Do as I Do – Identification of Student Behavioural Patterns to Predict Attainment on an Online Learning Platform

Kai Cooper*
EPFL, Mathematics (exchange from Imperial College London)
Lausanne, Vaud, Switzerland
kai.cooper@epfl.ch

Nicolas d'Argenlieu*
EPFL, Data Science (Master)
Lausanne, Vaud, Switzerland
nicolas.thierrydargenlieu@epfl.ch

María Isabel Ruiz Martínez*
EPFL, Computer Science (exchange from Universidad de Granada)
Lausanne, Vaud, Switzerland
maria.ruizmartinez@epfl.ch

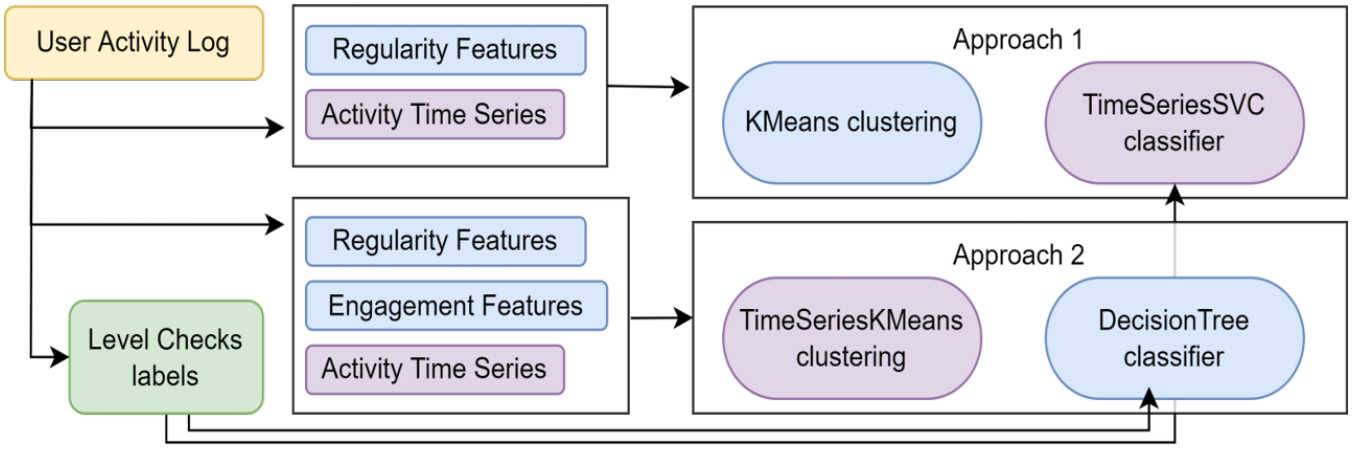


Figure 1. Overview of the modelling methodology employed in this report.

Abstract

In this report, we present the results of a learning analytics study. The analyzed data is produced by chronologically recorded clickstreams of Swiss secondary school student usage of the online maths and German learning platform Lernnavi. We quantify student behavioural patterns, in particular their regularity, and use these features to predict student attainment on the platform through a supervised learning pipeline. Our results reinforce that regular and persistent usage of the platform leads to better performance, however our models also serve to highlight some of the pitfalls of aiming to study human behaviour through opaque clickstreams.

Keywords: dataset, regularity, decision trees, random forest classifier, time series classification, support vector, kernel function, silhouette score, prediction, clustering, lernnavi, moocs, education, student attainment

*All authors contributed equally to this research.

MLBD-2022, Spring 2022, Lausanne, Switzerland

1 Introduction

Improved digital technologies have made online learning formats such as MOOCs and dedicated learning platforms such as Moodle more accessible. Lernnavi is one of these online education platforms, whose primary use is as an accompanying teaching and learning tool in secondary education. They offer maths and German lessons and exercises. Students track their progress through a mastery score updated by so-called **level checks**, unique to each topic taught on the platform.

1.1 Research Question

Innately, we are all aware of the fact that self-improvement demands commitment. This notion is omnipresent in education and is a persistent device used by educators to encourage learning. Consequently, we anticipate that students who choose (or are perhaps forced) to exhibit regular, recurrent patterns of learning behaviours will experience greater learning gains in the long-run. In this project, we wish to discover if this phenomenon is present within the Lernnavi environment, while going to a step further and identify which

specific behaviours *may* lead to higher attainment for a given student. In light of this, we state our research question:

> *Is it possible to identify one or more studying behaviors leading to a significant improvement in level check results?*

2 Data Description and Exploratory Analysis

The educational platform Lernnavi provided us with a series of datasets reflecting the usage of the platform, among which `learn_sessions_transactions`, `users`, `events` and `transactions` will be used to carry out our research work.

The `learn_sessions_transactions` dataset (502, 921 rows) contains information about the sessions of the platform. The sessions describe a learn or a level check session started and/or finished by a user and they are sorted chronologically, row by row. Additionally, for each session, there are one or more associated transactions (i.e., answers to tasks in that student's session).

The `users` dataset (13, 790 rows) contains demographic information of the users of the educational platform. The `events` dataset (3, 465, 559 rows) describe the events done by the users in the platform. Finally, the `transactions` dataset (800, 018 rows) reflects the multiple user transactions (e.g., answering a question) for each session. The detailed feature description of the datasets is shown in the Tables 6, 7, 8 and 9.

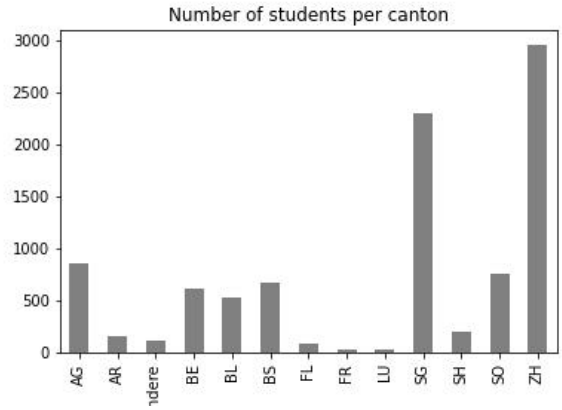


Figure 2. Bar plot of the number of students per canton. Zurich and St. Gallen contain most of the users on the platform.

A study on the `gender`, `canton` and `class_level` of the users of the platform was also performed. We remark that there is a geographical element to the data (shown in Figures 2 and 20), and this is important because it might introduce mixed effects between the groups of students by region. Consequently we investigated to some degree the cantonal representation within the dataset and how activity on the platform varied with location. At this stage it is quite preliminary is essentially an aperçu into what additional features we may need to include in the model for more accurate results (find more details in the associated notebook).

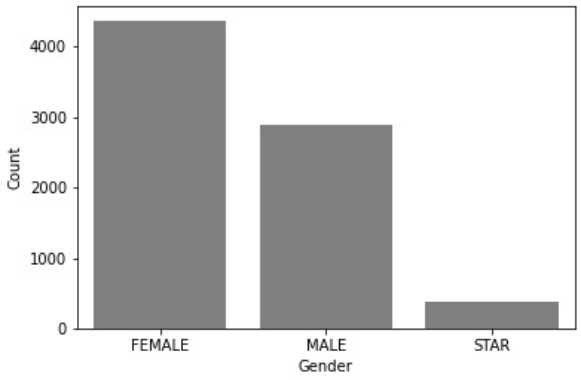


Figure 3. Bar plot of the number of students per gender. There are almost the same number of students per gender.

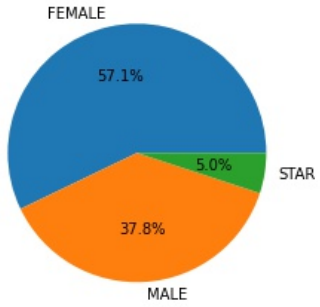


Figure 4. Proportion of students within each gener. STAR is an option reflecting if a student chooses to identify as neither male nor female.

Furthermore, we decide not to disclose based on the gender on the users since the difference between the percentage of females and males is not sufficiently large as shown in Figures 3 and 4. Finally, we found that the vast majority of students on the platform belong to year groups Gymnasium - 1. Jahr (0.24% of the users) and Gymnasium - 2. Jahr (0.16% of the users), the first group being the largest.

3 Data Processing

In the data processing phase, we first remove users with no transaction data (i.e. we keep users that are both in the events table and in the transactions table). To continue, inactive

users will also be removed from our dataset since we consider that they are not using the platform enough to extract any valuable conclusion from them.

Inactive students are students that are in one of the following situations:

- Students that have never performed a level check.
- Students that have not performed any `GO_TO_THEORY` action and neither any `SUBMIT_ANSWER` action.
- Students that have at least two significant weeks. A *significant week* is defined as a week with at least five significant actions. Moreover, the *significant actions* are the actions in the event table that are linked through the transaction token to the transactions table. As a consequence, we will exclude actions like `LOGIN`, `LOGOUT` or `NAVIGATE_DASHBOARD` from the list of significant actions.

Since our goal is to predict users' success in level checks based on their regularity in time, we are not interested in users who did not perform any level check. Moreover, we will remove untrackable students: those who never did any training question or theory reading event. The justification of the cleaning decision is simply that we are not interested in users that have not performed any training or preparation for the level checks because we cannot measure their regularity (since they have not used the platform enough).

Finally, we will remove users that have not been using the educational platform sufficiently during several weeks. Thus, we remove users with less than two significant weeks since we want to have data from different weeks to apply time series techniques. In addition, a significant week is defined like above since we consider that a student can reach the minimum of five significant actions per week in the platform very easily.

Additionally, features that are not useful for our study have been removed. The removed features are the following: `transaction_id`, `transaction_token`, `document_id`, `document_version`, `user_agent`, `validation`, `solution`, `type`, `learn_session_id`, `topic_id`, `is_closed`, `type_id`, `is_accepted`, `event_id`, `session_id`, `tracking_data`, `event_type`. Also, we are dropping the `evaluation` and `input` columns because we are not interested in whether a question is correct or not but in the result obtained in the whole level check.

Moreover, the timestamp we get from the original raw data will be transformed to a `datetime` object via the function `to_datetime` of the pandas library taking into account the fact that the timestamp given is in milliseconds to further extract the week number. We need to compute the week number to extract features such as the `num_actions_per_week` or `num_weeks_per_user` because we want to analyse the behaviour of the students over time.

Our dataset after the cleaning process consists of 790, 425 rows corresponding to actions of users on the platform. For each row, the features shown in Table 1 are listed as columns.

The information corresponding to the level checks performed by the users of the educational platform is located in the `learn_sessions_transactions` table and can be linked to the `transactions` table via the `transaction_id`. After fetching all the level checks' dates, we create our labels to represent a measure of the learning. Indeed, we track changes in the mastery score for each level check and attribute a label of 1 for the level checks featuring a mastery score improvement, and a label of 0 for the ones featuring a decreasing mastery score. Then, as the chosen granularity is *weekly*, we must aggregate all the level checks happening the same week for the same or different topics. To do so, we choose a simple approach by attributing to the given week the *majority label* i.e. the level check week gets attributed the label of the majority of the level checks occurring during the week.

