UNIVERSITÀ DI PARMA

UNIVERSITA' DEGLI STUDI DI PARMA

DOTTORATO DI RICERCA IN
"*INGEGNERIA INDUSTRIALE*"

CICLO XXXII

INDUSTRIAL APPLICATIONS OF MACHINE LEARNING AND DEEP LEARNING ALGORITHMS

Coordinatore:
Chiar.mo Prof. Gianni Royer Carfagni

Tutor:
Chiar.mo Prof. Ing. Bertolini Massimo
Chiar.mo Prof. Ing. Zammori Francesco

Co-tutor:
Chiar.mo Prof. Ganugi Piero

Dottorando: Davide Mezzogori

Anni 2016/2019

Summary

Publications

Part of the work within this thesis have been documented in the following publications:

D. Mezzogori, F. Zammori. "An Entity Embeddings Deep Learning approach for Demand Forecast of Highly Differentiated Products". *25th International Conference on Production Research Manufacturing Innovation*

D. Mezzogori, G. Romagnoli, F. Zammori. "Deep learning and card-based control systems: how to set realistic delivery dates in high-variety manufacturing systems". *9th IFAC Conference Manufacturing Modelling, Management and Control*

Under review:

D. Mezzogori, F. Zammori. "Industrial applications of Machine Learning algorithms: a literature review". *Expert Systems with Applications*

P. Ganugi, C. Ferretti, **D. Mezzogori**, F. Zammori. "Workload control with shifting bottlenecks: norms optimisation through Design of Experiments" *Production and Operations Management*

D. Bragoli, C. Ferretti, P. Ganugi, G. Marseguerra, **D. Mezzogori**, F. Zammori. "Machine Learning models for bankruptcy prediction in Italy: do industrial variables count?" *Spatial Economic Analysis*

Declaration

I declare that this thesis is my own work unless otherwise stated. No part of this thesis has previously been submitted for a degree or any other qualification at Università di Parma or any other institution.

1. Introduction

1.1. Background and objective

During the last decade, the advances in Artificial Intelligence algorithms, especially in Machine Learning and Deep Learning methodologies, have led to an explosion in breakthrough applications in many fields, ranging from image recognition, natural language processing, as well as autonomous driving and competitive strategy games. On the other one hand, many industrial national initiatives, such as Industry 4.0 or Smart Manufacturing, call for an urgent push of digitalization of industry and for industrial implementation of successful and meaningful data analytics and predictive models. Moreover, thanks to new emerging technologies, like Cyber Physical Systems and Internet of Things (IoT), that enable the accumulation of an ever increasing, and huge, amount of industrial data, and the possibility of real-time simulations of complex systems, more sophisticated decision-making approaches, enhanced by Intelligent Algorithms, can be conceived and developed.

Statistical-based methodologies have been extensively applied for data analysis and intelligent approaches for operation management, ranging from classical demand forecasting to complex production planning and control system, and in general as support to decision-making. These methodologies offer good results and are now consolidated, but they are not always adequate for a rapid, dynamic and automatic analysis of the amount of data which are made available by the adoption of innovative information technologies that are emerging in the industrial field, such as those that are mentioned above. Therefore, due to the recent advances of Machine Learning, mainly in the field of Deep Learning, arises the need to verify whether these methodologies offer comparable or superior performance relatively to the classic ones of purely statistical nature. Indeed, despite the fact that the use of Machine Learning methodologies is growing rapidly, and the results obtained are improving significantly year after year, there are still few industrial applications. The present thesis is placed within this context, and it is aimed at shedding light on this research question. Indeed, as later discussed in Chapter 2.2, Machine Learning applications can prove to be useful within diverse manufacturing domains: maintenance management, quality management, demand forecasting, and decision-support systems in general. Among them, this thesis focuses on proposing new methodologies to tackle three main manufacturing and industrial tasks: *i)* a

bankruptcy prediction problem*, ii)* a combined two-stage optimization technique for production planning and control technique, and *iii)* an industrial case-study for an innovative demand forecasting framework in the fashion market. All three are highly relevant and challenging tasks in the field of industrial engineering. To be successfully addressed, the ability to discover and harness patterns in the data and between different entities are needed. Machine Learning and Deep Learning, with their knowledge discovery and representation learning abilities, can prove to be effective tools to advance old application and discover new ones.

1.2. Thesis Structure

This thesis is organized into five chapters. After the present preliminary chapter, an introductory chapter about Machine Learning theory and a review of applications in manufacturing is presented. After, each chapter describe in detail novel applications of Machine Learning and Deep Learning algorithm to industrial problems. The structure and aim of the remaining chapters are as follows:

Chapter 2 – Background Research presents a quick overview of the Machine Learning theory, both in general and describing more in details the most frequently used techniques in this thesis, namely Neural Networks, Random Forests, and XGBoost. Moreover, a thorough review of Machine Learning applications in manufacturing is presented.

Chapter 3 – Machine Learning models for bankruptcy prediction in Italy presents the results of the comparison between traditional statistical learning methodologies, such as Logistic Regression, with more advanced and recent Machine Learning and Deep Learning algorithms, such as XGBoost, for the prediction of bankruptcy of Italian firms using balance sheets data made available by Bureau Van Dijk. The work studies the inclusion of innovative industrial variables to obtain better performances and shed new light on the interpretability of such predictive models. Moreover, a sound methodological approach to cope with the high imbalance ratio of the task at hand is described.

Chapter 4 – WLC enhancements through Deep Learning and Statistical Optimization introduces the key concepts necessary to understand the content of the chapter itself, with a conceptual introduction to the WLC theory. This is followed by the explanation of two different approaches that are combined to produce a state-of-the-art optimization framework for WLC: i) the first will be about the application of Deep Learning Neural Network for the forecast of lead-times in a shop-floor managed using the WLC framework, with the aim of reducing the tardiness of the system itself. The simulation model provides the possibility to contract Due Dates with the customer at the time of order release. ii) the second approach presents the application of Design of Experiments for statistical optimization of norms' values in the presence of shifting bottlenecks.

Chapter 5 – Demand Forecasting Framework for the Fashion Industry: a case study presents the proposed framework created to tackle the difficult task of predicting products' quantity sold for fashion garments. The probabilistic modeling was hindered by many practical limitations, as well as the impossibility of applying traditional time-series approaches. The framework proposed leverages a novel two-stage algorithm based on many cutting-edge Deep Learning methodologies, all combined to provide both the forecasting model as well as business intelligence by-products.

2. Background Research

2.1. Introduction to Machine Learning

2.1.1. Definition

A single definition of Machine Learning (ML) cannot be properly formulated, as this term encompasses a multitude of different approaches taken from the field of computer science and of multivariate statistics. Nonetheless, a good definition can be found in Murphy (Murphy, 2012) who defines ML as the "set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty". Although very clear, this definition gives too much emphasis on pattern recognition and decision-making that, as important as they may be, do not cover the whole spectrum of ML approaches and methodologies. So, in more general terms we could define ML as a set of methodologies and algorithms capable of extracting knowledge from data, and to continuously improve their capabilities, by learning from the data accumulating over time. Please note that, in this context, learning, as defined in Simon (Simon, 1983) "denotes changes in the system that is adaptive, in the sense that they enable the system to do the same task or tasks drawn from the same population more effectively the next time". ML models are characterized by the ability to self-adapt (at least to some extent), to changes in the data and/or in the environment, and to readjust their output accordingly. This pivotal element explains the recent an increasing interest in these disciplines, as it perfectly matches the need to process and to analyze the "Big Data" generated by a widespread use of electronic devices, web searches, social media and social media marketing.

ML is commonly divided into three areas: (i) Supervised Learning, (ii) Unsupervised Learning, and (iii) Reinforcement Learning [1]

2.1.2. Supervised Learning

Supervised Learning (SL), also called predictive learning, aims to learn a mapping f from inputs x to outputs y, using information contained in a dataset of training examples. More precisely, when the response variable is continuous, we talk about regression task, whereas if the response variable is categorical, we talk about classification task. Many

methods belong to the SL area and, among them, the best known are: Neural Networks, Support Vector Machines, Decision Trees (and their extensions, such as Random Forests and XGBoost), Logistic Regression and Naïve Bayes Classifiers. Anyhow, apart from their implementation differences, the common feature of all SL models is the requirement of a training and of a test set. Both sets are collected from the direct observation of the system of interest (for instance performing an experiment) and, for each observed example, the true value of the response variable y is measured and registered together with the known values of the inputs $\vec{x}$. By operating in this way, a dataset of examples is built, and it is used to train a learning algorithm, which aims to learn a good approximation $\hat{f}$ of the true relationship f among inputs and outputs, by minimizing a predefined cost (or loss) function. To make an example of classification, we could consider a dataset containing the physical properties (prediction variables $\vec{x}$) and the quality compliance level (response variables y), expressed on a categorical scale, of a batch of items manufactured in an industrial plant. In this scenario, the goal could be that to generate a classification model to predict the quality compliance level of new batches. Concerning regression, an industrial example could be the prediction of a certain physical property (measured on a continuous scale), such as the thickness or the surface roughness of items processed by a mechanical workstation. In this case, the task could be traced back to an image recognition problem, that is: images of the manufactured items are taken (presumably before and after the machining process) and images are converted into a vector of features to generate the prediction variables x.

2.1.3. Unsupervised Learning

Unsupervised Learning (UL) is concerned with datasets where no ground truth (i.e., labels) is available, and the goal is to detect and to extract patterns in the data, whose nature or even whose existence could be partially or completely unknow. For the aforementioned reasons, UL is sometimes referred as descriptive learning and it is associated with knowledge discovery techniques. Broadly speaking, UL could be divided into three main tasks: *i)* clustering, *ii)* density estimation and *iii)* dimensionality reduction. A classic example of clustering is the marketing-driven need to find sub-groups of customers, that are similar in terms of purchase behavior. In these cases, when information about class membership is unknown, well known algorithms, such as Hierarchical Clustering or K-Means, can be

effectively used. In other occasions, think for example to a “Big-Data environment”, it may be useful to pre-process data to reduce their initial dimensionality. Principal Component Analysis is the classical way to perform this task, but also neural networks topologies (such as Autoencoders) can be employed, with the goal of learning the best compressed representation of the original data. In a broader sense, we could say that all Deep Learning models can be thought as a way to capture both the hidden representation of the data and the most relevant relationships among them. From this perspective, Deep Learning is generally referred as Representational Learning (Goodfellow).

2.1.4. Reinforcement Learning

Reinforcement Learning (RL) differentiates from the other approaches, as it implements a computational approach to learn from interaction [2]: instead of seeking a map from input space to output space RL tries to learn a map from situations (environment state) to actions. Like in a trial-and-error approach, rather than starting with a pre-existing dataset (labeled or not), an agent learns “by doing”. More precisely, the agent is free to interact with the environment, by performing a predefined set of actions. Each action modifies the system’s state and such modification is quantified through a specific reward signal. Since the objective of the agent is that to earn as much as possible, from the reward signal, the agent will learn, by doing, the best reaction to each possible external scenario. In this regard it is worth noting that, the learning process is not purely governed by the reward signal, but it can be also supported by a superset of supervised and/or unsupervised algorithms, which should optimize the exploration and the exploitation of the action space of the agent. If part of the above-mentioned superset of algorithms are implemented as neural networks, we talk of Deep Reinforcement Learning. Anyhow, regardless of the implementation details, the final goal of a RL algorithm is to produce an artificial agent (or multiple agents interacting with each other) with the ability to make decisions based on current environmental conditions and on past experiences. From an industrial perspective, RL agents could be used to implement ordering strategies in complex environments (such as supply chain networks) or to steer production parameters in realistic simulated scenarios with the aim of costs’ minimization.

