An Empirical Analysis of Data Drift Detection Techniques in Machine Learning Systems

Lucas Helfstein[1] **and Kelly Rosa Braghetto**[1]

[1]Institute of Mathematics and Statistics - University of São Paulo (USP)
São Paulo – SP – Brazil

{lucashelfs,kellyrb}@ime.usp.br

Abstract. *Software systems that have machine learning (ML) components are being used in a wide range of domains. Developers of such systems face challenges different from those of traditional systems because the performance of ML systems is directly linked to their input data. This work shows that ML systems can be improved over time by actively monitoring the data that passes through them and retraining their models in case of drift detection. To this end, we first assess some widely used statistical and distance-based methods for data drift detection, discussing their pros and cons. Then, we present results from experiments performed using these methods in real-world and synthetic datasets to detect data drifts and improve the system's robustness automatically.*

1. Introduction

Software systems with machine learning components are being increasingly adopted across various domains. Machine learning system developers face challenges that differ from traditional software systems in that their performance is directly linked to input data. After a machine learning model has been deployed in a software system and is being used in production, it is subject to input data that may differ to several degrees from what was initially used to train the model [Gama and Castillo 2006]. At this stage, machine learning systems can exhibit behaviors unforeseen by their developers, which can lead to significant project failures. When developing software with machine learning components, decisions related to operational requirements for offering, monitoring, and retraining models are crucial for systems to function correctly [Sculley et al. 2015].

In many applications, machine learning models are deployed in environments where they continuously process incoming data streams without receiving immediate feedback on their performance. This lack of real-time feedback poses a significant challenge in maintaining the accuracy and robustness of the models. To address this challenge, developers can implement data drift detection techniques to monitor the input data and ensure its consistency with the data used during the model's training phase [Lu et al. 2018].

This paper explores the application of data drift detection techniques with the primary objective of enhancing classifier performance. As an approach, we employed nonparametric methods based on the Hellinger Distance (HD), the Janson-Shannon (JS) Divergence, and the Kolmogorov-Smirnov (KS) Test. More specifically, we evaluated the Hellinger Distance Drift Detection Method [Ditzler and Polikar 2011] and leveraged its adaptative threshold approach to devise a new detection method based on the JS Divergence.

These techniques offer distinct advantages: the HD and JS methods quantify the discrepancy between probability distributions, while the KS method provides a statistical framework that can be used to assess changes in multivariate data. By integrating drift detection methods into the classification pipeline, we aim to dynamically adapt to changing data environments, thereby improving the model.

Through empirical evaluation of real-world and synthetic datasets exhibiting concept and data drift under varied configuration scenarios, we demonstrate the effectiveness of our approach in maintaining classification performance in evolving data landscapes.

2. Data Drift Detection

In machine learning systems, a common component is the classifier. In a classification task, a model learns a function f that maps input variables ($\mathcal{X}$), representing the feature space, to discrete output labels ($\mathcal{Y}$). After the model is trained and the system is deployed, it is subjected to two different types of deviations: *data drifts* and *concept drifts*.

Data drift occurs when the distribution of data input to the system in production becomes distant from that of the data $\mathcal{D}$ used in model training, making it possible to detect a significant difference between them. Concept drift occurs when the relationship between the training attributes and the model classes changes [Webb et al. 2016]; that is, the attributes that determined one class start to determine another. In a general way, a drift occurs whenever one of the probability distributions $P(\mathcal{X})$, $P(\mathcal{X}|\mathcal{Y})$ or $P(\mathcal{Y}|\mathcal{X})$ changes over time. Moreover, drifts may occur with different velocities and patterns (e.g., abrupt, gradual, incremental, reoccurring) [Souza et al. 2020].

To detect drifts, it is necessary to monitor the extent of the divergence between the distributions of the sets. In this work, we focus on two categories of drift detection techniques very commonly used in practice: *statistical* and *distance-based* methods.

2.1. Statistical Methods

Statistical methods typically use a formal statistical hypothesis test to compare the distributions of two datasets (e.g., current data vs. reference data). They generate a statistical measure (e.g., p-value) indicating the likelihood that the two distributions are the same. A low p-value suggests that the distributions are significantly different, indicating drift. They can be sensitive to sample size and assumptions about the underlying distributions. When data scarcity is not a concern, nonparametric methods (which do not assume any specific data distribution) are more often used [Dasu et al. 2006]. Common nonparametric tests include the Wilcoxon, Kolmogorov-Smirnov, and multinomial tests.

2.1.1. Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov (KS) statistical test can be used to test whether two samples came from the same distribution. Assuming F and G are the empirical distribution functions of the two samples and the sample's sizes are n and m, respectively, the Kolmogorov–Smirnov statistic $KS(F, G)$ is defined as follows [Hodges Jr 1958]:

$$KS(F, G) = \sup_x |F(x) - G(x)| \tag{1}$$

The decision rule is to reject the hypothesis at the significance level α if $KS(F, G) > c(\alpha)\sqrt{\frac{n+m}{n\times m}}$, where $c(\alpha)$ is given in the Kolmogorov table [Hodges Jr 1958].

2.1.2. Multiple Univariate Kolmogorov-Smirnov Test

While the univariate KS Test assesses the difference between the empirical distribution of two datasets in one dimension, the Multiple Univariate KS Test does this across multiple dimensions simultaneously. However, when multiple hypotheses are tested, the probability of observing a rare event increases, increasing the likelihood of incorrectly rejecting a null hypothesis. According to [Rabanser et al. 2019], a conservative aggregation method can be used to reduce the probability of this type of error, the Bonferroni correction [Bland and Altman 1995], to adjust the significance level for multiple comparisons.