Therefore, each **sample** in our classification training data is constituted of chosen **features** and a **label**. The features (varying depending on the selected approach) are information up to a specific week preceeding the level check week. The label is the result of the aggregated change in mastery score i.e. whether the student mostly improved its level check score or decreased it. See section 4 for the details on each approach to illustrate the features-label in each case.

By plotting the histograms of some of the numerical features, see Figure 5 for examples, it is clear that counting student *actions* results in a positively skewed distribution, which demonstrates that few students have very high action counts on the platform.

4 The Proposed Approaches

We proceed onto experiments separated in 2 different approaches. Each of these approaches features a **clustering** and a **classification** part. Both techniques are applied to a different set of features to avoid data leakage through the experiment. They are chosen for the following reasons:

- **Clustering** - The goal is to try to **identify and separate** according to behaviors detected in the raw data. This is a first naive approach towards grouping students with similar learning processes. Moreover, in an attempt to leverage the clustering, we also train a classifier for each clustered group. We hope to increase the classifier's accuracy by proceeding this way.
- **Classification** - This technique is used to separate classes according to our created labels. Therefore, we train a group of classifiers on a class-balanced training set. The group of classifiers includes 1 classifier for the whole training data, and 1 classifier per cluster. Its aim is to measure the predictive potential of the data regarding students' improvement.

See Figure 1 for a flowchart depiction of our methodology.

Table 1. Features obtained after the cleaning process.

Feature	Description
user_id	identifier of the user of the educational platform
timestamp	timestamp information of when the action was performed
week	derived from the feature timestamp, it corresponds to the week when the action was performed
category	classification of the action performed
action	type of action performed
start_time	start time of the action
commit_time	commit time of the action
num_checks	total number of level checks that the user with user_id has performed
nun_participations	total number of SUBMIT_ANSWER and GO_TO_THEORY actions performed by an user of the platform
num_actions_per_week	total number of significant actions performed by a particular user in a particular week
num_weeks_per_user	total number of significant weeks performed by a particular user

4.1 A1

The first approach features a *time-agnostic* (no time dimension in the data) clustering and a time series classification. We describe both.

The **clustering** is performed based on a set of time agnostic features for each user. The features are computed over the whole study period of 40 weeks. They include: **counts of each significant action** (an action with a transaction token); **length of time spent** on the platform, and a set of so-called **regularity features** [1]. The latter is a set of metrics which measure to what extent user patterns repeat of chosen time periods, for example over hours of the day, days of the week, or weeks of the month. For instance, one feature, PWD – an abbreviation of *peak on weekday* – measures if a users activity is concentrated around days of the week, based on the entropy of the histogram of the user's weekly activity. Figure 6 illustrates the counts that the PWD metric is measuring. We refer the reader to [1] to find a detailed description of all of the features. Moreover, we use the **K-Means clustering** technique together with the **Silhouette Score** measure in order to compute the optimal number of cluster. K-Means is selected as it is a simple and scalable algorithm which produced reasonable computation times and results.

The **classification** is done over a set of time series. Contrary to the clustering phase during which each user in the training set corresponds to a sample, the classification uses time series, one for each level check. The training sample is formed by the weekly data for the student having the level check, up to the week preceding the level check. The weekly data includes the total *number of actions per week*, the **time spent** on the platform during this week, **count of actions** for the selected relevant actions. Please see Figure 2 for a complete list of the feature aliases. Moreover, we use the *tslearn* [5] library providing us with a collection of machine learning techniques adapted to time series. Among the proposed classicators, we select a **time series support vector** and a **time series K-NN** classification technique. Further details on their use are provided in section 5.

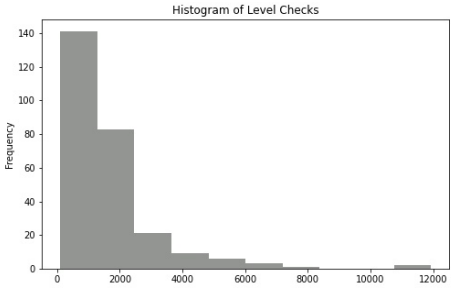
Table 2. Features details for clustering and classification in approach 1. For details regarding the regularity metrics, please refer to section 4.1 and [1].

Clustering Features	Classification Features
PHD	num_actions_per_week
PWD	elapsed_time
WS1	REVIEW_TASK
WS2	SKIP
WS3	CLOSE
FDH	SUBMIT_ANSWER
FWH	GO_TO_THEORY
FWD	NEXT
	VIEW_QUESTION
	GO_TO_BUG_REPORT
	OPEN_FEEDBACK
	CLOSE_FEEDBACK
	GO_TO_COMMENTS
	SHARE
	REQUEST_HINT

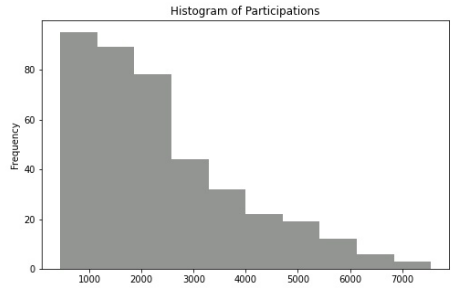
4.2 A2

The second approach inverts the type of features fed to the clustering and classification. In this approach, we first cluster our user time series, and then feed time agnostic features to the classifier. We describe both.

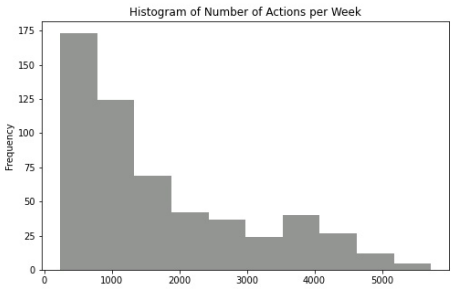
The **clustering** is done on the time series of the users. Similarly to the time series classification of approach 1, we aggregate data on a weekly basis. Thus for each user, we create a vector per week containing the following information: total **number of actions** performed during the week, **time spent** on the platform during the week, and the **count of some selected relevant actions** for the week. Please see table 3 for a complete list of the features aliases. Moreover,



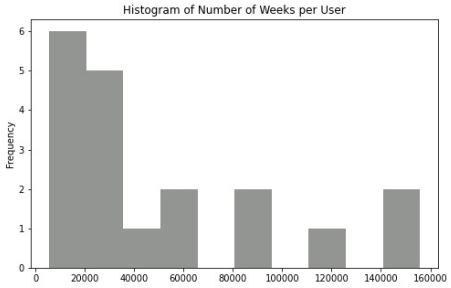
(a) Histogram of the number of *level checks* performed by the users of the platform.



(b) Histogram of the number of *participations* performed by the users of the platform.



(c) Histogram of the number of *actions per week* performed by the users of the platform.



(d) Histogram of the number of *weeks per user* in the platform.

Figure 5. Histograms of some of the numerical features presented.

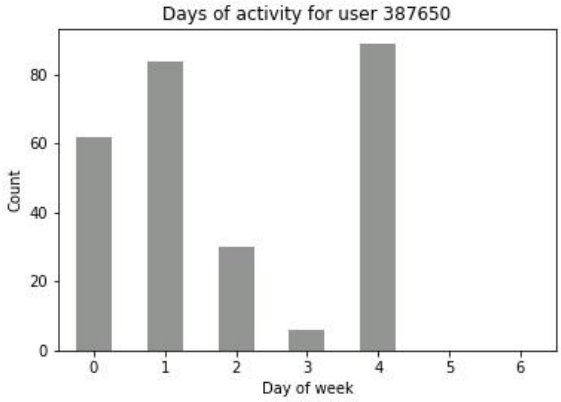


Figure 6. Illustrative plot of daily activity for user 387650. We find that this user completes significant actions on every day of the week except for the weekend, with peaks on Tuesday and Friday.

we again use the methods from the *tslearn* [5] library. Here we use a **time series K-Means** clustering technique and a **kernelized version of the time series K-Means** i.e. a kernel function is applied to the time series before performing K-Means.

Table 3. Features details for clustering and classification in approach 2.

Clustering Features	Classification Features
num_actions_per_week	num_weeks_on
elapsed_time	num_actions_per_week[1]
REVIEW_TASK	FDH
SKIP	FWH
CLOSE	FWD
SUBMIT_ANSWER	eng_score
GO_TO_THEORY	
NEXT	
VIEW_QUESTION	
GO_TO_BUG_REPORT	
OPEN_FEEDBACK	
CLOSE_FEEDBACK	
GO_TO_COMMENTS	
SHARE	
REQUEST_HINT	

For the **classification**, a **decision tree classifier** and a **random forest classifier** based on behavioural features were chosen due to the very interpretable nature of such models. Indeed, it allows for the categorisation of students based on quantitative metrics on usage of the platform, while also ranking the importance of the behavioural features, which helps us in answering our research question.

Table 4. Description of the time agnostic, aggregated features used in approach 2 for a given user i for all data recorded on the platform before level check t.

Feature	Description
num_weeks_on	number of weeks recorded having used the platform
num_actions_per_week[2]	average number of actions with a transaction token per week
FDH	measures the extent to which the hourly pattern of user's activities is repeating over days
FWH	measures if the hourly pattern of activities is repeating over weeks
FWD	captures if the daily pattern of activities is repeating over weeks
eng_score	binary indicator of 'high' user engagement

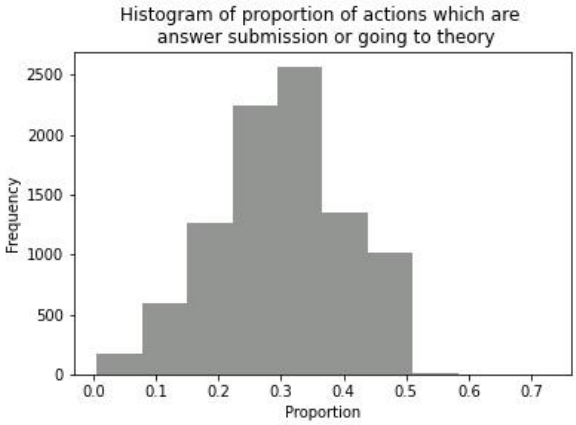


Figure 7. Proportion of actions which are answer submissions or going to theory.

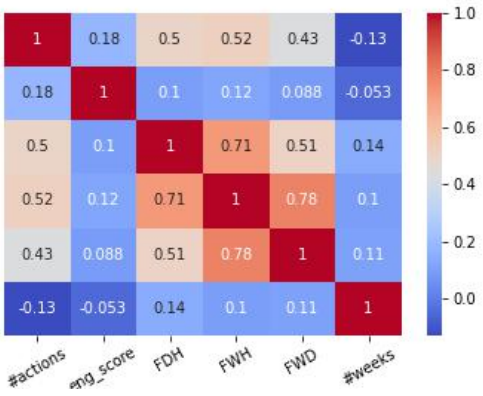


Figure 8. Correlation heatmap between all the features. Note we have relabelled num_actions_per_week to #actions and num_weeks_on to #weeks for display purposes.

