UNIVERSITÀ DEGLI STUDI

DI MODENA E REGGIO EMILIA

Dottorato di ricerca in Ingegneria dell'Innovazione Industriale

Ciclo XXXII

Management and classification of intermittent products

Candidato: Elia Balugani

Relatore (Tutor): Prof.ssa Bianca Rimini

Coordinatore del Corso di Dottorato: Prof. Franco Zambonelli

Index

Riassunto – italiano

Nell’ingegneria industriale i prodotti intermittenti sono caratterizzati da domanda intermittente, avente poche domande e alta variabilità in caso di domanda positiva. Una tale struttura è tipica di:

- Ricambi.
- Prodotti delle startup.
- Prodotti risultanti da catene di fornitura multilivello.

Questi prodotti richiedono tecniche ad-hoc per controllare le tre principali attività gestionali:

- Previsione della domanda.
- Gestione dell’inventario.
- Classificazione.

Dal lavoro di [1] vari sforzi di ricerca si sono concentrati su questi temi confluendo in un’ampia letteratura [2]. Gli obbiettivi della tesi sono di:

- Progettare nuove tecniche adatte a scenari specifici.
- Applicare algoritmi di machine learning alla soluzione di problemi tradizionali.

Nel Capitolo 5 si analizza uno scenario caratterizzato da serie storiche molto ridotte e domanda molto rara. Modelli caotici sono proposti come alternativa ai tradizionali modelli previsivi per la domanda intermittente. Nei Capitoli 6 e 7 si analizza uno scenario caratterizzato dalla domanda di prodotti deperibili a un sistema tradizionale per la gestione periodica di codici intermittenti [3] è adattato al caso in questione.

Gli algoritmi di machine learning [4] hanno impattato su molti settori della ricerca grazie alla loro capacità di gestire grandi quantità di dati in maniera automatica. Benché la letteratura sulla domanda intermittente si sia già avvantaggiata di queste tecniche in passato è possibile utilizzarle ulteriormente, in unione con metodi simulativi, per affrontare problemi tradizionalmente risolti con altre metodologie.

Nei Capitoli 2 e 3 tecniche supervisionate di machine learning sono usate per affrontare il problema della classificazione di prodotti intermittenti. Un piccolo numero di prodotti è simulato e queste simulazioni sono utilizzate per addestrare diversi algoritmi di machine learning e classificare i prodotti rimanenti senza la necessità di simularli tutti. Nel Capitolo 4 algoritmi di machine learning non supervisionati sono usati per evitare completamente la fase di simulazione.

La tesi è strutturata come segue: il Capitolo 1 contiene un’analisi della letteratura su tre dei principali temi di ricerca nell’ambito della domanda intermittente, la previsione della domanda, la gestione dell’inventario e la classificazione; Il Capitolo 2 sviluppa un metodo di classificazione per la gestione dell’inventario basato su simulazioni, il metodo in questione fa uso di reti neurali e support vector machines; Il Capitolo 3 affronta un problema simile utilizzando alberi decisionali e random forest in quanti più facilmente interpretabili; Il Capitolo 4 propone un’alternativa non supervisionata ai problemi di cui sopra basata sul K-means e sul metodo di Ward; Il Capitolo 5 descrive un metodo previsivo basato su modelli caotici adatto a prodotti caratterizzati da pochissima domanda; Il Capitolo 6 introduce un sistema di gestione dell’inventario periodico studiato per il controllo di prodotti deperibili; Il Capitolo 7 espande l’analisi sperimentale effettuata nel capitolo precedente.

Summary

In industrial engineering an intermittent product is characterized by intermittent demand patterns having rare demand occurrences and high variability when a positive demand takes place. Such a pattern is typical of:

- Spare parts.
- Start-up productions.
- Products resulting from multi-level supply chains.

These products require ad-hoc techniques to deal with the three main items management activities (see Chapter 1):

- Forecasting.
- Inventory management.
- Classification.

From the seminal work of [1] an increasing research effort focused on these themes, resulting in a vast literature [2]. The objectives of this thesis are to:

- Design new techniques to deal with specific scenarios.
- Apply machine learning algorithms to solve traditional problems.

In Chapter 5 a scenario characterized by very limited historical series and very low demand items is analysed. There chaotic models are proposed as an alternative to traditional intermittent demand forecasting methods. In Chapters 6 and 7 a scenario characterized by perishable items is analysed and a traditional intermittent demand periodic inventory system [3] in adjusted to deal with expiring stock.

Machine learning algorithms [4] have impacted many research fields leveraging large quantities of data in an automated fashion. While the intermittent demand literature took advantage of these techniques it is possible to further leverage them, in conjunction with simulation, for problems traditionally tackled by other means.

In Chapter 2 and 3 supervised machine learning algorithms are used to tackle the problem of intermittent items classification. A limited number of items are simulated, these simulations are used to train machine learning algorithms and classify the remaining items without simulating them. In Chapter 4 non-supervised machine learning algorithms are used to skip the simulation phase altogether.

The thesis is structured as follows: Chapter 1 contains a literature review on three main themes in the intermittent demand research, forecasting, inventory management and classification; Chapter 2 outlines a simulation-based classification method for inventory management purposes that leverages neural network and support vector machines algorithms; Chapter 3 tackles a similar classification problem employing the more interpretable decision trees and random forests; Chapter 4 proposes a non-supervised alternative based on K-means and Ward's method; Chapter 5 describes a forecasting method based on chaotic models for very-low demand items; Chapter 6 introduces a periodic inventory system designed to deal with perishable items; Chapter 7 expands the experimental analysis of such a system.

1. Intermittent demand main themes

Forecasting

The literature provides two main compound distributions for modelling the intermittent demand generation process:

- If time is treated as a discrete variable, demand can be assumed to be generated by a Bernoulli process [1]. The inter-demand intervals are geometrically distributed.
- If time is treated as a continuous variable, demand can be assumed to be generated by a Poisson process [5]. The inter-demand intervals are exponentially distributed.

When combining a Bernoulli or a Poisson demand arrival with a generic distribution of demand sizes, a compound distribution is obtained. An empirical goodness-of-fit investigation [6] showed the good fit of compound Poisson distributions to thousands of spare parts characterised with intermittent demand.

[1] is the seminal contribution on intermittent demand forecasting according to a Bernoulli probability of a demand occurrence, [7] commented on [1] correcting some of its equations. In [1] the demand z_i of a period t was defined as:

$$z_t = \begin{cases} d_t & with\, probability\; p \cdot \phi(d_t) \\ 0 & with\, probability\; 1-p \end{cases} \tag{3.01}$$

p was the probability a positive demand took place in a period and $\phi(d_t)$ was the probability or the probability density a positive demand had magnitude d_t if a positive demand took place.

[8] outlined the demand expected value and variance:

$$E(z_t) = p \cdot E(d_t) \tag{3.02}$$

$$Var(z_t) = p \cdot Var(d_t) + p \cdot (1-p) \cdot E(d_t)^2 \tag{3.03}$$

Initially the exponential smoothing (ES) was used as a forecasting method:

$$\hat{z}_t = (1-\alpha) \cdot z_{t-1} + \alpha \cdot \hat{z}_{t-1} \tag{3.04}$$

$\hat{z}_t$ was the estimated demand after the demand z_{t-1} in period $t-1$. The forecasts were updated every period using the smoothing parameter $\alpha \in (0,1)$ that defined how much new data impacted on the previous estimates.

[1] showed that ES was a biased if the forecast was computed after a positive demand occurrence:

$$E(\hat{z}_t) = E(d_t) \cdot \left(\alpha + \frac{1-\alpha}{p}\right) \tag{3.05}$$

[1] proposed a different estimator (Croston). A simple exponential smoothing was applied to both variables when a positive demand occurred, then an estimator of the demand expected value per period was computed as the ratio of these smoothing estimators:

$$\hat{d}_t = (1-\alpha) \cdot d_{t-1} + \alpha \cdot \hat{d}_{t-1} \tag{3.06}$$

$$\widehat{in}_t = (1-\alpha) \cdot in_{t-1} + \alpha \cdot \widehat{in}_{t-1} \tag{3.07}$$

$$\hat{z}_t = \frac{\hat{d}_t}{\widehat{in}_t} \tag{3.08}$$

$\hat{d}_t$ was the estimated magnitude of a positive demand after the positive demand d_{t-1} in period $t-1$ and $\widehat{in}_t$ was the estimate inter-arrival between positive demands calculated from the last inter-arrival in_{t-1} between positive demands. The forecasts were updated only after a positive demand using the smoothing parameter $\alpha \in (0,1)$.

[9] proposed a similar estimator (Schultz) where the two exponential smoothing used different smoothing parameters:

$$\hat{d}_t = (1-\alpha) \cdot d_{t-1} + \alpha \cdot \hat{d}_{t-1} \tag{3.09}$$

$$\widehat{in}_t = (1-\beta) \cdot in_{t-1} + \beta \cdot \widehat{in}_{t-1} \tag{3.10}$$

$\beta \in (0,1)$. The ratio of the two estimates was not used since the proposed $(S-1, S)$ base stock method relied on separate forecasts.

[10] showed that Croston was biased:

$$E(\hat{z}_t) = E(d_t) \cdot \left(-\frac{1}{1-p} \cdot ln\left(\frac{1}{p}\right)\right) \tag{3.11}$$

Its authors proposed an effective [11] and approximately unbiased modification of Croston, the Syntetos-Boylan Approximation (SBA) [12], which later became a benchmark in the intermittent demand forecasting literature:

$$\hat{z}_t = \left(1-\frac{\alpha}{2}\right) \cdot \frac{\hat{d}_t}{\widehat{in}_t} \tag{3.12}$$

Another modification of Croston was proposed in [13] (LS):

$$\hat{z}_t = \frac{d_{t-1}}{in_{t-1}} + (1-\alpha) \cdot \hat{z}_{t-1} \tag{3.13}$$

[14] demonstrated that it led to an even more biased estimator:

$$E(\hat{z}_t) = E(d_t) \cdot \left(1-\frac{1}{p-1} \cdot ln\left(\frac{1}{p}\right)\right) \tag{3.14}$$

[15] compared LS to SBA, Croston and an unbiased modification of Croston proposed by Syntetos in his Ph. D. thesis (Syntetos):

$$\hat{z}_t = \left(1-\frac{\alpha}{2}\right) \cdot \frac{\hat{d}_t}{\widehat{in}_t - \frac{\alpha}{2}} \tag{3.15}$$

SBA led to the lowest mean squared error (MSE) but, when the MSE was divided into its components [8]:

$$MSE = Var(z_t) + Var(\hat{z}_t) + bias(\hat{z}_t) \tag{3.16}$$

$bias(\hat{z}_t)$ was the expected squared difference between the estimator and the demand expected values:

$$bias(\hat{z}_t) = E(\hat{z}_t - z_t)^2 \tag{3.17}$$

Then SBA led to a lower estimator variance while Syntetos led to a lower bias.

In all the methods using two exponentials smoothing an estimate of the probability p could be obtained:

$$\hat{p}_t = \frac{1}{\widehat{in}_t} \tag{3.18}$$

A recent modification of Croston designed to deal with inventory obsolescence was proposed in [16] (TSB). TSB applied the second exponential smoothing to the demand probability instead of the inter-demand arrivals:

$$\hat{d}_t = (1-\alpha) \cdot d_{t-1} + \alpha \cdot \hat{d}_{t-1} \tag{3.19}$$

$$\hat{p}_t = (1-\beta) \cdot p_{t-1} + \beta \cdot \hat{p}_{t-1} \tag{3.20}$$

$$\hat{z}_t = \hat{d}_t \cdot \hat{p}_t \tag{3.21}$$

$\beta \in \{0,1\}$ was a second smoothing parameter, as in Schultz, and p_t was the demand event at time t:

$$p_t = \begin{cases} 1 & a\ positive\ demand\ takes\ place\ in\ t \\ 0 & no\ positive\ demand\ takes\ place\ in\ t \end{cases} \tag{3.22}$$

TSB, like ES, was unbiased for a random period t and biased if the forecast was computed after a positive demand occurrence.

Interested readers can refer to [8] for an analysis of expected value and variance of the most common traditional intermittent demand estimators.

Machine learning was applied for forecasting intermittent demand due to its ability to generalise a nonlinear process without requiring any distributional assumptions. [17] used a neural network (NN) having the previous period demand and the last inter-arrival as inputs:

$$\hat{z}_t = f(d_{t-1}, in_{t-1}) \tag{3.23}$$

The NN was composed of three layers, three hidden nodes and sigmoid activation functions as depicted in Figure 3.01.

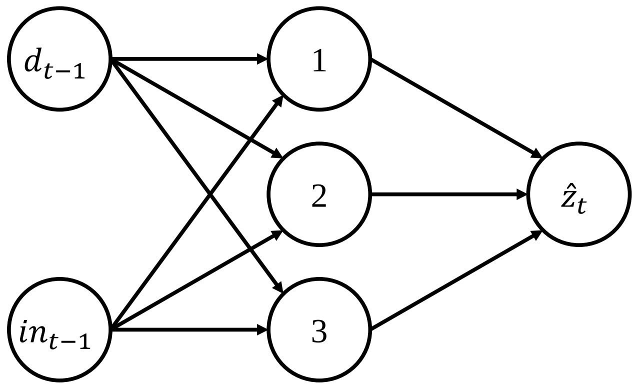

Figure 3.01. [17] neural network architecture.

[18] proposed two different NN architectures depicted in Figure 3.02 and 3.03 respectively. Both NN used previous intervals and positive demands as inputs, three hidden nodes and hyperbolic tangent activation functions. The first architecture predicted $\hat{d}_t$ and $\widehat{in}_t$ separately, then combined them with a factor c:

$$\hat{d}_t = f_d(d_{t-1}, \dots, in_{t-1}, \dots) \tag{3.24}$$

$$\widehat{in}_t = f_{in}(d_{t-1}, \dots, in_{t-1}, \dots) \quad (3.25)$$

$$\hat{z}_t = c \cdot \frac{\hat{d}_t}{\widehat{in}_t} \quad (3.26)$$

The second architecture predicted $\hat{z}_t$ directly:

$$\hat{z}_t = f(d_{t-1}, \dots, in_{t-1}, \dots) \quad (3.27)$$

Both NN were trained using a regularization function to avoid overfitting without the need for a validation set.

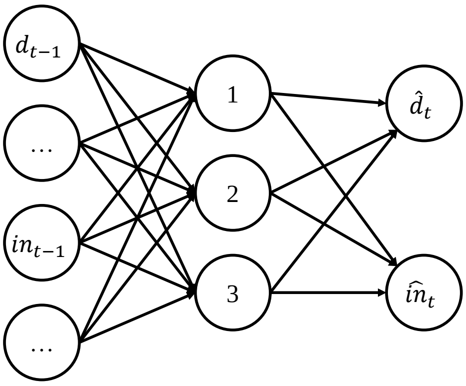

Figure 3.02. [18] first neural network architecture.

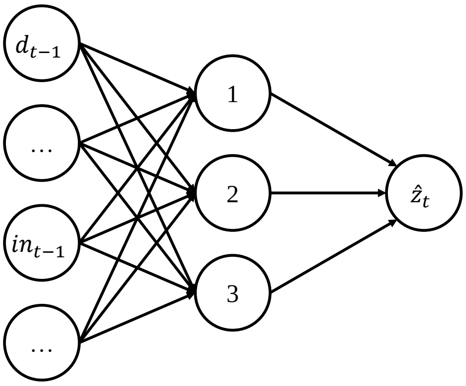

Figure 3.03. [18] second neural network architecture.

In [18] these two neural network were compared with the one proposed in [17], Croston, SBA and other traditional forecasting methods proposed in [19].

[20] proposed a NN architecture similar to the one in [17], depicted in Figure 3.04:

$$\hat{z}_t = f(d_{t-1}, n_{t-1}) \quad (3.28)$$

n_{t-1} was the number of periods between the last positive demand and period $t - 1$.

[21] compared a set of NN architectures with traditional forecasting methods and found [20] to be the most effective.

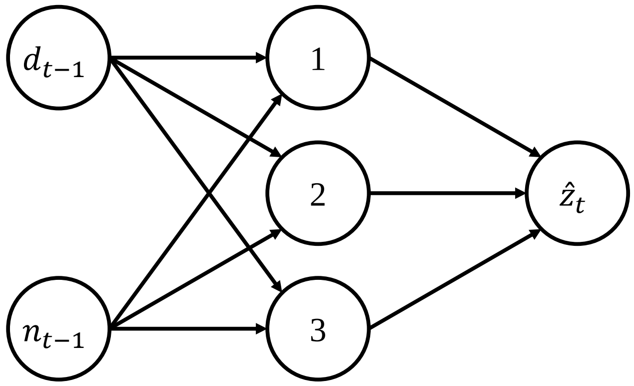

Figure 3.04. [20] neural network architecture.

Several researchers recommended the use of bootstrapping [22] as an alternative to traditional forecasting methods. A recent literature review on bootstrapping in the context of intermittent demand was presented in [23], where the authors reviewed six bootstrapping methods divided in:

- Bootstrap with distributional assumptions.
    - Parametric bootstrap.
    - Smooth bootstrap.
- Bootstrap without distributional assumptions.
    - Nonparametric bootstrap.

For a comparison between parametric and non-parametric approaches and a thorough investigation of bootstrapping, the readers can refer to [24]–[26].

[27] proposed a literature review containing most of the forecasting methods designed to deal with intermittent demand.

Inventory management

After the forecasting phase the management of intermittent items requires an inventory system to control:

- When orders are placed.
- How many items are ordered each time.

Which in turn affects how many items are available in stock at any given time, leading to:

- Ordering costs.
- Holding costs.
- Customer satisfaction.

Customer satisfaction can in turn be translated in backorder costs or in constraints on the items' availability, this second operation does not require an economical evaluation of the customer satisfaction which is often imprecise. The most widely used customer satisfaction performance measures for constraint purposes are:

- Cycle service level (CSL), the probability all the cycle demand is satisfied from stock without incurring in backorders.
- Fill rate (FR), the fraction of demanded items satisfied from stock without incurring in backorders.

Both these performance measures can be applied to the most common inventory systems:

- (s, Q) continuous inventory system, where every time the net stock reaches s an order of Q items is placed.
- (s, S) continuous inventory system, where every time the net stock reaches s an order is placed to reach S.
- (R, S) periodic inventory system, where every R periods an order is placed to reach S.

Traditional inventory management systems are not equipped to deal with intermittent patterns as they do not account for periods with no demand and undershoots at the beginning of the reorder phase. [28] proposed a (R, s, Q) period inventory system where an order of Q was placed every R periods if the net stock reached s. It was one of the first inventory systems adapted to the Bernoulli processes typical of intermittent demand, but it did not consider any specific forecasting method. [29] modified this original proposal transforming it into a (s, Q) inventory system and incorporating Croston as the forecasting method. It focused on the forecasting error in a period t:

$$e_t = z_t - \hat{z}_t \tag{3.29}$$

Since a single forecast was computed before the lead-time L the variance of the forecasting error $\sum_{t=1}^{L} e_t$ during such a lead-time was:

$$Var(\sum_{t=1}^{L} e_t) = L \cdot Var(z_t) + L^2 \cdot Var(\hat{z}_t) \tag{3.30}$$

Without considering the undershoot the lead-time demand had an expected value of:

$$E(\sum_{t=1}^{L} z_t) = \ L \cdot E(z_t) \tag{3.31}$$

And a variance, considering the forecasting error, of:

$$E((\sum_{t=1}^{L} z_t - \sum_{t=1}^{L} \hat{z}_t)^2) = Var(\sum_{t=1}^{L} e_t) \tag{3.32}$$

In order to model the lead-time demand a probability density function had to be fitted to the data, when the undershoot was disregarded $E(\sum_{t=1}^{L} z_t)$ and $E((\sum_{t=1}^{L} z_t - \sum_{t=1}^{L} \hat{z}_t)^2)$ could be used to fit a probability density function through the method of moments. If the undershoot was considered two cases had to be accounted for:

- If there was only the undershoot and no positive demand occurred during the lead-time.
- If, after the undershoot, positive demands occurred during the lead-time.

Two different probability density functions could be computed from the expected value and the variance of each case.

The expected value of the positive demand during the lead-time was computed as:

$$E(\sum_{t=1}^{L} z_t \,|\exists\, z_t > 0\ t \in 1, \dots, L) = \frac{E(\sum_{t=1}^{L} z_t)}{1-(1-p)^L} \tag{3.33}$$

$1-(1-p)^L$ was the probability of at least one positive demand during the lead-time.

Its variance was:

$$Var(\sum_{t=1}^{L} z_t \,|\exists\, z_t > 0\ t \in 1, \dots, L) = \frac{E\left((\sum_{t=1}^{L} z_t - \sum_{t=1}^{L} \hat{z}_t)^2\right)}{1-(1-p)^L} - (1-p)^L \cdot \frac{E(\sum_{t=1}^{L} z_t)^2}{(1-(1-p)^L)^2} \tag{3.34}$$

The undershoot expected value was approximated as:

$$E(u) \cong \frac{E(d_t^2)}{2 \cdot E(d_t)} \tag{3.35}$$

Its variance was approximated as:

$$Var(u) \cong \frac{E(d_t^3)}{3 \cdot E(d_t)} - E(u)^2 \tag{3.36}$$

The expected value of both the positive demand and the undershoot was computed as:

$$E(\sum_{t=1}^{L} z_t + u | \exists\ z_t > 0\ t \in 1, \dots, L) = E(\sum_{t=1}^{L} z_t\ | \exists\ z_t > 0\ t \in 1, \dots, L) + E(u) \tag{3.37}$$

Its variance was:

$$Var(\sum_{t=1}^{L} z_t + u | \exists\ z_t > 0\ t \in 1, \dots, L) = Var(\sum_{t=1}^{L} z_t\ | \exists\ z_t > 0\ t \in 1, \dots, L) + Var(u) \tag{3.38}$$

The optimal value of Q was the economic order quantity and, given the probability density function of both the undershoot and the positive lead-time demand with undershoot, an optimization problem was solved to find the optimal values of s given the fill rate:

$$1 - FR = (1 - (1 - p)^L) \cdot \frac{\left(E\left((\sum_{t=1}^{L} z_t + u - s)^+ | \exists\ z_t > 0\ t \in 1, \dots, L\right) - E\left((\sum_{t=1}^{L} z_t + u - s - Q)^+ | \exists\ z_t > 0\ t \in 1, \dots, L\right)\right)}{Q} + (1 - p)^L \cdot \frac{E((u-s)^+) - E((u-s-Q)^+)}{Q} \tag{3.39}$$

$x^+ = max(x, 0)$.

These equations required estimators for:

- $E(z_t)$, the expected value of the demand.
- $E(d_t)$, the expected value of the positive demand.
- $Var(d_t)$, the variance of the positive demand.
- $Var(\hat{z}_t)$, the variance of the demand forecast.

[8] estimated $E(z_t)$ and $E(d_t)$ as:

$$\hat{E}(z_t) = \hat{z}_t \tag{3.40}$$

$$\hat{E}(d_t) = \hat{d}_t \tag{3.41}$$

In [3] the variance of the positive demand, considering the forecasting error, was computed as:

$$\widehat{Var}(d_t) + \widehat{Var}(\hat{d}_t) = (1 - \beta) \cdot \left(\widehat{Var}(d_{t-1}) + \widehat{Var}(\hat{d}_{t-1})\right) + \beta \cdot \left(d_{t-1} - \hat{d}_{t-1}\right)^2 \tag{3.42}$$

The smoothing parameter $\beta \in (0,1)$ did not necessarily equal the smoothing parameter α.

In [30] the same variance was computed as:

$$MAD_t = (1 - \beta) \cdot MAD_{t-1} + \beta \cdot \left|d_{t-1} - \hat{d}_{t-1}\right| \tag{3.42}$$

$$\widehat{Var}(d_t) + \widehat{Var}(\hat{d}_t) \cong \sqrt{1.25 \cdot MAD_t} \tag{3.43}$$

In [8] the variance of the positive demand forecast, using the ES forecasting method, was computed as:

$$Var(\hat{d}_t) = \frac{\alpha}{2 - \alpha} \cdot Var(d_t) \tag{3.44}$$

Thus:

$$Var(d_t) = \left(1 - \frac{\alpha}{2}\right) \cdot \left(Var(d_t) + Var(\hat{d}_t)\right) \tag{3.45}$$

Estimated as:

$$\widehat{Var}(d_t) = \left(1 - \frac{\alpha}{2}\right) \cdot \left(\widehat{Var}(d_t) + \widehat{Var}(\hat{d}_t)\right) \tag{3.46}$$

In [29] the variance of the demand forecast, using the ES forecasting method, was approximated as:

$$\widehat{Var}(\hat{z}_t) \cong p^2 \cdot \frac{\alpha}{2-\alpha} \cdot (Var(d_t) + (1-p) \cdot E(d_t)^2) \tag{3.47}$$

And could be estimated as:

$$\widehat{Var}(\hat{z}_t) \cong \hat{p}^2 \cdot \frac{\alpha}{2-\alpha} \cdot \left(\widehat{Var}(d_t) + (1-\hat{p}) \cdot {\hat{d}_t}^{\,2}\right) \tag{3.48}$$

[8] argued against the use of this approximation and suggested using:

$$\widehat{Var}(\hat{z}_t) \cong p^2 \cdot \frac{\alpha}{2-\alpha} \cdot \left(Var(d_t) + (1-p) \cdot \left(E(d_t)^2 + \frac{\alpha}{2-\alpha} \cdot Var(d_t)\right)\right) \tag{3.49}$$

Estimated as:

$$\widehat{Var}(\hat{z}_t) \cong \hat{p}^2 \cdot \frac{\alpha}{2-\alpha} \cdot \left(\widehat{Var}(d_t) + (1-\hat{p}) \cdot \left({\hat{d}_t}^{\,2} + \frac{\alpha}{2-\alpha} \cdot \widehat{Var}(d_t)\right)\right) \tag{3.50}$$

[30] proposed a simpler (s, S) inventory system accounting for both the intermittent nature of the demand and the potential undershoot. [30] considered the forecasting error during a lead-time with undershoot:

$$\sum_{t=1}^{L} e_t + e_u = \sum_{t=1}^{L}(z_t - \hat{z}_t) + d_{t-1} - \hat{d}_{t-1} \tag{3.51}$$

Only a forecast was computed before the undershoot, thus:

$$\sum_{t=1}^{L} e_t + e_u = \sum_{t=1}^{L} z_t - L^2 \cdot \hat{z}_{t-1} + d_{t-1} - \hat{d}_{t-1} \tag{3.52}$$

The variance was corrected accounting for the covariance between forecasts:

$$Var\left(\sum_{t=1}^{L} e_t\right) = L \cdot Var(z_t) + L^2 \cdot Var(\hat{z}_t) + Var(d_t) + Var(\hat{d}_t) + 2 \cdot Cov(L \cdot \hat{z}_t, d_t) \tag{3.53}$$

Computed as:

$$Cov(L \cdot \hat{z}_t, d_t) = L \cdot p \cdot Var(\hat{z}_t) \tag{3.54}$$

The optimal value of s was computed approximating the lead-time demand, considering the undershoot, to a normal distribution:

$$s = L \cdot \hat{z}_t + \hat{d}_t + \Phi(CSL)^{-1} \cdot Var\left(\sum_{t=1}^{L} e_t\right) \tag{3.55}$$

$\Phi(CSL)^{-1}$ was the inverse of the probability a CSL was achieved.

