machine learning & knowledge extraction

Article

Comparison of Off-the-Shelf Methods and a Hotelling Multidimensional Approximation for Data Drift Detection

J. Ramón Navarro-Cerdán *, Vicent Ortiz Castelló and David Millán Escrivá

Instituto Tecnológico de Informática, Universitat Politècnica de València, 46022 Valencia, Spain; vortiz@iti.es (V.O.C.); dmillan@iti.es (D.M.E.)
* Correspondence: jonacer@iti.es

Abstract: Data drift can significantly impact the outcome of a model. Early detection of data drift is crucial for ensuring user confidence in predictions. It allows the user to check if a particular model needs retraining using updated data to adapt to the evolving process dynamics. This study compares five different statistical tests, namely four unidimensional and a new multidimensional test (MSPC), to identify data drift in both mean and deviation. While some are designed to detect drift in mean only, like our multidimensional proposal, others respond to changes in both mean and deviation. However, our Hotelling multidimensional method can be trained once and then applied in a single stage to any data stream with several attributes, and it can identify the most relevant variables causing a data drift with one execution, thus avoiding the need for a single univariate test for each attribute. Moreover, our method yields the relative importance of each attribute for drift and allows users to increase or decrease the relative weight of each variable regarding drift detection. It also may be capable of detecting drift due to changes in multivariate interactions. This behavior is especially suitable for real-world scenarios, such as industry, finance, or healthcare environments.

Keywords: data drift; Hotelling distance; statistical tests; anomaly detection; machine learning

Academic Editor: Andreas Holzinger

Received: 12 November 2024
Revised: 9 December 2024
Accepted: 20 December 2024
Published: 30 December 2024

Citation: Navarro-Cerdán, J.R.; Ortiz Castelló, V.; Millán Escrivá, D. Comparison of Off-the-Shelf Methods and a Hotelling Multidimensional Approximation for Data Drift Detection. *Mach. Learn. Knowl. Extr.* **2025**, *7*, 2. https://doi.org/10.3390/make7010002

1. Introduction

In the rapidly evolving landscape of machine learning and data-driven decision-making, the reliability and accuracy of predictive models are key. However, one significant challenge that often undermines the performance of these models is data drift—a phenomenon where the statistical properties of the data change over time, rendering previously trained models less effective. Dynamic changes can fatally degrade the usefulness of learned models as new distant unobserved patterns emerge in production [1,2]. These changes can appear due to seasonality or periodicity effects, faults derived from sensors or actuators, changes in the behavior of users, or aging consequences, among others [3], and appear suddenly, gradually, incrementally, or recurrently [4,5].

Thus, detecting and addressing data drift is crucial for ensuring that models continue making reliable predictions in dynamic environments. It is critical to know the serviceability of a model over a period including current data, which is directly related to the confidence in the predictions [6]. Drift can manifest as changes in the distribution of input features (data drift) and the relationship between input and output (concept drift) [7,8].

In particular, data drift is a phenomenon that occurs when the statistical distribution of input data changes over time compared to the distribution observed in the machine learning training step. Detecting it on time in a downstream task helps the data scientist decide the best moment to retrain the model so it can adapt to the new behavior within the current context. Moreover, data drift can affect confidence in the model output, and its detection can allow an estimation of the aforementioned confidence. This degradation is usually observed as an apparent increase in the residuals produced during the prediction step.

As real-world applications span across industries such as finance, healthcare, retail, and manufacturing, the ability to monitor and adapt to changes in the underlying data distributions becomes increasingly important [9]. Without timely detection of data drift, models risk providing outdated or biased results, potentially leading to poor decisions and costly consequences. Organizations can maintain model accuracy, improve decision-making, and ultimately drive better business outcomes in an ever-changing world by integrating effective drift detection mechanisms into operational workflows.

This paper compares our multivariate method, called the multivariate statistical process control (MSPC for short), with four different off-the-shelf, well-established algorithms for univariate drift detection via the Frouros implementation [10]: the Kolmogorov–Smirnov (KS) test [11], the Cramér–von Mises (CVM) test [12], Welch's *t*-test (WT) [13], and the Mann–Whitney U (MW-U) test [14]. All of these methods are based on hypothesis testing, evaluating the probability of a sample belonging to a specific population [15,16]. For a particular test, rejecting the null hypothesis would imply the detection of significant data drift.

However, as can be found below, our multidimensional approach can cope simultaneously with outliers for all the attributes as they start to appear over time. In contrast, the unidimensional tests would have to be repeated once per attribute and cannot take advantage of the overall analysis performed with a multidimensional test. Additionally, our method assigns a level of importance to each variable depending on its contribution to the multidimensional drift present in the data. Hence, humans can easily and quickly understand where the drifting problem is in the process and then react accordingly.

2. Materials and Methods

In this section, our controlled dataset and the data drift detection methods used, including MSPC, are presented. A final subsection on the computational cost of MSPC has been also included.

2.1. Dataset

It was crucial to have absolute control over the data analyzed for this comparison, as it is a means of comparing and validating our method. Instead, a real-world dataset could have led to a significant loss of sureness about the presence of a significant drift, its relative magnitude, or its time of appearance.

Therefore, we synthetically generated the dataset used in the experiments, with 10k samples composed of two informative features for a model (`f0` and `f1`), both with a mean of 0 and standard deviations of 1 and 5, respectively (scale). We randomly split the dataset into train (7k) and test (3k) subsets.

Later, we introduced varying levels of additive Gaussian noise to the test data regarding average and typical deviation values. This approach allows controlled variations in noise type and strength as a proportion of the original standard deviation of the modified signal, effectively modeling different types and levels of data drift and seeing if the data drift detection methods can detect it.

The generated dataset can well reflect real-world scenarios in the case of sources of abundant amounts of data sampled from a characterized distribution, the Gaussian being the most common case. However, the MSPC algorithm may also be applied to real-world datasets once carefully validated, which is done in this paper.

2.2. Test-Based Numerical Methods for Data Drift Detection

Most of the algorithms for data drift detection are conceived to be used in batch, whilst others are only designed to be applied in streams. Batch methods are, in turn, divided into distance-based approaches and hypothesis tests. This study analyzes a representative sample of the latter using their Frouros implementation [10].

2.2.1. Off-the-Shelf Methods

This section presents four existing methods for data drift detection over data batches. These algorithms are representative of the hypothesis testing-based methods available in tools such as River [17], Menelaus [18], and Frouros [10], the latter being the most complete and recent of them. Moreover, these methods are commonly used in control charts within the industry.

Kolmogorov–Smirnov (KS) Test

This test originally compares the theoretical cumulative distribution function (CDF) of a population with the empirical one from a sample (called ECDF) to assess the likelihood that the sample belongs to the original population [11]. However, it can be applied to two ECDFs as well. The test analyzes the maximum vertical distance between the two. Formally, let A and B be two data samples. Then, the KS statistic, D_n, is:

$$D_n = max_x(|F_a(x) - F_b(x)|) \tag{1}$$

where $F_a(x)$ and $F_b(x)$ are the ECDF functions, assuming that they are independent and identically distributed observations. It can be seen that the KS test is based on the worst ECDF difference along the x-axis, allowing it to detect data deviation of any kind, but with high sensitivity, even to slight changes if the samples are relatively small.

Cramér–von Mises (CVM) Test

This is an alternative to the KS test and is also used as a criterion to judge the goodness of fit of two samples [12]. CVM also compares the CDFs of two samples (ECDFs in our case), but its statistic is as follows:

$$w^2 = \int_{-\infty}^{\infty} (F_a(x) - F_b(x))^2 dF(x) \tag{2}$$

In this case, it is straightforward to see that the test is based on the energy of the ECDF difference signal along the x-axis, thus conveying a more robust detection method for discrepancies of any kind.

Welch's t-Test (WT)

This test is a modification of the original Student's original t-test to assess if two samples are derived from two populations with equal means [13]. Compared to the Student's test, WT is more reliable when the two samples have unequal variances or sample sizes. While it still assumes normality in data, unlike Student's, it can deal with distributions with different variances. The WT statistic is as follows:

$$t = \frac{\bar{x}_a - \bar{x}_b}{\sqrt{s_{\bar{x}_a}^2 + s_{\bar{x}_b}^2}} \tag{3}$$

where

$$s_{\bar{x}_i} = \frac{s_i}{\sqrt{n_i}} \tag{4}$$

and n_i, $\bar{x}_i$, and s_i are the number of examples, the mean and the standard deviation, respectively, where sub-indices a and b refer to two data samples. As defined, WT aims to detect differences in the mean rather than other types of deviation.

Mann–Whitney U (MW-U) Test

This non-parametric test is applied to a pair of independent samples for ordinal or continuous variables and examines distributional differences in position between the two samples [14]. The U statistic is as follows:

$$U = \min(U_a, U_b) \tag{5}$$

$$U \sim \mathcal{N}\left(\mu = \frac{n_a n_b}{2}, \sigma = \sqrt{\frac{n_a n_b (n_a + n_b + 1)}{12}}\right) \tag{6}$$

where

$$\begin{aligned} U_a &= n_a n_b + \frac{n_a(n_a+1)}{2} - R_a \\ U_b &= n_a n_b + \frac{n_b(n_b+1)}{2} - R_b \end{aligned} \tag{7}$$

being n_i the number of examples of the corresponding samples a and b, and R_i the sum of ranges from the ordered concatenated list that includes all data.

2.2.2. Our Proposed Multivariate Algorithm (MSPC)

In this section, we look for an alternative statistical technique to find data drift in a multidimensional way, considering all the individual numerical variable values simultaneously in a single statistical test. Mahalanobis distance [19] is a metric used to find the distance within a distribution of points. Figure 1 shows the graphical idea of this multivariate test through an ellipse-shaped frontier for anomaly detection.

The relative distance between the multidimensional point and the corresponding multidimensional centroid of the distribution can be measured, as shown in Equation (8):

$$d_M = (\mathbf{x} - \mathbf{m})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{m}) \tag{8}$$

where $\mathbf{x}$ is a multidimensional vector that denotes the point, $\mathbf{\Sigma}^{-1}$ is the inverse covariance matrix, and $\mathbf{m}$ is a multidimensional vector representing the coordinates of the distribution centroid.

Furthermore, when we are interested in specifically influencing the sensitivity to certain features of data drift, it is possible to tweak this behavior through a diagonal matrix $\mathbf{W}$ that weighs each of these sensitivities, as shown in Equation (9):

$$d_{M_w} = (\mathbf{x} - \mathbf{m})^T \mathbf{W} \mathbf{\Sigma}^{-1} \mathbf{W} (\mathbf{x} - \mathbf{m}) \tag{9}$$

where $\mathbf{W}$ is a diagonal matrix with the corresponding relevant feature weights or diagonal ones in case of non-informed relevance. The greater the weight, the greater the feature relevance for data drift detection. To accomplish this using a 2-norm, the sum of squared relevance values must equal the number of variables in the multidimensional test. Corresponding corrections can be introduced when there is deep knowledge of the underlying process, similar to how feature relevance is applied in a prediction model, or they could be estimated by using a set of supervised anomaly data to detect the majority of anomalies [20,21], among other methods.

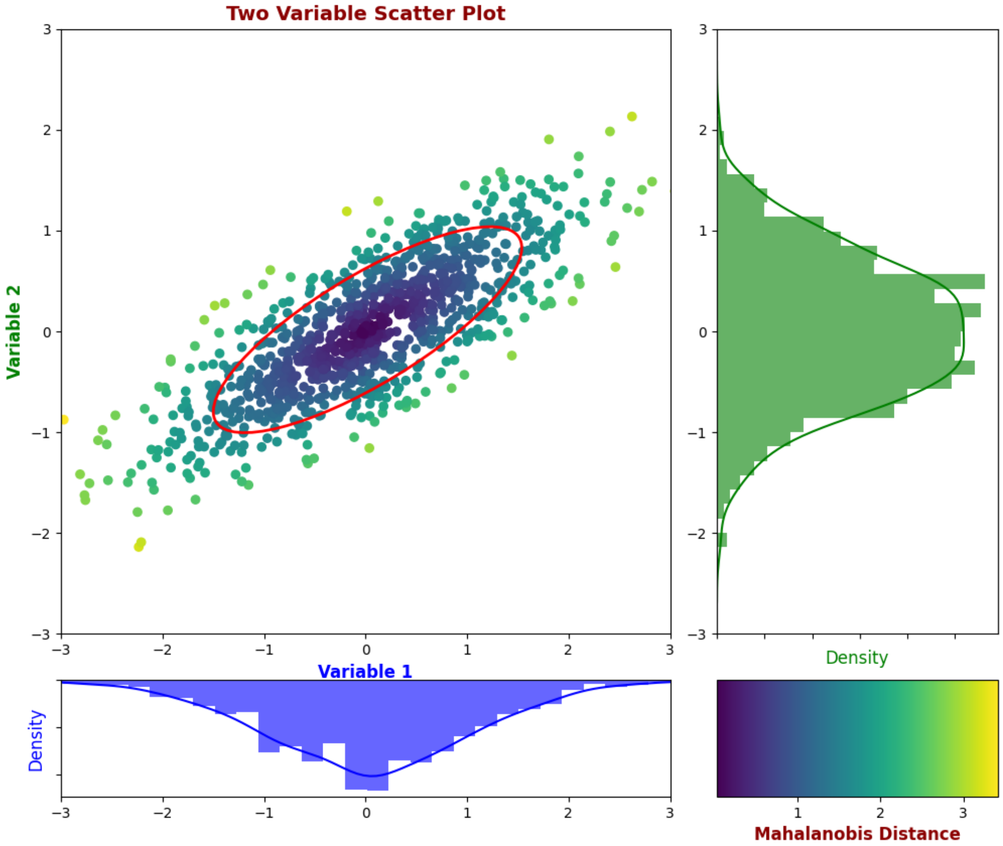

Figure 1. Two-variable scatter plot showing the relation between both variables and the threshold of Mahalanobis distance through the corresponding ellipsoid.

