 8 shows that the precision is highest for both the midterm and final grades in the year 2018. We consider it because the setup of Marmoset was better than in 2017, and there were additional coding labs.

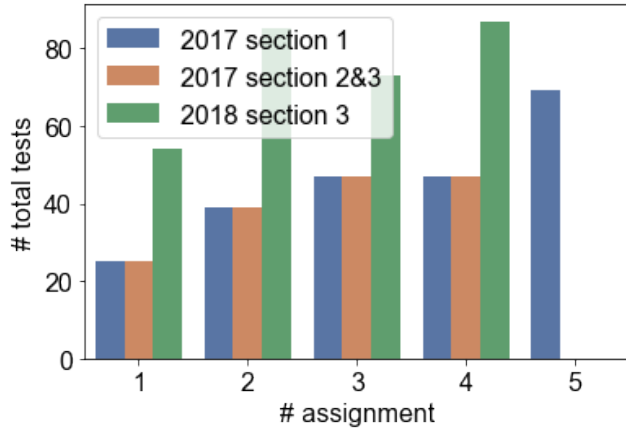


Figure 4: The total number of tests of assignments in different year before the midterm. Section 1 students in year 2017 had one more assignment which was the 5<sup>th</sup> assignment.

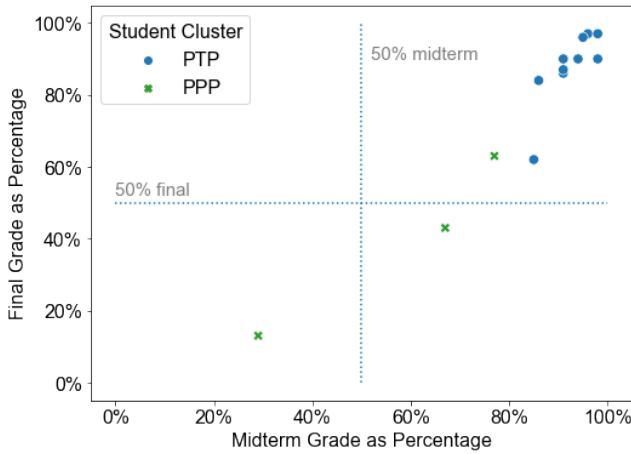


Figure 5: Midterm Grades and Final Exam Grades (as a percentage) for section 1 students in year 2017.

## 5. DISCUSSION

The benefit of using clustering techniques over classification techniques is that clustering does not need training data. It is handy when it is a new course or if the course structure changes. Also, data split is not necessary. It is not doable for classification techniques when the data size is small, but it is doable for clustering techniques. Lastly, the clustering process can be conducted repeatedly over time. It is worth exploring how the clustering results change over time. Our study identified that midterm time is a good time point. It is even better if students can be identified even earlier.

We noticed that PPP Students typically would not change their behaviour after the midterm. Although from some clustering algorithms, they seem to make more submissions. However, they still ended up with poor performances. For those students, an early intervention that can change their learning behaviour should be helpful.

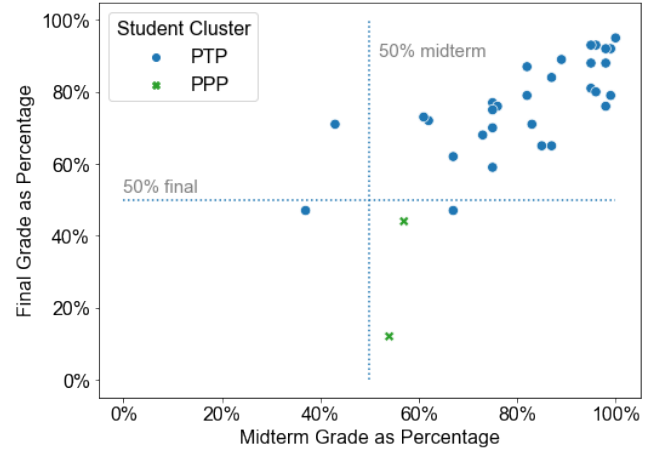


Figure 6: Midterm Grades and Final Exam Grades (as a percentage) for section 2&3 students in year 2017.