2.1.5. In-Depth explanation of main Machine Learning techniques

2.1.5.1. Multi-Layer Perceptron

The feedforward neural network is the simplest type of artificial neural network. This network is based on computational units called neurons, that are grouped into layers and that are interconnected in a feed-forward way. In other words, neuron in one layer has directed connections to the neurons of the subsequent layer but not with the neurons of the same layer. Also, the signal moves only forward, from the input nodes, through the hidden nodes (if any) and to the output node(s).

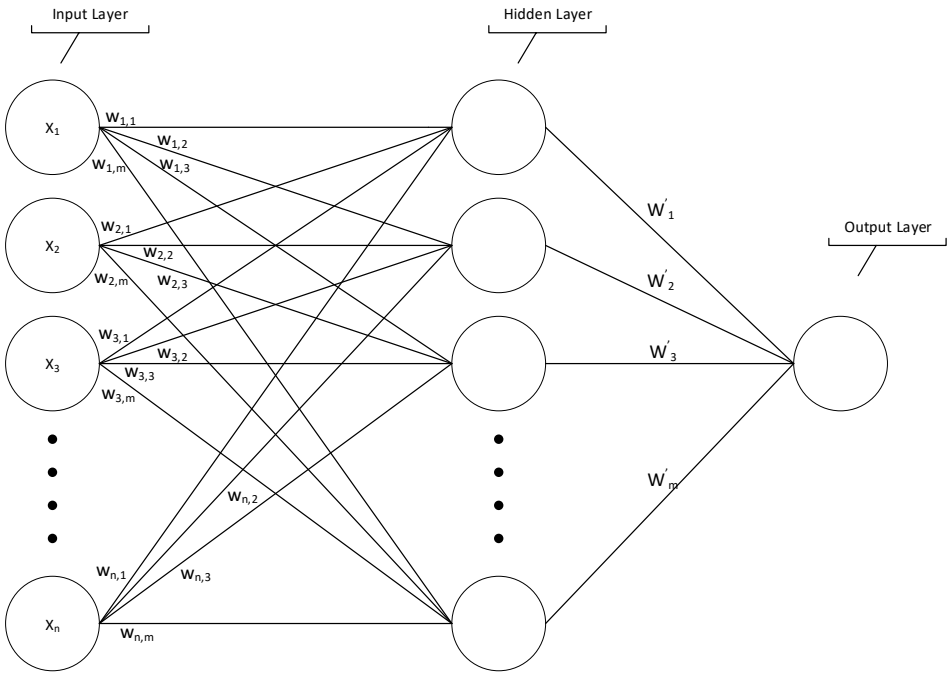


Figure 1 - Minimal Neural Network topology

As can be seen in Figure 1, the first layer, the input one, associates one neuron to each covariate which must be processed. These values are then processed by the second layer, known as the hidden layer. Briefly the functioning is as follow: each neuron *j* in the hidden layer is connected with weighted links w_{ij} to all preceding neurons *i*. Weights are used by neuron *j* to compute a weighted sum of the input covariates. This weighted sum is then passed through a non-linear activation function, to produce the final output of the neuron, which will

be passed forward to the output layer. The choice of the activation function is up to the analyst, but common choices are the hyperbolic tangent or the rectified linear unit (Relu [3]).

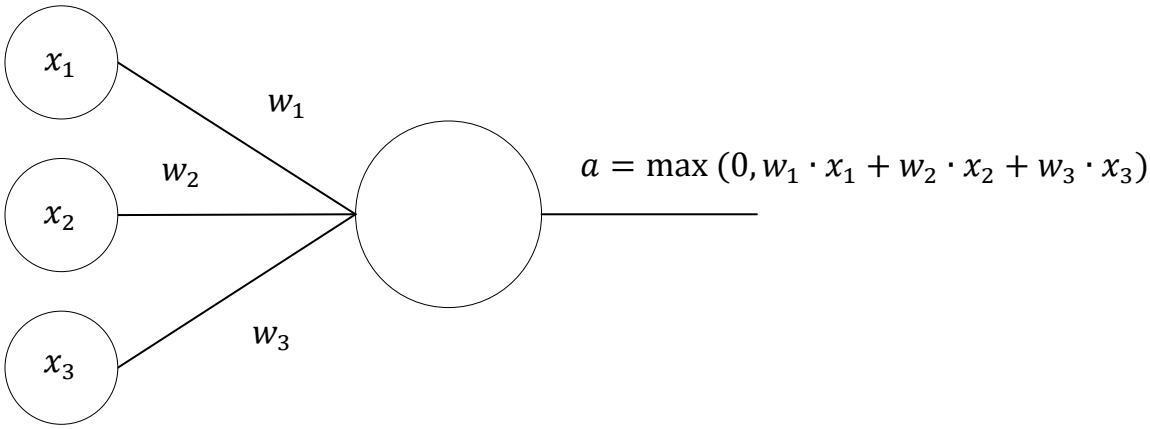


Figure 2 - Neuron computation

The neuron computation here described is depicted in Figure 2, in which the Relu function is used. In this layer there are as many neurons *k* as the number of response variables (a single neurons in case of binary classification), fully connected with all preceding neurons with weighted links w_{ik}. As before, a weighted sum is made by each neuron *k*. In case of regression, this is the final output of the neuron, while in case of binary classification a sigmoid function is used to constrain the output in the range [0,1]. All these weights are learned during the training phase, characterized by a forward and a backward propagation phase. In the first one each observation of the training set is fed into the neural network and the resulting output is collected. In the backward step the error between the collected output and the ground truth (i.e. the true value of the response) is computed and the weights are adjusted to minimize a global loss function, using a gradient descent optimization method. The most common one is the backpropagation algorithm [4]. The backpropagation algorithm essentially computes, using the chain-rule of derivatives, the partial derivative of the loss function with respect to all weights. This value, coupled with a learning rate of choice, is used to iteratively adjust the weights in a gradient descent fashion. Weights are updated after a batch of *k* observations has been processed: if the batch size *k* equals the number of observation *n* the procedure is the classic gradient descent, while if $k<n$ the procedure is called stochastic gradient descent. To reach convergence, the training is repeated a certain number of times, known as epochs.

2.1.5.2. Convolutional Neural Network

Convolutional Neural Networks, also known as Convolutional Networks or ConvNets (CNN for short), are a specialized kind of neural network designed to process information with a geometric structure which is known a priori, i.e. grid-like topology. Examples of such data includes images, which are often represented as 2D grid structure of pixels, but also time-series data, which can be seen as a special case of 1D grid-like structure taking samples at regular time intervals. Originally introduced by [5] and later improve by [6], the name itself implies that the networks applies a mathematical operation called convolution, which in this case it is a specific variant of the more general mathematical operator as defined with the same name, in mathematics. As stated by [7], *convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers*.

The convolution operator in neural networks takes as input a grid-like structure, typically an image, and sequentially applies a filter (kernel) over the whole input, to produce as output what is usually referred to as feature map. Usually, the input is a multidimensional array of data (i.e. an image is usually a 3D tensor, with a matrix whose components specify the color intensity for each pixel, for each channel, Red, Green, and Blue) and the kernel is usually a two-dimensional array of parameters that can be optimized (learned) by the learning algorithm.

Convolution possesses three important properties that introduces an inductive bias that can help when applied to a dataset which shows the aforementioned grid-structure. The first property is *sparse interaction*, meaning that, usually, the kernel has a smaller dimensionality than the input. This leverages the fact that, while the input could be very high-dimensional, only a small subset of the features carries important information (for example, edges in an image). This enables the network to store fewer parameters, improving efficiency while preserving performances. Also, the computational complexity is highly reduced. The second property is *parameter sharing*, which further reduce the storage requirements of the model. Finally, the last property is *equivariance* to translation, which means that the model is able to recognize the same patterns no matters where it is located within the input grid. However, other mechanisms are needed for handling other transformations, such as changes in scale or rotation of an image.

2.1.5.3. Recurrent Neural Networks

The Recurrent Neural Network (RNN) topology arises to address those problems in which the input data is arranged into a variable-length sequence, such as Speech recognition or Time-series prediction problems. Recurrent neural networks are artificial neural networks where connections between units can form cycles. An RNN is a neural network which models a discrete-time dynamical system with input x_t, output y_t, and a hidden state h_t, where t represents time. The relationships between these quantities are as follows:

$$h_t = f_h(x_t, h_{t-1}) \quad (1)$$

$$y_t = f_o(h_t) \quad (2)$$

In which f_h and f_o represent state transition and output function, respectively, which are parametrized by parameters ϑ_h and ϑ_o. In particular, in a conventional RNN, the following applies:

$$h_t = \varphi_h(W^t h_{t-1} + U^T x_t) \quad (3)$$

$$y_t = \varphi_o(V^T h_t) \quad (4)$$

where W, U, and V are the transition, input, and output matrices respectively; φ_o is usually the hyperbolic tangent function.

As it can be seen, at each time-step t the output of the RNN depends both on the input at that time-step and on the hidden state at the previous time-step. Indeed, the intermediate state can store information about past inputs for a time not fixed a priori. Undeniably, the main difference with a traditional Multilayer Perceptron is the ability to cope with time dependencies and to recognize temporal dependencies within the input sequence.

Moreover, RNNs show a similar benefit as shown by the CNN architecture, in which efficiency is gained by sharing weights in time. Indeed, the matrices W, U, and V are not time dependent, as they propagate unmodified through each time step of the sequence processing. Indeed, the traditional training algorithm must be effectively modified to account for the temporal unrolling of the RNN module. The gradient-based technique which is able to cope with this dynamic is called the Backpropagation-Through-Time (BPTT) [8].

One of the major issues that may arise during the training of RNNs is the vanishing gradient problem. First introduced by [9], the problem particularly arises during the processing of long sequences, where temporal dependencies may span many time steps. In this scenario, the error signal gradually diminishes as it is propagated back, until eventually fading out completely. Thus, long-term dependencies cannot be effectively learned because of insufficient weight updates. In order to overcome effectively such issue, more complex recurrent models have been introduced, such as the Long-short Term Memory (LSTM) [10] and the Gated Recurrent Unit (GRU) [11].

2.1.5.4. Random Forest

Random Forest [12] is an ensemble learning method based on Decision Trees. Decision trees can be applied to both regression and classification problems and involve segmenting the covariate space into a number of non-overlapping regions (partitions) using simple rules. The set of splitting rules used to partition the input space can be summarized in a flow-chart structure which can be represented using binary trees.