Let N be the number of users and L_i be the number of level checks user i has completed. For each $i \in \{1, \ldots, N\}$ and $t \in \{1, \ldots, L_i\}$, the data fed into the decision tree classifier is a vector $x_i^t = (x_{i1}^t, x_{i2}^t, \ldots, x_{i6}^t)$, where x_{i1} is a positive integer, x_{ik} for $k = 2, 3, 4$ is a positive real number and $x_{i6} \in \{0, 1\}$. The features x_{ik} for $k = 1, \ldots, 6$ are listed in Table 4. Note the index t is ordered in time.

To elaborate further, x_{ik} for $k = 2, 3, 4$ are **frequency-based** regularity features taken from [1]. In brief, these features find the spectral density of the time series generated by recording student actions, and subsequently detecting any important frequencies within the time series. This serves to identify with what frequency students repeat certain patterns of specified time periods: for the features FDH, FWH and FWD these are hours-across-day patterns (e.g. a user is active at 8AM every morning), hour-across-week patterns (e.g. a user is active 8AM every *Monday*) and days-across-week (e.g. a user is active every *Monday*) patterns respectively. Notably, the features decrease in granularity, while each capturing interesting snapshots of student behaviour.

An 'engagement score', inspired by [2], records whether a student demonstrated that they performed activities directly related to learning (submitting answers to questions and going to theory sections of the app) more than other students. To quantify this, if for a given user, the two aforementioned actions comprised more than 25% of all actions recorded in num_actions_per_week, then they were given an engagement score of 1 and 0 otherwise. We remark that each of these features aim to quantify some form of consistent behavioural pattern on the platform, ranging from simply being present on the platform, to using it in a repeated manner over a long period of time.

5 Experiments and Results

For each of the aforementioned approaches, we derive a series of experiments. In this section we detail the design of the experiments and then explore the results.

5.1 Experiments

5.1.1 A1. For the first approach, the clustering was done using the **K-Means** algorithm. For this experiment, we fine-tune the number of clusters $k \in \{2, 3, \ldots, 10\}$. Then we use the **Silhouette Score** in order to see which clustering is most coherent to our data.

The classification is done using 2 different classification techniques applied to time series. Namely, we classify for the

clustered and non-clustered setting using the `TimeSeriesSVC` (support-vector technique) and `KNeighborsTimeSeriesClassifier` (K-NN classification technique). For the former method, we could not perform experiments by varying the `kernel` parameter as the *Global Alignment Kernel* `gak` **kernel** is the only one working for the time series classification. *The GAK kernel is responsible for aligning the different time series before classification.* For the latter method, we had the possibility to tune the **number of considered neighbors**. However preliminary results with the default parameter showed that the model performed significantly worse than its support-vector counterpart. Therefore, we did not perform experiments on this dimension of the model. The default parameter used was `k=3`. Finally due to the significant amount of time necessary to train the classifier, we chose to perform experiments by varying the **number of samples** used for training in the range $[10, 3953]$. Let us note that the upper bound of this range is not reached for all experiments, however all experiments go up to at least 828 samples.

5.1.2 A2. For the second approach, the clustering was done using 2 different time series clustering techniques. The techniques are `TimeSeriesKMeans` (K-NN based clustering techniques) and `KernelKMean` (variation of the former applying a kernel function before clustering). For the first technique, we vary the **number of clusters** in the integer range $[2, 6]$, the **maximum iterations** performed by the algorithm in the list $[50, 100]$, the tolerance criterion to indicate convergence in the list $[1e-5, 1e-6, 1e-7]$, the **distance metric** in the list [`euclidean`, `dtw`], and whether the algorithm should use **inertia** or not. the second technique features hyperparameter tuning on the applied kernel function in the list [`gak`, `linear`, `poly`, `sigmoid`], and the number of clusters in the integer range $[2, 4]$.

In the second part, we performed two experiments. In the first experiment, the classification step is carried out with all the behavioral features shown in Table 4, whereas in the second experiment we left out the feature `num_weeks_on`.

Both a **Decision Tree Classifier** and a **Random Forest Classifier** were trained on both experiments using the whole dataset since the clustering was not relevant. Hyperparameter tunning was performed using **Randomized Search** to determine the best parameters of each of the models using the grids described in Tables 16 and 17.

The parameters tuned for each of the models in both of the experiments with their respective best combinations can be found in the Tables 18, 19, 20 and 21.

The Figures 21 and 22 show accuracy results for several trainings done with different combinations of the parameters of the Decision Tree Classifier for the first and the second experiment respectively. Equivalently, the Figures 23 and 24 show accuracy results for several trainings done with different combinations of the parameters of the Random Forest Classifier for the first and the second experiment respectively.

The feature importance of each of the classifiers in both experiments is shown in Figures 25, 26, 27 and 28. As we can see, the most important feature in Experiment 1 is `num_weeks_on` whereas in Experiment 2 is `num_actions_per_week`. Surpringsingly, the regularity features computed (`FDH`, `FWH`, `FWD` and `eng_score`) have a low importance.

5.2 Model Results and Evaluation

After having run the experiments, we measure the results using different metrics. These metrics depend on the clustering or classification step in the approach. For the clustering step, we use the **Silhouette Score** to measure the cluster relation to our data. For the classification step, we use 6 different metrics [4], namely the **accuracy** (percentage of samples correctly classified), the **balanced accuracy** (accuracy taking class balance into account), the **adjusted balanced accuracy** (balanced accuracy accounting for chance in classification), the **F1 score** (harmonic mean of the precision and recall), the **weighted F1 score** (F1 score weighted by class support) and **Matthew's coefficient** (binary classification correlation coefficient).

5.2.1 A1. The silhouette scores linked to the clustering results features a clear downward trend (see Figure 9). Indeed, the maximum score of **0.66** is reached for **2** clusters. It then drops down to 0.53 at 3 clusters and keeps decreasing. The two groups feature **2'132** and **344** samples respectively, denoting an important size imbalance. Therefore, we ask ourselves if the clustering is relevant and proceed to the classification with both clustered and non-clustered experiments.

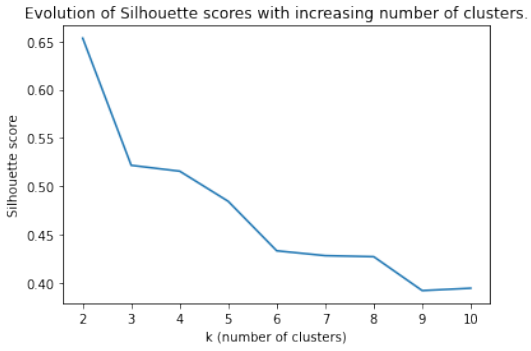


Figure 9. Silhouette score evolution with the number of clusters in A1 (time agnostic clustering).

Let us start by comparing the classification methods in the global (no clustering) setup. For `TimeSeriesSVC`, we can see that all scores increase significantly up to **828** samples (see figure 10). For the accuracy and F1-score, we reach a maximum of **79.9%** accuracy for 828 samples (79.6% for 1101

samples) and **86.8%** F1-score for 1101 samples (86.6% for 828 samples). Finally, we see a maximum correlation of **+0.469** (Matthew's coefficient) at 828 samples. However, using the `KNeighboursTimeSeriesClassifier`, the accuracy seems to oscillate between **43%** and **44.6%**, and F1-score oscillates between **45.1%** and **47.2%** (see figure 13). Both scores are below 50% and no clear trend can be observed in any of the metrics. Finally the maximum correlation coefficient is **+0.088** which is obtained with 10 samples.

Then, by looking at the clustered experiments, we can further confirm some insights. For `TimeSeriesSVC`, regardless of the clusters we observe that all scores increase sharply until **667** samples training-set-size (cluster 0) and **572** samples (cluster 1) (see figures 14 and 15). Following this point, the scores seem to decrease slightly for cluster 0 and stay steady for cluster 1. This variation in cluster 0 could be interpreted as overfitting. However the highest accuracy and F1-score are confidently reached for **886** samples (accuracy: **79.9%**; F1-score: **86.8%**) for cluster 0 and **853** samples (accuracy: **75.6%**; F1-score: **84.2%**) for cluster 1. Confidence comes from the fact that cluster 0 evaluates classifiers trained up to 3'953 samples and cluster 1 up to 1'556 samples, both of which are significantly higher than the number of samples for which we observe the best scores. For `KNeighboursTimeSeriesClassifier`, the results are very different between cluster 0 and cluster 1. Let us note that cluster 0 evaluates results only **4** times up to 886 samples, whereas cluster 1 has **29** evaluations up to 994 samples. However the trends are still significantly different (see figures 16 and 17). In cluster 0, the best accuracy and F1-score are reached for **10** samples with **50.3%** accuracy and **57.4%** F1-score. Cluster 1 has a more consistent behavior, and the best accuracy is reached at **115** samples with **61.7%**. Best F1-score is also reached at **115** samples with **73.2%**. Moreover, adjusted accuracy and correlation oscillate around **0** with best scores of **8.3%** and **0.099** at **10** samples. There is no trend for these two metrics as negative values are observed as early as **200** samples, before going back to positive values at around **400** samples and then negative again around **750** samples.

5.2.2 A2. The clustering for the second approach featured mixed silhouette scores depending on the technique. Indeed, the time series KMeans `TimeSeriesKMeans` clustering reached its highest silhouette score of **0.987** for the following configuration: **2** clusters, **50** iterations max, tolerance of **1e-05**, **euclidian** metric, and with **inertia** (see figure 18). However, the cluster imbalance is high with cluster 0 having only 2 samples. On the other hand, the time series kernel (`KernelKMeans`) clustering performed poorly. The trend goes downward with the number of clusters starting from the beginning. The highest silhouette score is reached at **0.137** with the **sigmoid kernel** function and **2** clusters (see figure 19). The score goes below 0 as soon as the number of clusters gets higher than 2. Moreover there is no distinguishable difference between the performances of the different kernels.

The best results were obtained by the **Random Forest classifier** trained in **Experiment 1**. The values obtained for the different metrics described at the beginning of this section can be seen in Table 5, where we obtain 97.8% **accuracy** and 98.5% **F1-score** for 962 samples. However, when we trained the **Random Forest Classifier** in **Experiment 2**, we obtain 56.1% **accuracy** and 65.8% **F1-score** (results shown in table 22). This significantly drop in the accuracy when we performed Experiment 2 may be due to a lack of expressiveness of the features used. The results obtain for the Decision Tree Classifiers trained on Experiments 1 and 2 can be found in Tables 23 and 24.

Figures 11 and 12 plot the accuracy and the F1-score against the number of samples in both experiments for the Random Forest model. Analogous graphs are shown in Figures 29 and 30.

6 Discussion

Let us first further elaborate on the relevance of our two approaches. Moving from A1 to A2 we switched the order of the clustering and classification steps, in which our use of time series data was an attempt at verifying which type of data was best suited for each specific type of classification. Moreover splitting the approaches in 2 steps allows us to use 2 different kinds of data in the same experiment and avoid data leakage as the features are completely different.

Another observation is that in both experiments, clustering did not bring much information or precision to the classification step. Indeed, even after hyperparameter fine-tuning, the time series clustering produced an intractable clustering with one cluster having 2 samples only whereas the second cluster holds all the other users. We deduce from it that distance (even with kernel transformations) is not a representative metric to interpret an optimal learning behavior. With respect to the classification results, we conclude that the separation in various dimensions is a more adapted way to catch the underlying patterns in the data.

