

Department of Industrial Engineering and Management

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Water Stress Detection in Banana Plants Using Machine Learning**

**Final Report – Semester B, June 2021**

Name: Yotam Baron

Supervisor: Prof. Boaz Lerner

Co-operation: Rahan Meristem

**Table of Contents**

[1. Introduction 5](#_Toc78421438)

[2. Background 7](#_Toc78421439)

[2.1 Concept drift and data streams 7](#_Toc78421440)

[2.1.1 Real and Virtual Concept Drift 7](#_Toc78421441)

[2.1.2 Concept Drift – Change Types 8](#_Toc78421442)

[2.2 Concept drift Models 9](#_Toc78421443)

[2.2.1 General Methods 10](#_Toc78421444)

[2.2.2 Error Based and Adaptive Window Methods 10](#_Toc78421445)

[2.2.3 Ensemble and Classifying Methods 11](#_Toc78421446)

[2.3 Bayesian Networks 11](#_Toc78421447)

[2.4 CDDRL – Concept Drift Detection and Re-Learning 13](#_Toc78421448)

[2.5 Precision Agriculture 14](#_Toc78421449)

[2.5.1 Computer Vision and Current Methods for Detecting Stress in Plants 15](#_Toc78421450)

[2.5.2 Stress in Banana Plants 15](#_Toc78421451)

[3. Planning 17](#_Toc78421452)

[3.1 Experimental Design 17](#_Toc78421453)

[3.2 Photographing 17](#_Toc78421454)

[3.3 Segmentation 19](#_Toc78421455)

[3.4 Registration 20](#_Toc78421456)

[3.5 Feature Extraction 21](#_Toc78421457)

[3.6 Pre-processing 23](#_Toc78421458)

[3.6.1 Missing Values and Smoothing 23](#_Toc78421459)

[3.6.2 Feature Selection 24](#_Toc78421460)

[3.6.3 Discretization 25](#_Toc78421461)

[4. Results and Evaluation 26](#_Toc78421462)

[4.1 CDDRL procedures 26](#_Toc78421463)

[4.2 Competition 27](#_Toc78421464)

[4.3 Performance Measures 28](#_Toc78421465)

[4.4 CDDRL vs. Human Expert 28](#_Toc78421466)

[28](#_Toc78421467)

[4.5 CDDRL vs. Other Algorithms 29](#_Toc78421468)

[4.6 Water Stress Process Evaluation 29](#_Toc78421469)

[5. Conclusion 31](#_Toc78421470)

[6. References 33](#_Toc78421471)

[7. Appendices 36](#_Toc78421472)

[7.1 Appendix A – Extracted Features Graphs 36](#_Toc78421473)

[7.1.1 Geometric Features 36](#_Toc78421474)

[7.1.2 Color Features 36](#_Toc78421475)

[7.1.3 Thermal and Number of Leaves Features 37](#_Toc78421476)

[7.2 Appendix B – In Depth Results CDDRL and Competitors 38](#_Toc78421477)

[7.2.1 CDDRL AUC and FA vs. DR graphs 38](#_Toc78421478)

[7.2.2 FP, FN, Precision, Recall Analysis 39](#_Toc78421479)

[7.2.3 By Plant Analysis 55](#_Toc78421480)

**List of Figures**

[Figure 1: Types of drifts: circles represent instances; different colors represent different classes 8](#_Toc78421481)

[Figure 2: Patterns of change over time (outlier is not concept drift) (Minku et al., 2010) 9](#_Toc78421482)

[Figure 3: BN change [Nielsen & Nielsen, 2008] 12](file:///C:\Users\yotam\Desktop\yotam\לימודים\שנה%20ד\פרוייקט%20גמר\Final%20Report%20-%20133.docx#_Toc78421483)

[Figure 4: An example of a photo taken by the RGB camera 18](#_Toc78421484)

[Figure 5: photos taken by the multispectral camera (left to right): 550, 670, 700 wavelengths 18](#_Toc78421485)

[Figure 6: photo taken by the depth camera (blue indicates a closer object) 18](#_Toc78421486)

[Figure 7: photo taken by the thermal camera 19](#_Toc78421487)

[Figure 8: (left) the original photo, (right) the cropped photo around the wanted plant 19](#_Toc78421488)

[Figure 9: marked in red are the contours around each leaf made by the segmentation tool 20](#_Toc78421489)

[Figure 10: left image shows the leaves segmentor identifying neighbor leaves, right image 20](#_Toc78421490)

[Figure 11: (left) the ‘Max Perimeter’ feature not smoothed, (right) the same feature smoothed 24](#_Toc78421491)

[Figure 12: Data processing workflow 25](#_Toc78421492)

[Figure 13: The CDDRL process 27](#_Toc78421493)

[Figure 14: DR score over time for CDDRL and expert 28](#_Toc78421494)

[Figure 15: F1 scores over time for all algorithms 29](#_Toc78421495)

[Figure 16: The BNs learned by the CDDRL throughout the experiment days 30](#_Toc78421496)

**Abstract**

The banana is the fourth largest fruit worldwide in terms of production volume. Diseases and stresses in plants such as water stress cause significant loss of crops. Therefore, the task of identifying diseases and stresses in plants, as well as understanding their development process, are critical matters. Using a botanical expert for this purpose, especially at the orchard level, is an expensive and inefficient solution. Therefore, there is a great need in automatic, especially machine learning (ML), tools that can deal with this issue in a cost-effective, fast, and accurate manner.

To build a model that could detect and explain the process of water stress in banana plants, we conducted an experiment, with our close partner Rahan Meristem, where 192 banana plants were randomly divided into four treatment groups and grown in a greenhouse for 41 days. During the first 13 days, all plants received an optimal amount of water (100%). From the 14th day forward, each group of plants received a different amount of water - 100%, 80%, 60%, and 40% of the optimal amount of water. Data extracted from images of the plants taken by RGB, thermal, depth, and multi-spectral cameras was processed by us and used to train an ML model based on a Bayesian network (BN). The BN graphical model was used for the purpose of identifying and analysing the development of water stress in the plants by measuring changes in the connections between variables (nodes) representing plant characteristics.

The performance of our model was compared to two human experts, one classified the plants based on the plants’ images, and the other based on physically inspecting them in the greenhouse. For further comparison, five additional algorithms addressed the same detection problem. Our model started to identify plants in stress from day 20 and the greenhouse expert on day 15. The detection rate of our model was higher than that of the two experts in days 20-27 and showed a similar performance to the image expert and was slightly inferior to the greenhouse expert in days 28-41. The performance of our model was superior according to the F1 measure over three of the five competing algorithms in almost all days between day 20 to 41, identical to one and slightly inferior to the fifth algorithm.

Our model, unlike all other competitors, provided important information on the development process of the plants’ water stress. The stress first affects the temperature of the banana plant, then the size of its leaves, and finally the number of leaves it has and various colour indices. The proposed system – using processed data from images of the plants taken by different cameras and a BN model for predicting and understanding the plant's stress process – can be a worthy replacement for the human expert, and consequently allow the farmer accurate and specific care for plants in a quick and a cost-efficient manner.

**Keywords:** water stress, Bayesian networks, detection rate, biotic and abiotic stress

# 1. Introduction

Many machine learning algorithms assume that the data it is using to generate a model comes from a stable process, meaning that the inputs’ distribution as well as the target value’s distribution stay fixed through time. In the real world this conception is not always correct as concepts are often not stable but change over time. This phenomenon of change in the distribution underlying the data is known as concept drift (Lu et al., 2014). A change can arise for many reasons, e.g., price growth due to inflation and weather prediction rules that may vary radically with the season (Tsymbal, 2004).

Dealing with data streams usually includes dealing with concept drifts as data streams demonstrate continuous activity over long periods of time, and in real-life applications it is natural that the underlying process can change over time, resulting in a concept drift (Mendelson, 2020). To deal with online data streaming and the problem of concept drift, special approaches, different from commonly used techniques, which treat arriving instances as equally important contributors to the final concept are needed (Tsymbal, 2004).

A recent method called CDDRL for dealing with concept drift detection and knowledge representation of the change’s development was introduced by (Mendelson, 2020). The method uses Bayesian Networks and a parameter distance comparison to alert when a change has occurred, after a change was detected the method learns the updated network’s structure in a smart and efficient manner. Since the method uses Bayesian Networks, it can alert not only on when a change occurs, but also on where in the domain it appeared. This characteristic is very useful in understanding the development process of the concept drift. (Mendelson, 2020) shows that the method was found robust to different types of change, e.g., gradual and sudden, as well as when dealing with different kind of data distributions, e.g., skewed and non-skewed. Besides testing the CDDRL with empirical experiments, the method was also tested in a real-life application in the form of a stress detection in Banana plants, as described in (Mendelson et al., 2020).

Stress and diseases of plants, such as water stress, fertilizer stress and various pathogens cause at least a 10% loss of the global food production (Strange & Scott, 2005). Therefore, the task of identifying stress and diseases in plants as well as understanding the development process of the disease in such way that will allow adequate treatment to the plant is a crucial matter. Using a botanic expert for this purpose, especially in the orchard level, is very expensive, inefficient and is admittedly subjective and error-prone (Ghosal et al., 2018). Hence, the use of quick and precise machine learning techniques is destined to take a central role in precision agriculture tasks e.g., detection of plant biotic and abiotic stress, diseases, and optimized cropping time prediction.

In the proposed work frame, an experiment was conducted with the purpose of testing the CDDRL’s performance as a detector of water stress in banana plants and as an explainer of the process in which the plant goes into water stress, using images of the plants taken in their growing stage by different cameras e.g., RGB, depth, thermal and multispectral. A human expert also dealt with the detection problem, once by looking at the plants’ images and again by observing the plants physically in the greenhouse. The CDDRL showed encouraging results in comparison to the human expert and provided interesting insights on the water stress process.

# 2. Background

## 2.1 Concept drift and data streams

A data stream is a data set in which the objects have timestamps, which, depending on the granularity of the stamps, induces either a total or a partial order between observations (Webb et al., 2017). In recent years, advances in hardware technology have facilitated the ability to collect data continuously. Simple transactions of everyday life such as using a credit card, a phone or browsing the web lead to automated data storage. The volume of such data is so large that it may not be possible to store all of it at once (Aggarwal, 2007). Since the data is infinite and cannot be stored for a long time, algorithms dealing with data streams are challenged to be designed in a way that they pass each piece of data only once. Another common characteristic of streaming data is that it may change over time. This phenomena of change in the data’s distribution over time is commonly known as concept drift. The term concept drift can be sub-categorized into two types: a change in the input’s data distribution, known as virtual concept drift and a change in the target variable given the input, known as real concept drift (Gama et al., 2014).

### 2.1.1 Real and Virtual Concept Drift

Lu (Lu et al., 2014) explains the difference between the two terms as follows: If we denote the feature vector as x and the class label as y, then the data stream will be an infinite sequence of (x, y). If the concept drifts, it means the distribution of p(x, y) is changing between the current data chunk and the yet to come data. If we decompose p(x, y) into the following two parts as p(x, y) = p(x) × p(y|x), we could say that there are two sources of concept drift: one is p(x), which evolves with time t, and can also be written as p(x|t), and the other is p(y|x), the conditional probability of feature x. The former source is noted as virtual concept drift and the latter as real concept drift.

(Pesaranghader et al., 2017) further evaluates the impact of the two types of concept drift on the decision boundaries of a classifier as follows: A real concept drift refers to the changes in p(y|X) which affects the decision boundaries or the target concept. On the other hand, virtual drift is the result of a change in p(X), and subsequently in p(X|y), but not in p(y|X). That is, a virtual drift is a change in the distribution of the incoming data which implies that the decision boundaries remain unaffected. Pesaranghader et al. (2017) also note that when dealing with a prediction problem, the classifier only needs to be adapted when a real concept drift is detected, since only a real concept drift changes the classifier decision boundaries. The figure below illustrates this concept. However, if the model has to represent the real underlying data distribution, it also has to be adapted in the case of a virtual concept drift.

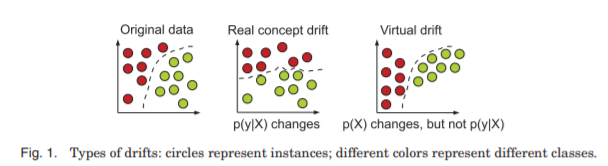


Figure : Types of drifts: circles represent instances; different colors represent different classes

Virtual concept drift and real concept drift can occur separately as well as at the same time. An example of a virtual concept drift is the case of spam categorization, While our understanding of an unwanted message may remain the same over a relatively long period of time, the relative frequency of different types of spam may change drastically with time (Tsymbal, 2004). A typical example of the real concept drift is a change in users’ interests when following an online news stream. While the distribution of the incoming news documents often remains the same, the conditional distribution of the interesting (and thus not interesting) news documents for that user changes (Gama et al., 2014).

Virtual and real concept drifts reflect on the source of the drift; a change in the input distribution or a change in the target function. However, concept drifts can be further categorized into more groups, which will now be discussed.

### 2.1.2 Concept Drift – Change Types

Change types refer to the configuration patterns of the data sources over time. The structural types of change are usually defined based on those configurations (Žliobaitė, 2010). The change types are modified in different ways in the literature, the most common types are sudden (abrupt) and gradual change as (Tsymbal, 2004) defines them.

A sudden drift refers to the scenario where a current concept is suddenly replaced by a different concept, resulting in a sudden change in the data’s distribution. For example, someone graduating from college might suddenly have completely different monetary concerns (Tsymbal, 2004).

A gradual drift refers to a slow transition from the current concept to another concept. In this type of change, there are two active concepts and the probability of sampling from the first concept is slowly decreasing as the probability of sampling from the second concept is increasing accordingly (Žliobaitė, 2010).

There are some extensions in the literature to the gradual concept drift. Stanley (Stanley, 2003) divides gradual drift further into moderate and slow drifts, depending on the rate of the changes. He also gives the example of a slowly wearing piece of factory equipment as a cause of a gradual change in the quality of output parts. Another change type often mixed with gradual change, is known as incremental change. An incremental drift is such that a sequence of data distributions appear during the transition (Pesaranghader et al., 2017). However, the difference between the different data distributions is very small, thus the drift is noticed only when looking at a longer time period, (Baena-García et al., 2006).

The last common change type mentioned in the literature is known as a recurring drift. That is when previously active concept reappears after some time. It differs from common seasonality notion in a way that it is not certainly periodic, it is not clear when the source might reappear (Žliobaitė, 2010). An example of such drift is the demand for ice cream, since the demand goes up every summer and down again after it, but the exact time of the increase in demand is unknown and changes every year.

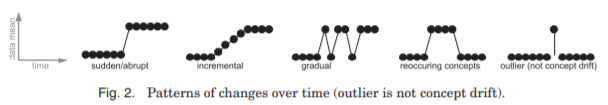
 Finally, a less common approach is to categorize change types into mutually exclusive categories (Minku et al., 2010) based on number of reoccurrences, severity, speed and predictability. In principle the proposed categorization tries to quantify the main aspects of the learner design process into change categorization. However, (Žliobaitė, 2010) argues that the categories cannot be mutually exclusive, because the change frequency count, speed, severity is relative to the length of the subsequence, at which one is looking. In the figure below the different change types are visualized.

Figure : Patterns of change over time (outlier is not concept drift) (Minku et al., 2010)

## 2.2 Concept drift Models

In general, approaches to cope with concept drift can be classified into two categories: the first, approaches that adapt a learner at regular intervals without considering whether changes have really occurred, named blind learners; and the second, approaches that first detect concept changes, and next, the learner is adapted to these changes, these learners are known as informed learners. The adaptive models can be further sub-categorized into static window and dynamic window approaches. Furthermore, there are general approaches that can be placed on any learner, there are approaches that use a specific machine learning model and there are approaches that are based on ensembles. All of these will be reviewed.

### 2.2.1 General Methods

(KIFER et al., 2004) offer a scheme for a general change detection problem in a data stream. They test the difference in distributions between two samples. The first sample includes the first X observations in the data stream, and it remains constant. The second sample includes the last Y observations and is updated with every new observation that arrives. They aim to offer a generalized scheme that makes no assumptions on the form of the underlying distribution, and thus they use a non-parametric test to measure the difference between those two samples and compare it to some threshold . As it may be an excellent scheme for general problems that require no assumptions, it may suffer from some disadvantages if using with a complex parametric model since it will fail to capture the internal relations in the network. It is easy to think of an example in which some conditional distributions of a variable are changed in some way although its marginal distribution remains the same. In this case, the non-parametric test is prone to fail in detecting the change.

Another general approach for drift detection was recently introduced in (Khamassi et al., 2015), they name their method EDIST2. This method is designed to fit every data streaming classifying learner. The method uses the learner’s predictions and monitors the amount of prediction errors between instances in a global adaptive window – increases if no drift was found and decreases otherwise, and instances in the current window. An increase in the number of errors in the current window indicates of a possible concept drift. This approach also guarantees a preset false alarm rate using statistical testing to flag that a change has occurred. The advantage of this approach is that it is general and can be fitted to all stationary assuming classifiers but suffers from some disadvantages since it must constantly have the true labels of the observations which is not always possible, and the increase in the window size might be challenging as the algorithm saves in memory all the instances in it and therefore can lead to extreme memory consumption, especially when dealing with data streaming tasks.

### 2.2.2 Error Based and Adaptive Window Methods

Besides the EDIST2, there are other common methods that use an adaptive window and many of them use error monitoring to detect a possible drift. Some examples of these methods are: Drift Detection Method (DDM) introduced in (Gama et al., 2004), it uses a single window and continuously adjusts it by monitoring the misclassifying average µi and i for each instance i. Then, it compares them to µ min and min reached during stable period in order to detect a drift (Khamassi et al., 2015). DDM was later expended by (Baena-García et al., 2006) to the Early Drift Detection Method (EDDM), this approach assumes that if the distribution of the instances is stationary, the learning model will improve its prediction and the error distance will increase as the number of instances increases. Thus, a significant decrease in the error distance implies a drift. EDDM calculates the average distance between two errors µi and its i and compares them to µ max and max reached when the distribution of distances between errors is maximum. (Khamassi et al., 2015) claim that both the DDM and the EDDM are sensitive to noise and therefore to high false alarm rates due to the correlation of the adaptive window size to the min/max µ and . Another similar and common method is the adaptive windowing (ADWIN) introduced by (Bifet & Gavaldà, 2007), the ADWIN splits the window into two sub-windows using a formula to set the cut point and then compares the average error in each of them to detect change. (Khamassi et al., 2015) found that ADWIN is good for sudden drifts but its performance deteriorates when dealing with gradual concept drifts. Also, since the ADWIN alerts on a drift whenever there is any kind of significant difference in the average error between the two sub-windows, it might wrongly alert on a drift in cases where the average error goes down, as this can be explained by the improvement of the learner due to additional data received and not because of a drift.

### 2.2.3 Ensemble and Classifying Methods

Many concept drift supervised algorithms are based on ensembles. For example, the accuracy-weighted ensemble [AWE;(Wang & Yu, 2002)] weighs each model based on its expected classification accuracy on the test data, the dynamic weighted majority [DWM; (Kolter & Maloof, 2007)] uses such weights to remove models from the ensemble and to add new models based on the global performance of the ensemble, and LNSE [(Elwell & Polikar, 2011)] dynamically weighs the classifiers based on their accuracy in the current compared with past environments. A recent ensemble algorithm is named streaming random patches [SRP;(Gomes et al., 2019)] it combines random subspaces and bagging to classify under concept drift. Although having good results, especially for real concept drift, these algorithms cannot detect the source of the change. For example, in a marketing problem, searching for concept drift changes in customer preferences, the ability to identify what caused changes in preferences, which can be used in future marketing campaigns, is missing (Mendelson, 2020).

## 2.3 Bayesian Networks

A Bayesian network (BN) is a statistical and graphical model that efficiently encodes dependency relations for a large set of variables and allows the identification and understanding of the causality in the problem domain. The structure of a BN is composed of nodes representing the domain variables and directed edges connecting these nodes, representing probabilistic and causal relations between the corresponding variables (Pearl, 2000). Learning the BN structure is NP-hard (Chickering, 2003) which requires substantial computing resources or sub-optimal procedures, but once the structure is learned, the network parameters may easily be learned from the model and data (Heckerman, 2020).

The adaption of BN to new data is also related to sequential updating of the network, regardless of a possible drift. (Friedman & Goldszmidt, 1997) stated that because of errors in model construction and the dynamics of the domain, we cannot afford to ignore the information of new data and suggested sequential updating, which can also overcome errors made in learning the initial model. They offer an incremental learning procedure to learn BN from an online stream of data. As for the part of sequential updating of the network structure, they compare the network structure with its neighbours’ structures every K observations. Meaning, they do not deal with the task of detecting if a change occurred or not and instead, continually update the network whether a change has occurred or not. As mentioned earlier, learning, and updating the network may require considerable computing resources. In order to reduce the computational complexity, (Nielsen & Nielsen, 2008) offer a structure updating procedure (also called structural adaptation) with a change detection method, in which structure updating is performed only if a change is detected. To detect changes, they measure how well each of the last K observations fits the local structure of the monitored node in comparison with the fitness of the previous K measurements. If a significant difference is discovered between these two groups of K measurements, then it is assumed that a change in the network has happened. However, two main limitations arise with this approach. First, the measure of fitness of the local structure may detect nodes that do not change but their local structures do.

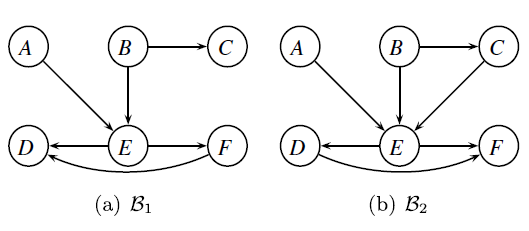


Figure : BN change [Nielsen & Nielsen, 2008]

For example, in Figure 3, (Nielsen & Nielsen, 2008) show that a change is detected for nodes A, C, and E, although A has not changed. Moreover, when an edge is reversed, but the linked nodes keep their Markov blankets, e.g., edge F→D, the change is not detected. Second, a limitation arises from the comparison of the last K observations only with the K observations before them. In the case of a gradual parameter shift, when the parameters are slowly drifted, this method may fail to detect a shift. However, it needs to be noted that gradual parameter-shift detection is not part of the goals of (Nielsen & Nielsen, 2008).

Another recent BN-based approach is locally adaptive-MB-MBC (Borchani et al., 2016), applying the Page-Hinkley test to the average log-likelihood in order to update locally around each node that is detected as changed.

## 2.4 CDDRL – Concept Drift Detection and Re-Learning

The CDDRL method was introduced by (Mendelson, 2020) and is in fact a union of two previous works, the first was done by (Nahum, 2015) and it offers a concept drift detection method using BN. The second work was done by (Cohen, 2018) and it offers an efficient method for re-learning the BN structure after a change was detected, using the first method’s output. Hence, the CDDRL includes two phases: First, the network parameters are learned at each time point t using the recent observations from the data stream. Then, the statistical distances are computed by measuring the distances of network CPDs from their initial values, which are assumed to be correct and serve as a baseline for the period before the change, and then aggregating these distances to a single distance per node. For detecting changes in the BN as revealed by the measured distances, a sequential monitoring methodology is used, which is based on statistical process control (SPC) charts. By this methodology, each node is monitored separately and continuously, which allows to detect not only when a change has occurred but also where in the network it occurred, i.e., which nodes have been affected by the change.