The building process of a decision tree follows a top-down greedy approach known as *recursive binary splitting*, meaning that at each iteration the best split of the predictor space is found relatively to the partitions already made. In order to find the best split, each predictor is selected, and for each predictor the best cut-point *s* is found such that it optimizes a given metric that measures the progress of the learning of the tree. For regression tasks the metric is usually the residual sum of squares (RSS), while for classification problems either the Gini index or the entropy index can be used. The Gini index G and the entropy D are calculated as follows:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{5}$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log\, \hat{p}_{mk} \tag{6}$$

In which $\hat{p}_{mk}$represents the proportion of training observation in the *m*-th region that are from the *k*-th class. Once the best predictor and its corresponding best cut-point is found,

the process carries on until a stopping criterion is reached, for example until no region contains more than a given number of observation, or until a maximum number of region is reached, or until the optimization metrics does not improve more than a given threshold. Additional techniques such as tree pruning can be used to reduce the tree complexity, reducing possible overfitting.

While decision trees can be simple and useful for interpretation, they are typically not competitive with other supervised techniques. Most common issues are overfitting and high model variance. Random forest prevents these shortcomings constructing a multitude of decision trees and combining each tree output in a final response.

The forest is composed by a given number of trees trained as follows: 1) Draw B bootstrap samples from the original data, 2) Grow a tree for each bootstrap sample. At each node, select at random m out of p covariates where $m \in \{1,...,p\}$ is a parameter chosen by the analyst at the start (usually m=sqrt(p)). The splitting can be stopped when a minimum node size is reached or using different stopping criteria such as maximum depth [13]. The methodology in training has two main characteristics. The first is that the trees are fed with different versions of the dataset obtained through a bootstrap sampling. The second is that, at each node, in order to identify the best predictor to use for splitting, the tree can choose from a reduced random subset of m predictors. The first characteristic is shared with the bagging algorithm, while the second is introduced with the aim of decorrelating the trees' outputs. The RF method can be seen as a refinement of the bagging algorithm introduced by [14].

During the prediction phase, for a given test observation, the output of each decision tree in the ensemble is recorded. The final output is obtained by averaging the different trees outputs, when the predictive variable is continuous, or computing the majority vote, in case of discrete predictive variable (such as our bankruptcy prediction task).

2.1.5.5. XGBoost

XGBoost is used for supervised learning problems, both for regression and classification tasks. XGBoost is a particularly efficient implementation [15] of Gradient Tree Boosting [16]. The Gradient Tree Boosting algorithm tries to optimize a cost function over a function space by sequentially choosing a CART tree, in order to minimize a given objective function.

Generally, given a dataset of n examples, the prediction for the i-th example is calculated as follows:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(\boldsymbol{x}_i), \quad f_k \in \mathcal{F} \tag{7}$$

where K additive functions (trees) are used sequentially and where $\mathcal{F}$ is the function space of all possible decision tree.

A decision rule q_k is associated to each decision tree f_k. Moreover, a leaf scores w_{kj} is associated to each *j-th* leaf of the *k-th* tree. Specifically, the decision rule q_k maps an example $\boldsymbol{x}_i$ to a specific leaf of the decision tree to which corresponds a score w_{kj}. All scores are then summed up to calculate the final prediction y_i of example x_i.

To choose the set of trees f_k from $\mathcal{F}$ one has to minimize the following objective function:

$$\mathcal{L} = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \tag{8}$$

in which l is a differentiable loss function, such as MSE for regression and logistic error for classification, and $\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2$ is a regularization term (T is the number of leaf in a given tree) which penalizes the complexity of the model, to avoid overfitting.

Since $\hat{y}_i$ is a sum of non-parametric functions (trees), parameters of equation (8) are functions themselves; thus, it cannot be optimized using traditional optimization methods in Euclidean space. The model is thus trained, step-by-step, in an additive fashion. At generic

step *t* a tree f_k which optimize equation (8) is added the ensemble created so far. Moreover, using a second-order approximation, equation (8) can be simplified yielding the following objective at step *t*:

$$\tilde{\mathcal{L}}_t = \sum_i [g_i f_t(\boldsymbol{x}_i) + \frac{1}{2} h_i f_t^2(\boldsymbol{x}_i)] + \Omega(f_t) \tag{9}$$

where g_i and h_i are the first and second order gradient statistics of the loss function, w.r.t y_i^{t-1}, defined for each example. For example, in case of square loss (i.e. $l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$), the first order gradient statistics is proportional to the residual (i.e. $g_i = 2(\hat{y}_i^{t-1} - y_i)$) and the second order gradient statistics is fixed at 2 (i.e. $h_i = 2$).

If the tree structure q_k of tree f_k was known a priori, it can be proved to optimize $\tilde{\mathcal{L}}_t$, the optimal weight in each leaf and the resulting objective value are as follows:

$$\tilde{\mathcal{L}}_t(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \tag{10}$$

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{11}$$

in which the gradient statistics are summed over all example belonging to the *j*-th leaf.

However, the best tree structure cannot be known a priori, being infeasible to enumerate all possible trees structures. So, the tree is built using another greedy approach. Specifically, the tree is built as a traditional CART, in which at each branch the best split is chosen, using equation (10) rather than the Gini index (or entropy index). Indeed, formula (10) is a generalization for a wider range of objective functions of the impurity score of decision trees.

Informally, at each stage of the sequential procedure, the gradient boosting algorithm tries to improve over the preceding imperfect model, by constructing a new estimator to add to the ensemble. In this way a better overall model is obtained. In case of regression tasks, the new estimator is fitted to the residual (error) of the preceding imperfect model, to correct previous mistakes.

2.2. Machine Learning Applications in Manufacturing

2.2.1. Introduction

In the new global economy, competition fosters complexity, which directly affects manufacturing environments, products, processes, company and supply chain dynamics. Now that we are entering into the Industry 4.0 era [17], the new managerial paradigm is shifting from the need of low variability, through products' commonalities and processes' repeatability (as advocated in the Lean Thinking theory), toward the so-called Mass-Customization where, conversely, wide-markets goods should be rapidly modified and re-manufactured, at low cost, to satisfy a specific customer's need. In this scenario, resilience, reconfigurability and flexibility are key issues of competitiveness, as clearly expressed by the Smart Manufacturing concept, indicating a company that has the potential to fundamentally change how products are designed, manufactured, supplied, used, remanufactured and eventually retired. Information Technology, sensor networks, computerized controls, production management software and, more in general, Industrial Internet of Things are basic prerequisites for a company to be smart. Yet, alone these devices are not enough, and a manufacturing system cannot be considered Smart, unless its overall functioning is regulated by intelligent control technologies, for a quick, accurate and reliable response to internal and external events [18]. Furthermore, as noted by [19], smart manufacturing must embrace big data and, to this aim, information system and production management software must be coupled/enriched with deep analytical skills [20] and with learning ability [21], to ensure competitiveness and effectiveness.

Evidences also suggest that data are one of the most valuable assets of a firm and, especially for innovative companies, Big Data management is a key issue [22]. Non only a proper data management may help in differentiating from competitors and gaining competitive advantage, but companies that use data-driven decision-making approaches have proven to easily outperform their competitors, being, on average, 5% more productive and 6% more profitable [23]. Unfortunately, while in many cases companies perceive the utility of their data, often they do not have the knowledge needed to exploit their data-silos and lack the clear understanding of what is important to be measured; as a result, the informative content of the data is missed, and real and valuable knowledge get lost [22]. Large amounts

of data can be more harmful than a lack of data if the analyst does not know how to choose the truly meaningful data. We could say that "quality trumps quantity", an issue that calls for an overall improvement of data collection, usage and sharing [19]. In this regard, Machine Learning (ML), a branch of Artificial Intelligence (AI) concerned with algorithms capable to learn directly from data, could provide the missing link between the need and the industrial implementation of successful and meaningful data analytics and predictive models. Not surprisingly, many works indicate ML as one the principal enablers to evolve a traditional manufacturing system toward an Industry 4.0 level. After the publication of the report by Pham and Afify [24], there has been a spike of academical interest in ML, and researchers started to consider ML applications also within industrial fields such as: pattern and image recognition, natural language processing, operations optimization, data mining and knowledge discovery. Since then, the number of papers published on this topic has ever increased and the trend has been recently fueled by many government initiatives, like Industry 4.0 (Germany), Smart Factory (South Korea), and Smart Manufacturing (USA), calling for a radical change in the manufacturing paradigm, based on processes' augmentation and enhancements due to Information Technologies. Especially in the last decade, the state of the art of ML techniques has made a huge step forward, as demonstrated by the algorithms used by autonomous driving cars or by electronic strategy games. Both tasks were considered many years away from a practical solutions, yet autonomous driving cars are already being tested in real urban environments, and AlphaGo has overwhelmed the world champion of the Go game [25]. Similarly, DeepMind [26] recently reported the development of an AI that successfully learned to play better than humans to many other strategy games. Furthermore, the enabling technologies (i.e., sensors, open source software, public datasets, computational power, cloud services, etc.) are now mature and available at low cost and government initiatives offer interest-free (or even non-refundable) loan and/or fiscal incentives to support investments in IT projects.

Owing to these favorable issues, the time seems to be right to implement ML in the industry, yet practical examples are still rare, and the industrial use of these technologies is still confined among an exiguous number of international companies. Presumably, a detrimental element of acceptance can be found in the widespread concern that AI could jeopardize many jobs, increasing the unemployment phenomenon. In our opinion this is a misconception: if on the one hand it is true that automation will replace human labor, on the

other one hand replacement will concern redundant and repetitive tasks. As a matter of fact, having the ability of learning representations autonomously, Machine and especially Deep Learning (DL) models can extract knowledge directly from raw data, freeing researchers from the expensive and time-consuming step of feature extraction and feature engineering [27]. Thus, it is not daredevil to assume that the most successful implementations will be those ones augmenting and assisting human decision making, freeing people from low value-added tasks. Apart from that, one of the main barriers for a pervasive industrial adoption, is the lack of understanding of these methodologies and the inability to envision how they could be exploited in and industrial context to improve business' performances [28]. This issue is even more critical, if we consider the vast number of algorithms (and possible variations) that have been proposed in technical literature. Lacking a repository of best use-cases (for each industry and organization) how to select and fine tuning the right methodology to be used becomes challenging, especially for industrial practitioners.

The present paper deals with these topics, and aims to clarify the real potentialities, as well as potential flaws, of ML algorithms applied in the field of operation management. To this aim, a comprehensive review is presented and organized in a way that should facilitate the orientation of practitioners in this field. Specifically, papers from 2000 to date will be reviewed and categorized in terms of applied algorithm and application field. Insights, concerning trends and evolutions in the subject matter will be provided, and possible future developments will be investigated as well.