Applying it to the Multiple Univariate KS Test for d distributions, the decision rule is to reject the hypothesis at the significance level α if $\min_{k=1,2,...,d} KS(F_k, G_k) > c(\frac{\alpha}{d})\sqrt{\frac{n+m}{n\times m}}$, where $KS(F_k, G_k)$ is the KS Test for the empirical distribution functions F and G of the k-th dimension, whose respective sample sizes are n and m. Note that Bonferroni correction is applied by testing the hypothesis at a significance level of α/d.

2.2. Distance-based Methods

Distance-based methods offer a more direct measure of the distance or dissimilarity between two probability distributions than statistical methods. Moreover, they do not rely on specific distributional assumptions. They provide a numerical value indicating the degree of difference: the larger the distance, the greater the difference, indicating drift.

2.2.1. Kullback–Leibler Divergence

For two discrete probability distributions P and Q, the Kullback–Leibler (KL) Divergence (or relative entropy) $KL(P||Q)$ from Q to P is defined as [Dasu et al. 2006]:

$$KL(P||Q) = \sum_{x\in\mathcal{X}} P(x)log(\frac{P(x)}{Q(x)}) \tag{2}$$

2.2.2. Jensen–Shannon Divergence

The Jensen–Shannon (JS) Divergence is a method based on KL Divergence but with an improvement: it is symmetric. For two discrete probability distributions P and Q, the Jensen–Shannon Divergence $JS(P||Q)$ is defined as:

$$JS(P||Q) = \frac{1}{2}KL(P||\frac{(P+Q)}{2}) + \frac{1}{2}KL(Q||\frac{(P+Q)}{2}) \tag{3}$$

2.2.3. Hellinger Distance

For two discrete probability distributions $P(p_1, p_2, \ldots, p_n)$ and $Q(q_1, q_2, \ldots, q_n)$, the Hellinger distance $H(P, Q)$ is defined as:

$$H(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{n}(\sqrt{p_i} - \sqrt{q_i})^2} \tag{4}$$

2.3. Drift Detection Methods

Choosing a distance function to measure change is only one aspect of the drift detection problem. Another crucial aspect is determining the statistical significance of the observed change [Dasu et al. 2006]. This involves specifying a null hypothesis (i.e., that no change has occurred) and assessing how likely it is that the observed measurement could occur under this hypothesis. Some drift detection approaches, such as HDDDM, monitor the magnitude of the change in some distance metric between a new distribution and a baseline distribution, which can or cannot be updated every time a drift is detected. This way, these methods can be used to detect changes in data distribution over time, which can help maintain the performance of machine learning models in dynamic environments.

The following sections define some methods that use this approach and are evaluated in this work. The first one, HDDDM, was presented by [Ditzler and Polikar 2011]. The second and third methods, JSDDM and KSDDM, are the new methods proposed in this work. All of these approaches assume an incremental learning setting, where new datasets are presented in batches over time; they are not based on the classifier's performance (i.e., rely only on raw features); and they are classifier free.

It is important to note that, in terms of computational cost, these techniques are comparable. Each method derives empirical distributions from the same input data. The computation of drift is linear with respect to the number of bins, which is dictated by the batch size.

2.3.1. Hellinger Distance Drift Detection Method (HDDDM)

The Hellinger Distance-based Drift Detection Method (HDDDM) proposed by [Ditzler and Polikar 2011] assumes that data arrives in batches, with dataset $\mathcal{D}_t$ becoming available at time t. The following steps summarize the method, assuming as initialization $\lambda = 1$, $\mathcal{D}_\lambda = \mathcal{D}_1$, and for $t = 2, 3, \ldots$:

1. Generate histograms P from $\mathcal{D}_t$ and Q from $\mathcal{D}_\lambda$, each one with $b = \left\lfloor \sqrt{|\mathcal{D}_t|} \right\rfloor$ bins.
2. Calculate the Hellinger distance $\delta_H(t)$ between P and Q and its difference from $\delta_H(t-1)$:

$$\delta_H(t) = \frac{1}{d}\sum_{k=1}^{d}\sqrt{\sum_{i=1}^{b}(\frac{P_{i,k}}{\sum_{j=1}^{b}P_{j,k}} - \frac{Q_{i,k}}{\sum_{j=1}^{b}Q_{j,k}})^2} \quad \epsilon(t) = \delta_H(t) - \delta_H(t-1) \tag{5}$$

where d is the dimensionality of the data, and $P_{i,k}$($Q_{i,k}$) is the frequency count in bin i of the histogram corresponding to P (Q) of feature k.

3. Update the adaptative threshold $\hat{\epsilon}$ and the standard deviation $\hat{\rho}$:

$$\hat{\epsilon} = \frac{1}{t-\lambda-1}\sum_{i=\lambda}^{t-1}|\epsilon(i)| \qquad \hat{\rho} = \sqrt{\frac{\sum_{i=\lambda}^{t-1}(|\epsilon(i)|-\hat{\epsilon})^2}{t-\lambda-1}} \tag{6}$$

4. Compute the actual threshold $\beta(t) = \hat{\epsilon} + \gamma\hat{\rho}$, where γ is some positive constant, indicating how many standard deviations of change around the mean indicate drift.
5. Determine if drift occurred:
   Drift is present if $|\epsilon(t)| > \beta(t)$. In this case, it is necessary reset $\mathcal{D}_\lambda$ by making $\mathcal{D}_\lambda = \mathcal{D}_t$ and $\lambda = t$.
   When drift is not present, $\mathcal{D}_\lambda$ must be expanded to include $\mathcal{D}_t$: $\mathcal{D}_\lambda = \{\mathcal{D}_\lambda, \mathcal{D}_t\}$.