[3] proposed a (R, S) inventory system analysed in Chapters 6 and 7.

Classification

Given a set of items there are many possible forecasting methods and inventory systems. While it is possible to test multiple scenarios for each one, the literature provides classification schemas based on the time series features.

[31] proposed one of the first classification schemas for forecasting purposes (Figure 3.05) where the lead-time variance was split into two:

- $\frac{1}{p \cdot L} \cdot \frac{Var(d_t)}{E(d_t)^2}$, the relative variability of the demand.
- $\frac{1}{p \cdot L}$, the number of lead-times between positive demands.

With $\frac{1}{p \cdot L} \cdot \frac{Var(d_t)}{E(d_t)^2} = 0.5$, $\frac{1}{p \cdot L} = 0.7$ and $\frac{1}{p \cdot L} = 2.8$ as cut points.

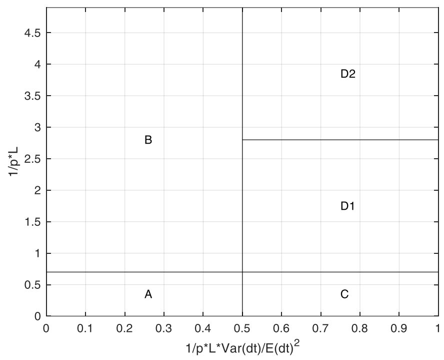

Figure 3.05. [31] classification schema.

The chosen features were not theoretically grounded and the cut points were arbitrary. [32] proposed an alternative (Figure 3.06) based on [1] intermittent demand generation process. The demand was assumed to occur as a compound Bernoulli process, allowing for the computation of theoretical MSE for Croston, SBA and exponentially weighted moving average (EWMA) [33]. These theoretical MSE were compared to identify regions of superior performance for each method:

$$MSE_{Croston} > MSE_{SBA} \quad if \ \frac{Var(d_t)}{E(d_t)^2} > \frac{(1-p)\cdot(2-\alpha)\cdot\left(\frac{4\cdot(1-p)}{p\cdot(2-\alpha)} - \frac{p\cdot(2-\alpha)}{1-p} - p\cdot(\alpha-4)\right)}{(\alpha-4)\cdot\left(\frac{2}{p}-\alpha\right)} \tag{3.56}$$

$$MSE_{EWMA} > MSE_{SBA} \quad if \ \frac{Var(d_t)}{E(d_t)^2} > \frac{\frac{1-p}{p^2}\cdot\left(\frac{4}{p}-(2-\alpha)^2\right) - \alpha\cdot(2-\alpha)}{(2-\alpha)\cdot\left(\frac{2}{p^2}-\frac{\alpha}{p}\right) - \frac{4}{p^3}} \tag{3.57}$$

$$MSE_{EWMA} > MSE_{Croston} \quad if \ \frac{Var(d_t)}{E(d_t)^2} > p - 1 \tag{3.58}$$

It was found that EWMA was outperformed by both Croston and SBA, and Croston overperformed SBA only for time series characterized by low intermittence. Following the MSE equations the two classification features were changed into:

- $\frac{Var(d_t)}{E(d_t)^2}$, the relative variability of the demand.
- p, the number of lead-times between positive demands.

With $\frac{Var(d_t)}{E(d_t)^2} = 0.49$ and $\frac{1}{p} = 1.32$ as approximate cut points.

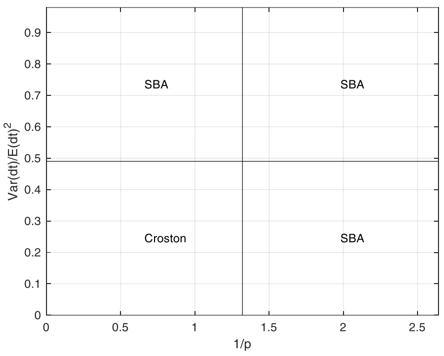

Figure 3.06. [32] classification schema.

[34], [35] criticized the use of approximate cut points and solved the MSE equations exactly. It was also proposed a simpler and more precise approximation, depicted in Figure 3.07, based on a single linear boundary dividing the feature space in two. [36] verified the superior performance of the new approximation over the old one, while the difference between exact and approximated solution was found to be negligible.

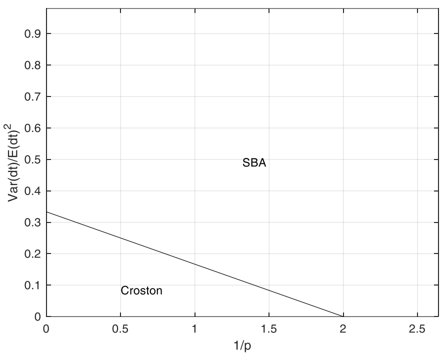

Figure 3.07. [34] classification schema.

[37] extended the theoretical classification to inventory management purposes suggesting different probability density functions for different time series. A Poisson process was combined with different positive demand probability density functions to obtain compound probability density functions during the lead-time:

- If $M(d_t) = 1$ and $0 < \frac{Var(d_t)}{E(d_t)^2} < 1$ a Poisson-Geometric probability density function was suggested.
- If $M(d_t) \geq 1$ and $0 < \frac{Var(d_t)}{E(d_t)^2} < 1$ a Poisson-Poisson probability density function was suggested.
- If $M(d_t) = 1$ and $\frac{Var(d_t)}{E(d_t)^2} \geq 1$ a Poisson-Logarithmic probability density function was suggested.
- If $M(d_t) \geq 1$ and $\frac{Var(d_t)}{E(d_t)^2} \geq 1$ a Poisson-Pascal probability density function was suggested.

[27] proposed a literature review containing most of the classification schemas designed to deal with intermittent demand. A literature review on classification for forecasting purposes is also outlined in Chapters 2, 3 and 4.

2. Neural networks and Support Vector Machines for inventory classification

This chapter is adapted from the paper "Machine learning for multi-criteria inventory classification applied to intermittent demand" published in 2019 in Production Planning & Control, volume 30, issue 1, pages 76-89 [38].

Introduction

Forecasting, inventory control and multi-criteria inventory classification (MCIC) are interrelated fields of research. When dealing with a huge number of items, firms group them in classes to simplifying their management. The classification framework can be used:

- To determine the importance of each item and the level of control placed upon it [39].
- For managerial convenience, the same inventory control system or forecasting technique can be applied to each item within a class.

In the first case the inventory manager associates the same target CSL or the same target fill rate to the items of the same class. In the second case, and if a single inventory control system is applied, an appropriate reorder policy needs to be selected [40]. If classification is used to select a reorder policy, standard classification models based on criteria and threshold values may not be adequate to associate reorder policies and classes [41]. Often in those models three ABC classes of ordered importance are defined and then inventory control policies are attached to them. In this framework items are classified according to a single criterion, which is often the usage value given by the product of the unit cost and the total annual demand. The assignment is typically based on an arbitrary percentage, for example class A receives 20% of the items with the highest usage value, class B receives the next 30% and class C the remaining 50%. [42] recognised that multiple criteria would give more precision in the definition of classes by augmenting the item characterisation. Therefore, several multi-criteria methods have been proposed for enriching the inventory classification:

- Analytical Hierarchy Process (AHP) and its extensions [43]–[48].
- TOPSIS [49].
- The weighted linear optimisation [50]–[57].
- Fuzzy logic [58].
- Case-based reasoning [59], [60].

However, classifying items through MCIC approaches is still a separate action from finding appropriate strategies for each class [41], [61]. The performance of MCIC methods does not necessarily include the performance of the inventory control system and little attention has been paid to the empirical implications of a simultaneous inventory classification and inventory control [62]. The original goal of minimising the total relevant cost is therefore forgotten when MCIC and inventory system design are not carried out jointly.

An exhaustive simulation of the inventory system performed at a single item level would avoid recourse to MCIC by optimally classifying all the items ex post in an inventory cost perspective. However, an exhaustive simulation of the whole population of items requires computation efforts that are too high for time varying settings with thousands of items to be managed. Meta-heuristics [61], [63], [64] and exact methods [65] have been proposed to solve this combinatorial problem through simplified assumptions without recurring to the exhaustive solution, but the classification remains opaque.

This chapter investigates the use of two supervised machine learning classifiers as tools for MCIC:

- Support vector machines with Gaussian kernel (SVM).
- Deep neural networks (DNN).

Machine learning needs a large amount of data to work well and enables complex decision-making processes to be automated, MCIC is a complex process often applied to thousands of items. Machine learning tools may therefore be promising in MCIC. This chapter proposes a two-stage automatic classification approach:

1. Classification via simulation of the in-sample items.
2. Classification via machine learning of the out-of-sample items

For experimental purposes only intermittent demand patterns are considered, but the inventory system selected here is flexible enough to also suit non-intermittent demand. A periodic order-up-to level inventory system is adopted, where the decision variable to change among classes, which therefore drives the item classification, is the review interval. The higher the criticality of an item, the lower the review interval should be, and vice versa.

Literature review

Traditional multi-criteria inventory classification

A pioneering contribution on MCIC was provided by [43]. They applied the AHP to classify items in terms of usage value, average unit cost, criticality, and the lead-time. This represents one of the early attempts to overcome the weakness shown by the mono-criterion classification on usage value in representing the whole criticality of an item. AHP was also adopted by other authors [44]–[46], [48] and in particular for spare parts classification [66]. Other multi-criteria methods have been used, such as weighted optimisation [50]–[54], [67], [68] and case-based reasoning [59], [60].

Since the final goal is to associate the classes with specific inventory control methods, a further open issue to investigate is the correlation between the classification criteria and the efficiency of the inventory system. Once the classes have been established, a unique inventory control method is selected for all items of the same class [69]. However, there is no certitude that the criteria used for the classification are appropriate to guarantee the best performance of the inventory method. Indeed, it has been empirically shown that MCIC methods based on the annual dollar usage and the unit cost criteria have a low cost-service performance [41]. Moreover, different MCIC methods reach different classifications [67] when applied to the same dataset, this proves that these methods are not robust. [67] introduced new procedures for reaching the consensus among different MCIC approaches, but the relationship of the criteria with the inventory system was not explored, and the class cardinalities were predefined.

Only two papers have proposed an inventory classification from an inventory cost perspective. [70] presented an item classification scheme that is consistent with their cost optimisation model. Instead of a cost-based model, including order and shortage costs, a model constrained to the target CSL and order frequency was proposed through a continuous review inventory system. It derived an expression for reorder points to obtain an item classification ratio: an item was ranked as high (i.e. with a high target CSL) if the demand rate was large, or if the squared holding cost and the replenishment lead time were small. Starting with a total cost model including the shortage cost, [71] derived an exact expression of the CSL with a different ratio for item classification. An item was ranked high if the demand rate and shortage cost were large or if the holding cost and the order quantity were small. Empirically the new classification criterion outperformed all the other methods compared to it.

Regardless of the classifying criteria and the multi-criteria approach used, item classification requires a sorting method, so that the exercise of ranking items does not achieve the objective [72]. In other words, the cardinalities of the clusters should be subjectively established by decision makers based on their expertise, see the criticism in [48]. It follows that the classes are disjointed in relation to inventory control, with items not being robustly classified.

Machine learning applied to multi-criteria inventory classification

Only a few contributions have focused on applying machine learning classification algorithms to MCIC, and none have been extended to the inventory systems.

The first work on applying machine learning classification algorithms to MCIC dates back to [73], where NN were used to classify small populations of items by training the networks on a sample of items previously classified by decision makers on subjective criteria. However, the inventory system was not considered, and thus the classification obtained was not cost-oriented. Similarly, [74] experimented with SVM, NN and the k-nearest neighbour algorithm on Reid's famous dataset [75] of forty-seven items using the approaches of [43], [50], [52] as benchmarks. SVM was shown to be the most accurate machine learning classifier to replicate these MCIC approaches. However, it did not take into account the cost-oriented optimality of the classifications. [76] tested single and two-hidden layer NNs on a dataset of 351 items, which had already been classified by a fuzzy AHP priority scoring method. Again the classification was not cost-oriented and the MCIC remained disjointed from the inventory system. [77] exploited Logical Analysis of Data, a supervised data mining technique able to generate patterns into classes by a Boolean function, in order to detect biases or inconsistencies in the classification provided by decision makers. [78] combined several multicriteria decision making approaches with machine learning algorithms for MCIC. Multicriteria approaches were used to classify items into classes, and then machine learning classifiers (naive Bayes, Bayesian network, NN, and SVM) were trained on the resulting classifications in order to predict the classes to which the items belong. NN and SVM were shown to have the highest accuracy. However, the accuracy of machine learning classifiers was again evaluated with respect to classifications without considering the inventory system. Consequently, the selection of classification criteria from an inventory cost perspective remains an open issue.

Solution approach

The forecast-based inventory control system

A forecast-based periodic review system is employed given its simplicity and compatibility with physical warehouse reviews and periodic orders issued to the suppliers [11]. A (R, S) periodic review system is selected, with review interval R and order-up-to level S dynamically computed per item on a forecast basis every R periods [79]. S depends on the time period t and the item i and is identified as $S_{i,t}$, while R_i needs to be established for each item i and drives the inventory classification. Since time is treated as a discrete variable in periodic order-up-to level policies, let t be a generic time period at the end of a replenishment cycle of R_i periods. Supposing that orders are placed at the very end of the replenishment cycles, on the boundary with the next period, and item i has a lead time of L_i periods, the order of events in t is assumed as follows:

1. If an order is placed in $t - L_i - 1$, at the very end of this period, it arrives at the beginning of period t.
2. A demand occurs and $S_{i,t}$ is re-computed on the forecast basis.

The rule for emitting orders is: after every R_i periods if the inventory position $IP_{i,t}$ (composed of $NI_{i,t} + planned\ orders_{i,t} - backorders_{i,t}$) is below $S_{i,t+1}$ an order to reach $S_{i,t+1}$ is placed. The inventory position $IP_{i,t}$ and the net inventory $NI_{i,t}$ have different meanings:

- $IP_{i,t}$ contains the balance between issued but not yet delivered orders, and backorder units to be delivered as soon as available.
- $NI_{i,t}$ is the physical inventory status.

If backorders do not occur and $R_i \geq L_i + 1$, i.e. an order always arrives before the next one is issued, the net inventory equals the inventory position. Conversely, if $R_i < L_i + 1$, replenishment cycles might overlap, and a new order can be placed before the last one issued arrives.

Since a pure cost objective function cannot be evaluated due to the unavailability of backordering costs, a constraint perspective is suggested. In this scenario the minimisation of a cost function including the holding and ordering cost is pursued, while satisfying a pre-specified service level. The CSL_i is used for safety stock calculation by defining a target $tCSL_i$ for each item i. The higher the criticality of an item, the higher its $tCSL_i$ assigned by a decision-maker.

Both Croston [1] and SBA [12] are applied in the experimental analysis as estimators of the mean demand per period. When applying a forecast-based stock control, the variance of forecast errors can be used to calculate the safety stocks necessary to achieve $tCSL_i$. According to [79], the single exponential smoothing of the mean squared error MSE provides an estimator of the variance $MSE_{i,t}$.

In order to dynamically compute $S_{i,t}$ at the end of period $t-1$, the following equation is applied:

$$S_{i,t} = (L_i + R_i)F_{i,t} + \phi^{-1}(tCSL_i)\sqrt{MSE_{i,t}(L_i + R_i)} \qquad (4.01)$$

$\phi^{-1}(tCSL_i)$ is the inverse cumulative standard normal distribution for $tCSL_i$. The first right-hand-side term corresponds to the forecast demand for periods $L_i + R_i$ in a stationary mean model, with $F_{i,t}$ given either by Croston or SBA. The second term is the safety stock required to reach $tCSL_i$ by applying a safety factor of $\phi_i^{-1}(tCSL_i)$.

Given either Coston or SBA as the forecasting method, the following parameters depend directly on item i:

- L_i is deterministic and given for each item i.
- $tCSL_i$ refers to the criticality level of item i.

All the other parameters can be optimised.

This inventory system can also be applied to non-intermittent demands, in this case Croston becomes a single exponential smoothing. Nevertheless, the exhaustive search enables all the predictors to be tested and all factors to be optimised.

The review interval R_i may vary in a range of K feasible values for all the population items. This parameter is exhaustively tested via simulation on the sample I', driving by this way the classification of the $|\mathrm{I}'|$ in-sample items. In fact, the value minimising a total cost function is selected per item into I', and thus it follows that each item is classified into the class C_k associated with its best review interval. These in-sample items are then used to train different machine learning classifiers for comparison, whose goal is therefore to classify the remaining $|\mathrm{I}| - |\mathrm{I}'|$ items into the K classes without having to resort to an exhaustive search on all of them.

Exhaustive simulation of the in-sample items

The forecast method and the smoothing parameters need to be established per item. Given a time horizon of T periods the smoothing parameters are optimised per item in a warm-up period of the first n periods by minimising the mean squared error between demands and estimators. The performance of the inventory system obtained by changing R_i among K values is collected for the remaining $T -$

n periods. For a fixed R_i, this performance is given by the net inventory NI_{i,t,R_i} and the emitted orders NO_{i,t,R_i} for each period t between $n+1$ and T, defined as:

$$NO_{i,t,R_i} = \begin{cases} 1 & if\ an\ order\ is\ emitted\ in\ period\ t\ with\ review\ interval\ R_i \\ 0 & otherwise \end{cases} \tag{4.02}$$

The total relevant cost TRC_{i,R_i} per period depends on the review interval R_i as follows:

$$TRC_{i,R_i} = h_i \cdot C_i \cdot \sum_{t=n+1}^{T} \frac{NI_{i,t,R_i}}{T-n} + o_i \cdot \sum_{t=n+1}^{T} \frac{NO_{i,t,R_i}}{T-n} \tag{4.03}$$

The first and second terms are the average holding and ordering costs per time period, respectively. h_i is the unitary holding cost, the percentage of the unitary purchasing cost C_i for holding item i in stock for a certain period, and o_i is the unit ordering cost.

The R_i minimizing $TRC_{i,R}$ is selected for each item i. This is the most efficient R_i in terms of TRC_{i,R_i} after optimising all the forecasting parameters. The R_i selected for each item is the driver for classifying the $|\mathrm{I}'|$ in-sample items into K classes from an inventory perspective, that is all the items $i \in \mathrm{I}'$ are now classified into a class C_k, with $k \in \{1, \dots, K\}$. These items are therefore the references used for classifying the whole population of $|\mathrm{I}|$ items by means of J classification criteria.

Classification criteria

In this chapter, the classification is inventory oriented. The criteria expected to impact on the performance of the inventory control system, and consequently the item classification, are selected. These criteria belong to two different groups, which refer to:

- The statistics of the time series.
- The characteristics of the items.

The first group includes the parameters required for setting the demand generation. They refer to the two stochastic variables of an intermittent demand process, i.e. the inter-arrival between successive demands and the positive demand size:

- Expected value of the positive demand probability $E(p_i)$, which is calculated as the ratio between the number of periods with positive demands and the total number of periods over T.
- Expected value of the positive demand size $E(d_i)$, which is calculated as the average of positive demands values over T.
- Expected value of the standard deviation of the positive demand $E[Std(d_i)]$, which is calculated as $\sqrt{MSE_i}$ over T.

The second group of criteria derives from the inventory control system as well as from the approach adopted for the selection of the best policy per item:

- Relative unitary cost $\frac{o_i}{C_i h_i}$.
- Safety factor $\phi_i^{-1}(tCSL_i)$.

It is assumed that J criteria have been assessed for the item classification, and that an item i shows a value x_{ij} on criterion j, with $i = 1, \dots, |\mathrm{I}|$ and $j = 1, \dots, J$.

Machine learning classifiers

Two machine learning classifiers are used:

- SVM with Gaussian kernel [80].
- DNN [4].

Both algorithms can be very effective in nonlinear contexts and present a limited number of meta-parameters to be optimised.

For the SVM, the meta-parameters are:

- Box constraint C, penalty applied to incorrectly separated items.
- Kernel $scale$, the Gaussian kernel is multiplied by this scale factor.

While for the DNN, the relevant meta-parameters are:

- Number of hidden layers N_{hidden}.
- Number of neurons per hidden layer $N_{neurons}$.

For each algorithm and meta-parameter, a tenfold cross-validation is applied to assess the combination performance. A fold uses nine tenths of the items available (training set) to train the method and the remaining one tenth to predict the classes (validation set). The overall performance is evaluated by pooling the validation set predictions and reconstructing the original dataset, these pooled predictions are evaluated against the true classes of items and the ratio of correct predictions is measured. An optimal set of meta-parameters for each algorithm is identified as the one maximising the prediction performance. This cross-validation strategy enables the meta-parameters to be optimised while leveraging the entire dataset and limiting the impact of the single fold division on the measured performance.

SVM with radial basis function kernel

The standard SVM is a two-class linear classifier dividing the features space with a hyperplane. The hyperplane is designed to maximise the margin m, double the distance between the hyperplane and its closest items in the features space, whilst maintaining the correct class division.

The distance between the hyperplane $\vec{w}^T\vec{x} + b$ and item i is:

$$r_i = \frac{\vec{w}^T\vec{x}_i + b}{\|\vec{w}\|} \tag{4.04}$$

If i is one of the closest items, then $\vec{x}_i$ is a support vector and r_i equals $\frac{m}{2}$.

An item can be classified into class 1 or class -1 according to its relative position to the hyperplane:

$$class_i = sign(\vec{w}^T\vec{x}_i + b). \tag{4.05}$$

Following this notation, the optimal hyperplane is not well defined as it can be expressed by infinite combinations of $\vec{w}$ and b. To obtain an unambiguous optimal solution, the following constraint is imposed:

$$|\vec{w}^T\vec{x}_{SV} + b| = 1 \tag{4.06}$$

$\vec{x}_{SV}$ is any support vector.

This constraint does not change the optimal solution but enables m to be expressed in a simple form:

$$m = 2\frac{\vec{w}^T\vec{x}_{SV} + b}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|} \tag{4.07}$$

The constraints bounding the items to their correct class are also expressed in simple terms. The original equations of the constraints are:

$$\frac{\vec{w}^T\vec{x}_i + b}{\|\vec{w}\|} \le \frac{m}{2} \quad if\ y_i = -1 \tag{4.08}$$

$$\frac{\vec{w}^T\vec{x}_i+b}{\|\vec{w}\|} \geq \frac{m}{2} \quad if\ y_i = 1 \tag{4.09}$$

y_i is the true class of item i.

They become:

$$\vec{w}^T\vec{x}_i + b \leq 1 \quad if\ y_i = -1 \tag{4.10}$$

$$\vec{w}^T\vec{x}_i + b \geq 1 \quad if\ y_i = 1 \tag{4.11}$$

The optimisation problem can be expressed with quadratic programming as:

$$min \quad \|\vec{w}\|^2 \tag{4.12}$$

s. t.

$$y_i(\vec{w}^T\vec{x}_i + b) \geq 1 \quad \forall i = 1, \dots, |I'| \tag{4.13}$$

This optimisation problem can be rewritten with the Wolfe Dual as:

$$max \quad \sum_{i=1}^{|I'|} \alpha_i - \frac{1}{2}\sum_{i=1}^{|I'|}\sum_{l=1}^{|I'|} \alpha_i\alpha_l y_i y_l \vec{x}_i^T\vec{x}_l \tag{4.14}$$

s. t.

$$\sum_{i=1}^{I} \alpha_i y_i = 0 \tag{4.15}$$

$$\alpha_i \geq 0 \quad \forall i = 1, \dots, |I'| \tag{4.16}$$

The solution to the primal problem is:

$$\vec{w} = \sum_{i=1}^{|I'|} \alpha_i y_i \vec{x}_i \tag{4.17}$$

$$b = y_l - \sum_{i=1}^{|I'|} \alpha_i y_i \vec{x}_i^T\vec{x}_i \quad \forall l: \alpha_l \neq 0 \tag{4.18}$$

It is possible to implement the SVM in cases when the dataset is not linearly separable. In these cases, the system can incorrectly classify some examples, thus paying a price in the objective function. The incorrect classification penalty is a box constraint C multiplied by the classification error, this SVM is called soft margin SVM.

$$\min \quad \|\vec{w}\|^2 + C\sum_{i=1}^{|I'|} \varepsilon_i \tag{4.19}$$

s. t.

$$y_i(\vec{w}^T\vec{x}_i + b) \geq 1 - \varepsilon_i \quad \forall i = 1, \dots, |I'| \tag{4.20}$$

Its Wolfe Dual is:

$$\max \quad \sum_{i=1}^{I} \alpha_i - \frac{1}{2}\sum_{i=1}^{|I'|}\sum_{l=1}^{|I'|} \alpha_i\alpha_l y_i y_l \vec{x}_i^T\vec{x}_l \tag{4.21}$$

s. t.

$$\sum_{i=1}^{|I'|} \alpha_i y_i = 0 \tag{4.22}$$

$$C \geq \alpha_i \geq 0 \quad \forall i = 1, \dots, |I'| \tag{4.23}$$

With the primal problem solution:

$$\vec{w} = \sum_{i=1}^{|I'|} \alpha_i y_i \vec{x}_i \tag{4.24}$$

$$b = y_l(1 - \varepsilon_l) - \sum_{i=1}^{|I'|} \alpha_i y_i \vec{x}_i^T \vec{x}_l \quad \forall l: \alpha_l \neq 0 \tag{4.25}$$

This SVM classifier can be made nonlinear by substituting the features' internal product $\vec{x}_i^T \vec{x}_l$ with a nonlinear kernel $k(\vec{x}_i, \vec{x}_l)$. This is equivalent to a nonlinear expansion of the features space, what was not linearly separable in the original space might be linearly separable in the new one. The use of a kernel prevents the computational costs associated with the calculation of the new features space.