Indeed, we believe that there is an important opportunity to implement ML approaches to many industrial fields where these techniques are still shallowly explored. For instance, recent developments such as Deep Reinforcement Learning, with outstanding results in seemingly unrelated applications, could help obtain concrete improvements over state-of-the-art results in traditional industrial problems such as scheduling and inventory management. Yet, to do that, a clear understanding of the main ML's models and the awareness of what ML can and cannot do is fundamental. As posed by the "No Free Lunch Theorem" [29], ML cannot solve all industrial problems and its practical adoption (as an alternative to more mature technologies) must be carefully evaluated and pondered. It is important to make an informed decision, without being influenced by the trend and the fashion of the moment, and by an overwhelming expectation generated by sensationalistic journalism. At the present, in fact, according to the Gartner Hype Cycle for Artificial Intelligence, ML and especially DL

have reached the peak of inflated expectation; it is thus probable that a disillusion phase will probably follow soon, whereas many industrial applications, such as Smart Robotics, or more general Intelligent Applications, are still in the first phase of Innovation Trigger. Indeed, also Reinforcement Learning is listed in this phase. In these cases, plateau of productivity is predicted to be reached in 5 to 10 years.

2.2.2. Searching Methodology

The ability to choose an algorithm (or a subset of algorithms), suitable for a specific task or problem, is a core competence for researchers and/or practitioners who want to apply ML to industrial cases. This choice can make the difference between failure and success, as the adopted algorithm has a significant impact on the obtained performances. The choice should be based on data availability and on the type of task to be performed, and it should also be supported by past experience and/or on previous studies of similar nature. In case of ML, the number of alternative algorithms is extremely vast and variegated, and the same problem can be tackled with different approaches, differentiating in terms of operating characteristics and of complexity. Given the novelty of ML (relatively to an audience of industrial practitioners), this variety can be disorienting and misleading, at least in a first phase. Given these issues we believe that a systematic literature review, focused on the historical developments of ML for Industrial Applications, may be extremely useful to highlight present and future trends and, above all, to help practitioners orienting in the field and coping with the subject matter in a more structured way.

Owing to identify trends, potentialities and criticalities concerning the use of ML for operation management, the review focuses on the following Research questions (Rq):

- Rq1 - *Which are the most popular ML methodologies applied in manufacturing environment?*
- Rq2 - *Which are the main application domains (i.e., industrial processes) where ML has been successfully adopted?*
- Rq3 - *Is the trend stable or has it modified through time, starting from 2000?*
- Rq4 - *Are there any issues/criticalities in the use of ML algorithms for Industrial Applications?*

- Rq5 - *Which are the least studied domains and algorithms, which could benefit from renewed approaches?*
- Rq6 - *Are there any links (i.e., cross references, key words and citations) among published articles which could highlight interesting development patterns?*

To give an answer to the above-mentioned questions, the whole publications' domain was investigated following a specific search-protocol, based on four main steps: *i*) Initial query-based search on scientific database, *ii*) Search enlargement through cross reference analysis, *iii*) Relevance assessment through citation graph analysis, *iv*) Final screening based on abstract analysis.

At step one, we gathered as many publications as possible, leveraging on known and trustable scientific databases: on May 2019, a "key-words" based search was made on Scopus, Web of Science and Google Scholar. Specifically, to limit the analysis to the papers addressing the application of Machine and/or Reinforcement Learning for Operations, possibly with a straight focus on Industry 4.0, data were filtered using the following query, formulated using the syntax required by Scopus:

```
KEY (("manufacturing" OR {supply chain} OR {industry 4*})
        AND ({machine learning} OR {reinforcement learning}))
AND PUBYEAR ≥ 2000
AND DOCTYPE (Ar)
AND (LIMIT-TO (LANGUAGE, "English"))
```

With minor adjustments the same query was used to collect papers also from Web of Science. Conversely, papers retrieved from Google Scholar were manually filtered, due to the binding restriction, set by the search engine, to make searches only by title. Anyhow, the filter (either performed manually or automatically) returned papers with at least a keyword belonging to the Set A = {"manufacturing", "supply chain", "industry 4.0"} and a keyword to Set B = {"machine learning", "reinforcement learning"}, provided that all the following inclusion criteria were met:

- C1 - Studies must be either conference of journals peer-reviewed publications (i.e., other kind of scientific works, such as books, patents and PhD were not considered);

- C2 - Language must be English;
- C3 - Only recent studies, published starting from 2000, are considered.

Such an extensive search returned a total of 455 publications, 240 from Scopus, 161 from Web of Science and 54 from Google Scholar.

At step two, despite the high number of collected papers, to avoid the omission of relevant articles, we enlarged the search by considering the list of the citations found in the collected papers. Such list was automatically generated leveraging on the Scopus APIs (https://dev.elsevier.com/sc_apis.html), which allows retrieving all citations and their related metadata (i.e., keywords, abstract, authors, etc.). Also, to exclude papers unrelated with the stream of research herein considered, the citation list was programmatically filtered considering the above-mentioned inclusion criteria, but the constraints on the keywords were partially relaxed. Specifically, we accepted papers with at least one keywords belonging to set A or to set B. In this way we obtained a list of 707 filtered citations.

At step three, the 707 citations and the original set of 455 publications were joined and used as input to build a citation graph, that was made using Gephi©, a freeware software application. Briefly, in a citation graph, nodes correspond to papers and directed arcs (connecting nodes) correspond to the citations among papers. So, taking the number of citations as an indication of relevance, the more arcs enter a node, the greater the importance of the corresponding paper. Using this classification criterion, we decided to include in the original list all the nodes with more than three incoming arcs; by doing so the original list increased from 455 to 514 publications.

Lastly, at step four, to refine the selection, the abstract of all the 514 collected papers were read and filtered using three additional criteria:

- C4 - Only works with an informative abstract clearly stating the papers' contributions and industrial results are considered;
- C5 - Studies must be unique, copies (or very similar papers) are removed;
- C6 - Purely theoretical or conceptual studies were not considered. Specifically, to be included, studies should present industrial applications tested on experimental data or, at least, tested on accessible datasets (used as benchmark by the research community).

By operating in this way, mainly due to the application of criteria C4 and C6, 409 papers were considered of low operating value and so they were discarded. The final corpus of 105 papers, which will be analyzed in the next sections, is listed in table A.1 of the appendix section.

2.2.3. Systematic Review

2.2.3.1. Preliminary classification

All papers were carefully read and classified in terms of Machine Learning Areas (i.e., Supervised, Unsupervised and Reinforcement Learning) and of Application Domain, that is the industrial area (or process) to which ML has been applied to. Concerning the application domain, in view of the content of the articles found in the literature, we tried to define classes sufficiently detailed and distinct, avoiding at the same time classes with too many or too few papers. In light of this, a good compromise was reached considering the following classes:

1. *Maintenance Management (MM) includes 18 papers dealing with:*
   - Failure modes classification and prediction (6),
   - Condition monitoring and fault detection (9),
   - Downtime minimization and maintenance planning (3).
2. *Quality Management (QM) includes 37 papers dealing with:*
   - On-line quality control (6),
   - Defects detection and classification (22),
   - Image recognition for defect identification (8),
   - Life cycle management (1).
3. *Production Planning and Control (PPC) includes 35 papers dealing with:*
   - Performance prediction and maximization (13),
   - Job scheduling and dispatching (13),
   - Dynamic process control (9).
4. *Logistic & Supply Chain (LSC) includes 13 papers dealing with:*
   - Demand planning and forecasting (3),
   - Inventory management (4),
   - Supply chain modelling and coordination (6).

The above-mentioned classification is graphically displayed in Figure 3, where the distribution of the papers in terms of application domain and of machine learning area is clearly shown. Please note that, in the picture, an additional category (namely Engineering Design) has been added, due to two relevant papers in the field of technical design that could not be included in any of the other categories.

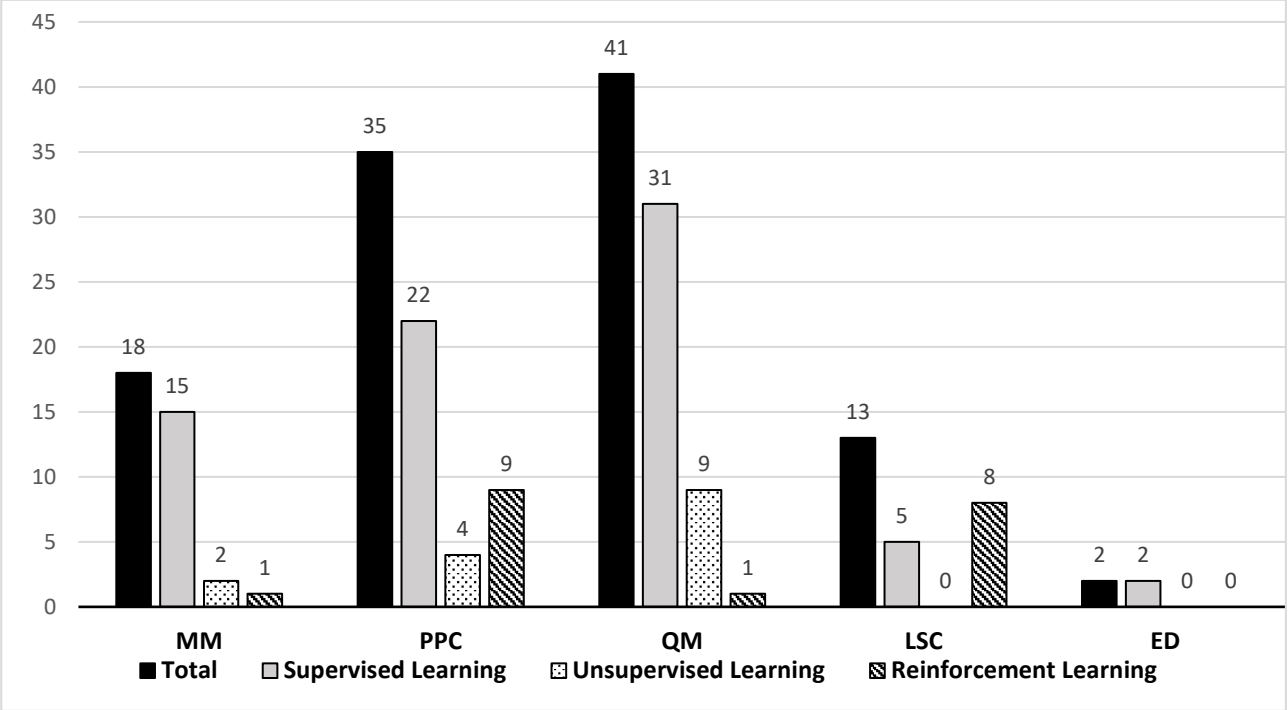


Figure 3 - Distribution of papers by application domain and of algorithm

The trend in number of publications, for each application domain, is shown in Figure 4 and, similarly, the evolution through time of the distribution of the publications for each ML area is shown in Figure 5. Note that in both figures, papers published in 2019 were not considered; since 2019 was not over yet, at the time of the analysis, publications were too few and they would have altered the real trend.