(Nahum, 2015) has tested different types of distance functions, aggregation methods and techniques for setting the process control limits and reported on a preferred combination with another slightly different combination that tradeoff between them in detection and false alarm rates. With respect to the aggregation method, the weighted mean function is preferred, the method gives weights to each distance measure by the proportion of its parent’s configuration and then averages all the distances with the appropriate weights. The SPC chart that out-performed all others was the Exponentially weighted moving average (EWMA), by this method at each time point the distance used for comparison to the change detection threshold is calculated not only with the current distance but also gives weight to past distances with a λ parameter that is suggested to be set between 0.1 and 0.3. Finally, two different distance functions were suggested, the first was the Chi-Squared function yielding the highest detection rate but also higher false alarm rates and the second was the total variation function which yielded slightly lower detection rates accompanied with slightly lower false alarm rates.

The second phase of the CDDRL as proposed by (Cohen, 2018) is a search and score (S&S) methodology exploring the space of possible BN structures. Its input is the BN of the stable process and the nodes which have been detected as changed at each time step (by the first phase). At each time step, the algorithm in the second phase searches for local changes in the nodes which have been detected as changed, thus making the task of learning the BN structure less computationally expensive, as it doesn’t learn the structure from scratch.

Finally, (Mendelson, 2020) combined the two into one framework and conducted experiments to test the algorithm. Mendelson reported that in supervised settings, compared with streaming classification algorithms, e.g., ensemble methods, the CDDRL was slightly better for both sudden and gradual concept drift structures in both real and virtual concept drifts, even though, the CDDRL was not developed for classification as were the other competitors. In unsupervised settings, which the former competitors cannot handle, the CDDRL was significantly superior compared to the other static approaches. The algorithm was also tested in a real-life application: detection of fertilizer stress in banana plants and was found to be as quick a detector as a human expert.

In addition, two versions of the CDDRL were tested in the fertilizer stress experiment: a static version and a dynamic version. The static version keeps the BN that was learned in the preliminary stable process as a reference for comparison to the current time stamp throughout the whole process, with no consideration to weather a change was detected or not. In the dynamic version, the BN that will be compared to the current time stamp is updated and relearned every time a change is detected. Both versions were used in the fertilizer stress experiment, but no significant difference in their performance was observed. The correct usage in each approach is task-dependent and is a subject for further research in different types of concept drift settings.

## 2.5 Precision Agriculture

Plants live in constantly changing environments that are often unfavorable or stressful for growth and development. These adverse environmental conditions include biotic stress, such as pathogen infection and herbivore attack, and abiotic stress, such as drought, heat, cold and nutrient deficiency (Jian-Kang Zhu, 2016). Plant diseases cause immense damage to the agriculture industry—damage that is reflected in productivity and economic aspects. In the USA, crop losses due to plant pathogens are estimated at $33,000,000,000 every year (Savary et al., 2012). One of the causes of such plant diseases is water stress (Osakabe et al., 2014). Water stress is developed in crops when the evaporative demand exceeds the supply of water from the soil (Slayter, 1967). Previous research, e.g., [(González et al., 1999; Sibomana et al., 2013)] has shown that water stress affects the total yield; therefore, early detection of plant stress is critical to minimize the loss of productivity. The severity of the damage depends on the duration between onset and time of detection. At the orchard level, the effectiveness of any remedial measures depends on the timely detection and identification of the cause of stress (Kim et al., 2011).

### 2.5.1 Computer Vision and Current Methods for Detecting Stress in Plants

Traditional machine-learning approaches for plant stress detection are computer-vision based, e.g, [(Seginer et al., 1992); (Kacira et al., 2002)]. These approaches use image analysis tools to extract features to identify stressed plants. Kacira et al. (2002) studied plant movement with top-projected canopy area extracted by machine vision, whereas Seginer et al. (1992) studied the movement of leaf tips to detect the onset of wilt by a computer vision system. Modern machine-learning approaches for plant stress detection are deep-learning-based, e.g, (Ghosal et al., 2018) and they apply deep-neural-networks directly on images to detect stressed plants. (Ghosal et al., 2018) created an algorithm that can classify different types of diseases in soybeans and can give visual explanation on the way the deep neural network algorithm made his classification by revealing its most high-resolution feature maps.

### 2.5.2 Stress in Banana Plants

Early stress detection is crucial in bananas since this is the fourth largest fruit crop in the world and is considered to be a staple food in many countries (Surendar et al., 2013). Stress in banana plants cause visual and non-visual changes to its leaves. Water stress, for example, closes stomata and impedes photosynthesis and transpiration, resulting in changes in leaf color and temperature (Nilsson, 1995). Other symptoms of water stress include morphological changes such as leaf curling or wilting due to loss of cell turgidity (Kim et al., 2011).

Recent research was conducted (Mendelson et al., 2020) on fertilizer stress in banana plants. The experiment monitored four different groups of banana plants each receiving a different amount of fertilizer treatment, with one of the groups receiving no treatment at all. The plants were pictured with an RGB camera for 28 days and features of geometry and color were extracted from the photos using a segment tool to contour the leaves in every image. The study shows that the untreated group developed characteristics that differ from the treated groups in size and color.

(Kim et al., 2011) conducted a water stress experiment in apple trees using various types of cameras to monitor the apple tree’s leaves for water stress detection. (Kim et al., 2011) notes that plant leaves absorb most of the radiance in the visible band by plant pigments but reflect most radiance in the near-infrared (NIR) band. Plant stress changes the reflectance pattern due to an efficiency drop of photosynthetic absorbance and causes reflectance to increase in the visible band and to decrease in the NIR band. Therefore, Kim used a hyperspectral sensor, an active-illuminated spectral sensor, and a digital camera to monitor the leaves and reported that through analysis of indices extracted from the entire spectral wavelengths the stressed plants show an increase of reflectance in the green and red bands and a decrease in the NIR band in comparison with healthy plants.

To sum up, the detection of water stress in plants can be done successfully by using deep learning algorithms as well as by computer vision methods that combine imagery from various cameras, e.g., (thermal, infra-red, RGB and hyper/multi spectral) and analyzing the plant’s leaves with respect to its temperature, pigmentation, reflectance, and shape. A comprehensive review on the different remote sensors (the different cameras) and on the way that they can be used to detect stress and diseases in plants is thoroughly discussed in (Mahlein, 2016).

# 3. Planning

## 3.1 Experimental Design

We conducted an experiment with the purpose of testing the CDDRL algorithm’s performance in a real-life application - detecting water stress in banana plants.

We took 192 pre-mature banana plants and randomly divided them into four separate groups: each consisting of 48 plants. The plants were mixed and located separately inside a greenhouse. The plants were grown in these settings for 41 days; in the first 13 days (the control period) all plants got treated with the same amount of water – 100% of the recommended amount. Starting from day 14 and forward each group of plants received a different amount of water treatment as follows:

* Group A received 100% of the recommended amount.
* Group B received 80% of the recommended amount.
* Group C received 60% of the recommended amount.
* Group D received 40% of the recommended amount and was counted for the ‘stressed group’.

Each plant in each of the 41 days was photographed from above using several cameras (RGB, multispectral, thermal, depth) placed on a moving cart. At the end of the growing period, all photos were collected and features of geometry, color, temperature and depth were extracted and used by the CDDRL and five other data streaming algorithms (KNN-ADWIN, AWE, SRP, LNSE and DWM) to build a time-series model that predicts for each day which plants are from group D (stressed plants) using data from the previous days. Apart from the algorithms, a human expert also gave his predictions in two different manners: once every day during the growing time by physically observing the plants and the second time by evaluating all photos of the plants at the end of the growing period.

## 3.2 Photographing

As mentioned above, the banana plants were photographed from above every day by several cameras all placed on a moving cart. Cameras of different types were used in order to evaluate different aspects of the plants e.g., color, size, radiation. The cameras thar were used:

*RGB camera:* the RGB camera was the most contributing camera. Each pixel in the photo created by the RGB camera consists of three numbers remarking how ‘red’, ‘green’ and ‘blue’ the pixel is. These numbers were later used to extract color indices for every plant. The RGB photos were also used to extract geometrical indices.



Figure : An example of a photo taken by the RGB camera

*Multispectral camera:* the multispectral camera is used to measure the reflectance of the plants in different wavelength, we used the following wavelength: 480, 520, 550, 670, 700, 730, 780 (an image was created for each wavelength). Indices known in the literature to be good predictors of water stress in plants were calculated using the values from the different wavelength images.

Figure : photos taken by the multispectral camera (left to right): 550, 670, 700 wavelengths

*Depth camera:* the depth camera is similar to the RGB, only it can be used to compute the depth of the object it is photographing, meaning how far the object is from the camera. This measure helped us to compute the height of each plant, which is important since plants under water stress experience a growing defect that can effect its height.

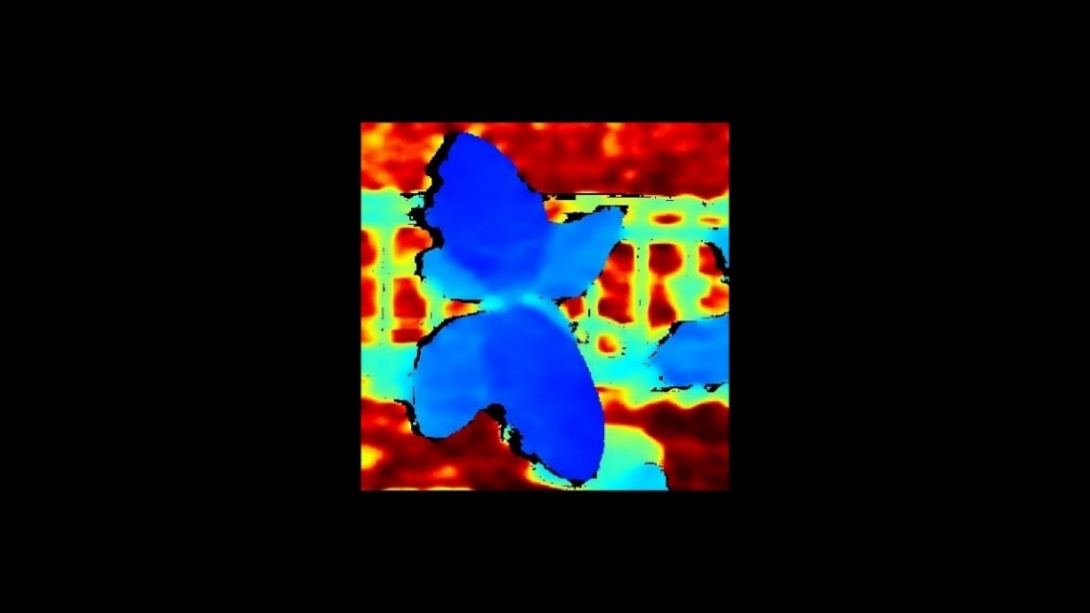


Figure : photo taken by the depth camera (blue indicates a closer object)

*Thermal camera:* the thermal camera was used to obtain each plant’s temperature value. The temperature of the plant (in particular its leaves) is changed by the amount of water that the plant contains, thus can be a good indicator of a plant experiencing water stress.

A picture containing text, window, white, tub

Description automatically generated

Figure : photo taken by the thermal camera

## 3.3 Segmentation

 Since the plants were placed in the greenhouse relatively close to each other and the cameras used had a broad lens, neighbouring plants that are not the photographed plant entered the frame. Therefore, in the first stage we cropped each photo to center it around the wanted plant as can be seen in the following example:

Figure : (left) the original photo, (right) the cropped photo around the wanted plant

Since all features extracted from the different cameras mentioned above were computed by the values of the plants’ leaves, we wanted to be able to know which part of the photo is a leaf and which is the background. Therefore, with the help of the Phenomix, we developed a segmentation tool based on a CNN algorithm and trained it using a train set of RGB images of plants. The training set of images was created by pasting single cut out leaves of different size and shape on background images in a position that recreates an artificial plant. With respect to the inference part of the segmentor, the algorithm took as input a plant’s RGB image and produced text files each with the contour points of every leaf in the photo. An example of the segmentor’s work is presented below:

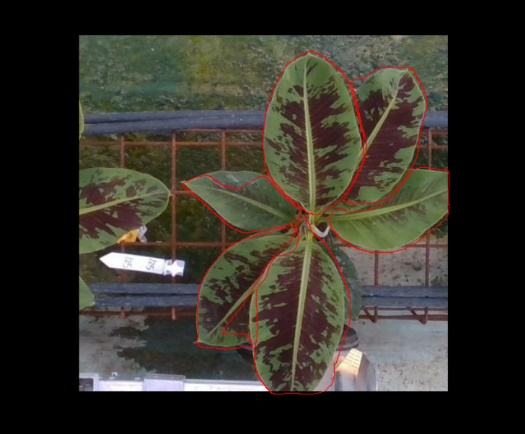


Figure : marked in red are the contours around each leaf made by the segmentation tool

The leaf segmentor suffered from a few issues that needed to be dealt with; it had a hard time identifying leaves that were small/hidden/curled, these issues are planned to be dealt with in the near future by re-training the segmentor. However, a critical issue that we were able to deal with was that the segmentor wrongfully identified leaves of neighbouring plants. To deal with this issue a plant segmentor was trained and used. The plant segmentor identifies the whole plant in the image and not each of its leaves, hence making it robust to the neighbouring leaves problem. At the feature extraction stage, information was extracted from leaves that were identified by the leaves segmentor only if they were also placed inside the contours of the plant segmentor. An example of the neighbouring leaves problem and its solution is presented below:

A picture containing text, plant, vegetable

Description automatically generatedA picture containing plant, green, palm, vegetable

Description automatically generated

Figure : left image shows the leaves segmentor identifying neighbor leaves, right image

shows the plant segmentor identifying the whole plant and ignoring the neighbor leaves

## 3.4 Registration

As explained above, we used different cameras to photograph each plant. The cameras were placed on the same cart next to each other, resulting in that the plant photographed was placed in a slightly different position in each of the photos. In addition, the images from the different cameras are not the same size – different resolution. Since we wanted to use the same contours of the leaves drawn from the RGB photos by the segmentation tool to extract features from the other channels (thermal, depth, multispectral), we needed to align all images. For that reason, a registration algorithm was used. The RGB photos were the baseline for this algorithm; all photos from the other channels were then resized to the size of the RGB photos and then repositioned in a way that every pixel of the plant in them was shifted to the position of the equivalent pixel in the RGB photo, thus reflecting the same information. After applying registration, the contours of the segmentation tool can be used to extract values from the plants’ leaves in all the different photos.

## 3.5 Feature Extraction

After applying registration on all photos and after the contours of the leaves were produced by the segmentation tool, features were extracted from the different type of photos. In addition, the depth camera was used to identify the two recently grown leaves of every plant, this was done by marking the two leaves in each plant that had the smallest distance to the depth camera. The new leaves contain valuable information about the plant’s condition and hence features were extracted for the whole plant as well as for each of its two newest leaves, marked as *min leaf* and *2nd min leaf*.

*Geometrical Features:* one of the aspects that plants under water stress differ from normal plants is in shape. The lack of water supply to the leaves results in smaller and distorted leaves. Hence, we extracted the following geometrical features:

* Mean size, max size, std size: for each plant we calculated the area of every one of its leaves, the average area of the plant’s leaves was noted as the ‘Mean size’. The largest leaf area was noted as the plant’s ‘Max size’ and the standard deviation of the leaves’ areas was noted as the plant’s ‘Std size’. The area of the min and 2nd min leaves was also calculated separately.
* Mean perimeter, max perimeter, std perimeter: the length of the contours around each plant’s leaves was calculated to be that plant’s perimeter. Once again, the average perimeter of the plant’s leaves became its ‘Mean Perimeter’, the leaf with the largest perimeter became the plant’s ‘Max perimeter’ and the standard deviation of the leaves’ perimeters became the plant’s ‘Std perimeter’. The perimeter of the min and 2nd min leaves was also calculated separately.
* Mean angle, max angle, std angle: since water stressed plant’s leaves get distorted and crooked, we wanted to account for that a curvature feature. The angles between every three points in each leaf’s contour was calculated and an average angle was computed for every leaf as well as the maximum and standard deviation of angles of each leaf. As before, averaging the leaves angles, taking the leaf with the maximum angle, and calculating the standard deviation of the leaves’ angles, gave us the angle features for every plant – ‘Mean angle’, ‘Max angle’, ‘Std angle’. Regarding the min and 2nd min leaves, their average and maximum angles as well as the standard deviation of the angles were noted.

*Color Features:* another aspect of water stressed plants is the transformation in color, therefore we evaluated the leaves’ color and computed four color indices. All indices were also computed separately for the min and 2nd min leaves:

* Visible Atmospherically Resistant Index (VARI): is designed to emphasize vegetation in the visible portion of the spectrum, while mitigating illumination differences and atmospheric effects. It is ideal for RGB images; it utilizes all three color bands and is calculated as follows:

*VARI = (Green - Red)/ (Green + Red - Blue)*

We calculated the VARI index for every pixel of every leaf, averaged the measure for every leaf over the pixels, and then averaged the score over the leaves, calculated the standard deviation and the range to get – ‘Mean VARI’, ‘Max VARI’, ‘Std VARI’, ‘Range VARI’.

* Excess green index (EGI): Another color index measuring greenness and is calculated by:

*EGI = 2 \* Green - Blue – Red*

For this index we also extracted the - ‘Mean EGI’, ‘Max EGI’, ‘Std EGI’, ‘Range EGI’.

* Normalized Difference Index (NDI): is widely applied for monitoring vegetation, the same features were extracted - ‘Mean NDI’, ‘Max NDI’, ‘Std NDI’, ‘Range NDI’ and it is calculated as follows:

*NDI = (normalized red – normalized green) / (normalized red + normalized green)*

* Hue: another color measure that uses a different color scale, we used it to extract six features – ‘Mean green’, ‘Mean std green’, ‘Mean range green’, ‘Std mean green’, ‘Std std green’, ‘Std range green’.

*Thermal and Depth Features:* regarding the thermal features, we extracted the temperature of every pixel in the plant’s leaves and noted the average leaf’s temperature as the ‘Avg temp’, the maximum temperature as ‘Max temp’, and the average temperature in the lowest third of the temperature histogram of every leaf as ‘Third temp’. Standard deviations of the leaf’s temperature, the third temperatures and between leaves temperatures were calculated as well. Same features were calculated for the min and 2nd min leaves separately. Important to mention, all temperatures were normalized with the temperature measured inside the greenhouse at the time the image was taken. For example, if the plant had an average temperature of 30 degrees extracted from its image and the greenhouse temperature was 25 degrees at the time the image was taken, then the plant’s temperature was written as 30 – 25 = 5 degrees.

The depth of the plant is a measure of the plant’s height. For every leaf of a plant, the center pixel was found and the average depth of a 10-pixel square around it was noted as that leaf’s depth, the average depths of every plant’s leaves was our ‘Depth’ feature. As explained before, the two leaves with the minimum depth in every plant, were noted as the min leaf and 2nd min leaf of the plant.

*Number of Leaves:* Using both segmentors (plant segmentor and leaves segmentor), we were able to calculate for each plant how many leaves it has. The leaves segmentor’s output are text files that contain the contours of each leaf it identified in the plant, therefore we counted how many text files the segmentor produced for each plant, taking only these that their contours fall also inside the contours found by the plant segmentor and writing this calculation as the number of leaves.

*Multispectral Features:* As explained before, a multispectral camera takes photos in different wavelengths, we had photos in 480, 520, 550, 670, 700, 730 and 780 wavelengths. There are endless measures that were presented in the literature for water stress detection, we used the following measures that were reported to be good indicators and use our wavelengths. The leaf pigments indices:

* Modified Chlorophyll Absorption Ratio Index *(MCARI)* - calculated by:

*[(Blue700-Blue670) – 0.2\*(Blue700-Blue550)] \*Blue700/Blue670*

* Transformed Chlorophyll Absorption in Reflectance Index *(TCARI)* - calculated by:

*3\*(Blue700-Blue670) – 0.2\*(Blue700-Blue550) \*(Blue700/Blue670)*

* Carotenoid Reflectance Index 2 *(CRI2)* - calculated by:

*1/Blue550-1/Blue700*

- The last multispectral feature is a broadband greenness measure – *Optimized Soil-Adjusted Vegetation Index (OSAVI)* – and is calculated by:

*1.16\*(Blue780-Blue670)/(Blue780+Blue670+0.16)*

Eventually, these indices were not used, since the images taken from the multispectral camera were not reliable. The multispectral camera’s calibration process was not done as needed, resulting in low quality images that are much less reliable. In future experiences, the photographing process with the multispectral camera should be done more carefully to gain important information about the reflectance of the plants.

## 3.6 Pre-processing

### 3.6.1 Missing Values and Smoothing

For some plants, the segmentation tool was not able to identify any leaves, hence not allowing us to extract features for that plant in that specific day. Plants that had no data for two consecutive days were completely removed from the dataset. Plants missing values of only one day were filled with the average of the same plant’s values from the day before and the day after. Plants missing values of the first day, were filled with the values of that plant in the second day, and those missing the last day were filled with the value of the previous day.

After handling missing values, since the data was noisy, we smoothed every feature of every plant by a moving average with a window size of 3. Starting from the third day, all features of every plant were averaged with the previous values of the last two days of the same plant, that way we ‘cleaned’ the data and reduced the noise. An example of a feature before and after smoothing:

Chart

Description automatically generatedChart

Description automatically generated

Figure : (left) the ‘Max Perimeter’ feature not smoothed, (right) the same feature smoothed

### 3.6.2 Feature Selection

After extracting all 71 features and smoothing the data, a dimensionality reduction was needed. For that purpose, the dataset was processed through a random forest classifier with the default parameters, separately for each day. We ran the data through the classifier 100 times and every time the features’ importance of every day was saved in an array. Eventually, each feature got an importance score – its average feature importance over the 100 replications over all days. In addition, graphs showing the features by treatment over the days were observed, and those that showed good separation between the treatments were noted. Eventually, 16 features were selected, taking the combination of features with the highest feature importance and those found to be good separators between the treatments, the following features were selected:

*‘Max perimeter ’ , ‘Std perimeter’, ‘Max size ’, ‘Std size’, ‘Mean green 2nd min leaf’, ‘Mean Mean green’ , ‘Mean std green’, ‘Std green 2nd min leaf’, ‘Std std green’, ‘Mean VARI’ , ‘Mean VARI 2nd min leaf’ , ‘Range VARI min leaf’, ‘Norm all plant avg third temp’ , ‘Norm all plant min temp ’, ‘Norm avg third 2nd min leaf temp’, ‘Num of leaves’*

Graphs showing features extracted over time by the treatments can be found in appendix A.

### 3.6.3 Discretization

The CDDRL algorithm which is based on a Bayesian Network and on the monitoring of statistical distances between parameters, hence prior to using the algorithm, the data needs to be discretized so the CDDRL can handle it. A static method - meaning the discretization was done on the complete dataset all days at once – was used, the method discretized the data based on the entropy of the target variable. All plants from scheme D were labeled as class ‘1’ (water stressed) and all plants from the other schemes (A, B, C) were labeled as class ‘0’ (not stressed). Another discretization method – Equal bins with 3 and 4 bins – was tried. The bins cut points were determined separately for each day, taking only plants from Scheme A (healthy plants) as the range of data for the cut points. The equal bins method was found slightly inferior in these settings to the entropy method and therefore was not used here eventually.