HDDDM signalizes a drift when the magnitude of the change ($|\epsilon(t)|$) is significantly greater than the average of the change since the last detected change ($\hat{\epsilon}$). The significance is controlled by γ and the standard deviation of the divergence differences ($\hat{\rho}$).

2.3.2. Jensen-Shannon Drift Detection Method (JSDDM)

To create a drift detection method based on the JS Divergence, we adapted the HDDDM method by replacing the Hellinger Distance in Equation 5 in Step 2 of the method with the JS Divergence defined in Equation 3, as shown in the following:

$$\delta_{JS}(t) = \frac{1}{d}\sum_{k=1}^{d} JS(P_k||Q_k) \qquad \epsilon(t) = \delta_{JS}(t) - \delta_{JS}(t-1) \tag{7}$$

where d is the data dimensionality, and P_k(Q_k) is the distribution histogram of feature k.

2.3.3. Kolmogorov-Smirnov Drift Detection Method (KSDDM)

In order to detect data drifts using the Kolmogorov-Smirnov (KS) Test, we devised the following simple algorithm, which assumes that data arrives in batches, with dataset $\mathcal{D}_t$ becoming available at time t, $\lambda = 1$, $\mathcal{D}_\lambda = \mathcal{D}_1$, and for $t = 2, 3, \ldots$:

1. Generate the empirical distribution functions F_k from $\mathcal{D}_t$ and G_k from $\mathcal{D}_\lambda$ for each feature k in data.
2. Obtain the minimum of the KS Test of all features' empirical distributions:

$$\delta_{KS}(t) = \min_{k=1,2,\ldots,d}\{KS(F_k, G_k)\} \tag{8}$$

   where F_k (G_k) is the empirical distribution function of feature k in $\mathcal{D}_t$ ($\mathcal{D}_\lambda$).
3. Determine if drift occurred:
   Drift is present at the significance level α if $\delta_{KS}(t)$ is greater than $c(\frac{\alpha}{d})\sqrt{\frac{n+m}{n\times m}}$, where $m = |\mathcal{D}_t|$, $n = |\mathcal{D}_\lambda|$, d is the dimensionality of the data and $c(\frac{\alpha}{d})$ is given in the Kolmogorov table.
   If drift is detected, it is necessary to reset $\mathcal{D}_\lambda$ by making $\mathcal{D}_\lambda = \mathcal{D}_t$ and $\lambda = t$.
   When drift is not present, $\mathcal{D}_\lambda$ must be expanded to include $\mathcal{D}_t$: $\mathcal{D}_\lambda = \{\mathcal{D}_\lambda, \mathcal{D}_t\}$.

3. Datasets Used in the Experiments

In our experiments, we used datasets cited by [Lu et al. 2018], a literature review on learning under concept drift. The objective of using them is to demonstrate how methods for the detection of data drift react in a scenario of concept drift. In particular, we are using two datasets that keep the distributions stable but just drifting in the concept.

3.1. Insects Datasets

The Insects Datasets [Souza et al. 2020] is derived from a real-world streaming application that uses optical sensors to capture data and recognize flying insect species in real time using a Smart Trap. The temperature is the main environmental factor influencing the insect behavior in the trap and, consequently, the data captured. Observations were ordered over time based on temperature patterns, and examples were uniformly sampled within each temperature to isolate temperature-induced drifts. Then, data were split into 11 datasets with 33 features, showcasing different patterns of change (incremental, abrupt, gradual, reoccurring), balanced and unbalanced classes, and class distribution changes.

3.2. SEA Datasets

The Streaming Ensemble Algorithm (SEA) synthetic dataset [Street and Kim 2001] simulates a data stream with abrupt drift. Each observation consists of three features, with only the first two being relevant. The target is binary and positive if the sum of the relevant features exceeds a specific threshold. Concept drift is introduced by switching the threshold (i.e. changing the classification function).

We used an implementation of the SEA dataset provided by River, a library to build online machine-learning models. In the library, there are four different variants to be chosen as threshold configuration: (0) threshold = 8, (1) threshold = 9, (2) threshold = 7, and (3) threshold = 9.5. Using them, we built two different experimental setups:

1. **SEA**: a stream that starts with variant 0, has variant 3 in the middle, then returns to variant 0, simulating a deviation in concept for a period;
2. **MULTISEA**: a scenario that has 12,500 items from each of the four variants.

3.3. STAGGER Datasets

The STAGGER synthetic datasets introduced by [Schlimmer and Granger 1986], like SEA, simulate data with abrupt concept drift. In STAGGER, objects are described by three features – size (small, medium, and large), shape (circle, square, and triangle), and color (red, blue, and green) – and the target is a boolean value given by a function f. The River library provides three variants for f: (0) True if the size is small and the color is red, (1) True if the color is green or the shape is a circle, and (2) True if the size is medium or large. Changing f causes concept drift. Using River, we built two experimental setups:

1. **STAGGER**: a stream that starts with variant 0, has variant 1 in the middle, then returns to variant 0, simulating a deviation in concept for a period;
2. **MULTISTAGGER**: a scenario with 16,666 items from each of the three variants.

3.4. Electricity

The Electricity Pricing dataset from [Harries 1999] is used in our experiment to simulate the concept drifting and class imbalance environment, which originally contains 45, 312 samples drawn from 7 May 1996 to 5 December 1998 with one sample for every half an hour from the electricity market in New South Wales, Australia.

3.5. Magic Gamma Telescope

The Magic Gamma Telescope dataset is Monte Carlo generated to simulate high-energy gamma particle registration in a ground-based atmospheric Cherenkov gamma telescope using the imaging technique. The images captured by the telescope allow the discrimination of primary gammas (signal) from hadronic showers caused by cosmic rays (background) [Bock 2007]. The original dataset has little drift; hence, it has been modified by sorting the feature *fConc1* in ascending order and generating incremental batches with meaningful data drift, as described in [Ditzler and Polikar 2011].