Understanding the phenomenon of data drift as an outlier detection for a set of points, unlike a classical detection of outlier points, implies transforming the equation as shown in (10), by considering $\bar{\mathbf{x}}$ (sample mean) instead of $\mathbf{x}$ (individual feature values).

$$d_{M_w} = (\bar{\mathbf{x}} - \mathbf{m})^T \mathbf{W} \hat{\mathbf{\Sigma}}^{-\mathbf{1}} \mathbf{W} (\bar{\mathbf{x}} - \mathbf{m}) \tag{10}$$

$$\hat{\mathbf{\Sigma}} = \frac{(n_s - 1)\mathbf{\Sigma_s} + (n_t - 1)\mathbf{\Sigma_t}}{n_s + n_t - 2} \tag{11}$$

Let n_s, $\mathbf{\Sigma_s}$, n_t, and $\mathbf{\Sigma_t}$ be the number of examples and covariance matrix used to compute the corresponding training (source, s) and for production (test, t), respectively. All these elements allow us to compute the Hotelling statistic to compare populations, training vs production, as shown in Equation (12):

$$t^2 = \frac{n_t n_s}{n_t + n_s} (\bar{\mathbf{x}} - \mathbf{m})^T \mathbf{W} \hat{\mathbf{\Sigma}}^{-1} \mathbf{W} (\bar{\mathbf{x}} - \mathbf{m}) \tag{12}$$

Next, it is possible to establish a statistical threshold through the corresponding Fisher statistical distribution by using the relation shown in Equation (13):

$$\frac{n_t + n_s - p - 1}{(n_t + n_s - 2)p} t^2 \sim \mathbf{F_{p, n_t + n_s - p - 1}} \tag{13}$$

where p represents the number of features that this test considers simultaneously. Moreover, knowing independently the relevance of each feature can help us identify the primary drift problem sources by applying the Equation (14):

$$r_i = \frac{100 \cdot t_i}{\sum_i t_i} \tag{14}$$

where r_i is the percentage of relevance for the variable i in the currently analyzed data and t_i could be computed as shown in Equation (15):

$$t_i = \left| (\bar{\mathbf{x}} - \mathbf{m})^T \mathbf{W} \hat{\mathbf{\Sigma}}^{-1} \right|_i \tag{15}$$

Finally, this kind of measure assumes a certain degree of data normality. To facilitate this, previous (though not mandatory) univariant Gaussian data transformations, based on relations between percentiles, could be applied, such as the one shown in Algorithm 1, where a previous Gaussian model transformation is trained for the non-Gaussian variables. In this case, the common figure between distributions is the percentile that allows the corresponding value to be estimated from one distribution and vice-versa. Additionally, Figure 2 shows a graphical schema and Algorithm 2 includes the pseudocode to implement the corresponding training and production procedure to perform the multidimensional data drift computation in a general way.

Algorithm 1 Gaussian transformation.

```
function VALUETOPERCENTILE($\mathbf{x_i}$)
    percentile = $\mathcal{P}(x_i)$ ▷ $\mathcal{P}$: Function which relates variable values with its corresponding percentile.
    Return $\mathcal{P}$
end function

function GAUSSIANTRANSFORMATION($\mathcal{P}$, $\mathbf{x_i}$)
    $\mathbf{p} \leftarrow \mathcal{P}(\mathbf{x_i})$        ▷ Model that relates variable values with percentile
    $\mathbf{x_{ig}} \leftarrow \mathcal{N}.ppf(\mathbf{p})$    ▷ Asking for the normal value that belongs to the corresponding percentile
    Return $x_g$
end function

procedure MAIN        ▷ Transforming an X variable from a free distributed variable to Gaussian distribution
    $\mathcal{P} \leftarrow ValueToPercentile(\mathbf{x_i})$    ▷ Function that returns the percentile for a value from the free distribution.
    $\mathbf{X_g} \leftarrow \emptyset$
    for $\mathbf{x_i} \in \mathbf{X}$ do
        $\mathbf{x_{ig}} \leftarrow GaussianTransformation(\mathcal{P}, \mathbf{x_i})$
        $\mathbf{X_g} \leftarrow append(\mathbf{X_g}, \mathbf{x_{ig}})$
    end for
end procedure
```

Optionally, in the case of applying a Gaussian transformation and attempting to favor anomaly detection, during the production phase, only values located between the minimum and maximum univariate training values are transformed, thus not applying such transformation for the original tail values. Further, a `log` transformation could be another classical direct transformation for positively skewed distributions in greater than zero variables.

Algorithm 2 MSPC draft algorithm.

```
$\mathbf{X_s}$: Matrix with source
$n_s$: individuals,
$\mathbf{w}$: feature weight vector
$\mathbf{p}$: features.
function TRAINING ($\mathbf{X_s}$,$\mathbf{w}$)
    $\mathbf{X_{sg}} \leftarrow \emptyset$
    while $\mathbf{x_{si}} \in cols(\mathbf{X_s})$ do
        $\mathcal{P} \leftarrow ValueToPercentile(\mathbf{x_{si}})$
        $\mathbf{x_{sig}} \leftarrow GaussianTransformation(\mathcal{P}, \mathbf{x_{si}})$
        $\mathbf{X_{sg}} \leftarrow append_{cols}(\mathbf{X_{sg}}, x_{sig})$
    end while
    $\mathbf{\Sigma_{sg}} \leftarrow covariance(\mathbf{X_{sg}})$
    $\mathbf{W} \leftarrow matrix_{eye}(\mathbf{w})$
    $\boldsymbol{\mu}_{sg} \leftarrow mean_{cols}(\mathbf{X_{sg}})$
    Return $\mathcal{P}$, $\mathbf{\Sigma_{sg}}$, $\boldsymbol{\mu}_{sg}$, $p$, $n_s$, $\mathbf{W}$
end function

$\mathbf{X_t}$: Matrix with test
$\mathbf{n_t}$: individuals
$\mathbf{p}$: features
function PRODUCTION ($\mathbf{X_t}$, $\mathcal{P}$, $\mathbf{\Sigma_{sg}}$, $\boldsymbol{\mu}_{sg}$, $\mathbf{W}$, $p$, $n_s$)
    $\mathbf{X_{tg}} \leftarrow \emptyset$
    while $x_{ti} \in cols(\mathbf{X_t})$ do
        $x_{tig} \leftarrow GaussianTransformation(\mathcal{P}, x_{ti})$
        $\mathbf{X_{tg}} \leftarrow append_{cols}(\mathbf{X_{tg}}, x_{tig})$
    end while
    $\mathbf{\Sigma_{tg}} \leftarrow covariance(\mathbf{X_{tg}})$
    $\boldsymbol{\mu}_{tg} \leftarrow mean_{cols}(\mathbf{X_{tg}})$
    $\hat{\mathbf{\Sigma}} \leftarrow \frac{(n_s - 1)\mathbf{\Sigma_{sg}} + (n_t - 1)\mathbf{\Sigma_{tg}}}{n_s + n_t - 2}$
    $t^2 \leftarrow \frac{n_t n_s}{n_t + n_s}(\boldsymbol{\mu}_{tg} - \boldsymbol{\mu}_{sg})^T \mathbf{W} \hat{\mathbf{\Sigma}}^{-1} \mathbf{W} (\boldsymbol{\mu}_{tg} - \boldsymbol{\mu}_{sg})$
    Return $t^2$
end function

function FEATUREIMPORTANCE ($\mathbf{X}, \boldsymbol{\mu}, \mathbf{W}, \hat{\mathbf{\Sigma}}$)
    $\mathbf{r} \leftarrow \emptyset$
    $\bar{\mathbf{x}} \leftarrow mean_{cols}(\mathbf{X})$
    $\mathbf{t} = \left|(\bar{\mathbf{x}} - \boldsymbol{\mu})^T \mathbf{W} \hat{\mathbf{\Sigma}}^{-1}\right|$
    for $t_i \in \mathbf{t}$ do
        $r_i \leftarrow \frac{100 \cdot t_i}{\sum_i t_i}$
        $\mathbf{r} \leftarrow append(\mathbf{r}, r_i)$
    end for
    Return $\mathbf{r}$
end function

function STATISTIC($n_t$,$n_s$,$p$,$t^2$)
    $pValue \leftarrow \frac{n_t + n_s - p - 1}{(n_t + n_s - 2)p} t^2 \sim \mathbf{F_{p, n_t + n_s - p - 1}}$
    Return pValue
end function
```

Algorithm 2 *Cont.*

```
procedure MAIN
    $\mathbf{X_s}$: Matrix with source
    $\mathbf{W}$: Feature weight vector
    $n_t$: Number of individuals
    $p$: Number of features
    $\mathcal{P}, \mathbf{\Sigma_{sg}}, \boldsymbol{\mu_{sg}}, p, n_s, \mathbf{W} \leftarrow Training(\mathbf{X_s}, \mathbf{w})$
    while $\mathbf{X_t} \in batchData$ do
        $t^2 \leftarrow Production(\mathbf{X_t}, \mathcal{P}, \mathbf{\Sigma_{sg}}, \boldsymbol{\mu_{sg}}, \mathbf{W}, p, n_s)$
        $fImp \leftarrow FeatImportance(\mathbf{X_t}, \boldsymbol{\mu_{sg}}, \mathbf{W}, \hat{\mathbf{\Sigma}}_{\mathbf{sg}})$
        $pValue \leftarrow Statistic(n_t, n_s, p, t^2)$
        $send(t^2, pValue, \mathbf{featImportances})$
    end while
end procedure
```

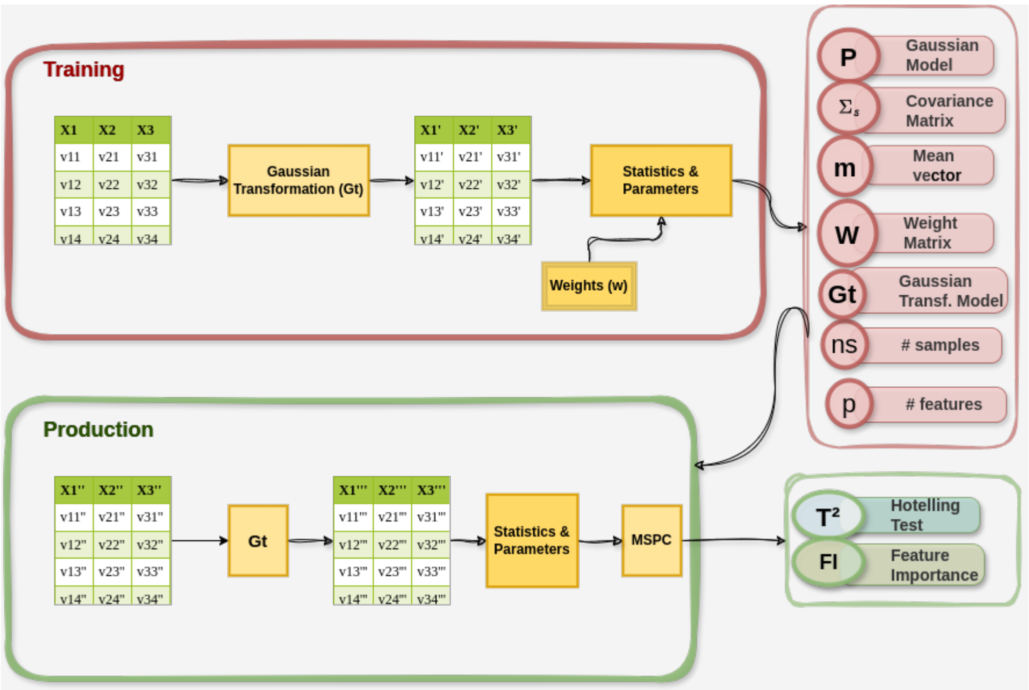

Figure 2. Proposed training and production data drift MSPC graphical schema.