Finally, the workflow is described in the figure below. The classification and evaluation parts will be discussed next:

Figure : Data processing workflow

# 4. Results and Evaluation

## 4.1 CDDRL procedures

The next step following the pre-processing of the data was to test our CDDRL algorithm with the task of predicting in a data streaming manner which plants are water stressed (from scheme D) and which are not.

The CDDRL has two main stages. In the first stage the CDDRL continuously monitors the network’s parameters (the probabilities of each node’s value given its parents) and alerts that a change has occurred if the statistical distance between these probabilities exceed the threshold calculated in the initial period. In this experiment the first three days were used as the initial period. We used the first three days to learn the initial network using the MMHC algorithm. Then the UCL (Upper control limit) or the ‘change threshold’ was calculated. For every day in the initial period the statistical distance using the ‘Chi-Squared’ distance was measured between every node’s parameters in the current day and the node’s parameters in the initial network. The ‘Chi-Squared’ distance is measured by the equation:

Where K stands for the number of parents’ configurations the node has, stands for the observed parameters set of the Kth parents configuration (the current parameters) and stands for the expected parameters set of the Kth parents configuration (the initial network parameters).

Since this measure gives us K distances for each node, an aggregation method of these distances is applied using a weighted mean – each distance coming from the Kth parents configuration is weighted by its proportional appearance in the data, resulting in one aggregated distance per node per day. The average and standard deviation of the distances measured through the initial period are calculated and an upper control limit is determined separately for each node using EWMA:

Where is a hyperparameter used to set the UCL width and is also the weight given to current distances in contrast to past distances. In our case, was chosen to be 0.8.

The second stage of the CDDRL is learning the BN structure that best fits the current data, using information from the first stage about the changed nodes.

Having the change threshold (UCL), the CDDRL learns only the parameters of the network using data from the current day and the day before (the network structure is kept unchanged). The parameters learned at the first time step (the first and second days) are compared to the parameters’ values in the initial network learned, all nodes that their distance exceed the change threshold are noted and the right network structure is learned by the CDDRL in an efficient way, by only searching a sub space of graphs that are constructed according to the nodes that have been detected as changed. In the preceding time steps, the same process is applied except that the initial network is replaced with the best learned BN from the previous time step – the current parameters are learned on the previous best BN structure and are compared to the best previous BN parameters. The output of the CDDRL are the best learnt BNs for each day of the experiment. The CDDRL process is summarized in the figure below:

Figure : The CDDRL process

The last step of the process was to take the data of each day t and make a prediction using the best learned network by the CDDRL in day t-1. The prediction task was to predict whether the plant is ‘1’ from scheme D or ‘0’ otherwise, making it an unbalanced task 1:3. Therefore, a prediction threshold of 0.3 was used, that way the probability of every plant in each day to be from scheme D was calculated using its Markov-blanket, and if that probability was above 0.3, that plant was labeled as water stressed.

## 4.2 Competition

To get a sense of how well our algorithm performed, we compared it to five other competing data streaming known algorithms - KNN-ADWIN, AWE, SRP, LNSE and DWN. The main goal of the experiment was to see if the CDDRL can be as fast and as good a predictor as a human expert which is an expensive resource. Therefore, an expert also gave his prediction to the same plants, once by evaluating the plants’ photos and the second time by evaluating the plants themselves in the greenhouse.

## 4.3 Performance Measures

With respect to performance measures, as this is an imbalanced task, the F1 score was used for comparison between the CDDRL and the five other algorithms:

F1

Recall

The detection rate (DR) was used to compare the CDDRL to the human expert. The DR measure means how many plants that were actually water stressed were also predicted as water stressed.

## 4.4 CDDRL vs. Human Expert

## Chart, line chart Description automatically generated

Figure : DR score over time for CDDRL and expert

Figure 14 shows the detection rate (DR) of the CDDRL (in blue), the image-based expert (in solid green) and the greenhouse-based expert (in dashed green). Our model started to identify plants in stress from day 20, the greenhouse expert on day 15 and the image expert on day 9, however any detection before day 13 is a false alarm since all plants received 100% of water until this day. The detection rate of our model was higher than that of the two experts in days 20-27 and showed a similar performance to the image expert and was slightly inferior to the greenhouse expert in days 28-41. It is important to notice that the greenhouse-based expert has an advantage over the CDDRL for two main reasons; first, the greenhouse expert can physically observe the plant – touch it, look at it from all angles – the CDDRL does not have this privilege. Secondly, the expert might remember the position inside the greenhouse of plants he already detected as stressed in previous days and might not review it every day like the CDDRL does.

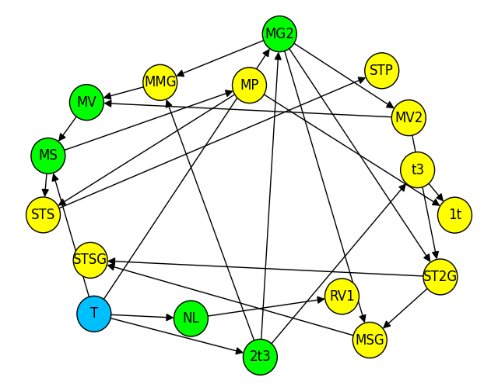
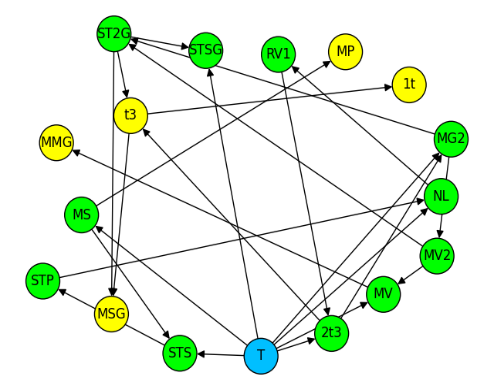
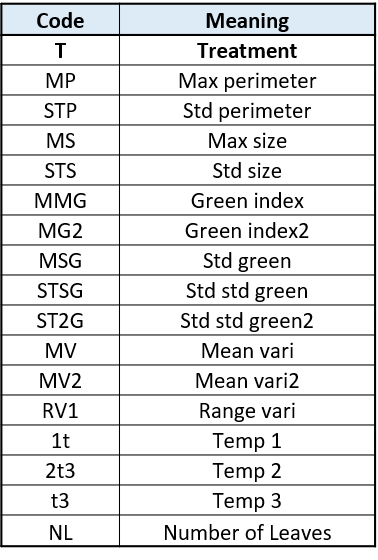
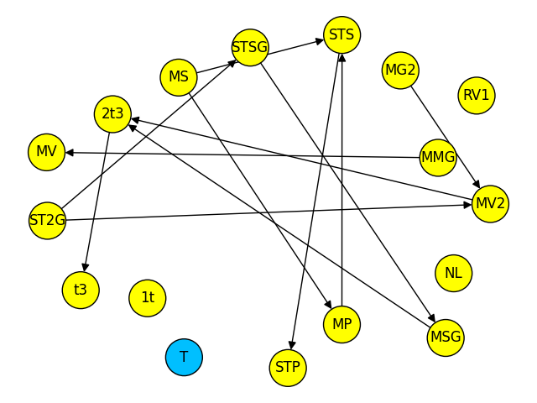
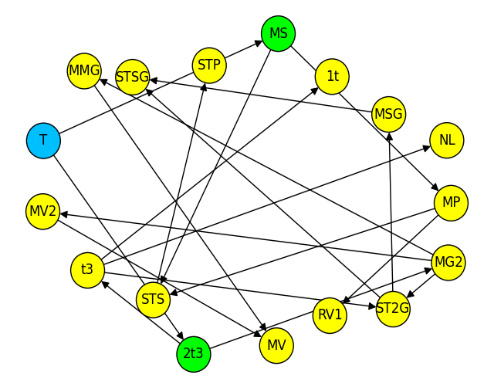
## Chart, line chart Description automatically generated4.5 CDDRL vs. Other Algorithms

Figure : F1 scores over time for all algorithms

Figure 15 shows the F1 score of both the CDDRL (in solid blue) and the five other competing algorithms (in dashed colors). The CDDRL as mentioned before, detects stressed plants starting from day 20. From day 20 until the end of the experiment, the CDDRL’s general performance is higher than three of the competing algorithms (AWE, DWM and LNSE), similar to SRP and slightly lower than the KNN-ADWIN. Important to mention, all algorithms except for the CDDRL have false alarms before day 13, as they identify stressed plants even though there aren’t any in that period. In addition, all competing algorithms do not show improvement from the stable period (until day 13) to the stressed period (after day 13). In depth analysis of all algorithms performances and explanations for the discussed issues can be found in Appendix B. An important advantage of the CDDRL over all other competitors, is the ability to graphically explain the process in which a plant goes into water stress.

## 4.6 Water Stress Process Evaluation

The CDDRL learns the BN structure that best fits the data of the current day, hence learned 41 BNs each corresponding to an experiment day. The BN structure shows connections between the different variables in the domain, thus can be evaluated over time and can provide information about the evolving effect of lack of water on the plant’s characteristics. The figure below shows this process:



**BN\_36:**

**BN\_20:**

**BN\_26:**

**BN\_1:**

Figure : The BNs learned by the CDDRL throughout the experiment days

Figure 16 shows the water stress process of the plants through the experiment days. Each node is a variable in the domain, the ‘T’ node colored in blue is the ‘treatment’ variable, meaning weather the plant is from treatment D or not. The green nodes are those who are in the treatment node’s markov blanket, meaning they are affected by it. In BN\_1 (top left – the BN learned in the first day) no nodes are affected by the treatment, corresponding with the reality that all plants received the same treatment in that period. Going forward to BN\_20 (top right), the treatment the plant got started to affect its leaves maximum size (MS) and its 2nd min leaf’s average coolest third temperatures (2t3). Observing day 26 (BN\_26 bottom left), we can see that besides affecting the plant’s temperature and size, the treatment also starts to affect the number of leaves the plant has (NL) and two color measures: the VARI index (MV), and the greenness of the 2nd min leaf (MG2). Eventually, BN\_36 (bottom right) shows that besides all mentioned variables, the treatment now affects most of the plant’s characteristics including the standard deviation of the plant’s size, perimeter, overall greenness, and the 2nd min greenness (STS, STP, STSG, ST2G accordingly).

# 5. Conclusion

In this framework, we suggest a complete pipeline for replacing a human expert in the task of identifying water stress in banana plants. The pipeline includes two main parts: the first is data collection, where every day the plants are photographed by several cameras e.g., depth, thermal, RGB. The cameras can be placed on a moving cart or on a small drone. In the second part of the pipeline, the photos taken are processed e.g., applying the registration and segmentation processes, extracting relevant indices, cleaning, and smoothing the data. Then, the data is streamed into the CDDRL and a stress prediction is generated for each plant.

The prediction can be made in two different forms, the first is by using already learnt BNs that match the detection task in hand, for example we can take the BNs learnt for each growing day in our experiment and use them to make predictions in a new similar domain – a farmer growing banana plants. The second method to make a prediction is to construct new BNs in a similar way done in our experiment. The BN learnt in day t-1 will be used to make a prediction in day t. An approach that combines the two methods can also be thought of, for example we can at first use BNs from other similar domains (method 1) to make predictions, meanwhile, learning the BNs best suiting the current domain (method 2) and slowly integrating them as predictors.

The CDDRL was found to be in this experiment a worthy competitor to the human expert as well as to other state-of-the-art algorithms with respect to the water stress detection task. The CDDRL also provided us interesting information about the water stress process of the plant – first affecting the plant’s temperature and size, then the number of leaves it has and some color indices and finally some more color and geometrical measures. Although not falling in performance from all other competitors, we still expect to receive better results from the CDDRL and some future work is needed to be done in order for this to happen.

In the future, we plan to explore a number of directions with the target of improving the results of the CDDRL and the whole detection process. First, the segmentation tool will be retrained with leaves that are hidden/small/curled, hopefully improving the leaves segmentation process, and providing us valuable and accurate data about the plants. Second, we are working on a few improved versions of the CDDRL e.g., adding a tabu list for the BN structure search, learning the parameters using a different method, updating the UCLs in every timestamp and more. These new versions will be tested on the current experiment and on other domains and will hopefully show an improvement.

Finally, conducting similar experiments, but observing different types of plants and diseases/stresses, will improve the suggested pipeline and will allow it to be suitable for a larger range of problems. In these future experiments, the multispectral cameras should be properly used, as they can provide crucial information about the water stress process of the plant and will for sure improve the detection performance. Going in these directions and making the proposed pipeline more robust and effective, can bring great contribution to the precision agriculture world.

# 6. References

Aggarwal, C. C. (2007). An Introduction to Data Streams. *Data Streams*, 1–8. https://doi.org/10.1007/978-0-387-47534-9\_1

Baena-García, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavaldà, R., & Morales-Bueno, R. (2006). Early Drift Detection Method. *4th ECML PKDD International Workshop on Knowledge Discovery from Data Streams*, *6*, 77–86. https://doi.org/10.1.1.61.6101

Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. *Proceedings of the 7th SIAM International Conference on Data Mining*, 443–448. https://doi.org/10.1137/1.9781611972771.42

Borchani, H., Larrañaga, P., Gama, J., & Bielza, C. (2016). *Mining multi-dimensional concept-drifting data streams using Bayesian network classifiers*. *20*, 257–280. https://doi.org/10.3233/IDA-160804

Chickering, D. M. (2003). Optimal structure identification with greedy search. *Journal of Machine Learning Research*, *3*(3), 507–554. https://doi.org/10.1162/153244303321897717

Cohen, M. (2018). *Concept drift in machine learning - detection and relearning*.

Elwell, R., & Polikar, R. (2011). Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, *22*(10), 1517–1531. https://doi.org/10.1109/TNN.2011.2160459

Friedman, N., & Goldszmidt, M. (n.d.). *Sequential Update of Bayesian Network Structure*.

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3171*, 286–295. https://doi.org/10.1007/978-3-540-28645-5\_29

Ghosal, S., Blystone, D., Singh, A. K., Ganapathysubramanian, B., Singh, A., & Sarkar, S. (2018). An explainable deep machine vision framework for plant stress phenotyping. *Proceedings of the National Academy of Sciences of the United States of America*, *115*(18), 4613–4618. https://doi.org/10.1073/pnas.1716999115

Gomes, H. M., Read, J., & Bifet, A. (2019). Streaming random patches for evolving data stream classification. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2019*-*Novem*(February 2020), 240–249. https://doi.org/10.1109/ICDM.2019.00034

González, A., Martı́n, I., & Ayerbe, L. (1999). Barley yield in water-stress conditions. *Field Crops Research*, *62*(1), 23–34. https://doi.org/10.1016/s0378-4290(99)00002-7

Heckerman, D. (2020). A Tutorial on Learning With Bayesian Networks. *ArXiv*, *1995*(November). https://doi.org/10.1007/978-94-011-5014-9\_11

Jian-Kang Zhu. (2016). Abiotic stress signaling and responses in plants. *Cell*, *167*(3), 313–324. https://doi.org/10.1016/j.cell.2016.08.029.Abiotic

J. Gama, I. Zliobait˙e, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. ACM computing surveys (CSUR), 46(4):1–37, 2014.

Kacira, M., Ling, P. P., & Short, T. H. (2002). Machine vision extracted plant movement for early detection of plant water stress. *Transactions of the American Society of Agricultural Engineers*, *45*(4), 1147–1153. https://doi.org/10.13031/2013.9923

Khamassi, I., Sayed-Mouchaweh, M., Hammami, M., & Ghédira, K. (2015). Self-Adaptive Windowing Approach for Handling Complex Concept Drift. *Cognitive Computation*, *7*(6), 772–790. https://doi.org/10.1007/s12559-015-9341-0

KIFER, D., BENDAVID, S., & GEHRKE, J. (2004). Detecting Change in Data Streams. *Proceedings 2004 VLDB Conference*, 180–191. https://doi.org/10.1016/b978-012088469-8/50019-x

Kim, Y., Glenn, D. M., Park, J., Ngugi, H. K., & Lehman, B. L. (2011). Hyperspectral image analysis for water stress detection of apple trees. *Computers and Electronics in Agriculture*, *77*(2), 155–160. https://doi.org/10.1016/j.compag.2011.04.008

Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, *8*, 2755–2790.

Lu, N., Zhang, G., & Lu, J. (2014). Concept drift detection via competence models. *Artificial Intelligence*, *209*(1), 11–28. https://doi.org/10.1016/j.artint.2014.01.001

Mahlein, A.-K. (2016). Present and Future Trends in Plant Disease Detection. *Plant Disease*, *100*(2), 1–11. https://doi.org/10.1007/s13398-014-0173-7.2

Mendelson, S. (2020). *Concept-drift detection and re-learning using Bayesian networks*.

Mendelson, S., Halbersberg, D., & Learner, B. (2020). *Fertilizer Stress in Banana Plants*. https://doi.org/10.4271/510048

Minku, L. L., White, A. P., & Yao, X. (2010). The impact of diversity on online ensemble learning in the presence of concept drift. *IEEE Transactions on Knowledge and Data Engineering*, *22*(5), 730–742. https://doi.org/10.1109/TKDE.2009.156

Nahum, L. (2015). *Concept-drift detection in Bayesian networks by parameter sequential monitoring*. *March*.

Nielsen, S. H., & Nielsen, T. D. (2008). Adapting Bayes network structures to non-stationary domains. *International Journal of Approximate Reasoning*, *49*(2), 379–397. https://doi.org/10.1016/j.ijar.2008.02.007

Nilsson, H. E. (1995). Remote sensing and image analysis in plant pathology. *Annual Review of Phytopathology*, *33*, 489–527. https://doi.org/10.1146/annurev.phyto.33.1.489

Osakabe, Y., Osakabe, K., Shinozaki, K., & Tran, L. S. P. (2014). Response of plants to water stress. *Frontiers in Plant Science*, *5*(MAR), 1–8. https://doi.org/10.3389/fpls.2014.00086

Pesaranghader, A., Viktor, H. L., & Paquet, E. (2017). McDiarmid drift detection methods for evolving data streams. *ArXiv*, *October*.

Savary, S., Ficke, A., Aubertot, J. N., & Hollier, C. (2012). Crop losses due to diseases and their implications for global food production losses and food security. *Food Security*, *4*(4), 519–537. https://doi.org/10.1007/s12571-012-0200-5

Sibomana, I. C., Aguyoh, J. N., & Opiyo, a M. (2013). Water stress affects growth and yield of container grown tomato (Lycopersicon esculentum Mill) plants. *Bangladesh Journal of Agricultural Research*, *2*(4), 461–466.

Slatyer, R. O., & Markus, D. K. (1968). Plant-water relationships. *Soil Science*, *106*(6), 478.‏

Stanley, K. O. (2003). Learning Concept Drift with a Committee of Decision Trees. *Computer Science Department University of Texas*, *Cdc*, 1–17. http://scholar.google.com/scholar?hl=en&q=Learning+Concept+Drift+with+a+Committee+of+Decision+Trees&btnG=Search&as\_sdt=2000&as\_ylo=&as\_vis=0#0

Strange, R. N., & Scott, P. R. (2005). Plant disease: A threat to global food security. *Annual Review of Phytopathology*, *43*(Figure 1), 83–116. https://doi.org/10.1146/annurev.phyto.43.113004.133839

Street, C. (2003). by Judea Pearl. *Econometric Theory*, *19*, 675–685.

Surendar, K. K., Devi, D. D., Ravi, I., Krishnakumar, S., Kumar, S. R., & Velayudham, K. (2013). Impact of Water Deficit on Photosynthetic Pigments and Yield of Banana Cultivars and Hybrids. *Plant Gene and Trait*, *4*(4), 17–24. https://doi.org/10.5376/pgt.2013.04.0004

Tsymbal, A. (2004). The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*.

Wang, H., & Yu, P. S. (2002). *Mining Concept-Drifting Data Streams using Ensemble Classiﬁers\_kdd\_03\_526.pdf*.

Webb, G. I., Lee, L. K., Goethals, B., & Petitjean, F. (2017). Understanding concept drift. *ArXiv*, *June*.

Žliobaitė, I. (2010). *Learning under Concept Drift: an Overview*. 1–36. http://arxiv.org/abs/1010.4784

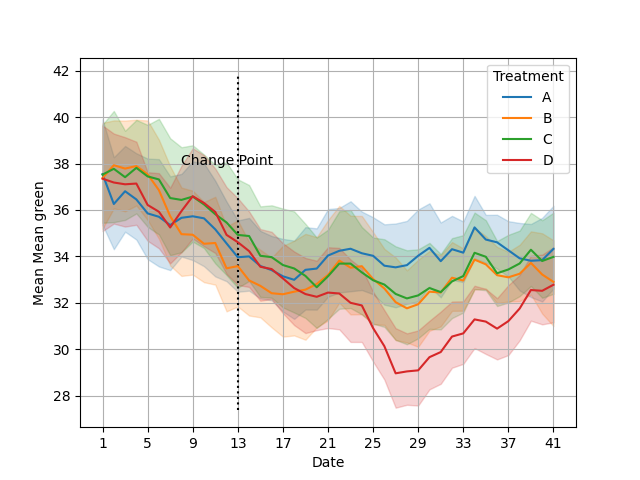
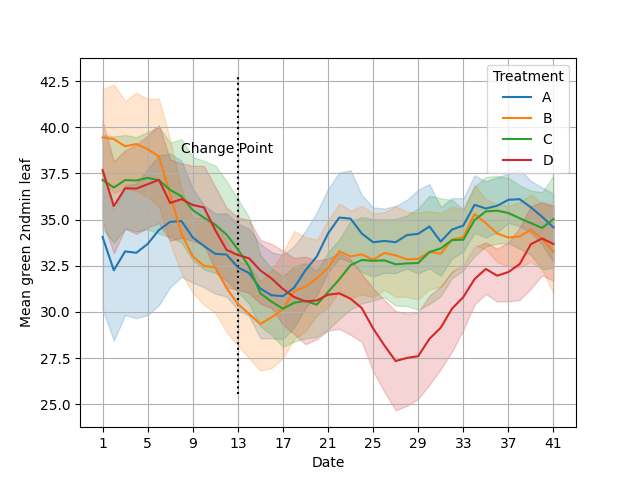
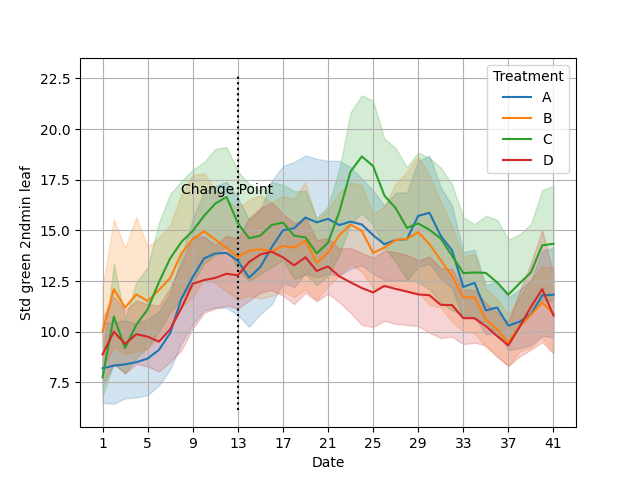
# 7. Appendices

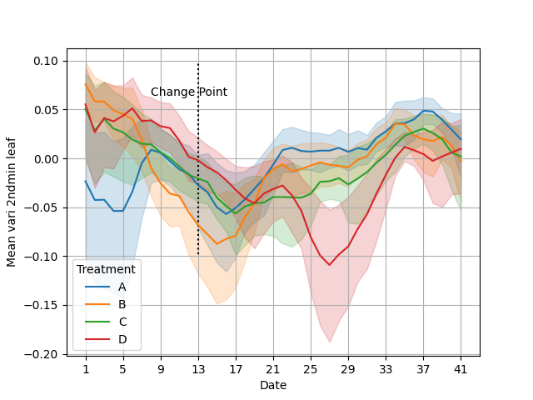
## 7.1 Appendix A – Extracted Features Graphs

The following part shows graphs of the 16 features extracted. The graphs show the features’ value through the days of the experiment separately for each treatment group.