A remaining interrogation concerns the very volatile classification accuracy in the second approach. We suspect that there is an error in the data that we used, leading to some form of data leakage between the training and the evaluation. This could indeed explain the very high classification accuracy (97.8%) and then the sudden drop following a feature removal. However, we were unable to find the mistake and can thus only formally state our doubts.

Our results demonstrate that differing patterns of student behaviours certainly influence their performance on the platform. We could not identify a specific optimal behavior, and our highest prediction accuracy confirmed the 'common sense' answer to our question: the more time you spend on

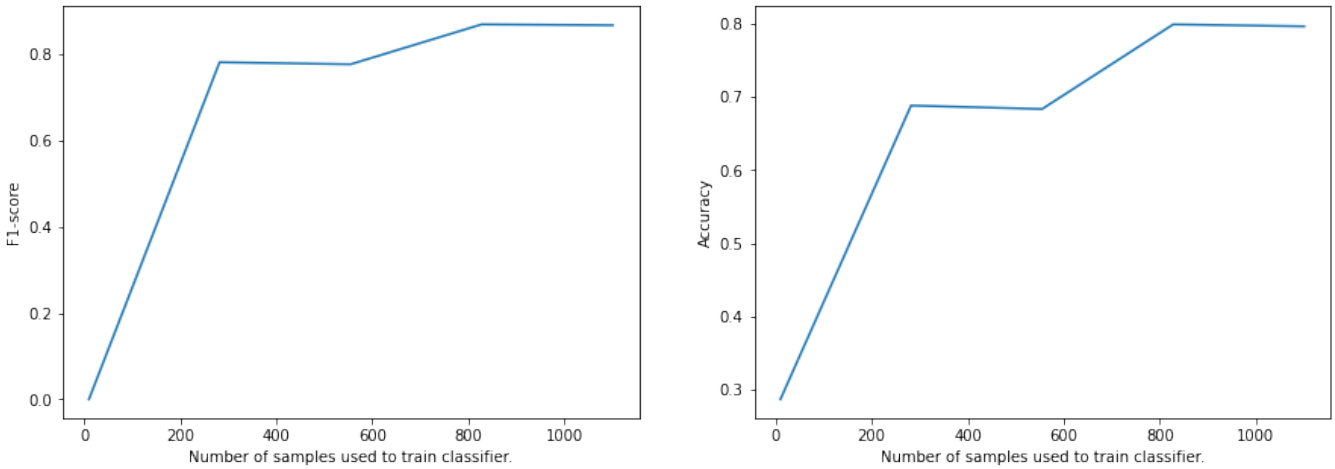


Figure 10. Evolution of F1-score and accuracy with the number of samples for the support vector classification in A1 (best performing in approach 1).

Table 5. Accuracy, **balanced accuracy**, **adjusted balanced accuracy**, **F1 score**, **weighted F1 score** and **Matthew's coefficient** for the *Random Forest Classifier* trained in *Experiment 1.*

Samples	Accuracy	Balanced Accuracy	Adj. Balanced Accuracy	F1	Weighted F1	Matthew's coeff.
10	0.888	0.917	0.834	0.918	0.893	0.763
168	0.903	0.930	0.860	0.930	0.907	0.792
327	0.913	0.941	0.881	0.938	0.917	0.814
486	0.975	0.981	0.963	0.983	0.975	0.939
644	0.978	0.982	0.963	0.985	0.978	0.945
803	0.984	0.985	0.969	0.989	0.984	0.960
962	0.987	0.988	0.976	0.991	0.987	0.967

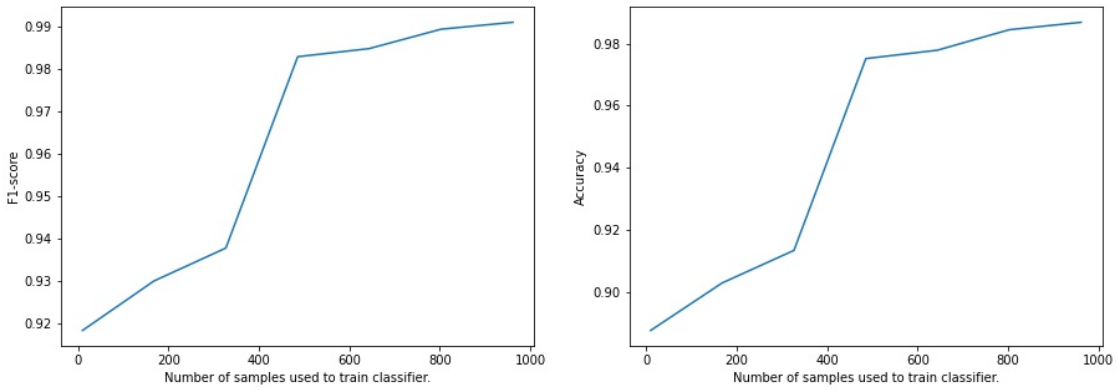


Figure 11. *Accuracy* and *F1-score* against number of samples of the *Random Forest Classifier* trained in *Experiment 1.*

the platform, the better you will tend to achieve. Succinctly, consistency begets attainment.

These results, however, highlight one key difficulty with our approach. Extracting a meaningful measure of human behaviour through myriad, anonymous clickstreams is a monumental task with many concerns and limitations [3]. Human behaviour is an extremely nuanced piece of our being, and the advent of Big Data gives scientists a new

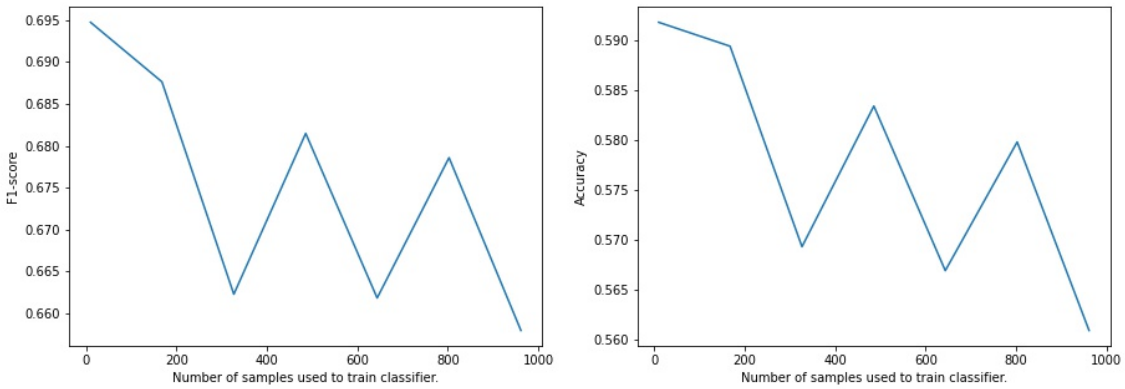


Figure 12. *Accuracy* and *F1-score* against number of samples of the *Random Forest Classifier* trained in *Experiment 2*.

avenue through which to understand it. Nevertheless, more advanced techniques and methods need to be developed in order to extract precise patterns that could reasonably have a causal effect on the outcome, as this is the ultimate goal; that is, for each student to have a personalised education.

7 Conclusion and Future Works

To conclude, this report details how we looked into student's behavior to improve the examinations results. It details the dataset and the processing work done to select features and samples to be used in our proposed approaches. Two approaches are then detailed to attempt to answer the research question. The first one features a clustering followed by a time series classification, whereas the second one starts with a time series clustering and follows with a time agnostic classification. Within each approach, we run various experiments, in order to fine tune our results and select the best models and methods. Very interestingly, we get promising results for our first approach with the best accuracy going up to 79.9% with a non-clustered support vector classification over the designated set of features. More suprisingly, we reach a 97.8% accuracy for the second approach, leading us to investigate our work.

Put altogether, this project helped us assess the potential of machine learning methods to leverage real-life data and gain insights from production data. We built a full data processing pipeline and could try different methods to explore, process, transform and learn from it. The discussions in section 6 elaborate on our thoughts regarding the employed methods, their constraints and their limitations. However, we believe that our work paves the way for further in-depth research into Lernnavi's data and online learning platform insights in general.

In future work, we suggest that the results of our work be used to inform an experiment in which the inverse approach is taken. That is, informed by data and neuroscientific and psychological literature, conduct a (conditionally) randomised experiment in which students are exposed to – or, forced to exhibit – a particular behaviour or structure, and appeal to the field of causal inference in order to determine any effect on attainment, by comparing and contrasting enforced behavioural patterns. For example, students might be use the platform solely as a homework device, or it might be used in conjunction with classroom learning. In this way, educators can begin to narrow the scope on what behavioural patterns lead to success, which could work in tandem with the development of machine learning models, in order to refine the patterns for which they search.

Acknowledgments

To Paola, for the sustained interest and really fruitful and engaging topical discussions.

References

[1] Mina Shirvani Boroujeni, Kshitij Sharma, Łukasz Kidziński, Lorenzo Lucignano, and Pierre Dillenbourg. 2016. How to Quantify Student's Regularity? In *Adaptive and Adaptable Learning*. Springer, Cham, Switzerland, 277–291. https://doi.org/10.1007/978-3-319-45153-4_21

[2] Mushtaq Hussain, Wenhao Zhu, Wu Zhang, and Syed Muhammad Raza Abidi. 2018. Student Engagement Predictions in an e-Learning System and Their Impact on Student Course Assessment Scores. *Comput. Intell. Neurosci.* 2018 (Oct. 2018), 6347186. https://doi.org/10.1155/2018/6347186

[3] David Lazer, Eszter Hargittai, Deen Freelon, Sandra Gonzalez-Bailon, Kevin Munger, Katherine Ognyanova, and Jason Radford. 2021. Meaningful measures of human society in the twenty-first century. *Nature* 595 (July 2021), 189–196. https://doi.org/10.1038/s41586-021-03660-7

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

[5] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, and Eli Woods. 2020. Tslearn, A Machine Learning Toolkit for Time Series Data. *Journal of Machine Learning Research*

21, 118 (2020), 1–6. https://www.jmlr.org/papers/v21/20-091.html

A Appendix

A.1 Further Tables and Figures

Table 6. Feature description of the `learn_sessions_transactions` dataset.

Feature	Description
learn_session_id	unique identifier for each learn session
transaction_id	transaction identifier, used to link with transactions table
topicId	identifier of the topic this session was about
max_num_tasks	maximum number of tasks included in this session (only relevant for the learn session)
is_closed	if this session has been finished (1: finished; 0: not finished)
type_id	if this session is a learn or level check (1: learn; 2: level check)
is_accepted	if the user finally accepted the result of this session (1: accpeted; 0: refused).

Table 7. Feature description of the `users` dataset.

Feature	Description
user_id	personal identifier of user in database
gender	only three values: male, female or missing
canton	swiss canton
class_level	school year in swiss system
class_code	identifier of student's class. There are students that are using the platform as part of their school work

Table 8. Feature description of the `events` dataset.

