Journal of System Management
2019, Issue 2, pp. 081-106

Online Mean Shift Detection in Multivariate Quality Control Using Boosted Decision Tree learning

Abbas Asadi
Department of Industrial Management,
Science and Research Branch, Islamic Azad University, Tehran, Iran

Yaghoub Farjami
Department of Computer & IT Engineering,
University of Qom, Qom, Iran
(Corresponding Author)
farjami@qom.ac.ir

Received: **01 January 2019** Accepted: **13 March 2019**

Abstract. The rapid development of communication technologies and information and online computers and their usage in processes of the industrial production have facilitated simultaneous monitoring of multiple variables (characteristics) in a process. In this work, we applied boosted decision tree (DT_{boost}) and Monte Carlo simulation to propose an efficient method for detecting in-control and out-of-control states in multivariate control processes.In this work, four classifiers (methods) - $\chi^2_{\bar{X}}$, $\chi^2_{\bar{X}_{\mathrm{new}}}$, DT_{χ^2}, T_c– are used for detecting the process control states. Then, with converting detection results these four classifiers, the boosted decision tree is made and provides the ultimate result as the in-control or the out-of-control states. To show how the proposed model works and the superiority of this method over $\chi^2_{\bar{X}}$, $\chi^2_{\bar{X}_{\mathrm{new}}}$, DT_{χ^2}, and$T_c$methods, we run it on a standardized trivariate normal process. To compare and evaluate the performance of classifiers, we used ARL functions and the evaluation measures including Accuracy (ACC), Sensitivity (TPR), Specificity (SPC), and Precision (PPV). The findings not only showed the superiority of the proposed method over the tradition Chi-square but also confirmed former results on the efficiency of decision tree for rapid detecting of mean shifts in multivariate processes in which data are gathered automatically.

Keywords: Multivariate Quality Control, Mean Shift Detection, Boosted Decision Tree learning, Moving Window.

1. Introduction

In real-world situations, we typically deal with morethan one variable simultaneously. For example, we may want to simultaneously monitor and control both the length and the inside diameter of a pipe, which both must be acceptable for the pipe. However, monitoring and controlling both characteristics separately may not yield an output in which both variables are acceptable (Mitra, 2016). Therefore, multivariate methods are needed when monitoring and controlling several quality characteristics(variables) and take advantage of any relationships among them (Woodall & Montgomery, 2014) Such Statistical Process Control (SPC) problems are often referred to as multivariate SPC (MSPC) problems, which are the main emphasize of the present work. Multivariate control charts are useful tools for monitoring and controlling the quality of a multivariate process or product. These charts are divided into various types based on their characteristics and results. For example, multivariate means control charts provide key information about process variations through determining the mean shift among a number of qualitative characteristics. One idea of natural is to apply a joint monitoring scheme that includes a set of univariate SPC charts, where each control chart is applied for monitoring a single quality characteristic and the joint monitoring scheme gives a signal of process distributional shift when one individual control chart gives a signal at least. In practice, some people prefer this idea because it only has single-variable SPC charts involved (Qiu, 2013). Montgomery and Mastrangelo (1991) showed that using multiple Shewhart univariate control charts in a multivariate quality process will produce misleading results in the simultaneous monitoring of variables. It becomes even worse by an increase in the number of variables.As an alternative, T^2 control chart (Hotteling, 1947), Multivariate Cumulative Sum Control Chart (Crosier, 1988; Pignatiello & Runger, 1990; Woodall & Ncube, 1985), and Multivariate Exponential Weighted Moving Average Chart (Lowry, Woodall, Champ, & Rigdon, 1992), are among the most common methods for monitoring the mean vector in multivariate processes. The most known procedure is the control chart of HotelingT 2 for monitoring the process mean vector, which is a direct analog of the univariate Shewhart x chart" (Montgomery, 2013). Hoteling'sT^{2}statistic _ a generalized form of Student's t statistic _ is a multiple of statistical distance or Mahalanobis (or quadratic distance). The basic assumption that precedes any discussion on distribution characteristics of Hoteling'sT^{2}statistic is that multivariate observations of a random sample follow a normal multivariate population with amean of μ and a covariance matrix ofΣ. Based on this assumption and depending on the environmental situation, T^{2}can take three different density functions; i.e.,β, F, and χ^2. β distribution can be used in phase I, F distribution can be used in phase II, and χ^2 distribution can be used in both phases. For more information about phase I and phase II, see Alt (1985) and Woodall (2000). Assume that a process or a product has p interdependent quality characteristics $X = (x_1, x_2, \ldots, x_p)'$ that must be controlled simultaneously. When the process is in-control, X has a normal distribution with the mean vector $\boldsymbol{\mu}$ and covariance$\boldsymbol{\Sigma}$, and is denoted as follows:

$$X \sim \boldsymbol{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where $\mu = (\mu_1, \mu_2, \ldots, \mu_p)'$ and

$$\Sigma = (\sigma_{11}, \sigma_{12}, \ldots, \sigma_{ip}; \sigma_{21}, \sigma_{22}, \ldots, \sigma_{2p}; \ldots; \sigma_{p1}, \sigma_{p2}, \ldots, \sigma_{pp})'$$

Multivariate normal distribution and its major characteristics are discussed in detail in Tong(2012)and Miller, Vandome, and John (2010).

Since the assumption in this work is that the mean vector is analyzed in phase II of statistical control, μ and Σ have already been determined in phase I and thus are known. Therefore, to detect the mean shift and monitor the process to see if it is in-control, we use the following statistic:

$$T^2 = n(\bar{X} - \mu)' \, \Sigma^{-1}(\bar{X} - \mu) \sim \chi_p^2$$

where $\overline{X} = (\bar{x}_1, \bar{x}_1, \ldots, \bar{x}_p)'$ is the mean of p quality characteristics in samples of size n andχ^2 depends only on p, the number of quality characteristics. With the α-level known (α is the error signal on the chart or the suitable level of significance. For example, $\alpha = 0.005$ or $\alpha = 0.001$), the control limit of the chart is determined from the above equation. The control chart of χ^2 has a lower limit of zero and an upper limit of $\mathrm{UCL} = \chi^2_{\alpha,p}$ (the αth upper percentile of χ^2 distribution with p degrees of freedom). If at least one of the means is shifted to a new (out of control) value, then it is expected that $\chi^2_0 > \chi^2_{\alpha,p}$ and, consequently, the process is declared out of control. Otherwise, if $\chi^2_0 < \chi^2_{\alpha,p}$, it is concluded that the process is in-control.