The kernel used is the radial basis function:

$$k(\vec{x}_i, \vec{x}_l) = e^{-scale \cdot \|\vec{x}_i - \vec{x}_l\|^2} \tag{4.26}$$

The radial basis function is a well-established kernel, capable of achieving a significant performance while dealing with nonlinear datasets.

The SVM classifier can be used in contexts with more than two classes by training a classifier for each pair of classes. When a new item i needs to be classified, it is fed to each SVM and the results are compared using an error-correcting output code model $ECOC$ [81]. In a three-class model, the $ECOC$ is:

$$ECOC = \begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{pmatrix} \tag{4.27}$$

The item i is associated with the class minimizing:

$$min \ \sum_{j=1}^{J} \sum_{l=1}^{J} \frac{max(0, 1 - ECOC_{i,l} \cdot s_i)}{2} \tag{4.28}$$

$\vec{s}_i$ is a logical vector representing the class that item i is classified in. In a three-class model:

$$\vec{s}_i = (1,0,0) \quad represents\ the\ first\ class \tag{4.29}$$

$$\vec{s}_i = (0,1,0) \quad represents\ the\ second\ class \tag{4.30}$$

$$\vec{s}_i = (0,0,1) \quad represents\ the\ third\ class \tag{4.31}$$

Deep neural network

A DNN is a multiclass nonlinear classifier that passes the item features $\vec{x}_i$ through a set of layers and outputs a vector $\vec{s} \epsilon R^K$, where s_k (i.e. the kth component of $\vec{s}$) is the probability that i belongs to a certain class C_k.

As shown in Figure 4.01, each layer is a vector connected to the next one through a matrix of weights W_l and a set of nonlinear functions. In this structure, consider $\vec{x}_i$ and $\vec{s}$ as the input and output layers and $\vec{z}_l$ with $l = 1, \ldots, N_{hidden}$ as hidden layers. Alongside W_l, the nonlinear functions in layer l are obtained by a vector of biases $\vec{b}_l$ not represented in Figure 4.01. Each element of bias b_{lj} enters one of the functions preceding layer l. The last function is linked to layer N_{hidden} by a matrix of weights $W_{softmax}$ and a vector of biases $\vec{b}_{softmax}$.

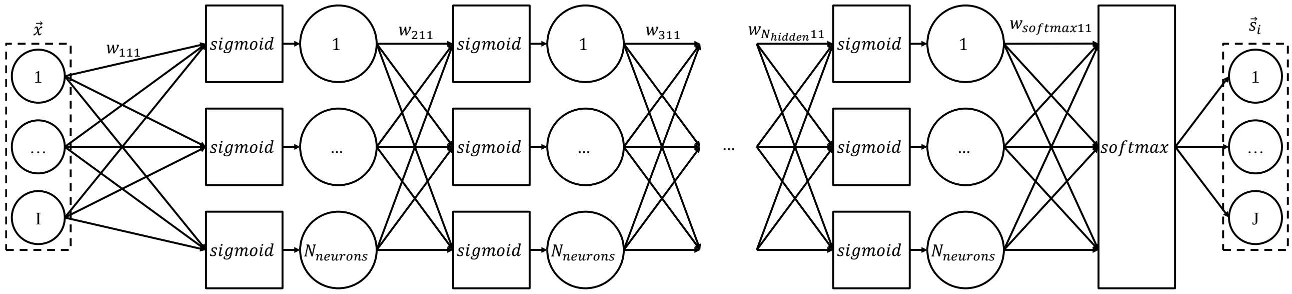

Figure 4.01. Deep neural network structure.

$\vec{z}_1$ is linked to $\vec{x}_i$ by the equation:

$$\vec{z}_1 = sigmoid\left(W_1 \cdot \vec{x}_i + \vec{b}_1\right) \tag{4.32}$$

The $sigmoid$ function of $W_1 \cdot \vec{x}_i + \vec{b}_1$ is:

$$\vec{z}_1 = \frac{\vec{1}}{\vec{1}+e^{-W_1 \cdot \vec{x}_i - \vec{b}_1}} \tag{4.33}$$

The same logic applies to each layer $\vec{z}_l$ with $l = 1, \ldots, N_{hidden}$.

$$\vec{z}_l = \frac{\vec{1}}{\vec{1}+e^{-W_{l-1} \cdot \vec{z}_{l-1} - \vec{b}_{l-1}}} \tag{4.34}$$

$\vec{s}_i$ is linked to $\vec{z}_{N_{hidden}}$ by the $softmax$ function:

$$s_{ik} = \frac{e^{\vec{w}_{softmaxk}^T \cdot \vec{z}_{N_{hidden}} + b_{softmaxk}}}{\sum_{j=1}^{J} e^{\vec{w}_{softmaxk}^T \cdot \vec{z}_{N_{hidden}} + b_{softmaxk}}} \tag{4.35}$$

Each layer is linearly combined using weights and biases to produce the next one. If only linear equations were applied, the result would be a multivariate logistic regression, a linear classifier using $\vec{x}_i$ as an input. This is since a combination of linear functions is a linear function. In order to add nonlinearity, the $sigmoid$ function must be used after each linear combination. The $sigmoid$ collapses each vector component in the range $(-1,1)$, making each $\vec{z}_l$ a nonlinear combination of the previous layer components. $\vec{s}_i$ is a multivariate logistic regression of $\vec{z}_{N_{hidden}}$, a linear classification of the new features as they emerge from the last hidden layer. The use of a linear classifier in the final layer is common to DNN and SVM with the radial basis function kernel; both algorithms expand the features space and use linear classifiers to obtain nonlinear classifications in the original space.

The performance of DNN can be measured by a cross-entropy $cross_i$ function:

$$cross_i = -\vec{y}_i^T \cdot log(\vec{s}_i) \tag{4.36}$$

The $cross_i$ function is minus the log-likelihood of the multivariate logistic regression classifier used in the output layer, the performance of DNN can be optimised by minimising the overall cross-entropy of the training set:

$$cross = \sum_{i=1}^{|I'|} cross_i \tag{4.37}$$

This optimisation is carried out by modifying the weights and biases of each layer, including the output one. In order to know how to modify weights and biases, the gradient of $cross$ is obtained.

Using the multivariable chain rule it is possible to calculate the derivatives of $cross$ over W_l and $\vec{b}_l$ for the layer l from the gradient of $cross$ over $\vec{z}_l$. This facilitates an efficient reuse of calculations with a considerable computational time saving. Once each derivative has been computed, a gradient descend algorithm is applied to move the value of $cross$ downwards, thus increasing the DNN performance on the training set. This procedure is called backpropagation.

Part of the training set (30%) is used as a holdout. It is not directly fed to the backpropagation algorithm for the DNN training but is kept aside to check the $cross$ performance on it as an unseen dataset. This prevents overfitting since the backpropagation is stopped after reaching the lowest $cross$ on the holdout set. In practice weights and biases of the best performing DNNs are memorised as they emerge, and the backpropagation algorithm is stopped after a certain number of suboptimal results. The optimisation algorithm used is a scaled conjugated gradient descent backpropagation due to its effectiveness and not excessive computational complexity.

Experimental analysis

The procedure is applied to two generated datasets:

1. The first dataset contains items with a small relative standard deviation in terms of the positive demands $\frac{E[Std(d_i)]}{E(d_i)}$. This feature produces time series that are less intermittent.
2. The second dataset contains items with higher values of $\frac{E[Std(d_i)]}{E(d_i)}$, and the resulting time series are more intermittent since the positive demand quantities are less predictable.

Each time series is obtained using two probability distributions, i.e. a Bernoulli distribution generating the positive demand events and a normal distribution generating the quantities demanded when a positive demand takes place. When in the series generation, a positive demand assumes values lower than zero, as a result of the unrestricted normal distribution domain, this demand is approximated to zero in the first dataset, whereas it is recalculated in the second one.

Experimental setting

Two experimental datasets are generated, the first one composed of 28,350 time series and the second one containing 39,690 time series. Each time series is randomly generated on a time horizon of $T = 40{,}000$ periods with a warm-up of $n = 20{,}000$ periods.

The following features, uniformly distributed between different ranges, generate each time series:

- $E(p_i)$ between 0.2 and 0.7 with linear steps of 0.1.
- $E(d_i)$ between 100 and 500 with linear steps of 50.
- $\frac{E[Std(d_i)]}{E(d_i)}$ between 0.1 and 0.3 with linear steps of 0.05 for the first dataset and between 0.7 and 1 with linear steps of 0.05 for the second one.

Non statistical features:

- $h_i \cdot C_i$ between 2 and 14 with linear steps of 4.
- L_i equal to 2.
- $\phi_i^{-1}(tCSL_i)$ between 1 and 4 with linear steps of 0.5.

Where C_i is the unitary purchasing cost of item i, h_i is its unitary holding cost and the unitary ordering cost o_i is obtained defining the theoretical $R_{th,i}$ equal to 3, 5 and 7 [38].

Exhaustive simulation and classification

Given a dataset, each item time series is exhaustively simulated through the inventory system. A warm-up of $n = 20.000$ periods is used to optimize, at the single item level, the smoothing coefficients, thus minimising the mean squared error between forecast and demand. Three values of R (equal to 3, 5, and 7 periods) are tested for each time series and the value leading to the minimum $TRC_{i,R}$ is selected. At the end of this procedure, each item has both a set of classification criteria and an experimentally optimal value of R, named $R_{opt,i}$.

Six classification experiments are conducted on each dataset, each experiment involves a different combination of numbers of cross-validation training folds and classifiers. Since the folds are used for training purposes, the higher the number of folds, the lower the ratio of items used in each training set.

Dataset 1	Simulation 1	Simulation 2	Simulation 3
SVM	50 folds	20 folds	10 folds
DNN	50 folds	20 folds	10 folds
Dataset 2	Simulation 1	Simulation 2	Simulation 3
SVM	50 folds	20 folds	10 folds
DNN	50 folds	20 folds	10 folds

Table 4.01. Number of classification experiments.

Table 4.01 outlines all the experiments, for instance the 20-fold SVM test divides a dataset into 20 folds. In this case 5% of the dataset items (training set) are used to train a Gaussian kernel SVM which is then used to classify the items outside the fold (95% of the dataset items, i.e. test set), the resulting classes are compared with the real test set classes to obtain performance measures. This procedure is repeated for each fold and, at the end, the single performance measures are averaged.

Two performance measures are obtained for each experiment:

- The average correct classification ratio.
- The average confusion matrix.

The correct classification ratio is the ratio of correctly classified items in the test set over the total number of items in the test set. The average correct classification ratio is the average of these classification ratios over the number of folds.

The confusion matrix is a $K \times K$ matrix containing a ratio of the test set items in each cell. The cell i, j for instance contains the number of items that the simulation sorted in class i and the classifier classified in class j, divided by the total number of items in the test set. The trace of the confusion matrix equals the correct classification ratio. The confusion matrix belonging to different folds is averaged cell-wise, weighting for the number of items in each test set, to obtain the average confusion matrix.

The meta-parameters analysed in each training fold to train an SVM are:

- C distributed between 10^{-5} and 10^{5} with geometrical steps of magnitude 10.
- $scale$ distributed between 10^{-5} and 10^{5} with geometrical steps of magnitude 10.

The meta-parameters analysed in each training fold to train a DNN are:

- N_{hidden} distributed between 1 and 3 with linear steps of 1.
- $N_{neurons}$ distributed between 10 and 100 with linear steps of 10.

This meta-parameter optimisation follows the procedure previously outlined and is carried out inside each training fold without using any test fold sample.

Results

Table 4.02 contains the average correct classification ratio for each experiment, Table 4.03 contains the average confusion matrices. The sum of probabilities exceeding 1 result from the rounding process.

	Dataset 1			Dataset 2		
SVM	0.972	0.977	0.982	0.972	0.980	0.983
DNN	0.954	0.972	0.978	0.963	0.975	0.981

Table 4.02. Average correct classification ratio.

		Dataset 1											
		Simulation 1				Simulation 2				Simulation 3			
			Predicted				Predicted				Predicted		
			A	B	C		A	B	C		A	B	C
SVN	True	A	0.538	0.006	0.000	True	0.539	0.006	0.000	True	0.540	0.004	0.000
		B	0.010	0.227	0.006		0.009	0.230	0.004		0.009	0.232	0.003
		C	0.000	0.006	0.207		0.000	0.004	0.208		0.000	0.002	0.210
DNN			Predicted				Predicted				Predicted		
	True	A	0.528	0.014	0.003	True	0.537	0.007	0.000	True	0.539	0.006	0.000
		B	0.011	0.222	0.010		0.009	0.227	0.007		0.009	0.231	0.004
		C	0.000	0.009	0.203		0.000	0.005	0.207		0.000	0.003	0.209
SVN		Dataset 2											
			Predicted				Predicted				Predicted		
	True	A	0.670	0.008	0.000	True	0.673	0.005	0.000	True	0.674	0.005	0.000
		B	0.008	0.196	0.006		0.006	0.200	0.004		0.005	0.201	0.004
		C	0.000	0.006	0.106		0.000	0.005	0.107		0.000	0.004	0.108
DNN			Predicted				Predicted				Predicted		
	True	A	0.667	0.011	0.000	True	0.671	0.007	0.000	True	0.673	0.005	0.000
		B	0.011	0.191	0.008		0.007	0.1972	0.005		0.005	0.200	0.005
		C	0.000	0.007	0.105		0.000	0.005	0.107		0.000	0.005	0.107

Table 4.03. Average confusion matrix.

Three phenomena can be assessed by analysing Table 4.02:

1. The average correct classification ratio increases as the percentage of items in the training set increases, these increments are small but are present in each combination of dataset and classification methodology.

2. The second dataset, which is the most intermittent, achieves a better performance than the first. This is true for each combination of classification methodology and training percentage.
3. The SVM outperforms the DNN in each combination of dataset and training percentage, the differences are small but always in favour of the SVM.

Figure 4.02 plots the results of Table 4.02 against the number of items in the average training set. It shows that the first and second phenomenon listed above are generated by a single cause: the average correct classification ratio scales with the number of items in the average training set. The differences between datasets alone are not obvious from the plot and the performance seems to scale more consistently with the absolute number of items in the average training set than with its relative one. Figure 4.02 shows how the performance gap between SVM and DNN depends on the number of items in the average training set, and the gap shrinks as the number of items increases. Despite this trend, SVM always outperforms DNN.

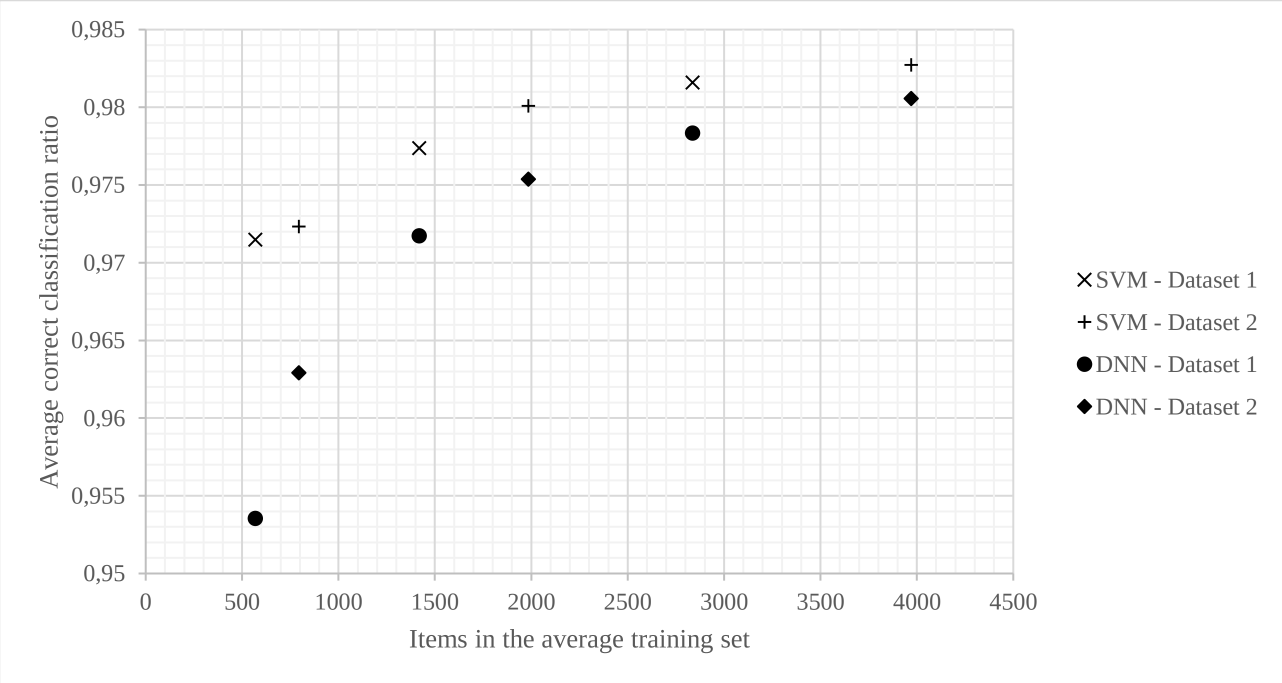

Figure 4.02. Average correct classification ratio over the number of items in the training set.

The average confusion matrices in Table 4.03 show that, despite the imbalance between classes, the classifiers do not overfit the most common class $R = 3$. This finding is consistent throughout the datasets, classifiers and training percentages. Since no preventive measures have been taken beforehand to prevent overfitting phenomena, the classifiers appear robust in dealing with unbalanced classes.

	Ratio
Dataset 1	0.725
Dataset 2	0.561

Table 4.04. Theoretical classification ratio.

Table 4.04 shows the theoretical correct classification ratio for both datasets. This is obtained by comparing the theoretical classes [38] with those emerging from the simulations. The theoretical model performance in both datasets is worse than that achieved by the supervised classifiers and decreases as the data become more intermittent. This is exemplified by the 0.164 performance drop between the first and second datasets. The theoretical confusion matrixes in Table 4.05 show how as the items become more intermittent, they shift away from their theoretical classification towards lower values of the best R_i. The classification algorithms can adapt and catch this drift, with the subsequent increased class imbalance, while the theoretical classification does not.

Dataset 1					Dataset 2				
		Predicted					Predicted		
		A	B	C			A	B	C
True	A	0.333	0.000	0.000	True	A	0.333	0.000	0.000
	B	0.154	0.180	0.000		B	0.218	0.115	0.000
	C	0.058	0.064	0.212		C	0.127	0.094	0.112

Table 4.05. Theoretical confusion matrix.

Conclusions

Managing many items involves both MCIC methods and inventory control theory. A common solution is to classify similar items first and then define a unique inventory control policy for all the items belonging to a class. The tasks of classifying and finding appropriate control policies for the classes are generally kept separate, and as a result the original objective is often forgotten. When the items demonstrate intermittent consumption, this issue is further exacerbated.

In this chapter, given a forecast-based periodic review inventory system, an exhaustive search on a sample of items is performed to obtain in the first stage, their best classification from an inventory perspective. This step is coupled with SVM and DNN algorithms to classify the out-of-sample items, using the classification as an optimum reference along with the relevant set of criteria from an inventory control perspective.

The overall approach was validated through a large experimentation with satisfactory results, both classifiers lead to significant improvements in the classification accuracy in comparison with a theoretical approach.

This chapter attempts to bridge the gap between inventory theory and MCIC since they are strictly interrelated research fields. The decision to apply this methodology to intermittent demand is due to the highest analytical and operative criticality of this type of demand pattern, however the adopted inventory system is flexible, and the overall framework is general and thus applicable to other inventory systems and demand patterns.

It is worth highlighting the importance of adopting an exhaustive search in order to fix a set of reference items as inputs for the classification procedures. This is in line with case-based reasoning for MCIC, however the reference items are optimally classified here and not subjectively established by decision-makers. In other words, the classification of the in-sample items, which affects the overall classification, is free from human error. The experimental research confirms the positive impact of the exhaustive search and highlights the importance of this phase.

The proposal has some limitations that could be the subject for further research:

- This chapter adopts a single-item forecast-based inventory system. In the case of items used in an assembly process, a multi-item inventory system would be more appropriate. The forecasts could be made for the end products and then propagated to the components according to the bills of materials.
- The analysed datasets were generated in order to better control the experimental procedure. However, real data might enrich the validation of this approach.
- Other inventory systems could be evaluated and compared with the one implemented here.

3. Decision trees and random forests for inventory classification

This chapter is adapted from the paper "Decision Trees for Supervised Multi-criteria Inventory Classification" published in 2017 in Procedia Manufacturing, volume 11, pages 1871-1881 [82].

Introduction

This chapter faces a problem equivalent to the one tackled in Chapter 2, it attempts bridge the gap between MCIC and inventory control theories when the exhaustive classification is impractical on the whole set of items. In this case the classification rules will be generated through supervised classifiers well established into machine learning field by starting from the exhaustive solution on a subset of items, on which the classifiers are trained. Decision trees and random forests are compared as effective tools for overcoming the main concerns of MCIC, which are:

- The need for a set of predefined criteria that are not robustly linked with the inventory control system.
- The predefined cardinalities of the generated classes defined a priori without any justification.

Among the machine learning techniques available for classification purposes, decision trees and random forests have been selected for theoretical simplicity and readability of the results. The connection between input features and obtained results in Chapter 2 methods, for instance, is harder to analyse. Examining a decision tree, it is possible to rank the splits and obtain a visual representation of what features are the most impacting in the classification process. Said features can be monitored by the management, controlling unwanted shifts towards categories more expensive to manage. For the intermittent spare parts related to new products, attention on the most critical features can drive the design phase. Standardization procedures should be tuned not only to group as many components as possible but also to substitute hard to manage components with those belonging to the least expensive classes. A similar analysis can be carried on in a random forest scenario using a ranking averaged among the different trees composing the forest.

Methodology

Given a set of items and an inventory system composed by more reorder policies, the objective is to associate each item to one reorder policy. An exhaustive simulative approach is applied to a subset of items in order to achieve optimal reorder policies. The optimal classification for the non-simulated items is subsequently obtained by means of two supervised classifiers: the decision tree and the random forest.

Step 1: selection of the forecast system.

Given an item with a known demand history, this must be managed by means of an inventory policy. In stochastic systems, the input of the inventory policy is the forecast of the future demand, determined by applying a forecast system to the item demand history. In first that forecast system is selected.

Step 2: selection of the inventory control system.

It is selected an inventory control system to be used as a reference. For each item, this inventory control system can be applied with different system parameters, which are item specific and thus different choices lead to different performance. This chapter does not specifically focus on the choice of the inventory control system, being the methodology not dependent by such a choice. It is to note that the set of parameter values or possible combinations of parameter values determines the number of the classes. For instance, in an inventory system constrained by a target service level $tCSL_i$, supposing that $tCSL_i$ is the only system parameter, if three values of $tCSL_i$ are chosen (e.g. 99%, 95%, and 90%) then three classes of items have to be generated.

Step 3: exhaustive simulation search.

An exhaustive search is performed by varying the adopted system parameter on a randomly chosen subset of items. Each item in the subset is simulated over its historical demand, applying the inventory control system defined in Step 2 based on the forecasts obtained with the forecast system defined in Step 1. For each simulation a performance measure related to the system costs or service level is calculated. This leads to the identification, for each analysed item, of the system parameter value optimizing the performance measure. Each analysed item is then put into the class characterized by the system parameter or combination of system parameters capable of optimizing the inventory control system performance for the items in said class.

Step 4: choice of classification criteria.

A set of criteria able to characterize the items under analysis is selected. For example, in the field of intermittent demand, several criteria have been proposed in literature for classification purposes, especially for spare parts management. Some criteria refer to the demand pattern, like the coefficient of variation and the average demand interval [32], while others to the items features (e.g. criticality). Each criterion is a dimension the decision tree can split along, in order to classify the items into the classes defined before.

Step 5: training folds definition.

The subset of items defined in Step 3 is divided into m folds. Such subdivision is generated casually while respecting the classes frequencies in each folder. This last strategy is taken in order to add variability to the folds while coping with significant frequencies differences among classes, otherwise some folds might not contain all the classes or only few elements of the last represented classes might be selected.

Step 6: decision tree.

A decision tree is trained on each fold, and each tree node can binary split the items on a single criterion among those presented in Step 4. The split is performed by following a predetermined splitting rule (e.g. Gini index). Each decision tree is tested against the items not belonging to its training fold, and the test is performed both for the fully-grown tree and for all its pruned versions. For each fold and pruning level a performance in terms of percentage of items correctly classified in the test phase is memorized. For each pruning level, its average performance among the folds is calculated and the pruning level leading to the best performance is selected. The items belonging to all the folds are then used to train a final tree, pruned up to the best pruning level.

Step 7: random forest.

Instead of a decision tree, a random forest approach can be applied to the same folds. A set of random forests is trained on each fold and tested on the items not belonging to the fold at hand. A set of random forests is designed by varying the number of trees in the forests and the number of criteria randomly chosen for splitting the items at each node. Each random forest yields a classification performance in the test phase, thus the performance of the combination number of trees and number of criteria, in terms of percentage of items correctly classified, can be identified. Given a combination the average performance among all the folds is calculated and the combination yielding the best performance is selected. A final random forest in trained, using the items belonging to all the folds with the best combination of number of trees and number of criteria for splitting the items at each node.

Step 8: classification of the remaining items.

At the end of Steps 6 and 7, a decision tree and a random forest with optimal parameters trained on all the simulated items is available. The items not used for training are those not simulated in Step 3; such items are classified by means of these classifiers. Each class is tied to a set of parameters, as defined in Step 2, and the classification leads to the identification of the best possible parameters for each item to be used in that item inventory control system.

Experimental validation

Data overview

A firm producing make-to-stock electric resistances through a multi-stage assembly system has thousands of items delivered by its suppliers. Among these items, a high percentage (more than one thousand) shows weekly intermittent consumptions due to the high heterogeneity of the final customers and the large portfolio of final products. These items cannot be managed in a push system through the Material Requirement Planning (MRP) because their replenishment lead times are higher than the maximum allowed by the constraints of the assembly process and the time-to-market. A representative sample of 104 items is selected, and their demand history data are collected over 150 weeks.