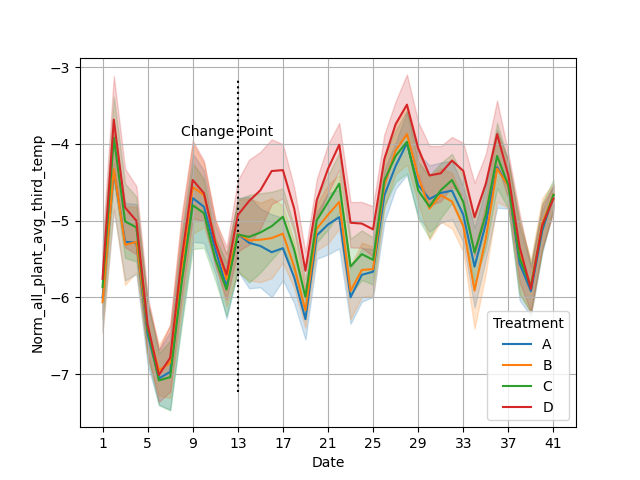
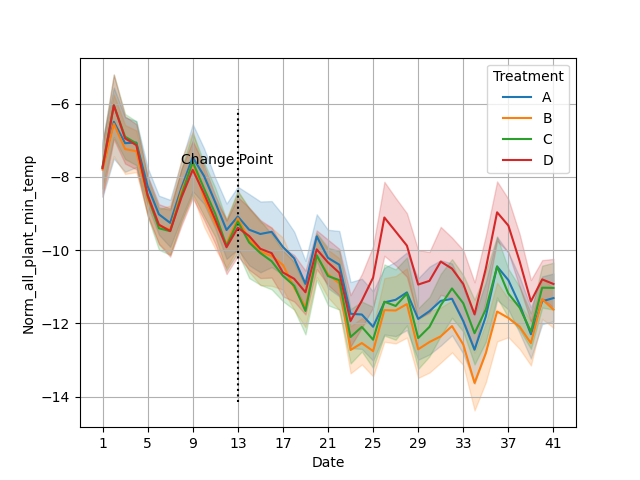
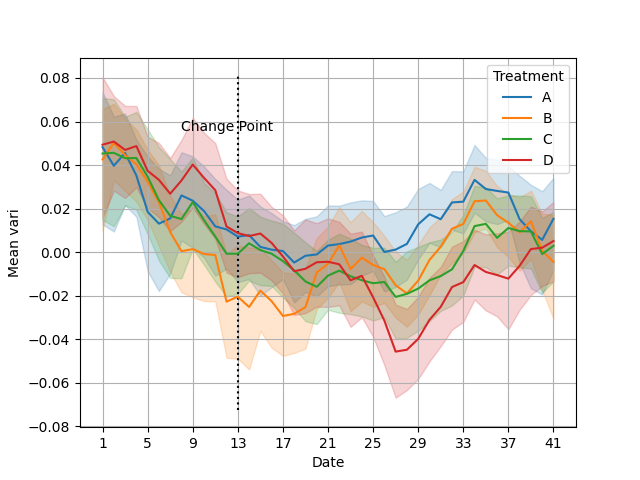
### 7.1.1 Geometric Features

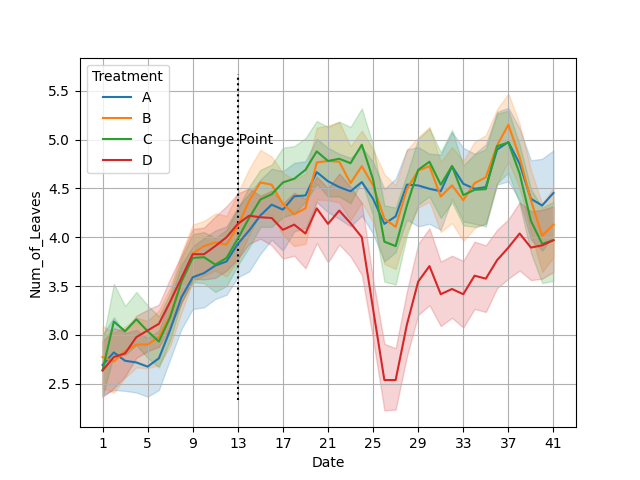
### 7.1.2 Color Features





### 7.1.3 Thermal and Number of Leaves Features



**

## 7.2 Appendix B – In Depth Results CDDRL and Competitors

In this section further results will be shown and discussed.

### 7.2.1 CDDRL AUC and FA vs. DR graphs

Chart, line chart

Description automatically generated

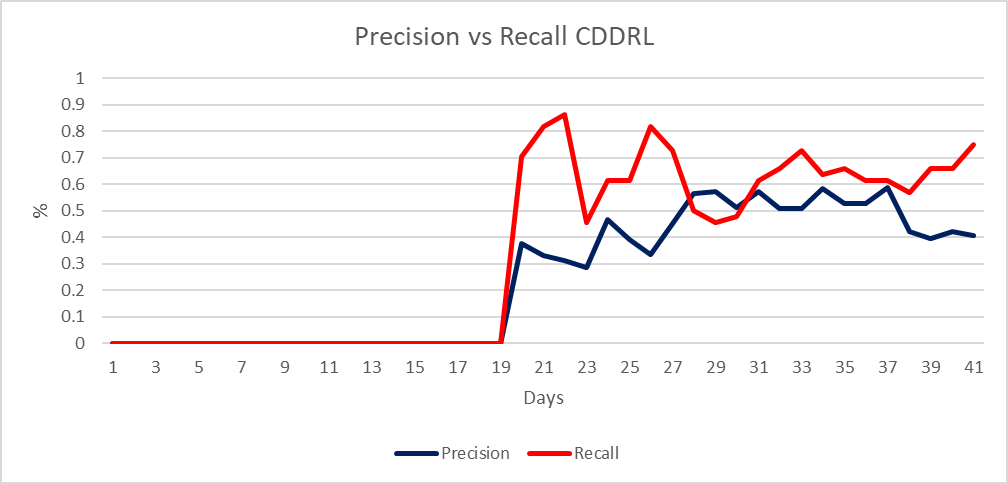
*Chart, line chart

Description automatically generated*

### 7.2.2 FP, FN, Precision, Recall Analysis

#### 7.2.2.1 CDDRL Analysis

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **3** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **4** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **5** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **6** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **7** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **8** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **9** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **10** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **11** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **12** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **13** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **14** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **15** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **16** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **17** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **18** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **19** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **20** | 31 | 51 | 13 | 76 | 0.70 | 0.40 | 0.38 | 0.70 | 0.49 |
| **21** | 36 | 73 | 8 | 54 | 0.82 | 0.57 | 0.33 | 0.82 | 0.47 |
| **22** | 38 | 83 | 6 | 44 | 0.86 | 0.65 | 0.31 | 0.86 | 0.46 |
| **23** | 20 | 50 | 24 | 77 | 0.45 | 0.39 | 0.29 | 0.45 | 0.35 |
| **24** | 27 | 31 | 17 | 96 | 0.61 | 0.24 | 0.47 | 0.61 | 0.53 |
| **25** | 27 | 42 | 17 | 85 | 0.61 | 0.33 | 0.39 | 0.61 | 0.48 |
| **26** | 36 | 71 | 8 | 56 | 0.82 | 0.56 | 0.34 | 0.82 | 0.48 |
| **27** | 32 | 39 | 12 | 88 | 0.73 | 0.31 | 0.45 | 0.73 | 0.56 |
| **28** | 22 | 17 | 22 | 110 | 0.50 | 0.13 | 0.56 | 0.50 | 0.53 |
| **29** | 20 | 15 | 24 | 112 | 0.45 | 0.12 | 0.57 | 0.45 | 0.51 |
| **30** | 21 | 20 | 23 | 107 | 0.48 | 0.16 | 0.51 | 0.48 | 0.49 |
| **31** | 27 | 20 | 17 | 107 | 0.61 | 0.16 | 0.57 | 0.61 | 0.59 |
| **32** | 29 | 28 | 15 | 99 | 0.66 | 0.22 | 0.51 | 0.66 | 0.57 |
| **33** | 32 | 31 | 12 | 96 | 0.73 | 0.24 | 0.51 | 0.73 | 0.60 |
| **34** | 28 | 20 | 16 | 107 | 0.64 | 0.16 | 0.58 | 0.64 | 0.61 |
| **35** | 29 | 26 | 15 | 101 | 0.66 | 0.20 | 0.53 | 0.66 | 0.59 |
| **36** | 27 | 24 | 17 | 103 | 0.61 | 0.19 | 0.53 | 0.61 | 0.57 |
| **37** | 27 | 19 | 17 | 108 | 0.61 | 0.15 | 0.59 | 0.61 | 0.60 |
| **38** | 25 | 34 | 19 | 93 | 0.57 | 0.27 | 0.42 | 0.57 | 0.49 |
| **39** | 29 | 44 | 15 | 83 | 0.66 | 0.35 | 0.40 | 0.66 | 0.50 |
| **40** | 29 | 40 | 15 | 87 | 0.66 | 0.31 | 0.42 | 0.66 | 0.51 |
| **41** | 33 | 48 | 11 | 79 | 0.75 | 0.38 | 0.41 | 0.75 | 0.53 |



**Change Point**

The CDDRL is the only algorithm not to warn of stressed plants (those that belong to treatment D) during the stable period. This result is desirable since all plants received the same treatment during this period and should not have gone into water stress. The CDDRL correctly identifies no connections between the treatment node and any other node during the stable period, thus making its predictions based on the a-priori probabilities. Since the a-priori probability of belonging to treatment D is a quarter of the whole plants (0.25) and the threshold for identification is 0.3 – the CDDRL doesn’t report of any stressed plants (or that the plant belongs to treatment D).

Moving forward, we can observe that at the first days of detection (20-27), the CDDRL suffers from many false positives and from fluctuations. This could perhaps be explained by that the plants are just starting to get into stress and are still not distinct enough from each other. In the following period, we see a drop in recall on day 28 followed by a general mild increase until the end of the experiment. As far as precision goes, the CDDRL shows a mild but quite steady increase starting from 40% on day 20 and rising to about 60% on day 37. At the final days of the experiment, there is a fall in precision and a rise in the number of false positives, I ascribe this finding to the fact that the segmentor performed poorly trying to identify small, hidden or curled leaves. At this point in the experiment, plants from treatment D had a significant number of such hard to detect leaves, causing the segmentor to peek up only those leaves that are in better shape and eventually making plants from treatment D look more similar to the other plants. This could also be validified looking at the features graphs presented in Appendix A, where we can observe that many characteristics of plants from treatment D become much more similar to plants from other treatments starting from around day 37.

#### 7.2.2.2 ADWIN Analysis

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 3 | 8 | 41 | 119 | 0.07 | 0.06 | 0.27 | 0.07 | 0.11 |
| **3** | 13 | 15 | 31 | 112 | 0.30 | 0.12 | 0.46 | 0.30 | 0.36 |
| **4** | 16 | 2 | 28 | 125 | 0.36 | 0.02 | 0.89 | 0.36 | 0.52 |
| **5** | 17 | 17 | 27 | 110 | 0.39 | 0.13 | 0.50 | 0.39 | 0.44 |
| **6** | 19 | 7 | 25 | 120 | 0.43 | 0.06 | 0.73 | 0.43 | 0.54 |
| **7** | 23 | 6 | 21 | 121 | 0.52 | 0.05 | 0.79 | 0.52 | 0.63 |
| **8** | 19 | 13 | 25 | 114 | 0.43 | 0.10 | 0.59 | 0.43 | 0.50 |
| **9** | 22 | 8 | 22 | 119 | 0.50 | 0.06 | 0.73 | 0.50 | 0.59 |
| **10** | 21 | 13 | 23 | 114 | 0.48 | 0.10 | 0.62 | 0.48 | 0.54 |
| **11** | 30 | 10 | 14 | 117 | 0.68 | 0.08 | 0.75 | 0.68 | 0.71 |
| **12** | 29 | 10 | 15 | 117 | 0.66 | 0.08 | 0.74 | 0.66 | 0.70 |
| **13** | 27 | 9 | 17 | 118 | 0.61 | 0.07 | 0.75 | 0.61 | 0.68 |
| **14** | 24 | 16 | 20 | 111 | 0.55 | 0.13 | 0.60 | 0.55 | 0.57 |
| **15** | 20 | 15 | 24 | 112 | 0.45 | 0.12 | 0.57 | 0.45 | 0.51 |
| **16** | 22 | 10 | 22 | 117 | 0.50 | 0.08 | 0.69 | 0.50 | 0.58 |
| **17** | 16 | 13 | 28 | 114 | 0.36 | 0.10 | 0.55 | 0.36 | 0.44 |
| **18** | 21 | 11 | 23 | 116 | 0.48 | 0.09 | 0.66 | 0.48 | 0.55 |
| **19** | 18 | 4 | 26 | 123 | 0.41 | 0.03 | 0.82 | 0.41 | 0.55 |
| **20** | 23 | 9 | 21 | 118 | 0.52 | 0.07 | 0.72 | 0.52 | 0.61 |
| **21** | 27 | 14 | 17 | 113 | 0.61 | 0.11 | 0.66 | 0.61 | 0.64 |
| **22** | 31 | 9 | 13 | 118 | 0.70 | 0.07 | 0.78 | 0.70 | 0.74 |
| **23** | 25 | 11 | 19 | 116 | 0.57 | 0.09 | 0.69 | 0.57 | 0.63 |
| **24** | 27 | 10 | 17 | 117 | 0.61 | 0.08 | 0.73 | 0.61 | 0.67 |
| **25** | 25 | 9 | 19 | 118 | 0.57 | 0.07 | 0.74 | 0.57 | 0.64 |
| **26** | 33 | 18 | 11 | 109 | 0.75 | 0.14 | 0.65 | 0.75 | 0.69 |
| **27** | 34 | 12 | 10 | 115 | 0.77 | 0.09 | 0.74 | 0.77 | 0.76 |
| **28** | 22 | 9 | 22 | 118 | 0.50 | 0.07 | 0.71 | 0.50 | 0.59 |
| **29** | 20 | 2 | 24 | 125 | 0.45 | 0.02 | 0.91 | 0.45 | 0.61 |
| **30** | 19 | 4 | 25 | 123 | 0.43 | 0.03 | 0.83 | 0.43 | 0.57 |
| **31** | 24 | 2 | 20 | 125 | 0.55 | 0.02 | 0.92 | 0.55 | 0.69 |
| **32** | 21 | 6 | 23 | 121 | 0.48 | 0.05 | 0.78 | 0.48 | 0.59 |
| **33** | 26 | 6 | 18 | 121 | 0.59 | 0.05 | 0.81 | 0.59 | 0.68 |
| **34** | 20 | 5 | 24 | 122 | 0.45 | 0.04 | 0.80 | 0.45 | 0.58 |
| **35** | 25 | 2 | 19 | 125 | 0.57 | 0.02 | 0.93 | 0.57 | 0.70 |
| **36** | 24 | 11 | 20 | 116 | 0.55 | 0.09 | 0.69 | 0.55 | 0.61 |
| **37** | 25 | 7 | 19 | 120 | 0.57 | 0.06 | 0.78 | 0.57 | 0.66 |
| **38** | 17 | 6 | 27 | 121 | 0.39 | 0.05 | 0.74 | 0.39 | 0.51 |
| **39** | 16 | 13 | 28 | 114 | 0.36 | 0.10 | 0.55 | 0.36 | 0.44 |
| **40** | 21 | 18 | 23 | 109 | 0.48 | 0.14 | 0.54 | 0.48 | 0.51 |
| **41** | 24 | 4 | 20 | 123 | 0.55 | 0.03 | 0.86 | 0.55 | 0.67 |

It can be observed that the KNN-ADWIN has some false alarms in the stable period. I will try to explain this finding by explaining the process of the algorithm. At each time step the KNN-ADWIN adds the currently received samples to those saved in the past, then it splits the existing samples into two sub-windows of changing size until it reaches such a split that the absolute difference between the two sub-windows’ means is large enough according to some confidence level. Once finding such a split, the ADWIN algorithm first alerts of a change found and then gets read of the samples from the older window, leaving it with the most updated samples representing the new concept in the data. Upon making a classification, the KNN part of the algorithm comes into play. The algorithm finds the K past observations that are closest (Euclidean) to the sample that is being classified, in our experiment we chose K to be 5. The final classification of a plant is made by a majority vote of the K samples from the last step that are most similar to the observed samples, meaning that if at least 3 of the 5 nearest neighbors are from treatment D than the current sample will be classified as D (stressed). Therefore, even if there is no significant difference between plants in the stable period, plants can still be classified as stressed in that period if most of the nearest neighbors happen to come from treatment D. For this reason, if there’s no difference between the samples in a data chunk, the KNN-ADWIN is basically guessing when forced to make a classification. Changing K to a larger number reduces the false alarms but also harms the global performance of the algorithm in this specific task.

#### 7.2.2.3 AWE Analysis

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **3** | 41 | 107 | 3 | 20 | 0.93 | 0.84 | 0.28 | 0.93 | 0.43 |
| **4** | 39 | 105 | 5 | 22 | 0.89 | 0.83 | 0.27 | 0.89 | 0.41 |
| **5** | 39 | 104 | 5 | 23 | 0.89 | 0.82 | 0.27 | 0.89 | 0.42 |
| **6** | 39 | 100 | 5 | 27 | 0.89 | 0.79 | 0.28 | 0.89 | 0.43 |
| **7** | 37 | 104 | 7 | 23 | 0.84 | 0.82 | 0.26 | 0.84 | 0.40 |
| **8** | 34 | 99 | 10 | 28 | 0.77 | 0.78 | 0.26 | 0.77 | 0.38 |
| **9** | 33 | 92 | 11 | 35 | 0.75 | 0.72 | 0.26 | 0.75 | 0.39 |
| **10** | 35 | 93 | 9 | 34 | 0.80 | 0.73 | 0.27 | 0.80 | 0.41 |
| **11** | 37 | 100 | 7 | 27 | 0.84 | 0.79 | 0.27 | 0.84 | 0.41 |
| **12** | 36 | 100 | 8 | 27 | 0.82 | 0.79 | 0.26 | 0.82 | 0.40 |
| **13** | 35 | 94 | 9 | 33 | 0.80 | 0.74 | 0.27 | 0.80 | 0.40 |
| **14** | 21 | 25 | 23 | 102 | 0.48 | 0.20 | 0.46 | 0.48 | 0.47 |
| **15** | 18 | 16 | 26 | 111 | 0.41 | 0.13 | 0.53 | 0.41 | 0.46 |
| **16** | 34 | 89 | 10 | 38 | 0.77 | 0.70 | 0.28 | 0.77 | 0.41 |
| **17** | 5 | 4 | 39 | 123 | 0.11 | 0.03 | 0.56 | 0.11 | 0.19 |
| **18** | 4 | 3 | 40 | 124 | 0.09 | 0.02 | 0.57 | 0.09 | 0.16 |
| **19** | 4 | 3 | 40 | 124 | 0.09 | 0.02 | 0.57 | 0.09 | 0.16 |
| **20** | 15 | 19 | 29 | 108 | 0.34 | 0.15 | 0.44 | 0.34 | 0.38 |
| **21** | 33 | 81 | 11 | 46 | 0.75 | 0.64 | 0.29 | 0.75 | 0.42 |
| **22** | 31 | 82 | 13 | 45 | 0.70 | 0.65 | 0.27 | 0.70 | 0.39 |
| **23** | 36 | 92 | 8 | 35 | 0.82 | 0.72 | 0.28 | 0.82 | 0.42 |
| **24** | 36 | 91 | 8 | 36 | 0.82 | 0.72 | 0.28 | 0.82 | 0.42 |
| **25** | 32 | 95 | 12 | 32 | 0.73 | 0.75 | 0.25 | 0.73 | 0.37 |
| **26** | 30 | 97 | 14 | 30 | 0.68 | 0.76 | 0.24 | 0.68 | 0.35 |
| **27** | 26 | 95 | 18 | 32 | 0.59 | 0.75 | 0.21 | 0.59 | 0.32 |
| **28** | 20 | 14 | 24 | 113 | 0.45 | 0.11 | 0.59 | 0.45 | 0.51 |
| **29** | 18 | 12 | 26 | 115 | 0.41 | 0.09 | 0.60 | 0.41 | 0.49 |
| **30** | 13 | 4 | 31 | 123 | 0.30 | 0.03 | 0.76 | 0.30 | 0.43 |
| **31** | 14 | 7 | 30 | 120 | 0.32 | 0.06 | 0.67 | 0.32 | 0.43 |
| **32** | 12 | 5 | 32 | 122 | 0.27 | 0.04 | 0.71 | 0.27 | 0.39 |
| **33** | 13 | 4 | 31 | 123 | 0.30 | 0.03 | 0.76 | 0.30 | 0.43 |
| **34** | 9 | 4 | 35 | 123 | 0.20 | 0.03 | 0.69 | 0.20 | 0.32 |
| **35** | 29 | 31 | 15 | 96 | 0.66 | 0.24 | 0.48 | 0.66 | 0.56 |
| **36** | 28 | 41 | 16 | 86 | 0.64 | 0.32 | 0.41 | 0.64 | 0.50 |
| **37** | 21 | 9 | 23 | 118 | 0.48 | 0.07 | 0.70 | 0.48 | 0.57 |
| **38** | 14 | 7 | 30 | 120 | 0.32 | 0.06 | 0.67 | 0.32 | 0.43 |
| **39** | 14 | 9 | 30 | 118 | 0.32 | 0.07 | 0.61 | 0.32 | 0.42 |
| **40** | 9 | 10 | 35 | 117 | 0.20 | 0.08 | 0.47 | 0.20 | 0.29 |
| **41** | 15 | 16 | 29 | 111 | 0.34 | 0.13 | 0.48 | 0.34 | 0.40 |

The average weighted ensemble algorithm is an ensemble method (as the name suggests), the algorithm trains an expert on the most recently received chunk of data and computes its MSE on that same data – this MSE is used as baseline. Then every past trained expert is given the recent data received and they give back predictions. Final prediction is made by weighted majority vote, meaning that the prediction of each past expert is multiplied by its current weight and the sum of these multiplications determines the final prediction made. Each expert’s weight is then recomputed proportionate to their performance on the recent data and with respect to the MSE baseline. In our experiment we observe quiet disturbing results – in the stable period, AWE suffers from a massive number of false positives which then drops drastically right at the change point and then has big fluctuations in the following days. The AWE algorithm implemented in python uses a Naïve Bayes classifier as its expert making it all the more not logical that the algorithm classifies most of the plants in the stable period as being stressed – as they are the smaller class - which brought me to the conclusion that we are not using the algorithm correctly.