2. Literature review

Additional methods and more details on MSPC can be found in Alt (1985), Mason, Tracy, and Young (1995), Runger, Alt, and Montgomery (1996), Doganaksoy, Faltin, and Tucker (1991), Jackson (1956, 1959, 1985, 1991), Hawkins (1991, 1993), Fuchs and Benjamini (1994), and Fuchs and Kennel (1998). Some other reviews,comparisons,and works in MSPC literature can be found in Wade and Woodall (1993), Wierda (1994), Lowry and Montgomery (1995), Umit and Aricigil (2001), Venkatasubramanian et al. (2003), Bersimis, Psarakis, and Panaretos (2007), Frisén (2011), Dhini and Surjandari (2016), Margavio and Conerly (1995), Sullivan and Woodall (1996), Mason et al. (1997), and Puig and Perrer (2014). In recent years, the development and use of information and communication technologies(ICT) in manufactur, such as Computer Integrated Manufacturing (CIM), modern Data Acquisition (DAQ) tool, and online computers, has made simultaneous monitoring of multiple quality characteristics easier and more practical. Moreover, the development of soft computing technology has enabled researchers to employ machine learning algorithms such as neural networks (NN) and decision trees (DT) to control multivariate statistical processes. A review of NN technique in MSPC can be found in Zorriassatine and Tannok (1998), Psarkis (2011), and Atashgar (2015). Following the DT learning techniques success in the single-variable SPC process monitoring (Guh, 2005; Guh & Shiue, 2005), many researchers have been appealed in applying these techniques in multivariate SPC. Guh and Shiue (2008), HE and Xio (2011), He, He, and Wang (2013), He et al. (2013) and Jiang and Song (2017) have proposed methods for monitoring the multivariate processes and detecting failure through DT techniques. Guh and Shiue (2008) used one classifier for both process monitoring and failure detection. HE and Xio (2011) used one classifier for monitoring and p DT classifiers for failure detection (p is the number of variables). Elsewhere, HE et al. (2013a) used one classifier for monitoring and another for failure detection. In another work, HE et al. (2013b) used 2p+1 classifiers where one classifier was for process monitoring, p classifiers were for detecting a mean shift, and p classifiers were for detecting variance shifts. Finally, Jiang and Song (2017) proposed "an ensemble method based on bagging and decision tree to resolve the problem of diagnosing out-of-control signals in a multivariate SPC"(MSPC).The above studies have mainly used two-variable processes with normal distribution and Average Run Length (ARL) and Classification Accuracy (CCR) as a common measure for evaluation and comparison of their methods performance with those of others. All these studies have shown the superiority of DT compared with traditional methods and neural networks in detecting the mean shift and causes of process violation of in-control situation. In general, these algorithms can handle requirements of the online multivariate quality control, and can automatically and quickly detect the causes of the out-of-control situations. In this work, unlike the majority of DT techniques in MSPC literature, we utilize a boosted decision tree for quick detection of the mean vector shift of a trivariate normal distribution process against χ^2 method. We also utilize other measures like Recall or sensitivity (TPR), specificity (SPC), and precision (PPV) for evaluating the performance of the proposed model, in addition to ARL and ACC. This method can adequately detect small, medium, and large shifts as $\pm k_i \sigma_i$, where$0.25 \leq k_i \leq 3.0$, which has received little attention in previous research. The remainder of this paper is organized as follows. The proposed model, which is based on a boosted decision tree for online mean shift detection in multivariate quality control charts, is introduced in Section 2. A normal standard trivariate example is provided to support the working of the proposed model in Section 3. Finally, Section 4 discusses the results and findings and also implications for future research.

3. Method

The purpose of the present work is to develop a model based on boosted DT so that it can work online to quickly and automatically detect mean shifts in multivariate production processes. In this model, the assumption is that when the process is in-control, quality characteristics follow a multivariate normal distribution, whereμ_0 and $\Sigma_0$from phase I of quality control have already been known from the dataset of past observations. The covariance matrix is also assumed to be constant. Our proposed model for detecting whether a multivariate process, either in-control or out-of-control, including model training and evaluation steps. It includes a boosted classified decision tree (DT boost) based on four classifiers $\mathrm{DT_c}$, DT_{χ^2}, $\chi^2_{\bar{X}}$, and $\chi^2_{X_{\mathrm{new}}}$. $\mathrm{DT_c}$ is the classification decision tree for directly detecting the process control state and DT_{χ^2} is the regression decision tree for estimating χ^2_μ, where $\chi^2_{\bar{X}}$ and $\chi^2_{X_{\mathrm{new}}}$ are traditional statistical methods for detecting the in and out control state of multivariate processes. Here, $\chi^2_{\bar{X}}$ is the Mahalanobis distance and $\chi^2_{X_{\mathrm{new}}}$ is the observation of a sample with a large Mahalanobis distance from the desirable mean. During process monitoring, $\chi^2_{X_{\mathrm{new}}}$ calculates χ^2 when a new observation enters the control model. By combining the results of the above four methods, $\mathrm{DT_{boost}}$ is created in order to determine the ultimate result as “in-control” or “out-of-control” situation.

Assuming that the process follows a multivariate normal distribution, we use Monte Carlo simulation to generate the required data that show the mean shifts. Before introducing these data to train decision tree model, we implement a pre-processing by passing them through a linear transformation to transform them to normal standard data with amean of zero and a variance of unity. In this regard, we employ t For this purpose, we use the following equation and transform the data from $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{X} = (x_1, x_2, \cdots, x_p)$ to$\boldsymbol{X} \sim \mathcal{N}_{\boldsymbol{p}}(\mathbf{0}, \boldsymbol{\Sigma})$:

$$x_i \leftarrow (x_i - \mu_i)/\sigma_{ii}$$

where $\sigma_{ii} = 1, \sigma_{ij} = \rho_{ij}, \delta = \chi^2(\boldsymbol{X}) \coloneqq \mathbf{X'\Sigma^{-1}}\boldsymbol{X}$. For more information about normalization methods of multivariate data, see (Koch, 2013) and (Rencher & Christensen, 2012).

In the standard state, $\mu = 0$indicatesthat the process is in-control and there is no shift of mean. However, in case of a shift in the mean from the desirable state zero to $\mu \neq 0$, then $X \sim \mathcal{N}_p(\mu, \Sigma)$ and the mean shift vector can be written as $\mu = \left(k_1, \ldots, k_i, \ldots, k_p\right)$. Here,$k_i$ shows the size of a shift or changes in the mean vector, assuming that $-3 < k_i < +3$. The changes in each of the mean vector elements are created in the training set with a distance of 0.25,which are then inserted into the model by affecting the mean vector. To specify the control state during training, we use $(\chi^2 \leq 0.5)$, which suggests thatonly shifts that are greater than 0.5 $(\chi^2 > 0.5)$ are considered as the out-of-control state.

To detect mean shifts during training, we used the moving window method for sequence observations. In this approach, data are inserted into the model as datasets of a specified size, i.e. window size. These datasets that are sequences of observations are known as identification moving window and are inserted into the model, assuming that they are observation data from a real process. In an identification moving window, it is assumed that the process is initially in-control; in the next step, a permanent change is applied to the observations. The window moves forward at each step, providing new observations to the model. In other words, the window plays the role of a mask for the observations(Atashgar, 2015). The advantage of the moving window is that it not only provides new observations to the model but also previous records of recent observations and thus gives a better understanding of the process situation and their change. Therefore, moving window generally outperforms traditional control charts that only use current observations.For each observation at time t, we measure the p process variables $X_t = (x_{1t}, \ldots, x_{pt})$ and the distance of the observation from the desirable state δ_t, which is shown as $(x_{1t}, \ldots, x_{pt}, \delta_t) = (X_t, \delta_t)$. A moving window with a length of w is a sequence of observations as follows:

$$W_t = [(X_1, \delta_1), \cdots (X_w, \delta_w)]$$
$$= \left[\left(x_{1,1}, x_{1,2}, \ldots, x_{1,p}, \delta_1\right), \left(x_{2,1}, x_{2,2}, \ldots, x_{2,p}, \delta_2\right), \ldots, \left(x_{w,1}, x_{w,2}, \ldots, x_{w,p}, \delta_w\right)\right]$$

where $\delta_i = (X_i - \mu)'\Sigma^{-1}(X_i - \mu)$. A moving window with the length of w has $(p + 1) * w$data, where pis the number of variables.

Since detecting the mean shift is harder for smaller δvalues, the weight of each sample is set as follows:

$$\text{weight}\big((x_1,\delta_1),(x_2,\delta_2),\ldots(x_{\mathrm{w}},\delta_{\mathrm{w}})\big) = 1/(1+\chi_\mu^2)$$

The above weights are defined so that more priority is given to small shifts detection in training the decision tree. They also prevent over fitting.

Learning is usually defined as "acquiring new or alteration of existing knowledge or behavior". The procedures applied to solve learning problems are called learning algorithms or learning machines or just learners (Grąbczewski, 2013, p. 2). Using inductive learning machines is one way of extracting knowledge from large volume data. These machines formulate the extracted knowledge as either decision rules or decision tree. This formulation facilitates understanding and interpreting of the classification knowledge. The machine learning technique for extracting a decision tree from data is called decision tree learning. Decision trees are a method of efficient nonparametric that can be used for both classification and regression tasks that called predictive. They are hierarchical data structures for supervised learning whereby the input space is split into local regions in order to predict the dependent variable. The decision trees consist of nodes, branches, and leaves (Barros, de Carvalho, & Freitas, 2015). When the output of a decision tree is a discrete set, it is called a classification tree while the output is a real number, it is called a regression tree (Rokach & Maimon, 2015). There are various learning algorithms in decision tree literature, such as Iterative Dichotomiser 3 (ID3), C4.5, Classification and Regression Tree(CART), Chi-squared Automatic Interaction Detector (CHAID),and C4.5. However, we do not intend to discuss these algorithms here, as detailed discussions on this area of soft computing are already available. For more information on this matter, see (Gupta, Rawat, Jain, Arora, & Dhami, 2017) and Grąbczewski (2013). In multivariate processes for training decision trees, we first record observations from the process during control based on the assumption that Σ is known and constant and $\mu = 0$ and $\sigma_{ii} = 1$ and $\sigma_{ij} = \sigma_{ji} = \rho_{ij}$are correlation coefficients and $-1 < \sigma_{ij} < +1$.