4. Using Drift Detection to Improve ML System's Performance

This study aims to provide an empirical demonstration of how statistical and distance-based methods for detecting data drift in input streams can be used to improve the performance metrics of a machine learning algorithm subjected to concept drifts. We used the datasets in Section 3 to (re)train classifiers and get their performance metrics. The experiments were implemented using the scikit-learn, scipy, river, and Menelaus libraries [1]. The source code of the experiments is available at `https://github.com/lucashelfs/EmpiricalAnalysisOfDataDrift`.

The following sections first present visualizations of the drifts detected and then describe the approach we used for improving classifiers. The datasets were segmented into batches of equal size, specifically 1000, 1500, 2000, and 2500 instances per batch. We chose these batch sizes considering the different sizes of the datasets used. To have a sufficient number of batches to evaluate the drift detection techniques in all datasets, we could not increase the batch sizes too much. And for the larger datasets, having too many batches makes the drift visualizations and interpretation more difficult. The segmentation also aimed to demonstrate the sensitivity of each technique to varying batch sizes.

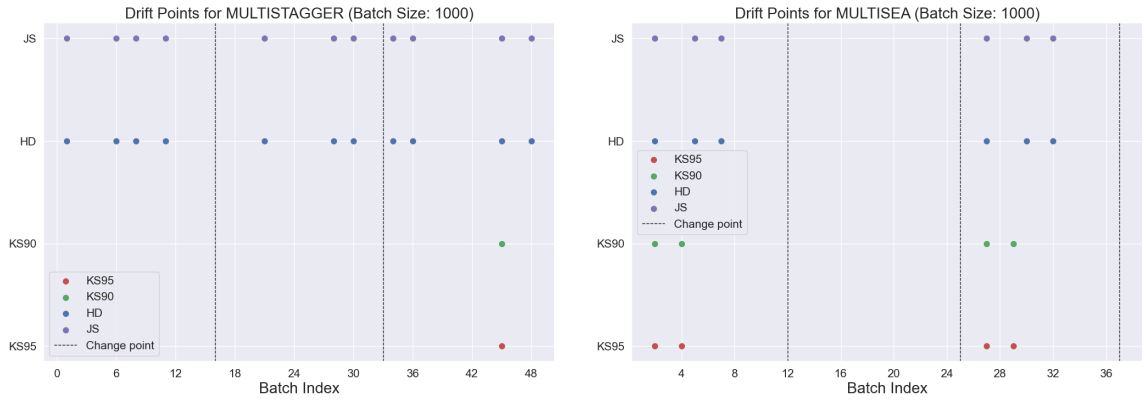
4.1. Visualization of the Detected Data Drifts

Figure 1 shows how the drift detection techniques behaved for the MULTISTAGGER and MULTISEA datasets, while Figure 2 shows the behavior for the Insects' abrupt balanced and imbalanced datasets. In these figures, a point in the graph denotes a detected drift in a given batch; the color differentiates the detection techniques. The vertical dashed lines denote the location of the known concept drifts (annotated in the datasets). It is interesting to point out that even though drifts are being detected, they are not being detected specifically at the datasets' concept-changing points.

To visualize the drifts of all the features individually, heatmap plots can be used, as shown in Figure 3 for the Magic dataset. This type of plotting offers valuable insights into the drift patterns detected from the batches and the reference set. In Figure 3, darker color shades indicate a greater degree of divergence between specific batches and the reference batch. In Figure 3(A), referent to the KSDDM technique, lower p-values indicate greater divergence, consequently leading to darker hues on the heatmap. On the other hand, in Figure 3(B), of the HDDDM technique, larger distances mean greater divergence.

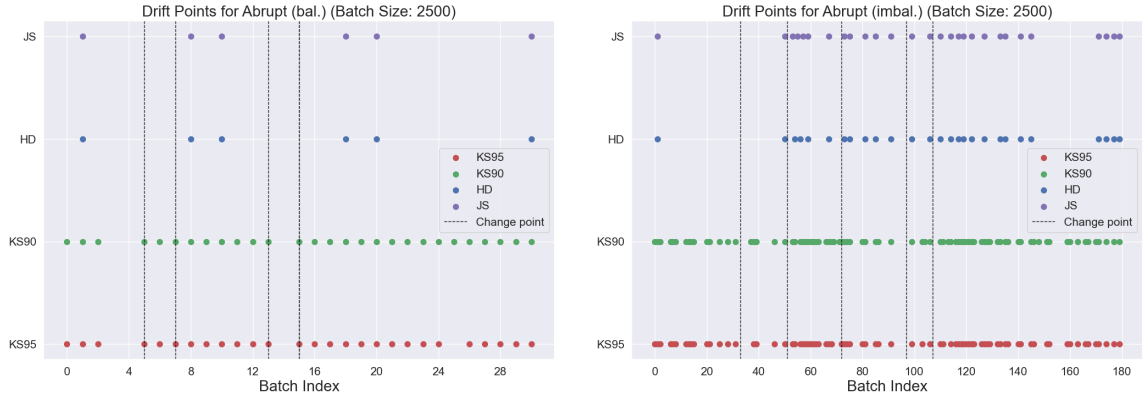
We can see in Figure 3(A) that, for the ordered feature column in the heatmap fourth row, the KS Tests used on KSDDM had p-values closer to zero. For the Helinger

[1] `scikit-learn.org/`, `scipy.org/`, `riverml.xyz/`, `pypi.org/project/menelaus/`