One thing I understood we’ve been doing wrong is using the default parameters of the algorithm. One of the parameters is the window size, this parameter decides after what number of observations the experts (Naïve Bayes) are being trained. The default parameter value is 200 and our jump is 171 – clearly this can be a source to many problems. I hoped fixing the parameter to the correct value will solve the FP issue, but that was not the case – it still classified most of the observations in the stable period as positives. I then tried to train a simple AWE classifier on the first jump data and made a prediction on the same data with the trained classifier, even in this case the algorithm classified all samples as positives. I’m having a very hard time explaining this phenomena. I think this is some statistical attribute of our data but I cannot understand what it is, because if all observations are quite identical (which they should be in the stable period), then I’d expect the Naïve Bayes to classify based on the a-priori probabilities – hence classifying all to the negative class since it’s the largest. In case there is a difference between the observations from treatment D to the rest in the stable period (shouldn’t happen), I’d expect to see it with the CDDRL, but this is not observed in the CDDRL. This issue is disturbing and so far, I was not able to explain it - it requires further investigation and perhaps an expert’s advice. In any case I’m attaching the results I got with the fixed parameter:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 127 | 44 | 0 | 0 | 0 | 0 | 0 |
| **2** | 39 | 109 | 18 | 5 | 0.89 | 0.86 | 0.26 | 0.89 | 0.41 |
| **3** | 41 | 111 | 16 | 3 | 0.93 | 0.87 | 0.27 | 0.93 | 0.42 |
| **4** | 40 | 109 | 18 | 4 | 0.91 | 0.86 | 0.27 | 0.91 | 0.41 |
| **5** | 40 | 104 | 23 | 4 | 0.91 | 0.82 | 0.28 | 0.91 | 0.43 |
| **6** | 38 | 101 | 26 | 6 | 0.86 | 0.80 | 0.27 | 0.86 | 0.42 |
| **7** | 37 | 101 | 26 | 7 | 0.84 | 0.80 | 0.27 | 0.84 | 0.41 |
| **8** | 35 | 106 | 21 | 9 | 0.80 | 0.83 | 0.25 | 0.80 | 0.38 |
| **9** | 35 | 111 | 16 | 9 | 0.80 | 0.87 | 0.24 | 0.80 | 0.37 |
| **10** | 37 | 109 | 18 | 7 | 0.84 | 0.86 | 0.25 | 0.84 | 0.39 |
| **11** | 37 | 110 | 17 | 7 | 0.84 | 0.87 | 0.25 | 0.84 | 0.39 |
| **12** | 36 | 108 | 19 | 8 | 0.82 | 0.85 | 0.25 | 0.82 | 0.38 |
| **13** | 39 | 107 | 20 | 5 | 0.89 | 0.84 | 0.27 | 0.89 | 0.41 |
| **14** | 2 | 1 | 126 | 42 | 0.05 | 0.01 | 0.67 | 0.05 | 0.09 |
| **15** | 3 | 1 | 126 | 41 | 0.07 | 0.01 | 0.75 | 0.07 | 0.13 |
| **16** | 3 | 0 | 127 | 41 | 0.07 | 0.00 | 1.00 | 0.07 | 0.13 |
| **17** | 4 | 1 | 126 | 40 | 0.09 | 0.01 | 0.80 | 0.09 | 0.16 |
| **18** | 3 | 1 | 126 | 41 | 0.07 | 0.01 | 0.75 | 0.07 | 0.13 |
| **19** | 3 | 1 | 126 | 41 | 0.07 | 0.01 | 0.75 | 0.07 | 0.13 |
| **20** | 40 | 106 | 21 | 4 | 0.91 | 0.83 | 0.27 | 0.91 | 0.42 |
| **21** | 40 | 107 | 20 | 4 | 0.91 | 0.84 | 0.27 | 0.91 | 0.42 |
| **22** | 39 | 104 | 23 | 5 | 0.89 | 0.82 | 0.27 | 0.89 | 0.42 |
| **23** | 37 | 102 | 25 | 7 | 0.84 | 0.80 | 0.27 | 0.84 | 0.40 |
| **24** | 21 | 14 | 113 | 23 | 0.48 | 0.11 | 0.60 | 0.48 | 0.53 |
| **25** | 21 | 16 | 111 | 23 | 0.48 | 0.13 | 0.57 | 0.48 | 0.52 |
| **26** | 25 | 44 | 83 | 19 | 0.57 | 0.35 | 0.36 | 0.57 | 0.44 |
| **27** | 23 | 12 | 115 | 21 | 0.52 | 0.09 | 0.66 | 0.52 | 0.58 |
| **28** | 16 | 3 | 124 | 28 | 0.36 | 0.02 | 0.84 | 0.36 | 0.51 |
| **29** | 13 | 4 | 123 | 31 | 0.30 | 0.03 | 0.76 | 0.30 | 0.43 |
| **30** | 10 | 3 | 124 | 34 | 0.23 | 0.02 | 0.77 | 0.23 | 0.35 |
| **31** | 13 | 7 | 120 | 31 | 0.30 | 0.06 | 0.65 | 0.30 | 0.41 |
| **32** | 9 | 8 | 119 | 35 | 0.20 | 0.06 | 0.53 | 0.20 | 0.30 |
| **33** | 17 | 11 | 116 | 27 | 0.39 | 0.09 | 0.61 | 0.39 | 0.47 |
| **34** | 15 | 8 | 119 | 29 | 0.34 | 0.06 | 0.65 | 0.34 | 0.45 |
| **35** | 25 | 21 | 106 | 19 | 0.57 | 0.17 | 0.54 | 0.57 | 0.56 |
| **36** | 26 | 25 | 102 | 18 | 0.59 | 0.20 | 0.51 | 0.59 | 0.55 |
| **37** | 20 | 9 | 118 | 24 | 0.45 | 0.07 | 0.69 | 0.45 | 0.55 |
| **38** | 15 | 9 | 118 | 29 | 0.34 | 0.07 | 0.63 | 0.34 | 0.44 |
| **39** | 12 | 12 | 115 | 32 | 0.27 | 0.09 | 0.50 | 0.27 | 0.35 |
| **40** | 31 | 74 | 53 | 13 | 0.70 | 0.58 | 0.30 | 0.70 | 0.42 |
| **41** | 33 | 74 | 53 | 11 | 0.75 | 0.58 | 0.31 | 0.75 | 0.44 |

#### 7.2.2.4 DWM Analysis

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 44 | 126 | 0 | 1 | 1.00 | 0.99 | 0.26 | 1.00 | 0.41 |
| **3** | 43 | 113 | 1 | 14 | 0.98 | 0.89 | 0.28 | 0.98 | 0.43 |
| **4** | 36 | 100 | 8 | 27 | 0.82 | 0.79 | 0.26 | 0.82 | 0.40 |
| **5** | 37 | 95 | 7 | 32 | 0.84 | 0.75 | 0.28 | 0.84 | 0.42 |
| **6** | 20 | 49 | 24 | 78 | 0.45 | 0.39 | 0.29 | 0.45 | 0.35 |
| **7** | 18 | 28 | 26 | 99 | 0.41 | 0.22 | 0.39 | 0.41 | 0.40 |
| **8** | 8 | 13 | 36 | 114 | 0.18 | 0.10 | 0.38 | 0.18 | 0.25 |
| **9** | 7 | 7 | 37 | 120 | 0.16 | 0.06 | 0.50 | 0.16 | 0.24 |
| **10** | 7 | 10 | 37 | 117 | 0.16 | 0.08 | 0.41 | 0.16 | 0.23 |
| **11** | 8 | 13 | 36 | 114 | 0.18 | 0.10 | 0.38 | 0.18 | 0.25 |
| **12** | 8 | 10 | 36 | 117 | 0.18 | 0.08 | 0.44 | 0.18 | 0.26 |
| **13** | 4 | 10 | 40 | 117 | 0.09 | 0.08 | 0.29 | 0.09 | 0.14 |
| **14** | 4 | 6 | 40 | 121 | 0.09 | 0.05 | 0.40 | 0.09 | 0.15 |
| **15** | 3 | 0 | 41 | 127 | 0.07 | 0.00 | 1.00 | 0.07 | 0.13 |
| **16** | 3 | 0 | 41 | 127 | 0.07 | 0.00 | 1.00 | 0.07 | 0.13 |
| **17** | 7 | 6 | 37 | 121 | 0.16 | 0.05 | 0.54 | 0.16 | 0.25 |
| **18** | 3 | 1 | 41 | 126 | 0.07 | 0.01 | 0.75 | 0.07 | 0.13 |
| **19** | 4 | 1 | 40 | 126 | 0.09 | 0.01 | 0.80 | 0.09 | 0.16 |
| **20** | 3 | 6 | 41 | 121 | 0.07 | 0.05 | 0.33 | 0.07 | 0.11 |
| **21** | 4 | 5 | 40 | 122 | 0.09 | 0.04 | 0.44 | 0.09 | 0.15 |
| **22** | 4 | 5 | 40 | 122 | 0.09 | 0.04 | 0.44 | 0.09 | 0.15 |
| **23** | 9 | 5 | 35 | 122 | 0.20 | 0.04 | 0.64 | 0.20 | 0.31 |
| **24** | 25 | 32 | 19 | 95 | 0.57 | 0.25 | 0.44 | 0.57 | 0.50 |
| **25** | 32 | 46 | 12 | 81 | 0.73 | 0.36 | 0.41 | 0.73 | 0.52 |
| **26** | 36 | 58 | 8 | 69 | 0.82 | 0.46 | 0.38 | 0.82 | 0.52 |
| **27** | 30 | 24 | 14 | 103 | 0.68 | 0.19 | 0.56 | 0.68 | 0.61 |
| **28** | 20 | 9 | 24 | 118 | 0.45 | 0.07 | 0.69 | 0.45 | 0.55 |
| **29** | 16 | 5 | 28 | 122 | 0.36 | 0.04 | 0.76 | 0.36 | 0.49 |
| **30** | 14 | 7 | 30 | 120 | 0.32 | 0.06 | 0.67 | 0.32 | 0.43 |
| **31** | 15 | 8 | 29 | 119 | 0.34 | 0.06 | 0.65 | 0.34 | 0.45 |
| **32** | 12 | 8 | 32 | 119 | 0.27 | 0.06 | 0.60 | 0.27 | 0.38 |
| **33** | 16 | 12 | 28 | 115 | 0.36 | 0.09 | 0.57 | 0.36 | 0.44 |
| **34** | 13 | 7 | 31 | 120 | 0.30 | 0.06 | 0.65 | 0.30 | 0.41 |
| **35** | 16 | 12 | 28 | 115 | 0.36 | 0.09 | 0.57 | 0.36 | 0.44 |
| **36** | 16 | 9 | 28 | 118 | 0.36 | 0.07 | 0.64 | 0.36 | 0.46 |
| **37** | 19 | 8 | 25 | 119 | 0.43 | 0.06 | 0.70 | 0.43 | 0.54 |
| **38** | 14 | 8 | 30 | 119 | 0.32 | 0.06 | 0.64 | 0.32 | 0.42 |
| **39** | 14 | 9 | 30 | 118 | 0.32 | 0.07 | 0.61 | 0.32 | 0.42 |
| **40** | 13 | 13 | 31 | 114 | 0.30 | 0.10 | 0.50 | 0.30 | 0.37 |
| **41** | 13 | 14 | 31 | 113 | 0.30 | 0.11 | 0.48 | 0.30 | 0.37 |

The dynamically weighted majority (DWM) is an online algorithm, meaning it trains its classifiers sample by sample. This is also an ensemble method. The algorithm has a hyperparameter called ‘period’ that determines after how many observations a classifier can be added or removed – we set this parameter to be equal to jump. A classifier is added if the general ensemble made any prediction error (even one) during the ‘period’. A classifier making an error gets its weight reduced and is removed from the ensemble if its weight drops under some threshold (also a hyperparameter). The base classifier used in python is also the Naïve Bayes. Although the algorithm removes/adds classifiers and cuts the weight of classifiers that made a mistake, it still trains the classifiers after every sample, thus giving meaning to the order in which the observations are streamed into the algorithm even inside each batch. For that reason, I believe the DWM algorithm might not suit tasks such as ours, where a batch of observations (jump\day in our case) has a meaning as a batch and cannot be streamed correctly one sample at a time. I tried changing the position of the observations inside the first 3 batches (switching the position of observations from treatment A with those from treatment D), and I got slightly different results. I believe the false alarms that we observe in the stable period and the patterns observed in recall and FP (drops, goes up and drops again), are due to the online manner in which the DWM works by i.e., training the classifiers after every sample.

#### 7.2.2.6 SRPC Analysis

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 44 | 127 | 0 | 0 | 1.00 | 1.00 | 0.26 | 1.00 | 0.41 |
| **3** | 37 | 83 | 7 | 44 | 0.84 | 0.65 | 0.31 | 0.84 | 0.45 |
| **4** | 34 | 67 | 10 | 60 | 0.77 | 0.53 | 0.34 | 0.77 | 0.47 |
| **5** | 28 | 71 | 16 | 56 | 0.64 | 0.56 | 0.28 | 0.64 | 0.39 |
| **6** | 37 | 90 | 7 | 37 | 0.84 | 0.71 | 0.29 | 0.84 | 0.43 |
| **7** | 18 | 31 | 26 | 96 | 0.41 | 0.24 | 0.37 | 0.41 | 0.39 |
| **8** | 39 | 105 | 5 | 22 | 0.89 | 0.83 | 0.27 | 0.89 | 0.41 |
| **9** | 22 | 31 | 22 | 96 | 0.50 | 0.24 | 0.42 | 0.50 | 0.45 |
| **10** | 16 | 5 | 28 | 122 | 0.36 | 0.04 | 0.76 | 0.36 | 0.49 |
| **11** | 14 | 9 | 30 | 118 | 0.32 | 0.07 | 0.61 | 0.32 | 0.42 |
| **12** | 27 | 27 | 17 | 100 | 0.61 | 0.21 | 0.50 | 0.61 | 0.55 |
| **13** | 39 | 58 | 5 | 69 | 0.89 | 0.46 | 0.40 | 0.89 | 0.55 |
| **14** | 41 | 72 | 3 | 55 | 0.93 | 0.57 | 0.36 | 0.93 | 0.52 |
| **15** | 26 | 28 | 18 | 99 | 0.59 | 0.22 | 0.48 | 0.59 | 0.53 |
| **16** | 34 | 46 | 10 | 81 | 0.77 | 0.36 | 0.43 | 0.77 | 0.55 |
| **17** | 26 | 18 | 18 | 109 | 0.59 | 0.14 | 0.59 | 0.59 | 0.59 |
| **18** | 18 | 9 | 26 | 118 | 0.41 | 0.07 | 0.67 | 0.41 | 0.51 |
| **19** | 11 | 4 | 33 | 123 | 0.25 | 0.03 | 0.73 | 0.25 | 0.37 |
| **20** | 30 | 32 | 14 | 95 | 0.68 | 0.25 | 0.48 | 0.68 | 0.57 |
| **21** | 31 | 48 | 13 | 79 | 0.70 | 0.38 | 0.39 | 0.70 | 0.50 |
| **22** | 30 | 24 | 14 | 103 | 0.68 | 0.19 | 0.56 | 0.68 | 0.61 |
| **23** | 20 | 12 | 24 | 115 | 0.45 | 0.09 | 0.63 | 0.45 | 0.53 |
| **24** | 23 | 11 | 21 | 116 | 0.52 | 0.09 | 0.68 | 0.52 | 0.59 |
| **25** | 24 | 9 | 20 | 118 | 0.55 | 0.07 | 0.73 | 0.55 | 0.62 |
| **26** | 27 | 23 | 17 | 104 | 0.61 | 0.18 | 0.54 | 0.61 | 0.57 |
| **27** | 29 | 24 | 15 | 103 | 0.66 | 0.19 | 0.55 | 0.66 | 0.60 |
| **28** | 29 | 9 | 15 | 118 | 0.66 | 0.07 | 0.76 | 0.66 | 0.71 |
| **29** | 20 | 4 | 24 | 123 | 0.45 | 0.03 | 0.83 | 0.45 | 0.59 |
| **30** | 17 | 2 | 27 | 125 | 0.39 | 0.02 | 0.89 | 0.39 | 0.54 |
| **31** | 17 | 4 | 27 | 123 | 0.39 | 0.03 | 0.81 | 0.39 | 0.52 |
| **32** | 21 | 4 | 23 | 123 | 0.48 | 0.03 | 0.84 | 0.48 | 0.61 |
| **33** | 22 | 1 | 22 | 126 | 0.50 | 0.01 | 0.96 | 0.50 | 0.66 |
| **34** | 12 | 2 | 32 | 125 | 0.27 | 0.02 | 0.86 | 0.27 | 0.41 |
| **35** | 15 | 1 | 29 | 126 | 0.34 | 0.01 | 0.94 | 0.34 | 0.50 |
| **36** | 19 | 7 | 25 | 120 | 0.43 | 0.06 | 0.73 | 0.43 | 0.54 |
| **37** | 21 | 5 | 23 | 122 | 0.48 | 0.04 | 0.81 | 0.48 | 0.60 |
| **38** | 10 | 4 | 34 | 123 | 0.23 | 0.03 | 0.71 | 0.23 | 0.34 |
| **39** | 13 | 3 | 31 | 124 | 0.30 | 0.02 | 0.81 | 0.30 | 0.43 |
| **40** | 13 | 3 | 31 | 124 | 0.30 | 0.02 | 0.81 | 0.30 | 0.43 |
| **41** | 10 | 6 | 34 | 121 | 0.23 | 0.05 | 0.63 | 0.23 | 0.33 |

The streaming random patches algorithm (SRPC) is also an ensemble method that uses a subspace of features and samples to train its learners (Hoeffding trees). SRPC also uses ADWIN for drift detection, enabling it to know when and using which samples to train a new learner. We can observe many false alarms in the stable period that decreases as the experiment progresses. The SRPC has a steady increase in precision and a steady decrease in recall through the days of the experiment. The use of Hoeffding trees as learners might explain the false alarms received in the stable period – eventually each sample arrives at some leaf with some other samples making it similar to a KNN operation. The use of KNN and its tendency to have false alarms when samples are all similar to each other was explained in the KNN-ADWIN analysis above. Another possible explanation for the false alarms is the fact that we used a relatively small number of estimators, combining the fact that the SRPC uses only a randomized portion of the samples given for each learner may lead to false alarms in the initial period. After experiencing with the number of estimators and with the algorithm’s hyperparameter that decides what portion of the samples is used to train each learner, I observed a decrease in the number of false alarms during the stable period when using a high number of estimators and a high number of samples proportion (100 and 80% respectively). However, the decrease was not very significant but the increase in runtime was very significant. I assign the rest of the false alarms observed to the use of Hoeffding trees as explained before.

#### 7.2.2.5 LNSE Analysis

The learning in nonstationary environment (LNSE) is also an ensemble method that is unified by the special way it punishes classifiers – using a sigmoid function. LNSE learns a classifier at every batch regardless if a change was detected or not. The LNSE does not remove classifiers from the ensemble, but zeros their voting weight in case their performance is poor. This mechanism allows the LNSE to use old classifiers in the future in cases of recurrent concept drifts. The LNSE punishes classifiers more harshly if they are mistaken on the newest data and benefits classifiers that perform well in the current timestamp. The classifier used with LNSE is a decision tree classifier - as discussed in KNN-ADWIN and SRPC, can explain the reason we observe false alarms in the stable period. Here, we had the same problem as with the AWE algorithm – we used the default parameter that decides the batch size i.e., after how many observations a new classifier will be added. The default value is 250 and our jump is 171, I fixed this issue and the new results are reported below:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Day** | **TP** | **FP** | **FN** | **TN** | **DR** | **FA** | **Precision** | **Recall** | **F1** |
| **1** | 0 | 0 | 44 | 127 | 0 | 0 | 0 | 0 | 0 |
| **2** | 21 | 40 | 23 | 87 | 0.48 | 0.31 | 0.34 | 0.48 | 0.4 |
| **3** | 11 | 18 | 33 | 109 | 0.25 | 0.14 | 0.38 | 0.25 | 0.30 |
| **4** | 24 | 28 | 20 | 99 | 0.55 | 0.22 | 0.46 | 0.55 | 0.50 |
| **5** | 18 | 45 | 26 | 82 | 0.41 | 0.35 | 0.29 | 0.41 | 0.34 |
| **6** | 25 | 23 | 19 | 104 | 0.57 | 0.18 | 0.52 | 0.57 | 0.54 |
| **7** | 20 | 20 | 24 | 107 | 0.45 | 0.16 | 0.50 | 0.45 | 0.48 |
| **8** | 15 | 32 | 29 | 95 | 0.34 | 0.25 | 0.32 | 0.34 | 0.33 |
| **9** | 20 | 23 | 24 | 104 | 0.45 | 0.18 | 0.47 | 0.45 | 0.46 |
| **10** | 25 | 24 | 19 | 103 | 0.57 | 0.19 | 0.51 | 0.57 | 0.54 |
| **11** | 23 | 15 | 21 | 112 | 0.52 | 0.12 | 0.61 | 0.52 | 0.56 |
| **12** | 24 | 12 | 20 | 115 | 0.55 | 0.09 | 0.67 | 0.55 | 0.60 |
| **13** | 23 | 35 | 21 | 92 | 0.52 | 0.28 | 0.40 | 0.52 | 0.45 |
| **14** | 22 | 25 | 22 | 102 | 0.50 | 0.20 | 0.47 | 0.50 | 0.48 |
| **15** | 19 | 34 | 25 | 93 | 0.43 | 0.27 | 0.36 | 0.43 | 0.39 |
| **16** | 25 | 24 | 19 | 103 | 0.57 | 0.19 | 0.51 | 0.57 | 0.54 |
| **17** | 26 | 30 | 18 | 97 | 0.59 | 0.24 | 0.46 | 0.59 | 0.52 |
| **18** | 22 | 11 | 22 | 116 | 0.50 | 0.09 | 0.67 | 0.50 | 0.57 |
| **19** | 23 | 16 | 21 | 111 | 0.52 | 0.13 | 0.59 | 0.52 | 0.55 |
| **20** | 30 | 41 | 14 | 86 | 0.68 | 0.32 | 0.42 | 0.68 | 0.52 |
| **21** | 20 | 23 | 24 | 104 | 0.45 | 0.18 | 0.47 | 0.45 | 0.46 |
| **22** | 23 | 25 | 21 | 102 | 0.52 | 0.20 | 0.48 | 0.52 | 0.50 |
| **23** | 20 | 21 | 24 | 106 | 0.45 | 0.17 | 0.49 | 0.45 | 0.47 |
| **24** | 23 | 19 | 21 | 108 | 0.52 | 0.15 | 0.55 | 0.52 | 0.53 |
| **25** | 33 | 21 | 11 | 106 | 0.75 | 0.17 | 0.61 | 0.75 | 0.67 |
| **26** | 33 | 42 | 11 | 85 | 0.75 | 0.33 | 0.44 | 0.75 | 0.55 |
| **27** | 32 | 17 | 12 | 110 | 0.73 | 0.13 | 0.65 | 0.73 | 0.69 |
| **28** | 21 | 9 | 23 | 118 | 0.48 | 0.07 | 0.70 | 0.48 | 0.57 |
| **29** | 27 | 22 | 17 | 105 | 0.61 | 0.17 | 0.55 | 0.61 | 0.58 |
| **30** | 23 | 12 | 21 | 115 | 0.52 | 0.09 | 0.66 | 0.52 | 0.58 |
| **31** | 23 | 18 | 21 | 109 | 0.52 | 0.14 | 0.56 | 0.52 | 0.54 |
| **32** | 20 | 12 | 24 | 115 | 0.45 | 0.09 | 0.63 | 0.45 | 0.53 |
| **33** | 31 | 20 | 13 | 107 | 0.70 | 0.16 | 0.61 | 0.70 | 0.65 |
| **34** | 25 | 24 | 19 | 103 | 0.57 | 0.19 | 0.51 | 0.57 | 0.54 |
| **35** | 23 | 18 | 21 | 109 | 0.52 | 0.14 | 0.56 | 0.52 | 0.54 |
| **36** | 31 | 25 | 13 | 102 | 0.70 | 0.20 | 0.55 | 0.70 | 0.62 |
| **37** | 23 | 16 | 21 | 111 | 0.52 | 0.13 | 0.59 | 0.52 | 0.55 |
| **38** | 23 | 21 | 21 | 106 | 0.52 | 0.17 | 0.52 | 0.52 | 0.52 |
| **39** | 28 | 19 | 16 | 108 | 0.64 | 0.15 | 0.60 | 0.64 | 0.62 |
| **40** | 21 | 16 | 23 | 111 | 0.48 | 0.13 | 0.57 | 0.48 | 0.52 |
| **41** | 28 | 25 | 16 | 102 | 0.64 | 0.20 | 0.53 | 0.64 | 0.58 |

The LNSE shows similar results for recall and precision throughout the experiment reaching best performance of 0.75 and 0.7 respectively. We can observe that the algorithm is very unstable with respect to FP with many peaks stabilizes on around 17 FP starting from day 27. With respect to FN, it shows a mild and steady decrease and stabilizes like FP starting from day 27 on around 17 plants. F1 results has slightly improved but are still not very encouraging.