The main features of each item are summarised in the sequel:

- L_i is replenishment lead time of 2, 4 or 6 weeks.
- C_i is the unitary purchasing cost varying among the 104 items between 0.10 € and 9.08 €.
- h_i is the unitary holding charge fixed at 1% of C_i per week.
- o_i is the ordering cost fixed at 4.40 € per order for any item.

Table 5.01 reports the main descriptive statistics of the demand history over the 150 weeks:

- The minimum and maximum mean $E(d_i)$ and variance $Var(d_i)$ of the positive demand among all the time series.
- The minimum and maximum mean $E(z_i)$ and variance $Var(z_i)$ of the demand per period (both positive and null).
- The minimum and the maximum ratio of null demands $1 - p_i$.

It is to notice the great heterogeneity of the items, which underlines the need for their classification and for the subsequent differentiation of their inventory system parameters.

Extreme values	$E(d_i)$	$Var(d_i)$	$E(z_i)$	$Var(z_i)$	$1 - p_i$
Min	0.297531	0.068025	0.063899	0.029109	0.12
Max	2555.409	3672179	2244.544	3922768	0.95

Table 5.01. Descriptive statistics of the dataset.

In order to measure the performance of decision trees and random forests in this scenario, all the items are exhaustively simulated, and a leave-one-out approach is used. For each item, all the remaining items are used to generate the final decision tree (Step 6), and the final random forest (Step 7). The item not used in the training step is then classified with both classifiers (Step 8), and its predicted class is memorized. At the end of the experimental validation, a predicted class for each methodology is attached to each item. These classes can be compared with those defined by the simulation (Step

3), and performance measures can be calculated accordingly. In order to assess the performance increase determined by the simulation step, that makes possible the use of supervised classifiers as decision trees and the random forests, the proposed methodology is compared with an unsupervised MCIC approach, that is not involving the simulation step.

Simulation overview

The analysed items present an intermittent consumption. Said historical demand pattern is managed in this experiment by applying Croston [1] and SBA [12]. The exponential smoothing of the MSE of forecasts is used as estimator of the demand variance, in accordance with [79]. The historical demand is used to assess, for each item, which forecast system performs better and to define its smoothing coefficients. In this chapter four different smoothing coefficients [82] can take two values: 0.1 and 0.2, because higher values are not recommended in the literature [79]. All the combinations are tested for both Croston and SBA on a warm-up period of the first half of time series (75 observations). The initial forecast $F_{i,t}$ is initialised with the averages of the first three observations per each time series. It is then chosen for each item the combination of forecast system and smoothing coefficients leading to the lowest MSE at the end of the warm-up period.

A forecast-based periodic order-up-to level policy (R, S) is adopted for all the items. A pure cost-based model is not suitable due to the uncertainty of backorder costs, and therefore a service constrained model is preferred. In the model selected, the review interval is the only system parameter to optimize by means of an exhaustive search, and the company managers are interested in renegotiating R_i with the suppliers for each item i. The review interval is tied to contractual agreements because it determines how often a replenishment order can be issued. An order is issued every time, in a review period, the inventory position falls under $S_{i,t}$. In the current experimentation R_i can be set to 2, 4 or 6 weeks; these values are exhaustively tested with the already optimised smoothing coefficients by means of a discrete event simulation working on weekly periods. The simulation operates from the first observation, but results are collected only on last 75 observations. At the end of the simulation a cost-oriented performance measure equivalent to the one presented in Chapter 2 is computed and all the other simulation parameters not previously defined are exogenously assigned:

- $tCSL_i$ relates to the criticality level of the item i. In this case the managers assign a $tCSL_i$ equal to 90% to all the items.
- ϕ_i is a standardised normal distribution, and the safety factor $\phi_i^{-1}(tCSL_i)$ is thus also known.

Classification overview

The criteria chosen for classification are quantitative and impact directly the inventory control system functionality:

- The unitary purchasing cost C_i. Given the fixed unitary holding cost h_i, only C_i affects the total inventory cost.
- The replenishment lead time L_i. This feature is specific for each item and depends on its supplier, the higher L_i, the more critical the item.
- The mean positive demand $E(d_i)$.
- The number of null demands during the simulation period $1 - p_i$.

Among these criteria $E(d_i)$ and $1 - p_i$ are calculated only with the last 75 observations of the simulation. The purpose is to avoid distorted relationships of the reorder policy with $E(d_i)$ and $1 - p_i$. These values may significantly change over time, while the inventory policy overcomes the warm-up period. Other criteria like ordering cost and holding charge are not considered because they are the same for all the items, and thus they do not affect the classification.

Training folds definition

Each item is analysed with a leave-one-out approach. Given a reference item, the remaining items are divided into three balanced folds of 34, 35 and 35 items respectively in order to optimize and train both a decision tree and a random forest. The frequency of each class is approximately respected in each fold. The original dataset is composed by 104 items, 74 of which belong to class 2, 20 to class 4 and 10 to class 6. The characteristics of each fold are outlined in Table 5.02.

Fold	Number of items in each class			
	2	4	6	Total
Fold 1	24	7	3	34
Fold 2	25	7	3	35
Fold 3	25	6	4	35

Table 5.02. Number of items in each fold divided by class.

Decision tree

Given a fold, its decision tree is grown with the following parameters:

- The criterion chosen for splitting at each node is the Gini Index.
- The minimum number of items in a branching node is 2.
- The pruning sequence follows the classification errors.

The Gini index for splitting and the classification errors for pruning are chosen as common methodologies for the decision trees. The number of items in branching nodes is kept low in order to let the trees grow unconstrained, and the reduction of the trees size is left to the pruning phase optimized among the folders.

Random forest

Given a fold, its random forest is grown with the following parameters:

- The criterion chosen for splitting at each node is the Gini Index.
- The minimum number of items in a branching node is 2.
- No pruning is performed on the forest trees.

The trees belonging to a forest are left unpruned, and the overfitting tendencies of the individual trees is countered by the number of trees balancing each other biases and by the random choice of characteristics at each split. Both these parameters are optimized among the folders (Step 7).

Classification of the remaining items

Each decision tree and random forest classifies the item not used for its training, leading to the confusion matrix reported in Table 5.03.

Classification methodology	Predicted			
		2	4	6
Decision tree	2	66	6	2
	4	9	10	1
	6	2	2	6
Random forest	2	68	4	2
	4	12	8	0
	6	3	2	5

Table 5.03. Supervised methods confusion matrices.

Comparison with an unsupervised MCIC approach

These results reported in Table 5.03 are compared with those obtained with the LBL-model proposed by [67], and presented in Table 5.04.

Classification methodology	Predicted			
LBL		2	4	6
	2	61	9	4
	4	12	6	2
	6	1	5	4

Table 5.04. Unsupervised method confusion matrix

The LBL-model is unsupervised, and its class cardinalities are to be established a priori. In order to compare its best performance with the results achieved by the decision trees and the random forests, the optimum cardinalities reached through the exhaustive search are fixed, i.e. 74-20-10 respectively for the classes 2, 4 and 6.

The LBL-model is a DEA-based optimization approach that considers both a good and a bad index for each item and can be solved without an optimizer. It needs a composite index to be set, defined in the current case by assigning equal weights to the good and bad indexes, and a ranking of the classification criteria. In order to obtain the LBL-model best performance, all the twenty-four possible rankings of criteria (i.e. 4!) are tested and only the one leading to the best classification is considered.

The degree of agreement between different classification methods is established by means of the number of misclassified items from one class to another. A single misclassification is verified if a 2-class and a 4-class item is respectively classified into class 4 and class 6, and vice versa. A double misclassification arises when an item of class 2 is classified into class 6, and vice versa. The total number of misclassifications represents an accuracy measure for the methods under comparison (Table 5.05).

	Decision tree	Random forest	LBL
Misclassifications	26	28	38
Misclassifications percentage compared to LBL	68%	74%	100%
Items incorrectly classified	22	23	33
Percentage of items incorrectly classified	21%	22%	32%

Table 5.05. Misclassification performance comparison.

The number of misclassifications reached by decision trees and random forests is lower than those achieved by the best possible case of LBL model usage, obtaining respectively 32% and 26% less misclassifications. These findings are case sensitive but show promising results for decision trees and machine learning techniques adoption into MCIC field. Table 5.05 presents also the same results from a different perspective showing for each methodology the raw percentage of items incorrectly classified, without the misclassification weighting. Both perspectives show similar results for the decision tree and the random forest methodologies, while the LBL technique presents significantly worse performance. This comparison suggests that the implemented machine learning methodologies impact mostly the items misclassified by the LBL technique with a reclassification capable of finding their correct place, the effect is not limited to a reclassification of said items in slightly less incorrect classes.

Conclusions

When managing many items, MCIC is widely used to classify similar items and associate to each class a specific inventory control model. Different CSL, typologies of reorder policies as well as lengths of review interval may be used for differentiating the inventory control system among the generated classes. However, the item classification remains opaque if not jointly optimised with the inventory system. If MCIC is tackled separately, the classification criteria and the cardinalities of the classes must be established a priori without any robust assessment from an inventory cost perspective.

In this chapter, decision trees and random forests are adopted to joint optimize MCIC and inventory systems. To generate the needed training set, an exhaustive simulative approach is performed on a sample of items while the trained decision tree and random forest indicate for each non-simulated item the class to which it belongs.

A real case study referring to a firm producing electric resistances is used for validating this proposal on a set of 104 items with intermittent demand. The exhaustive classification adopts a well-known periodic inventory control system [79] where the classes are defined on the basis of the length of the review intervals. Four classification criteria are then considered, and the most accurate tree and random forest is searched by a tenfold approach. The comparison with a recent unsupervised MCIC method [67] confirms the better performance of the proposed methodology.

It is to be noted that the performance of the compared unsupervised model is measured at its peak while the decision tree and random forest performance refer to standard working conditions. The correct class cardinalities are provided to the LBL model, but not to the decision tree or the Random forest, being unknown in real settings. The LBL criteria ranking is also chosen in order to optimize the LBL performance according with the simulation results obtained for all the items. Such optimization is unfeasible in real settings just because the exhaustive simulation is impracticable, which makes it necessary to adopt unsupervised classification methods.

The experimental investigation on other inventory control systems for intermittent demand, e.g. the binomial approach [3], as well as other machine learning tools are considered as a part of the further research agenda.

4. K-means and Ward's method for inventory clustering

This chapter is adapted from the paper "Clustering for inventory control systems" published in 2017 in IFAC-PapersOnLine, volume 51, issue 11, pages 1174-1179 [83].

Introduction

When an inventory control system is difficult to model, as in the intermittent demand case, simulation techniques can be exploited to optimize the implemented policies. The time series are simulated multiple times with different settings to identify the best inventory control system parameters. This simulation phase can be very computationally expensive and supervised classification techniques are implemented in Chapters 2 and 3 to reduce the simulation workload to a subset of items. The aim of this chapter is to further reduce such a workload by implementing clustering techniques and dividing the items in a non-supervised fashion before the simulation occurs. This initial subdivision allows for the simulation of a limited number of items in each cluster, which leads to considerable time savings.

Inventory control system

Periodic inventory control system

The inventory control system used in this chapter is periodic: for each item i an order is placed each R_i periods and requires L_i periods to arrive. This policy evaluates both unitary holding costs $c_i = C_i \cdot h_i$ and unitary order costs o_i, the holding costs are paid each period per stocked unit, while the order costs are paid whenever an order is placed. The total relevant cost per period TRC_i is:

$$TRC_i = c_i \frac{S_i}{2} + \frac{o_i}{R_i} \tag{6.01}$$

S_i is the order up to a level, and can be written as:

$$S_i = E(d_i) \cdot p_i \cdot (R_i + L_i) \tag{6.02}$$

$E(d_i)$ is the expected positive demand (with standard deviation σ) for a single period, and p_i is the probability a positive demand occurs. This notation can model both intermittent and non-intermittent items.

TRC_i can be derived and equated to zero to minimize it, thereby obtaining $R_{best,i}$ as the optimal value for R_i:

$$R_{best,i} = \sqrt{\frac{o_i}{c_i} \cdot \frac{2}{E(d_i) \cdot p_i}} \tag{6.03}$$

The model does not consider safety stock costs, if a safety stock is required, it is defined after the optimization of TRC_i. The safety stock requirements can be expressed in terms of $tCSL_i$, the probability of there being a stockout during $R_i + L_i$ periods. A value of $tCSL_i$ equal to 0.5 leads to no safety stock.

Simulation

While the inventory control system analysed can be optimized analytically, the proposed methodology is general and can be also applied to policies that are analytically unsolvable. From a different perspective, even with analytically tractable inventory control systems, real items might not behave exactly like the hypothesized theoretical model since their $R_{best,i}$ could be shifted.

Both these issues can be solved by exhaustive simulation. The historical series of an item is simulated multiple times with the chosen periodic inventory control system and different values of R_i. The simulation leading to the lowest empirical value of TRC_i defines $R_{best,i}$. The only drawback of simulation is that it is time consuming, particularly the simulation of multiple values of R_i for each item. The proposed methodology deals with this limitation.

Clustering algorithms

K-means

The K-means algorithm [84] is one of the most commonly-used methods for clustering.

Given for each item i out of n a vector of features $\vec{x}_i$ and given a predetermined number of clusters K the algorithm:

1. Randomly initializes K cluster centres $\vec{x}_k$.
2. Assigns each item, with features $\vec{x}_i$, to the nearest cluster C_k using Euclidian distances:
   $C_k = \vec{x}_i : \|\vec{x}_i - \vec{x}_k\| \leq \|\vec{x}_i - \vec{x}_j\| \quad i = 1, \dots, n \; \forall j \epsilon \{1, \dots, K\}$ (6.04)
3. If the iteration is not the first one and at least one item changed cluster in step 2, the cluster centres are recalculated:
   $\vec{x}_k = \sum_{i \in C_k} \vec{x}_i$ (6.05)
   When cluster centre is in its items' mean the algorithm moves to step 2.
   If no item changed cluster in step 2, the algorithm terminates.

The K-means is guaranteed to converge. However, different choices for the initial cluster centres can lead to different end results. In order to weight all the features fairly, the K-means is used on normalized features. The algorithm produces spherical clusters and linear divisions between clusters in the features space.

Ward's method

Ward's method [85] is a hierarchical clustering algorithm that minimizes the sum of square errors within clusters.

Given the item features, the method:

1. Initializes a cluster for each item and defines $\vec{x}_k$ with $k = 1, \dots, K'$ initial cluster centres where initially $K' = n$.
2. Calculates the sum of squared errors between clusters as:
   $SE(\vec{x}_k, \vec{x}_j) = \|\vec{x}_k - \vec{x}_j\|^2 \quad \forall k, j = 1, \dots, K'$ (6.06)
   Merges the clusters with the minimum sum of squared errors between and calculates the new cluster centre:
   $\vec{x}_k = \sum_{i \in C_k} \vec{x}_i$ (6.07)
   This step can be simplified using a recursive algorithm [86] that avoids an explicit calculation of the new $\vec{x}_k$.
   Step 2 is repeated until the desired number of clusters K is reached.

The minimization of the sum of squared errors between merged clusters coincides with the minimization of the sum of squared errors within the remaining clusters, and the sum of squared errors of the system is fixed. As in the K-means case, the algorithm produces spherical clusters and linear divisions between clusters in the features space.

Proposed methodology

The clustering algorithms are applied to a set of items following an inventory control system. The items are not exhaustively simulated, instead the clustering algorithms operate on the items' features by dividing them into clusters with homogeneous $R_{best,i}$. The hypothesis is that clusters obtained using meaningful features in terms of inventory control can be used to define inventory control systems. In practice, if the items fall into clusters with homogeneous $R_{best,i}$, it is possible to characterize each cluster afterwards by simulating just a limited number of items, which thereby decreases the overall simulation time. Once a $R_{best,i}$ is assigned to a cluster, the items within the cluster are managed with an optimized periodic inventory control system with a $R_{best,i}$ review time.

When an analytical solution is available, but the data do not follow the theory exactly, a theory-based feature transformation might be advantageous to simplify the feature space for the clustering algorithms. The policy previously outlined, for example, can be rewritten in spherical coordinates:

$$r_i = \sqrt{o_i^2 + (c_i \cdot E(d_i) \cdot p_i)^2} \tag{6.08}$$

$$\alpha_i = arctan\left(\frac{o_i}{c_i \cdot E(d_i) \cdot p_i}\right) \tag{6.09}$$

$$R_{best,i} = \sqrt{2} \cdot \sqrt{tan(\alpha_i)} \tag{6.10}$$

In this coordinate system, $R_{best,i}$ is a linear function of the single variable $\sqrt{tan(\alpha_i)}$.

Experimental setting and results

In this experiment 625 items are generated by equally spacing 25 values of r_i, 25 values of $\sqrt{tan(\alpha_i)}$ and computing all their combinations. The extreme values for both features are listed in Table 6.01, the limits of $\sqrt{tan(\alpha_i)}$ are chosen to ensure theoretical values of $R_{best,i}$ equally spaced between 2 and 8. A value of c_i and p_i within intervals is randomly assigned to each item.

		Min	Max
Fixed features	L_i	2	
	$tCSL_i$	0.5	
	$\frac{\sigma_i}{E(d_i)}$	0.5	
Equally spaced features	r_i	100	200
	$\sqrt{tan(\alpha_i)}$	1.35	5.70
Randomly generated features	c_i	0.1	0.6
	p_i	0.5	0.7

Table 6.01. Simulated feature ranges.

The remaining features needed for simulation and clustering are computed inverting the previous equations. Each item is assigned the same $tCSL_i$, L_i, and relative standard deviation $\frac{\sigma_i}{E(d_i)}$, as defined in Table 6.01.

From the features p_i and $E(d_i)$, an intermittent historical series of 200 periods is generated for each item. Each period has probability p_i of presenting a positive demand and, when a positive demand takes place, its size is normally distributed with expected value $E(d_i)$ and standard deviation σ_i. All the historical series are simulated three times, using the inventory control system previously outlined and values of R_i equal to 3, 5 and 7 respectively. After the simulations, the items are assigned to three homogenous classes according to the value of R_i minimizing their empirical TRC_i. The value of R_i of each class is an estimate of its items' $R_{best,i}$.

After the division into classes, the overall number of items is reduced. The clustering algorithms group items by similarity in the features space and divide them if they are dissimilar. The generated items are homogeneously spaced in the spherical coordinates, thus there is no clear division between clusters. From a different perspective, only three integer values of R_i are simulated, while most items could benefit from intermediate real values of $R_{best,i}$. Therefore, the difference in the simulated TRC_i for two subsequent R_i is sizable for some items and small for others, and this difference is measured as:

$$\Delta trc_i = \begin{cases} |TRC_i - TRC_{R2}| & if\ R_{best,i} \epsilon \{3,7\} \\ min\{|TRC_i - TRC_{R1}|, |TRC_i - TRC_{R3}|\} & if\ R_{best,i} = 5 \end{cases} \quad (6.13)$$

Where TRC_{R1} is the empirical average total cost per period when the item is simulated with $R_i = 1$, TRC_{R2} is the cost when $R_i = 2$, and TRC_{R3} is the cost when $R_i = 3$. Only 25% of items per class with the highest Δtrc_i are kept in order to differentiate the classes and retain only the most meaningful items.

The selected items are clustered with three different sets of features using both the K-means and Ward's method. The three sets of features are designed as follows:

- All the inventory control system features (o_i, c_i, $E(d_i)$, $\frac{\sigma_i}{E(d_i)}$, p_i, L_i, $tCSL_i$), normalized between 0 and 1.
- The two combined features o_i and $c_i \cdot E(d_i) \cdot p_i$, normalized between 0 and 1.
- The transformed features r_i and $\sqrt{tan(\alpha_i)}$, normalized between 0 and 1.

The results for the three sets are reported respectively in Tables 6.02, 6.03, and 6.04. Figures 6.01 and 6.02 plot the clustering obtained using the second set of features for the K-means and Ward's method respectively. Figure 6.03 plots the clustering obtained using the third set of features since the solution for both algorithms is the same.

	$R_{best,i}$	3	5	7
K-means	Precision	0.91	0.41	0.54
	Recall	0.59	0.57	0.54
	F-measure	0.72	0.48	0.54
Ward's method	Precision	0.95	0.42	0.58
	Recall	0.70	0.45	0.69
	F-measure	0.81	0.43	0.63

Table 6.02. All the features results.

	$R_{best,i}$	3	5	7

K-means	Precision	1	0.46	0.52
	Recall	0.83	0.49	0.58
	F-measure	0.91	0.48	0.55
Ward’s method	Precision	0.98	0.49	0.57
	Recall	0.94	0.73	0.33
	F-measure	0.96	0.59	0.41

Table 6.03. The combined features results.

	$R_{best,i}$	3	5	7
K-means	Precision	0.98	0.84	0.93
	Recall	1	0.92	0.83
	F-measure	0.99	0.88	0.88
Ward’s method	Precision	0.98	0.84	0.93
	Recall	1	0.92	0.83
	F-measure	0.99	0.88	0.88

Table 6.04. The transformed features results.

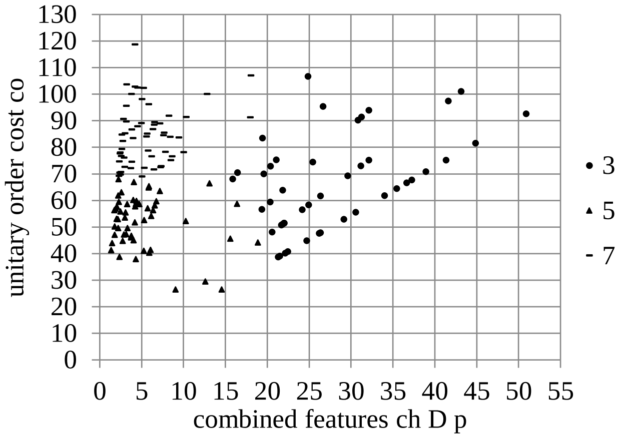

Figure 6.01. K-means clustering obtained using of the second set of features.

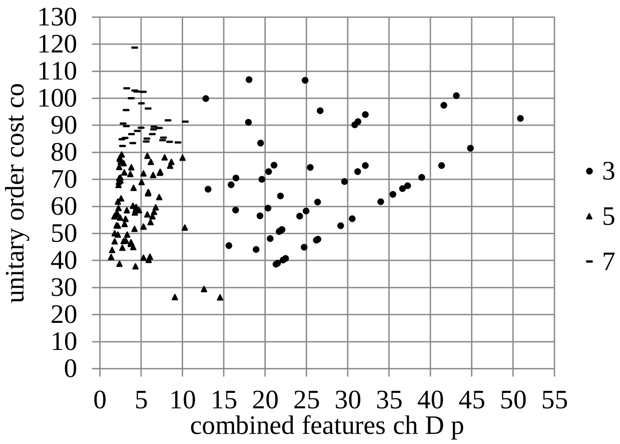

Figure 6.02. Ward's method clustering obtained using the second set of features.

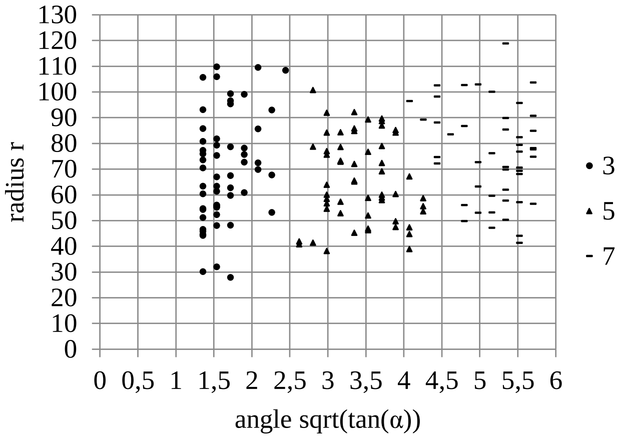

Figure 6.03. Clustering obtained using of the third set of features.

In order to capture the quality of the solutions, three widely-used performance measures [87] are computed for each experiment, clustering algorithm, and cluster:

- Precision, ratio between the number of items correctly classified in the cluster and the total number of items in the cluster.
- Recall, ratio between the number of items correctly classified in the cluster and the total number of items that are assumed to belong to the cluster.
- F-measure, harmonic mean of precision and recall.

Using the first set of features and the K-means algorithm the cluster $R_{best,i} = 3$ has high precision and small recall; the cluster is homogeneous but contains just a fraction of the items belonging to class $R_{best,i} = 3$. The remaining two clusters have low levels of both precision and recall. They contain a non-homogenous mix of items from all the three classes.

Ward's method performs better than the K-means using the first set of features, the recall for cluster $R_{best,i} = 3$ is significantly higher with 70% of the items of class $R_{best,i} = 3$ packed together. The cluster $R_{best,i} = 7$ increases its recall performance, contains 69% of the items of class $R_{best,i} = 7$, but is polluted by items from class $R_{best,i} = 5$ with a precision of 0.58. The cluster $R_{best,i} = 5$ is, as in the previous case, a non-homogenous mix of items from different classes.

Using the second set of features and the K-means algorithm, the cluster $R_{best,i} = 3$ has both high precision and recall. The cluster is homogeneous and contains 83% of the items of class $R_{best,i} = 3$. The remaining two clusters have a low level of both precision and recall. They contain a non-homogeneous mix of items from class $R_{best,i} = 5$ and class $R_{best,i} = 7$.

Ward's method performs better than the K-means using the second set of features. The recall for cluster $R_{best,i} = 3$ is higher, packing 94% of the items of class $R_{best,i} = 3$ the cluster near completely separates these items from the others. The cluster $R_{best,i} = 5$ increases its recall while remaining polluted by elements of class $R_{best,i} = 7$, while the recall of cluster $R_{best,i} = 7$ decreases sharply.

Overall, the use of second set of features seems slightly more effective than the first one. Nevertheless, as reported in Figure 6.01 and 6.02, both clustering algorithms define spherical linearly divisible clusters in the features space, while a correct classification should refer to the slope of the items. This is the intrinsic behaviour of these clustering algorithms.

The third set of features overcomes the linearity problem with a transformation that generates linearly separable classes. As a result, the precision and recall for all the clusters are high, the items are well separated into homogenous groups, as shown in Figure 6.03. In the last scenario the solutions for both the K-means and Ward's method are the same.