Decision trees used in this study include two trees. The classification tree ($\mathrm{DT_c}$) and regression tree (DT_{χ^2}), where $\mathrm{DT_c}$is used to determine the control state and $\mathrm{DT}_{\chi^2}$is used to estimateχ^2. To execute the system, we used fit tree and firrtree functions in MATLAB® that are used for building and training decision trees (Chan, 2017) and (Mathworks. Click Here. Decision Trees - MATLAB & Simulink.

http://www.mathworks.com/help/stats/decision-trees.html. Updated March 28, 2017. Accessed March 28, 2017.).Setting and implementation steps of $\mathrm{DT_c}$and $\mathrm{DT_{\chi^2}}$ learning algorithms are presented in detail in the following pseudo-codes:

```
1: Input $\boldsymbol{\mu} = (\mu_1, \mu_2, \cdots, \mu_p), \boldsymbol{\Sigma} = (\sigma_{ij})_{p\times p}, \alpha$;\\ $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{X} = (x_1, x_2, \cdots, x_p)$
2: Normalization, $x_i \leftarrow (x_i - \mu_i)/\sigma_{ii}$;\\ hence $\boldsymbol{X} \sim \mathcal{N}_{\boldsymbol{p}}(\boldsymbol{0}, \boldsymbol{\Sigma})$; $\sigma_{ii} = 1, \sigma_{ij} = \rho_{ij}, \delta = \chi^2(\boldsymbol{X}) \coloneqq \mathbf{X}'\boldsymbol{\Sigma}^{-1}\boldsymbol{X}$
3: Set lattice of shifted means $\boldsymbol{\mu}_{i_1 i_2 \cdots i_p} := (\mu_{i_1}, \mu_{i_2}, \cdots \mu_{i_p}), \mu_{i_j} = -3, -2.75, \cdots, -0.25, 0, +0.25, \cdots + 2.75, +3.00$;
4:Define function $(A < B) \coloneqq$ "In control" if $A < B$ else "Out of control";\\A useful notation as in Matlab syntax.
5: $L_{\boldsymbol{i_1 i_2 \cdots i_p}} := \left(\chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right) < 0.5\right)$;\\Label for control state of the process based on the shifted mean.
6: $\omega_{i_1 i_2 \cdots i_p} \coloneqq 1/\left(1 + \chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right)\right)$; \\more weights for small shifts in decision trees as they are harder to detect.
7: For each $\boldsymbol{\mu}_{\boldsymbol{i_1 i_2 \cdots i_p}}$ do \\ for every mean shift scenario of type $\boldsymbol{\mu} = \boldsymbol{0} \rightarrow \boldsymbol{\mu}_{\boldsymbol{i_1 i_2 \cdots i_p}}$construct new observation window.
7.1:Generate $w$ random vectors $\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p}, \boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \cdots \boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p} \sim \boldsymbol{\mathcal{N}}_{\boldsymbol{p}}(\boldsymbol{0}, \boldsymbol{\Sigma})$; \\ process is in control before running out.
7.2: Set monitoring window $\boldsymbol{W}^0_{i_1 i_2 \cdots i_p} \coloneqq \left[\left(\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p}, \delta^{(1)}_{i_1 i_2 \cdots i_p}\right), \left(\boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \delta^{(2)}_{i_1 i_2 \cdots i_p}\right), \cdots \left(\boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p}, \delta^{(w)}_{i_1 i_2 \cdots i_p}\right)\right]$ where $\delta^{(j)}_{i_1 i_2 \cdots i_p} \coloneqq \chi^2\left(\boldsymbol{X}^{(j)}_{i_1 i_2 \cdots i_p}\right)$.
7.3: For $k = 1$ to $k = w$; \\ insert into monitoring window the new observation from shifted mean process $\boldsymbol{\mathcal{N}}_p\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}, \boldsymbol{\Sigma}\right)$.
7.3.1: Generate new random vector $\boldsymbol{X}^{(\mathrm{new})}_{i_1 i_2 \cdots i_p} \sim \boldsymbol{\mathcal{N}}_p\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}, \boldsymbol{\Sigma}\right), \delta^{(\mathrm{new})}_{i_1 i_2 \cdots i_p} \coloneqq \chi^2\left(\boldsymbol{X}^{(\mathrm{new})}_{i_1 i_2 \cdots i_p}\right)$;
```

```
7.3.2: Set new monitoring window
    $\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p} \coloneqq \left[\left(\boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \delta^{(2)}_{i_1 i_2 \cdots i_p}\right), \left(\boldsymbol{X}^{(3)}_{i_1 i_2 \cdots i_p}, \delta^{(3)}_{i_1 i_2 \cdots i_p}\right), \cdots \left(\boldsymbol{X}^{(\text{new})}_{i_1 i_2 \cdots i_p}, \delta^{(\text{new})}_{i_1 i_2 \cdots i_p}\right)\right]$; \\ slide monitoring window one step forward.
7.3.3: Compute sample mean
    vector$\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p} \coloneqq (\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p} + \boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p} \cdots + \boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p})/w$ , $\overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p} \coloneqq \chi^2\left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}\right)$;
8: Learning set of observation windows $\left[\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, L_{i_1 i_2 \cdots i_p}\right]$; \\ Training data for classification tree $\boldsymbol{DT}_c$ to detect $L_{i_1 i_2 \cdots i_p}$, control state of the process.
9: Learning set of observation windows
    $\left[\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, \left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}, \ \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p}\right), \chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right)\right]$; \\ Training data for regression tree $\boldsymbol{DT}_{\chi^2}$ to estimate $\chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right)$.
10:
    $\boldsymbol{DT}_c \coloneqq$ ClassificationTree$\left(\text{input} = \boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, \left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}, \ \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p}\right), \text{output} = L_{i_1 i_2 \cdots i_p}; \text{weight} = \omega_{i_1 i_2 \cdots i_p}\right)$; Train Classification Tree for detecting control state of the process.
11:
    $\boldsymbol{DT}_{\chi^2} \coloneqq$ RegressionTree$\left(\text{input} = \left[\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, \left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}, \ \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p}\right)\right], \text{output} = \chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right); \text{weight} = \omega_{i_1 i_2 \cdots i_p}\right)$; Train Regression Tree for estimating $\chi^2\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}\right)$.
```

After running learning algorithms of $\mathrm{DT_c}$ and DT_{χ^2} (the above pseudo-codes), the decision trees are ready for implementation. Now, their performance must be evaluated against traditional methods, i.e. $\chi^2_{\overline{\mathrm{X}}} \coloneqq \chi^2(\overline{\mathrm{X}})$and $\chi^2_{\mathrm{X_{new}}} \coloneqq \chi^2(\mathrm{X_{new}})$. For this purpose, we use Monte Carlo simulation method. The measure used for evaluation of the four classification methods is Average Run Length (ARL). By definition, the ARL is "the average number of points that must be plotted before a point indicates an out-of-control condition"(Montgomery,2013). ARL0 shows that the process is in-control, while ARL1 shows that the process is out of control. At the beginning of training and testing, the ARL of all four methods is reset to zero, which is incremented one at a time by every progress stage. There are various methods for detecting an out-of-control state. However, some of these methods detect an out-of-control state wrongly while the process is really in-control; i.e., they have a small ARL0. On the other hand, some methods do not show an out-of-control state while the process is out-of-control; i.e., they have long ARL1.Unfortunately, few processes monitoring methods and measures can quickly detect true out-of-controls while at the same time, avoid false detection of the out-of-control situation when the process is really in control. For example, the χ^2-based method has the advantage of long ARL0, but it has also a long ARL1. The boosted decision tree, which will be introduced in the following sections, has a short ARL1 while benefitting from a fairly longer ARL0 compared to traditional statistical methods likeχ^2. To enhance the above methods, we create a boosted classification decision tree ($\mathrm{DT_{boost}}$) that combines detection results of $\mathrm{DT_c}, \mathrm{DT}_{\chi^2}, \chi^2_{\bar{X}}, \chi\chi^2_{\mathrm{new}}$and provides the ultimate result as the in-control or the out-of-control state. The boosting classifiers idea was presented for the first time by Freund and Schapire (1995, 1996; 1999). They have proven some important properties justifying the solutions. In general, boosting is a method of converting "weak" learning algorithm to a "strong "algorithm with an arbitrarily high accuracy(Grąbczewski,2013) and includes various methods such as AdaBoost,Gradient Boosting, and XG Boostand so on (Schapire & Freund, 2012).