## 6. LIMITATIONS

The limitation of this study was that the data used in the study was of a limited amount. We appreciate any replicate studies to help validate the results in our study.

## 7. CONCLUSION

This study applied clustering techniques on pre-midterm students' behaviour data, namely how early students make their last submissions, how many submissions they make, and the best score. We found that although different clustering algorithms may produce different clusters in terms of the sizes and adjust rand index metrics (in other words, they may label students differently and put them into different clusters), we found those clusters share the same characteristics. We can summarize those clusters as potential-top-performance (PTP) cluster, potential-poor-performance (PTP) cluster, and Mixed-Performance (MP) cluster (in which students might be put into different clusters from different clustering algorithms).

Before the midterm, PTP students generally achieved very high scores on assignments and coding labs before the midterm, and they submitted their last submissions early. On the contrary, PPP students achieved low grades on assignments and coding labs before the midterm, and they tended to submit late. MP students shared some similarities between those two clusters of students, but their behaviour was typically in the middle. That is also why some of them will be labelled differently from different clustering algorithms.

We checked the exam grades of different clusters of students. We found that if we exclude the MP students. Then we can see that PTP students and PPP students have entirely different exam grades. We found that result is promising, and it can predict student performance.

## 8. FUTURE WORK

Future work can involve a deep analysis of what part of the course was improved and how it contributes to the clustering results from the year 2017 to the year 2018.