Conclusions

The use of the K-means and Ward's method is a viable substitute for inventory control system simulation when the clustering features are not directly obtained from the inventory control system. Theoretically grounded feature transformations are needed in order to obtain spherical linearly divisible clusters in the features space.

Future research could apply nonlinear and thus non-spherical clustering algorithms (e.g. spectral clustering) to overcome the need for theoretically grounded feature transformations, thus extending the methodology to inventory control systems that lack a strong analytical background.

5. Spare Parts Replacement Policy Based on Chaotic Models

This chapter is adapted from the paper "Spare Parts Replacement Policy Based on Chaotic Models" published in 2018 in IFAC-PapersOnLine, volume 51, issue 11, pages 945-950 [88].

Introduction

Manufacturing operations management includes the management of the materials needed for machine maintenance [89]. Common maintenance materials used by various maintenance crews, such as paints, grinding wheels, or welding, usually follow predictable consumption patterns. Crews can also share technical materials, such as bearings, circuit breakers, and lamps, whose consumption, although lower, is still predictable. Finally, spare parts are specific to certain machines and usually have erratic, unpredictable consumption patterns [37], [90]. Common maintenance and technical materials have regular consumption patterns modelled by exponential, normal, lognormal distribution or other well-fitted distributions [91]. Logistics techniques, such as demand forecasting, reorder point and the management of the replenishment time can increase the likelihood of success in their purchasing and replenishment operations [92], [93].

Maintenance crews interact with industrial machinery by breakdown interventions, when the production runs into unscheduled stoppages, solved as quickly as possible by replacing the damaged parts. Maintenance crews also interact by means of preventive procedures, scheduled by probabilistic methods, to be implemented before the next stoppage. Finally, by predictive procedures, maintenance crews monitor the evolution of the main failure modes and gather data to forecast the time up to the next stoppage. Spare parts policies play an important role in these three types of human-machine interaction [94], [95].

Regarding spare parts, due to the erratic behaviour characterised by the low consumption rate [37], the reorder point and management of the replenishment time may not suffice. The low consumption of parts leads to major difficulties in the management process [96]. In fact, sometimes, the consumption is zero, because the spare part may be stored for long periods, without being required, as the original part remains fully functioning. In advanced manufacturing, the obsolescence of spare parts, even before their employment, is not uncommon [97].

The application of logistic techniques such as Poisson processes (PP), appropriate for predictable, high consumption rates, may fail with erratically performing items. Due to small samples, confidence intervals are excessively large, weakening the power of the technique. Moreover, manufacturing has recently incorporated features from complex adaptive systems (CAS), with nonlinearities and mutual interactions among components leading to unpredictability. Logistic techniques relying on complex or chaotic models, rather than on predictable and high consumption patterns, are effective in such cases [98].

This chapter presents a modelling technique for the analysis of spare parts policies of items with a low consumption rate based on chaotic models. The research object is a set of four 1,500 HP AC motors that drive rolling mills in a steelmaking plant, the modelled variable is the time between failures that require changing one of the four motors. The research technique relies on two chaotic models:

- A logistic map with a single dimension.
- A Hénon map with two dimensions.

The specific objectives are to:

- Analyse the spare parts situation of the set of motors by non-homogeneous PP.

- Repeat the analysis supported by chaotic models.
- Compare the results.

Poisson point processes and spare parts replacement policies

Spare parts consumption behaves usually like counting processes [99]. A counting process is a type of Poisson point process, or simply a PP. A PP consists of points randomly located in a numerical space, and can model random events in stochastic processes. PP has an intensity parameter λ related to the expected number of points in a given bounded region of the numerical space [100]. Four types of counting processes are useful for analysing spare parts policies:

- Homogeneous Poisson processes (HPP).
- Renewal processes.
- Non-homogeneous Poisson processes (NHPP).
- Imperfect repair processes.

Renewal and imperfect processes involve, respectively, a complete overhaul ("as good as new" repair) and partial modifications in machines along with the spare part replacement. The focus is on HPP and NHPP, which more realistically describe the "as bad as old" repair policy prevalent in manufacturing activities. In manufacturing, a system consisting of a large number of sub-systems fails when a single part fails and the crew replaces it after a negligible downtime [101]. Therefore, it is reasonable to assume that the overall reliability remains the same after the repair, thus characterising the minimal repair policy [102].

The numerical space of a PP is a unidimensional timeline, where individual numbers correspond to time intervals between zero and infinity. HPP has a constant failure rate function λ and NHPP has a power-law failure rate function $\lambda(t)$. In spare part consumption processes, an HPP acts as a counting process and models times to failure according to a constant failure rate λ. From the properties of HPP, times to failure are independent, and the number of failures in any interval T is a Poisson random variable with mean λT [101]. The main implication is that if the consumption of a spare part follows a certain constant failure rate λ, the probability P of x events in the time interval T is given by:

$$P(x) = \frac{(\lambda T)^x e^{-\lambda T}}{x!} \quad (7.01)$$

If the failure rate of a certain part is not constant, which means that the part has improved or deteriorated, then a NHPP should model the failure sequence and consequently the spare part consumption. Such an NHPP is a generalisation of an HPP, relaxing the stationary pattern of failures and assuming a power-law pattern, which can be positive or negative. If the power-law has a unitary exponent, the NHPP turns into an HPP [101]. The main implication of an NHPP is that if the consumption of a spare part follows a power-law $\lambda(t)$ with shape factor γ and scale factor θ, the probability of x events in the time interval T is as follows:

$$P(x) = \frac{\left(\frac{T}{\theta}\right)^{\gamma x} e^{-\left(\frac{T}{\theta}\right)^{\gamma}}}{x!} \quad (7.02)$$

Fitting the failure data to a Weibull distribution by maximum likelihood estimation (MLE) provides the shape and scale factors. On occasions, there may also be a shift parameter t_0, the failure free time. The cumulative intensity function provides the expected number of failures by time T. The mean time between failures (MTBF) and the reliable lifetime T_R with a confidence level of 0.95 are:

$$\Lambda(T) = \int_0^T \lambda(t)dt \quad (7.03)$$

$$MTBF = t_0 + \theta \cdot \Gamma\left(\frac{1}{\gamma} + 1\right) \quad (7.04)$$

$$T_{R,0.95} = t_0 + \theta \cdot \left(-ln(0.95)\right)^{\frac{1}{\gamma}} \quad (7.05)$$

If a maintenance crew knows about the failure rate function and can estimate the time to the next failure, they can provide a spare part replacement policy [103]. The simplest policy is to predict the probabilities of the next failure over time or calculate the reliable life and then define a certain number of spare parts. However, a problem arises in the case of a very low failure rate. As the MTBF is very large, the sample size is low, thus producing large confidence intervals for the parameters. Moreover, even if a distribution fits a small sample size, it is not possible to ensure that the assumptions that characterise HPP and NHPP remain over time [98]. For example, it is not possible to ensure that there has been no change in the requirement of the manufacturing process or the required workload. To help solve this problem, it is proposed a method to manage spare parts with a low consumption rate based on chaotic models.

Chaotic model

Chaotic models describe phenomena with deterministic formation laws, but which, at first glance, seem to be random. Such behaviour originates from the interactions among internal parts of complex and dynamic systems, with a fundamental instability, i.e. the sensitivity to the initial conditions. Although they originate from deterministic rules, the recurrence of the application of the rule, under certain circumstances, makes chaotic phenomena unpredictable in the long term. The extreme dependence on the initial conditions of the parameters determines that the output of a chaotic phenomenon will become unstable over time [104]. The consequence of this instability is that the results of deterministic systems, even with definite evolution laws, are extremely sensitive to disturbances and noise, thus making them unpredictable. Even in the absence of noise, nonlinearities and interactions among components amplify minimal errors in parameters, generating deterministic chaos [105].

Positive feedback generates chaotic situations, which lead to points of inflection and rupture, i.e. bifurcation points. The simplest chaotic model is the logistic map, a positive feedback process involving a quadratic function of itself. The logistic map is a recurrence relation that exemplifies how complex and chaotic behaviours arise from the application of a simple and deterministic rule [106]. The transition from order to chaos justifies the use of the expression “deterministic chaos” for such models [107]. The logistic map model is given by:

$$x_{t+1} = a \cdot x_t \cdot (1 - x_t) \quad (7.06)$$

$4 > a > 0$ and:

- If $a < 3$, the process converges to a fixed value.
- If $a < 1$, the process converges to zero.
- If $1 < a < 3$, the process converges to $s = \frac{a-1}{a}$, a fixed attractor.
- If $3 < a < 3.57$, the process becomes cyclic, with bifurcations that lead to multiple values, a cyclic attractor.
- If $a > 3.57$, the outcome is nearly unpredictable with oscillating behaviour, a strange attractor.
- If $a > 4$, the outcome is chaotic [106], [108].

The $a > 3.57$ region is the chaos edge where, at first glance, the time series of the logistic map does not differ from a random time series while, by plotting $x_{(t+1)}$ and x_t, a regular pattern arises. This reinforces the notion of chaos out of the order or deterministic chaos.

Figure 7.01 to 7.06 illustrate fifty executions of the logistic map under different parameters, as well as the difference between the subjacent relationships in random execution and chaotic execution. It shows executions of the logistic map with $s(0) = 0.5$ and:

- $a = 1$ in Figure 7.01.
- $a = 3$ in Figure 7.02.
- $a = 3.95$ in Figure 7.03.
- A series generated by a random generator in Figure 7.04.
- A dispersion graph of a logistic map execution without a noticeable pattern in Figure 7.05.
- A dispersion graph of a logistic map execution with an almost linear pattern in Figure 7.06.

The two last figures illustrate the difference of a random process, without a subjacent formation law, and a chaotic process, with a deterministic formation law.

The other chaotic model of interest is the Hénon map. This map takes a point (x, y) and maps it to a new point (x', y'):

$$x_{t+1} = y_{t-1} + 1 - a \cdot x_t^2 \tag{7.07}$$

$$y_{t+1} = b \cdot x_t \tag{7.08}$$

For $a = 1.4$ and $b = 0.3$, the map approaches the Hénon attractor. For other values, the map may be also chaotic, but it may also be intermittent, or converge to a periodic orbit [108]. Figure 7.07 illustrates one thousand executions of the Hénon map, forming the Hénon attractor, with $x_0 = 1.2$ and $y_0 = 1.4$.

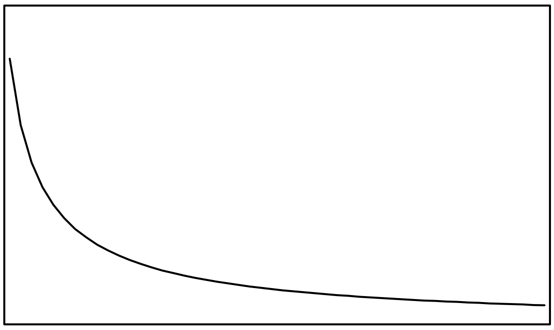

Figure 7.01. Logistic map with $s(0) = 0.5$ and $a = 1$.

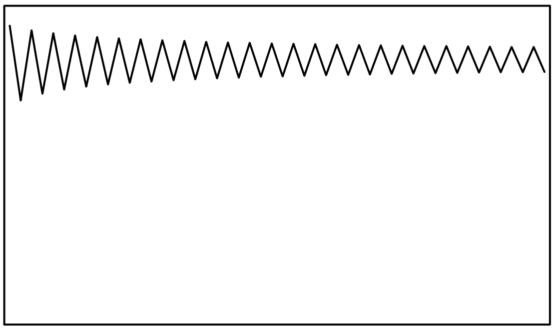

Figure 7.02. Logistic map with $s(0) = 0.5$ and $a = 3$.

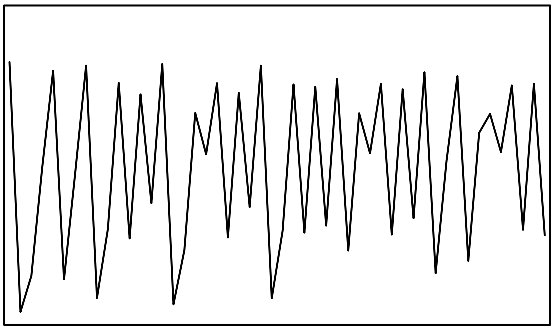

Figure 7.03. Logistic map with $s(0) = 0.5$ and $a = 3.95$.

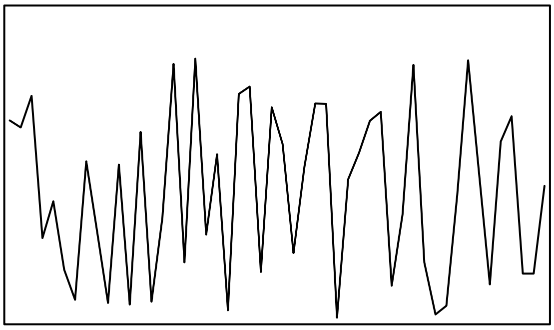

Figure 7.04. Random generator.

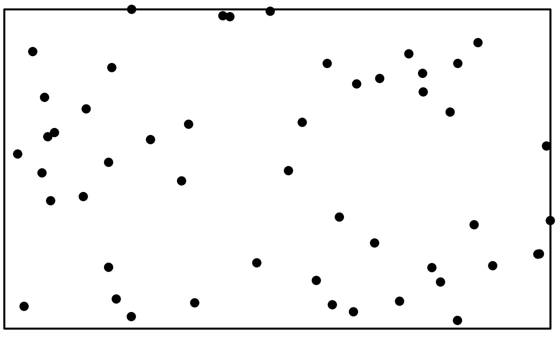

Figure 7.05. Logistic map execution without a noticeable pattern.

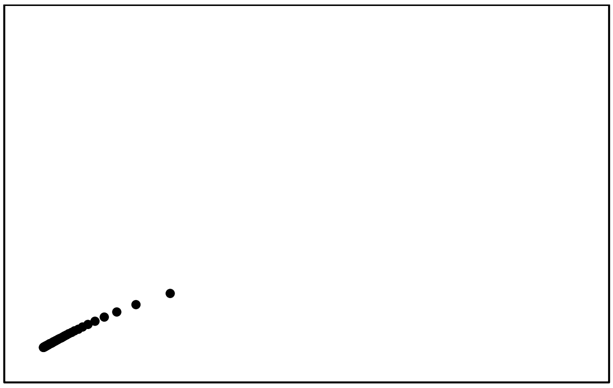

Figure 7.06. Logistic map execution with an almost linear pattern.

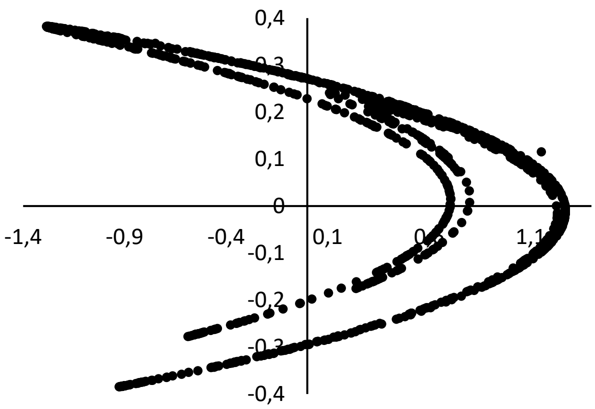

Figure 7.07. Hénon attractor.

The subject and the research

This chapter focus is the rolling mill plant of a semi-integrated steelmaking plant. There are two technological routes for steel production:

- Integrated plants, with iron ore.
- Semi-integrated plants, with metallic scrap and pig iron.

Semi-integrated plants operate the steel refining stage in melt shops, and the conformation stage in rolling mill shops. Usually, the main drivers of rolling mills are medium voltage alternate current squirrel-cage induction motors. This chapter studies a set of five 1,500 HP motors that drive four hot rolling mills plus a spare motor. Each time a motor fails, the spare motor replaces it. Once repaired, the damaged motor returns to the warehouse until a new failure occurs, and so on. As a typical capital good, the motors have a long lifespan, overcoming several decades [109].

Large equipment failures follow a degradation path that might take up to several months or even years until the occurrence of a breakdown event [110]. Two components represent the time to failure (TTF):

- The time elapsing between the last repair and the beginning of a potential failure, the initial event of the degradation process that will cause the failure.
- The time between the beginning of the potential failure and the functional failure, the breakdown event that requires maintenance actions [111].

According to [112]–[114] the TTF follows an exponential model for random failures and a normal distribution can approach TTF for wear-out failures. As the focus is in the wear-out phase, the normal distribution is used in the calculation of the reliable life.

The research has limitations:

- Other chaotic models could be used, such as the Lorenz attractor.
- The time to repair (TTR) considered is deterministic, which requires the use of simulation techniques.
- The experiment assumed normality for the determination of the variability in forecasting.

Although this last assumption is supported in the literature, future research could include methodologies to bridge the last two gaps.

Results and discussion

The TTR are eight and twelve months if the damage requires major repairs and the construction of new subsystems, respectively. Table 7.01 shows the cumulative time of the process, the time between failures (TBF) of the motors in months and the normalised TBF, a more useful form for the analysis.

Failure	Cumulative time	TBF	Normalised TBF
1	0	11	0.275
2	36	25	0.625
3	76	40	1.000
4	99	23	0.575
5	119	20	0.500
6	152	33	0.825
7	178	26	0.650
8	210	32	0.800
9	228	18	0.450

Table 7.01. TBF (in months).

An MLE fitted a Weibull distribution with $t_0 = 3.01$, $\gamma = 2.51$, and $\theta = 24.51$, with a significance level of 0.23 (χ^2 test) and 0.14 (Kolmogorov-Smirnoff test), while $MTBF = 24.76$ and $T_{R,0.95} = 10.52$ months. The 0.95 confidence intervals for γ and θ are respectively $(1.20, 3.69)$ and $(17.50, 34.14)$. Applying the previous equations to the MLE values of γ and θ, and considering eight and twelve months for TTR, the model leads to the results reported in Table 7.02.

x	Probability of x failures in 12 months	Cumulative probability	Probability of x failures in 8 months	Cumulative probability
0	0.923	0.923	0.95599	0.956
1	0.074	0.997	0.04302	0.999
2	0.003	1.000	0.00097	1.000

Table 7.02. NHPP of the replacement of motors.

As $T_{R,0.95} = 10.52$ months, maintaining no spare parts other than the five motors is a safe policy for major repair, but is not for subsystem manufacturing. Maintaining one spare part, the probabilities of a shortage are $(1, 0.997)$ and $(1, 0.999)$ respectively, which is a safe policy. In any case, it is necessary to consider that $T_{R,0.95} = 10.52$ months results from the adoption of MLE parameters. Considering uncertainty, the extreme situations are:

- $\gamma = 3.69$, $\theta = 34.14$ and $T_{R,095} = 18.27$ months.
- $\gamma = 1.20$, $\theta = 17.50$, $T_{R,0.95} = 4.48$ months.

The observations lasted for more than twenty years, in which the manufacturing evolved, approaching CAS. Among other transformations, there was a learning period and the technological development of maintenance techniques, with the introduction of remote sensing, predictive analysis, and corrective maintenance procedures. In the first instance, such evolutions reduced the downtime, which encouraged the company to increase production, which increased downtime further, in a feedback loop. Another example is the use of multiple fuels in the reheating furnace, which changed the billet temperature and reduced the load on the drives. The company then increased production, increasing the load on the same drives. Such feedback loops, characteristic of CAS, may violate a premise of PP, i.e. the independence between failure events, opening the possibility of using complex, alternative methods.

It is proposed an alternative method to analyse the spare part policy. Table 7.03 shows the time evolution, with a step of five positions, of the logistic map with $s(0) = 0.896316$ and $a = 3.698554$. A commercial solver finds the values minimising the minimum square error (MSE) of the regression formed by the map and the normalised TBF.

Failure	s	TBF norm	SE
1	0.344	0.275	0.005
6	0.710	0.625	0.007
11	0.912	1.000	0.008
16	0.505	0.575	0.005
21	0.666	0.500	0.027
26	0.736	0.825	0.008
31	0.626	0.650	0.001
36	0.797	0.800	0.000
41	0.450	0.450	0.000
Forecast	0.802	MSE	0.087

Table 7.03. Time evolution for the logistic map.

The model provides the most likely normalised time to the next failure, that is $s(48) = 0.802$. As $\sigma = 0.21$ and assuming a normal distribution (one-tailed test), the new $T_{R,0.95} = 40 \cdot (0.802 - 1.64 \cdot 0.219) = 17.71$ months.

Table 7.04 shows the time evolution, with a step of three positions, of the Hénon map with $x(0) = 0.027266$ and $y(0) = 0.281145$ for outcome x and $x(0) = -0.33122$ and $y(0) = -1.48856$ for outcome y. The solver finds these values minimising the MSE of the regression formed by the map evolution and the normalised TBF.

Failure	x norm	y norm	TBF norm	SE $x(t)$	SE $y(t)$
2	0.000	0.000	0.28	0.076	0.076
5	0.624	0.596	0.63	0.000	0.001
8	1.000	1.000	1.00	0.000	0.000
11	0.601	0.497	0.58	0.001	0.006

14	0.674	0.526	0.50	0.030	0.001
17	0.796	0.905	0.83	0.001	0.006
20	0.788	0.502	0.65	0.019	0.022
23	0.846	0.897	0.80	0.002	0.009
26	0.376	0.443	0.45	0.005	0.000
Forecast	0.682	0.610	MSE	0.129	0.123

Table 7.04. Time evolution for the Hénon map.

The two models provide the normalised time to the next failure, $x(48) = 0.682$ and $y(26) = 0.610$, which are then used to calculate $T_{R,0.95} = 40 \cdot (0.682 - 1.64 \cdot 0.219) = 12.71$ months and $T_{R,0.95} = 40 \cdot (0.610 - 1.64 \cdot 0.219) = 10.03$ months.

Table 7.05 summarises the results of the study and evaluates the two spare parts policies.

Process and policy	NHPP	Logistic Map	Hénon x	Hénon y
$T_{R,095}$ (months)	10.52	17.71	12.71	10.03
No spare part				
Major repairs (10 months)	Safe	Safe	Safe	Safe
New part (12 months)	Unsafe	Safe	Safe	Unsafe
One spare part				
Major repairs (10 months)	Safe	Safe	Safe	Safe
New part (12 months)	Safe	Safe	Safe	Safe

Table 7.05. Analysis of spare parts policies.

The first policy is to have no spare part, meaning that the company rotates five motors around four positions. The second policy is with one spare part, meaning that the company maintains an additional motor in the warehouse, thus being able to rely on six motors to provide four drives. The table shows that only the second policy is entirely safe. This conclusion does not necessarily imply the purchase of new equipment, for very short periods the company can manufacture with only three drives. In this last case one of the two rolling mills must operate with only one drive, producing only heavy, rough sections.

Finally, regarding accuracy, Figure 7.08, 7.09 and 7.10 show the relationship between the models (logistic map, Hénon x, and Hénon y respectively) and the set of life data. All R^2 are near to 1, which means that the forecasting provided by the models can be reasonably considered for decisions on spare parts.

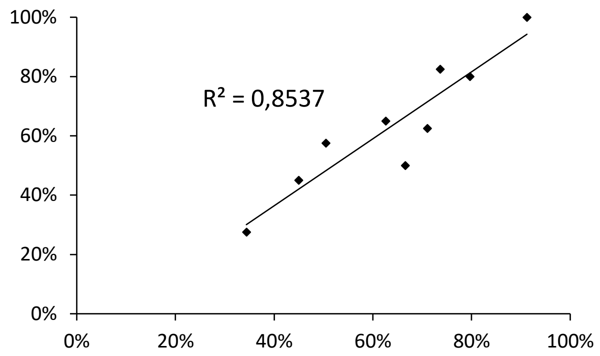

Figure 7.08. Relationship between life data and the logistic model.

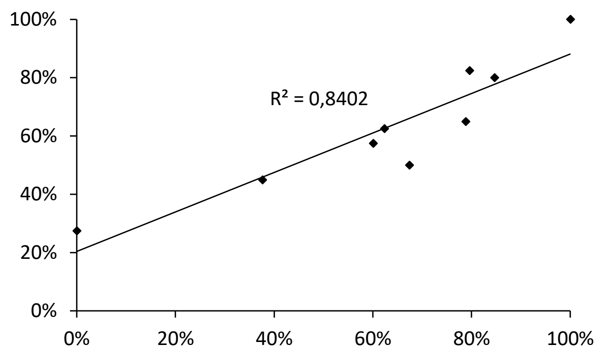

Figure 7.09. Relationship between life data and the three Hénon x model.

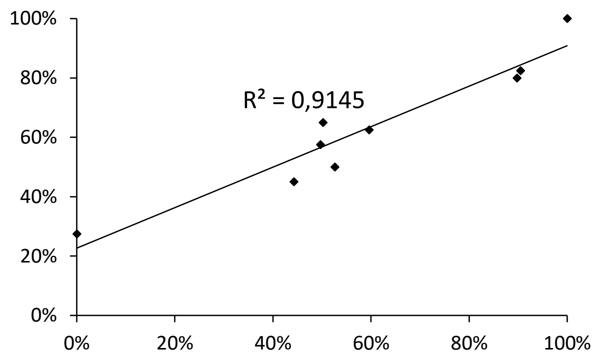

Figure 7.10. Relationship between life data and the Hénon y model.

Conclusions

It was presented a modelling technique for the analysis of spare parts policies of items with a low consumption rate, specifically a set of five medium voltage motors that drive four rolling mills in a steelmaking plant. It was found that, according to all the tested models and to ensure full safety, the company needs an additional spare part. Alternatively, the company could reduce production for short periods with only three drives.

The study focused on a system with few large parts that fail with low frequency. Larger and complex systems with imbricated mutual relationships usually provide much more life data, thus solutions obtained from stochastic methods are usually satisfying. Nonetheless, even a spare part replacement policy for larger, imbricated systems could benefit with chaotic methods, when the life data amount is not enough to produce suitable confidence intervals.

Further research shall bridge the gaps of this chapter, the use of more chaotic model, other models than the normal for the forecasting, and considering the stochastic nature of the TTR variable.

6. Inventory control with perishability - 1

This chapter is adapted from the paper "Inventory control system for intermittent items with perishability" published in 2017 in 24th International Conference on Production Research (ICPR), pages 780-785 [115].