The following pseudo-codes present implementation of the prepared and boosted decision tree algorithm for detecting the process control states using four classifiers (methods):

```
12: For each $\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}$ do  \\ for every mean shift scenario of type
    $\boldsymbol{\mu} = \mathbf{0} \rightarrow \boldsymbol{\mu}_{i_1 i_2 \cdots i_p}$construct new observation window.
12.1:Generate $w$ random vectors $\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p}, \boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \cdots \boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p} \sim \boldsymbol{\mathcal{N}_p}(\mathbf{0}, \boldsymbol{\Sigma})$;
    \\ process is in control before running out.
12.2: Set monitoring window
    $\boldsymbol{W}^0_{i_1 i_2 \cdots i_p} :=$
    $\left[\left(\boldsymbol{X}^{(1)}_{i_1i_2\cdots i_p},\delta^{(1)}_{i_1i_2\cdots i_p}\right),\left(\boldsymbol{X}^{(2)}_{i_1i_2\cdots i_p},\delta^{(2)}_{i_1i_2\cdots i_p}\right),\cdots\left(\boldsymbol{X}^{(w)}_{i_1i_2\cdots i_p},\delta^{(w)}_{i_1i_2\cdots i_p}\right)\right]$ where
    $\delta^{(j)}_{i_1i_2\cdots i_p} \coloneqq \chi^2\left(\boldsymbol{X}^{(j)}_{i_1i_2\cdots i_p}\right)$.
12.3: Set $ARL_{\boldsymbol{DT}_{\chi^2}} \coloneqq 0,\ ARL_{\boldsymbol{DT}_c} \coloneqq 0,\ ARL_{\chi^2_{\overline{\boldsymbol{X}}}} \coloneqq 0, ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}} \coloneqq 0$; \\
    Average Run Length of the four method is reset to zero
12.4: while (
    $\boldsymbol{DT}_{\chi^2} < \chi^2_{\boldsymbol{\alpha},p} OR\ \boldsymbol{DT}_c =$ incontrol $OR\ \chi^2_{\overline{\boldsymbol{X}}} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right) OR\ \chi^2_{\boldsymbol{X}_{\text{new}}} <$
    $\chi^2_{\boldsymbol{\alpha},p}; k++)$ \\ while the four criteria do not detect process run out
    12.4.1: Generate new random vector
        $\boldsymbol{X}^{(\text{new})}_{i_1i_2\cdots i_p} \sim \boldsymbol{\mathcal{N}}_p\left(\boldsymbol{\mu}_{i_1i_2\cdots i_p},\boldsymbol{\Sigma}\right),\delta^{(\text{new})}_{i_1i_2\cdots i_p} \coloneqq \chi^2\left(\boldsymbol{X}^{(\text{new})}_{i_1i_2\cdots i_p}\right)$;
    12.4.2: Set new monitoring window
        $\boldsymbol{W}^k_{i_1i_2\cdots i_p} \coloneqq$
        $\left[\left(\boldsymbol{X}^{(2)}_{i_1i_2\cdots i_p},\delta^{(2)}_{i_1i_2\cdots i_p}\right),\left(\boldsymbol{X}^{(3)}_{i_1i_2\cdots i_p},\delta^{(3)}_{i_1i_2\cdots i_p}\right),\cdots\left(\boldsymbol{X}^{(\text{new})}_{i_1i_2\cdots i_p},\delta^{(\text{new})}_{i_1i_2\cdots i_p}\right)\right]$; \\ slide
        monitoring window one step forward.
    12.4.3: Compute sample mean
        vector$\overline{\boldsymbol{X}}^k_{i_1i_2\cdots i_p} \coloneqq (\boldsymbol{X}^{(1)}_{i_1i_2\cdots i_p} + \boldsymbol{X}^{(2)}_{i_1i_2\cdots i_p} \cdots + \boldsymbol{X}^{(w)}_{i_1i_2\cdots i_p})/w\,,\ \overline{\delta}^{(k)}_{i_1i_2\cdots i_p} \coloneqq$
        $\chi^2\left(\overline{\boldsymbol{X}}^k_{i_1i_2\cdots i_p}\right)$;
    12.4.4: If $\boldsymbol{DT}_c \coloneqq \boldsymbol{DT}_c\left[\boldsymbol{W}^k_{i_1i_2\cdots i_p},\left(\overline{\boldsymbol{X}}^k_{i_1i_2\cdots i_p},\ \overline{\delta}^{(k)}_{i_1i_2\cdots i_p}\right)\right] =$ incontrol then
        $ARL_{\boldsymbol{DT}_c}++$;
    12.4.5: If $\boldsymbol{DT}_{\chi^2} \coloneqq \boldsymbol{DT}_{\chi^2}\left[\boldsymbol{W}^k_{i_1i_2\cdots i_p},\left(\overline{\boldsymbol{X}}^k_{i_1i_2\cdots i_p},\ \overline{\delta}^{(k)}_{i_1i_2\cdots i_p}\right)\right] < \chi^2_{\boldsymbol{\alpha},p}$ then
        $ARL_{\boldsymbol{DT}_{\chi^2}}++$;
    12.4.6: If $\chi^2_{\overline{\boldsymbol{X}}} \coloneqq \chi^2\left[\overline{\boldsymbol{X}}^k_{i_1i_2\cdots i_p}\right] = \overline{\delta}^{(k)}_{i_1i_2\cdots i_p} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right)$ then $ARL_{\chi^2_{\overline{\boldsymbol{X}}}}++$;
    12.4.7: If $\chi^2_{\boldsymbol{X}_{\text{new}}} \coloneqq \chi^2\left[\boldsymbol{X}^{(\text{new})}_{i_1i_2\cdots i_p}\right] = \delta^{(\text{new})}_{i_1i_2\cdots i_p} < \chi^2_{\boldsymbol{\alpha},p}$ then $ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}}++$;
    12.4.8: $\boldsymbol{P}\left(i_1i_2\cdots i_p,k\right) \coloneqq \left[\boldsymbol{DT}_c,\left(\boldsymbol{DT}_{\chi^2} < \chi^2_{\boldsymbol{\alpha},p}\right),\left(\chi^2_{\overline{\boldsymbol{X}}} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right)\right),\left(\chi^2_{\boldsymbol{X}_{\text{new}}} <\right.\right.$
        $\left.\left.\chi^2_{\boldsymbol{\alpha},p}\right),L_{i_1i_2\cdots i_p}\right]$;\\ record detection performance of the four measures.
    12.4.9: $\boldsymbol{Q}\left(i_1i_2\cdots i_p,k\right) \coloneqq \left[\boldsymbol{DT}_c,\boldsymbol{DT}_{\chi^2},\chi^2_{\overline{\boldsymbol{X}}},\chi^2_{\boldsymbol{X}_{\text{new}}},L_{i_1i_2\cdots i_p}\right]$; \\ record values
        of the four measures for constructing boosted tree.
12.5: $\mathbf{ARL}\left(i_1i_2\cdots i_p\right) \coloneqq \left[ARL_{\boldsymbol{DT}_c},ARL_{\boldsymbol{DT}_{\chi^2}},ARL_{\chi^2_{\overline{\boldsymbol{X}}}},ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}}\right]$\\ record the
    ARL of the four measures for each shifted mean.
```

```
13: Learning set of observations $\boldsymbol{\mathcal{D}} \coloneqq [\boldsymbol{Q}(i_1 i_2 \cdots i_p, k)]$; \\ Training data
    for boosted tree constructed $\boldsymbol{DT}_{\chi^2}, \boldsymbol{DT}_c, \chi^2_{\bar{\boldsymbol{X}}}, \chi^2_{\boldsymbol{X}_{\text{new}}}$ to learn control state
    of the process.
14:
    $\boldsymbol{DT}_{\text{boost}} \coloneqq$
    $\text{ClassificationTree}\Big(\text{input} = \big[\boldsymbol{DT}_c(i_1 i_2 \cdots i_p, k), \boldsymbol{DT}_{\chi^2}(i_1 i_2 \cdots i_p, k),$
    $\chi^2_{\bar{\boldsymbol{X}}}(i_1 i_2 \cdots i_p, k), \chi^2_{\boldsymbol{X}_{\text{new}}}(i_1 i_2 \cdots i_p, k)\big]; \text{weight} = \omega_{i_1 i_2 \cdots i_p};\ \text{output} =$
    $L_{i_1 i_2 \cdots i_p}\Big)$; Train Boosted Classification Tree for detecting control state
    of the process.
```