2.2.3. MSPC Computational Cost

High-dimensional covariance inversion is a computational bottleneck for the MSPC method whose classical Gauss–Jordan inversion time complexity is $O(n^3)$, where n is the number of variables considered for the multivariate test. Some approximations based on Cholesky decomposition, feature projection, linear projection, and distributed algorithms can alleviate this problem [22,23]. Nonetheless, in data drift detection for model predictors or classifiers, the dimensionality problem of explanatory variables is not usually a computational problem in most cases.

3. Experiments

We are interested in assessing the suitability and power of the presented off-the-shelf methods to detect data drift in the synthetically generated dataset and comparing them with our proposed test method based on Hotelling distances (MSPC). For this reason, different typologies and levels of Gaussian additive noise were inserted into each of the input attributes (`f0` or `f1`) in terms of both mean and standard deviation of the noise added, as described in Section 2. In addition to these two main scenarios, a third one is considered in which we simultaneously apply the noise to both. Expressly, we set a bi-dimensional grid for all possible combinations of mean and standard deviation sizes as follows:

$$\begin{aligned} mean_{size} &= [0, 0.02, 0.04, 0.06, 0.08, 0.1] \\ stdev_{size} &= [0, 0.1, 0.2, 0.3, 0.5, 0.7, 1] \end{aligned} \tag{16}$$

Each value has to be interpreted as a proportion of the original variable standard deviation. For instance, for parameters mean = 0.06 and stdev = 0.1, we add additive Gaussian noise of mean equal to 6% and standard deviation of 10% of the original standard deviation of this attribute. This grid yields 42 different combinations of both variables to analyze if each of the methods considered can detect such drift in the modified input attributes through a significant p-value. Therefore, for a particular standard deviation value of the range presented, each possible value for the mean is set, allowing us to explore all the grid possibilities orderly.

The maximum values of additive drift considered within the study are 10% in the mean, as all methods can detect it, but we go for higher relative values of increase in the standard deviation, up to 100%, to enable all algorithms to show their capabilities concerning detection of this kind of drift. Anyhow, the suitability of this selection can be checked in the numerical results obtained. These various levels and typologies of additive noise can represent many of the situations that can be found in a real-world scenario.

The p-value of a hypothesis test represents the probability of observing a particular sample, or a more extreme one, under the assumption that the null hypothesis is correct, that is, the sample belongs to the original population. For this reason, when the p-value descends below a typical value (e.g., 0.05), one can be confident of a significant change, in this case, a data drift, as it would be valid 95 out of 100 times.

For each scenario, the four off-the-shelf methods return a single p-value per attribute (`f0` and `f1`). Conversely, our MSPC test is multivariate, so it yields a single p-value for all the input variables as a whole, but it also provides an estimated importance of each variable as a percentage, adding up to 100. It is worth noting that an equal data drift sensitivity for both variables in the MSPC test was considered through the identity matrix ($\mathbf{W} = \mathbf{I}$).

In Tables 1–3, we present univariate p-values for the four off-the-shelf methods apart from a multivariate p-value along with the corresponding importance (Imp-`f0` (%), Imp-`f1` (%)) for our MSPC test-based method. Also, to visually assess the drift of the modified input attribute, in addition to the train vs test baseline for each attribute shown in Figures 3 and 4, two histograms—the original one and the resulting one with a particular level of drift inclusion—are plotted overlapped for each scenario in Figures 5 and 6. Therefore, each of these histograms corresponds to one row in Tables 1–3. To illustrate a minimal but positive case, we show the first significant drift scenario for additive noise of a null standard deviation factor corresponding to a mean factor of 0.08. Finally, the p-values are represented in 3D surface plots in Figures 7–17 for each algorithm and attribute in which drift was introduced (`f0`, `f1`, and the combination of both for MSPC).

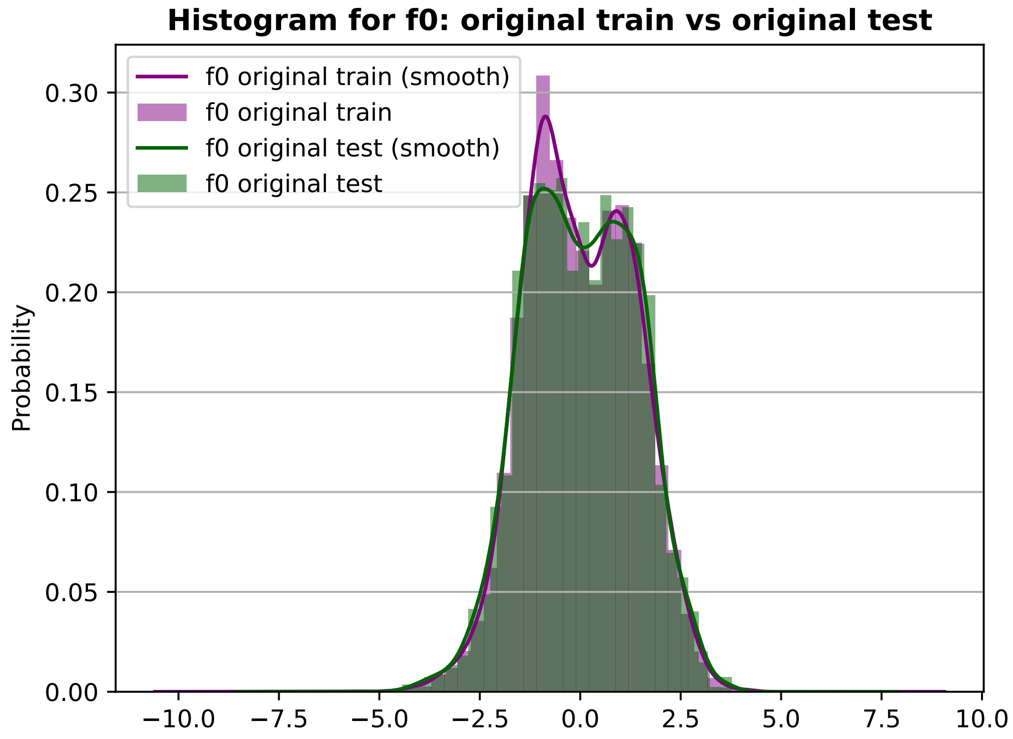

Figure 3. Histogram for `f0` attribute in original train and test subsets (no drift).

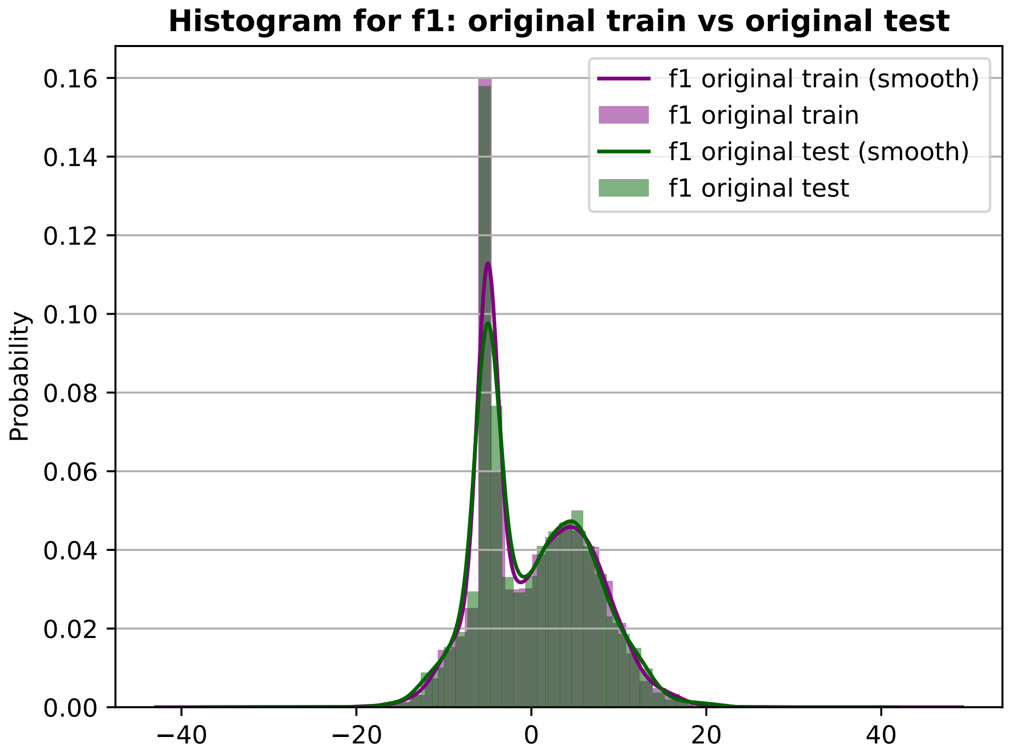

Figure 4. Histogram for `f1` attribute in original train and test subsets (no drift).

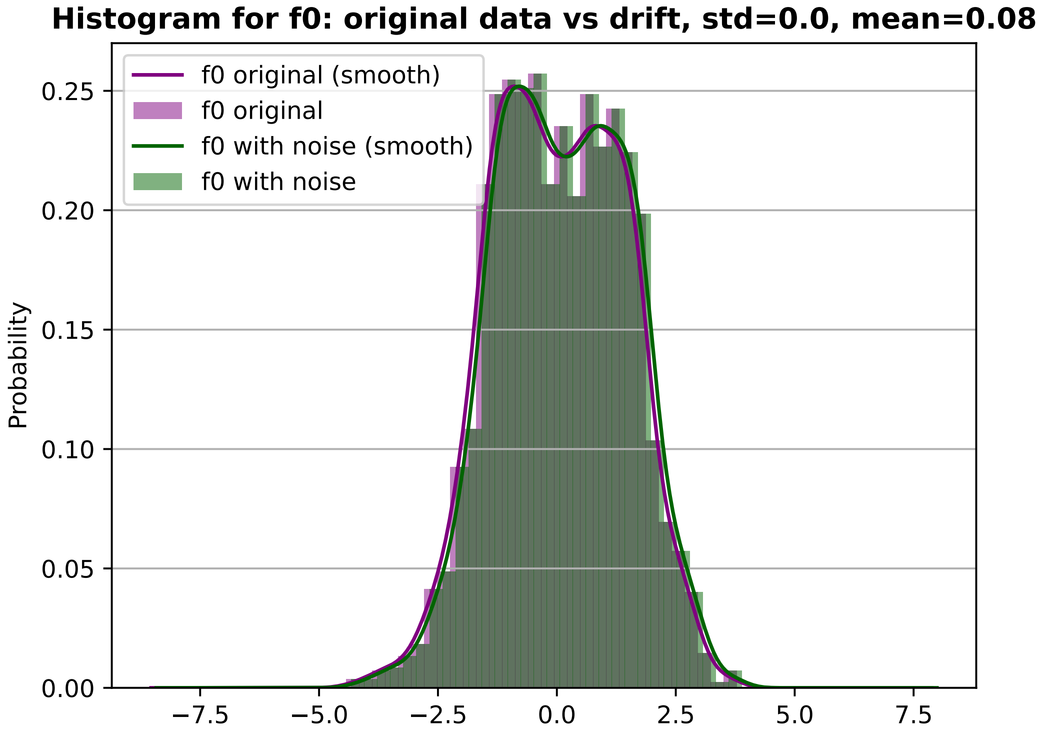

Figure 5. Histogram for `f0` attribute in the test: original vs lower drift detected by almost all algorithms with constant level noise.