Feature	Description
event_id	identifier of event in database
user_id	user who performed the event
timestamp	timestamp of event
category	classification of action (task, general, statistics, etc)
action	type of action performed
event_type	whether the event required students to view or to click
transaction_token	used to link to `transactions` table
session_id	during which session the event took place
tracking_data	optional content associated to this event (e.g., the new points mastered for a topic)

Table 9. Feature description of the `transactions` dataset. The features `document_id` and `document_version` can be used to merge with the `document` table

Feature	Description
`transaction_id`	unique identifier for each transaction
`transaction_token`	used to join with events table
`user_id`	user who performed the transaction
`document_id`	document that was answered in transaction
`document_version`	version of document that was answered
`evaluation`	whether the user answered correctly or not. It is possible that it was only partially right
`input`	answer the user gave
`start_time`	timestamp of when the user started answering
`commit_time`	timestamp of when the user submitted the answer
`user_agent`	the browser that the user used
`validation`	used to validate the format of the input
`solution`	solution to question
`type`	type of question

Table 10. Table of metrics for the non-clustered support vector classification in approach 1.

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.287	0.500	0.000	0.000	0.128	0.000
282	0.688	0.621	0.242	0.781	0.689	0.241
555	0.683	0.620	0.240	0.776	0.685	0.237
828	0.799	0.702	0.403	0.868	0.784	0.469
1101	0.796	0.700	0.400	0.866	0.782	0.462

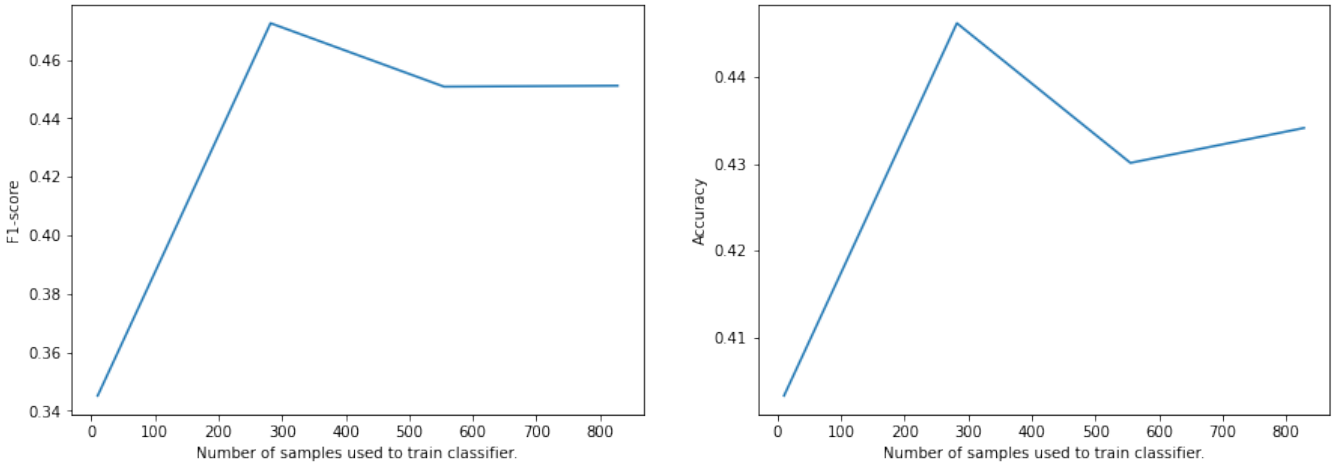


Figure 13. Evolution of F1 score and accuracy for the non-clustered KNN classification in approach 1.

Table 11. Table of metrics for the non-clustered KNN classification in approach 1.

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.403	0.539	0.078	0.345	0.376	0.088
282	0.446	0.519	0.039	0.472	0.457	0.037
555	0.430	0.506	0.012	0.451	0.438	0.011
828	0.434	0.514	0.029	0.451	0.441	0.028

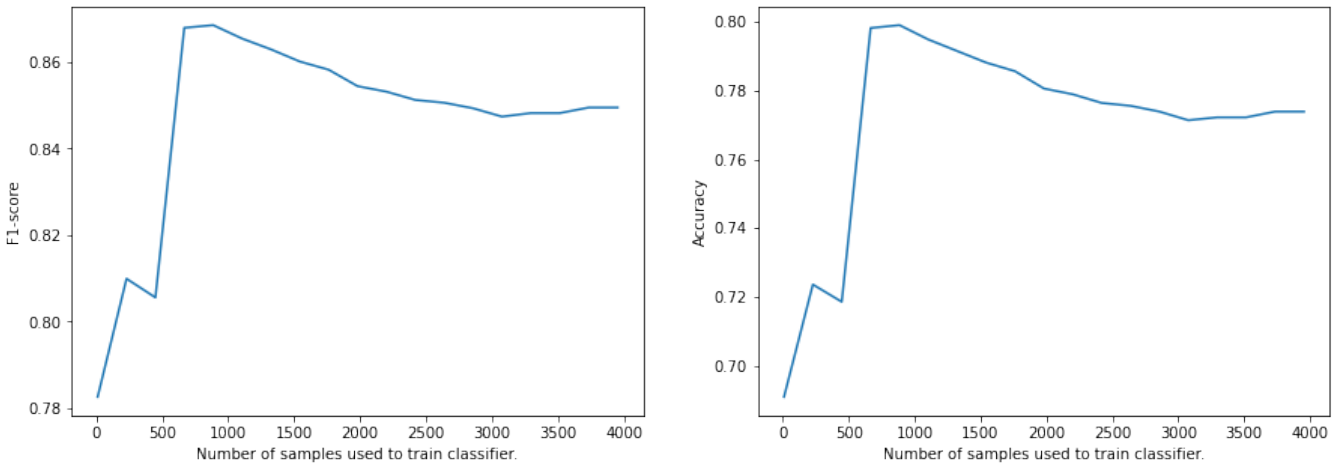


Figure 14. Evolution of F1 score and accuracy for the support vector classification in approach 1 (cluster 0).

Table 12. Table of metrics for the support vector classification in approach 1 (cluster 0).

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.691	0.627	0.254	0.783	0.694	0.249
229	0.724	0.649	0.299	0.810	0.721	0.304
448	0.719	0.647	0.294	0.806	0.717	0.297
667	0.798	0.701	0.403	0.868	0.785	0.460
886	0.799	0.702	0.404	0.869	0.786	0.462
1105	0.795	0.699	0.398	0.865	0.782	0.452
1324	0.791	0.697	0.393	0.863	0.779	0.444
1543	0.788	0.696	0.392	0.860	0.777	0.437
1762	0.786	0.694	0.389	0.858	0.775	0.431
1981	0.781	0.691	0.382	0.854	0.770	0.420
2200	0.779	0.690	0.379	0.853	0.769	0.416
2419	0.776	0.688	0.376	0.851	0.767	0.410
2638	0.776	0.687	0.375	0.851	0.766	0.409
2857	0.774	0.686	0.372	0.849	0.765	0.405
3076	0.771	0.684	0.369	0.847	0.762	0.399
3295	0.772	0.684	0.368	0.848	0.763	0.400
3514	0.772	0.684	0.368	0.848	0.763	0.400
3734	0.774	0.685	0.371	0.849	0.764	0.404
3953	0.774	0.685	0.371	0.849	0.764	0.404

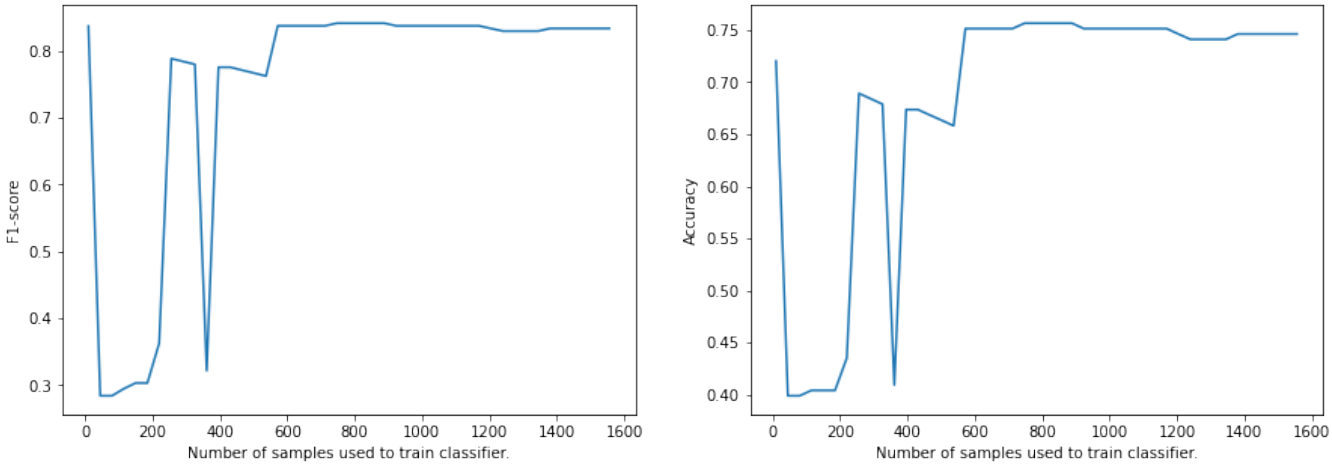


Figure 15. Evolution of F1 score and accuracy for the support vector classification in approach 1 (cluster 1).

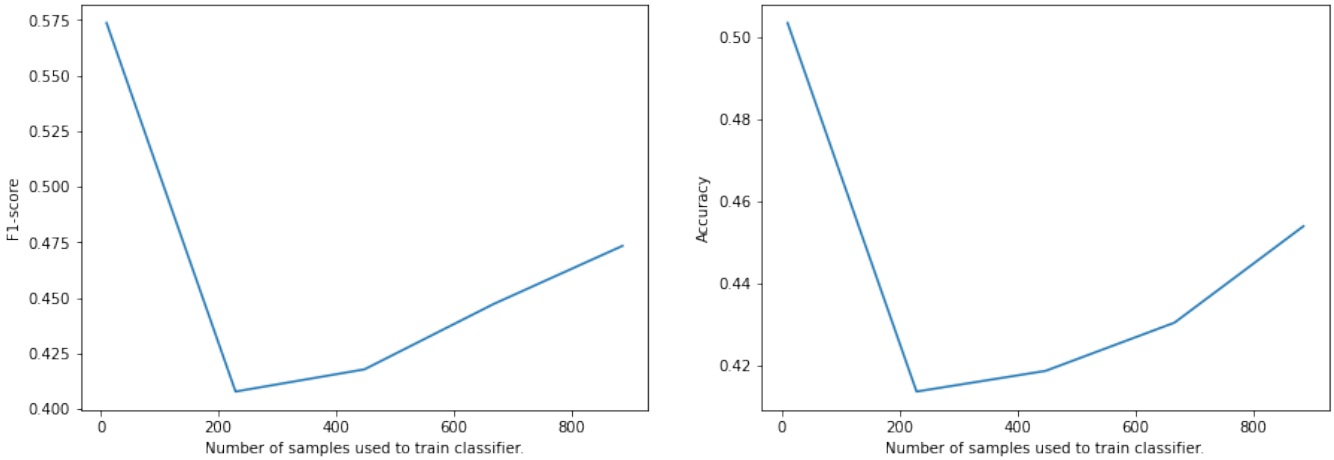


Figure 16. Evolution of F1 score and accuracy for the KNN classification in approach 1 (cluster 0).