### 7.2.3 By Plant Analysis

#### 7.2.3.1 Plants Detection Mistakes for All Algorithms

The following table shows the number of detection mistakes all algorithms had during days 20-41 (the CDDRL’s stress detection days) for the CDDRL and during days 14-41 (the stress period) for all other algorithms. Plants with at least 15 mistakes are colored in red and those with less than three mistakes are colored in green. In addition, the average number of mistakes for all algorithms together is also given:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **CDDRL (22 days)** | | **ADWIN (28 days)** | | **AWE (28 days)** | | **DWM (28 days)** | | **LNSE (28 days)** | | **SRPC (28 days)** | | **Average All** | |
| **No.** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **Avg # of mistakes** |
| **1** | **B14** | **19** | **D8** | **23** | **D23** | **26** | **D43** | **28** | **D29** | **24** | **D32** | **25** | **D32** | **20.5** |
| **2** | **C28** | **18** | **D7** | **21** | **D24** | **21** | **D48** | **28** | **D7** | **23** | **D43** | **22** | **D7** | **20.3** |
| **3** | **D39** | **18** | **D14** | **20** | **D3** | **21** | **D24** | **27** | **D16** | **22** | **D47** | **22** | **D24** | **19.5** |
| **4** | **D40** | **18** | **D32** | **19** | **D40** | **21** | **D30** | **27** | **D21** | **21** | **D7** | **20** | **D8** | **19.5** |
| **5** | **C19** | **17** | **D13** | **18** | **D45** | **21** | **D31** | **27** | **D32** | **21** | **D24** | **19** | **D45** | **19.2** |
| **6** | **C14** | **16** | **D43** | **18** | **D31** | **20** | **D32** | **27** | **D48** | **21** | **D29** | **19** | **D16** | **19.0** |
| **7** | **C15** | **16** | **D45** | **18** | **D32** | **20** | **D40** | **27** | **D24** | **20** | **D45** | **19** | **D48** | **18.8** |
| **8** | **D38** | **16** | **D48** | **18** | **D39** | **20** | **D47** | **26** | **D31** | **20** | **D48** | **19** | **D23** | **18.6** |
| **9** | **D16** | **15** | **D5** | **18** | **D47** | **20** | **D7** | **26** | **D8** | **20** | **D8** | **19** | **D43** | **18.4** |
| **10** | **D23** | **15** | **D16** | **17** | **D6** | **20** | **D23** | **25** | **D22** | **19** | **D14** | **18** | **D47** | **18.4** |
| **11** | **D37** | **15** | **D24** | **17** | **D2** | **19** | **D8** | **25** | **D45** | **19** | **D16** | **18** | **D31** | **17.8** |
| **12** | **D46** | **15** | **D10** | **16** | **D21** | **19** | **D10** | **24** | **D14** | **18** | **D30** | **18** | **D21** | **17.7** |
| **13** | **D6** | **15** | **D33** | **16** | **D43** | **19** | **D16** | **24** | **D33** | **18** | **D15** | **17** | **D30** | **17.0** |
| **14** | **A17** | **14** | **D39** | **16** | **D7** | **19** | **D21** | **24** | **D47** | **18** | **D21** | **17** | **D40** | **16.7** |
| **15** | **A18** | **14** | **D15** | **15** | **D16** | **18** | **D33** | **24** | **D6** | **18** | **D23** | **17** | **D39** | **16.2** |
| **16** | **A27** | **14** | **D21** | **15** | **D33** | **18** | **D45** | **24** | **D12** | **17** | **D36** | **17** | **D6** | **16.0** |
| **17** | **B2** | **14** | **D29** | **15** | **C19** | **17** | **D22** | **22** | **D34** | **17** | **D39** | **16** | **D33** | **15.7** |
| **18** | **D45** | **14** | D12 | 14 | **D10** | **17** | **D3** | **22** | **D43** | **17** | **D40** | **16** | **D46** | **15.7** |
| **19** | **D8** | **14** | D36 | 14 | **D18** | **17** | **D46** | **22** | **D18** | **16** | **D31** | **15** | **D14** | **15.6** |
| **20** | **A10** | **13** | D47 | 14 | **D30** | **17** | **D36** | **21** | **D19** | **16** | **D33** | **15** | **D29** | **15.4** |
| **21** | **A21** | **13** | D23 | 13 | **D36** | **17** | **D5** | **21** | **D30** | **16** | **D37** | **15** | **D5** | **15.2** |
| **22** | **B15** | **13** | D28 | 13 | **D46** | **17** | **D6** | **21** | **D5** | **16** | **D38** | **15** | D10 | 14.7 |
| **23** | **B41** | **13** | D3 | 13 | **D48** | **17** | **D11** | **20** | **B28** | **15** | **D46** | **15** | D38 | 14.5 |
| **24** | **C29** | **13** | D38 | 13 | **A21** | **16** | **D2** | **20** | **D10** | **15** | D34 | 14 | D36 | 14.2 |
| **25** | **C41** | **13** | D20 | 12 | **D25** | **16** | **D15** | **19** | **D11** | **15** | D12 | 13 | D15 | 13.9 |
| **26** | **D24** | **13** | D37 | 12 | **D38** | **16** | **D25** | **19** | **D15** | **15** | D28 | 13 | D22 | 13.7 |
| **27** | **D30** | **13** | D17 | 11 | **D5** | **16** | **D29** | **19** | **D2** | **15** | D3 | 13 | D3 | 13.5 |
| **28** | **D31** | **13** | D19 | 11 | **D8** | **16** | **D14** | **18** | **D23** | **15** | D5 | 13 | D2 | 13.5 |
| **29** | **D4** | **13** | D2 | 11 | **A34** | **15** | **D39** | **18** | **D38** | **15** | D18 | 12 | D37 | 12.9 |
| **30** | **D7** | **13** | D26 | 11 | **B28** | **15** | **D19** | **17** | **D46** | **15** | D19 | 12 | D19 | 12.3 |
| **31** | B28 | 12 | D30 | 11 | **C28** | **15** | **D17** | **15** | D27 | 14 | D22 | 12 | D34 | 11.9 |
| **32** | C13 | 12 | D31 | 11 | B41 | 14 | **D37** | **15** | D36 | 14 | D6 | 12 | D18 | 11.8 |
| **33** | C5 | 12 | D46 | 10 | D14 | 14 | C19 | 13 | A27 | 13 | D25 | 11 | C19 | 11.5 |
| **34** | D32 | 12 | A3 | 9 | D22 | 14 | D12 | 13 | D1 | 13 | B1 | 10 | C28 | 11.4 |
| **35** | C21 | 11 | C13 | 9 | D29 | 14 | D13 | 13 | D28 | 13 | D2 | 10 | D11 | 11.3 |
| **36** | D47 | 11 | D11 | 9 | A1 | 13 | D34 | 13 | C21 | 12 | B2 | 9 | D25 | 11.2 |
| **37** | C27 | 10 | D18 | 9 | C17 | 13 | D38 | 12 | D13 | 12 | C17 | 9 | D12 | 10.8 |
| **38** | D21 | 10 | D34 | 9 | C29 | 13 | B41 | 11 | D26 | 12 | C28 | 9 | D28 | 10.6 |
| **39** | D48 | 10 | D6 | 9 | D11 | 13 | D18 | 11 | D3 | 12 | D13 | 9 | D13 | 10.6 |
| **40** | A3 | 9 | D9 | 9 | D19 | 13 | D26 | 11 | D37 | 12 | D20 | 9 | B14 | 10.2 |
| **41** | A6 | 9 | D22 | 8 | A33 | 12 | D28 | 11 | C14 | 11 | A21 | 8 | A21 | 9.9 |
| **42** | B1 | 9 | D27 | 8 | A6 | 12 | D9 | 11 | C19 | 11 | A6 | 8 | B28 | 9.8 |
| **43** | B21 | 9 | D4 | 8 | B1 | 12 | B1 | 10 | D17 | 11 | B14 | 8 | D4 | 9.6 |
| **44** | B27 | 9 | D40 | 8 | B14 | 12 | B14 | 10 | D25 | 11 | B6 | 8 | C14 | 9.5 |
| **45** | C2 | 9 | A27 | 7 | B17 | 12 | C2 | 10 | D9 | 11 | C4 | 8 | D17 | 9.5 |
| **46** | A1 | 8 | B17 | 7 | C14 | 12 | C28 | 10 | A3 | 10 | D10 | 8 | B41 | 8.9 |
| **47** | A12 | 8 | C28 | 7 | C34 | 12 | D4 | 10 | D20 | 10 | D11 | 8 | D9 | 8.9 |
| **48** | A22 | 8 | D25 | 7 | C4 | 12 | A17 | 9 | D40 | 10 | D26 | 8 | A27 | 8.7 |
| **49** | A25 | 8 | A6 | 6 | D17 | 12 | A21 | 9 | A18 | 9 | B17 | 7 | D20 | 8.7 |
| **50** | A33 | 8 | A15 | 5 | D28 | 12 | B6 | 9 | B11 | 9 | B7 | 7 | B1 | 8.6 |
| **51** | A34 | 8 | A21 | 5 | A10 | 11 | D1 | 9 | C28 | 9 | C10 | 7 | A17 | 8.5 |
| **52** | B12 | 8 | A30 | 5 | A17 | 11 | D20 | 9 | C7 | 9 | C27 | 7 | D26 | 8.2 |
| **53** | B19 | 8 | B1 | 5 | A28 | 11 | D27 | 9 | D39 | 9 | D1 | 7 | A3 | 8.0 |
| **54** | C1 | 8 | B48 | 5 | A3 | 11 | C1 | 8 | D4 | 9 | D4 | 7 | A6 | 7.7 |
| **55** | C18 | 8 | B7 | 5 | A9 | 11 | C14 | 8 | A10 | 8 | D9 | 7 | B2 | 7.6 |
| **56** | C31 | 8 | C17 | 5 | B12 | 11 | A34 | 7 | A17 | 8 | A22 | 6 | D27 | 7.6 |
| **57** | C32 | 8 | A20 | 4 | B15 | 11 | B13 | 7 | A21 | 8 | B11 | 6 | C21 | 7.5 |
| **58** | D34 | 8 | B13 | 4 | C10 | 11 | B28 | 7 | A39 | 8 | B22 | 6 | A18 | 7.4 |
| **59** | A13 | 7 | B14 | 4 | D15 | 11 | C34 | 7 | A6 | 8 | C1 | 6 | B17 | 7.3 |
| **60** | B18 | 7 | B21 | 4 | D20 | 11 | A3 | 6 | B14 | 8 | C14 | 6 | C29 | 7.1 |
| **61** | B4 | 7 | B28 | 4 | D34 | 11 | B17 | 6 | B18 | 8 | C19 | 6 | C34 | 7.1 |
| **62** | B6 | 7 | B3 | 4 | D9 | 11 | B37 | 6 | B19 | 8 | C3 | 6 | C15 | 7.0 |
| **63** | C17 | 7 | B41 | 4 | A13 | 10 | C21 | 6 | B38 | 8 | C6 | 6 | C2 | 7.0 |
| **64** | C23 | 7 | C1 | 4 | A18 | 10 | C29 | 6 | B41 | 8 | D17 | 6 | A10 | 6.9 |
| **65** | C3 | 7 | C19 | 4 | A25 | 10 | C47 | 6 | C12 | 8 | D27 | 6 | B15 | 6.7 |
| **66** | C30 | 7 | C36 | 4 | A27 | 10 | A1 | 5 | C2 | 8 | A1 | 5 | C1 | 6.7 |
| **67** | C34 | 7 | D1 | 4 | A39 | 10 | A2 | 5 | C34 | 8 | A10 | 5 | A1 | 6.6 |
| **68** | D10 | 7 | A17 | 3 | A7 | 10 | A33 | 5 | A16 | 7 | A17 | 5 | A34 | 6.5 |
| **69** | D15 | 7 | A18 | 3 | B13 | 10 | B18 | 5 | B48 | 7 | A18 | 5 | C17 | 6.4 |
| **70** | D22 | 7 | A22 | 3 | B2 | 10 | B2 | 5 | C11 | 7 | A20 | 5 | D1 | 6.3 |
| **71** | D43 | 7 | A33 | 3 | B26 | 10 | A10 | 4 | C15 | 7 | A30 | 5 | B6 | 6.3 |
| **72** | D5 | 7 | A45 | 3 | C1 | 10 | B15 | 4 | C30 | 7 | B10 | 5 | C13 | 6.2 |
| **73** | A2 | 6 | A9 | 3 | C15 | 10 | B40 | 4 | C41 | 7 | B12 | 5 | C27 | 6.0 |
| **74** | A20 | 6 | B11 | 3 | C2 | 10 | B7 | 4 | C46 | 7 | B13 | 5 | A33 | 5.8 |
| **75** | A28 | 6 | B19 | 3 | C21 | 10 | C15 | 4 | C48 | 7 | B15 | 5 | A22 | 5.8 |
| **76** | A35 | 6 | B22 | 3 | D13 | 10 | C45 | 4 | A1 | 6 | B27 | 5 | B18 | 5.7 |
| **77** | A45 | 6 | B30 | 3 | D4 | 10 | C5 | 4 | A22 | 6 | B28 | 5 | B13 | 5.7 |
| **78** | B11 | 6 | B36 | 3 | A11 | 9 | A18 | 3 | A34 | 6 | B5 | 5 | B7 | 5.6 |
| **79** | B13 | 6 | B44 | 3 | A16 | 9 | A22 | 3 | A7 | 6 | C23 | 5 | C4 | 5.6 |
| **80** | B16 | 6 | B5 | 3 | A2 | 9 | A27 | 3 | B1 | 6 | C34 | 5 | B19 | 5.6 |
| **81** | B17 | 6 | B6 | 3 | A20 | 9 | A29 | 3 | B10 | 6 | A11 | 4 | B12 | 5.5 |
| **82** | B36 | 6 | B8 | 3 | A22 | 9 | A39 | 3 | B17 | 6 | A12 | 4 | B11 | 5.5 |
| **83** | B7 | 6 | C11 | 3 | B11 | 9 | A6 | 3 | B2 | 6 | A13 | 4 | C11 | 5.2 |
| **84** | B8 | 6 | C12 | 3 | B18 | 9 | B19 | 3 | B22 | 6 | A15 | 4 | C5 | 5.2 |
| **85** | C10 | 6 | C14 | 3 | B21 | 9 | B21 | 3 | B4 | 6 | A27 | 4 | B27 | 5.2 |
| **86** | C11 | 6 | C22 | 3 | B22 | 9 | B27 | 3 | B6 | 6 | A28 | 4 | C3 | 5.1 |
| **87** | C20 | 6 | C27 | 3 | B27 | 9 | B3 | 3 | C13 | 6 | A43 | 4 | B21 | 5.0 |
| **88** | C4 | 6 | C34 | 3 | B30 | 9 | B30 | 3 | C18 | 6 | A7 | 4 | C48 | 4.9 |
| **89** | C42 | 6 | C38 | 3 | B36 | 9 | C18 | 3 | C29 | 6 | A9 | 4 | B30 | 4.9 |
| **90** | C45 | 6 | C6 | 3 | B47 | 9 | C27 | 3 | C32 | 6 | B18 | 4 | C18 | 4.9 |
| **91** | C48 | 6 | A1 | 2 | B48 | 9 | C3 | 3 | A13 | 5 | B19 | 4 | C10 | 4.8 |
| **92** | D14 | 6 | **A12** | **2** | C11 | 9 | C37 | 3 | A30 | 5 | B20 | 4 | A2 | 4.7 |
| **93** | D18 | 6 | **A2** | **2** | C47 | 9 | C38 | 3 | A33 | 5 | B21 | 4 | C30 | 4.7 |
| **94** | A16 | 5 | **A25** | **2** | D37 | 9 | C39 | 3 | A35 | 5 | B30 | 4 | A39 | 4.6 |
| **95** | A32 | 5 | **A28** | **2** | A19 | 8 | C48 | 3 | B12 | 5 | B48 | 4 | B22 | 4.6 |
| **96** | B26 | 5 | **A35** | **2** | A29 | 8 | C8 | 3 | B15 | 5 | B8 | 4 | B36 | 4.6 |
| **97** | B29 | 5 | **A41** | **2** | A30 | 8 | **A11** | **2** | B24 | 5 | C11 | 4 | A13 | 4.5 |
| **98** | B3 | 5 | **B12** | **2** | A32 | 8 | **A25** | **2** | B27 | 5 | C13 | 4 | A35 | 4.4 |
| **99** | B30 | 5 | **B15** | **2** | A35 | 8 | **A35** | **2** | B30 | 5 | C18 | 4 | B48 | 4.3 |
| **100** | B44 | 5 | **B16** | **2** | A43 | 8 | **B12** | **2** | B31 | 5 | C29 | 4 | C36 | 4.3 |
| **101** | C12 | 5 | **B2** | **2** | A45 | 8 | **B34** | **2** | B32 | 5 | C30 | 4 | A20 | 4.3 |
| **102** | C26 | 5 | **B26** | **2** | A46 | 8 | **B4** | **2** | B36 | 5 | C36 | 4 | A25 | 4.3 |
| **103** | C36 | 5 | **B38** | **2** | A48 | 8 | **B45** | **2** | B7 | 5 | C5 | 4 | A9 | 4.2 |
| **104** | C38 | 5 | **B45** | **2** | B20 | 8 | **B5** | **2** | B8 | 5 | C8 | 4 | A30 | 4.1 |
| **105** | D19 | 5 | **C15** | **2** | B29 | 8 | **C11** | **2** | C26 | 5 | A2 | 3 | C23 | 4.1 |
| **106** | D2 | 5 | **C2** | **2** | B34 | 8 | **C30** | **2** | C27 | 5 | A35 | 3 | C41 | 4.1 |
| **107** | A24 | 4 | **C21** | **2** | B43 | 8 | **C4** | **2** | C3 | 5 | B26 | 3 | B26 | 4.0 |
| **108** | A29 | 4 | **C3** | **2** | B44 | 8 | **C6** | **2** | C36 | 5 | B29 | 3 | B4 | 4.0 |
| **109** | A39 | 4 | **C31** | **2** | C18 | 8 | **C7** | **2** | C4 | 5 | B31 | 3 | A12 | 4.0 |
| **110** | A48 | 4 | **C40** | **2** | C22 | 8 | **A15** | **1** | C42 | 5 | B36 | 3 | A28 | 4.0 |
| **111** | A7 | 4 | **C46** | **2** | C23 | 8 | **A16** | **1** | C5 | 5 | B4 | 3 | A7 | 4.0 |
| **112** | A9 | 4 | **C48** | **2** | C27 | 8 | **A4** | **1** | A11 | 4 | C21 | 3 | A16 | 3.9 |
| **113** | B10 | 4 | **A11** | **1** | C3 | 8 | **A40** | **1** | A12 | 4 | C48 | 3 | C31 | 3.9 |
| **114** | B20 | 4 | **A13** | **1** | C36 | 8 | **A44** | **1** | A15 | 4 | C7 | 3 | C6 | 3.9 |
| **115** | B38 | 4 | **A34** | **1** | C43 | 8 | **A45** | **1** | A29 | 4 | **A19** | **2** | C38 | 3.9 |
| **116** | B5 | 4 | **A39** | **1** | C48 | 8 | **A47** | **1** | B26 | 4 | **A3** | **2** | A11 | 3.8 |
| **117** | C24 | 4 | **A44** | **1** | A47 | 7 | **A9** | **1** | B43 | 4 | **A32** | **2** | B38 | 3.8 |
| **118** | D9 | 4 | **A46** | **1** | B19 | 7 | **B22** | **1** | B45 | 4 | **A47** | **2** | B3 | 3.7 |
| **119** | A11 | 3 | **A47** | **1** | B38 | 7 | **B23** | **1** | C1 | 4 | **A48** | **2** | C47 | 3.6 |
| **120** | A19 | 3 | **A48** | **1** | B42 | 7 | **B29** | **1** | C10 | 4 | **B24** | **2** | A15 | 3.5 |
| **121** | A4 | 3 | **B18** | **1** | B7 | 7 | **B36** | **1** | C17 | 4 | **B3** | **2** | A45 | 3.5 |
| **122** | A40 | 3 | **B20** | **1** | C26 | 7 | **B39** | **1** | C23 | 4 | **B41** | **2** | B5 | 3.4 |
| **123** | A41 | 3 | **B24** | **1** | C30 | 7 | **B48** | **1** | C24 | 4 | **B47** | **2** | A29 | 3.4 |
| **124** | A44 | 3 | **B29** | **1** | C31 | 7 | **C17** | **1** | C31 | 4 | **C15** | **2** | B44 | 3.4 |
| **125** | B22 | 3 | **B34** | **1** | C38 | 7 | **C31** | **1** | C39 | 4 | **C2** | **2** | C26 | 3.4 |
| **126** | B23 | 3 | **B4** | **1** | D12 | 7 | **C32** | **1** | C45 | 4 | **C20** | **2** | B29 | 3.3 |
| **127** | B24 | 3 | **B43** | **1** | D27 | 7 | **A12** | **0** | C6 | 4 | **C22** | **2** | C7 | 3.3 |
| **128** | B32 | 3 | **C10** | **1** | A12 | 6 | **A13** | **0** | A2 | 3 | **C26** | **2** | B20 | 3.3 |
| **129** | B34 | 3 | **C23** | **1** | A41 | 6 | **A19** | **0** | A24 | 3 | **C38** | **2** | B10 | 3.3 |
| **130** | B46 | 3 | **C24** | **1** | B23 | 6 | **A20** | **0** | A32 | 3 | **C40** | **2** | B8 | 3.1 |
| **131** | C16 | 3 | **C26** | **1** | B40 | 6 | **A24** | **0** | A4 | 3 | **A16** | **1** | C45 | 3.1 |
| **132** | C22 | 3 | **C30** | **1** | C13 | 6 | **A28** | **0** | A46 | 3 | **A24** | **1** | A32 | 3.0 |
| **133** | C37 | 3 | **C4** | **1** | C40 | 6 | **A30** | **0** | A47 | 3 | **A25** | **1** | C22 | 3.0 |
| **134** | C39 | 3 | **C44** | **1** | C5 | 6 | **A31** | **0** | B20 | 3 | **A29** | **1** | C8 | 3.0 |
| **135** | C44 | 3 | **C8** | **1** | C6 | 6 | **A32** | **0** | B3 | 3 | **A33** | **1** | **B34** | **2.9** |
| **136** | C46 | 3 | **A10** | **0** | C8 | 6 | **A41** | **0** | B37 | 3 | **A34** | **1** | **C12** | **2.9** |
| **137** | C6 | 3 | **A16** | **0** | D26 | 6 | **A43** | **0** | B39 | 3 | **A39** | **1** | **B43** | **2.7** |
| **138** | C7 | 3 | **A19** | **0** | A15 | 5 | **A46** | **0** | B40 | 3 | **A41** | **1** | **B45** | **2.7** |
| **139** | C8 | 3 | **A24** | **0** | B10 | 5 | **A48** | **0** | B44 | 3 | **B16** | **1** | **A48** | **2.7** |
| **140** | D25 | 3 | **A29** | **0** | B3 | 5 | **A7** | **0** | B47 | 3 | **B23** | **1** | **C32** | **2.6** |
| **141** | D33 | 3 | **A31** | **0** | B4 | 5 | **B10** | **0** | B5 | 3 | **B32** | **1** | **C46** | **2.6** |
| **142** | D36 | 3 | **A32** | **0** | B45 | 5 | **B11** | **0** | C20 | 3 | **B34** | **1** | **A19** | **2.5** |
| **143** | **A15** | **2** | **A4** | **0** | B6 | 5 | **B16** | **0** | C38 | 3 | **B38** | **1** | **B40** | **2.4** |
| **144** | **A30** | **2** | **A40** | **0** | C44 | 5 | **B20** | **0** | C43 | 3 | **B43** | **1** | **A41** | **2.4** |
| **145** | **B31** | **2** | **A43** | **0** | D1 | 5 | **B24** | **0** | C47 | 3 | **B44** | **1** | **C42** | **2.4** |
| **146** | **B39** | **2** | **A7** | **0** | B37 | 4 | **B26** | **0** | **A19** | **2** | **B45** | **1** | **C43** | **2.4** |
| **147** | **B42** | **2** | **B10** | **0** | B46 | 4 | **B31** | **0** | **A20** | **2** | **B46** | **1** | **A47** | **2.3** |
| **148** | **B43** | **2** | **B23** | **0** | B5 | 4 | **B32** | **0** | **A25** | **2** | **C12** | **1** | **B47** | **2.3** |
| **149** | **B45** | **2** | **B27** | **0** | C16 | 4 | **B38** | **0** | **A41** | **2** | **C16** | **1** | **C20** | **2.3** |
| **150** | **C43** | **2** | **B31** | **0** | B31 | 3 | **B42** | **0** | **A45** | **2** | **C24** | **1** | **A46** | **2.2** |
| **151** | **C47** | **2** | **B32** | **0** | B39 | 3 | **B43** | **0** | **A9** | **2** | **C31** | **1** | **B37** | **2.2** |
| **152** | **D11** | **2** | **B37** | **0** | C20 | 3 | **B44** | **0** | **B13** | **2** | **C37** | **1** | **B23** | **2.2** |
| **153** | **D28** | **2** | **B39** | **0** | C41 | 3 | **B46** | **0** | **B16** | **2** | **C39** | **1** | **C40** | **2.2** |
| **154** | **D29** | **2** | **B40** | **0** | C45 | 3 | **B47** | **0** | **B23** | **2** | **C41** | **1** | **B31** | **2.1** |
| **155** | **A31** | **1** | **B42** | **0** | C7 | 3 | **B8** | **0** | **B29** | **2** | **C42** | **1** | **A43** | **2.1** |
| **156** | **A46** | **1** | **B46** | **0** | **A44** | **2** | **C10** | **0** | **B34** | **2** | **C43** | **1** | **B16** | **2.0** |
| **157** | **B40** | **1** | **B47** | **0** | **C24** | **2** | **C12** | **0** | **B42** | **2** | **C45** | **1** | **C24** | **2.0** |
| **158** | **C40** | **1** | **C16** | **0** | **C37** | **2** | **C13** | **0** | **B46** | **2** | **C46** | **1** | **B42** | **1.9** |
| **159** | **D12** | **1** | **C18** | **0** | **C42** | **2** | **C16** | **0** | **C22** | **2** | **C47** | **1** | **C44** | **1.9** |
| **160** | **D13** | **1** | **C20** | **0** | **C46** | **2** | **C20** | **0** | **C37** | **2** | **A4** | **0** | **C37** | **1.9** |
| **161** | **D17** | **1** | **C29** | **0** | **A40** | **1** | **C22** | **0** | **C40** | **2** | **A31** | **0** | **C39** | **1.9** |
| **162** | **D20** | **1** | **C32** | **0** | **B16** | **1** | **C23** | **0** | **C44** | **2** | **A40** | **0** | **B24** | **1.8** |
| **163** | **D26** | **1** | **C37** | **0** | **B32** | **1** | **C24** | **0** | **A28** | **1** | **A44** | **0** | **B32** | **1.7** |
| **164** | **D27** | **1** | **C39** | **0** | **B8** | **1** | **C26** | **0** | **A31** | **1** | **A45** | **0** | **B46** | **1.7** |
| **165** | **A43** | **0** | **C41** | **0** | **A24** | **0** | **C36** | **0** | **A40** | **1** | **A46** | **0** | **B39** | **1.6** |
| **166** | **A47** | **0** | **C42** | **0** | **A31** | **0** | **C40** | **0** | **A43** | **1** | **B37** | **0** | **A44** | **1.4** |
| **167** | **B37** | **0** | **C43** | **0** | **A4** | **0** | **C41** | **0** | **A44** | **1** | **B39** | **0** | **A24** | **1.3** |
| **168** | **B47** | **0** | **C45** | **0** | **B24** | **0** | **C42** | **0** | **A48** | **1** | **B40** | **0** | **C16** | **1.3** |
| **169** | **B48** | **0** | **C47** | **0** | **C12** | **0** | **C43** | **0** | **B21** | **1** | **B42** | **0** | **A4** | **1.2** |
| **170** | **D1** | **0** | **C5** | **0** | **C32** | **0** | **C44** | **0** | **C8** | **1** | **C32** | **0** | **A40** | **1.0** |
| **171** | **D3** | **0** | **C7** | **0** | **C39** | **0** | **C46** | **0** | **C16** | **0** | **C44** | **0** | **A31** | **0.3** |