## References

- [1] M. An, H. Zhang, J. Savelka, S. Zhu, C. Bogart, and M. Sakr. Are Working Habits Different Between Well-Performing and at-Risk Students in Online Project-Based Courses? In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*, pages 324–330, Virtual Event Germany, June 2021. ACM. ISBN 978-1-4503-8214-4. doi: 10.1145/3430665.3456320. URL <https://dl.acm.org/doi/10.1145/3430665.3456320>.
- [2] M. Ankerst, M. M. Breunig, and H.-P. Kriegel. OPTICS: Ordering Points To Identify the Clustering Structure. page 12.
- [3] H. Chen and P. A. S. Ward. Predicting Student Performance Using Data from an Auto-Grading System. In *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering, CASCON '19*, pages 234–243, USA, 2019. IBM Corp. event-place: Toronto, Ontario, Canada.
- [4] R. W. Crues, G. M. Henricks, M. Perry, S. Bhat, C. J. Anderson, N. Shaik, and L. Angrave. How do Gender, Learning Goals, and Forum Participation Predict Persistence in a Computer Science MOOC? *ACM Transactions on Computing Education*, 18(4):1–14, Nov. 2018. ISSN 1946-6226. doi: 10.1145/3152892. URL <https://dl.acm.org/doi/10.1145/3152892>.
- [5] S. H. Edwards and M. A. Pérez-Quinones. WebCAT: Automatically Grading Programming Assignments. page 1.
- [6] A. Emerson, A. Smith, F. J. Rodriguez, E. N. Wiebe, B. W. Mott, K. E. Boyer, and J. C. Lester. Cluster-Based Analysis of Novice Coding Misconceptions in Block-Based Programming. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education, SIGCSE '20*, pages 825–831, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-6793-6. doi: 10.1145/3328778.3366924. URL <https://doi-org.proxy.lib.uwaterloo.ca/10.1145/3328778.3366924>. event-place: Portland, OR, USA.
- [7] M. Ester, H.-P. Kriegel, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. page 6.
- [8] S. C. Goldstein, H. Zhang, M. Sakr, H. An, and C. Dashti. Understanding How Work Habits influence Student Performance. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, pages 154–160, Aberdeen Scotland Uk, July 2019. ACM. ISBN 978-1-4503-6895-7. doi: 10.1145/3304221.3319757. URL <https://dl.acm.org/doi/10.1145/3304221.3319757>.
- [9] V. Gramoli, M. Charleston, B. Jeffries, I. Koprinska, M. McGrane, A. Radu, A. Viglas, and K. Yacef. Mining autograding data in computer science education. In *Proceedings of the Australasian Computer Science Week Multiconference*, pages 1–10, Canberra Australia, Feb. 2016. ACM. ISBN 978-1-4503-4042-7. doi: 10.1145/2843043.2843070. URL <https://dl.acm.org/doi/10.1145/2843043.2843070>.
- [10] A. Hellas, P. Ihantola, A. Petersen, V. V. Ajanovski, M. Gutica, T. Hynninen, A. Knutas, J. Leinonen, C. Messom, and S. N. Liao. Predicting academic performance: a systematic literature review. In *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, pages 175–199, Larnaca Cyprus, July 2018. ACM. ISBN 978-1-4503-6223-8. doi: 10.1145/3293881.3295783. URL <https://dl.acm.org/doi/10.1145/3293881.3295783>.
- [11] J.-L. Hung, M. C. Wang, S. Wang, M. Abdelrasoul, Y. Li, and W. He. **Identifying At-Risk Students for Early Interventions**A Time-Series Clustering Approach. *IEEE Transactions on Emerging Topics in Computing*, 5(1):45–55, Jan. 2017. ISSN 2168-6750. doi: 10.1109/TETC.2015.2504239. URL <http://ieeexplore.ieee.org/document/7339455/>.
- [12] H. Khosravi and K. M. Cooper. Using Learning Analytics to Investigate Patterns of Performance and Engagement in Large Classes. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 309–314, Seattle Washington USA, Mar. 2017. ACM. ISBN 978-1-4503-4698-6. doi: 10.1145/3017680.3017711. URL <https://dl.acm.org/doi/10.1145/3017680.3017711>.
- [13] S. N. Liao, D. Zingaro, K. Thai, C. Alvarado, W. G. Griswold, and L. Porter. A Robust Machine Learning Technique to Predict Low-Performing Students. *ACM Trans. Comput. Educ.*, 19(3), Jan. 2019. doi: 10.1145/3277569. URL <https://doi.org/10.1145/3277569>. Place: New York, NY, USA Publisher: Association for Computing Machinery.
- [14] M. I. López, J. M. Luna, C. Romero, and S. Ventura. Classification via clustering for predicting final marks based on student participation in forums. page 4. URL <https://eric.ed.gov/?id=ED537221>.
- [15] V. J. Marin, T. Pereira, S. Sridharan, and C. R. Rivero. Automated Personalized Feedback in Introductory Java Programming MOOCs. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 1259–1270, San Diego, CA, USA, Apr. 2017. IEEE. ISBN 978-1-5090-6543-1. doi: 10.1109/ICDE.2017.169. URL <http://ieeexplore.ieee.org/document/7930065/>.
- [16] K. Mierle, K. Laven, S. Roweis, and G. Wilson. Mining Student CVS Repositories for Performance Indicators. page 5, May 2005.
- [17] S. Mojarad, A. Essa, S. Mojarad, and R. S. Baker. Data-Driven Learner Profiling Based on Clustering Student Behaviors: Learning Consistency, Pace and Effort. In R. Nkambou, R. Azevedo, and J. Vassileva, editors, *Intelligent Tutoring Systems*, volume 10858, pages 130–139. Springer International Publishing, Cham, 2018. ISBN 978-3-319-91463-3 978-3-319-91464-0. doi: 10.1007/978-3-319-91464-0\_13. URL [http://link.springer.com/10.1007/978-3-319-91464-0\\_13](http://link.springer.com/10.1007/978-3-319-91464-0_13). Series Title: Lecture Notes in Computer Science.