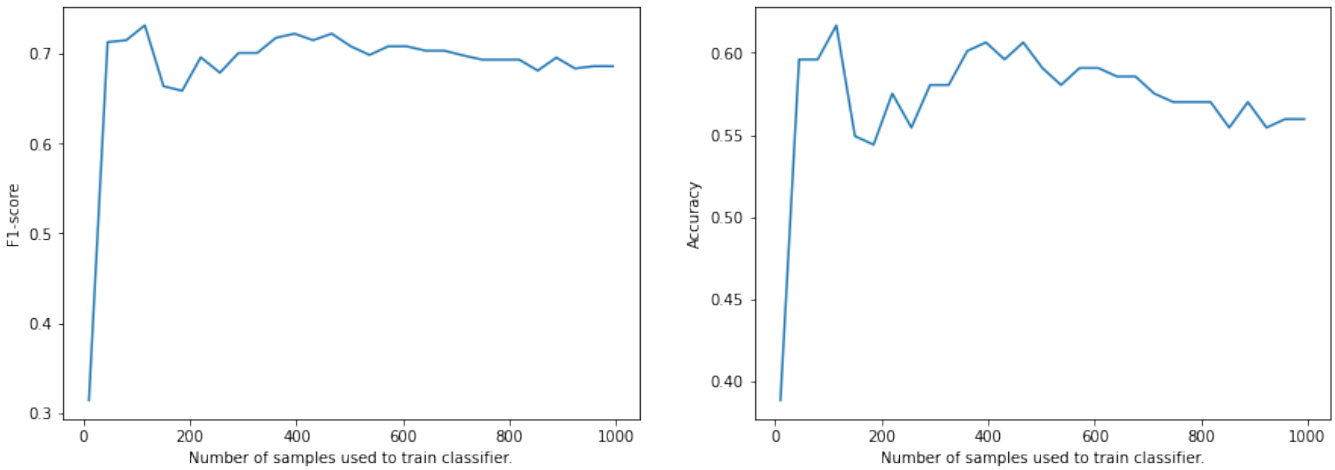


Figure 17. Evolution of F1 score and accuracy for the KNN classification in approach 1 (cluster 1).

Table 13. Table of metrics for the support vector classification in approach 1 (cluster 1).

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.720	0.500	0.000	0.837	0.603	0.000
45	0.399	0.583	0.165	0.284	0.339	0.229
80	0.399	0.583	0.165	0.284	0.339	0.229
115	0.404	0.586	0.173	0.294	0.348	0.235
150	0.404	0.581	0.161	0.303	0.352	0.212
185	0.404	0.581	0.161	0.303	0.352	0.212
220	0.435	0.602	0.205	0.363	0.399	0.247
256	0.689	0.597	0.195	0.789	0.683	0.202
291	0.684	0.594	0.187	0.784	0.679	0.193
326	0.679	0.590	0.180	0.780	0.675	0.185
361	0.409	0.579	0.157	0.321	0.365	0.197
396	0.674	0.587	0.173	0.776	0.671	0.176
431	0.674	0.587	0.173	0.776	0.671	0.176
466	0.668	0.583	0.166	0.771	0.666	0.168
502	0.663	0.579	0.159	0.767	0.662	0.160
537	0.658	0.576	0.151	0.763	0.658	0.151
572	0.751	0.640	0.281	0.838	0.734	0.324
607	0.751	0.640	0.281	0.838	0.734	0.324
642	0.751	0.640	0.281	0.838	0.734	0.324
677	0.751	0.640	0.281	0.838	0.734	0.324
712	0.751	0.640	0.281	0.838	0.734	0.324
748	0.756	0.644	0.288	0.842	0.738	0.336
783	0.756	0.644	0.288	0.842	0.738	0.336
818	0.756	0.644	0.288	0.842	0.738	0.336
853	0.756	0.644	0.288	0.842	0.738	0.336
888	0.756	0.644	0.288	0.842	0.738	0.336
923	0.751	0.640	0.281	0.838	0.734	0.324
958	0.751	0.640	0.281	0.838	0.734	0.324
994	0.751	0.640	0.281	0.838	0.734	0.324
1029	0.751	0.640	0.281	0.838	0.734	0.324
1064	0.751	0.640	0.281	0.838	0.734	0.324
1099	0.751	0.640	0.281	0.838	0.734	0.324
1134	0.751	0.640	0.281	0.838	0.734	0.324
1169	0.751	0.640	0.281	0.838	0.734	0.324
1204	0.746	0.637	0.274	0.834	0.730	0.312
1240	0.741	0.633	0.267	0.830	0.725	0.301
1275	0.741	0.633	0.267	0.830	0.725	0.301
1310	0.741	0.633	0.267	0.830	0.725	0.301
1345	0.741	0.633	0.267	0.830	0.725	0.301
1380	0.746	0.637	0.274	0.834	0.730	0.312
1415	0.746	0.637	0.274	0.834	0.730	0.312
1450	0.746	0.637	0.274	0.834	0.730	0.312
1486	0.746	0.637	0.274	0.834	0.730	0.312
1521	0.746	0.637	0.274	0.834	0.730	0.312
1556	0.746	0.637	0.274	0.834	0.730	0.312

Table 14. Table of metrics for the KNN classification in approach 1 (cluster 0).

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.503	0.534	0.067	0.574	0.526	0.061
229	0.414	0.518	0.036	0.408	0.411	0.036
448	0.419	0.520	0.039	0.418	0.418	0.039
667	0.430	0.517	0.034	0.447	0.438	0.033
886	0.454	0.542	0.084	0.473	0.462	0.081

Table 15. Table of metrics for the KNN classification in approach 1 (cluster 1).

Number of samples	accuracy	balancedaccuracy	adjusted balanced accuracy	f1	f1 weighted	matthews
10	0.389	0.542	0.083	0.314	0.352	0.099
45	0.596	0.516	0.031	0.713	0.602	0.030
80	0.596	0.510	0.020	0.715	0.600	0.019
115	0.617	0.530	0.060	0.732	0.619	0.059
150	0.549	0.495	-0.011	0.664	0.566	-0.010
185	0.544	0.491	-0.018	0.659	0.562	-0.017
220	0.575	0.496	-0.009	0.696	0.583	-0.009
256	0.554	0.475	-0.049	0.679	0.565	-0.047
291	0.580	0.499	-0.002	0.701	0.588	-0.002
326	0.580	0.499	-0.002	0.701	0.588	-0.002
361	0.601	0.519	0.038	0.718	0.606	0.037
396	0.606	0.523	0.046	0.723	0.610	0.045
431	0.596	0.510	0.020	0.715	0.600	0.019
466	0.606	0.523	0.046	0.723	0.610	0.045
502	0.591	0.512	0.024	0.708	0.598	0.023
537	0.580	0.505	0.010	0.699	0.589	0.009
572	0.591	0.512	0.024	0.708	0.598	0.023
607	0.591	0.512	0.024	0.708	0.598	0.023
642	0.585	0.508	0.017	0.704	0.594	0.016
677	0.585	0.508	0.017	0.704	0.594	0.016
712	0.575	0.490	-0.020	0.699	0.582	-0.020
748	0.570	0.486	-0.027	0.694	0.577	-0.026
783	0.570	0.486	-0.027	0.694	0.577	-0.026
818	0.570	0.486	-0.027	0.694	0.577	-0.026
853	0.554	0.470	-0.060	0.681	0.563	-0.058
888	0.570	0.481	-0.039	0.696	0.576	-0.038
923	0.554	0.464	-0.072	0.684	0.561	-0.070
958	0.560	0.473	-0.053	0.686	0.567	-0.051
994	0.560	0.473	-0.053	0.686	0.567	-0.051

Table 16. Grid used for finding the best parameter combination for a *Decision Tree Classifier*.

Parameter	**Values**
`min_samples_split`	{2, 5, 10}
`min_samples_leaf`	{1, 2, 4}
`max_features`	{`auto`, `sqrt`}
`max_depth`	{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, `None`}

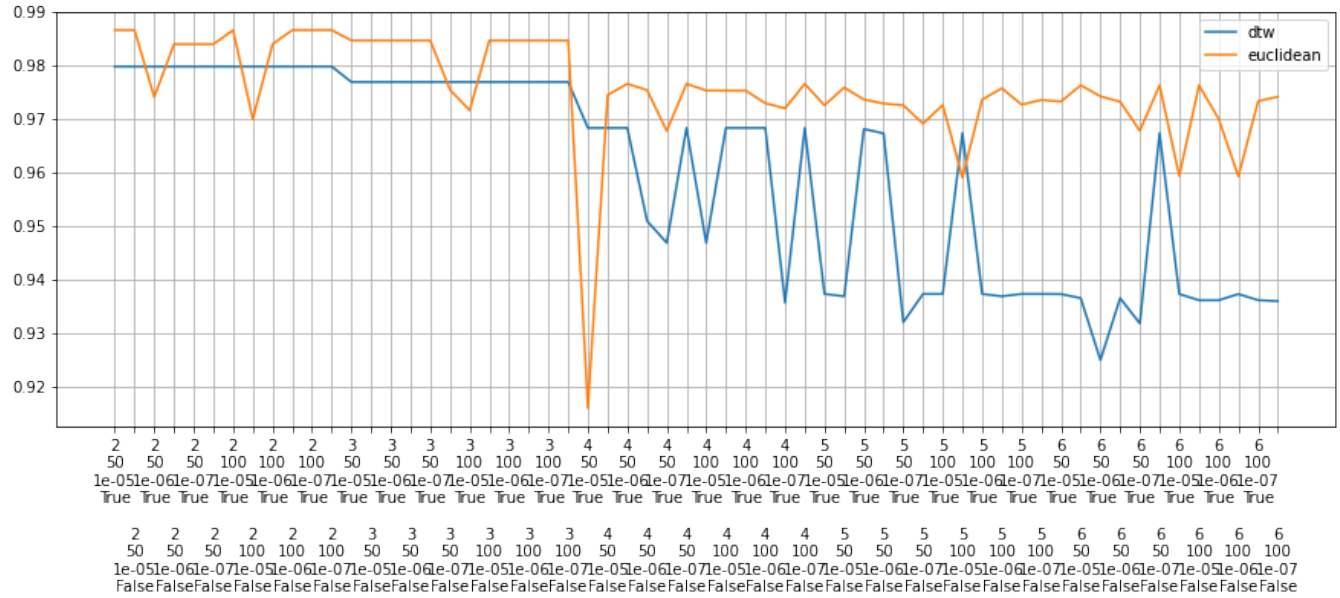


Figure 18. Evolution of the silhouette score with the hyperparameter tuning for the time series KMeans clustering in A2. Parameters respectively are number of clusters, max iterations, tolerance, distance metric, inertia. (Please refer to section 5 for explanations on each parameter).