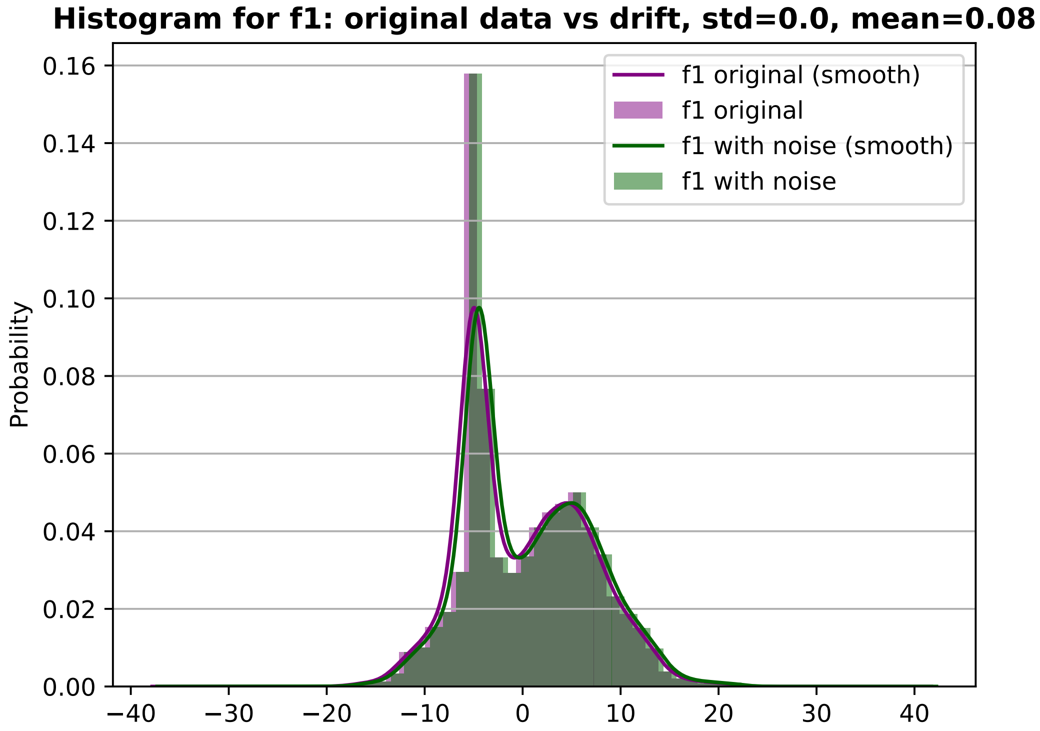

Figure 6. Histogram for `f1` attribute in the test: original vs lower drift detected by almost all algorithms with constant level noise.

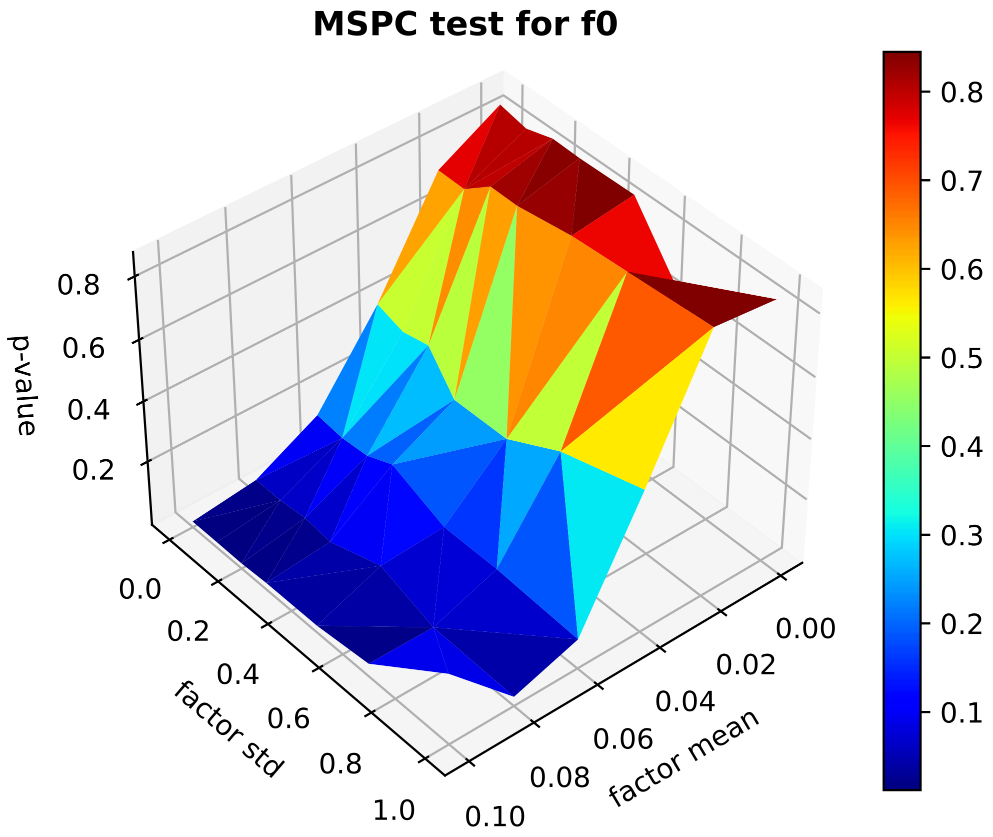

Figure 7. MSPC test p-values for different typologies of drift (mean and standard deviations) added to `f0` attribute.

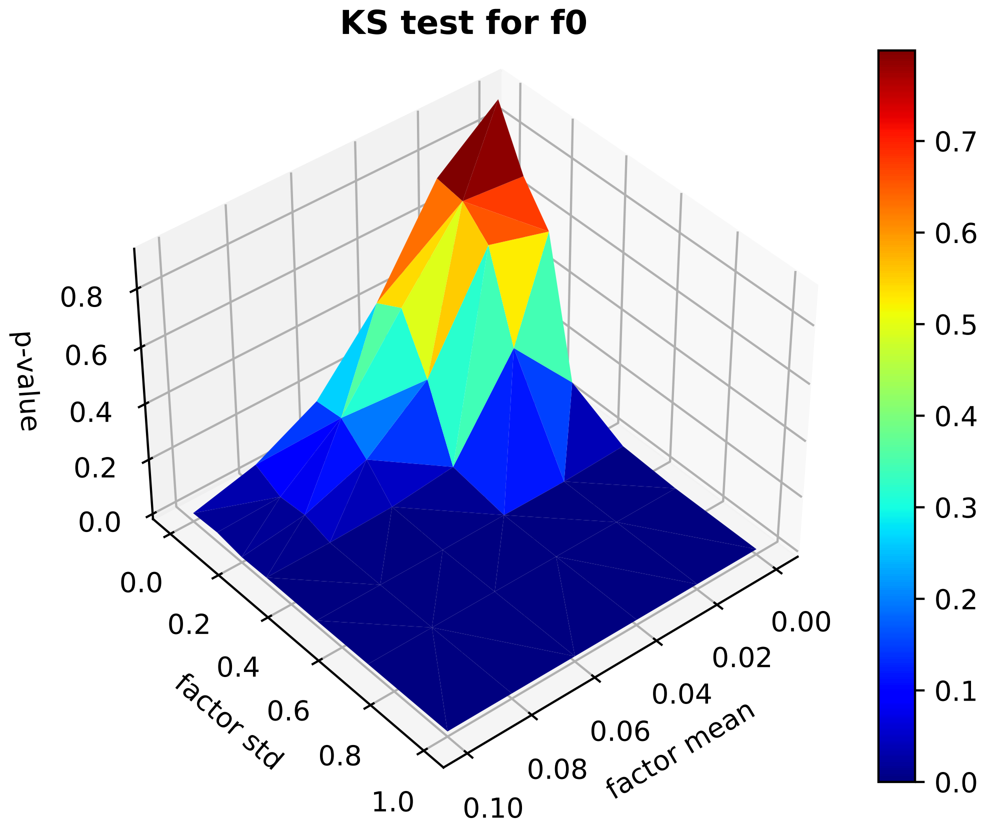

Figure 8. Kolmogorov–Smirnov test p-values for different typologies of drift (mean and standard deviations) added to `f0` attribute.

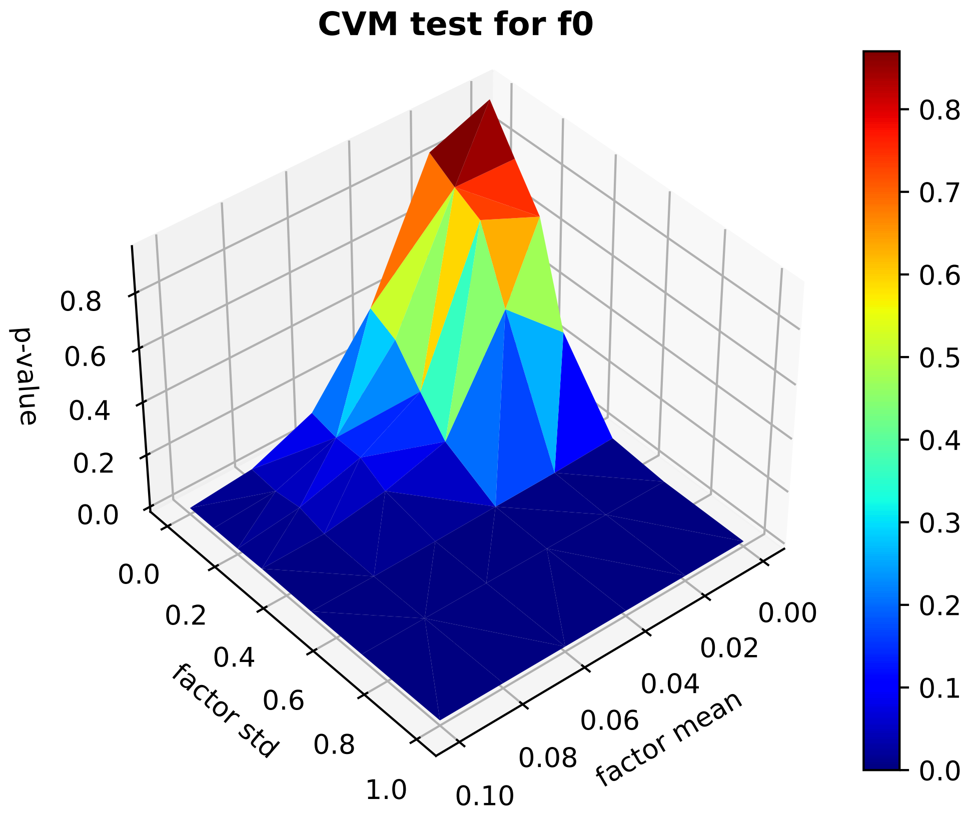

Figure 9. Cramér–von Mises test *p*-values for different typologies of drift (mean and standard deviations) added to `f0` attribute.

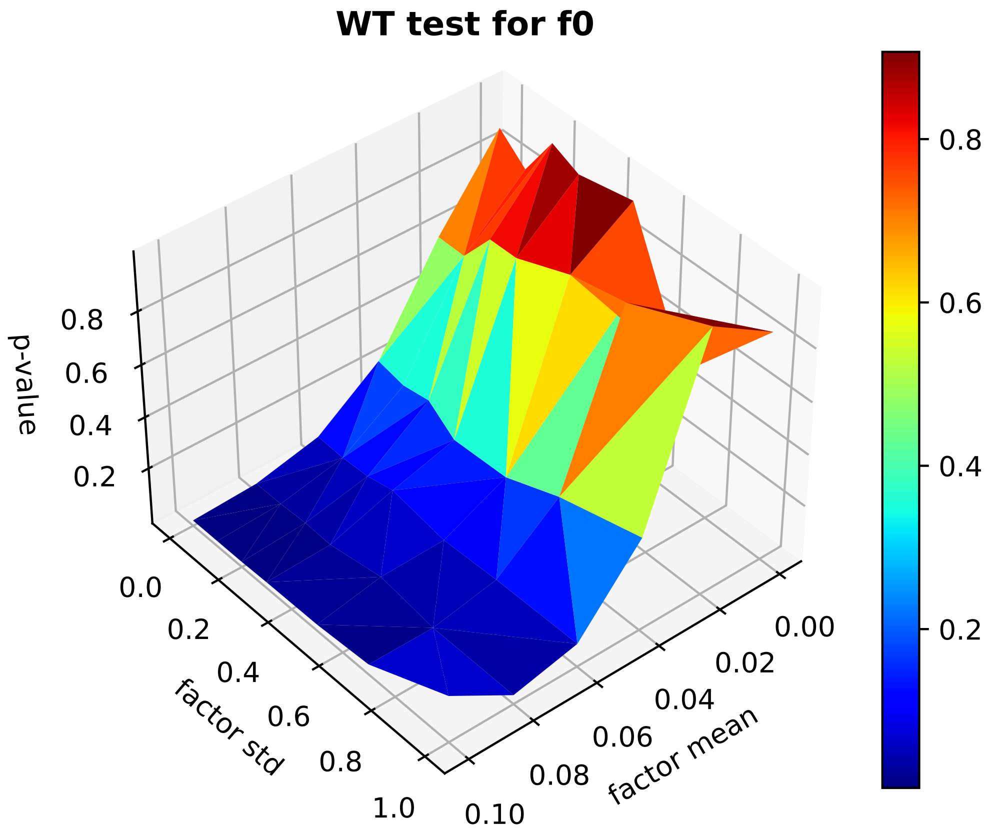

Figure 10. Welch *t*-test *p*-values for different typologies of drift (mean and standard deviations) added to `f0` attribute.