- [18] F. Pedregosa. Scikit-learn: Machine Learning in Python. *MACHINE LEARNING IN PYTHON*, page 6.
- [19] D. M. Rao. Experiences With Auto-Grading in a Systems Course. page 8.
- [20] V. Sher, M. Hatala, and D. Gaevi. Analyzing the consistency in within-activity learning patterns in blended learning. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 1–10, Frankfurt Germany, Mar. 2020. ACM. ISBN 978-1-4503-7712-6. doi: 10.1145/3375462.3375470. URL <https://dl.acm.org/doi/10.1145/3375462.3375470>.
- [21] D. B. Silva and C. N. Silla. Evaluation of students programming skills on a computer programming course with a hierarchical clustering algorithm. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–9, Uppsala, Sweden, Oct. 2020. IEEE. ISBN 978-1-72818-961-1. doi: 10.1109/FIE44824.2020.9274130. URL <https://ieeexplore.ieee.org/document/9274130/>.
- [22] S. Sisovic, M. Matetic, and M. B. Bakaric. Clustering of imbalanced moodle data for early alert of student failure. In *2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 165–170, Herlany, Slovakia, Jan. 2016. IEEE. ISBN 978-1-4673-8740-8. doi: 10.1109/SAMI.2016.7423001. URL <http://ieeexplore.ieee.org/document/7423001/>.
- [23] S. E. Sorour, T. Mine, K. Goda, and S. Hirokawa. A Predictive Model to Evaluate Student Performance. *Journal of Information Processing*, 23(2):192–201, 2015. ISSN 1882-6652. doi: 10.2197/ipsjjip.23.192. URL [https://www.jstage.jst.go.jp/article/ipsjjip/23/2/23\\_192/\\_article](https://www.jstage.jst.go.jp/article/ipsjjip/23/2/23_192/_article).
- [24] J. Spacco, W. Pugh, N. Ayewah, and D. Hove-meyer. The Marmoset project: an automated snapshot, submission, and testing system. In *Companion to the 21st ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications - OOPSLA '06*, page 669, Portland, Oregon, USA, 2006. ACM Press. ISBN 978-1-59593-491-8. doi: 10.1145/1176617.1176665. URL <http://portal.acm.org/citation.cfm?doid=1176617.1176665>.
- [25] D. Steinley. Properties of the Hubert-Arable Adjusted Rand Index. *Psychological Methods*, 9(3):386–396, 2004. ISSN 1939-1463, 1082-989X. doi: 10.1037/1082-989X.9.3.386. URL <http://doi.apa.org/getdoi.cfm?doi=10.1037/1082-989X.9.3.386>.
- [26] R. Venant, K. Sharma, P. Vidal, P. Dillenbourg, and J. Broisin. Using Sequential Pattern Mining to Explore Learners Behaviors and Evaluate Their Correlation with Performance in Inquiry-Based Learning. In . Lavoué, H. Drachsler, K. Verbert, J. Broisin, and M. Pérez-Sanagustín, editors, *Data Driven Approaches in Digital Education*, volume 10474, pages 286–299. Springer International Publishing, Cham, 2017. ISBN 978-3-319-66609-9 978-3-319-66610-5. doi: 10.1007/978-3-319-66610-5\_21. URL [http://link.springer.com/10.1007/978-3-319-66610-5\\_21](http://link.springer.com/10.1007/978-3-319-66610-5_21). Series Title: Lecture Notes in Computer Science.
- [27] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec. 2007. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-007-9033-z. URL <http://link.springer.com/10.1007/s11222-007-9033-z>.
- [28] J. H. Ward. Hierarchical Grouping to Optimize an Objective Function. page 10, 2022.
- [29] B. L. Welch. The Generalization of ‘Student’s’ Problem when Several Different Population Variances are Involved. *Biometrika*, 34(1/2):28, Jan. 1947. ISSN 00063444. doi: 10.2307/2332510. URL <https://www.jstor.org/stable/2332510?origin=crossref>.