Introduction

In this chapter, the inventory system proposed in [3] (TSB) is adopted, but some relevant modifications are introduced enabling it to deal with perishable items. Inventory systems for perishable goods (e.g., food and pharmaceutical products) have been the focus of much attention in the academic literature. The assumption that an item can be stored indefinitely in warehouses does not hold for perishable goods, and this complicates their inventory control. In the literature [116]–[118] the first element of interest for a perishable item is its lifetime, which may be:

- Fixed.
- Distributed according to a certain probability distribution.
- Characterised by a time-inventory dependent deterioration rate [119]–[122].

Deterioration can occur in various ways, but it must be distinguished from obsolescence, which refers to the loss of value due to technological changes or the entry of new products into the market. In this chapter the lifetime of goods is fixed and known a priori while the demand is stochastic. When perishable goods also exhibit intermittent consumptions, their inventory control results in a further complication due to the ineffectiveness of traditional inventory systems in this context. Intermittent demand with obsolescence has been addressed in [16] with the aim of increasing the demand estimators accuracy. Conversely, the perishability of intermittent demand has never been addressed even if this kind of demand generation process is relevant in real settings. A recent approach focusing on non-stationary demand subjected to perishability is reported in [123], where the well-known Silver and Meal heuristic for time-varying demand was adapted to the case of perishable items. However, the intermittency was not considered specifically.

Literature review

Recent contributions referred to the stochastic demand of perishable goods with fixed lifetimes. [124] dynamically determined replenishment quantities for perishable goods with fixed lifetimes that satisfy multiple service level constraints during a specific period, and extended its model to non-stationary demand. [125] addressed a joint pricing and inventory control problem for stochastic perishable inventory systems in which both backlogging and lost-sales cases were studied. It provided an approach able to deal with both continuous and discrete demand distributions. Similarly, [126] dealt with the dynamic pricing and production rate for stochastic and price-dependent demand of items with fixed lifetime in a continuous-time environment. [127] addressed the production planning of perishable products with fixed lifetimes and non-stationary demand; it described an MILP model containing a service level constraint. [128] proposed another MILP model for a fill rate constraint. [129] addressed the normally distributed demand of perishable items with fixed lifetimes to reach the optimum lot size. It evaluated the probability of a product remaining in stock beyond the end of its lifetime, and determined the best order size, the time at which the inventory level drops to zero, and the cycle time minimising the expected total cost. [123] achieved the expected inventory level at different ages for the non-stationary stochastic demand of perishable items with fixed lifetimes. It also extended Silver's heuristic [130] to deal with these conditions by means of analytical and simulation-based variants of the original heuristic. [131] adopted a periodic review system for the stochastic demand of items with fixed lifetimes, adding the closing days constraint as a typical feature of groceries. [121] showed the value of dual-sourcing in the context of perishable items with fixed lifetimes and a Poisson-distributed demand. It considered an age-based control with a base stock policy. Perishable items with stochastic demand and fixed lifetimes were also studied in [132], it proposed an aged-based replenishment policy solved by a reinforcement learning algorithm.

Methodology

The Teunter-Syntetos-Babai model

In TSB the underlying intermittent demand is assumed to follow the structure described in [1]. From this pattern a periodic order-up-to (R, S) policy is defined, assuming the positive demand random variable follows a known distribution whose parameters are unknown.

An order can be placed every R periods, collectively defining the constant review time, and requires a fixed lead-time L to arrive. In this scenario, at the beginning of a review time, a stock $S \geq 0$ is available and an order O can be placed. The total amount $S + O$ is expected to cover the demand of R periods after the lead time, a new order can in fact be placed only after $R - L$ periods and requires L periods to arrive. The performance measure associated to this model is the fill rate over the specified t periods, here defined as the probability a positive demand taking place in one of those periods is satisfied by $O + S$ and thus generates no stock out.

Given a stock s and order quantity O, the fill rate is:

$$FR = \frac{1}{R}\sum_{t=1}^{R}\sum_{k=0}^{t+L-1}\binom{t+L-1}{k}p^k(1-p)^{t+L-1-k}\cdot\Phi(O+S,k+1) \quad (8.01)$$

$\Phi(x, y)$ is the cumulative probability y positive demand periods yield a total demand inferior to x.

The main component of the previous equation is the probability a positive demand taking place in t is satisfied with $O + S$:

$$FR_t = \sum_{k=0}^{t+L-1}\binom{t+L-1}{k}p^k(1-p)^{t+L-1-k}\cdot\Phi(O+S,k+1) \quad (8.02)$$

Inventory replenishment model

The proposed replenishment model divides the stock S into two separate stocks:

- S_e the amount of goods that will expire at the end of one of the R periods after L.
- S_{ne} the amount of goods that will not expire in said time frame.

These quantities are updated as in TBS at the beginning of each lead-time, before the order O is placed. The expired stocks are discarded, and the expiring stock is moved from S_{ne} to S_e. The stock S_e is assumed to expire at the end of period t_e, calculated from the update period before the lead time, while the ordered quantity O is assumed not to expire in the time frame.

Given a hypothetical positive demand d_t at period t, two mutually exclusive cases can arise:

- The period t occurs before the expiration date.
- The period t occurs after the expiration date.

In the first case the perishability has no effect, thus the TSB equation is used. In the second case S_e has expired and a different equation is required. Given a positive demand d_i in period t after L, all the possible demands in the previous periods $\sum_{k=0}^{t+L-1} d_k$ must be considered. This leads to two scenarios:

- The demands before the expiration date partially or totally consumed the expiring stock, i.e. $\sum_{k=0}^{t_e+L-1} d_k \leq S_e$.

- The demands before the expiration date consumed more than the expiring stock, i.e. $S_e < \sum_{k=0}^{t_e+L-1} d_k \leq S_e + S_{ne} + O$.

The fill rate FR_{t1} of the first scenario is the probability that the demands before t_e are satisfied by S_e and the demands after t_e are satisfied by $S_{ne} + O$:

$$FR_{t1} = \sum_{k=0}^{t_e} \sum_{h=0}^{t+L-t_e-1} \binom{t_e}{k} p(k, t_e) \Phi(S_e, k) \cdot \binom{t+L-t_e-1}{h} p(h, t+L-t_e-1) \Phi(O + S_{ne}, h+1) \quad (8.03)$$

$p(x, y) = p^x (1-p)^{y-x}$ is the probability that x periods over y present a positive demand. $\Phi(x, 0) = 1$ $\forall x \geq 0$ since in absence of positive demands no stock out can occur.

The fill rate FR_{t2} of the second scenario is the probability the demands d_{be} before t_e are satisfied by $O + S$ and the demands after t_e including d_i are satisfied by the remaining stock $O + S - d_{be}$ with $d_{be} > s_e$:

$$FR_{t2} = \sum_{k=1}^{t_e} \sum_{h=0}^{t+L-t_e-1} \sum_{d_{be}=S_e+1}^{O+S} \binom{t_e}{k} p(k, t_e) \phi(d_{be}, k) \cdot \binom{t+L-t_e-1}{h} p(h, t+L-t_e-1) \Phi(O + S - d_{be}, h+1) \quad (8.04)$$

These scenarios are mutually exclusive, thus the fill rate FR_t of period t is:

$$FR_t = FR_{t1} + Fr_{t2} \quad (8.05)$$

The overhaul fill rate FR accounts for the individual fill rate of R periods after the lead time:

$$FR = \frac{1}{R} \sum_{t=1}^{R} FR_t \quad (8.06)$$

This methodology expands the one previously defined considering a portion S_e of the stock as perishable. The calculations above refer to a single expiration date but similar considerations can be applied to address multiple expiration dates in the frame of analysis. From a computational perspective the proposed methodology is more demanding than the original one, and the calculation of FR_{t2} requires an analysis of $O + S_{ne}$ demands before t_e. This calculation is necessary since the last component of FR_{t2}, the probability a demand after t_e does not produce a stock out, requires the number of units $O + S - d_{be}$ left in stock.

Modified inventory replenishment model

When $O + S_{ne}$ is high the proposed methodology is expensive from a computational standpoint. In order to reduce the computational effort a different equation for FR_{t2} is proposed:

$$FR_{t2} = \sum_{k=1}^{t_e} \sum_{h=0}^{t+L-t_e-1} \binom{t_e}{k} p(k, t_e) \binom{t+L-t_e-1}{h} \cdot p(h, t+L-t_e-1) \cdot \left(\Phi(O+S, k+h+1) - \sum_{d_{be}=0}^{S_e} \phi(d_{be}, k) \cdot \Phi(O + S - d_{be}, h+1) \right) \quad (8.07)$$

It requires the analysis of $S_e + 1$ values of d_{be} instead of $O + S_{ne}$, thus the following solving procedure can be applied:

- If $O + S_{ne} \leq S_e + 1$ compute Equation 8.04.
- If $O + S_{ne} > S_e + 1$ compute the Equation 8.07.

This procedure aims at mitigating the computational costs of FR_{t2} calculation but it does not reduce the effort when $O + S_{ne} \cong S_e$ or when both $O + S_{ne}$ and S_e are high.

Inventory replenishment model solution

The proposed model aims at defining the order quantity O_{min} at the beginning of lead time L. O_{min} is the minimum order capable of achieving a target fill rate FR_{target} for R periods after the lead time L. In contrast, the previous equations calculate the fill rate FR of R periods after the lead time L given a predefined order quantity O. Those equations are not easy to invert, and no direct method is available to solve the problem at hand. A common solution in the relevant literature involves a stepwise search:

- Start assuming $O = 0$.
- Calculate FR for the value of O under analysis.
- If $FR \geq FR_{target}$ then stop, $O_{min} = O$.
- If $FR < FR_{target}$ then increase O by one unit and go back to the second step.

This procedure is feasible if the computational cost for the fill rate calculation is limited. In the case at hand such cost is significant and increases with O, thus the algorithm reactivity decreases as it goes on. An alternative procedure, based on the secant method, is proposed to decrease the amount of calculations involved. The optimum is formally defined as:

$$O_{min} = min\{O: FR(O, S_e, S_{ne}) \geq FR_{target}\} \tag{8.08}$$

$FR(O, S_e, S_{ne})$ is the fill rate relative to the order quantity O and the stocks S_e and S_{ne}.

Since, fixed the stocks S_e and S_{ne}, the fill rate can grow only if O increases, it can be rewritten as:

$$O_{min}: FR(O_{min}, S_e, S_{ne}) = min\{FR(O, S_e, S_{ne}): FR(O, S_e, S_{ne}) \geq FR_{target}\} \tag{8.09}$$

Two properties of the fill rate provide two extremes O_{sup} and O_{inf} to initialize the secant method. This initialization requires no initial calculation of the overhaul FR:

- Ceteris paribus a decrease in S_e reduces FR.
- Ceteris paribus substituting part of S_e with stock not expiring in t increases FR.

From these properties two quantities can be defined:

$$O_{sup}: FR\left(O_{sup}, 0, S_{ne}\right) = min\{FR(O, 0, S_{ne}): FR(O, 0, S_{ne}) \geq FR_{target}\} \tag{8.10}$$

$$O_{inf}: FR\left(O_{inf}, 0, S\right) = min\{FR(O, 0, S): FR(O, 0, S) \geq FR_{target}\} \tag{8.11}$$

With the property:

$$O_{inf} \leq O_{min} \leq O_{sup} \tag{8.12}$$

In Equation 8.10, starting from the optimum order quantity, the elimination of S_e reduces FR. From this point, in order to achieve $FR(O, 0, S_{ne}) \geq FR_{target}$ fixed S_{ne}, the order quantity now defined O_{sup} increases. A similar effect takes place in Equation 8.11 where the expiring stock is fully substituted by stock non expiring. The substitution increases the fill rate and, for this new configuration, the initial order quantity is no longer the minimum required to achieve $FR(O, 0, S_{ne}) \geq FR_{target}$. The order quantity now defined O_{inf} decreases to reach the minimum fill rate required. Equations 8.10 and 8.11 contain no expiring stock, and thus the computationally expensive calculations are not required. Equation 8.01 is iteratively applied to define both O_{sup} and O_{inf}.

In order to apply the bisection method, the fill rate expressed in Equation 8.06 is shifted by FR_{target}:

$$FR_{shifted} = FR - FR_{target} \tag{8.13}$$

The fill rate function is strictly increasing fixed S_e and S_{ne}. If the previous equation has roots in the interval $\left[O_{inf}, O_{sup}\right]$ it has a single root, if it has no roots in the interval then $FR\left(O_{inf}, S_e, S_{ne}\right) > FR_{target}$. In this last scenario $o_{min} = o_{inf}$ and the algorithm terminates during the calculation of $FR\left(O_{inf}, S_e, S_{ne}\right)$ in the first step as described below.

Given O_{sup} and O_{inf} the calculation of their fill rate using Equation 8.06 is required at the beginning of the bisection algorithm. This defines the extreme values of $FR_{shifted}$ and makes possible the initial secant calculation:

$$FR_{sup} = FR(O_{sup}, S_e, S_{ne}) \tag{8.14}$$

$$FR_{inf} = FR(O_{inf}, S_e, S_{ne}) \tag{8.15}$$

During the generation of new FR_{sup} and FR_{inf}, and the respective O_{inf} and O_{sup}, the algorithm operates only on integer values of O. The new quantity O identified by the secant must be approximated by the nearest integer. If it falls over the current O_{sup} or under the current O_{inf}, the value is respectively approximated by $\lfloor O \rfloor$ and $\lceil O \rceil$. The algorithm terminates when $FR(O, S_e, S_{ne}) = FR_{target}$, when a floor approximation reaches o_{min} or a ceil approximation reaches O_{sup}. In the last two scenarios the interval of analysis has unitary length and by construction $O_{inf} < O_{min}$, thus $O_{min} = O_{sup}$.

Experimental analysis

Probability distribution and estimations

In order to compute Equation 8.06, both the distribution function $\phi(x, y)$ and the cumulative distribution function $\Phi(x, y)$ of a positive demand x during y periods must be known. It is also required the knowledge of the probability p a positive demand occurs during a period. These three components of Equation 8.06 vary across time and must be indirectly forecasted from the item time series. As suggested in [3], the positive demand distribution is hard to determine over an arbitrary number of periods, unless the multiple periods distribution can be defined from the single period distribution. This experimental analysis assumes that the positive demand during a single period is negative binomial distributed. The sum of independent negative binomial distributions is a negative binomial distribution itself, with different parameters depending on the number of random variables added. In the case at hand, the number of random variables is the number of periods. The use of a discrete random variable, instead of a continuous one as in [3], is coherent with Equation 8.04 and Equation 8.07 where d_{be} moves through integer values.

In order to estimate the single period parameters of the negative binomial distribution, a time series analysis is required. The methodology used for this experimental analysis is the same applied in [3], SBA is applied to define z_i and $\widehat{Var}(z_i)$ for the positive demand. From these values the negative binomial distribution parameters are derived, while the forecasting technique estimates the probability p directly.

Experiment settings

The experimental analysis consists in a set of simulations, carried over with different parameters, where the proposed methodology is tested on both generated and historical series. Each simulation outputs a set of performance measures that are compared to each other and evaluated against a benchmark. The model effectiveness is assessed in three different scenarios:

- Ideal scenario.
- Approximate model scenario.
- Real scenario.

The first scenario refers to an ideal case in which time invariant intermittent distributions are generated. As previously defined their positive demand magnitude follows a negative binomial distribution with known parameters:

- p_b success probability.
- r required number of successes before the first failure.

The probability p a positive demand occurs in a period is also known and time invariant. In this first scenario the estimations are not used since the simulation is fed from the beginning with the correct parameters.

The second scenario measures the effectiveness of the proposed methodology in a context where components of stock S_e expire in two different dates during the $R + L$ periods following the update period. Equation 8.06 is not modified to admit a multiple number of expiration dates since this experiment measures the method effectiveness while it is implemented as an approximation. In the case at hand Equation 8.06 is implemented considering the first expiration date only and attributing to said period all the expiring stock S_e. Like the previous one this scenario does not utilize the estimation methodology proposed previously.

The third scenario applies the proposed methodology to historical demand series. In this case the expiration date is single, as in the first scenario, while the positive demand parameters p_b and r are unknown as the positive demand occurrence probability p.

For each scenario a set of fixed parameters is defined while a second set of parameters varies among the simulations. All the possible combinations of the second set of parameters are tested in order to assess the method performance in different contexts. Table 8.01 and 8.02 summarize both the fixed and the variable parameters for the first, second and third scenarios.

Parameter	Description	Category	Values
n_c	Number of cycles	Fixed	100
R	Periods in a cycle	Fixed	10
n_e	Periods before expiration	Fixed	12 first sc., 19 sec. sc.
L	Lead time	Variable	3, 5
FR_{target}	Fill rate target	Variable	0.8, 0.9
p	Demand probability	Variable	0.1, 0.3
p_b	Success probability	Variable	0.3, 0.5, 0.7
r	Required number of successes	Variable	1, 3, 5, 7

Table 8.01. Parameters in the first and second scenarios.

Parameter	Description	Category	Values
n	Number of periods	Fixed	120
n_w	Warm up periods	Fixed	30
n_s	Simulation periods	Fixed	90
R	Periods in a cycle	Fixed	10
n_e	Periods before expiration	Fixed	12
L	Lead time	Fixed	3
FR_{target}	Fill rate target	Variable	0.8, 0.9

Table 8.02. Parameters in the third scenario.

The only fixed simulation parameter that changes between the first and the second scenario is the number of periods before expiration n_e. When a product is ordered it arrives after L periods and then lasts for n_e periods before expiring, if the value of n_e follows the relation:

$$\exists x \in N: \ \ x \cdot R < n_e \leq x \cdot (2R - L) \tag{8.16}$$

The generated system presents only a single expiration date during the $R + L$ periods of analysis. Otherwise n_e can generate two different expiration dates to be managed in each analysis.

The results are collected for each simulation period after the first lead-time, when the first order has already arrived. In this context no initial level of backorders and stock is required to make the first measurements fair. In the third scenario 30 initial warm up periods are also required to initialise the intermittent demand forecast. These periods are for initialisation purposes only and the actual simulation starts from period 31.

Performance metrics

For each experiment a performance measurement takes place in each simulation period. If the period presents a positive demand, then the total number of positive demands in the simulation is updated. At the same time, the performance record keeps track of the number of positive demands that have been satisfied from the stock, not adding to the backlog. The ratio between these two raw measurements is the fill rate of the simulation FR_{sim}.

In each simulation the procedure is repeated for each cycle to define the optimal order quantity O_{min}. The parameters S_e or S_{ne} cannot be changed and only positive values of O_{min} are produced. The fill rate is thus set to achieve $FR(O_{min}, S_e, S_{ne}) \geq FR_{obj}$. This goal setting leads to difficulties while comparing FR_{sim} and FR_{obj} since, by construction, on average $FR_{sim} \geq FR_{obj}$ thus the difference $FR_{sim} - FR_{obj}$ is designed to be ≥ 0. To avoid this unfair comparison the values of $FR(O_{min}, S_e, S_{ne})$ are collected in each simulation as they are generated and their average is compared with FR_{sim} instead of FR_{obj}:

$$\Delta FR = FR_{sim} - avg\big(FR(O_{min}, S_e, S_{ne})\big) \tag{8.17}$$

Results

Table 8.03 contains the expected value and sample standard deviation of ΔFR for each scenario.

Scenario	Expected value	Sample STD
Scenario 1	-0.01	0.036
Scenario 2	-0.09	0.04
Scenario 3	0.02	0.13

Table 8.03. Performance obtained.

In the first scenario the expected value of ΔFR is low, and the proposed model presents a small negative bias in its calculation of the optimal fill rate. The standard deviation is quite low as well, revealing the good performance of the model under ideal conditions. An Anderson-darling test is used in the first scenario on the ΔFR data to identify if the performance follows a Normal distribution. The test yields a p-value of 0.56, thus such hypothesis cannot be rejected.

In the second scenario the expected value of ΔFR increases, and this increased bias reveals that the approximation underestimates the expected value of the fill rate. At the contrary the standard deviation remains quite stable and thus, except for the bias, the approximation retains a performance close to the ideal one. An Anderson-darling test is used in the second scenario on the ΔFR data to identify if the performance follows a Normal distribution. The test yields a p-value of 0.002, and thus the normality hypothesis must be rejected.

In the third scenario the expected value of ΔFR is comparable in magnitude to the one in scenario 1 while the increased standard deviation identifies a sharp performance decrease. The performance reduction can be attributed both to the use of approximate values for the negative binomial parameters and to the lack of fit between the negative binomial itself and the positive demand values.

Conclusions

The results presented in the experimental analysis validate the effectiveness of the proposed method for the management of intermittent items characterised by perishability. The best performance, achieved when the methodology is applied in ideal conditions, proves the correctness of the underlying theory and acts as a benchmark for both the other analysis presented and for future applications.

A critical aspect emerged during the solution technique discussion in relation to the calculations' computational costs. In case multiple expiration dates arise, the use of approximations to reduce this computational cost presents significant bias, as outlined in the experimental analysis. Nevertheless, the standard deviation of the measured error remains close to the ideal scenario. Future research can thus focus on the isolation and correction of this bias term in order to uniform the performance of the approximate case with that of the ideal one.

The validation with real items presented an error standard deviation far higher than in the ideal case. This phenomenon can relate to a mismatch between the expected positive demand distribution and the proposed one. Future research could focus on the closure of this gap, with the application of sampling techniques to adapt the model to the positive demand distribution in real cases.

7. Inventory control with perishability - 2

This chapter is adapted from the paper "A periodic inventory system of intermittent demand items with fixed lifetimes" in press in 2019 in International Journal of Production Research [133].

Introduction

This chapter extends Chapter 6. The inventory control model presented there is validated through a two-level full factorial design experiment around the most significant variables, whereas in Chapter 6 only a scenario analysis is presented. The experimental results are statistically analysed with a linear regression, proving that the variables do not impact its performance and suggesting that the underlying model is unbiased. In addition, differently from Chapter 6, it is conducted a more realistic numerical investigation where the demand distribution parameters are forecasted, and this is integrated in the inventory model.

Experimental analysis

Experiment settings

The same forecasting technique used in Chapter 6 is applied to define $\hat{p}$, $\hat{z_i}$ and $\widehat{MSE}(z_i)$ and the parameter α is optimized over the initial warmup periods. From $\hat{z_i}$ and $\widehat{MSE}(z_i)$ the negative binomial distribution parameters are derived using the method of moments.

The experimental analysis consists of two experiments, carried out with different parameters, where the proposed methodology is tested on a generated series and compared against the case where the standard (R, S) policy is applied without accounting for the perishability constraint. In these simulations both methodologies consume the stock following a FIFO policy, which is in line with a fixed number of periods before expiration.

Intermittent demands are generated, and their positive demand size follows a negative binomial distribution, while the probability p that a positive demand occurs is time invariant. If the demand size distribution yields a null demand it is still considered a positive demand to avoid uncontrolled changes in p.

The fixed and variable parameters, varying in each simulation, for experiments 1 and 2 are summarised in Table 9.01. In both experiments the values for n_e are greater than those for t to avoid expirations before the end of a single cycle. All the possible combinations of the second set of parameters are tested 10 times to assess the method performance in different contexts.

	Parameter	Description	Value
Experiment 1 Fixed parameters	n_w	Number of warm-ups	100
	n_p	Number of simulated periods	1000
	R	Periods in a cycle	10
	L	Lead time	5
Experiment 1 Variable parameters	p	Demand probability	0.1, 0.5
	$E(z)$	Positive demand expected value	10, 20
	$\frac{\sqrt{MSE(z)}}{E(z)}$	Relative positive demand mean squared error	0.5, 1.4

	n_e	Periods before expiration	10, 15
Experiment 2 Fixed parameters	n_w	Number of warm-ups	100
	n_p	Number of simulated periods	1000
	FR_{target}	Fill rate target	0.8
Experiment 2 Variable parameters	p	Demand probability	0.1, 0.5
	$E(z)$	Positive demand expected value	10, 20
	$\frac{\sqrt{MSE(z)}}{E(z)}$	Relative positive demand mean squared error	0.5, 1.4
	n_e	Periods before expiration	20, 25
	R	Periods in a cycle	10, 20
	L	Lead time	2, 5

Table 9.01. Fixed and variable parameters for experiments 1 and 2.

The parameters of experiment 1 are designed to cover extreme scenarios:

- Low intermittence ($p = 0.5$) vs. high intermittence ($p = 0.1$).
- Low demand ($E(z) = 10$) vs. high demand ($E(z) = 20$).
- Low lumpiness ($\frac{\sqrt{MSE(z)}}{E(z)} = 0.5$) vs. high lumpiness ($\frac{\sqrt{MSE(z)}}{E(z)} = 1.4$).
- Close expiration date ($n_e = 10$) vs. distant expiration date ($n_e = 15$).

The parameters of experiment 2 are designed to cover a wider range:

- Low intermittence ($p = 0.5$) vs. high intermittence ($p = 0.1$).
- Low demand ($E(z) = 10$) vs. high demand ($E(z) = 20$).
- Low lumpiness ($\frac{\sqrt{MSE(z)}}{E(z)} = 0.5$) vs. high lumpiness ($\frac{\sqrt{MSE(z)}}{E(z)} = 1.4$).
- Distant expiration date ($n_e = 20$) vs. very distant expiration date ($n_e = 25$).
- Short cycle ($R = 10$) vs. long cycle ($R = 20$).
- Short lead-time ($L = 2$) vs. long lead-time ($L = 5$).

The aim of these experiments is to identify a structure in the algorithms' behaviour in order to define outperforming regions for the two methods. The results are collected for each simulation period after the first lead time, when the first order has already arrived. Using this precaution, no initial level of backorders and stock is required to make the first measurements fair.

Performance metrics

Performance measurements are recorder after each simulation period. If the period presents a positive demand, then the total number of positive demands in the simulation is updated. Simultaneously, the performance record keeps track of the number of positive demands that have been satisfied from the stock, not adding to the backlog. The ratio between these two raw measurements is the fill rate of the simulation FR_{sim}. At the beginning of each cycle the optimal order quantity O_{min} is defined. In this context parameters S_e or S_{ne} cannot be changed and only positive values of O_{min} are produced. The fill rate is thus set to achieve $FR(O_{min}, S_e, S_{ne}) \geq FR_{obj}$. This goal-setting leads to difficulties when comparing FR_{sim} and FR_{obj} as, by construction, on average $FR_{sim} \geq FR_{obj}$ thus the difference $FR_{sim} - FR_{obj}$ is designed to be ≥ 0. To avoid this unfair comparison the values of $FR(O_{min}, S_e, S_{ne})$ are collected in each simulation as they are generated, and their average is compared with FR_{sim} instead of FR_{obj}:

$$\Delta FR = FR_{sim} - avg\big(FR(O_{min}, S_e, S_{ne})\big) \tag{9.01}$$

To measure the benefit of using the proposed method rather than the standard method that does not consider perishability, ΔFR_{new} is also calculated. In the standard method the order-up-to-level (R, S) inventory policy does not account for the perishability constraint, as described in [3]. Such a standard method is expected to be biased and to underachieve the target fill rate.

$$\Delta FR_{new} = \left|\Delta FR_{pr}\right| - |\Delta FR_{st}| \tag{9.02}$$

ΔFR_{pr} is the ΔFR of the proposed method and ΔFR_{st} is the ΔFR of the standard (R, S) method.