(A) (B)

Figure 1. Detected drifts for the (A) MULTISTAGGER and (B) MULTISEA datasets



(A) (B)

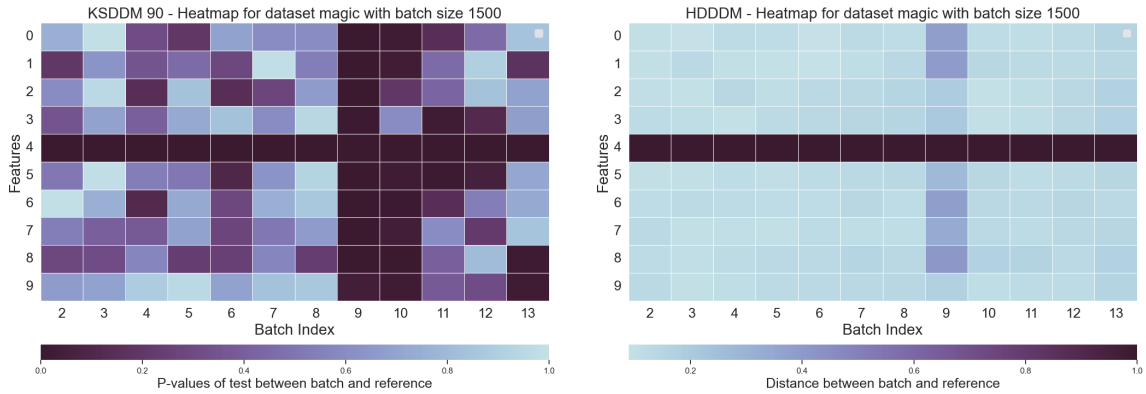
Figure 2. Detected drifts for the Insects' (A) Abrupt balanced and (B) Abrupt imbalanced datasets

Distances used in HDDDM, the pattern was repeated, but with the values being closer to 1, as we can see in Figure 3(B). Moreover, the KS-based technique was much more sensitive to deviations in data, given the overall darker tones that can be observed in Figure 3(A).

4.2. An Approach for Using Detected Drifts to Improve a Classifier

The classifiers were evaluated using a prequential (test-then-train) approach, wherein a classifier is reset upon the detection of a drift by the used technique, an approach similar to the one used by [Souza et al. 2020]. We used the algorithm described in the following for creating a Naive Bayes classifier and evolving it using a drift detection technique; the algorithm also creates a baseline classifier to compare the performances:

1. **Input**: Labeled data batches and a drift detection technique.
2. Use batch 1 to train a Naive Bayes classifier C_B – the baseline model.
3. Use batch 1 to train a Naive Bayes classifier C_D – the model that benefits from drift detection.
4. Set batch 1 as the reference batch.
5. **From batch 2 onwards:**
   (a) Store the predictions of both classifiers C_B and C_D for the current batch.



(A) (B)

Figure 3. Heatmaps for visualizing the drifts of the Magic dataset detected with the techniques (A) Kolmogorov-Smirnov and (B) Hellinger Distance

(b) Check for drift between the reference set and the current batch using the drift detection technique.
(c) Update C_B classifier with the current batch.
(d) **If no drift is detected:**
- Update C_D classifier with the current batch.
- Update the reference set by merging it with the current batch.

(e) **If drift is detected:**
- Set the reference set to the current batch.
- Reset classifier C_D training only with the new reference set.

6. **At the end of all batches:** Compute the performance metrics.

4.3. Experimental Results and Discussion

The algorithm described in Section 4.2 was implemented to compare the following drift detection techniques: Base (no drift detection), KS95 (KSDDM with $\alpha = 5\%$), KS90 (KSDDM with $\alpha = 10\%$), HDDDM (with $\gamma = 1$, as specified in [Ditzler and Polikar 2011]), and JSDDM (with $\gamma = 1$). We evaluated the techniques using the datasets described in Section 3 and measured different model metrics for all classifiers. Given the diverse datasets used in this study, the metrics more suited to assess the performance of the classifiers are the Area Under the Curve (AUC) and the F1 score.

Tables 1-4 show the performance metrics obtained. Values in bold are the best F1 metrics for each dataset, while values in italics are the best AUC. Each dataset has a different size and, hence, a specific number of batches. The column *Drift* has the number of drifts that a technique has detected on that specific dataset scenario.

According to the experimental results, utilizing drift detection techniques to determine the optimal times for retraining classifiers can significantly enhance overall performance.

The use of smaller batches provided the best results in terms of the F1 and AUC metrics. In terms of the number of detected drifts of each technique over the number of batches of each experiment, the KS90 technique triggered more resets to the classifiers than the other techniques, being closely followed by the KS95. The HDDDM and JSDDM