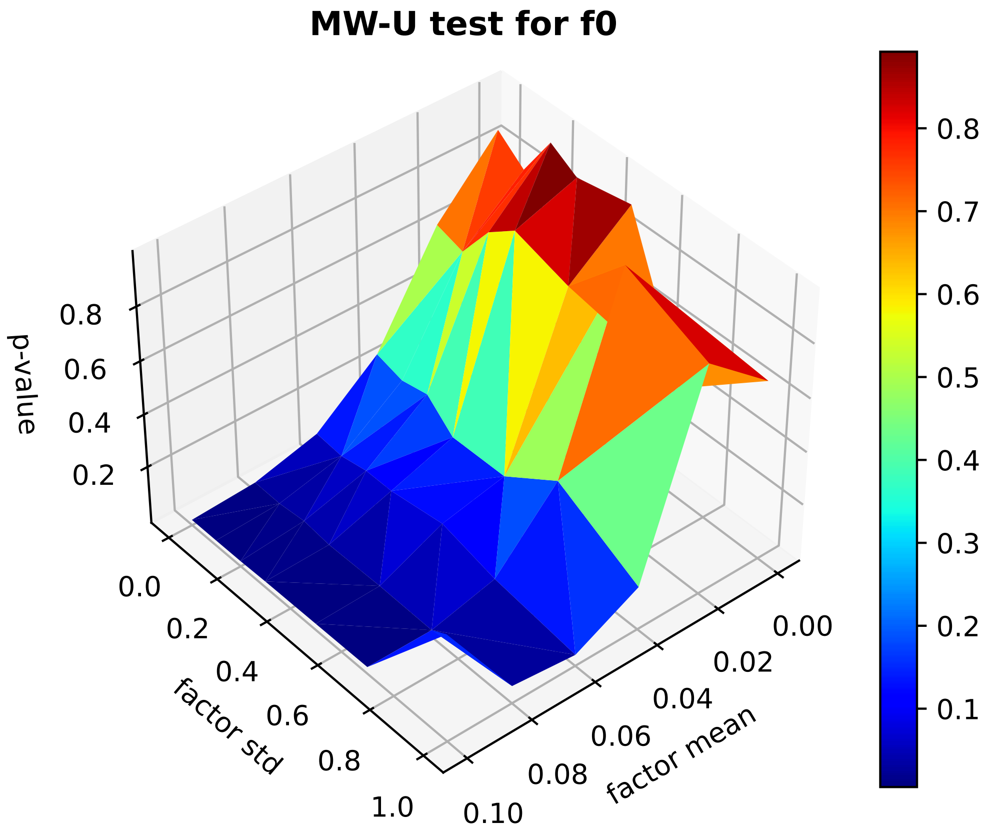

Figure 11. Mann–Whitney U test *p*-values for different typologies of drift (mean and standard deviations) added to `f0` attribute.

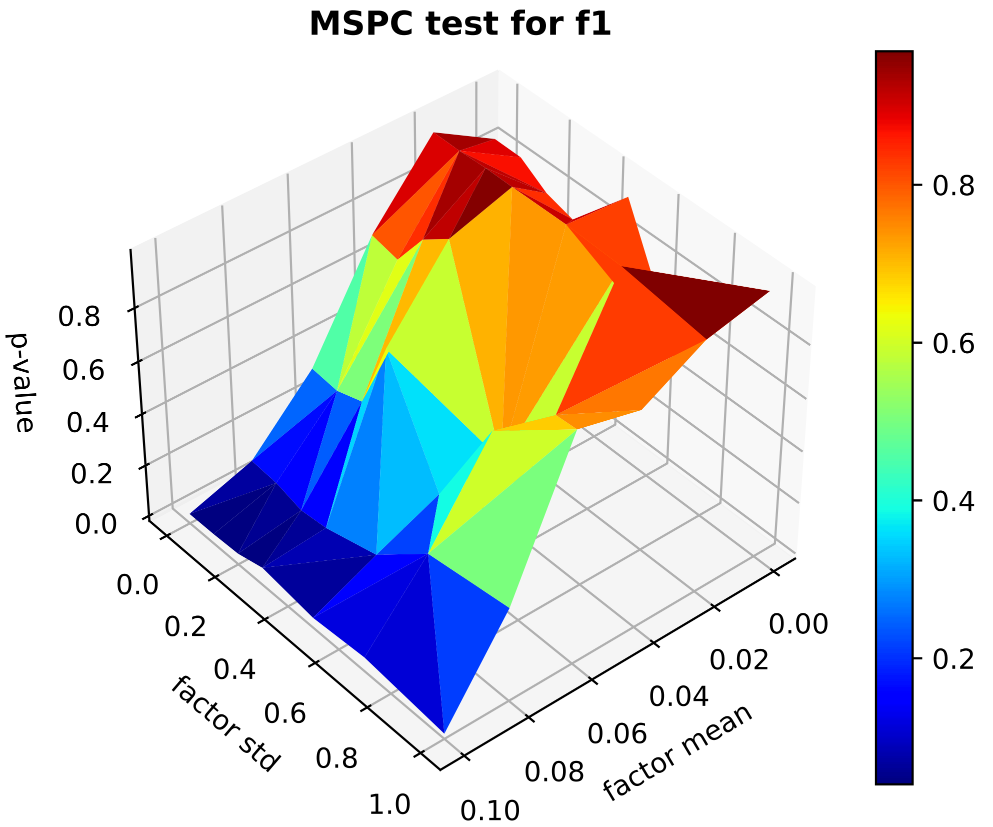

Figure 12. MSPC test *p*-values for different typologies of drift (mean and standard deviations) added to the `f1` attribute.

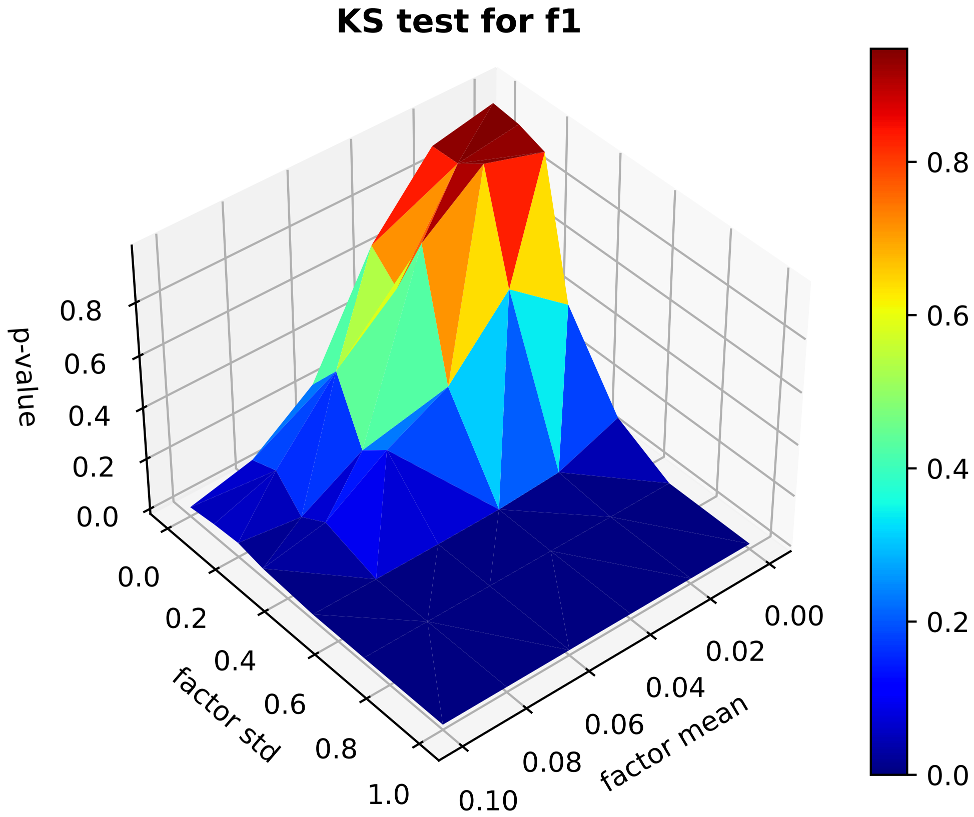

Figure 13. Kolmogorov–Smirnov test *p*-values for different typologies of drift (mean and standard deviations) added to the `f1` attribute.

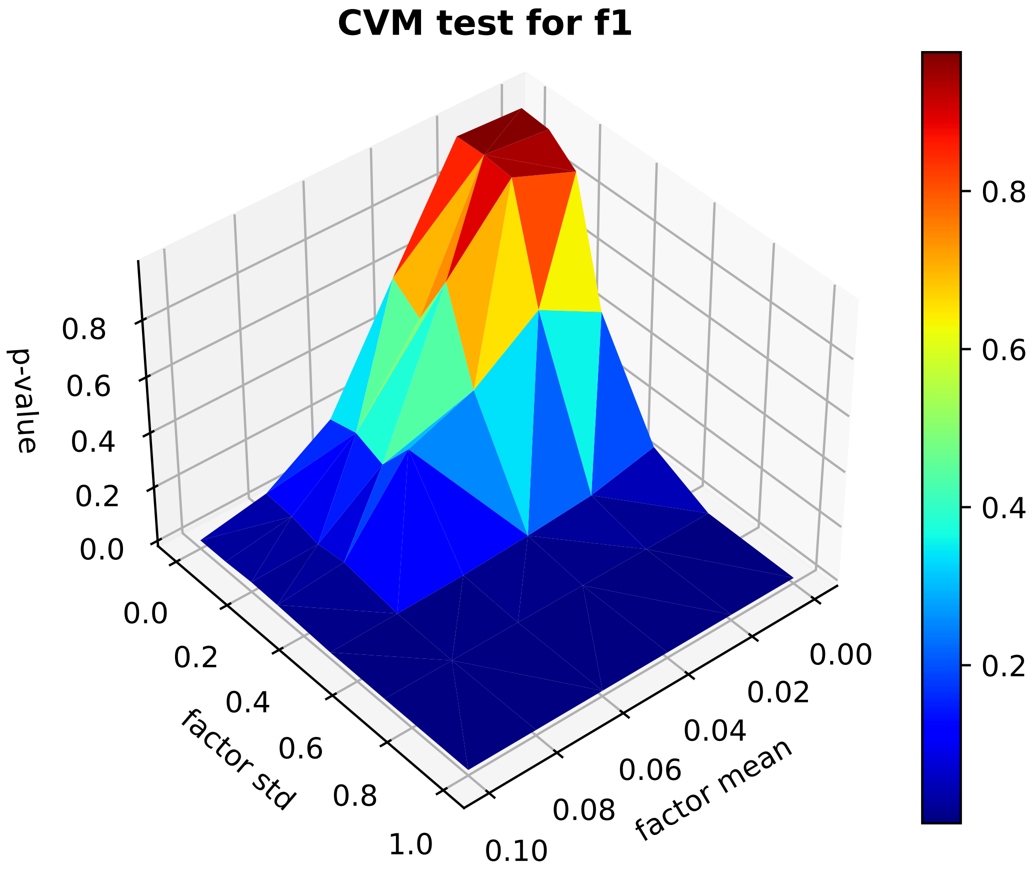

Figure 14. Cramér–von Mises test *p*-values for different typologies of drift (mean and standard deviations) added to the `f1` attribute.