Many measures are available for evaluating a quality control method (and generally a classification method). Some of these measures are Accuracy (ACC), Sensitivity (TPR), Specificity (SPC), and Precision (PPV) (Rokach and Maimon,2015).

Typically, when a control method performs well at the in-control state (long ARL0), it performs weakly when the process becomes out of control, and vice versa. In this connection, it is difficult to find a method that has a long ARL0 and a short ARL1. Performance evaluation measures like ARL cannot be defined as target values in training a decision tree. These measures (including ARL) can only be estimated through Monte Carlo simulation after implementing the detection method. In most application of control sates detection, the CCR is not a sufficient measure for evaluation, especially when the number of generated in-control testing samples is much more than the number of the out-of-control ones. This behavior is due to the fact that a control system with a tendency toward the in-control detection (i.e., to always announce a process in-control)shows a high CCR, while it may act weakly in face of the out-of-control state. Therefore, in these cases, the sensitivity and specificity and Precision measures can be used as another possibility or choice to the accuracy measures(Rokach and Maimon,2015).In this work, we used Monte Carlo simulation to evaluate the performance and used PPV, TPR, SPC, and ACC to evaluate the obtained results. In addition, ARL functions were also calculated for each method. The following pseudo-codes show the algorithm for calculating the measures.

```
15: For each $\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}$ do \\ for every mean shift scenario of type
    $\boldsymbol{\mu} = \mathbf{0} \to \boldsymbol{\mu}_{i_1 i_2 \cdots i_p}$ construct new observation window.
15.1: Generate $w$ random vectors $\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p}, \boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \cdots \boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p} \sim \boldsymbol{\mathcal{N}_p}(\mathbf{0}, \boldsymbol{\Sigma})$;
    \\ process is in control before running out.
15.2: Set monitoring window
    $\boldsymbol{W}^{0}_{i_1 i_2 \cdots i_p} := \left[\left(\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p}, \delta^{(1)}_{i_1 i_2 \cdots i_p}\right), \left(\boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \delta^{(2)}_{i_1 i_2 \cdots i_p}\right), \cdots \left(\boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p}, \delta^{(w)}_{i_1 i_2 \cdots i_p}\right)\right]$ where
    $\delta^{(j)}_{i_1 i_2 \cdots i_p} := \chi^2\left(\boldsymbol{X}^{(j)}_{i_1 i_2 \cdots i_p}\right)$.
15.3: Set $ARL_{\boldsymbol{DT}_{\text{boost}}} := 0, ARL_{\boldsymbol{DT}_{\chi^2}} := 0$, $ARL_{\boldsymbol{DT_c}} := 0$, $ARL_{\chi^2_{\overline{\boldsymbol{X}}}} := 0, ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}} := 0$; \\ Average Run Length of the five method is reset
    to zero
15.4: while
    $(\boldsymbol{DT}_{\text{boost}} = \text{incontrol OR } \boldsymbol{DT}_{\chi^2} < \chi^2_{\boldsymbol{\alpha},p} OR\ \boldsymbol{DT}_c = \text{incontrol } OR\ \chi^2_{\overline{\boldsymbol{X}}} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right) OR\ \chi^2_{\boldsymbol{X}_{\text{new}}} < \chi^2_{\boldsymbol{\alpha},p}; k++)$ \\ while the five criteria do not
    detect process run out
15.4.1: Generate new random vector
    $\boldsymbol{X}^{(\text{new})}_{i_1 i_2 \cdots i_p} \sim \boldsymbol{\mathcal{N}}_p\left(\boldsymbol{\mu}_{i_1 i_2 \cdots i_p}, \boldsymbol{\Sigma}\right), \delta^{(\text{new})}_{i_1 i_2 \cdots i_p} := \chi^2\left(\boldsymbol{X}^{(\text{new})}_{i_1 i_2 \cdots i_p}\right)$;
15.4.2: Set new monitoring window
    $\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p} := \left[\left(\boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p}, \delta^{(2)}_{i_1 i_2 \cdots i_p}\right), \left(\boldsymbol{X}^{(3)}_{i_1 i_2 \cdots i_p}, \delta^{(3)}_{i_1 i_2 \cdots i_p}\right), \cdots \left(\boldsymbol{X}^{(\text{new})}_{i_1 i_2 \cdots i_p}, \delta^{(\text{new})}_{i_1 i_2 \cdots i_p}\right)\right]$; \\ slide
    monitoring window one step forward.
15.4.3: Compute sample mean
    vector $\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p} := (\boldsymbol{X}^{(1)}_{i_1 i_2 \cdots i_p} + \boldsymbol{X}^{(2)}_{i_1 i_2 \cdots i_p} \cdots + \boldsymbol{X}^{(w)}_{i_1 i_2 \cdots i_p})/w$, $\overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p} := \chi^2\left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}\right)$;
15.4.4: If $\boldsymbol{DT}_c := \boldsymbol{DT}_c\left[\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, \left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}, \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p}\right)\right]$ = incontrol then
    $ARL_{\boldsymbol{DT_c}}++$;
15.4.5: If $\boldsymbol{DT}_{\chi^2} := \boldsymbol{DT}_{\chi^2}\left[\boldsymbol{W}^{k}_{i_1 i_2 \cdots i_p}, \left(\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}, \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p}\right)\right] < \chi^2_{\boldsymbol{\alpha},p}$ then
    $ARL_{\boldsymbol{DT}_{\chi^2}}++$;
15.4.6: If $\chi^2_{\overline{\boldsymbol{X}}} := \chi^2\left[\overline{\boldsymbol{X}}^{k}_{i_1 i_2 \cdots i_p}\right] = \overline{\delta}^{(k)}_{i_1 i_2 \cdots i_p} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right)$ then $ARL_{\chi^2_{\overline{\boldsymbol{X}}}}++$;
```

```
15.4.7: If $\chi^2_{\boldsymbol{X}_{\text{new}}} \coloneqq \chi^2\left[\boldsymbol{X}^{(\text{new})}_{i_1 i_2 \cdots i_p}\right] = \delta^{(\text{new})}_{i_1 i_2 \cdots i_p} < \chi^2_{\boldsymbol{\alpha},p}$ then $ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}} ++;$
15.4.8: If $\boldsymbol{DT}_{\text{boost}} \coloneqq \boldsymbol{DT}_{\text{boost}}\left[\boldsymbol{DT}_c, \boldsymbol{DT}_{\chi^2}, \chi^2_{\overline{\boldsymbol{X}}}, \chi^2_{\boldsymbol{X}_{\text{new}}}\right] = \text{incontrol}$ then
    $ARL_{\boldsymbol{DT}_{\text{boost}}} ++;$
15.4.9:
    $\boldsymbol{P}(i_1 i_2 \cdots i_p, k) \coloneqq$
    $\left[\boldsymbol{DT}_{\text{boost}}, \boldsymbol{DT}_c, \left(\boldsymbol{DT}_{\chi^2} < \chi^2_{\boldsymbol{\alpha},p}\right), \left(\chi^2_{\overline{\boldsymbol{X}}} < \left(\chi^2_{\boldsymbol{\alpha},p}/p\right)\right), \left(\chi^2_{\boldsymbol{X}_{\text{new}}} <\right.\right.$
    $\left.\left.\chi^2_{\boldsymbol{\alpha},p}\right), L_{i_1 i_2 \cdots i_p}\right]$;\\ record detection performance of the five measures.
15.5: $\mathbf{ARL}(i_1 i_2 \cdots i_p) \coloneqq \left[ARL_{\boldsymbol{DT}_{\text{boost}}}, ARL_{\boldsymbol{DT}_c}, ARL_{\boldsymbol{DT}_{\chi^2}}, ARL_{\chi^2_{\overline{\boldsymbol{X}}}}, ARL_{\chi^2_{\boldsymbol{X}_{\text{new}}}}\right]$\\
    record the ARL of the four measures for each shifted mean.
16: $P \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } L_{i_1 i_2 \cdots i_p} = \text{out of control}\}$, \\ real
    out of control states in the process,
17: $N \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } L_{i_1 i_2 \cdots i_p} = \text{in control}\}$, \\ real in
    control states in the process,
18: $TP(\boldsymbol{DT}_{\text{boost}}) \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } \boldsymbol{DT}_{\text{boost}} = L_{i_1 i_2 \cdots i_p} =$
    $\text{out of control}\}$, \\ using 15.4.9
19: $TN(\boldsymbol{DT}_{\text{boost}}) \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } \boldsymbol{DT}_{\text{boost}} = L_{i_1 i_2 \cdots i_p} =$
    $\text{in control}\}$ ,\\ using 15.4.9
20:
    $FP(\boldsymbol{DT}_{\text{boost}}) \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } \boldsymbol{DT}_{\text{boost}} =$
    $\text{out of control} \neq L_{i_1 i_2 \cdots i_p} = \text{in control}\}$,\\ using 15.4.9
21: $FN(\boldsymbol{DT}_{\text{boost}}) \coloneqq \#\{\text{rows in } \boldsymbol{P}(i_1 i_2 \cdots i_p, k) \text{ with } \boldsymbol{DT}_{\text{boost}} = \text{in control} \neq$
    $L_{i_1 i_2 \cdots i_p} = \text{out of control}\}$,\\ using 15.4.9
22: $TPR(\boldsymbol{DT}_{\text{boost}}) \coloneqq \left(TP(\boldsymbol{DT}_{\text{boost}})\right)/P$; \\ true positive rate or
    sensitivity,
23: $SPC(\boldsymbol{DT}_{\text{boost}}) \coloneqq \left(TN(\boldsymbol{DT}_{\text{boost}})\right)/N$; \\ true negative rate or
    specificity,
24: $PPV(\boldsymbol{DT}_{\text{boost}}) \coloneqq \left(TP(\boldsymbol{DT}_{\text{boost}})\right)/\left(TP(\boldsymbol{DT}_{\text{boost}}) + FP(\boldsymbol{DT}_{\text{boost}})\right)$; \\
    positive predictive value or precision,
25: $ACC(\boldsymbol{DT}_{\text{boost}}) \coloneqq \left(TP(\boldsymbol{DT}_{\text{boost}}) + TN(\boldsymbol{DT}_{\text{boost}})\right)/(P + N)$; \\ accuracy,
26: $\cdots$\\ ratios for other $\boldsymbol{DT}_c, \boldsymbol{DT}_{\chi^2}, \chi^2_{\overline{\boldsymbol{X}}}, \chi^2_{\boldsymbol{X}_{\text{new}}}$ computed similarly.
```