#### 7.2.3.2 Plants Early Detection for All Competing Algorithms

The following table shows the number of early detections all algorithms had during days 1-13 (the stable period). The CDDRL did not have any early detections therefore it is not presented in the table. Plants with at least 10 early detections are colored in red and those with less than two early detections are colored in green. In addition, the average number of early detections of plants for all algorithms together is also given:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ADWIN (13 days)** | | **AWE (13 days)** | | **DWM (13 days)** | | **LNSE (13 days)** | | **SRPC (13 days)** | | **Average All (13 days)** | |
| **No.** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **# of mistakes** | **Plant** | **Avg # of mistakes** |
| **1** | **D33** | **11** | **A1** | **11** | **A11** | **12** | **D2** | **8** | **D43** | **12** | **D4** | **8.8** |
| **2** | **D43** | **11** | **A10** | **11** | **D16** | **12** | **D17** | **7** | **C17** | **11** | **D2** | **8.8** |
| **3** | **D3** | **10** | **A16** | **11** | **D5** | **12** | **D25** | **7** | **D10** | **11** | **D25** | **8.6** |
| **4** | **D4** | **10** | **A17** | **11** | **C17** | **11** | **D39** | **7** | **D17** | **11** | **D10** | **8.0** |
| **5** | **D24** | **9** | **A19** | **11** | **D10** | **11** | **D4** | **7** | **D2** | **11** | **D17** | **8.0** |
| **6** | **D28** | **9** | **A20** | **11** | **C10** | **10** | B12 | 6 | **D31** | **11** | **D31** | **8.0** |
| **7** | **D47** | **9** | **A21** | **11** | **D31** | **10** | B6 | 6 | **D33** | **11** | **D33** | **8.0** |
| **8** | **D48** | **9** | **A22** | **11** | C40 | 9 | D3 | 6 | **D25** | **10** | **D43** | **8.0** |
| **9** | **D2** | **8** | **A28** | **11** | A20 | 8 | D37 | 6 | **D30** | **10** | **D5** | **7.6** |
| **10** | **D25** | **8** | **A29** | **11** | B12 | 8 | A20 | 5 | **D4** | **10** | **C17** | **7.4** |
| **11** | **D21** | **7** | **A30** | **11** | B30 | 8 | B24 | 5 | A6 | 9 | **D48** | **7.4** |
| **12** | **D29** | **7** | **A32** | **11** | B5 | 8 | B5 | 5 | B10 | 9 | **A20** | **7.0** |
| **13** | **D31** | **7** | **A41** | **11** | B6 | 8 | B8 | 5 | B36 | 9 | **D27** | **7.0** |
| **14** | D36 | 7 | **A43** | **11** | C13 | 8 | C15 | 5 | B5 | 9 | **D30** | **7.0** |
| **15** | D39 | 7 | **A45** | **11** | C6 | 8 | D11 | 5 | C13 | 9 | C13 | 6.8 |
| **16** | D40 | 7 | **A46** | **11** | D9 | 8 | D18 | 5 | C20 | 9 | D16 | 6.8 |
| **17** | D10 | 6 | **A47** | **11** | A18 | 7 | D40 | 5 | C36 | 9 | D26 | 6.8 |
| **18** | D26 | 6 | **A48** | **11** | A2 | 7 | D5 | 5 | D16 | 9 | B30 | 6.6 |
| **19** | D30 | 6 | **A6** | **11** | B17 | 7 | A17 | 4 | D26 | 9 | B6 | 6.6 |
| **20** | D34 | 6 | **B14** | **11** | B19 | 7 | B16 | 4 | D27 | 9 | D37 | 6.6 |
| **21** | D45 | 6 | **B15** | **11** | B22 | 7 | B2 | 4 | D3 | 9 | D21 | 6.6 |
| **22** | D46 | 6 | **B17** | **11** | B3 | 7 | B27 | 4 | D36 | 9 | D34 | 6.4 |
| **23** | C29 | 5 | **B18** | **11** | C4 | 7 | B3 | 4 | D5 | 9 | D19 | 6.4 |
| **24** | D17 | 5 | **B2** | **11** | D19 | 7 | B36 | 4 | D7 | 9 | D47 | 6.4 |
| **25** | D19 | 5 | **B20** | **11** | D25 | 7 | B38 | 4 | A11 | 8 | D39 | 6.2 |
| **26** | D27 | 5 | **B21** | **11** | D39 | 7 | B40 | 4 | A17 | 8 | D12 | 6.2 |
| **27** | D37 | 5 | **B22** | **11** | A17 | 6 | B42 | 4 | A18 | 8 | D28 | 6.2 |
| **28** | D5 | 5 | **B23** | **11** | A9 | 6 | C4 | 4 | A19 | 8 | D36 | 6.2 |
| **29** | D9 | 5 | **B26** | **11** | B1 | 6 | C7 | 4 | A20 | 8 | B36 | 6.2 |
| **30** | A41 | 4 | **B27** | **11** | B11 | 6 | D27 | 4 | A7 | 8 | A17 | 6.2 |
| **31** | A48 | 4 | **B29** | **11** | B2 | 6 | D32 | 4 | A9 | 8 | B22 | 6.2 |
| **32** | C24 | 4 | **B30** | **11** | B23 | 6 | D34 | 4 | B11 | 8 | C10 | 6.2 |
| **33** | C43 | 4 | **B31** | **11** | B4 | 6 | D36 | 4 | B17 | 8 | D13 | 6.0 |
| **34** | D12 | 4 | **B34** | **11** | B7 | 6 | D45 | 4 | B22 | 8 | B12 | 6.0 |
| **35** | D22 | 4 | **B36** | **11** | C1 | 6 | D48 | 4 | B29 | 8 | D11 | 5.8 |
| **36** | D8 | 4 | **B43** | **11** | C20 | 6 | A30 | 3 | B3 | 8 | D46 | 5.8 |
| **37** | A20 | 3 | **B46** | **11** | C26 | 6 | A32 | 3 | B30 | 8 | D9 | 5.8 |
| **38** | A24 | 3 | **B47** | **11** | C3 | 6 | A33 | 3 | B7 | 8 | D29 | 5.8 |
| **39** | B12 | 3 | **B7** | **11** | C5 | 6 | A4 | 3 | C10 | 8 | D8 | 5.8 |
| **40** | B22 | 3 | **C1** | **11** | D1 | 6 | A48 | 3 | C18 | 8 | D3 | 5.8 |
| **41** | B27 | 3 | **C10** | **11** | D12 | 6 | B11 | 3 | C23 | 8 | C20 | 5.8 |
| **42** | B30 | 3 | **C13** | **11** | D13 | 6 | B23 | 3 | C27 | 8 | B7 | 5.8 |
| **43** | B6 | 3 | **C14** | **11** | D14 | 6 | B30 | 3 | C29 | 8 | C29 | 5.8 |
| **44** | B7 | 3 | **C17** | **11** | D17 | 6 | B32 | 3 | D12 | 8 | B27 | 5.8 |
| **45** | C13 | 3 | **C18** | **11** | D2 | 6 | C11 | 3 | D13 | 8 | C6 | 5.8 |
| **46** | D11 | 3 | **C19** | **11** | D20 | 6 | C13 | 3 | D19 | 8 | A48 | 5.8 |
| **47** | D18 | 3 | **C20** | **11** | D26 | 6 | C17 | 3 | D21 | 8 | C4 | 5.8 |
| **48** | D23 | 3 | **C21** | **11** | D27 | 6 | C26 | 3 | D24 | 8 | D18 | 5.6 |
| **49** | D32 | 3 | **C22** | **11** | D30 | 6 | C27 | 3 | D28 | 8 | D40 | 5.6 |
| **50** | D38 | 3 | **C23** | **11** | D4 | 6 | C31 | 3 | D29 | 8 | D45 | 5.6 |
| **51** | A17 | 2 | **C26** | **11** | D45 | 6 | C34 | 3 | D47 | 8 | D14 | 5.6 |
| **52** | A30 | 2 | **C27** | **11** | D6 | 6 | C41 | 3 | D48 | 8 | D20 | 5.6 |
| **53** | A31 | 2 | **C28** | **11** | A1 | 5 | C43 | 3 | D8 | 8 | D22 | 5.6 |
| **54** | A32 | 2 | **C29** | **11** | A10 | 5 | C48 | 3 | A16 | 7 | D7 | 5.6 |
| **55** | B29 | 2 | **C3** | **11** | A13 | 5 | C6 | 3 | A28 | 7 | C36 | 5.6 |
| **56** | B31 | 2 | **C30** | **11** | A15 | 5 | D1 | 3 | A29 | 7 | C27 | 5.6 |
| **57** | B36 | 2 | **C31** | **11** | A19 | 5 | D13 | 3 | A4 | 7 | C15 | 5.6 |
| **58** | B41 | 2 | **C36** | **11** | A47 | 5 | D14 | 3 | B19 | 7 | C7 | 5.6 |
| **59** | B48 | 2 | **C4** | **11** | A48 | 5 | D21 | 3 | B21 | 7 | C43 | 5.6 |
| **60** | B8 | 2 | **C42** | **11** | A6 | 5 | D24 | 3 | B27 | 7 | A6 | 5.4 |
| **61** | C20 | 2 | **C43** | **11** | A7 | 5 | D28 | 3 | B6 | 7 | B5 | 5.4 |
| **62** | C23 | 2 | **C47** | **11** | B13 | 5 | D33 | 3 | C15 | 7 | A11 | 5.4 |
| **63** | C27 | 2 | **C48** | **11** | B18 | 5 | D43 | 3 | C3 | 7 | A18 | 5.4 |
| **64** | C36 | 2 | **C7** | **11** | B20 | 5 | D9 | 3 | C30 | 7 | B17 | 5.4 |
| **65** | C47 | 2 | **D12** | **11** | B21 | 5 | A11 | 2 | C48 | 7 | C23 | 5.4 |
| **66** | C5 | 2 | **D13** | **11** | B36 | 5 | A12 | 2 | C6 | 7 | B23 | 5.4 |
| **67** | C6 | 2 | **D14** | **11** | B48 | 5 | A15 | 2 | C7 | 7 | B11 | 5.2 |
| **68** | D13 | 2 | **D15** | **11** | C15 | 5 | A18 | 2 | C8 | 7 | B29 | 5.2 |
| **69** | D16 | 2 | **D17** | **11** | C18 | 5 | A24 | 2 | D14 | 7 | B3 | 5.2 |
| **70** | D20 | 2 | **D19** | **11** | C19 | 5 | A28 | 2 | D15 | 7 | C18 | 5.2 |
| **71** | **A10** | **1** | **D2** | **11** | C36 | 5 | A3 | 2 | D20 | 7 | B19 | 5.2 |
| **72** | **A15** | **1** | **D20** | **11** | C7 | 5 | A35 | 2 | D22 | 7 | C48 | 5.2 |
| **73** | **A21** | **1** | **D21** | **11** | D11 | 5 | A45 | 2 | D34 | 7 | A30 | 5.2 |
| **74** | **A22** | **1** | **D22** | **11** | D15 | 5 | A47 | 2 | D37 | 7 | A32 | 5.2 |
| **75** | **A29** | **1** | **D25** | **11** | D18 | 5 | A6 | 2 | D39 | 7 | B2 | 5.2 |
| **76** | **A3** | **1** | **D26** | **11** | D38 | 5 | A7 | 2 | D46 | 7 | D1 | 5.0 |
| **77** | **A33** | **1** | **D27** | **11** | D48 | 5 | B10 | 2 | D9 | 7 | D15 | 5.0 |
| **78** | **A39** | **1** | **D30** | **11** | D7 | 5 | B13 | 2 | A13 | 6 | A9 | 5.0 |
| **79** | **A40** | **1** | **D31** | **11** | A16 | 4 | B19 | 2 | A15 | 6 | C3 | 5.0 |
| **80** | **A43** | **1** | **D33** | **11** | A21 | 4 | B20 | 2 | A21 | 6 | B20 | 5.0 |
| **81** | **A46** | **1** | **D34** | **11** | A22 | 4 | B22 | 2 | A22 | 6 | A41 | 5.0 |
| **82** | **A9** | **1** | **D37** | **11** | A28 | 4 | B26 | 2 | A25 | 6 | A19 | 4.8 |
| **83** | **B1** | **1** | **D4** | **11** | A29 | 4 | B28 | 2 | A30 | 6 | A28 | 4.8 |
| **84** | **B11** | **1** | **D43** | **11** | A3 | 4 | B31 | 2 | A32 | 6 | C31 | 4.8 |
| **85** | **B13** | **1** | **D46** | **11** | A30 | 4 | B41 | 2 | A48 | 6 | B31 | 4.8 |
| **86** | **B16** | **1** | **D47** | **11** | A32 | 4 | B45 | 2 | B1 | 6 | C26 | 4.8 |
| **87** | **B20** | **1** | **D48** | **11** | A4 | 4 | B46 | 2 | B15 | 6 | D38 | 4.6 |
| **88** | **B23** | **1** | **D7** | **11** | A40 | 4 | B48 | 2 | B20 | 6 | A7 | 4.6 |
| **89** | **B38** | **1** | **D8** | **11** | A41 | 4 | C1 | 2 | B23 | 6 | A29 | 4.6 |
| **90** | **B39** | **1** | **A13** | **10** | A43 | 4 | C23 | 2 | B26 | 6 | B21 | 4.6 |
| **91** | **B40** | **1** | **A18** | **10** | A45 | 4 | C32 | 2 | B38 | 6 | C30 | 4.6 |
| **92** | **B43** | **1** | **A9** | **10** | A46 | 4 | C45 | 2 | B4 | 6 | C8 | 4.6 |
| **93** | **B47** | **1** | **B19** | **10** | B10 | 4 | C5 | 2 | B8 | 6 | A15 | 4.6 |
| **94** | **C10** | **1** | **B28** | **10** | B14 | 4 | C8 | 2 | C12 | 6 | B1 | 4.6 |
| **95** | **C15** | **1** | **B40** | **10** | B15 | 4 | D10 | 2 | C22 | 6 | B26 | 4.6 |
| **96** | **C17** | **1** | **C15** | **10** | B26 | 4 | D12 | 2 | C31 | 6 | B8 | 4.6 |
| **97** | **C18** | **1** | **C34** | **10** | B27 | 4 | D20 | 2 | C4 | 6 | B40 | 4.6 |
| **98** | **C2** | **1** | **C41** | **10** | B28 | 4 | D22 | 2 | C40 | 6 | C1 | 4.6 |
| **99** | **C28** | **1** | **C8** | **10** | B29 | 4 | D23 | 2 | C43 | 6 | D32 | 4.4 |
| **100** | **C37** | **1** | **D1** | **10** | B31 | 4 | D26 | 2 | D1 | 6 | D24 | 4.4 |
| **101** | **C38** | **1** | **D10** | **10** | B34 | 4 | D29 | 2 | D11 | 6 | A16 | 4.4 |
| **102** | **C39** | **1** | **D11** | **10** | B37 | 4 | D30 | 2 | D23 | 6 | A21 | 4.4 |
| **103** | **C4** | **1** | **D16** | **10** | B38 | 4 | D6 | 2 | D32 | 6 | A22 | 4.4 |
| **104** | **C41** | **1** | **D18** | **10** | B40 | 4 | D7 | 2 | D38 | 6 | B38 | 4.4 |
| **105** | **C48** | **1** | **D29** | **10** | B41 | 4 | D8 | 2 | D40 | 6 | C22 | 4.4 |
| **106** | **C7** | **1** | A15 | 9 | B43 | 4 | **A2** | **1** | D45 | 6 | A10 | 4.4 |
| **107** | **D14** | **1** | A25 | 9 | B44 | 4 | **A27** | **1** | A1 | 5 | A43 | 4.4 |
| **108** | **D15** | **1** | A44 | 9 | B46 | 4 | **A31** | **1** | A10 | 5 | B46 | 4.4 |
| **109** | **D6** | **1** | B1 | 9 | B8 | 4 | **A39** | **1** | A2 | 5 | A13 | 4.2 |
| **110** | **D7** | **1** | B6 | 9 | C11 | 4 | **A41** | **1** | A27 | 5 | B15 | 4.2 |
| **111** | **D1** | **0** | C12 | 9 | C12 | 4 | **A43** | **1** | A31 | 5 | A1 | 4.2 |
| **112** | **A27** | **0** | C2 | 9 | C14 | 4 | **A44** | **1** | A33 | 5 | B28 | 4.2 |
| **113** | **B17** | **0** | C6 | 9 | C2 | 4 | **A46** | **1** | A41 | 5 | C14 | 4.2 |
| **114** | **A6** | **0** | A12 | 8 | C21 | 4 | **B1** | **1** | A43 | 5 | C19 | 4.2 |
| **115** | **B14** | **0** | A7 | 8 | C22 | 4 | **B17** | **1** | B12 | 5 | C21 | 4.2 |
| **116** | **B21** | **0** | B11 | 8 | C23 | 4 | **B29** | **1** | B13 | 5 | C28 | 4.2 |
| **117** | **B28** | **0** | B12 | 8 | C27 | 4 | **B37** | **1** | B14 | 5 | C34 | 4.2 |
| **118** | **B3** | **0** | B41 | 8 | C28 | 4 | **B4** | **1** | B16 | 5 | C42 | 4.2 |
| **119** | **C1** | **0** | B44 | 8 | C29 | 4 | **B43** | **1** | B2 | 5 | A46 | 4.2 |
| **120** | **C19** | **0** | B48 | 8 | C30 | 4 | **B47** | **1** | B28 | 5 | C41 | 4.2 |
| **121** | **A18** | **0** | C11 | 8 | C31 | 4 | **B7** | **1** | B31 | 5 | C47 | 4.2 |
| **122** | **A45** | **0** | C24 | 8 | C42 | 4 | **C10** | **1** | B46 | 5 | A47 | 4.2 |
| **123** | **B19** | **0** | D36 | 8 | C43 | 4 | **C14** | **1** | B47 | 5 | D23 | 4.0 |
| **124** | **B44** | **0** | D38 | 8 | C48 | 4 | **C16** | **1** | C11 | 5 | B10 | 4.0 |
| **125** | **B5** | **0** | D40 | 8 | C8 | 4 | **C18** | **1** | C14 | 5 | B13 | 4.0 |
| **126** | **C11** | **0** | A3 | 7 | D21 | 4 | **C2** | **1** | C19 | 5 | B14 | 4.0 |
| **127** | **C12** | **0** | A33 | 7 | D22 | 4 | **C20** | **1** | C21 | 5 | B47 | 4.0 |
| **128** | **C14** | **0** | A34 | 7 | D23 | 4 | **C21** | **1** | C24 | 5 | C11 | 4.0 |
| **129** | **C22** | **0** | B13 | 7 | D28 | 4 | **C22** | **1** | C28 | 5 | C24 | 4.0 |
| **130** | **C34** | **0** | B3 | 7 | D32 | 4 | **C29** | **1** | C34 | 5 | C5 | 4.0 |
| **131** | **A1** | **0** | B38 | 7 | D33 | 4 | **C3** | **1** | C37 | 5 | A45 | 4.0 |
| **132** | **A12** | **0** | D28 | 7 | D34 | 4 | **C30** | **1** | C38 | 5 | B43 | 4.0 |
| **133** | **A2** | **0** | D5 | 7 | D37 | 4 | **C36** | **1** | C42 | 5 | B48 | 4.0 |
| **134** | **A25** | **0** | A2 | 6 | D46 | 4 | **C38** | **1** | C5 | 5 | D6 | 3.8 |
| **135** | **A28** | **0** | B4 | 6 | D47 | 4 | **C39** | **1** | D18 | 5 | B4 | 3.8 |
| **136** | **A35** | **0** | B8 | 6 | D8 | 4 | **C42** | **1** | D6 | 5 | C12 | 3.8 |
| **137** | **B15** | **0** | C37 | 6 | A12 | 3 | **C46** | **1** | A39 | 4 | C40 | 3.8 |
| **138** | **B2** | **0** | D45 | 6 | A24 | 3 | **C47** | **1** | A46 | 4 | A2 | 3.8 |
| **139** | **B26** | **0** | D9 | 6 | A25 | 3 | **D15** | **1** | B24 | 4 | B18 | 3.8 |
| **140** | **B45** | **0** | A11 | 5 | A35 | 3 | **D16** | **1** | B37 | 4 | B41 | 3.8 |
| **141** | **C21** | **0** | A27 | 5 | C16 | 3 | **D19** | **1** | B40 | 4 | A25 | 3.6 |
| **142** | **C3** | **0** | A35 | 5 | C24 | 3 | **D31** | **1** | C1 | 4 | A33 | 3.6 |
| **143** | **C31** | **0** | B10 | 5 | C34 | 3 | **D38** | **1** | C26 | 4 | B34 | 3.6 |
| **144** | **C40** | **0** | B32 | 5 | C41 | 3 | **D46** | **1** | C41 | 4 | C2 | 3.6 |
| **145** | **C46** | **0** | B37 | 5 | C47 | 3 | **D47** | **0** | C47 | 4 | A4 | 3.4 |
| **146** | **A11** | **0** | B42 | 5 | D3 | 3 | **A1** | **0** | A12 | 3 | A3 | 3.4 |
| **147** | **A13** | **0** | B5 | 5 | D36 | 3 | **A10** | **0** | A3 | 3 | A12 | 3.2 |
| **148** | **A34** | **0** | C46 | 5 | D43 | 3 | **A13** | **0** | A34 | 3 | B44 | 3.0 |
| **149** | **A44** | **0** | C5 | 5 | A27 | 2 | **A16** | **0** | A35 | 3 | C37 | 2.8 |
| **150** | **A47** | **0** | D23 | 5 | A31 | 2 | **A19** | **0** | A40 | 3 | B37 | 2.8 |
| **151** | **B18** | **0** | D32 | 5 | A33 | 2 | **A21** | **0** | A45 | 3 | A44 | 2.8 |
| **152** | **B24** | **0** | D6 | 5 | A34 | 2 | **A22** | **0** | A47 | 3 | A27 | 2.6 |
| **153** | **B34** | **0** | C38 | 4 | A39 | 2 | **A25** | **0** | B18 | 3 | A31 | 2.6 |
| **154** | **B4** | **0** | C40 | 4 | A44 | 2 | **A29** | **0** | B34 | 3 | A35 | 2.6 |
| **155** | **C26** | **0** | A24 | 3 | B32 | 2 | **A34** | **0** | B41 | 3 | B16 | 2.4 |
| **156** | **C30** | **0** | A31 | 3 | B47 | 2 | **A40** | **0** | B43 | 3 | C38 | 2.4 |
| **157** | **C44** | **0** | A39 | 3 | C32 | 2 | **A9** | **0** | B44 | 3 | A34 | 2.4 |
| **158** | **C8** | **0** | A4 | 3 | C37 | 2 | **B14** | **0** | B48 | 3 | B32 | 2.4 |
| **159** | **A16** | **0** | C39 | 3 | C39 | 2 | **B15** | **0** | C2 | 3 | B42 | 2.4 |
| **160** | **A19** | **0** | D39 | 3 | C44 | 2 | **B18** | **0** | C39 | 3 | A24 | 2.4 |
| **161** | **A4** | **0** | B39 | 2 | C45 | 2 | **B21** | **0** | C46 | 3 | A39 | 2.2 |
| **162** | **A7** | **0** | C16 | 2 | C46 | 2 | **B34** | **0** | A44 | 2 | C46 | 2.2 |
| **163** | **B10** | **0** | C32 | 2 | D29 | 2 | **B39** | **0** | B32 | 2 | B24 | 2.0 |
| **164** | **B32** | **0** | C44 | 2 | D40 | 2 | **B44** | **0** | B39 | 2 | C39 | 2.0 |
| **165** | **B37** | **0** | **A40** | **1** | **B16** | **1** | **C12** | **0** | B42 | 2 | **A40** | **1.8** |
| **166** | **B42** | **0** | **B16** | **1** | **B24** | **1** | **C19** | **0** | C16 | 2 | **C16** | **1.6** |
| **167** | **B46** | **0** | **C45** | **1** | **B39** | **1** | **C24** | **0** | C32 | 2 | **C32** | **1.6** |
| **168** | **C16** | **0** | **D24** | **1** | **B42** | **1** | **C28** | **0** | C45 | 2 | **C45** | **1.4** |
| **169** | **C32** | **0** | **D3** | **1** | **B45** | **1** | **C37** | **0** | **A24** | **1** | **B39** | **1.2** |
| **170** | **C42** | **0** | **A24** | **0** | **C38** | **1** | **C40** | **0** | **B45** | **1** | **C44** | **1.0** |
| **171** | **C45** | **0** | **A31** | **0** | **D24** | **1** | **C44** | **0** | **C44** | **1** | **B45** | **0.8** |