Results

Tables 9.02 and 9.03 summarize the average ΔFR_{pr}, ΔFR_{st} and ΔFR_{new} results obtained for each combination of parameters. The values for $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ differ from those provided obtained from forecasting methods since the negative binomial the generated distribution, used for data generation, requires one of its parameters to be a natural number. Not any combination of $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ is allowed and their values end up changed when the parameter is rounded.

p	$E(z)$	$\frac{\sqrt{MSE(z)}}{E(z)}$	n_e	R	L	ΔFR_{pr}	ΔFR_{st}	ΔFR_{new}
0.1	10.5	0.488	10	10	5	-0.024	-0.289	-0.236
0.1	10.5	0.488	15	10	5	-0.067	-0.153	-0.047
0.5	10.5	0.488	10	10	5	-0.023	-0.140	-0.110
0.5	10.5	0.488	15	10	5	-0.033	-0.064	-0.029
0.1	18.6	1.027	10	10	5	0.015	-0.242	-0.163
0.1	18.6	1.027	15	10	5	0.004	-0.088	-0.029
0.5	18.6	1.027	10	10	5	0.011	-0.130	-0.110
0.5	18.6	1.027	15	10	5	-0.013	-0.058	-0.028
0.1	20	0.5	10	10	5	-0.003	-0.238	-0.195
0.1	20	0.5	15	10	5	-0.020	-0.133	-0.086
0.5	20	0.5	10	10	5	-0.004	-0.135	-0.112
0.5	20	0.5	15	10	5	0.008	-0.023	-0.009
0.1	38.2	1.013	10	10	5	-0.003	-0.230	-0.179

0.1	38.2	1.013	15	10	5	0.012	-0.068	-0.047
0.5	38.2	1.013	10	10	5	-0.002	-0.177	-0.153
0.5	38.2	1.013	15	10	5	-0.008	-0.052	-0.032

Table 9.02. Average performance obtained for each combination of parameters in experiment 1.

p	$E(z)$	$\frac{\sqrt{MSE(z)}}{E(z)}$	n_e	R	L	ΔFR_{pr}	ΔFR_{st}	ΔFR_{new}
0.1	10.5	0.488	20	10	2	-0.045	-0.158	-0.106
0.1	10.5	0.488	20	10	5	-0.019	-0.139	-0.095
0.1	10.5	0.488	20	20	2	-0.002	-0.196	-0.143
0.1	10.5	0.488	20	20	5	-0.009	-0.207	-0.153
0.1	10.5	0.488	25	10	2	0.016	-0.036	0.004
0.1	10.5	0.488	25	10	5	-0.011	-0.052	-0.03
0.1	10.5	0.488	25	20	2	-0.026	-0.168	-0.094
0.1	10.5	0.488	25	20	5	-0.045	-0.152	-0.089
0.1	18.6	1.027	20	10	2	0.033	-0.077	-0.024
0.1	18.6	1.027	20	10	5	0.018	-0.093	-0.058
0.1	18.6	1.027	20	20	2	-0.004	-0.23	-0.174
0.1	18.6	1.027	20	20	5	-0.088	-0.259	-0.13
0.1	18.6	1.027	25	10	2	0.048	0.006	-0.005
0.1	18.6	1.027	25	10	5	-0.001	-0.057	-0.019
0.1	18.6	1.027	25	20	2	0.006	-0.143	-0.116
0.1	18.6	1.027	25	20	5	0.035	-0.088	-0.054
0.1	20	0.5	20	10	2	-0.007	-0.135	-0.104
0.1	20	0.5	20	10	5	-0.028	-0.138	-0.097
0.1	20	0.5	20	20	2	0.01	-0.2	-0.149
0.1	20	0.5	20	20	5	-0.013	-0.209	-0.158
0.1	20	0.5	25	10	2	-0.003	-0.053	-0.004
0.1	20	0.5	25	10	5	-0.039	-0.091	-0.031
0.1	20	0.5	25	20	2	0	-0.143	-0.093

0.1	20	0.5	25	20	5	-0.001	-0.114	-0.036
0.1	38.2	1.013	20	10	2	0.011	-0.121	-0.083
0.1	38.2	1.013	20	10	5	0.067	-0.052	-0.031
0.1	38.2	1.013	20	20	2	0.011	-0.193	-0.145
0.1	38.2	1.013	20	20	5	0.049	-0.184	-0.131
0.1	38.2	1.013	25	10	2	0.047	0.002	0.001
0.1	38.2	1.013	25	10	5	0.03	-0.024	0.015
0.1	38.2	1.013	25	20	2	-0.015	-0.12	-0.09
0.1	38.2	1.013	25	20	5	-0.007	-0.15	-0.113
0.5	10.5	0.488	20	10	2	0.007	0.001	-0.001
0.5	10.5	0.488	20	10	5	-0.011	-0.031	-0.014
0.5	10.5	0.488	20	20	2	-0.021	-0.102	-0.065
0.5	10.5	0.488	20	20	5	-0.009	-0.079	-0.051
0.5	10.5	0.488	25	10	2	-0.019	-0.02	0
0.5	10.5	0.488	25	10	5	-0.02	-0.024	-0.004
0.5	10.5	0.488	25	20	2	0.008	-0.019	-0.017
0.5	10.5	0.488	25	20	5	-0.005	-0.039	-0.022
0.5	18.6	1.027	20	10	2	0.003	-0.032	-0.02
0.5	18.6	1.027	20	10	5	0.014	-0.018	-0.015
0.5	18.6	1.027	20	20	2	-0.015	-0.138	-0.106
0.5	18.6	1.027	20	20	5	0.003	-0.124	-0.101
0.5	18.6	1.027	25	10	2	0.023	0.009	0.004
0.5	18.6	1.027	25	10	5	-0.008	-0.025	0.001
0.5	18.6	1.027	25	20	2	-0.016	-0.087	-0.041
0.5	18.6	1.027	25	20	5	0.019	-0.053	-0.035
0.5	20	0.5	20	10	2	-0.01	-0.022	-0.009
0.5	20	0.5	20	10	5	0.007	-0.005	-0.006
0.5	20	0.5	20	20	2	0.005	-0.082	-0.068
0.5	20	0.5	20	20	5	0.004	-0.067	-0.052
0.5	20	0.5	25	10	2	-0.003	-0.005	-0.002

0.5	20	0.5	25	10	5	-0.004	-0.006	0.001
0.5	20	0.5	25	20	2	0.006	-0.025	-0.011
0.5	20	0.5	25	20	5	-0.001	-0.036	-0.018
0.5	38.2	1.013	20	10	2	0.006	-0.03	-0.019
0.5	38.2	1.013	20	10	5	0.001	-0.056	-0.034
0.5	38.2	1.013	20	20	2	-0.008	-0.126	-0.094
0.5	38.2	1.013	20	20	5	0.007	-0.13	-0.101
0.5	38.2	1.013	25	10	2	-0.005	-0.019	-0.007
0.5	38.2	1.013	25	10	5	0.002	-0.017	-0.004
0.5	38.2	1.013	25	20	2	0.018	-0.054	-0.026
0.5	38.2	1.013	25	20	5	-0.017	-0.093	-0.069

Table 9.03. Average performance obtained for each combination of parameters in experiment 2.

The standard (R, S) method is biased, leading to fill rate that are always lower than the target ($\Delta FR_{st} < 0$) in experiment 1 (Table 9.02), and lower than the target 93.7% of the times in experiment 2 (Table 9.03). In the proposed method, this bias is corrected and, as a result, the ΔFR_{pr} values are affected only by random error and thus characterized by varying signs. On average the new method produces a 9.78% fill rate performance increase for perishable items in experiment 1 and a 5.53% increase in experiment 2.

A linear regression is fitted over the non-averaged simulation results using p, $E(z)$, $\frac{\sqrt{MSE(z)}}{E(z)}$ and n_e, in the first experiment, and p, $E(z)$, $\frac{\sqrt{MSE(z)}}{E(z)}$, n_e, R and L, in the second experiment, as features to separately predict ΔFR and ΔFR_{new}. The values of $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ used as features are obtained from the negative binomial distribution parameters used for data generation, they differ from those listed in Tables 9.02 and 9.03 as one of the distribution parameters must be an integer and is rounded when calculated from the original $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$. As a result, $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ are correlated ($r = 0.64$), while the others are not. The way this issue is handled is outline below. The linear regressions coefficients and their t-tests are summarized in Table 9.04.

		Coefficient	Squared error	t	p-value
Experiment 1 ΔFR_{pr}	Intercept	-0.021	0.042	-0.490	0.625
	p	0.007	0.035	0.209	0.834
	$E(z)$	0.000	0.001	0.501	0.617

	$\frac{\sqrt{MSE(z)}}{E(z)}$	0.033	0.034	0.968	0.334
	n_e	-0.002	0.003	-0.744	0.458
Experiment 1 ΔFR_{st}	Intercept	-0.520	0.044	-11.752	0.000
	p	0.207	0.037	5.653	0.000
	$E(z)$	0.000	0.001	0.506	0.614
	$\frac{\sqrt{MSE(z)}}{E(z)}$	0.020	0.036	0.552	0.582
	n_e	0.024	0.003	8.046	0.000
Experiment 2 ΔFR_{pr}	Intercept	-0.016	0.025	-0.634	0.526
	p	-0.004	0.012	-0.365	0.715
	$E(z)$	0.000	0.000	1.582	0.114
	$\frac{\sqrt{MSE(z)}}{E(z)}$	0.021	0.012	1.733	0.084
	n_e	0.000	0.001	0.265	0.791
	R	-0.001	0.000	-1.423	0.155
	L	-0.001	0.002	-0.890	0.374
Experiment 2 ΔFR_{st}	Intercept	-0.267	0.028	-9.492	0.000
	p	0.189	0.014	13.928	0.000
	$E(z)$	0.000	0.000	0.612	0.541
	$\frac{\sqrt{MSE(z)}}{E(z)}$	-0.007	0.013	-0.504	0.614
	n_e	0.011	0.001	10.003	0.000
	R	-0.008	0.001	-14.342	0.000
	L	-0.001	0.002	-0.553	0.580

Table 9.04. Linear regression coefficients for ΔFR_{pr} and ΔFR_{st} in experiment 1 and 2.

The results of experiment 1 indicate that the performance of the proposed method is not affected by the simulation parameters, while experiment 2 shows that $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ together influence ΔFR_{pr}. The results of experiment 1 are highlighted by the t-tests in Table 9.04, which show no significance.

The correlation between $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ could hide a significant model behind non-significant t-tests. Consequently, to assess this scenario an F-test was performed which resulted in a non-significant p-value of 0.454. The results for experiment 2 are in line with findings from experiment 1. No t-test in experiment 2 reached a significant p-value, while the F-test in experiment 2 yielded a significant p-value of 0.005. Performing a PCA on standardized $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ and re-fitting the regression model revealed that those features in combination are the only ones that influence ΔFR_{pr}. The findings of experiments 1 and 2 are not inconsistent. In fact, the tests in experiment 2 leverage more simulations, resulting in a higher power and, as a result, the conclusions drawn from experiment 2 are more accurate.

Figure 9.01 highlights the coherence between experiments 1 and 2 and plots the confidence intervals (one std) for p, $E(z)$, $\frac{\sqrt{MSE(z)}}{E(z)}$ and n_e coefficients in experiment 1 (x) and experiment 2 (*). The coefficients for the most precise experiment are included in those for the least precise one.

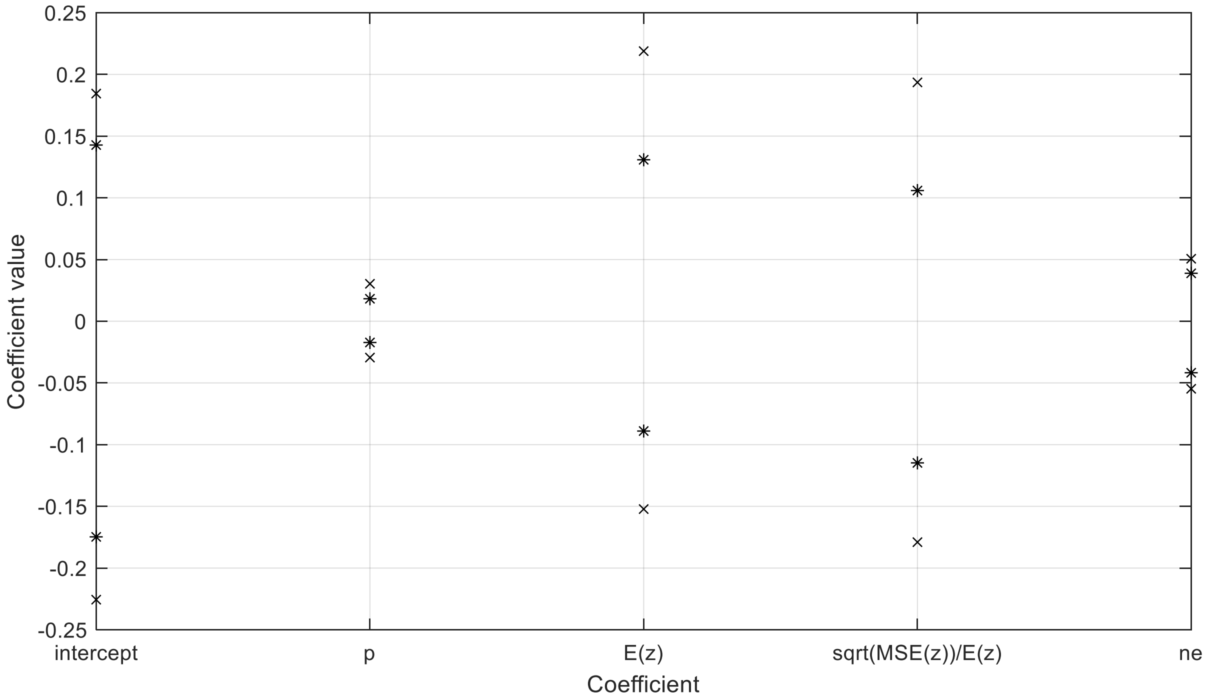

Figure 9.01. The confidence intervals of the coefficients in experiments 1 and 2.

According to experiment 1 (Table 9.04), the standard (R, S) is unaffected by demand size or variability as the p-values obtained are significantly higher than 0.05. The parameters affecting ΔFR_{st}, in addition to the intercept, are intermittence and periods before expiration. Experiment 2 confirms these findings, moreover re-fitting both experiments regression models after a PCA on standardized $E(z)$ and $\frac{\sqrt{MSE(z)}}{E(z)}$ does not highlight any impact of such features over ΔFR_{st}. The coefficients in experiments 1 and 2 are less coherent on their impact on ΔFR_{st} than those for ΔFR_{pr} depicted in Figure 9.01, suggesting more complex phenomena that cannot be defined with a simple linear model.

As a robustness test, the impact of the intermittent demand assumption on the proposed method is measured by rerunning experiment 1 while using a SES and erroneously assuming $p = 1$. The results are found to be reliant on the intermittence assumption as the Δfr_{pr} obtained in this scenario is even higher than the ΔFR_{st} achieved in experiment 1.

Conclusions

Managing the inventories of perishable items becomes a more challenging task when the demand for such items is intermittent. A simulation experiment is conducted to analyse the performance of the method proposed in Chapter 6. The results show that when a proportion of the stock is affected by perishability, the proposed methodology leads to a considerable benefit by reducing the bias in the fill rate, unlike the standard method. The experiments demonstrate that the proposed methodology bias is only affected by demand size. On the other hand, intermittence, lumpiness, or number of periods before expiration do not impact its performance. The standard method is also proven to be unaffected by lumpiness, its effectiveness is only dictated by the number of periods before expiration and intermittence. From a computational standpoint the new methodology is significantly more expensive than the old one, as a combinatorial number of cases must be analysed, so practitioners are advised to apply the new methodology to scenarios characterized by high intermittence and low demand size. The use of simulation techniques to manage multiple expiration dates is also advised, to overcome the difficulty of determining analytical solutions in this case, since this can provide reliable results in exchange for a reasonable computational effort. Further research efforts are expected to gauge the effectiveness of simulation techniques and compare them with the available analytical solutions. Alongside this research avenue, efforts will be directed towards the characterization of different compound Bernoulli distributions, with the aim of encompassing both integer and continuous positive demand sizes [6], [134]. Other distributions such as compound Poisson distributions [37], [135] or compound Erlang distributions [136] have also been used to model intermittent demand and can be considered in future research. Once this characterization is achieved, comparisons between different distributions can take place and the effect of incorrectly selecting the demand size distribution can be quantified. Another interesting avenue for further research would be to analyse the combined service and cost efficiency of the proposed methodology when compared to the standard one. Finally, it would be interesting to empirically show the benefits of the proposed model through an empirical investigation with real data as in [3].

References

[1] J. D. Croston, "Forecasting and Stock Control for Intermittent Demands," *Oper. Res. Q.*, vol. 23, no. 3, pp. 289–303, Sep. 1972, doi: 10.2307/3007885.

[2] Q. Hu, J. E. Boylan, H. Chen, and A. Labib, "OR in spare parts management: A review," *Eur. J. Oper. Res.*, vol. 266, no. 2, pp. 395–414, Apr. 2018, doi: 10.1016/j.ejor.2017.07.058.

[3] R. H. Teunter, A. A. Syntetos, and M. Z. Babai, "Determining order-up-to levels under periodic review for compound binomial (intermittent) demand," *Eur. J. Oper. Res.*, vol. 203, no. 3, pp. 619–624, Jun. 2010, doi: 10.1016/j.ejor.2009.09.013.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] R. M. Adelson, "Compound Poisson Distributions," *OR*, vol. 17, no. 1, p. 73, Mar. 1966, doi: 10.2307/3007241.

[6] A. A. Syntetos, M. Z. Babai, and N. Altay, "On the demand distributions of spare parts," *Int. J. Prod. Res.*, vol. 50, no. 8, pp. 2101–2117, Apr. 2012, doi: 10.1080/00207543.2011.562561.

[7] A. V. Rao, "A Comment on: Forecasting and Stock Control for Intermittent Demands," *Oper. Res. Q.*, vol. 24, no. 4, p. 639, Dec. 1973, doi: 10.2307/3008348.

[8] A. A. Syntetos and J. E. Boylan, "On the variance of intermittent demand estimates," *Int. J. Prod. Econ.*, vol. 128, no. 2, pp. 546–555, Dec. 2010, doi: 10.1016/j.ijpe.2010.07.005.

[9] C. R. Schultz, "Forecasting and Inventory Control for Sporadic Demand Under Periodic Review," *J. Oper. Res. Soc.*, vol. 38, no. 5, pp. 453–458, May 1987, doi: 10.1057/jors.1987.74.

[10] A. A. Syntetos and J. E. Boylan, "On the bias of intermittent demand estimates," *Int. J. Prod. Econ.*, vol. 71, no. 1–3, pp. 457–466, May 2001, doi: 10.1016/S0925-5273(00)00143-2.

[11] A. A. Syntetos and J. E. Boylan, "On the stock control performance of intermittent demand estimators," *Int. J. Prod. Econ.*, vol. 103, no. 1, pp. 36–47, Sep. 2006, doi: 10.1016/j.ijpe.2005.04.004.

[12] A. A. Syntetos and J. E. Boylan, "The accuracy of intermittent demand estimates," *Int. J. Forecast.*, vol. 21, no. 2, pp. 303–314, Apr. 2005, doi: 10.1016/j.ijforecast.2004.10.001.

[13] E. Levén and A. Segerstedt, "Inventory control with a modified Croston procedure and Erlang distribution," *Int. J. Prod. Econ.*, vol. 90, no. 3, pp. 361–367, Aug. 2004, doi: 10.1016/S0925-5273(03)00053-7.

[14] J. E. Boylan and A. A. Syntetos, "The accuracy of a Modified Croston procedure," *Int. J. Prod. Econ.*, vol. 107, no. 2, pp. 511–517, 2007, doi: 10.1016/j.ijpe.2006.10.005.

[15] R. H. Teunter and B. Sani, "On the bias of Croston's forecasting method," *Eur. J. Oper. Res.*, vol. 194, no. 1, pp. 177–183, Apr. 2009, doi: 10.1016/j.ejor.2007.12.001.

[16] R. H. Teunter, A. A. Syntetos, and M. Z. Babai, "Intermittent demand: Linking forecasting to inventory obsolescence," *Eur. J. Oper. Res.*, vol. 214, no. 3, pp. 606–615, Nov. 2011, doi: 10.1016/j.ejor.2011.05.018.

[17] R. S. Gutierrez, A. O. Solis, and S. Mukhopadhyay, "Lumpy demand forecasting using neural networks," *Int. J. Prod. Econ.*, vol. 111, no. 2, pp. 409–420, Feb. 2008, doi: 10.1016/j.ijpe.2007.01.007.

[18] N. Kourentzes, "Intermittent demand forecasts with neural networks," *Int. J. Prod. Econ.*, vol. 143, no. 1, pp. 198–206, May 2013, doi: 10.1016/j.ijpe.2013.01.009.

[19] E. A. Shale, J. E. Boylan, and F. R. Johnston, “Forecasting for intermittent demand: the estimation of an unbiased average,” *J. Oper. Res. Soc.*, vol. 57, no. 5, pp. 588–592, May 2006, doi: 10.1057/palgrave.jors.2602031.

[20] S. Mukhopadhyay, A. O. Solis, and R. S. Gutierrez, “The Accuracy of Non-traditional versus Traditional Methods of Forecasting Lumpy Demand,” *J. Forecast.*, vol. 31, no. 8, pp. 721–735, Dec. 2012, doi: 10.1002/for.1242.

[21] F. Lolli, R. Gamberini, A. Regattieri, E. Balugani, T. Gatos, and S. Gucci, “Single-hidden layer neural networks for forecasting intermittent demand,” *Int. J. Prod. Econ.*, vol. 183, no. May 2015, pp. 116–128, Jan. 2017, doi: 10.1016/j.ijpe.2016.10.021.

[22] B. Efron, “Bootstrap Methods: Another Look at the Jackknife,” *Ann. Stat.*, vol. 7, no. 1, pp. 1–26, Jan. 1979, doi: 10.1214/aos/1176344552.

[23] M. Hasni, M. S. Aguir, M. Z. Babai, and Z. Jemai, “Spare parts demand forecasting: a review on bootstrapping methods,” *Int. J. Prod. Res.*, vol. 57, no. 15–16, pp. 4791–4804, Aug. 2019, doi: 10.1080/00207543.2018.1424375.

[24] A. A. Syntetos, M. Z. Babai, and E. S. Gardner, “Forecasting intermittent inventory demands: simple parametric methods vs. bootstrapping,” *J. Bus. Res.*, vol. 68, no. 8, pp. 1746–1752, Aug. 2015, doi: 10.1016/j.jbusres.2015.03.034.

[25] V. Sillanpää and J. Liesiö, “Forecasting replenishment orders in retail: value of modelling low and intermittent consumer demand with distributions,” *Int. J. Prod. Res.*, vol. 56, no. 12, pp. 4168–4185, Jun. 2018, doi: 10.1080/00207543.2018.1431413.

[26] M. Hasni, M. Z. Babai, M. S. Aguir, and Z. Jemai, “An investigation on bootstrapping forecasting methods for intermittent demands,” *Int. J. Prod. Econ.*, no. October 2016, pp. 1–10, Mar. 2018, doi: 10.1016/j.ijpe.2018.03.001.

[27] A. Bacchetti and N. Saccani, “Spare parts classification and demand forecasting for stock control: Investigating the gap between research and practice,” *Omega*, vol. 40, no. 6, pp. 722–737, Dec. 2012, doi: 10.1016/j.omega.2011.06.008.

[28] F. Janssen, R. Heuts, and T. de Kok, “On the (R, s, Q) inventory model when demand is modelled as a compound Bernoulli process,” *Eur. J. Oper. Res.*, vol. 104, no. 3, pp. 423–436, Feb. 1998, doi: 10.1016/S0377-2217(97)00009-X.

[29] L. W. G. Strijbosch, R. M. J. Heuts, and E. H. M. van der Schoot, “A Combined Forecast-Inventory Control Procedure for Spare Parts,” *J. Oper. Res. Soc.*, vol. 51, no. 10, p. 1184, Oct. 2000, doi: 10.2307/253931.

[30] R. H. Teunter and B. Sani, “Calculating order-up-to levels for products with intermittent demand,” *Int. J. Prod. Econ.*, vol. 118, no. 1, pp. 82–86, Mar. 2009, doi: 10.1016/j.ijpe.2008.08.012.

[31] T. M. Williams, “Stock Control with Sporadic and Slow-Moving Demand,” *J. Oper. Res. Soc.*, vol. 35, no. 10, pp. 939–948, Oct. 1984, doi: 10.1057/jors.1984.185.

[32] A. A. Syntetos, J. E. Boylan, and J. D. Croston, “On the categorization of demand patterns,” *J. Oper. Res. Soc.*, vol. 56, no. 5, pp. 495–503, May 2005, doi: 10.1057/palgrave.jors.2601841.

[33] B. Sani and B. G. Kingsman, “Selecting the best periodic inventory control and demand forecasting methods for low demand items,” *J. Oper. Res. Soc.*, vol. 48, no. 7, pp. 700–713, Jul. 1997, doi: 10.1057/palgrave.jors.2600418.

[34] A. V Kostenko and R. J. Hyndman, "A note on the categorization of demand patterns," *J. Oper. Res. Soc.*, vol. 57, no. 10, pp. 1256–1257, Oct. 2006, doi: 10.1057/palgrave.jors.2602211.

[35] A. A. Syntetos, J. E. Boylan, and J. D. Croston, "Reply to Kostenko and Hyndman," *J. Oper. Res. Soc.*, vol. 57, no. 10, pp. 1257–1258, Oct. 2006, doi: 10.1057/palgrave.jors.2602182.