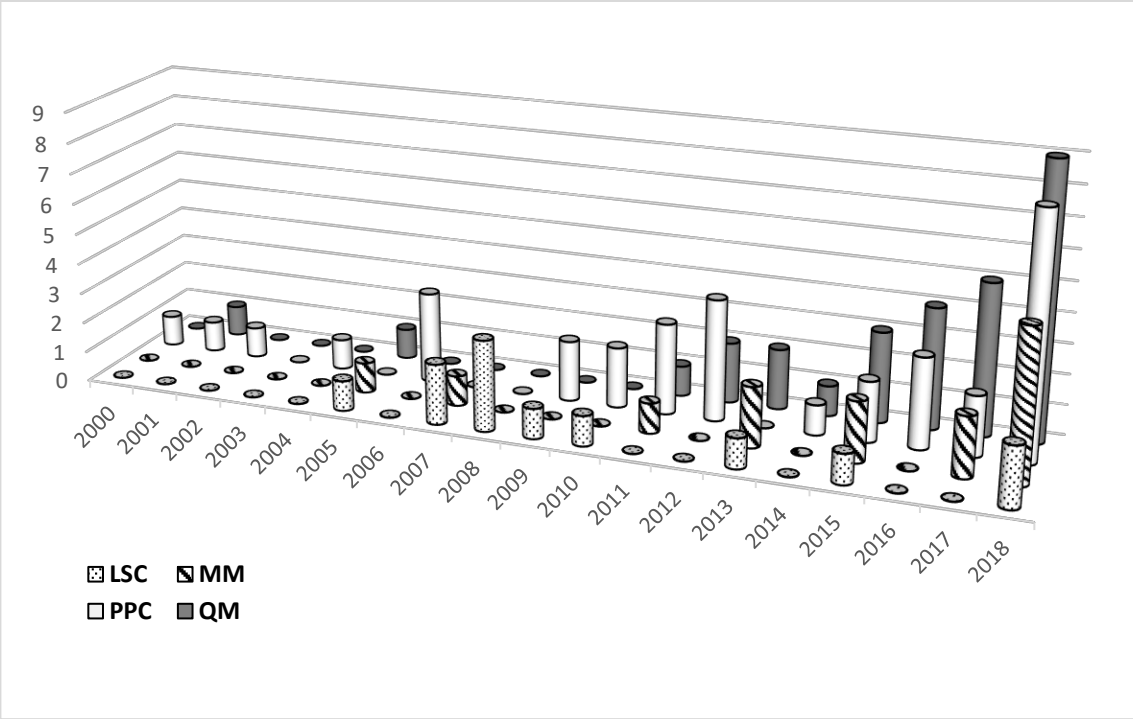


Figure 4 - Trend in number of publications, for each application domain

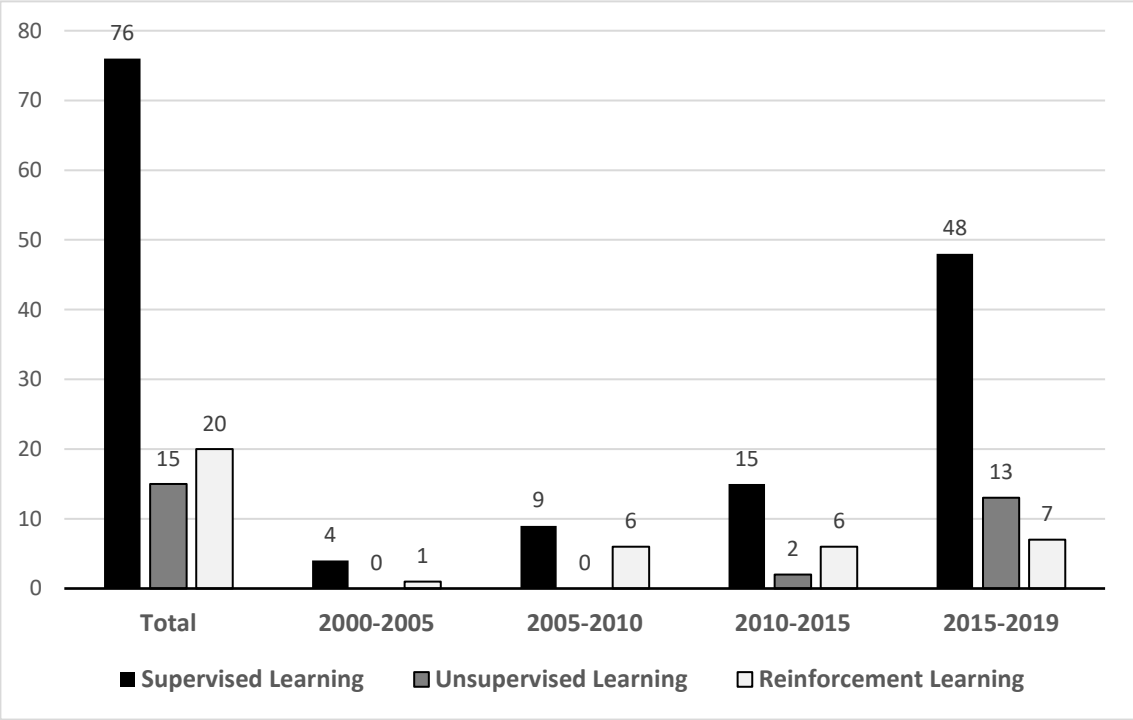


Figure 5 - Distribution of the publications for each ML area

It is evident that, in recent years, the industrial interest in ML has significantly grown, and the upward trend is clear, especially for supervised learning. This is not a surprising result, because when the researcher collects data from processes or experiments, he or she should

also be able to record ground-truth information at his or her disposal. If so, supervised learning algorithms are the ones that best exploit all available information and, for this reason, they have always been the most studied and applied ones. Nonetheless, as collecting and labeling data can be expensive and time consuming, also unsupervised method can be exploited, to successfully reach the desired goal. Indeed, as Figure 5 shows, an uplift trend of unsupervised learning in more recent years has occurred, too. Lastly, it is worth noting that, after a certain growth toward the end of the first decade of 2000, reinforcement learning has stabilized at a rather low level. Probably, notwithstanding the recent breaking developments in such area and the growing understanding of its potentialities, its higher complexity is slowing down its industrial application.

For the sake of completeness, Figure 6 focuses on Supervised Learning algorithms and shows the trend for three algorithms of this family, namely Neural Networks (NN), Support Vector Machines (SVM), and Tree-based techniques (Decision Trees, Random Forest, Gradient Boosting), which have shown to be the most used ones. As it is clear from the graph, SVM was the prominent technique until 2010, and although the use of this algorithm has not faded away, it has been overtaken by NN in more recent years. Albeit informally, the start of the Deep Learning era can be approximately placed around 2010-2012, and indeed in the last 7-8 years the use of Neural Network (especially of deep architectures), has been prominent.

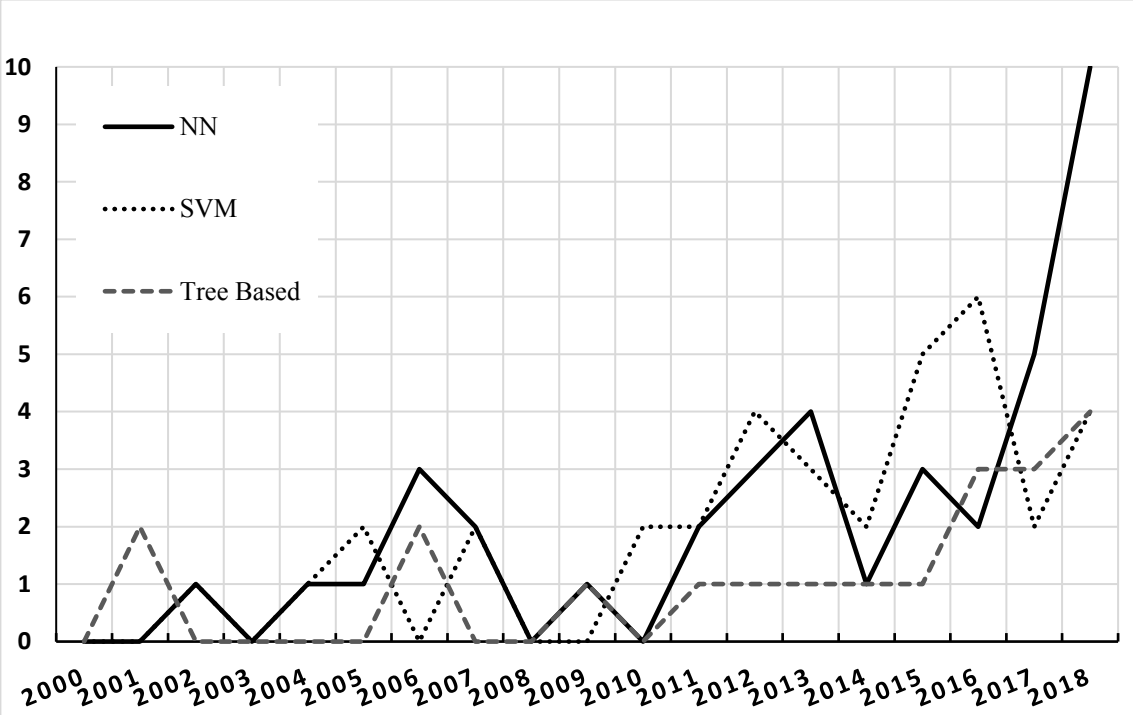


Figure 6 - Trend of main supervised algorithms

2.2.3.2. Keywords Analysis

From the corpus of the investigated papers, at total of around eighty different keywords was found. Getting rid of obvious keywords (e.g., Machine Learning, Supervised Learning, Unsupervised Learning, etc.), and by combining the remaining ones by synonyms, the total count of Figure 7 and Figure 8 relative to the used ML technique and to the application domain, was found.