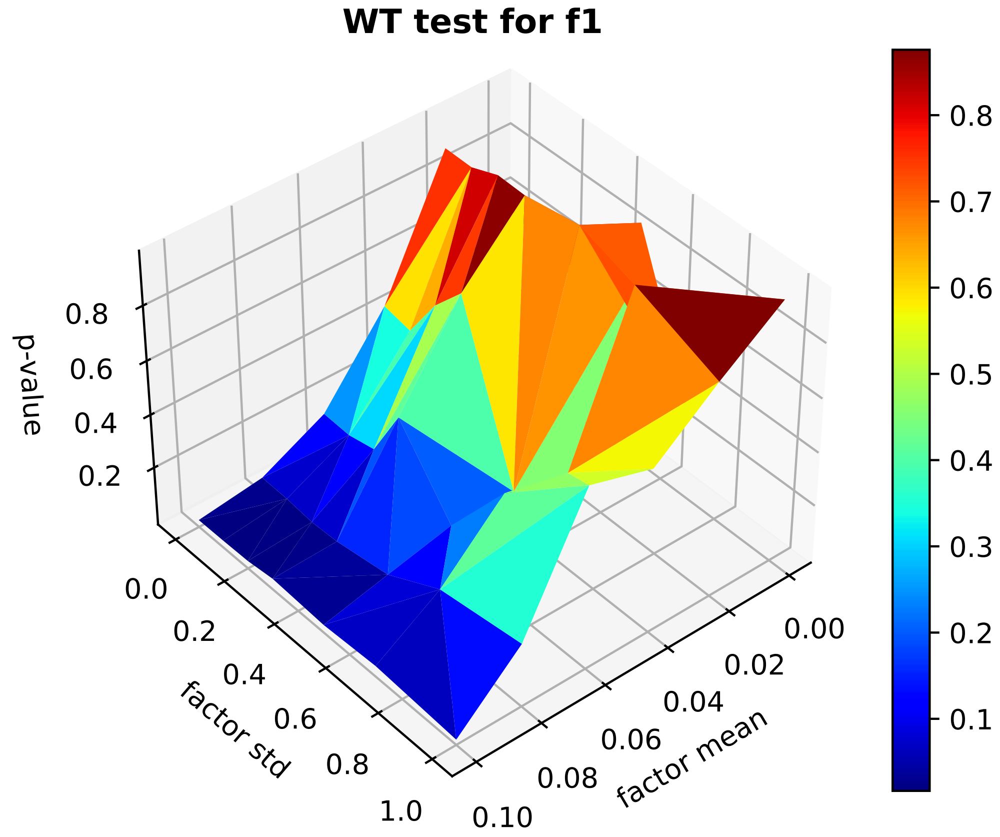

Figure 15. Welch *t*-test *p*-values for different typologies of drift (mean and standard deviations) added to the `f1` attribute.

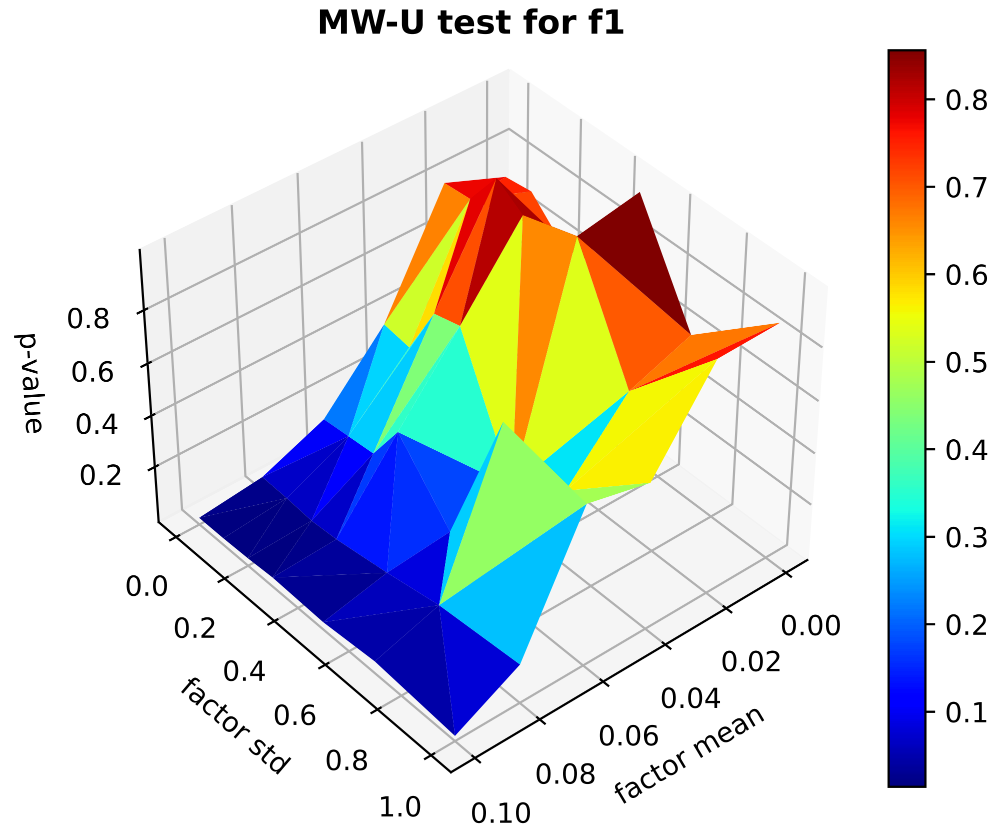

Figure 16. Mann–Whitney U test *p*-values for different typologies of drift (mean and standard deviations) added to the `f1` attribute.

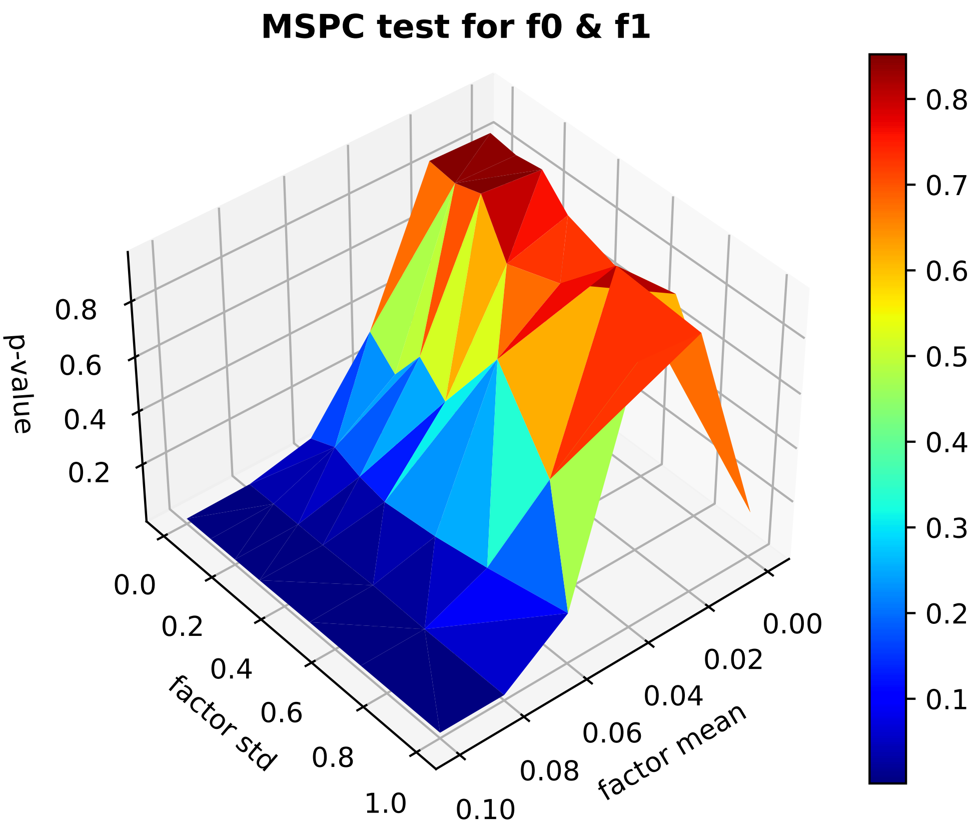

Figure 17. MSPC test *p*-values for different typologies of drift (mean and standard deviations) added to `f0` and `f1` attributes.

Table 1. Results for all five drift detection tests when introducing different levels and types of additive Gaussian noise to `f0` attribute.

Drift Params		MSPC			KS	CVM	WT	MW-U
Stdev	**Mean**	**Imp-`f0` (%)**	**Imp-`f1` (%)**	***p*-Value**	***p*-Value-`f0`**	***p*-Value-`f0`**	***p*-Value-`f0`**	***p*-Value-`f0`**
0	0	5.3713	94.6287	0.8467	0.9154	0.9464	0.8986	0.8719
0	0.02	48.6886	51.3114	0.7357	0.7482	0.8621	0.6022	0.6341
0	0.04	82.7008	17.2992	0.4025	0.4191	0.4005	0.2421	0.2621
0	0.06	92.0331	7.9669	0.1386	0.1798	0.1200	0.0691	0.0779
0	0.08	95.5082	4.4918	**0.0301**	0.0666	**0.0264**	**0.0137**	**0.0162**
0	0.1	97.1358	2.8642	0.0041	**0.0117**	0.0042	0.0019	0.0023
0.1	0	16.6359	83.3641	0.8279	0.7127	0.7955	0.8113	0.7925
0.1	0.02	48.9475	51.0525	0.7346	0.7331	0.8019	0.6013	0.6043
0.1	0.04	83.9692	16.0308	0.3746	0.4694	0.3521	0.2213	0.2349
0.1	0.06	92.3501	7.6499	0.1279	0.1898	0.0987	0.0637	0.0677
0.1	0.08	95.5300	4.4700	**0.0297**	**0.0216**	**0.0176**	**0.0137**	**0.0127**
0.1	0.1	97.2113	2.7887	0.0036	0.0157	0.0032	0.0016	0.0018
0.2	0	0.2936	99.7064	0.8540	0.5831	0.6493	0.9770	0.9580
0.2	0.02	28.7111	71.2889	0.8022	0.6453	0.7493	0.7355	0.7454
0.2	0.04	83.1063	16.8937	0.3939	0.2817	0.2258	0.2393	0.2562
0.2	0.06	92.1316	7.8684	0.1350	0.1094	0.0966	0.0695	0.0880
0.2	0.08	95.1511	4.8489	**0.0390**	**0.0294**	**0.0312**	**0.0184**	**0.0238**
0.2	0.1	97.3971	2.6029	0.0024	0.0005	0.0025	0.0012	0.0019

Table 1. *Cont.*

Drift Params		MSPC			KS	CVM	WT	MW-U
Stdev	**Mean**	**Imp-`f0` (%)**	**Imp-`f1` (%)**	***p*-Value**	***p*-Value-`f0`**	***p*-Value-`f0`**	***p*-Value-`f0`**	***p*-Value-`f0`**
0.3	0	2.7233	97.2767	0.8511	0.1046	0.2829	0.9300	0.8983
0.3	0.02	28.9301	71.0699	0.8013	0.3479	0.4896	0.7370	0.8208
0.3	0.04	87.6780	12.3220	0.2783	0.0322	0.1073	0.1609	0.1664
0.3	0.06	91.0287	8.9713	0.1737	**0.0081**	**0.0426**	0.0937	0.0878
0.3	0.08	95.6204	4.3796	**0.0278**	0.0024	0.0067	**0.0138**	**0.0111**
0.3	0.1	96.8001	3.1999	0.0074	0.0021	0.0043	0.0038	0.0045
0.5	0	0.4795	99.5205	0.8550	**0.0106**	**0.0183**	0.9716	0.9371
0.5	0.02	16.8320	83.1680	0.8284	0.0016	0.0099	0.8189	0.7591
0.5	0.04	87.7169	12.2831	0.2815	0.0018	0.0067	0.1716	0.1702
0.5	0.06	93.1942	6.8058	0.1013	0.0001	0.0029	0.0585	0.1196
0.5	0.08	93.5306	6.4694	0.0892	0.0001	0.0004	0.0522	**0.0136**
0.5	0.1	96.7720	3.2280	**0.0079**	0.0000	0.0003	**0.0054**	0.0061
0.7	0	67.8427	32.1573	0.6145	**0.0000**	**0.0000**	0.4760	0.4038
0.7	0.02	10.9368	89.0632	0.8399	0.0000	0.0001	0.8628	0.9777
0.7	0.04	84.1370	15.8630	0.3757	0.0000	0.0001	0.2560	0.3098
0.7	0.06	93.2027	6.7973	0.1012	0.0000	0.0000	0.0682	0.0675
0.7	0.08	95.7380	4.2620	**0.0260**	0.0000	0.0000	**0.0201**	**0.0084**
0.7	0.1	95.6538	4.3462	0.0279	0.0000	0.0000	0.0207	0.0078
1	0	13.3527	86.6473	0.8367	**0.0000**	**0.0000**	0.8525	0.6560
1	0.02	0.0024	99.9976	0.8575	0.0000	0.0000	0.9982	0.8446
1	0.04	80.3369	19.6631	0.4571	0.0000	0.0000	0.3435	0.1460
1	0.06	93.7628	6.2372	0.0838	0.0000	0.0000	0.0713	**0.0248**
1	0.08	96.2735	3.7265	**0.0161**	0.0000	0.0000	**0.0161**	0.0487
1	0.1	89.7701	10.2299	0.2217	0.0000	0.0000	0.1633	0.3900

One remarkable point is that hypothesis tests become more sensitive as the sample size increases [24,25], and this could give a false idea of the absolute sensitivity of the method, which could be perceived as counterproductive. To illustrate this point, in Figures 3 and 4, we show histograms of `f0` and `f1` attributes in train and test subsets (thus, randomly sampled from the same original population) without any additive drift added, which seem to show remarkable differences when compared to Figures 5 and 6, which show histograms of the original test partition and a significantly detected drift in the same subset. We are considering a large test sample (3k examples), so the hypothesis tests are very sensitive because the ample number of examples allows them to characterize the original population effectively.