We expect that our boosted decision tree (DT_{boost}) outperforms the other four methods.

4. Findings

To show how the proposed model works, we run it on a standardized trivariatenormal process. In this example, values of process parameters are as follows:

$$\boldsymbol{\mu} = [\mu_1, \mu_2, , \mu_3] = [0,0,0],$$
$$\boldsymbol{R} = \begin{bmatrix} 1 & \rho_{12} & \rho_{13} \\ \rho_{21} & 1 & \rho_{23} \\ \rho_{31} & \rho_{32} & 1 \end{bmatrix} = \begin{bmatrix} 1.0000 & -0.06531 & -0.2649 \\ -0.06531 & 1.0000 & 0.96587 \\ -0.2649 & 0.96587 & 1.000 \end{bmatrix}$$

where μ is the mean vector, R is correlation matrix, and$\rho_{ij}$is the correlation coefficient between variables. ρ_{ij}values were selected so that all three variables were strongly correlated. As a result, a situation is created that is difficult for traditional methods to analyze. For the in-control observations, we used a simulation. Several approaches exist for generating the random variables in the trivariate normal process. These approaches have been discussed in detail in the simulation literature. In this example, we generate random samples from a trivariate normal process using mvnrnd function in MATLAB®. Based on the method proposed by Ryan (Ryan, 2011) and methods used by Guh and Shieu (2008), the first type error α is determined so that $\alpha/2p=0.00135$, which is equivalent to 3σ in the single-variable state. If $p = 3$, then $\alpha = 0.0081$; thus, ARL is 124 (= $(1/\alpha)$). The upper limit control for χ^2 is $\mathrm{UCL}_{\chi^2_{\mathrm{x}}} = \mathrm{chi2inv}(1-\alpha, 3) = 11.8$ and $\mathrm{UCL}_{\chi^2_{\bar{\bar{\mathrm{x}}}}} = \mathrm{UCL}_{\chi^2_{\mathrm{x}}}/\mathrm{w} = 0.786$,where $\bar{\mathrm{x}} = (\mathrm{x}_1 + \cdots + \mathrm{x}_{\mathrm{w}})/\mathrm{w}$and w, the observation window length, is 15 (w = 15).

The out-of-control state condition in decision tree training is $\chi^2_{\mu} > \chi^2_{\alpha,3}$, where $\chi^2_{\alpha,3}$ is the upper control limit.

To train the in-control state $\mu = (0, \ldots, 0)$, a total of 5,000 random samples were generated and for each out-of-control state 100 samples were generated (a total of $(12^3) * 100 + 5000$mean vector shifts). Therefore, the number of observation samples for training was$(100 \times (12^3) + 5000) * \mathrm{w}$. We used 5000 * w samples for the in-control and $(100 \times (12^3)) * \mathrm{w}$ samples for each of the mean shift states (w = 15). We began training at point 16, where the first window of the out-of-control observations can be formed. An input to the decision tree is as follows:

$$((x_1, \delta_1), (x_2, \delta_2), \ldots, (x_{15}, \delta_{15}))$$

Classes label were 1 for the out-of-control state and 0 for the in-control state with equal weights. The uniform and priorfunctions inMATLAB® are used for this purpose. To prevent overfitting, there must be at least 40 observations in each leaf (minleaf = 40).

In this step, we train our decision trees and then can begin to test them. During the testing, we also estimate ARL0 and ARL1. To evaluate the performance, we first consider the multivariate process at the in-control state and generate a sample from$\mathcal{N}_3(0, \Sigma)$ with w = 15. From sample 16 onward, we generate random samples from $\mathcal{N}_3(\mu, \Sigma)$ and continue until the first out-of-control detection occurs. By considering an observation window with w = 15 and new observations (from 1 to 124 steps) from three-variable vectors and their χ^2 values, we have:

$$(12 \times 12 \times 12 * 100 + 5000) \times 15 \times 124 \times (3 + 1) = 1322832000$$

To compare χ^2_μ and $\mathrm{DT_{boost}}$, we use ARL.Table 1 shows ARL values for $\chi^2_{\bar{X}}$ and $\mathrm{DT_{boost}}$methods. As mentioned previously, our assumption is that distances, or shifts in each element of the mean vector, are 0.25.The ARL values, which are a function of δ, were obtained for the above two methods($\chi^2_{\bar{X}}$and$\mathrm{DT_{boost}}$) through the simulation. These values are presented in Table 1 and Fig.1. Due to lack of space, only some values are shown in the table.