Dataset(*)	Batches	Base			KS95			KS90			HDDDM			JSDDM		
		Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC
MULTISTAGGER	50	0	0.502	0.555	1	0.557	0.587	1	0.557	0.587	11	**0.841**	*0.837*	11	**0.841**	*0.837*
MULTISEA	50	0	0.687	0.651	4	0.691	0.653	4	0.691	0.653	6	**0.692**	*0.655*	6	**0.692**	*0.655*
SEA	50	0	**0.690**	0.659	4	0.690	*0.662*	6	0.690	0.661	4	0.690	0.660	4	0.690	0.660
STAGGER	50	0	0.576	0.529	0	0.576	0.529	1	0.781	0.713	12	**0.864**	*0.836*	12	**0.864**	*0.836*
Electricity	45	0	0.452	0.508	45	**0.559**	*0.556*	45	**0.559**	*0.556*	5	0.481	0.518	7	0.485	0.520
Magic	19	0	0.518	0.505	19	**0.964**	*0.950*	19	**0.964**	*0.950*	3	0.907	0.873	3	0.907	0.873
Inc (Bal)	57	0	0.500	0.487	15	**0.581**	0.454	17	0.581	0.454	1	0.500	*0.487*	1	0.500	*0.487*
Inc (Imbal)	452	0	0.344	*0.552*	47	**0.520**	0.514	68	0.520	0.513	22	0.519	0.521	23	0.519	0.520
Abr (Bal)	79	0	0.458	0.564	43	0.522	0.571	46	**0.527**	0.558	19	0.464	*0.593*	19	0.464	*0.593*
Abr (Imbal)	452	0	0.383	*0.585*	93	**0.501**	0.530	102	0.501	0.528	50	0.493	0.532	51	0.492	0.533
Inc-Grad (Bal)	24	0	0.325	*0.473*	10	**0.546**	0.349	10	**0.546**	0.349	5	0.517	0.354	5	0.517	0.354
Inc-Grad (Imbal)	143	0	0.466	*0.474*	21	0.550	0.440	26	**0.552**	0.437	10	0.548	0.437	11	0.548	0.436
Inc-Abr-Rec (Bal)	52	0	0.498	0.438	19	**0.581**	0.404	21	0.579	0.406	9	0.525	0.442	10	0.512	*0.464*
Inc-Abr-Rec (Imbal)	355	0	0.522	*0.544*	41	0.548	0.513	53	0.547	0.512	19	**0.552**	0.512	19	0.552	0.511
Inc-Rec (Bal)	79	0	0.335	*0.571*	42	0.519	0.527	45	**0.523**	0.520	17	0.481	0.568	19	0.475	0.566
Inc-Rec (Imbal)	452	0	0.419	*0.584*	90	0.500	0.528	103	**0.501**	0.527	52	0.499	0.532	52	0.498	0.532

(*) In the dataset names, the following abbreviations were used: "Abr" – Abrupt, "Grad" – Gradual, "Inc" – Incremental, "Rec" – Reoccurring, "Bal" – Balanced, "Imbal" – Imbalanced.

Table 1. F1 and AUC metrics for a batch size of 1000

Dataset	Batches	Base			KS95			KS90			HDDDM			JSDDM		
		Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC
MULTISTAGGER	33	0	0.502	0.555	0	0.502	0.555	0	0.502	0.555	4	**0.767**	*0.758*	4	**0.767**	*0.758*
MULTISEA	33	0	0.687	0.651	3	0.685	0.651	5	0.686	0.656	6	**0.693**	*0.658*	6	**0.693**	*0.658*
SEA	33	0	**0.691**	0.660	2	0.690	0.659	4	0.689	0.661	7	0.690	*0.662*	7	0.690	*0.662*
STAGGER	33	0	0.574	0.528	0	0.574	0.528	0	0.574	0.528	5	**0.840**	*0.800*	5	**0.840**	*0.800*
Electricity	30	0	0.438	0.508	30	**0.526**	*0.536*	30	**0.526**	*0.536*	4	0.450	0.510	10	0.462	0.512
Magic	12	0	0.517	0.504	12	**0.937**	*0.913*	12	**0.937**	*0.913*	1	0.860	0.814	1	0.860	0.814
Inc (Bal)	38	0	0.497	*0.488*	11	0.574	0.468	12	**0.575**	0.469	9	0.575	0.468	9	0.575	0.468
Inc (Imbal)	301	0	0.345	*0.552*	33	**0.521**	0.517	45	0.518	0.515	16	0.512	0.527	16	0.512	0.527
Abr (Bal)	53	0	0.445	0.570	38	0.464	0.591	39	**0.464**	0.591	13	0.402	*0.608*	13	0.402	*0.608*
Abr (Imbal)	301	0	0.382	*0.585*	89	**0.490**	0.534	103	0.490	0.533	46	0.487	0.544	46	0.482	0.542
Inc-Grad (Bal)	16	0	0.317	*0.476*	8	**0.531**	0.369	8	**0.531**	0.369	3	0.439	0.335	3	0.439	0.335
Inc-Grad (Imbal)	95	0	0.464	*0.475*	18	0.543	0.437	21	0.542	0.437	10	0.545	0.442	11	**0.545**	0.440
Inc-Abr-Rec (Bal)	35	0	0.492	0.440	17	0.550	0.439	21	**0.554**	0.437	6	0.496	*0.466*	6	0.496	*0.466*
Inc-Abr-Rec (Imbal)	236	0	0.521	*0.545*	36	**0.546**	0.512	46	0.543	0.513	16	0.543	0.515	14	0.543	0.513
Inc-Rec (Bal)	53	0	0.321	0.578	36	**0.478**	0.544	39	0.476	0.543	10	0.402	*0.585*	9	0.393	0.585
Inc-Rec (Imbal)	301	0	0.417	*0.585*	83	0.491	0.539	97	**0.491**	0.536	41	0.480	0.543	43	0.481	0.544

Table 2. F1 and AUC metrics for a batch size of 1500

techniques presented similar results between them, triggering way fewer resets for the classifiers than KS techniques.

The KS Test is very sensitive to data shifts, as pointed out in Section 4.1. This excessive detection of drifts by the KS techniques can result in an overfitted classifier, which might have good F1 and AUC measures but with a low generalization for new data. Conversely, the adaptative threshold of HDDDM and JSDDM techniques makes them more sensitive to drifts over time, which is good for datasets with more stable distributions, like MULTISTAGGER. Their detected drifts and the consequent full retraining of the model resulted in F1 and AUC metrics better than those of the KSDDM method in this case. It is also worth noting that the replacement of the Hellinger Distance by the JS Divergence in the HDDDM method, which resulted in the JSDDM framework, did not provide improvements over the HDDDM performance.