#### 7.2.3.3 Good and Bad Plants Summary and Visualization

The following table shows only the top/bottom average performance of plants over all algorithms and for the CDDRL separately. This table is a summary of the above tables.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bad Plants - Stress Days** | | | | **Good Plants - Stress Days** | | | | **Early Detected Plants** | |
| **Average All** | | **CDDRL (22 days)** | | **Average All** | | **CDDRL (22 days)** | | **Average All (13 days)** | |
| **Plant** | **Avg # of mistakes** | **Plant** | **# of mistakes** | **Plant** | **Avg # of mistakes** | **Plant** | **# of mistakes** | **Plant** | **Avg # of mistakes** |
| **D32** | **20.5** | **B14** | **19** | **A31** | **0.3** | **A43** | **0** | **D4** | **8.8** |
| **D7** | **20.3** | **C28** | **18** | **A40** | **1.0** | **A47** | **0** | **D2** | **8.8** |
| **D24** | **19.5** | **D39** | **18** | **A4** | **1.2** | **B37** | **0** | **D25** | **8.6** |
| **D8** | **19.5** | **D40** | **18** | **A24** | **1.3** | **B47** | **0** | **D10** | **8.0** |
| **D45** | **19.2** | **C19** | **17** | **C16** | **1.3** | **B48** | **0** | **D17** | **8.0** |
| **D16** | **19.0** | **C14** | **16** | **A44** | **1.4** | **D1** | **0** | **D31** | **8.0** |
| **D48** | **18.8** | **C15** | **16** | **B39** | **1.6** | **D3** | **0** | **D33** | **8.0** |
| **D23** | **18.6** | **D38** | **16** | **B32** | **1.7** | **A31** | **1** | **D43** | **8.0** |
| **D43** | **18.4** | **D16** | **15** | **B46** | **1.7** | **A46** | **1** | **D5** | **7.6** |
| **D47** | **18.4** | **D23** | **15** | **B24** | **1.8** | **B40** | **1** | **C17** | **7.4** |
| **D31** | **17.8** | **D37** | **15** | **C37** | **1.9** | **C40** | **1** | **D48** | **7.4** |
| **D21** | **17.7** | **D46** | **15** | **C39** | **1.9** | **D12** | **1** | **A20** | **7.0** |
| **D30** | **17.0** | **D6** | **15** | **B42** | **1.9** | **D13** | **1** | **D27** | **7.0** |
| **D40** | **16.7** | **A17** | **14** | **C44** | **1.9** | **D17** | **1** | **D30** | **7.0** |
| **D39** | **16.2** | **A18** | **14** |  |  | **D20** | **1** |
| **D6** | **16.0** | **A27** | **14** |  |  | **D26** | **1** |
| **D33** | **15.7** | **B2** | **14** |  |  | **D27** | **1** |
| **D46** | **15.7** | **D45** | **14** |
| **D14** | **15.6** | **D8** | **14** |
| **D29** | **15.4** | **A10** | **13** |
| **D5** | **15.2** | **A21** | **13** |

It is very clear that in general all competing algorithms are performing not well on detection of D plants. More disturbing is that although on average the competing algorithms pre detect mostly plants from treatment D, they fail in average to detect those same plant in the stress period.

I chose to visualize two top ranked plants from each category over some experiment days to observe weather there is a noticeable pattern for these specific plants.

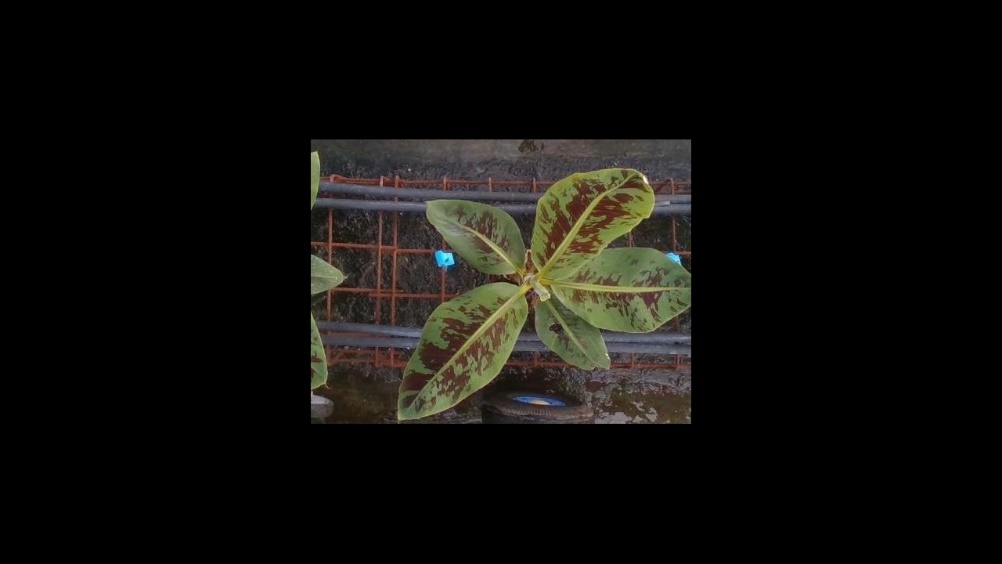
All images for the stress period are in this order: day 14 -> day 21 -> day 28 -> day 37.

Images for the stable period are in this order: day 1 -> day 7 -> day 13.

We can observe that most of the plants presented behave according to their description above i.e., bad plants from treatment D look healthier, early detected plants from treatment D look smaller than the other plants, well detected plants from treatment A look healthy as opposed to good detected plants from treatment D that look smaller and less healthy.

Representing the ‘Average All Bad Plants – Stress Days’ are plants D7 and D32:

**D7:**

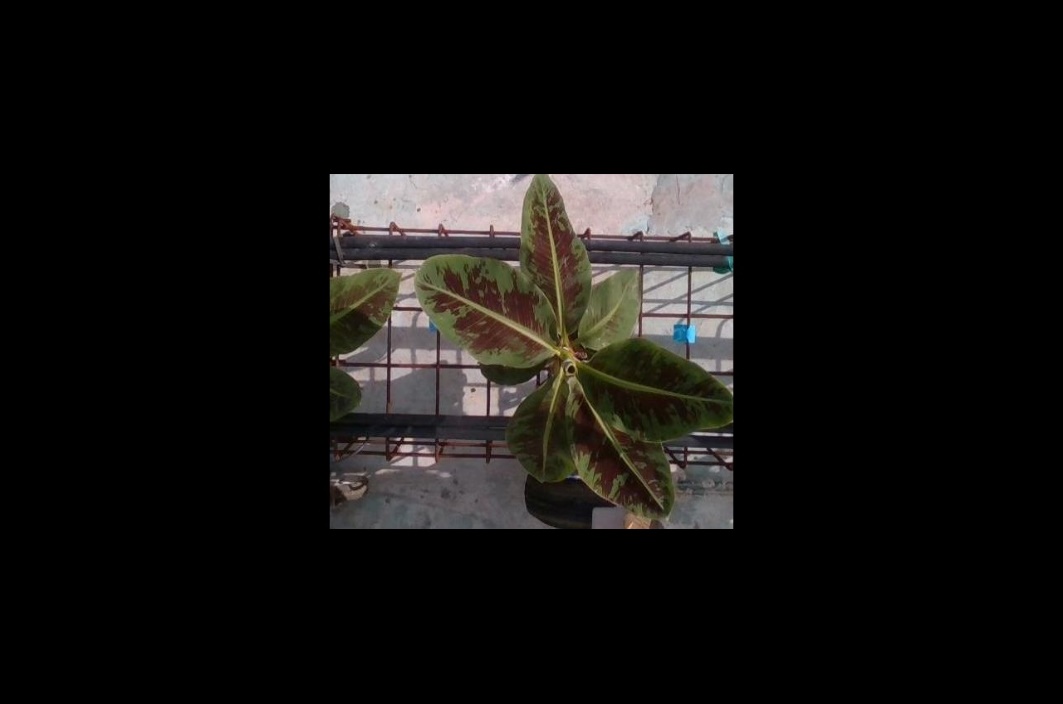
   

**D32:**

Representing the ‘CDDRL Bad Plants – Stress Days’ are plants B14 and C28:

**B14:**

**C28:**

A picture containing text, vegetable, picture frame

Description automatically generated  A picture containing text, screen

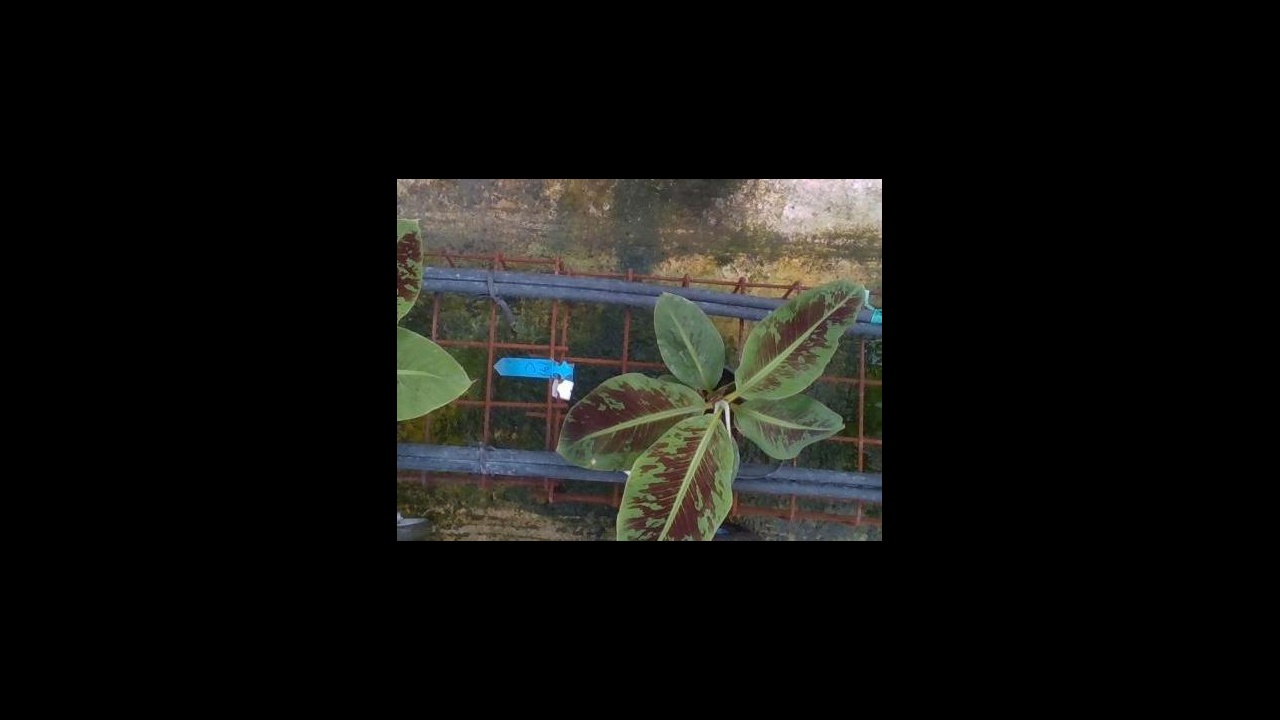
Description automatically generated 

Representing the ‘Average All Good Plants – Stress Days’ are plants A31 and A40:

**A31:**

**A40:**

 A picture containing text

Description automatically generated  

Representing the ‘CDDRL Good Plants – Stress Days’ are plants B37 and D3:

**B37:**

  Graphical user interface

Description automatically generated 

**D3:**

Representing the ‘Early Detected Plants – Stable Days’ are plants D2 and D4:

**D2:**

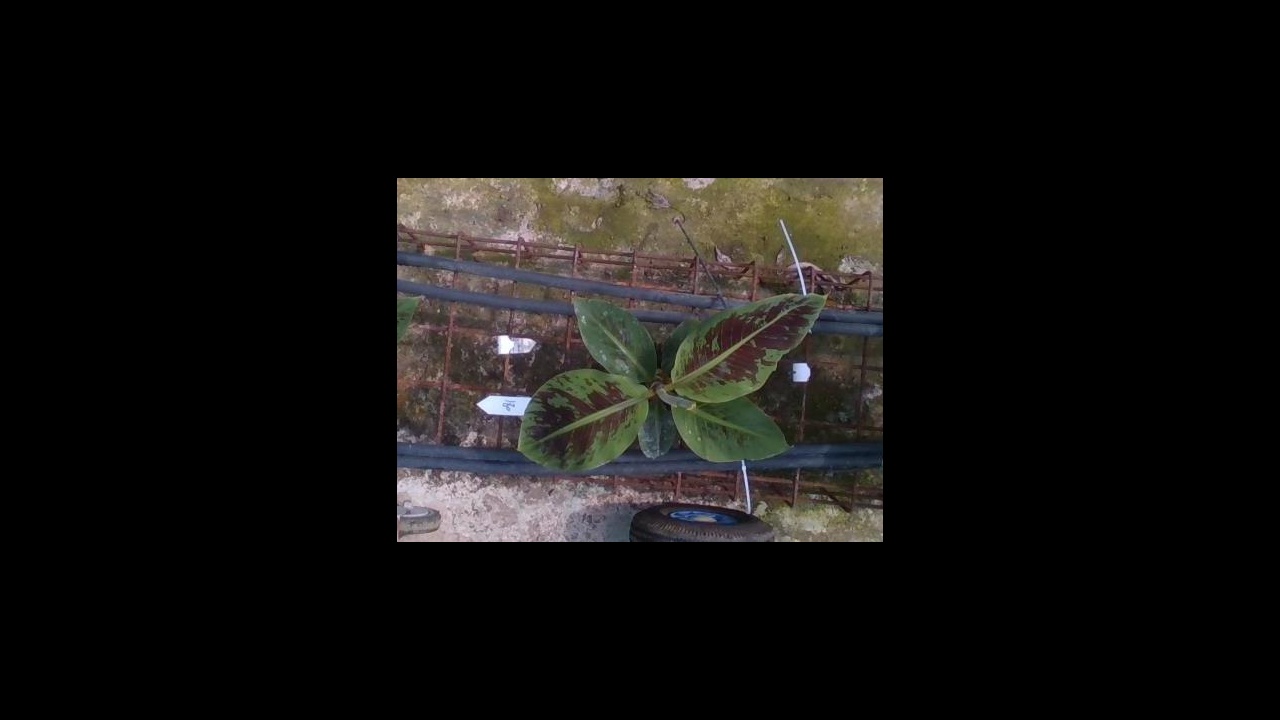
**D4:**

A picture containing text

Description automatically generated  

Representing the ‘Undetected Early Plants – Stable Days’ are plants B45 and C44:

**B45:**

**C44:**

Graphical user interface

Description automatically generated with low confidence  