Table 1. Comparison of ARLs between the $\chi^2_{\bar{X}}$and $\mathrm{DT_{boost}}$methods

$\delta = \chi^2_\mu$	$\mathrm{ARL}_{\chi^2_{\bar{X}}}$	$\mathrm{ARL}_{\mathrm{DT_{boost}}}$
0	127.00	135.36
0.07	112.00	29.00
0.27	75.80	35.80
0.39	33.80	31.00
0.4	33.60	20.60
0.43	32.20	16.80
0.54	23.60	10.20
0.62	16.80	12.20
0.79	11.20	6.20
0.8	15.30	16.40

$\delta = \chi_{\mu}^{2}$	$\mathrm{ARL}_{\chi_{\bar{x}}^{2}}$	$\mathrm{ARL}_{\mathrm{DT}_{\mathrm{boost}}}$
0.83	11.20	7.40
0.86	12.40	12.20
0.92	8.80	8.00
0.94	14.20	10.80
1	12.80	9.50
1.09	15.80	8.60
1.16	11.40	8.60
1.2	13.00	10.80
1.23	10.40	6.60
1.24	13.60	8.20
1.28	12.60	10.00
1.34	11.80	8.40
1.35	11.80	8.80
1.44	12.40	7.40
1.52	8.60	9.00
1.57	10.60	10.00
1.6	8.80	8.80
1.71	9.80	4.60
1.72	6.80	5.40
1.74	10.60	5.60
1.75	8.80	5.80
1.77	9.80	6.90
1.79	11.00	7.80
1.81	10.00	8.20
1.82	9.60	5.80
1.83	9.50	6.50
1.91	8.80	5.20
1.92	9.60	5.80
1.96	8.00	6.60
1.97	6.20	5.40

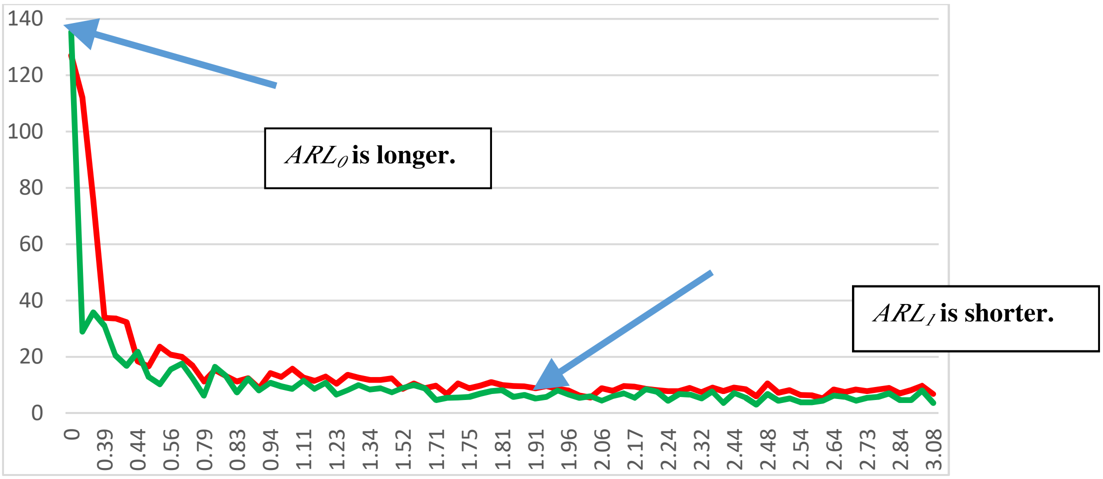

Figure 2. ARL Curves for the $\chi^2_{\bar{X}}$ and DT_{boost}

In Fig. 1, the vertical axis shows ARL values and the horizontal axis shows δ values. It is observed that DT_{boost} has longer in-control ARLs and shorter out-of-control ARLs. This curve and Table 1(less values of ARL in the third column) well demonstrate that boosted decision tree outperforms the chi-square method.Again, the ACC, SPC, TPR, and PPV of each method were measured through the simulation. The results are shown in Table 2.

Table 2. Performance of different classifiers for mean shifts detection

Classifier	Accuracy(ACC)	Sensitivity(TPR)	Specificity(SPC)	Precision(ppv)
$\chi^2_{\bar{X}}$	0.9568	0.8771	0.9918	.09791
DT_{χ^2}	0.9516	0.9274	0.9643	0.9172
$\chi^2_{\bar{X}_{new}}$	0.9618	0.8967	0.9903	0.9760
DT_c	0.9564	0.9217	0.9716	0.9343
DT_{boost}	0.9709	0.9245	0.9912	0.9788

According to Table 2, DT_{boost}outperforms the other methods in terms of accuracy, sensitivity, specificity, and precision. It is also observed that $\chi^2_{\bar{X}}$ and $\chi^2_{\bar{X}_{new}}$ have the weakest performance and the boosted decision tree has an improved performance compared to ordinary trees.

5. Discussion and Conclusions

Soft computing methods are a suitable alternative to traditional multivariate quality control charts in detecting the mean shifts. In the current work, we developed a boosted decision tree and showed that it outperforms theχ^2 control chart.

The performance of the proposed model was evaluated and confirmed, even in detecting small mean shifts, through a trivariate process example and by means of ARL functions and Accuracy, Sensitivity, Specificity, and Precision measures. The findings showed that the proposed method can adapt to the online multivariate quality control systems and can quickly and automatically detect the in-control and out-of-control states. Based on the findings, it is recommended applying the proposed method with non-normal data and comparing itwith other machine learning techniques such as support vector machines(SVM)and neural networks.

References

Alt, F. B. (1985). Multivariate quality control. Encyclopedia of Statistical Science, 6, 110–122.

Atashgar, K. (2015). Monitoring multivariate environments using artificial neural network approach: An overview. Scientia Iranica. Transaction E, Industrial Engineering, 22(6), 2527.

Barros, R. C., de Carvalho, A. C. P. L. F., & Freitas, A. A. (2015). Automatic Design of Decision-Tree Induction Algorithms. Springer International Publishing.

Bersimis, S., Psarakis, S., & Panaretos, J. (2007). Multivariate statistical process control charts: an overview. Quality and Reliability Engineering International, 23(5), 517–543. https://doi.org/doi:10.1002/qre.829

Chan, E. P. (2017). Machine Trading: Deploying Computer Algorithms to Conquer the Markets. Wiley.

Crosier, R. B. (1988). Multivariate Generalizations of Cumulative Sum

Quality-Control Schemes. Technometrics, 30(3), 291–303. https://doi.org/10.1080/00401706.1988.10488402

Dhini, A., & Surjandari, I. (2016). Review on Some Multivariate Statistical Process Control Methods for Process monitoring.

Doganaksoy, N., Faltin, F. W., & Tucker, W. T. (1991). Identification of out of control quality characteristics in a multivariate manufacturing environment. Communications in Statistics - Theory and Methods, 20(9), 2775–2790. https://doi.org/10.1080/03610929108830667

Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In European conference on computational learning theory (pp. 23–37). Springer.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In Icml (Vol. 96, pp. 148–156). Bari, Italy.

Frisén, M. (2011). On multivariate control charts. Production, 21(2), 235–241.

Fuchs, C., & Benjamini, Y. (1994). Multivariate Profile Charts for Statistical Process Control. Technometrics, 36(2), 182–195. https://doi.org/10.1080/00401706.1994.10485765

Fuchs, C., & Kenett, R. S. (1998). Multivariate Quality Control: Theory and Applications. Taylor & Francis.

Grąbczewski, K. (2013). Meta-Learning in Decision Tree Induction. Springer International Publishing.