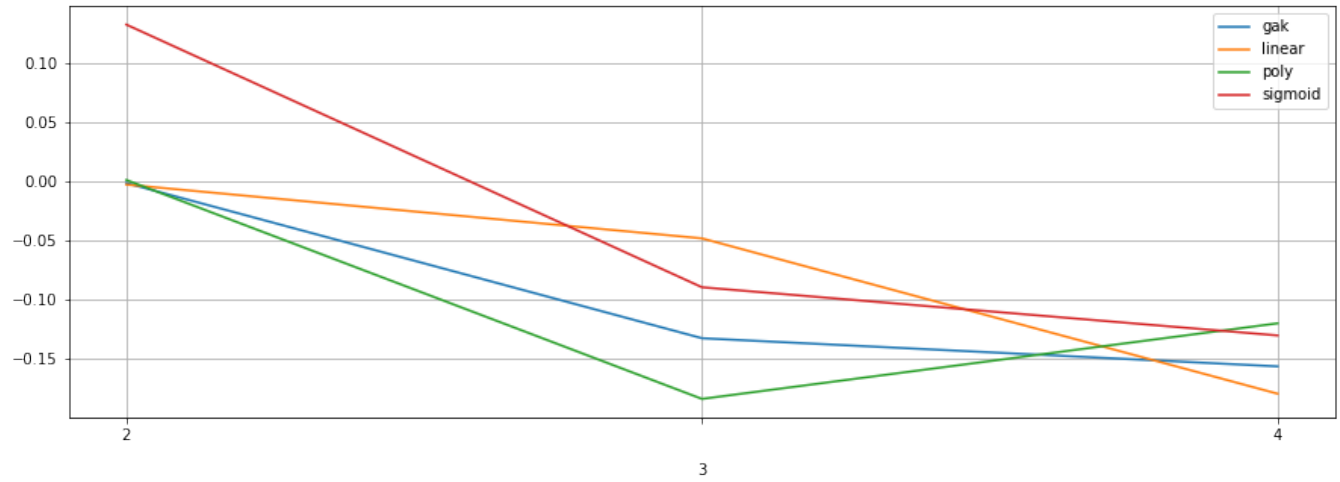


Figure 19. Evolution of the silhouette score with the hyperparameter tuning for the time series Kernel clustering in A2. Parameters respectively are number of clusters. The different lines indicate the different kernels used for the clustering. (Please refer to section 5 for explanations on each parameter).

Table 17. Grid used for finding the best parameter combination for a *Random Forest Classifier*.

Parameter	Values
n_estimators	{200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000}
min_samples_split	{2, 5, 10}
min_samples_leaf	{1, 2, 4}
max_features	{auto, sqrt}
max_depth	{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None}
bootstrap	{True, False}

Table 18. Best parameters combination for a *Decision Tree Classifier* in *Experiment 1*.

Parameter	Value
min_samples_split	5
min_samples_leaf	1
max_features	auto
max_depth	40

Table 19. Best parameters combination for a *Decision Tree Classifier* in *Experiment 2*.

Parameter	Value
min_samples_split	2
min_samples_leaf	1
max_features	sqrt
max_depth	80

Table 20. Best parameters combination for a *Random Forest Classifier* in *Experiment 1*.

Parameter	Value
n_estimators	1400
min_samples_split	10
min_samples_leaf	1
max_features	auto
max_depth	110
bootstrap	False

Table 21. Best parameters combination for a *Random Forest Classifier* in *Experiment 2*.

Parameter	Value
n_estimators	400
min_samples_split	2
min_samples_leaf	1
max_features	auto
max_depth	70
bootstrap	False

Table 22. Accuracy, **balanced accuracy**, **adjusted balanced accuracy**, **F1 score**, **weighted F1 score** and **Matthew's coefficient** for the *Random Forest Classifier* trained in *Experiment 2*.

Samples	Accuracy	Balanced Accuracy	Adj. Balanced Accuracy	F1	Weighted F1	Matthew's coeff.
10	0.592	0.557	0.114	0.695	0.613	0.103
168	0.589	0.568	0.136	0.688	0.612	0.121
327	0.569	0.566	0.132	0.662	0.595	0.117
486	0.583	0.564	0.129	0.681	0.607	0.114
644	0.567	0.560	0.119	0.662	0.592	0.106
803	0.580	0.560	0.120	0.679	0.604	0.107
962	0.561	0.550	0.100	0.658	0.587	0.089

Table 23. Accuracy, **balanced accuracy**, **adjusted balanced accuracy**, **F1 score**, **weighted F1 score** and **Matthew's coefficient** for the *Decision Tree Classifier* trained in *Experiment 1*.

Samples	Accuracy	Balanced Accuracy	Adj. Balanced Accuracy	F1	Weighted F1	Matthew's coeff.
10	0.540	0.544	0.089	0.632	0.567	0.078
168	0.880	0.895	0.789	0.914	0.885	0.732
327	0.965	0.969	0.939	0.976	0.966	0.914
486	0.875	0.880	0.759	0.911	0.879	0.710
644	0.999	0.999	0.999	0.999	0.999	0.998
803	0.960	0.959	0.918	0.972	0.960	0.900
962	1.000	1.000	1.000	1.000	1.000	1.000

Table 24. Accuracy, **balanced accuracy**, **adjusted balanced accuracy**, **F1 score**, **weighted F1 score** and **Matthew's coefficient** for the *Decision Tree Classifier* trained in *Experiment 2*.

Samples	Accuracy	Balanced Accuracy	Adj. Balanced Accuracy	F1	Weighted F1	Matthew's coeff.
10	0.560	0.545	0.089	0.660	0.586	0.079
168	0.573	0.558	0.116	0.671	0.598	0.103
327	0.569	0.557	0.113	0.666	0.594	0.100
486	0.562	0.545	0.091	0.661	0.587	0.080
644	0.553	0.543	0.086	0.650	0.579	0.076
803	0.572	0.559	0.119	0.668	0.597	0.105
962	0.550	0.534	0.068	0.651	0.576	0.060

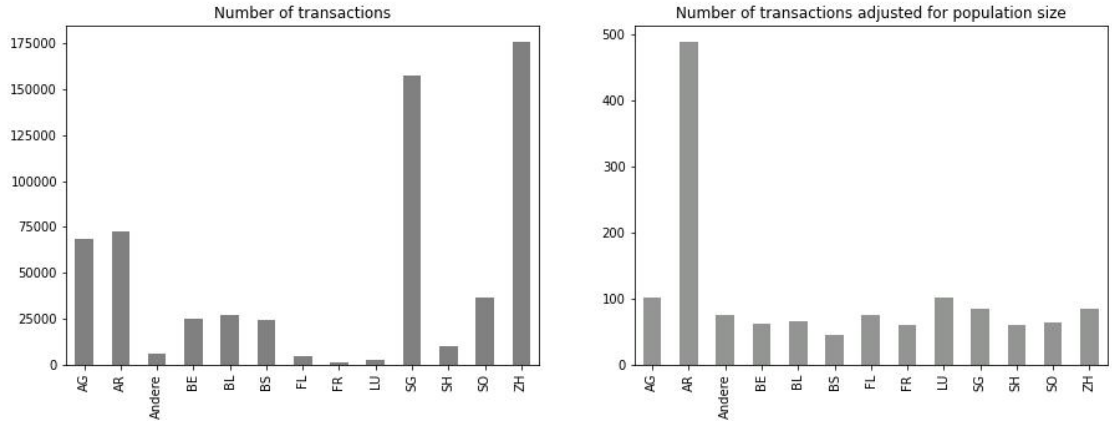


Figure 20. (Left) Number of transactions per canton. (Right) Number of transactions per canton capita.

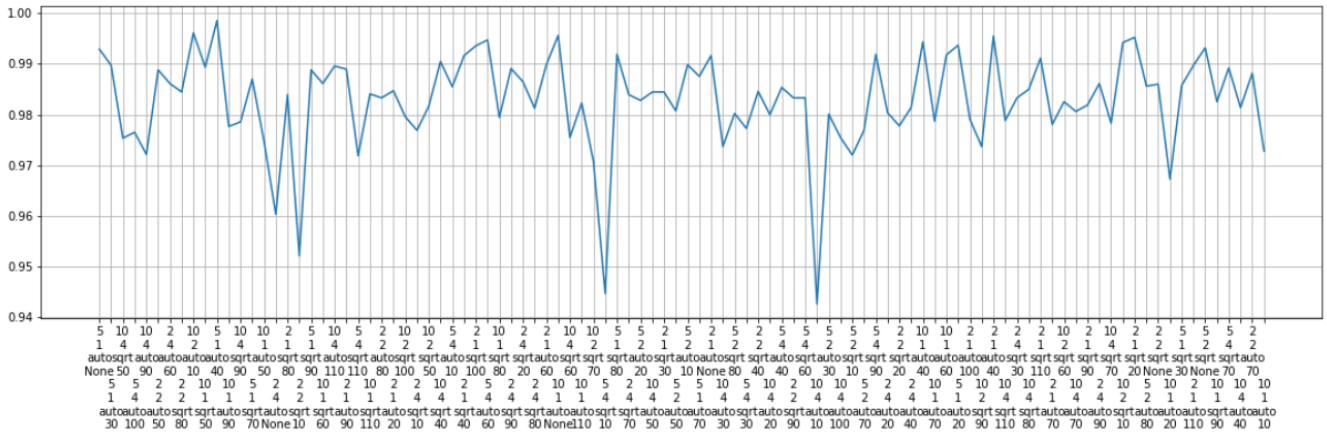


Figure 21. Hyperparameter tunning of a *Decision Tree Classifier* in Experiment 1.

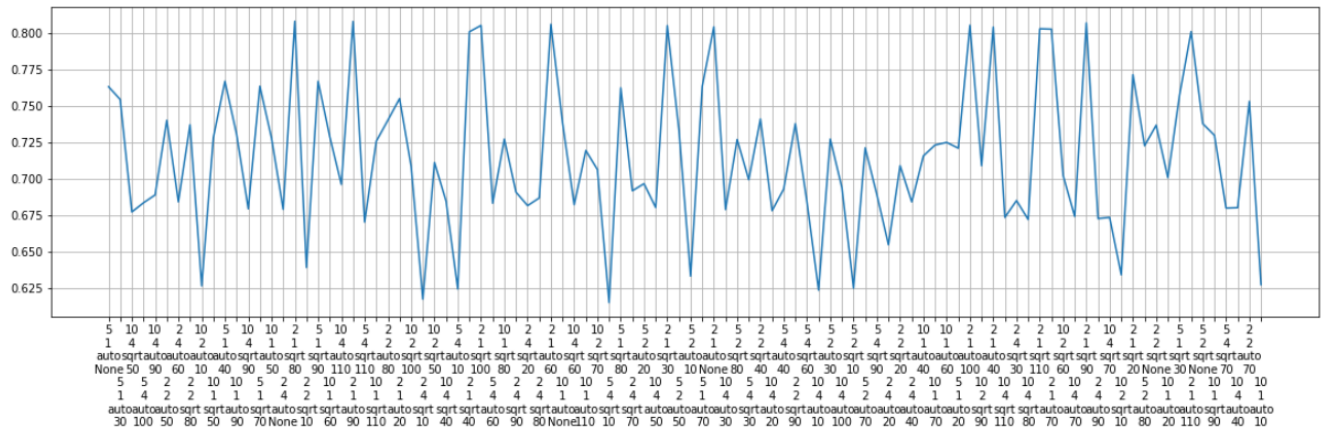


Figure 22. Hyperparameter tunning of a *Decision Tree Classifier* in Experiment 2.