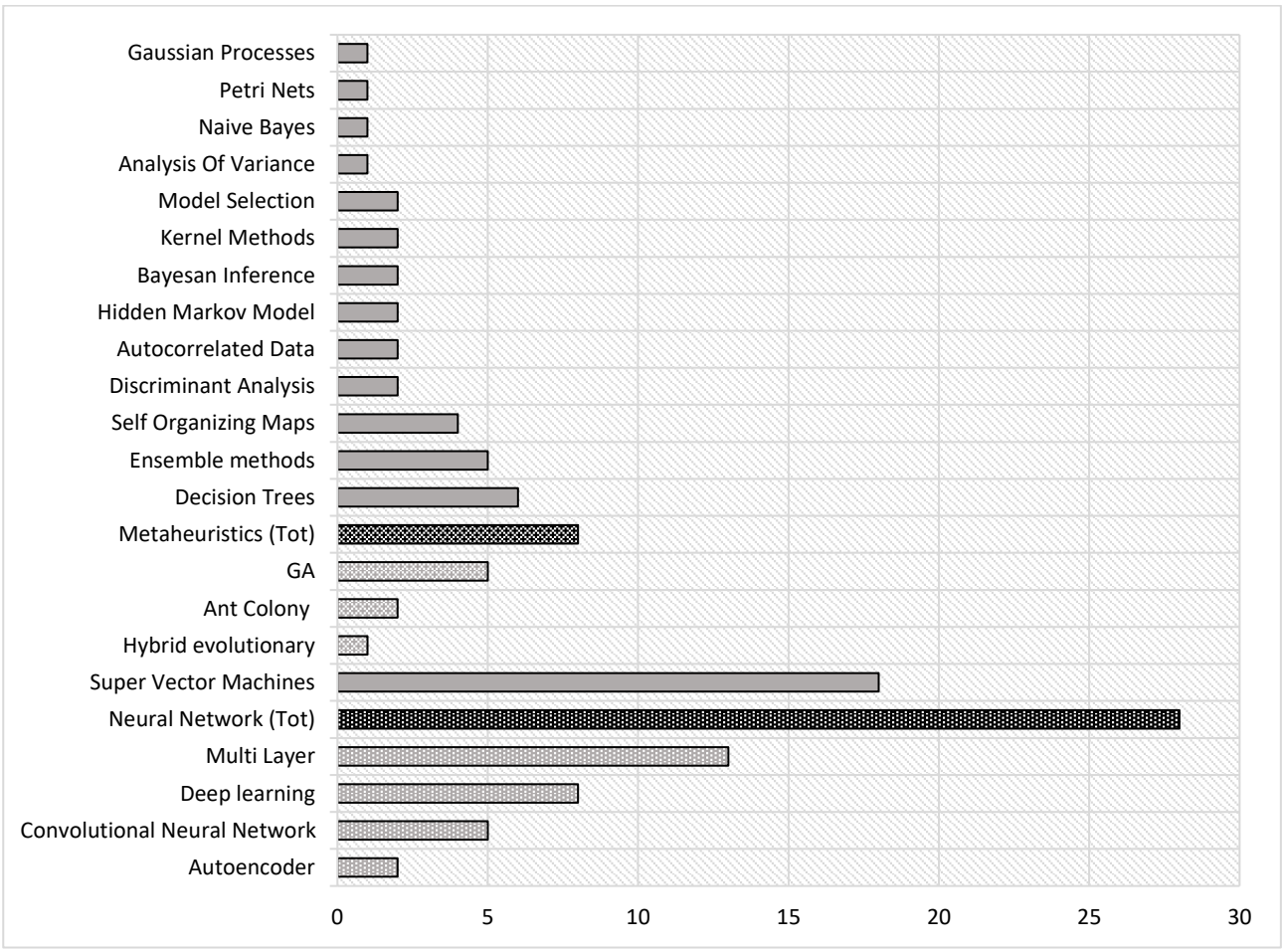


Figure 7 - Key Words relative to the adopted technique

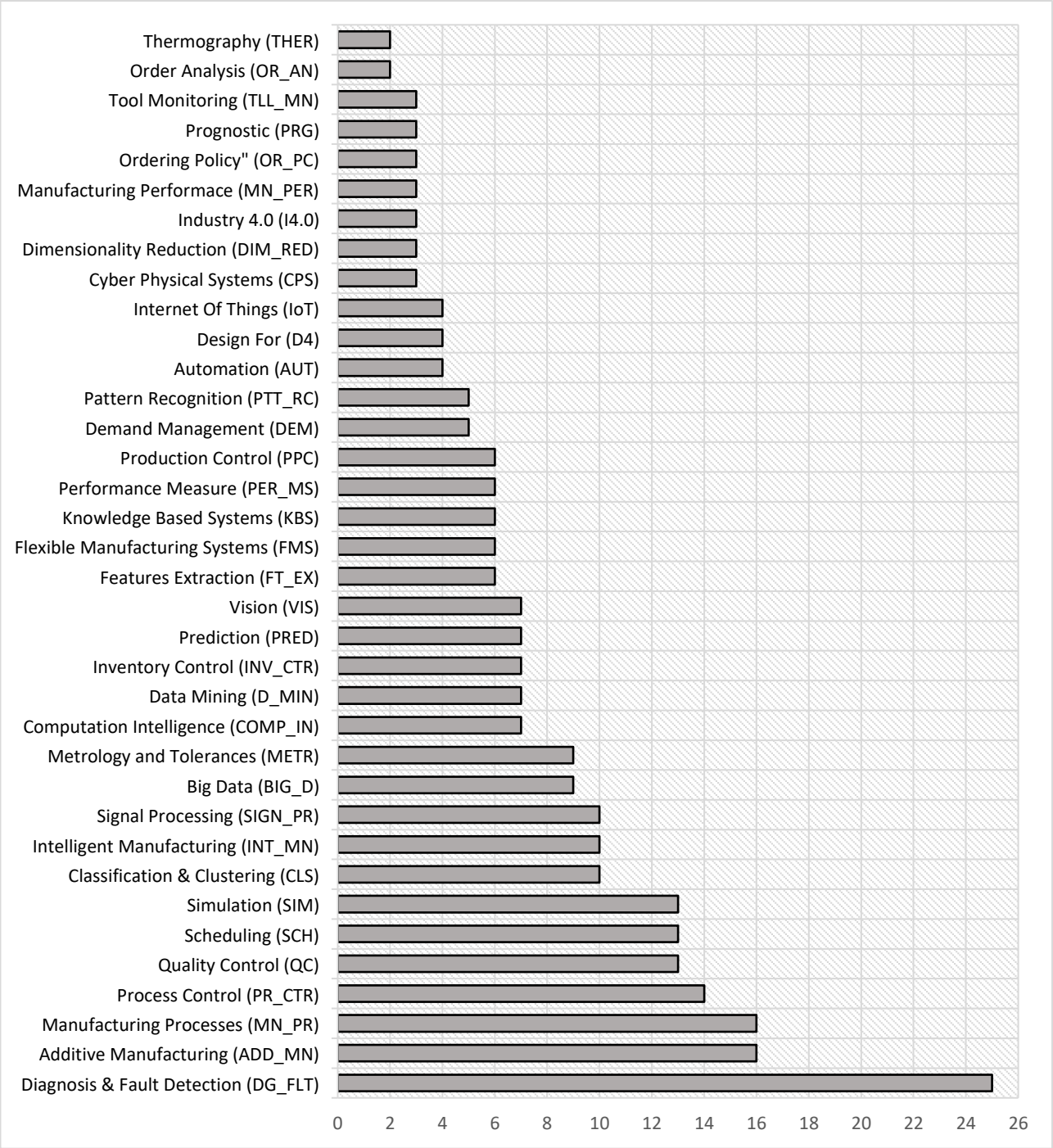


Figure 8 - Key Words relative to the solution domain

As it can be seen, in accordance with the results reported in the previous sections, “Neural Networks”, “Support Vector Machines” and “Genetic Algorithms” are, by far, the most adopted techniques. relatively to the industrial application domains, “Diagnosis and Fault Detection”, “Additive Manufacturing”, “Manufacturing Process”, “Process Control”, “Scheduling”, “Quality Control” and “Inventory Management” are the most frequent key words. In this regard, to get a better idea of the current trends of the research topics, we also

organized key words in a 3D bubble chart (Figure 9) were each key word *k*, identified with a triplet (*age, trend, size*) of data, is plotted as a sphere that expresses the *age* and the *trend* through its *xy* location and the *size* through its volume. Specifically, the above-mentioned data are defined as follows:

- *Size* (S_k) – The total number of occurrences of *k*,
- *Age* (A_k) – The number of years since the first occurrence of a *k*,
- *Trend* (T_k) – The percentage misalignment of the Center Of Gravity (COG) of *k*, as defined in eq. (12) and eq. (13):

$$T_k = \frac{\left(t_{COG,k} - (t_n - 0.5A_k)\right)}{A_k} = \frac{\left(t_{COG,k} - \bar{t}_k\right)}{A_k} \tag{12}$$

$$t_{COG,k} = \frac{\sum_{i=1}^{n}\left(s_{i,k} \cdot t_i\right)}{\sum_i s_{i,k}} = \frac{\sum_{i=1}^{n}\left(s_{i,k} \cdot t_i\right)}{S_k} \tag{13}$$

Where: t_n is the current year, $\bar{t}_k$ is the midpoint of the life of *k*, $s_{i,k}$ is the number of occurrences of *k* at year *i*, and $t_{COG,k}$ is the "temporal" coordinate of the COG of *k*.

Specifically, for a consolidated and stable keyword *k*, $t_{COG,k}$ should lay at the mid point of its life (i.e., $t_{COG,k} = \bar{t}_k$) and so T_k should be close to zero. Instead, a positive value of T_k indicate a keyword that is being used more and more frequently, or that has come back into vogue, after a period of latency. Conversely, a negative value of T_k denotes a keyword that is out of fashion or no longer in use.

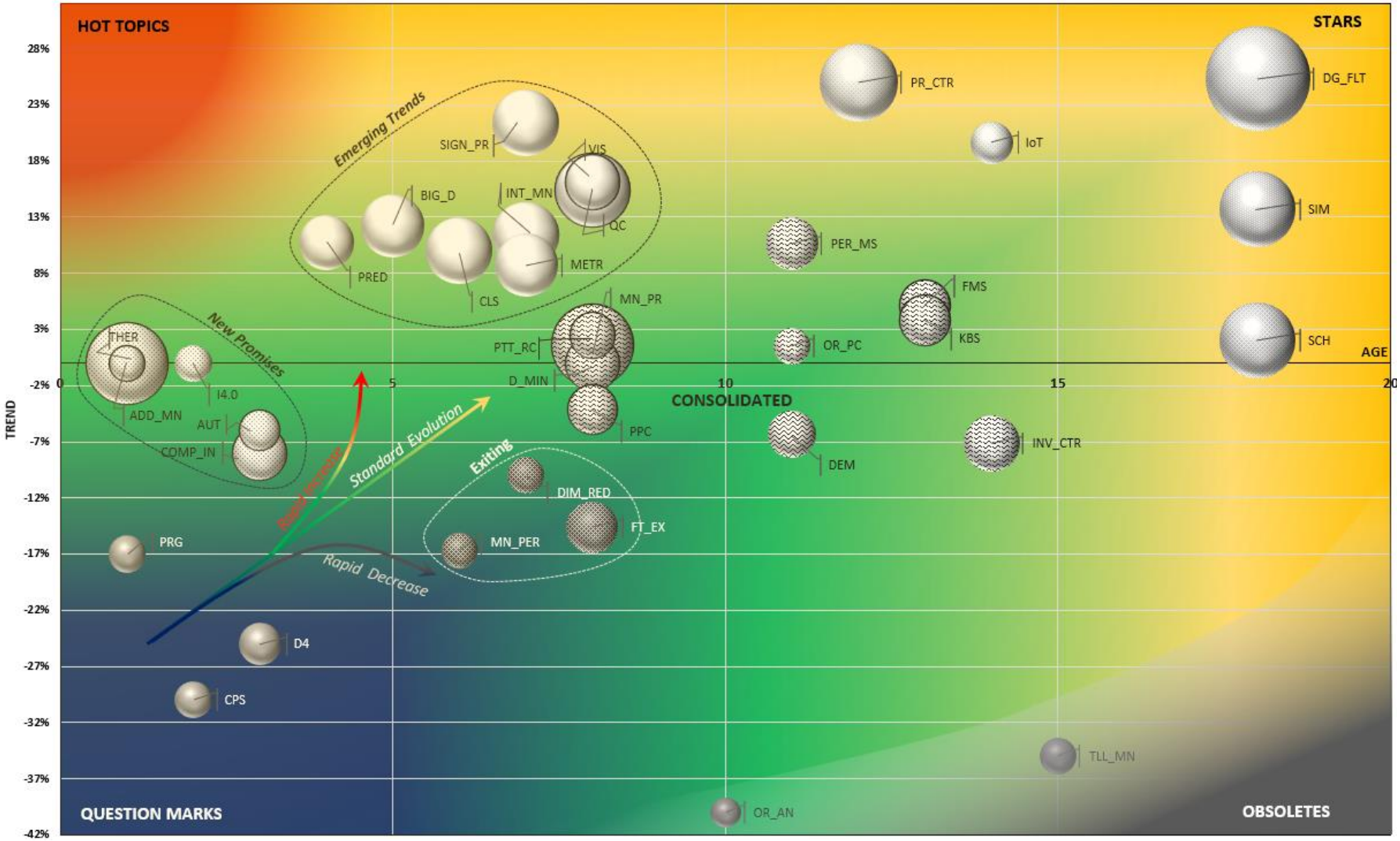


Figure 9 - Topics' Trend

Using these metrics, five areas can be identified in the graph:

- *Question Marks* (Low Age and Negative Trend) – These are recently introduced topics, that have not got a follow up, yet. Prognostic (PRG), Cyber Physical Systems (CPS) and Design For (D4) belongs to this category.
- *Hot Topics* (Low Age and Negative Trend) - These are very recent and also rapidly increasing topics. None of the identified keywords properly belongs to this category, but Prediction (PRED) and Big Data Analytics (Big_D) are the closest ones.
- *Consolidated* (Medium Age and Stable Trend) - Not recently introduced topics, which continue to be studied, without presenting peaks of interest. Topics such as Pattern Recognition (PTT_RC), Data mining (D_MIN) Production Planning and Control (PPC), Manufacturing Processes (MN_PR) and Performance Measurements (PER_MES) belong to this category
- *Stars* (High Age and Positive Trend) – Old and consolidated topics that are still attracting an increasing research interest. Diagnosis and Fault Detection (DG_FLT) and Simulation (Sim) belongs to this category and Internet of Things (IoT) is probably on its way to become a star topic.
- *Obsoletes* (High Age and Negative Trend) - Old topics that have never received a great scientific interest and success and that have practically disappeared from technical literature. Order Analysis and Tool Monitoring (TLL_MN) belong to this category.

We also note that, as shown by the arrows in Figure 9, in the standard case question marks should become consolidated topics, moving along a diagonal trajectory from the bottom left corner to the middle of the graph. Conversely, in case of a rapid success, question marks move almost vertically along the graph toward the hot topics area and, from here, they can proceed toward the stars' area in case of a continuously growing trend. In this regard, two additional clusters, namely New Promises and Emerging Trends, can be identified. The first one contains recent topics that have already overcome the initial phase of uncertainty and that are likely to remain of interest in the years to come. The second cluster contains rather recent topics growing in popularity, which can be expected to become consolidated or even star topics in the next few years.

2.2.4. Detailed Analysis of Selected Papers

2.2.4.1. Maintenance Management

With a total of 13 relevant papers from 2003 to present, maintenance management is one of the most studied area. As shown by in Figure 4, maintenance management is one of the most studied area. Maintenance management is about maintaining the resources of the company so that production proceeds effectively and that no money is wasted on inefficiency. Machine learning is a natural fit in this area: especially within the Supervised Learning framework, the task is naturally interpreted as a prediction task, where historical data is generally collected in the production floor, and faulty and non-faulty events of equipment are recorded as ground-truth data against which train a prediction model. Indeed, the most widely studied techniques are SVM (37 papers) and NN (41 papers).

In this section we will focus on work after 2007. For a review of earlier works please refer to [30], which reviewed the application of SVM to monitor machine working conditions and for fault-diagnosis. Indeed, while SVM seemed to show best performances compared to NN in the past, the popularity of the approach has been steadily decreasing, in favor of the more promising neural network approach. Indeed, in recent years more sophisticated algorithms have been introduced, such as those based on Deep Learning [31]: for example, Deep Belief Networks [32] and CNN [33]. Typically, the most studied applications are condition monitoring of rotating mechanical systems ([34], [35]) and ball bearing fault diagnosis ([36], [37]). Rotating machineries are ubiquitous in manufacturing environment, and bearings play a crucial role in mechanical components and one of the main causes of breakdown of rotating machines. The most common approach has been to record vibrations or acoustic signal through sensors. The signals are then used as inputs to classifiers for faulty or non-faulty condition predictions. While past works tackled the feature selection problem, as well as hyper-parameters optimization problem (e.g. implementing metaheuristics algorithms), more recent techniques offer the advantage of eliminating some of these limitations. The same applies to dimensionality reduction, for example by means of PCA, which no longer crucially undermines the Deep Learning algorithms performances. However, when Deep Learning techniques are employed, often more massive datasets are needed compared to older methodologies. This can be sometimes challenging, and often researchers resort to simulations. The most interesting works are the following:

[38] used neural networks to detect bearing fault detection, proposing a novel approach to monitoring of electrical machines, which took in consideration both local and distributed defects. They still needed to cope with dimensionality reduction, and proposed to apply first a curvilinear component analysis, and then a classifier based on two-level hierarchical NN.

[39] proposed an ensemble of Neural Networks and SVM with metaheuristic based hyperparameter optimization to condition monitoring of centrifugal pump for effective maintenance management. They analyzed the robustness for the approach with regards to corrupted or noisy data.

[40] used an ensemble of SVM and k-NN for predictive maintenance and applied it to a benchmark semiconductor manufacturing maintenance problem. The proposed approach aimed at minimizing expected costs, taking into consideration different prediction horizons to handle the tradeoff between unexpected breaks and machine unexploited lifetime.

[37] compared CNN and Dynamic Time Warping to more traditional ones for a simulation-driven machine learning bearing fault classification task. In this work, they tried to address main limits of past applications such as reliance on historical data, which are often limited to components with extended service life. Instead, they generated the training dataset by means of high-resolution simulations of roller bearing dynamics.

2.2.4.2. Quality Management

Three important application areas emerge from the studies which are going to be discussed: semiconductor and printed circuit board manufacturing ([41], [42], [43]), visual inspection ([44], [45]), and quality prediction and monitoring ([46], [47], [48], [49], [50]). Overall these studies clearly indicate the importance of Machine Learning and how the analysis of production data, sensors data and even after-sales data can be implemented to help predict and understand the drivers for a better quality. Some of these studies directly tackle the complexity of the class imbalance problem, which can be one of the main obstacles. This is vital if one wants to correctly tackle the quality management problem using machine learning techniques. All the studies reviewed here support the hypothesis that Machine Learning techniques are effective in giving special insight into the quality Management problem, as especially seen in the risen interest in the last two years. As already said, semiconductor manufacturing has received a lot of attention. [51] tested seven different ML

techniques against real-world semiconductor manufacturing data, with more than 150 input variables. Thus, dimensionality reduction techniques have been implemented to reduce the dimensionality of such dataset. TPR (True Positive Rate) has been used as a way of assessing the performance of each algorithm. The goal was to discriminate which test example was a novelty, meaning an object that was being manufactured outside the normal range of production settings. The most promising tools found is 1-SMV, a tweak to the SVM method suited for one-class classification problems.

[52] investigated the applicability of ML techniques to printed-circuit board-level functional diagnosis. They proposed an ensemble of NN and SVM based on weighted majority votes in order to successfully discriminate between defective and non-defective boards. They tested this approach against a real manufacturing dataset, augmented with synthetic data to cope with an imbalanced dataset.

[53] focused on the imbalance issue that is quite common in settings in which the objective is to discriminate failure (thus rare) events from success events, such as in quality management or maintenance management tasks. Indeed, to effectively produce high-performing fault-detection models, the semiconductor manufacturing industry highlighted class imbalance as a major impediment.

[54] proposed a Big-Data based long-term automated monitoring system in the semiconductor manufacturing to enable engineers to obtain significant yield enhancements. They also investigate Deep Learning techniques: indeed, CNN positively outperforms SVM.

Applying it to electronics-rich analog systems, [55] implemented Deep Belief Network, which can be interpreted as a composition of Restricted Boltzmann Machines, for fault detection and isolation, measuring a successful implementation compared to traditional feature-extraction methods: indeed, in one of the experiments, they successfully measured an accuracy level of 100%, which indicates the ability of these special Neural Networks to effectively capture highly discriminative semantic features.

Regarding the quality monitoring and prediction theme, [56] proposed a different approach to estimate the production quality level. They proposed a combined approach of Clustering and Supervised Learning to assess the product state during its production process lifetime and to link the classified state to the driver state. They used different characteristics as proxy of the product state at each stage. They implemented a first stage of unsupervised learning, by means of Hierarchical Clustering, in order to obtain labels needed for the second

stage, the supervised one, carried out by means of SVM. This hybrid approach is often called Semi-Supervised Learning. It tries to tackle the problem of having a training set. [57] proposed a framework to detect anomalies of heavy machinery engines integrating manufacturing, inspection and after-sales data. They pose the problem as a one-class classification task, given the severe imbalance problem they tackled. This is yet another example of directly addressing the pervasive issue of class imbalance. Gaussian Mixture Models and Parzen Window Density Estimation proved effective, compared to other techniques such as PCA or K-Means Clustering.

[58] used sound analysis to detect defective bearings in home appliances before the final shipping. Ball bearings are installed in the first steps of the production process and cannot be inspected until the assembly is completed. Thus, sound analysis is one of the main techniques used to assess components quality when direct inspection is not feasible. Many algorithms were tested, such as Support Vector Machine, Neural Networks, and Decision Trees. The latter proved the most successful, and it is the more intuitive and easier method to implement, which is of course a very appealing and interesting characteristic, especially for the everyday user. It must be noted that a thorough pre-processing step was required to increase the generalization performance. Nonetheless, the proposed solution has been implemented by the company that required the study.

[59] investigated the applicability of predictive machine learning models in the context of virtual metrology. DT, NN and SVR have been applied to predict the thickness of dielectric layers deposited during manufacturing of semiconductor wafers.

In the area of quality prediction, [49] proposed a data-driven quality prediction and monitoring solution based on SVM for an abrasion-resistant material manufacturing process, trying to address the issue of increased complexity and high dimensionality of this kind of processes.

In the visual inspection area, [60] studied the application of Machine Learning in the reduction of defects and yield enhancement in the manufacturing of CMOS image sensors via automatic optical inspection, aimed at reducing false alarm rate. They specifically focused on this metric, directly responding to a specific industrial user need: while it is common to measure classifiers' performances based on overall accuracy, domain engineers often demand to realistically reduce type I or type II errors, thus making decision-making more robust to everyday needs. [48] proposed the application of machine learning techniques to assess

tomato ripeness, being the latter the main driver in from the customer perspective, and thus a main concern for the industrial process. Posed as a multi-class classification problem, they tested a combined approach based on PCA for feature extraction and on SVM and LDA for classification. Based on a classification metrics benchmarking, they showed that SVM outperforms LDA. As an image classification task, they highlighted the dataset size issue in these contexts as a limiting factor since images are basically high dimensional vectors. Large datasets are thus crucial to effectively train image classifiers algorithms.

2.2.4.3. Process/Production Management

This section refers to those works that have investigated the application of Machine Learning algorithm to directly optimize such processes (e.g. scheduling), or to gain a better understanding of the underlying patterns (e.g. control chart patterns). As for the other sections, we found the same trend with respect to the interest or researcher in assessing the utility of Machine Learning algorithms to sometimes tackle old problem with new tools. Indeed, most of the work here described has been produced in the last three years.