Guh, R.-S. (2005). A hybrid learning-based model for on-line detection and analysis of control chart patterns. Computers & Industrial Engineering, 49(1), 35–62. https://doi.org/https://doi.org/10.1016/j.cie.2005.03.002

Guh, R.-S., & Shiue, Y.-R. (2005). On-line identification of control chart patterns using self-organizing approaches. International Journal of Production Research, 43(6), 1225–1254. https://doi.org/10.1080/0020754042000268884

Guh, R.-S., & Shiue, Y.-R. (2008). An effective application of decision tree learning for on-line detection of mean shifts in multivariate control charts. Computers & Industrial Engineering, 55(2), 475–493. https://doi.org/https://doi.org/10.1016/j.cie.2008.01.013

Gupta, B., Rawat, A., Jain, A., Arora, A., & Dhami, N. (2017). Analysis of Various Decision Tree Algorithms for Classification in Data Mining. International Journal of Computer Applications, 163(8), 5–9.

Hawkins, D. M. (1991). Multivariate Quality Control Based on Regression-Adiusted Variables. Technometrics, 33(1), 61–75. https://doi.org/10.1080/00401706.1991.10484770

Hawkins, D. M. (1993). Regression Adjustment for Variables in Multivariate Quality Control. Journal of Quality Technology, 25(3), 170–182. https://doi.org/10.1080/00224065.1993.11979451

He, S.-G., He, Z., & Wang, G. A. (2013). Online monitoring and fault identification of mean shifts in bivariate processes using decision tree learning techniques. Journal of Intelligent Manufacturing, 24(1), 25–34. https://doi.org/10.1007/s10845-011-0533-5

He, S., Wang, G. A., Zhang, M., & Cook, D. F. (2013). Multivariate process monitoring and fault identification using multiple decision tree classifiers. International Journal of Production Research, 51(11), 3355–3371. https://doi.org/10.1080/00207543.2013.774474

He, S., & Xiao, C. (2011). Multivariate process monitoring and fault identification model using decision tree learning techniques. In Proceedings of 2011 IEEE International Conference on Intelligence and Security Informatics (pp. 325–330). https://doi.org/10.1109/ISI.2011.5984107

Hotteling, H. (1947). Multivariate Quality Control in Techniques of statistical Analysis, edited by Eisenhart, Hastay and Walls. McGraw-Hill, New York, NY.

Jackson, J. E. (1956). Quality control methods for two related variables. Industrial Quality Control, 12(7), 4–8.

Jackson, J. E. (1959). Quality Control Methods for Several Related Variables. Technometrics, 1(4), 359–377. https://doi.org/10.1080/00401706.1959.10489868

Jackson, J. E. (1985). Multivariate quality control. Communications in Statistics - Theory and Methods, 14(11), 2657–2688. https://doi.org/10.1080/03610928508829069

Jackson, J. E. (1991). A User's Guide to Principal Components (1st ed.).

Wiley.

Jiang, J., & Song, H.-M. (2017). Diagnosis of Out-of-control Signals in Multivariate Statistical Process Control Based on Bagging and Decision Tree. Asian Business Research, 2(2), 1.

Koch, I. (2013). Analysis of Multivariate and High-Dimensional Data. Cambridge University Press.

Lowry, C. A., & Montgomery, D. C. (1995). A review of multivariate control charts. IIE Transactions, 27(6), 800–810. https://doi.org/10.1080/07408179508936797

Lowry, C. A., Woodall, W. H., Champ, C. W., & Rigdon, S. E. (1992). A Multivariate Exponentially Weighted Moving Average Control Chart. Technometrics, 34(1), 46–53. https://doi.org/10.1080/00401706.1992.10485232

Margavio, T. M., & Conerly, M. D. (1995). A comparison of multivariate moving average control charts for the process mean. International Journal of Production Research, 33(5), 1313–1321. https://doi.org/10.1080/00207549508930211

Mason, R. L., Champ, C. W., Tracy, N. D., Wierda, S. J., & Young, J. C. (1997). Assessment of multivariate process control techniques. Journal of Quality Technology, 29(2), 140.

Mason, R. L., Tracy, N. D., & Young, J. C. (1995). Decomposition of T 2 for Multivariate Control Chart Interpretation. Journal of Quality Technology, 27(2), 99–108. https://doi.org/10.1080/00224065.1995.11979573

Miller, F. P., Vandome, A. F., & John, M. B. (2010). Multivariate Normal Distribution. VDM Publishing.

Mitra, A. (2016). Fundamentals of Quality Control and Improvement (4th ed.). Wiley.

Montgomery, D. C. (2013). Introduction to statistical quality control (7th ed.). John Wiley & Sons.

Montgomery, D. C., & Mastrangelo, C. M. (1991). Some Statistical Process Control Methods for Autocorrelated Data. Journal of Quality Technology, 23(3), 179–193. https://doi.org/10.1080/00224065.1991.11979321

Pignatiello, J. J., & Runger, G. C. (1990). Comparisons of Multivariate CUSUM Charts. Journal of Quality Technology, 22(3), 173–186. https://doi.org/10.1080/00224065.1990.11979237

Psarakis, S. (2011). The use of neural networks in statistical process control charts. Quality and Reliability Engineering International, 27(5), 641–650. https://doi.org/doi:10.1002/qre.1227

Qiu, P. (2013). Introduction to Statistical Process Control. CRC Press.

Rencher, A. C., & Christensen, W. F. (2012). Methods of Multivariate Analysis. Wiley.

Rokach, L., & Maimon, O. (2015). Data Mining with Decision Trees: Theory and Applications. World Scientific.

Runger, G. C., Alt, F. B., & Montgomery, D. C. (1996). Contributors to a multivariate statistical process control chart signal. Communications in Statistics - Theory and Methods, 25(10), 2203–2213. https://doi.org/10.1080/03610929608831832

Ryan, T. P. (2011). Statistical Methods for Quality Improvement (3rd ed.). Wiley.

Schapire, R. E., & Freund, Y. (1999). A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence, 14(5), 771–780.

Schapire, R. E., & Freund, Y. (2012). Boosting: Foundations and algorithms. MIT press.

Sullivan, J. H., & Woodall, W. H. (1996). A Comparison of Multivariate Control Charts for Individual Observations. Journal of Quality Technology, 28(4), 398–408. https://doi.org/10.1080/00224065.1996.11979698

Tong, Y. L. (2012). The multivariate normal distribution. Springer Science & Business Media.

Umit, F., & Cigdem, A. (2001). Multivariate Quality Control: a historical perspective. Yilditz Technical University, 54–65.

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., & Yin, K. (2003). A review of process fault detection and diagnosis: Part III: Process history based methods. Computers & Chemical Engineering, 27(3), 327–346. https://doi.org/https://doi.org/10.1016/S0098-1354(02)00162-X

Vidal-Puig, S., & Ferrer, A. (2014). A Comparative Study of Different Methodologies for Fault Diagnosis in Multivariate Quality Control. Communications in Statistics - Simulation and Computation, 43(5), 986–1005. https://doi.org/10.1080/03610918.2012.720745

Wade, M. R., & Woodall, W. H. (1993). A Review and Analysis of Cause-Selecting Control Charts. Journal of Quality Technology, 25(3), 161–169. https://doi.org/10.1080/00224065.1993.11979450

Wierda, S. J. (1994). Multivariate statistical process control—recent results and directions for future research. Statistica Neerlandica, 48(2), 147–168. https://doi.org/doi:10.1111/j.1467-9574.1994.tb01439.x

Woodall, W. H. (2000). Controversies and Contradictions in Statistical Process Control. Journal of Quality Technology, 32(4), 341–350. https://doi.org/10.1080/00224065.2000.11980013

Woodall, W. H., & Montgomery, D. C. (2014). Some Current Directions in the Theory and Application of Statistical Process Monitoring. Journal of Quality Technology, 46(1), 78–94. https://doi.org/10.1080/00224065.2014.11917955

Woodall, W. H., & Ncube, M. M. (1985). Multivariate CUSUM Quality-Control Procedures. Technometrics, 27(3), 285–292. https://doi.org/10.1080/00401706.1985.10488053

Zorriassatine, F., & Tannock, J. D. T. (1998). A review of neural networks for statistical process control. Journal of Intelligent Manufacturing, 9(3), 209–224. https://doi.org/10.1023/a:1008818817588