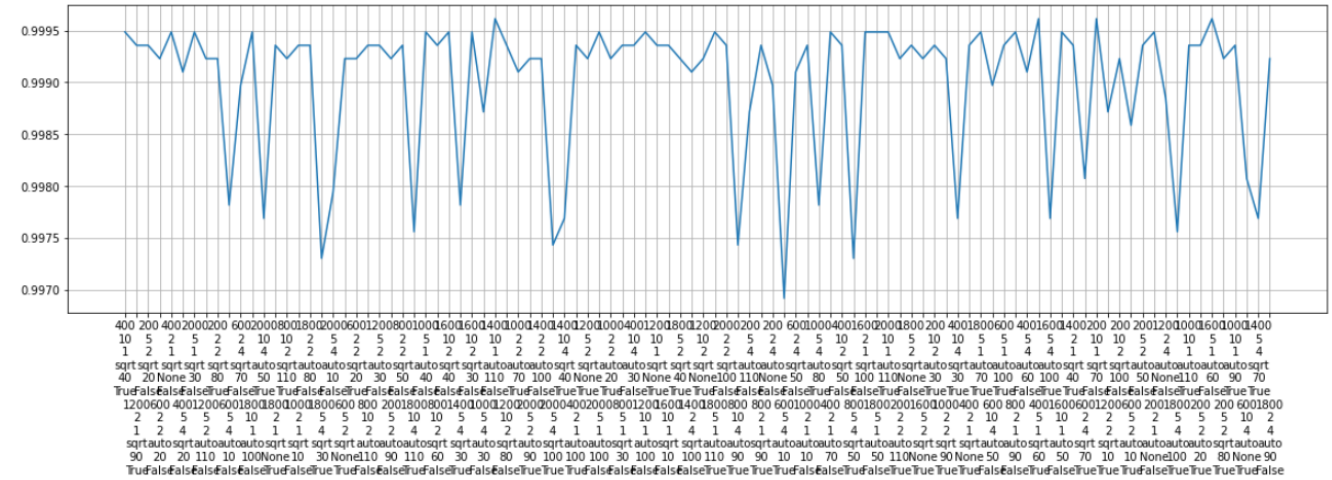


Figure 23. Hyperparameter tunning of a *Random Forest Classifier* in Experiment 1.

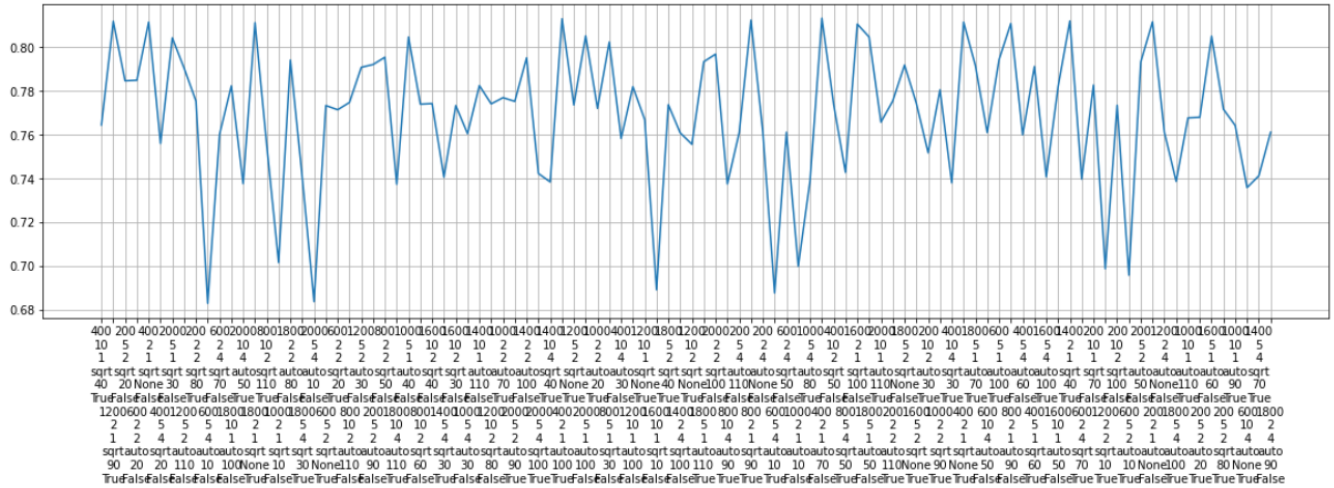


Figure 24. Hyperparameter tunning of a *Random Forest Classifier* in Experiment 2.

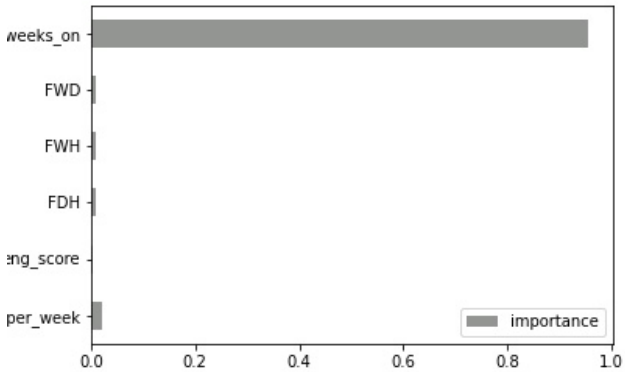


Figure 25. Bar chart of the *feature importance* of the *Decision Tree Classifier* trained in *Experiment 1.*

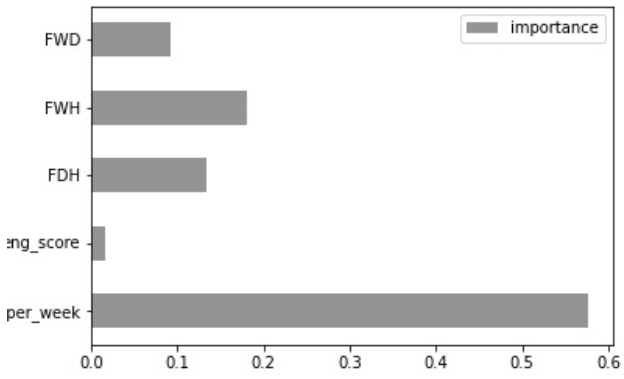


Figure 26. Bar chart of the *feature importance* of the *Decision Tree Classifier* trained in *Experiment 2*.

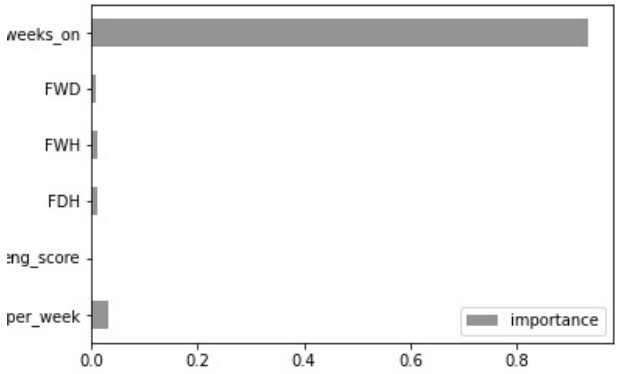


Figure 27. Bar chart of the *feature importance* of the *Random Forest Classifier* trained in *Experiment 1*.

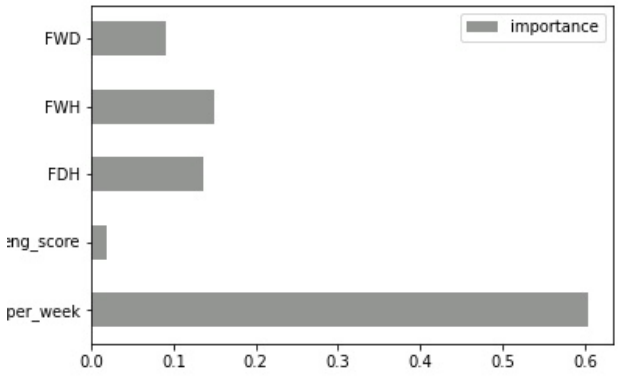


Figure 28. Bar chart of the *feature importance* of the *Random Forest Classifier* trained in *Experiment 2*.

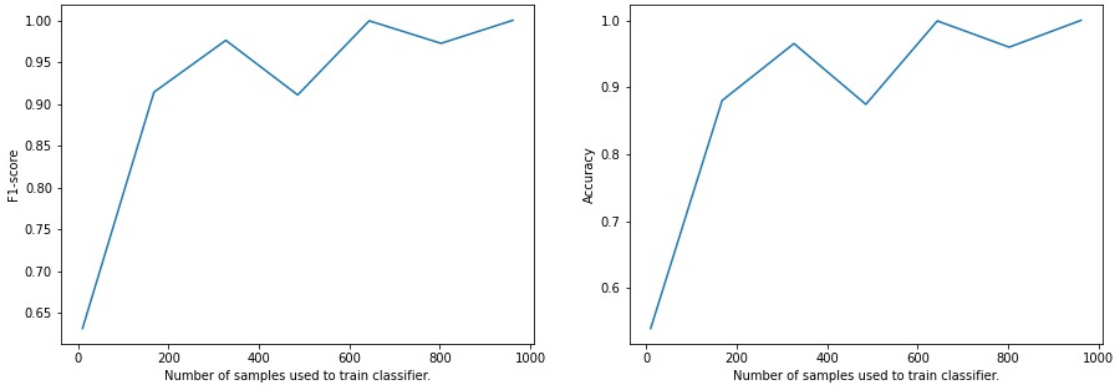


Figure 29. *Accuracy* and *F1-score* against number of samples of the *Decision Tree Classifier* trained in *Experiment 1.*

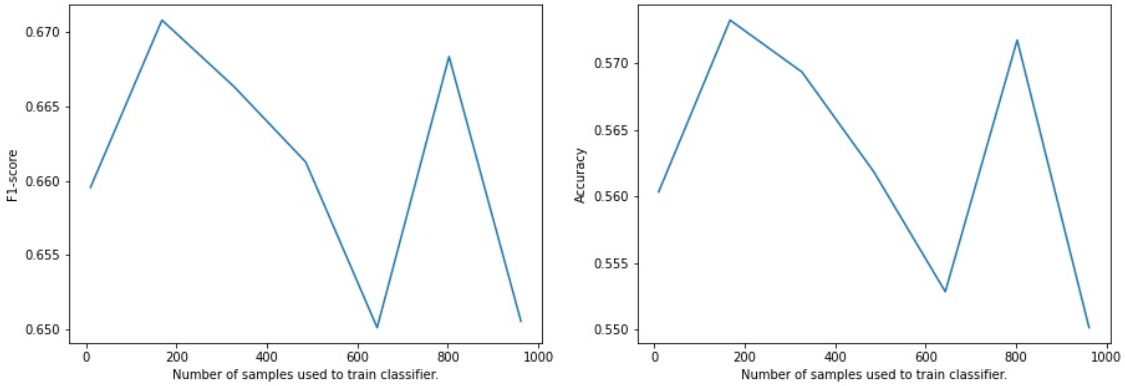


Figure 30. *Accuracy* and *F1-score* against number of samples of the *Decision Tree Classifier* trained in *Experiment 2.*