In sum, there is a trade-off between the sample size and the considered *p*-value threshold, so data scientists will need to define some specifications for the drift detection stage within their particular application and use case:

- How often is an analysis done?
- How many examples make up the sample to be analyzed (with the possibility of overlap)?
- Which *p*-value threshold is considered significant according to the effects to be detected in the real process?

For this reason, the histograms presented must not be linked to the *p*-values detected by the tests in an absolute way in each case. However, they must be put in the context of applying the hypothesis tests to sets of thousands of examples. Instead, our primary goal is to relatively compare the different test-based drift detection methods.

Table 2. Results for all five drift detection tests when introducing different levels and types of additive Gaussian noise to `f1` attribute.

Drift Params		**MSPC**			**KS**	**CVM**	**WT**	**MW-U**
Stdev	**Mean**	**Imp-`f0` (%)**	**Imp-`f1` (%)**	***p*-Value**	***p*-Value-`f1`**	***p*-Value-`f1`**	***p*-Value-`f1`**	***p*-Value-`f1`**
0	0	5.3713	94.6287	0.8467	0.9668	0.9664	0.5932	0.7090
0	0.02	54.9662	45.0338	0.9839	0.9187	0.9789	0.9083	0.8021
0	0.04	2.6960	97.3040	0.7178	0.6609	0.5960	0.4445	0.3809
0	0.06	0.8036	99.1964	0.3289	0.2429	0.2025	0.1574	0.1329
0	0.08	0.3790	99.6210	0.0947	0.0722	0.0541	**0.0391**	**0.0341**
0	0.1	0.2196	99.7804	**0.0171**	**0.0160**	**0.0111**	0.0067	0.0063
0.1	0	5.3377	94.6623	0.8458	0.9543	0.9583	0.5927	0.7231
0.1	0.02	54.0417	45.9583	0.9836	0.9218	0.9818	0.9067	0.8118
0.1	0.04	2.4710	97.5290	0.6967	0.5678	0.5219	0.4247	0.3656
0.1	0.06	0.7805	99.2195	0.3185	0.3721	0.2283	0.1520	0.1438
0.1	0.08	0.3640	99.6360	0.0860	0.1094	**0.0466**	**0.0355**	**0.0300**
0.1	0.1	0.2218	99.7782	**0.0178**	**0.0289**	0.0112	0.0071	0.0064
0.2	0	3.6058	96.3942	0.7805	0.9187	0.8784	0.5134	0.6252
0.2	0.02	77.5906	22.4094	0.9886	0.9886	0.9710	0.9457	0.9585
0.2	0.04	5.2186	94.7814	0.8427	0.8112	0.7249	0.5908	0.5711
0.2	0.06	0.8581	99.1419	0.3532	0.1362	0.1883	0.1747	0.1544
0.2	0.08	0.3123	99.6877	0.0576	**0.0024**	**0.0222**	**0.0247**	**0.0209**
0.2	0.1	0.2346	99.7654	**0.0223**	0.0358	0.0162	0.0091	0.0096
0.3	0	3.1196	96.8804	0.7513	0.4060	0.4474	0.4851	0.6018
0.3	0.02	74.7187	25.2813	0.9881	0.5883	0.5760	0.9420	0.8920
0.3	0.04	10.0569	89.9431	0.9153	0.3362	0.4097	0.7070	0.5987
0.3	0.06	1.9328	98.0672	0.6304	0.2165	0.3222	0.3724	0.3184
0.3	0.08	0.3329	99.6671	0.0690	0.0625	**0.0353**	**0.0298**	**0.0292**
0.3	0.1	0.2954	99.7046	**0.0491**	**0.0160**	0.0128	0.0220	0.0165
0.5	0	34.1730	65.8270	0.9743	0.1078	0.0961	0.8661	0.9995
0.5	0.02	93.3606	6.6394	0.9906	**0.0193**	**0.0443**	0.9742	0.9564
0.5	0.04	0.5991	99.4009	0.2268	0.0041	0.0258	0.1147	0.1264
0.5	0.06	0.6036	99.3964	0.2286	0.0012	0.0123	0.1174	0.0863
0.5	0.08	0.4306	99.5694	0.1275	0.0001	0.0074	0.0644	0.0629
0.5	0.1	0.2244	99.7756	**0.0193**	0.0008	0.0028	**0.0098**	**0.0089**
0.7	0	1.2908	98.7092	0.5053	**0.0008**	**0.0019**	0.3039	0.6122
0.7	0.02	48.2329	51.7671	0.9819	0.0001	0.0006	0.9034	0.5265
0.7	0.04	1.5061	98.4939	0.5583	0.0002	0.0008	0.3426	0.2745
0.7	0.06	1.8993	98.1007	0.6291	0.0001	0.0007	0.3996	0.6714
0.7	0.08	0.7240	99.2760	0.2980	0.0000	0.0001	0.1733	0.1043
0.7	0.1	0.2616	99.7384	**0.0349**	0.0000	0.0008	**0.0213**	**0.0271**
1	0	73.3841	26.6159	0.9881	**0.0000**	**0.0000**	0.9461	0.8821
1	0.02	13.5388	86.4612	0.9374	0.0000	0.0000	0.7781	0.8767
1	0.04	4.1385	95.8615	0.8088	0.0000	0.0000	0.5935	0.5480
1	0.06	6.2257	93.7743	0.8687	0.0000	0.0000	0.6653	0.6054
1	0.08	0.8171	99.1829	0.3420	0.0000	0.0000	0.2194	0.1317
1	0.1	0.0804	99.9196	**0.0000**	0.0000	0.0000	**0.0001**	**0.0001**

Table 3. Results for all five drift detection tests when introducing different levels and types of additive Gaussian noise to both `f0` and `f1` attributes.

Drift Params		**MSPC**			**KS**		**CVM**		**WT**		**MW-U**	
Stdev	**Mean**	**Imp-`f0` (%)**	**Imp-`f1` (%)**	***p*-Val**	***p*-Val-`f0`**	***p*-Val-`f1`**	***p*-Val-`f0`**	***p*-Val-`f1`**	***p*-Val-`f0`**	***p*-Val-`f1`**	***p*-Val-`f0`**	***p*-Val-`f1`**
0	0	5.3713	94.6287	0.8467	0.9154	0.9668	0.9464	0.9664	0.8986	0.5932	0.8719	0.7090
0	0.02	95.3279	4.6721	0.8549	0.7482	0.9187	0.8621	0.9789	0.6022	0.9083	0.6341	0.8021
0	0.04	70.0022	29.9978	0.3412	0.4191	0.6609	0.4005	0.5960	0.2421	0.4445	0.2621	0.3809
0	0.06	62.2468	37.7532	0.0539	0.1798	0.2429	0.1200	0.2025	0.0691	0.1574	0.0779	0.1329
0	0.08	58.7656	41.2344	**0.0034**	0.0666	0.0722	**0.0264**	0.0541	**0.0137**	**0.0391**	**0.0162**	**0.0341**
0	0.1	56.8045	43.1955	0.0001	**0.0117**	**0.0160**	0.0042	**0.0111**	0.0019	0.0067	0.0023	0.0063
0.1	0	21.2900	78.7100	0.8317	0.8719	0.8758	0.7888	0.9390	0.7897	0.6087	0.7385	0.7120
0.1	0.02	92.9932	7.0068	0.8410	0.5780	0.9737	0.7274	0.9814	0.5893	0.8821	0.6236	0.7904
0.1	0.04	75.5982	24.4018	0.2509	0.4416	0.6661	0.2919	0.6276	0.1687	0.4348	0.1920	0.3814
0.1	0.06	60.2588	39.7412	0.0957	0.2308	0.4933	0.1809	0.3176	0.1092	0.1946	0.1335	0.1928
0.1	0.08	57.1209	42.8791	**0.0047**	0.0899	0.1249	**0.0399**	0.0586	**0.0184**	**0.0418**	**0.0249**	**0.0364**
0.1	0.1	57.3627	42.6373	0.0001	**0.0126**	**0.0228**	0.0032	**0.0112**	0.0014	0.0062	0.0017	0.0061
0.2	0	0.2796	99.7204	0.8454	0.4789	0.9629	0.4995	0.8561	0.9769	0.5843	0.9926	0.7109
0.2	0.02	95.1652	4.8348	0.8700	0.4837	0.7229	0.6058	0.8820	0.6259	0.9125	0.6476	0.7958
0.2	0.04	70.6313	29.3687	0.3909	0.1774	0.3680	0.3209	0.4019	0.2746	0.4817	0.3259	0.3413
0.2	0.06	50.2453	49.7547	0.0578	0.1231	0.0955	0.1414	0.1511	0.1093	0.1101	0.1262	0.1054
0.2	0.08	55.6318	44.3682	**0.0032**	**0.0129**	0.0625	**0.0220**	**0.0432**	**0.0164**	**0.0326**	**0.0190**	**0.0319**
0.2	0.1	51.6875	48.3125	0.0001	0.0032	**0.0110**	0.0036	0.0081	0.0036	0.0051	0.0032	0.0060
0.3	0	23.9742	76.0258	0.7469	0.2429	0.2459	0.2382	0.3044	0.7243	0.5342	0.6067	0.6728
0.3	0.02	28.4924	71.5076	0.6849	0.2001	0.2055	0.3548	0.2622	0.6608	0.4901	0.7046	0.4646
0.3	0.04	62.1547	37.8453	0.2947	**0.0456**	0.2920	0.1205	0.3740	0.2467	0.3678	0.2355	0.3641
0.3	0.06	66.6460	33.3540	**0.0372**	0.0377	0.1213	**0.0478**	0.1299	**0.0482**	0.1647	**0.0487**	0.1299
0.3	0.08	47.0792	52.9208	0.0018	0.0046	**0.0092**	0.0107	**0.0115**	0.0229	**0.0152**	0.0304	**0.0140**
0.3	0.1	51.0563	48.9437	0.0002	0.0061	0.0083	0.0047	0.0082	0.0051	0.0061	0.0058	0.0059
0.5	0	56.1989	43.8011	0.6795	**0.0076**	0.0677	**0.0064**	0.0827	0.5441	0.5930	0.5464	0.7786
0.5	0.02	99.7637	0.2363	0.7523	0.0010	0.1362	0.0172	0.1709	0.4900	0.9730	0.4416	0.9866
0.5	0.04	49.8440	50.1560	0.5987	0.0033	**0.0060**	0.0177	**0.0450**	0.5124	0.5116	0.4530	0.3859
0.5	0.06	70.7899	29.2101	0.0610	0.0013	0.0012	0.0036	0.0165	0.0674	0.2436	0.0583	0.2155
0.5	0.08	50.6362	49.3638	**0.0070**	0.0000	0.0047	0.0012	0.0056	**0.0414**	**0.0431**	**0.0317**	**0.0324**
0.5	0.1	43.3304	56.6696	0.0017	0.0001	0.0004	0.0018	0.0016	0.0310	0.0140	0.0371	0.0143
0.7	0	92.0178	7.9822	0.7376	**0.0007**	**0.0013**	**0.0004**	**0.0026**	0.5003	0.8434	0.5590	0.9550
0.7	0.02	48.6952	51.3048	0.9542	0.0000	0.0003	0.0001	0.0010	0.8495	0.8466	0.8615	0.8831
0.7	0.04	69.1197	30.8803	0.3030	0.0000	0.0000	0.0000	0.0000	0.2526	0.4474	0.4460	0.3301
0.7	0.06	48.0169	51.9831	0.1003	0.0000	0.0000	0.0002	0.0003	0.1836	0.1698	0.2798	0.2604
0.7	0.08	60.3281	39.6719	**0.0016**	0.0000	0.0000	0.0000	0.0004	**0.0125**	**0.0429**	**0.0046**	**0.0303**
0.7	0.1	69.4348	30.5652	0.0011	0.0000	0.0001	0.0000	0.0009	0.0059	0.0671	0.0043	0.0344
1	0	0.6257	99.3743	0.1504	**0.0000**	**0.0000**	**0.0000**	**0.0000**	0.8963	0.1013	0.6468	0.0915
1	0.02	65.7453	34.2547	0.9307	0.0000	0.0000	0.0000	0.0000	0.7948	0.8510	0.9034	0.5344
1	0.04	0.2520	99.7480	0.9520	0.0000	0.0000	0.0000	0.0000	0.9894	0.7899	0.8997	0.8890
1	0.06	53.7951	46.2049	0.1721	0.0000	0.0000	0.0000	0.0000	0.2468	0.2791	0.1921	0.4751
1	0.08	75.1375	24.8625	**0.0010**	0.0000	0.0000	0.0000	0.0000	**0.0059**	0.1181	**0.0077**	**0.0500**
1	0.1	47.6246	52.3754	0.0015	0.0000	0.0000	0.0000	0.0000	0.0337	**0.0270**	0.0428	0.0256

Finally, Figure 2 shows a general graphical schema to summarize training and production MSPC test implementation.