Our experimental results demonstrate that the analyzed drift detection techniques are capable of improving the system's robustness, even in scenarios subjected to concept drifts. Even with prequential evaluation, which uses all the data to train the classifier, the approach of resetting the classifier improved the performance in case of drifts.

Dataset	Batches	Base			KS95			KS90			HDDDM			JSDDM		
		Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC
MULTISTAGGER	25	0	0.493	0.549	0	0.493	0.549	0	0.493	0.549	1	**0.550**	*0.582*	1	**0.550**	*0.582*
MULTISEA	25	0	0.687	0.651	1	0.687	0.651	7	**0.693**	*0.658*	1	0.686	0.651	1	0.686	0.651
SEA	25	0	**0.690**	0.659	1	**0.690**	0.659	5	0.689	0.661	1	0.690	*0.661*	1	0.690	*0.661*
STAGGER	25	0	0.573	0.528	0	0.573	0.528	0	0.573	0.528	3	**0.795**	*0.731*	3	**0.795**	*0.731*
Electricity	22	0	0.438	0.507	22	**0.519**	*0.535*	22	**0.519**	*0.535*	6	0.449	0.512	7	0.480	0.520
Magic	9	0	0.517	0.504	9	**0.909**	*0.875*	9	**0.909**	*0.875*	2	0.814	0.760	1	0.708	0.650
Inc (Bal)	28	0	0.496	0.489	12	0.574	0.468	13	**0.575**	0.468	2	0.502	*0.491*	7	0.571	0.472
Inc (Imbal)	226	0	0.344	*0.553*	41	0.515	0.522	53	**0.516**	0.521	18	0.508	0.525	17	0.508	0.529
Abr (Bal)	39	0	**0.435**	0.578	31	0.418	0.642	31	0.418	0.642	14	0.348	*0.643*	14	0.348	*0.643*
Abr (Imbal)	226	0	0.382	*0.585*	84	0.481	0.536	93	**0.484**	0.537	40	0.476	0.547	44	0.475	0.545
Inc-Grad (Bal)	12	0	0.296	*0.484*	8	**0.454**	0.355	8	**0.454**	0.355	3	0.425	0.328	3	0.425	0.328
Inc-Grad (Imbal)	71	0	0.463	*0.476*	19	0.541	0.437	21	**0.546**	0.441	11	0.528	0.446	11	0.528	0.446
Inc-Abr-Rec (Bal)	26	0	**0.486**	0.440	13	0.477	*0.452*	13	0.477	*0.452*	6	0.478	0.449	6	0.478	0.449
Inc-Abr-Rec (Imbal)	177	0	0.521	*0.544*	35	**0.545**	0.512	45	0.543	0.511	15	0.531	0.518	15	0.531	0.518
Inc-Rec (Bal)	39	0	0.312	0.588	33	0.469	0.560	35	**0.471**	0.547	10	0.391	0.606	15	0.412	*0.615*
Inc-Rec (Imbal)	226	0	0.417	*0.585*	85	**0.484**	0.541	89	0.484	0.540	47	0.479	0.545	47	0.479	0.546

Table 3. F1 and AUC metrics for a batch size of 2000

Dataset	Batches	Base			KS95			KS90			HDDDM			JSDDM		
		Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC	Drifts	F1	AUC
MULTISTAGGER	20	0	0.493	0.549	0	0.493	0.549	0	0.493	0.549	2	**0.744**	*0.736*	2	**0.744**	*0.736*
MULTISEA	20	0	0.687	0.651	0	0.687	0.651	8	**0.693**	*0.659*	4	0.690	0.655	4	0.690	0.655
SEA	20	0	**0.690**	0.659	2	0.689	0.661	4	0.689	0.660	3	0.689	*0.661*	3	0.689	*0.661*
STAGGER	20	0	0.574	0.529	0	0.574	0.529	0	0.574	0.529	3	**0.829**	*0.785*	3	**0.829**	*0.785*
Electricity	18	0	0.435	0.506	18	**0.520**	*0.535*	18	**0.520**	*0.535*	7	0.495	0.525	7	0.487	0.521
Magic	7	0	0.514	0.502	7	**0.849**	*0.801*	7	**0.849**	*0.801*	2	**0.849**	*0.801*	1	0.660	0.608
Inc (Bal)	22	0	0.494	*0.486*	13	**0.572**	0.470	13	**0.572**	0.470	0	0.494	*0.486*	0	0.494	*0.486*
Inc (Imbal)	180	0	0.343	*0.553*	41	0.509	0.523	47	**0.509**	0.520	19	0.498	0.534	19	0.498	0.534
Abr (Bal)	31	0	**0.424**	0.587	27	0.374	*0.646*	28	0.373	0.645	6	0.288	0.627	6	0.288	0.627
Abr (Imbal)	180	0	0.380	*0.585*	79	**0.474**	0.545	85	0.473	0.543	27	0.449	0.538	28	0.449	0.538
Inc-Grad (Bal)	9	0	0.286	*0.489*	6	**0.463**	0.347	6	**0.463**	0.347	3	0.442	0.410	3	0.442	0.410
Inc-Grad (Imbal)	57	0	0.461	*0.477*	15	0.540	0.441	18	**0.545**	0.434	8	0.529	0.463	8	0.529	0.463
Inc-Abr-Rec (Bal)	21	0	**0.483**	0.439	14	0.442	*0.471*	14	0.442	*0.471*	7	0.451	0.452	7	0.451	0.452
Inc-Abr-Rec (Imbal)	142	0	0.520	*0.545*	31	0.533	0.522	32	**0.533**	0.522	15	0.514	0.525	16	0.516	0.530
Inc-Rec (Bal)	31	0	0.309	*0.596*	26	**0.429**	0.594	26	**0.429**	0.594	6	0.375	0.580	6	0.375	0.580
Inc-Rec (Imbal)	180	0	0.415	*0.585*	79	0.471	0.546	85	**0.474**	0.546	36	0.467	0.552	36	0.467	0.552