[36] G. Heinecke, A. A. Syntetos, and W. Wang, "Forecasting-based SKU classification," *Int. J. Prod. Econ.*, vol. 143, no. 2, pp. 455–462, Jun. 2013, doi: 10.1016/j.ijpe.2011.11.020.

[37] D. Lengu, A. A. Syntetos, and M. Z. Babai, "Spare parts management: Linking distributional assumptions to demand classification," *Eur. J. Oper. Res.*, vol. 235, no. 3, pp. 624–635, Jun. 2014, doi: 10.1016/j.ejor.2013.12.043.

[38] F. Lolli, E. Balugani, A. Ishizaka, R. Gamberini, B. Rimini, and A. Regattieri, "Machine learning for multi-criteria inventory classification applied to intermittent demand," *Prod. Plan. Control*, vol. 30, no. 1, pp. 76–89, Jan. 2019, doi: 10.1080/09537287.2018.1525506.

[39] G. C. Onwubolu and B. C. Dube, "Implementing an improved inventory control system in a small company: a case study," *Prod. Plan. Control*, vol. 17, no. 1, pp. 67–76, Jan. 2006, doi: 10.1080/09537280500366001.

[40] G. Nenes, S. Panagiotidou, and G. Tagaras, "Inventory management of multiple items with irregular demand: A case study," *Eur. J. Oper. Res.*, vol. 205, no. 2, pp. 313–324, Sep. 2010, doi: 10.1016/j.ejor.2009.12.022.

[41] M. Z. Babai, T. Ladhari, and I. Lajili, "On the inventory performance of multi-criteria classification methods: empirical investigation," *Int. J. Prod. Res.*, vol. 53, no. 1, pp. 279–290, Jan. 2015, doi: 10.1080/00207543.2014.952791.

[42] B. E. Flores and D. Clay Whybark, "Multiple Criteria ABC Analysis," *Int. J. Oper. Prod. Manag.*, vol. 6, no. 3, pp. 38–46, Mar. 1986, doi: 10.1108/eb054765.

[43] B. E. Flores, D. L. Olson, and V. K. Dorai, "Management of multicriteria inventory classification," *Math. Comput. Model.*, vol. 16, no. 12, pp. 71–82, Dec. 1992, doi: 10.1016/0895-7177(92)90021-C.

[44] F. Y. Partovi and J. Burton, "Using the Analytic Hierarchy Process for ABC Analysis," *Int. J. Oper. Prod. Manag.*, vol. 13, no. 9, pp. 29–44, Sep. 1993, doi: 10.1108/01443579310043619.

[45] F. Y. Partovi and W. E. Hopton, "Analytic hierarchy process as applied to two types of inventory problems," *Prod. Invent. Manag. J.*, vol. 35, no. 1, pp. 13–19, 1994.

[46] O. Cakir and M. S. Canbolat, "A web-based decision support system for multi-criteria inventory classification using fuzzy AHP methodology," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 1367–1378, Oct. 2008, doi: 10.1016/j.eswa.2007.08.041.

[47] G. Kabir and M. A. Akhtar Hasin, "Multiple criteria inventory classification using fuzzy analytic hierarchy process," *Int. J. Ind. Eng. Comput.*, vol. 3, no. 2, pp. 123–132, Jan. 2012, doi: 10.5267/j.ijiec.2011.09.007.

[48] F. Lolli, A. Ishizaka, and R. Gamberini, "New AHP-based approaches for multi-criteria inventory classification," *Int. J. Prod. Econ.*, vol. 156, pp. 62–74, Oct. 2014, doi: 10.1016/j.ijpe.2014.05.015.

[49] A. Bhattacharya, B. Sarkar, and S. K. Mukherjee, "Distance-based consensus method for ABC analysis," *Int. J. Prod. Res.*, vol. 45, no. 15, pp. 3405–3420, Aug. 2007, doi: 10.1080/00207540600847145.

[50] R. Ramanathan, "ABC inventory classification with multiple-criteria using weighted linear optimization," *Comput. Oper. Res.*, vol. 33, no. 3, pp. 695–700, Mar. 2006, doi: 10.1016/j.cor.2004.07.014.

[51] P. Zhou and L. Fan, "A note on multi-criteria ABC inventory classification using weighted linear optimization," *Eur. J. Oper. Res.*, vol. 182, no. 3, pp. 1488–1491, Nov. 2007, doi: 10.1016/j.ejor.2006.08.052.

[52] W. L. Ng, "A simple classifier for multiple criteria ABC analysis," *Eur. J. Oper. Res.*, vol. 177, no. 1, pp. 344–353, Feb. 2007, doi: 10.1016/j.ejor.2005.11.018.

[53] A. Hadi-Vencheh, "An improvement to multiple criteria ABC inventory classification," *Eur. J. Oper. Res.*, vol. 201, no. 3, pp. 962–965, Mar. 2010, doi: 10.1016/j.ejor.2009.04.013.

[54] J.-X. Chen, "Peer-estimation for multiple criteria ABC inventory classification," *Comput. Oper. Res.*, vol. 38, no. 12, pp. 1784–1791, Dec. 2011, doi: 10.1016/j.cor.2011.02.015.

[55] S. A. Torabi, S. M. Hatefi, and B. Saleck Pay, "ABC inventory classification in the presence of both quantitative and qualitative criteria," *Comput. Ind. Eng.*, vol. 63, no. 2, pp. 530–537, Sep. 2012, doi: 10.1016/j.cie.2012.04.011.

[56] J. Park, H. Bae, and J. Bae, "Cross-evaluation-based weighted linear optimization for multi-criteria ABC inventory classification," *Comput. Ind. Eng.*, vol. 76, pp. 40–48, Oct. 2014, doi: 10.1016/j.cie.2014.07.020.

[57] S. M. Hatefi and S. A. Torabi, "A Common Weight Linear Optimization Approach for Multicriteria ABC Inventory Classification," *Adv. Decis. Sci.*, vol. 2015, pp. 1–11, Jan. 2015, doi: 10.1155/2015/645746.

[58] J. Rezaei and S. Dowlatshahi, "A rule-based multi-criteria approach to inventory classification," *Int. J. Prod. Res.*, vol. 48, no. 23, pp. 7107–7126, Dec. 2010, doi: 10.1080/00207540903348361.

[59] Y. Chen, K. W. Li, D. Marc Kilgour, and K. W. Hipel, "A case-based distance model for multiple criteria ABC analysis," *Comput. Oper. Res.*, vol. 35, no. 3, pp. 776–796, Mar. 2008, doi: 10.1016/j.cor.2006.03.024.

[60] B. Soylu and B. Akyol, "Multi-criteria inventory classification with reference items," *Comput. Ind. Eng.*, vol. 69, pp. 12–20, Mar. 2014, doi: 10.1016/j.cie.2013.12.011.

[61] D. Mohammaditabar, S. Hassan Ghodsypour, and C. O'Brien, "Inventory control system design by integrating inventory classification and policy selection," *Int. J. Prod. Econ.*, vol. 140, no. 2, pp. 655–659, Dec. 2012, doi: 10.1016/j.ijpe.2011.03.012.

[62] A. Bacchetti, F. Plebani, N. Saccani, and A. A. Syntetos, "Empirically-driven hierarchical classification of stock keeping units," *Int. J. Prod. Econ.*, vol. 143, no. 2, pp. 263–274, Jun. 2013, doi: 10.1016/j.ijpe.2012.06.010.

[63] C.-Y. Tsai and S.-W. Yeh, "A multiple objective particle swarm optimization approach for inventory classification," *Int. J. Prod. Econ.*, vol. 114, no. 2, pp. 656–666, Aug. 2008, doi: 10.1016/j.ijpe.2008.02.017.

[64] S.-T. Wang and M.-H. Li, "An Analysis of the Optimal Multiobjective Inventory Clustering Decision with Small Quantity and Great Variety Inventory by Applying a DPSO," *Sci. World J.*, vol. 2014, pp. 1–15, 2014, doi: 10.1155/2014/805879.

[65] M. A. Millstein, L. Yang, and H. Li, "Optimizing ABC inventory grouping decisions," *Int. J.*

Prod. Econ., vol. 148, pp. 71–80, Feb. 2014, doi: 10.1016/j.ijpe.2013.11.007.

[66] N. M. Scala, J. Rajgopal, and K. L. Needy, "Managing Nuclear Spare Parts Inventories: A Data Driven Methodology," *IEEE Trans. Eng. Manag.*, vol. 61, no. 1, pp. 28–37, Feb. 2014, doi: 10.1109/TEM.2013.2283170.

[67] T. Ladhari, M. Z. Babai, and I. Lajili, "Multi-criteria inventory classification: new consensual procedures," *IMA J. Manag. Math.*, vol. 27, no. 2, pp. 335–351, Apr. 2016, doi: 10.1093/imaman/dpv003.

[68] A. Ishizaka, F. Lolli, E. Balugani, R. Cavallieri, and R. Gamberini, "DEASort: Assigning items with data envelopment analysis in ABC classes," *Int. J. Prod. Econ.*, vol. 199, pp. 7–15, May 2018, doi: 10.1016/j.ijpe.2018.02.007.

[69] R. Ernst and M. A. Cohen, "Operations related groups (ORGs): A clustering procedure for production/inventory systems," *J. Oper. Manag.*, vol. 9, no. 4, pp. 574–598, Oct. 1990, doi: 10.1016/0272-6963(90)90010-B.

[70] R. Q. Zhang, W. J. Hopp, and C. Supatgiat, "Spreadsheet implementable inventory control for a distribution center," *J. Heuristics*, vol. 7, no. 2, pp. 185–203, 2001, doi: 10.1023/A:1009613921001.

[71] R. H. Teunter, M. Z. Babai, and A. A. Syntetos, "ABC Classification: Service Levels and Inventory Costs," *Prod. Oper. Manag.*, vol. 19, no. 3, pp. 343–352, Nov. 2010, doi: 10.1111/j.1937-5956.2009.01098.x.

[72] A. Ishizaka and P. Nemery, *Multi-Criteria Decision Analysis: Methods and Software*. John Wiley & Sons Inc, 2013.

[73] F. Y. Partovi and M. Anandarajan, "Classifying inventory using an artificial neural network approach," *Comput. Ind. Eng.*, vol. 41, no. 4, pp. 389–404, Feb. 2002, doi: 10.1016/S0360-8352(01)00064-X.

[74] M.-C. Yu, "Multi-criteria ABC analysis using artificial-intelligence-based classification techniques," *Expert Syst. Appl.*, vol. 38, no. 4, pp. 3416–3421, Apr. 2011, doi: 10.1016/j.eswa.2010.08.127.

[75] R. A. Reid, "The ABC method in hospital inventory management: A practical approach," *Prod. Invent. Manag. J.*, vol. 28, no. 4, pp. 67–70, 1987.

[76] G. Kabir and M. A. A. Hasin, "Multi-criteria inventory classification through integration of fuzzy analytic hierarchy process and artificial neural network," *Int. J. Ind. Syst. Eng.*, vol. 14, no. 1, p. 74, 2013, doi: 10.1504/IJISE.2013.052922.

[77] D. López-Soto, S. Yacout, and F. Angel-Bello, "Root cause analysis of familiarity biases in classification of inventory items based on logical patterns recognition," *Comput. Ind. Eng.*, vol. 93, pp. 121–130, Mar. 2016, doi: 10.1016/j.cie.2015.12.011.

[78] H. Kartal, A. Oztekin, A. Gunasekaran, and F. Cebi, "An integrated decision analytic framework of machine learning with multi-criteria decision making for multi-attribute inventory classification," *Comput. Ind. Eng.*, vol. 101, pp. 599–613, Nov. 2016, doi: 10.1016/j.cie.2016.06.004.

[79] A. A. Syntetos, M. Z. Babai, J. Davies, and D. Stephenson, "Forecasting and stock control: A study in a wholesaling context," *Int. J. Prod. Econ.*, vol. 127, no. 1, pp. 103–111, Sep. 2010, doi: 10.1016/j.ijpe.2010.05.001.

[80] V. N. Vapnik, *The Nature of Statistical Learning Theory*, vol. 8, no. 6. New York, NY: Springer New York, 2000.

[81] S. Escalera, O. Pujol, and P. Radeva, “Separability of ternary codes for sparse designs of error-correcting output codes,” *Pattern Recognit. Lett.*, vol. 30, no. 3, pp. 285–297, Feb. 2009, doi: 10.1016/j.patrec.2008.10.002.

[82] F. Lolli, A. Ishizaka, R. Gamberini, E. Balugani, and B. Rimini, “Decision Trees for Supervised Multi-criteria Inventory Classification,” *Procedia Manuf.*, vol. 11, pp. 1871–1881, 2017, doi: 10.1016/j.promfg.2017.07.326.

[83] E. Balugani, F. Lolli, R. Gamberini, B. Rimini, and A. Regattieri, “Clustering for inventory control systems,” *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1174–1179, 2018, doi: 10.1016/j.ifacol.2018.08.431.

[84] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” *Proc. 5th Berkeley Symp. Math. Stat. Probab.*, vol. 1, pp. 281–297, 1967.

[85] J. H. Ward, “Hierarchical Grouping to Optimize an Objective Function,” *J. Am. Stat. Assoc.*, vol. 58, no. 301, p. 236, Mar. 1963, doi: 10.2307/2282967.

[86] G. N. Lance and W. T. Williams, “A General Theory of Classificatory Sorting Strategies: 1. Hierarchical Systems,” *Comput. J.*, vol. 9, no. 4, pp. 373–380, Feb. 1967, doi: 10.1093/comjnl/9.4.373.

[87] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.

[88] M. A. Sellitto, E. Balugani, and F. Lolli, “Spare Parts Replacement Policy Based on Chaotic Models,” *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 945–950, 2018, doi: 10.1016/j.ifacol.2018.08.486.

[89] A. Crespo Marquez and J. N. D. Gupta, “Contemporary maintenance management: process, framework and supporting pillars,” *Omega*, vol. 34, no. 3, pp. 313–326, Jun. 2006, doi: 10.1016/j.omega.2004.11.003.

[90] J. Huiskonen, “Maintenance spare parts logistics: Special characteristics and strategic choices,” *Int. J. Prod. Econ.*, vol. 71, no. 1–3, pp. 125–133, May 2001, doi: 10.1016/S0925-5273(00)00112-2.

[91] J. R. do Rego and M. A. de Mesquita, “Demand forecasting and inventory control: A simulation study on automotive spare parts,” *Int. J. Prod. Econ.*, vol. 161, pp. 1–16, Mar. 2015, doi: 10.1016/j.ijpe.2014.11.009.

[92] M. Guajardo, M. Rönnqvist, A. M. Halvorsen, and S. I. Kallevik, “Inventory management of spare parts in an energy company,” *J. Oper. Res. Soc.*, vol. 66, no. 2, pp. 331–341, Feb. 2015, doi: 10.1057/jors.2014.8.

[93] A. Garg and S. G. Deshmukh, “Maintenance management: literature review and directions,” *J. Qual. Maint. Eng.*, vol. 12, no. 3, pp. 205–238, Jul. 2006, doi: 10.1108/13552510610685075.

[94] A. H. C. Tsang, “Strategic dimensions of maintenance management,” *J. Qual. Maint. Eng.*, vol. 8, no. 1, pp. 7–39, Mar. 2002, doi: 10.1108/13552510210420577.

[95] D. Sherwin, “A review of overall models for maintenance management,” *J. Qual. Maint. Eng.*, vol. 6, no. 3, pp. 138–164, Sep. 2000, doi: 10.1108/13552510010341171.

[96] S. Cavalieri, M. Garetti, M. Macchi, and R. Pinto, “A decision-making framework for

managing maintenance spare parts," *Prod. Plan. Control*, vol. 19, no. 4, pp. 379–396, Jun. 2008, doi: 10.1080/09537280802034471.

[97] J. T. Luxhøj, J. O. Riis, and U. Thorsteinsson, "Trends and perspectives in industrial maintenance management," *J. Manuf. Syst.*, vol. 16, no. 6, pp. 437–453, Jan. 1997, doi: 10.1016/S0278-6125(97)81701-3.

[98] K. Efthymiou, A. Pagoropoulos, N. Papakostas, D. Mourtzis, and G. Chryssolouris, "Manufacturing systems complexity: An assessment of manufacturing performance indicators unpredictability," *CIRP J. Manuf. Sci. Technol.*, vol. 7, no. 4, pp. 324–334, 2014, doi: 10.1016/j.cirpj.2014.07.003.

[99] D. M. Louit, R. Pascual, and A. K. S. Jardine, "A practical procedure for the selection of time-to-failure models based on the assessment of trends in maintenance data," *Reliab. Eng. Syst. Saf.*, vol. 94, no. 10, pp. 1618–1628, Oct. 2009, doi: 10.1016/j.ress.2009.04.001.

[100] L. M. Leemis, "Nonparametric Estimation of the Cumulative Intensity Function for a Nonhomogeneous Poisson Process," *Manage. Sci.*, vol. 37, no. 7, pp. 886–900, 1991.

[101] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*, 2nd ed. Wiley-Interscience, 2003.

[102] W. Kahle, "Optimal maintenance policies in incomplete repair models," *Reliab. Eng. Syst. Saf.*, vol. 92, no. 5, pp. 563–565, May 2007, doi: 10.1016/j.ress.2006.05.004.

[103] K. S. Park, "Optimal Number of Minimal Repairs before Replacement," *IEEE Trans. Reliab.*, vol. R-28, no. 2, pp. 137–140, Jun. 1979, doi: 10.1109/TR.1979.5220523.

[104] R. A. Thiétart and B. Forgues, "Chaos Theory and Organization," *Organ. Sci.*, vol. 6, no. 1, pp. 19–31, Feb. 1995, doi: 10.1287/orsc.6.1.19.

[105] R. Capeáns, J. Sabuco, M. A. F. Sanjuán, and J. A. Yorke, "Partially controlling transient chaos in the Lorenz equations," *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 375, no. 2088, p. 20160211, Mar. 2017, doi: 10.1098/rsta.2016.0211.

[106] D. Yang, G. Li, and G. Cheng, "On the efficiency of chaos optimization algorithms for global optimization," *Chaos, Solitons & Fractals*, vol. 34, no. 4, pp. 1366–1375, Nov. 2007, doi: 10.1016/j.chaos.2006.04.057.

[107] S. C. Phatak and S. S. Rao, "Logistic map: A possible random-number generator," *Phys. Rev. E*, vol. 51, no. 4, pp. 3670–3678, Apr. 1995, doi: 10.1103/PhysRevE.51.3670.

[108] R. Capeáns, J. Sabuco, and M. A. F. Sanjuán, "Parametric partial control of chaotic systems," *Nonlinear Dyn.*, vol. 86, no. 2, pp. 869–876, Oct. 2016, doi: 10.1007/s11071-016-2929-4.

[109] M. Hekimoğlu, E. van der Laan, and R. Dekker, "Markov-modulated analysis of a spare parts system with random lead times and disruption risks," *Eur. J. Oper. Res.*, vol. 269, no. 3, pp. 909–922, Sep. 2018, doi: 10.1016/j.ejor.2018.02.040.

[110] N. Gebraeel, A. Elwany, and Jing Pan, "Residual Life Predictions in the Absence of Prior Degradation Knowledge," *IEEE Trans. Reliab.*, vol. 58, no. 1, pp. 106–117, Mar. 2009, doi: 10.1109/TR.2008.2011659.

[111] A. K. S. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mech. Syst. Signal Process.*, vol. 20, no. 7, pp. 1483–1510, Oct. 2006, doi: 10.1016/j.ymssp.2005.09.012.

[112] A. H. Christer, "Developments in delay time analysis for modelling plant maintenance," *J.*

Oper. Res. Soc., vol. 50, no. 11, pp. 1120–1137, Nov. 1999, doi: 10.1057/palgrave.jors.2600837.

[113] A. Ahmadi, T. Fransson, A. Crona, M. Klein, and P. Soderholm, "Integration of RCM and PHM for the next generation of aircraft," in *2009 IEEE Aerospace conference*, 2009, pp. 1–9, doi: 10.1109/AERO.2009.4839684.

[114] I. Bazovsky, *Reliability Theory and Practice*. Dover Publications, 2004.

[115] E. Balugani, F. Lolli, R. Gamberini, and B. Rimini, "Inventory control system for intermittent items with perishability," in *24th International Conference on Production Research (ICPR)*, 2017, pp. 780–785, doi: 10.12783/dtetr/icpr2017/17708.

[116] F. Raafat, "Survey of Literature on Continuously Deteriorating Inventory Models," *J. Oper. Res. Soc.*, vol. 42, no. 1, pp. 27–37, Jan. 1991, doi: 10.1057/jors.1991.4.

[117] S. K. Goyal and B. C. Giri, "Recent trends in modeling of deteriorating inventory," *Eur. J. Oper. Res.*, vol. 134, no. 1, pp. 1–16, Oct. 2001, doi: 10.1016/S0377-2217(00)00248-4.

[118] M. Bakker, J. Riezebos, and R. H. Teunter, "Review of inventory systems with deterioration since 2001," *Eur. J. Oper. Res.*, vol. 221, no. 2, pp. 275–284, Sep. 2012, doi: 10.1016/j.ejor.2012.03.004.

[119] A. S. White and M. Censlive, "Simulation of Three-Tier Supply Chains with Product Shelf-Life Effects," *Supply Chain Forum An Int. J.*, vol. 16, no. 1, pp. 26–44, Jan. 2015, doi: 10.1080/16258312.2015.11517365.

[120] C. Kouki, Z. Jemai, E. Sahin, and Y. Dallery, "Analysis of a periodic review inventory control system with perishables having random lifetime," *Int. J. Prod. Res.*, vol. 52, no. 1, pp. 283–298, Jan. 2014, doi: 10.1080/00207543.2013.839895.

[121] C. Kouki, M. Z. Babai, Z. Jemai, and S. Minner, "A coordinated multi-item inventory system for perishables with random lifetime," *Int. J. Prod. Econ.*, vol. 181, pp. 226–237, Nov. 2016, doi: 10.1016/j.ijpe.2016.01.013.

[122] C. Kouki, M. Z. Babai, and S. Minner, "On the benefit of dual-sourcing in managing perishable inventory," *Int. J. Prod. Econ.*, vol. 204, pp. 1–17, Oct. 2018, doi: 10.1016/j.ijpe.2018.06.015.

[123] A. Gutierrez-Alcoba, R. Rossi, B. Martin-Barragan, and E. M. T. Hendrix, "A simple heuristic for perishable item inventory control under non-stationary stochastic demand," *Int. J. Prod. Res.*, vol. 55, no. 7, pp. 1885–1897, 2017, doi: 10.1080/00207543.2016.1193248.

[124] S. Minner and S. Transchel, "Periodic review inventory-control for perishable products under service-level constraints," *OR Spectr.*, vol. 32, no. 4, pp. 979–996, 2010, doi: 10.1007/s00291-010-0196-1.

[125] X. Chen, Z. Pang, and L. Pan, "Coordinating Inventory Control and Pricing Strategies for Perishable Products," *Oper. Res.*, vol. 62, no. 2, pp. 284–300, Apr. 2014, doi: 10.1287/opre.2014.1261.

[126] Y. Duan, Y. Cao, and J. Huo, "Optimal pricing, production, and inventory for deteriorating items under demand uncertainty: The finite horizon case," *Appl. Math. Model.*, vol. 58, pp. 331–348, Jun. 2018, doi: 10.1016/j.apm.2018.02.004.

[127] K. G. J. Pauls-Worm, E. M. T. Hendrix, R. Haijema, and J. G. A. J. van der Vorst, "An MILP approximation for ordering perishable products with non-stationary demand and service level constraints," *Int. J. Prod. Econ.*, vol. 157, no. 1, pp. 133–146, Nov. 2014, doi:

10.1016/j.ijpe.2014.07.020.

[128] K. G. J. Pauls-Worm, E. M. T. Hendrix, A. G. Alcoba, and R. Haijema, “Order quantities for perishable inventory control with non-stationary demand and a fill rate constraint,” *Int. J. Prod. Econ.*, vol. 181, pp. 238–246, 2016, doi: 10.1016/j.ijpe.2015.10.009.

[129] C. Muriana, “An EOQ model for perishable products with fixed shelf life under stochastic demand conditions,” *Eur. J. Oper. Res.*, vol. 255, no. 2, pp. 388–396, Dec. 2016, doi: 10.1016/j.ejor.2016.04.036.

[130] E. A. Silver, “Inventory control under a probabilistic time-varying, demand pattern,” *AIE Trans.*, vol. 10, no. 4, pp. 371–379, Dec. 1978, doi: 10.1080/05695557808975228.

[131] L. Janssen, J. Sauer, T. Claus, and U. Nehls, “Development and simulation analysis of a new perishable inventory model with a closing days constraint under non-stationary stochastic demand,” *Comput. Ind. Eng.*, vol. 118, pp. 9–22, Apr. 2018, doi: 10.1016/j.cie.2018.02.016.

[132] A. Kara and I. Dogan, “Reinforcement learning approaches for specifying ordering policies of perishable inventory systems,” *Expert Syst. Appl.*, vol. 91, pp. 150–158, Jan. 2018, doi: 10.1016/j.eswa.2017.08.046.

[133] E. Balugani, F. Lolli, R. Gamberini, B. Rimini, and M. Z. Babai, “A periodic inventory system of intermittent demand items with fixed lifetimes,” *Int. J. Prod. Res.*, Jan. 2019, doi: 10.1080/00207543.2019.1572935.

[134] A. A. Syntetos, D. Lengu, and M. Z. Babai, “A note on the demand distributions of spare parts,” *Int. J. Prod. Res.*, vol. 51, no. 21, pp. 6356–6358, Nov. 2013, doi: 10.1080/00207543.2013.798050.

[135] M. Z. Babai, Z. Jemai, and Y. Dallery, “Analysis of order-up-to-level inventory systems with compound Poisson demand,” *Eur. J. Oper. Res.*, vol. 210, no. 3, pp. 552–558, May 2011, doi: 10.1016/j.ejor.2010.10.004.

[136] S. Saidane, M. Z. Babai, M. S. Aguir, and O. Korbaa, “On the performance of the base-stock inventory system under a compound Erlang demand distribution,” *Comput. Ind. Eng.*, vol. 66, no. 3, pp. 548–554, Nov. 2013, doi: 10.1016/j.cie.2013.01.015.