4. Results

A comparative study was conducted to inspect the differences among the analyzed methods regarding drift detection capabilities. The main results are as follows:

- All five tests are well capable of detecting drift consisting of an additive constant level (mean), as expected. This is evident in Figures 7–17, where the p-value increases (indicated in bright colors) as we approach higher values along the mean axis, toward the right side of each plot.
- While Kolmogorov–Smirnov and Cramér–von Mises tests are able to detect drift regarding the modification of the deviation level (standard deviation), Welch's t-test, Mann–Whitney U-test, and MSPC are not. This is because they are mathematically designed to be unaware of deviation because higher and lower values tend to compensate and thus cancel each other out.
- The four off-the-shelf methods work similarly (in pairs, as analyzed) regarding the needed level of drift to be activated (i.e., to obtain a p-value lower than 0.05). Conversely, the decision from MSPC is like Welch's t-test and the Mann–Whitney U-test and is mainly based on the drift in the mean.
- The importance variation of MSPC works as expected: when more drift is included in the modified attribute, its importance rises. In Table 2, we see that the original much higher importance of `f1` on the base scenario gives way to the increasing importance of `f0` as we add more significant amounts of noise to it. Also, when noise is added to both attributes, their importance tends to balance each other out.
- In Table 3, where additive noise is included in both `f0` and `f1` attributes, their importance values tend to equalize because the noises applied to them have the same relative size.

5. Discussion

The results presented are derived from a fixed-size data batch sampled from a normal distribution. Hence, this scenario may affect the algorithm's behavior in terms of stability, sensitivity, and biases. However, several relative noise levels are considered in this study, so the potential bias of the results caused by them can be properly assessed. We also consider other limitations of our study, the most important being the restricted scope regarding the synthetic dataset used. Nevertheless, the results provide a starting point from which to build once the method has been precisely validated.

6. Conclusions and Future Work

We studied the behaviors of four off-the-shelf, data drift detection, batch, univariate, statistical-test-based algorithms using the Frouros implementation [10] (i.e., the Kolmogorov–Smirnov test, the Cramér–von Mises test, the Mann–Whitney U test, Welch's t-test), and our multivariate, statistical-test-based proposal based on the Hotelling statistic in a drift detection task, where we added different levels and types of Gaussian noise to each input variable of a synthetic dataset.

Each statistic possesses different properties, and based on the study's findings, the preferred univariate tests are the Kolmogorov–Smirnov and Cramér–von Mises tests. They have shown remarkable performance in detecting changes in position (mean) and variability (deviation). Moreover, both Welch's t-test and the Mann–Whitney U test can only detect changes in position.

Additionally, the Hotelling test introduced (MSPC) is convenient as a multivariate test due to its ability to detect position changes from a broader perspective. It naturally enables the introduction of weights related to the relevance of model variables, which offers users a comprehensive and concise overview of the multivariate data and the significance of each variable, enabling rapid detection of data that may contribute to the underlying problem in the presence of data drift.

Both univariate and multivariate approaches can detect different types of drift in data, and each type can exhibit strengths and weaknesses. For instance, while univariate methods can be less computationally demanding and work well for a few attributes, they might be less suitable for datasets containing many variables. Conversely, multivariate methods such as MSPC can benefit from a holistic view of the drift status of data in large datasets. However, their computational demands might pose a challenge for hardware with constrained resources.

By employing these selected tests, this study aims to ensure the detection of data drift and provide users with valuable insights into the current situations regarding their data. Moreover, although the tests discussed are designed primarily for quantitative data, it is easy to introduce qualitative variables as different state markers and develop a different model for each possible state.

Furthermore, in future work, we hope to apply the method to several real-world datasets [26,27] and to refine the Hotelling approach by using it on a hidden multidimensional feature projection [28], enabling us to detect changes in location as well as in the correlation breakdown between variables, which relates directly to covariance. A future study examining the relationship between noise in data drift and concept drift could be of value. Also, a test for qualitative data may be introduced by considering classes as markers of different states and learning a different data drift model for each possible class.

Finally, for each particular downstream application, a predefined system of alerts can assist the user in detecting data drift and in establishing automatic retraining schemes. This is important in cases where the system needs to adapt well to ongoing incoming data sampled from a new distribution.

Author Contributions: Conceptualization, J.R.N.-C., V.O.C. and D.M.E.; methodology, J.R.N.-C. and V.O.C.; validation, J.R.N.-C. and V.O.C.; formal analysis, J.R.N.-C. and V.O.C.; investigation, J.R.N.-C. and V.O.C.; resources, J.R.N.-C., V.O.C. and D.M.E.; writing—original draft preparation, J.R.N.-C. and V.O.C.; writing—review and editing, V.O.C. and J.R.N.-C.; visualization, J.R.N.-C., V.O.C. and D.M.E.; supervision, J.R.N.-C. and V.O.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Cervera Network of Excellence Project in Data-based Enabling Technologies (AI4ES), co-funded by the Centre for Industrial and Technological Development, E.P.E. (CDTI), and by the European Union through the Next Generation EU Fund, within the Cervera Aids program for Technological Centres, with the expedient number CER-20211030. Project AITANA MoMo: Monitoring of Models in Production, co-financed by the Valencian Institute of Business Competitiveness (IVACE) and the European Regional Development Fund (FEDER), with expedient number IMDEEA/2022/50, and by Generalitat Valenciana through IVACE (Valencian Institute of Business Competitiveness) distributed nominatively to Valencian technological innovation centres under project IMAMCA/2024/11.

Data Availability Statement: Dataset available on request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rahmani, K.; Thapa, R.; Tsou, P.; Chetty, S.C.; Barnes, G.; Lam, C.; Tso, C.F. Assessing the effects of data drift on the performance of machine learning models used in clinical sepsis prediction. *Int. J. Med. Inform.* **2022**, *173*, 104930. [CrossRef] [PubMed]
2. Ackerman, S.; Raz, O.; Zalmanovici, M.; Zlotnick, A. Automatically detecting data drift in machine learning classifiers. *arXiv* **2021**. [CrossRef]
3. Disabato, S. & Roveri, M. Tiny machine learning for concept drift. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *35*, 8470–8481. [CrossRef] [PubMed]

4. Hoens, T.; Polikar, R.; Chawla, N. Learning from streaming data with concept drift and imbalance: An overview. *Prog. Artif. Intell.* **2012**, *1*, 89–101. [CrossRef]
5. Lu, J.; Liu, A.; Dong, F.; Gu, F.; Gama, J.; Zhang, G. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 2346–2363. [CrossRef]
6. Duckworth, C.; Chmiel, F.P.; Burns, D.K.; Zlatev, Z.D.; White, N.M.; Daniels, T.W.V.; Kiuber, M.; Boniface, M.J. Using explainable machine learning to characterise data drift and detect emergent health risks for emergency department admissions during COVID-19. *Sci. Rep.* **2021**, *11*, 23017. [CrossRef] [PubMed]
7. Paka, A. How to Detect Model Drift in MLOps Monitoring. 2020. Available online: https://towardsdatascience.com/how-to-detect-model-drift-in-mlops-monitoring-7a039c22eaf9 (accessed on 2 September 2024).
8. Machiraju, S. Why Data Drift Detection Is Important and How Do You Automate It in 5 Simple Steps. 2021. Available online: https://towardsdatascience.com/why-data-drift-detection-is-important-and-how-do-you-automate-it-in-5-simple-steps-96d611095d93 (accessed on 2 September 2024).
9. Hodler, A. Drift in Machine Learning: How to Identify Issues Before You Have a Problem. 2022. Available online: https://www.fiddler.ai/blog/drift-in-machine-learning-how-to-identify-issues-before-you-have-a-problem (accessed on 2 September 2024).
10. Céspedes-Sisniega, J.; López-García, Á. Frouros: A Python library for drift detection in machine learning systems. *arXiv* **2022**. [CrossRef]
11. Massey, F.J., Jr. The Kolmogorov-Smirnov Test for Goodness of Fit. *J. Am. Stat. Assoc.* **1951**, *46*, 68–78. [CrossRef]
12. Cramér, H. On the composition of elementary errors. *Scand. Actuar. J.* **1928**, 1928, 13–74. [CrossRef]
13. Welch, B.L. *The Generalization of 'Student's' Problem When Several Different Population Variances Are Involved*; Biometrika Trust; Oxford University Press: Oxford, UK, 1947; Volume 34, pp. 28–35. [CrossRef]
14. Mann, H.B.; Whitney, D.R. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *Ann. Math. Stat.* **1947**, *18*, 50–60. [CrossRef]
15. Suárez, J.L.; García, S.; Herrera, F. A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges. *Neurocomputing* **2021**, *425*, 300–322. [CrossRef]
16. Dries, A.; Rückert, U. Adaptive concept drift detection. *Stat. Anal. Data Mining Asa Data Sci. J.* **2009**, *2*, 311–327. [CrossRef]
17. Montiel, J.; Halford, M.; Mastelini, S.; Bolmier, G.; Sourty, R.; Vaysse, R.; Zouitine, A.; Gomes, H.; Read, J.; Abdessalem, T.; et al. River: Machine learning for streaming data in Python. *J. Mach. Learn. Res.* **2021**, 22, 1–8. [CrossRef]
18. Leigh, N.; Schill, T.; Lindsay, I.; Srivastava, A.; McNamara, K.; Jarmale, S. Menelaus. 2022. Available online: https://github.com/mitre/menelaus (accessed on 8 April 2024).
19. Titouna, C.; Titouna, F.; Ari, A.A.A. Outlier Detection Algorithm based on Mahalanobis Distance for Wireless Sensor Networks. In Proceedings of the 2019 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 23–25 January 2019; pp. 1–6. [CrossRef]
20. Yin, L.; Lv, L.; Wang, D.; Qu, Y.; Chen, H.; Deng, W. Spectral Clustering Approach with K-Nearest Neighbor and Weighted Mahalanobis Distance for Data Mining. *Electronics* **2023**, *12*, 3284. [CrossRef]
21. Wölfel, M.; Ekenel, H.K. Feature weighted Mahalanobis distance: Improved robustness for Gaussian classifiers. In Proceedings of the 2005 13th European Signal Processing Conference, Antalya, Turkey, 4–8 September 2005; pp. 1–4. Available online: https://eurasip.org/Proceedings/Eusipco/Eusipco2005/defevent/papers/cr1853.pdf (accessed on 4 November 2024).
22. Yuan, M. High Dimensional Inverse Covariance Matrix Estimation via Linear Programming. *J. Mach. Learn. Res.* **2010**, *11*, 2261–2286.
23. Callot, L.; Caner, M.; Ulasan, E.; Özlem Önder, A. A Nodewise Regression Approach to Estimating Large Portfolios. *J. Bus. Econ. Stat.* **2019**, *39*, 520–531. [CrossRef]
24. Jones, S.R.; Carley, S.; Harrison, M. An introduction to power and sample size estimation. *Emerg. Med. J.* **2003**, *20*, 453–458. [CrossRef] [PubMed]
25. Serdar, C.C.; Cihan, M.; Yücel, D.; Serdar, M.A. Sample size, power and effect size revisited: Simplified and practical approaches in pre-clinical, clinical and laboratory studies. *Biochem. Med.* **2020**, *31*, 010502. [CrossRef] [PubMed]
26. Harries, M.; Wales, N.S. *Splice-2 Comparative Evaluation: Electricity Pricing*; University of New South Wales, School of Computer Science: Sydney, Australia, 1999. Available online: https://cgi.cse.unsw.edu.au/~reports/papers/9905.pdf (accessed on 11 November 2024).
27. De Vito, S.; Massera, E.; Piga, M.; Martinotto, L.; Di Francia, G. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sens. Actuators B Chemical.* **2008**, *129*, 750–757. [CrossRef]
28. Kong, X.; Jiang, X.; Zhang, B.; Yuan, J.; Ge, Z. Latent variable models in the era of industrial big data: Extension and beyond. *Annu. Rev. Control* **2022**, *54*, 167–199. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.