Table 4. F1 and AUC metrics for a batch size of 2500

5. Related Work

Several works approached the use of statistical and distance-based methods to detect drifts. For example, [Dasu et al. 2006] presented a method based on the KL Divergence that uses a bootstrapping approach to establish the statistical significance of the distances with demonstrated statistical guarantees. An empirical evaluation (on real and synthetic data) was performed to show the accuracy of the approach. [Pérez-Cruz 2008] proposed a method for estimating the KL Divergence between continuous densities. They showed that the divergence can be either estimated using the empirical cdf or k-nearest-neighbors density estimation. [Rabanser et al. 2019] investigated shift detection through the lens of statistical two-sample testing. In particular, they presented an empirical investigation on image datasets of how dimensionality reduction and two-sample testing might be combined in a practical pipeline for detecting distribution shifts in real-life ML systems.

[Souza et al. 2020] discussed the challenges faced by the stream learning community concerning the reduced number of real-world data and the lack of a benchmark to evaluate adaptive classifiers and drift detectors. To mitigate these issues, the authors created 11 new datasets based on data collected by an optical sensor that measures the flying behavior of insects and evaluated several learning techniques on these datasets. They also provided a new public repository with datasets from real problems.

6. Conclusion

This work presented empirical results on using statistical and distance-based methods to detect drift in the data inputs of classifiers and evolve their models based on that. We have presented a variation (JSDDM) of a consolidated method (HDDDM) for drift detection and proposed a new one, based on the Kolmogorov-Smirnov Test (KSDDM).

In the experiments, we used datasets from various fields and simulated concept drift scenarios. While the detection methods did not immediately signal specific concept drifts, they effectively identified data drifts that prompted necessary classifier resets. The experimental results showed that having a concern about data drifts is good for increasing the robustness of an ML system.

The effectiveness of each drift detection method can vary significantly across different datasets and their characteristics. We advise having a good understanding of the datasets before comparing the methods' results. On this matter, working with synthetic datasets provides more control over the data and concept changes, which facilitates performing comparison experiments and analyzing the results.

One of the controlled variables of our experiments, the batch size, showed that the best results for the datasets used in the work were obtained with the smallest batch sizes analyzed. This is an important finding since, in real-world streaming scenarios, working with large batches can be computationally expensive and even prohibitive.

Acknowledgments

This research was funded by grants CNPq proc. 420623/2023-0 and #2023/00779-0, São Paulo Research Foundation (FAPESP). It is also part of the INCT of the Future Internet for Smart Cities funded by CNPq proc. 465446/2014-0, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, FAPESP proc. 14/50937-1, and FAPESP proc. 15/24485-9.

References

[Bland and Altman 1995] Bland, J. M. and Altman, D. G. (1995). Multiple significance tests: the Bonferroni method. *Bmj*, 310(6973):170.

[Bock 2007] Bock, R. (2007). MAGIC Gamma Telescope. https://doi.org/10.24432/C52C8B.

[Dasu et al. 2006] Dasu, T., Krishnan, S., Venkatasubramanian, S., and Yi, K. (2006). An information-theoretic approach to detecting changes in multi-dimensional data streams. In *Symposium on the Interface of Statistics, Computing Science, and Applications (Interface)*.

[Ditzler and Polikar 2011] Ditzler, G. and Polikar, R. (2011). Hellinger distance based drift detection for nonstationary environments. In *2011 IEEE symposium on computational intelligence in dynamic and uncertain environments (CIDUE)*, pages 41–48.

[Gama and Castillo 2006] Gama, J. and Castillo, G. (2006). Learning with local drift detection. In *Advanced Data Mining and Applications: Second International Conference, ADMA 2006, Xi'an, China, August 14-16, 2006 Proceedings 2*, pages 42–55.

[Harries 1999] Harries, M. (1999). Splice-2 comparative evaluation: electricity pricing. Technical report, The University of New South Wales, Sydney.

[Hodges Jr 1958] Hodges Jr, J. (1958). The significance probability of the Smirnov two-sample test. *Arkiv för matematik*, 3(5):469–486.

[Lu et al. 2018] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. (2018). Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363.

[Pérez-Cruz 2008] Pérez-Cruz, F. (2008). Kullback-Leibler divergence estimation of continuous distributions. In *2008 IEEE international symposium on information theory*, pages 1666–1670.

[Rabanser et al. 2019] Rabanser, S., Günnemann, S., and Lipton, Z. (2019). Failing loudly: An empirical study of methods for detecting dataset shift. *Advances in Neural Information Processing Systems*, 32.

[Schlimmer and Granger 1986] Schlimmer, J. C. and Granger, R. H. (1986). Incremental learning from noisy data. *Machine learning*, 1:317–354.

[Sculley et al. 2015] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J. F., and Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 2015-Janua:2503–2511.

[Souza et al. 2020] Souza, V. M. A., Reis, D. M., Maletzke, A. G., and Batista, G. E. A. P. A. (2020). Challenges in benchmarking stream learning algorithms with real-world data. *Data Mining and Knowledge Discovery*, 34:1805–1858.

[Street and Kim 2001] Street, W. N. and Kim, Y. (2001). A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 377–382.

[Webb et al. 2016] Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., and Petitjean, F. (2016). Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994.