A typical problem in production is to assert whether a production process working parameters drift is occurring. Of obvious interest, if such anomaly detection is successfully implemented, then countermeasures can be applied to not waste time and money. A classic tool for handling these issues is the Control Chart Pattern. The most recent work about the application of Machine Learning algorithms has been done by [61]: a NN ensemble has been applied to handle autocorrelated data in control chart patterns. The proposed novelty is the ability to recognize seven typical types of unnatural CCP: quality practitioners are then equipped with a tool suitable to find the anomaly cause.

Probably the most studied problem in the production management field, scheduling problems pose different challenges and thus many works have tried to tackle it by means of Machine Learning algorithms. Scheduling is a very important subject because it is extremely challenging from a research perspective and a real issue from practitioner's perspective. When correctly handled, scheduling algorithm can yield an important competitive advantage. Indeed, it is a difficult problem because most of the times it is essentially impossible to find an optimal solution to all scheduling problem that may arise.

Tackling the problem of dynamic scheduling both in a flow-shop and job-shop environment, Priore studied the implementation of Machine Learning algorithms to learn the best dispatching rule to deploy as a function of the system state. Starting from simply investigating the feasibility of the Decision Tree algorithm C4.5 [62], it then benchmarked it against k-NN and Neural Networks [63] in which it showed, by means of ANOVA testing, that k-NN outperformed the other algorithms. Finally, in [64], it tested the SVM algorithm. In these works, the aim was to find the best dispatching rule based on current system conditions, so to maximize system performance, namely mean tardiness and mean flow time.

[65] proposed a combination of a Simulated Annealing, Reinforcement Learning and Neural Network framework to implement an adaptive iterative distributed scheduling algorithm for market-based production in which each machine and job is seen as an agent and can participate in a bid system with the global aim of minimizing the total production time in presence of unexpected disruptive events such as machine breakdown, introduction of a new machine, job arrival or job cancellation.

[66] proposed a combination of Genetic Algorithm and Neural Network for a flexible job-shop environment in which a product-centered event-driven dynamic rescheduling framework has been implemented. The Genetic Algorithm has been used to grow the training dataset, namely during a simulation phase, each time a specific kind of event was triggered, the genetic algorithm was deployed to regenerate an obsolete scheduling sequence. The new sequences generated were stored to get new learning example. the neural network had the goal of finding interesting patterns in the structure of these scheduling sequences. Thus, it is implemented as a product agent's decisional kernel.

[67] proposed a Reinforcement Learning approach based on Local Weighted Regression to effectively implement an order acceptance policy, in which the exploration is conducted using works from the rejection set, while exploitation is conducted using works from the acceptance set.

[68] implemented a Self-Organizing Map (SOM) to solve a real-time scheduling problem. While traditional supervised approaches to multiple scheduling rules system can perform well but need the time-consuming process of predefining the classes for which the training data must be collected (by means of simulations for example), the SOM-based approach overcome this problem. As Unsupervised Learning algorithm, it can autonomously

extrapolate the suitable classes which best explain the data. Better performances have been confirmed compared to traditional Machine Learning approaches.

[69] proposed an intelligent real-time re-scheduler, in which a Q-Learning algorithm has been implemented as the core method. The Relational Reinforcement Learning framework is tested against the problem of rescheduling due to unforeseen events (e.g. arrival of a rush order or breakdown of a working machine). Given a change in state due to unforeseen event, the value for each schedule repair policy is learned through simulation.

[70] investigated the application of Gaussian Processes to dynamically adjust parameters of dispatching rules based on current shop-floor conditions. As a result, their approach was able to drastically reduce the mean tardiness of jobs.

More recently, [71] proposed a combination of colored Petri-nets and Q-Learning for order picking scheduling in a warehouse, to minimize travel time.

[72] used Reinforcement Learning to tackle the problem of production ramp-up optimization, posing the problem as a sequence of technical decision to be made in order to stabilize a manufacturing environment to yield a desired state in the shortest amount of time possible. They tested their implementation against a highly automated production plant and showed that the generated policy had significant impact over the speed-up of this process.

Combining a Q-Learning Reinforcement Algorithm and SVM methodology, [73] successfully implemented an electricity consumption reduction framework in an automation system.

2.2.4.4. Supply Chain Management

The most studied theme in Supply Chain Management is Inventory Control, and most of the works here listed investigated the application of Reinforcement Learning frameworks, thus highlighting the suitability of such approach in scenarios where multiple agents that interact with the environment are present.

The two-stage supply chains with non-stationary demand model has been one of the most investigated, where Reinforcement Learning algorithms has been applied with the aim of optimizing multiple performance indicators. In this stream of research, [74] studied two different inventory control models, centralized and decentralized. The state of the

environment is described by customer-demand patterns, and the choice of best control parameter both of supplier and retailer represent the action set.

[75] implemented a case-based myopic approach to a vendor managed inventory model for satisfying service level. [76] expanded this model studying a simulated multi-agent supply-chain system with 80 retailers and 10 customers, in which each of these supply chain actors were independent reinforcement Learning agents. They made suggestion for future works to investigate the handling of the bullwhip effect.

[77] focused on a supply chain environment with perishable products, with random demand and deterministic lead-time. The aim was to minimize total cost of retailer, and two different ordering policies have been studied. They applied both Q-Learning and SARSA algorithms. In [78] the application of Q-Learning has been tested against the bullwhip effect, in the settings of the famous Beer Game. They compared it to an implementation with a Genetic Algorithm and another algorithm.

2.2.5. Conclusions

The hype surrounding Machine Learning and Deep Learning algorithms is ever growing. Indeed, the literature analysis measured and highlighted an increasing trend of publications which deals with the application of such algorithms to industrial problems. Most explored are Supervised Learning methodologies, closely followed by Unsupervised Learning algorithms in more recent years. Reinforcement Learning methods, given their higher complexities, still lack behind. Nonetheless, many manufacturing areas, such as Additive Manufacturing, Scheduling, Diagnosis and Fault Detection, have seen rising implementations of many algorithms, such as Neural Networks and Support Vector Machines. These trends clearly state that such methodologies can be effectively applied to tackle and solve many industrial problems. Moreover, if the trend will continue, more and more successful applications will be studied, as the most recent advances in these Artificial Intelligence methodologies will be translated and tested against industrial tasks. The institution of benchmark datasets could help researchers and practitioners develop and test new algorithms, for effective and efficient deployment in real case studies.

3. Machine Learning models for bankruptcy prediction in Italy

3.1. Introduction

The first application presented in this thesis deals with the bankruptcy prediction problem. Specifically, the focus will be on manufacturing companies operating in the Italian marketplace and a set of Machine Learning models, specifically designed to forecast bankruptcy events, will be presented and compared.

Bankruptcy prediction has always been a traditional and challenging task tackled both by academics and practitioners and literature in the subject matter is extensive. Indeed, due to its strong practical interest and to its complexity, it is frequently used as a testbench to assess the performance of alternative prediction models. Moreover, after the 2007/2008 financial crisis, credit risk management has become an even more impelling priority, as financial institutions, banks, governments, financial market players, as well as corporate firms have an urgent need for operating tools to efficiently assess the probability of insolvency of enterprises.

In more recent years, given recent breakthrough [27], academics and practitioners are exploring artificial intelligence and machine learning algorithms to tackle many challenging problems. However, regarding financial applications, these recent advances are shallowly explored, thus leaving a lot of potential to be discovered. Given that credit risk analysis can be cast as a pattern recognition problem, Machine Learning classification algorithms can thus be used to measure the probability of bankruptcy [79], [80], which could improve upon traditional models based on simpler multivariate statistical techniques such as discriminant analysis and logistic regression. However, [81] points out that no precise and definitive bankruptcy models are available. Indeed, such statement can be derived from the well-known "No Free Lunch Theorem" [29], which states that it cannot exist a globally optimal classification algorithm which can be employed for any bankruptcy prediction task. As a consequence, for each specific dataset, one needs to compare and then deploy different algorithms tailored to the task at hand. Surely, the work here presented is not merely about the test of a standard model, but indeed required a thorough exercise of data preprocessing, coupled with a thoroughly thought methodological approach to deal with the innate problematics of the task at hand.

This work aims to verify the performance traditional statistical methods (Logistic Regression) and of advanced Machine Learning (Random Forest and XGBoost [15]) and Deep Learning models applied to this traditional forecasting task. Another innovative aspect is the application to the Italian economy, focusing on manufacturing sectors, including not only financial variables (taken from the balance sheet), but also industrial variables, which will provide an additional point of view on the interpretation of the results.

3.2. Dataset and preprocessing

The analysis focused on balance sheet information of manufacturing Italian firms extracted from AIDA Bureau van Dijk, ranging from 2007 to 2015. Both the response variable (financial status) and the covariates can be found in this database, with the exception of the variable describing the district affiliation status. Such information is obtained using the Industrial District Database from the Italian National Statistical Institute (ISTAT), merging the AIDA dataset with the ISTAT dataset based on the ZIP code of the firm's operative headquarter.

Moreover, the response variable is further encoded as a binary variable, based on the field 'status' obtained from the AIDA database. The response variable 'bankruptcy' takes value 1 if the 'status' is equal to bankrupt, 0 otherwise. From now on, the group of bankrupt firms will be referred to as B, and NB (non-bankrupt) vice versa.

A first phase of data preprocessing, including exclusion of missing observation, inconsistencies, or extreme values, has been performed, before more complex elaboration could take place.

We stress the fact that the dataset obtained from the AIDA database has a cardinality which is at the level of single firm. Balance sheet entries describing each of the 9 balance sheets extracted for each firm are, at first, arranged as separate columns. Next, the dataset was rearranged so to obtain different statistical observation for each balance sheet for each year for each firm. Thus, the next data processing step implemented was to keep, for firms belonging to the B group, only the balance sheet of the year preceding the bankruptcy event. Conversely, for each NB firms all available years are considered.

This first pre-processing exacerbated the already aggravating issue of high imbalance ratio of B firms over NB firms, which is around 15%. To tackle such issue, two different

strategies are adopted, namely partially downsizing the NB class and using class weighted loss functions in each methodology (with the notable exception of the XGBoost algorithm). For the latter approach, for each NB firm, we keep only three different balance sheets, equally spaced in time. A third of the firms are associated to years 2007, 2010, 2013, a third to years 2008, 2011, 2014, and the last third to years 2009, 2012, 2015.

The final dataset consists of 4,774 B and of 22,359 NB considered in three equispaced years (67,077 NB observations), for a total of n =71,851 balance sheets, with a final imbalance ratio of 7.12%. The class weighted loss function approach used to tackle the imbalance issue is inspired by [82]. Namely, a different weight is associated at each class: $w_i = \frac{n}{2n_i}$, with i $\in$ {B, NB} and n_i = the number of observations in the corresponding class. In our analysis we obtain w_B = 7.52 and w_{NB} = 0.53.

3.3. Predictive variables

One of the first most important work is the one by [83], which identified a set of five 3financial ratios and applied the multivariate linear discriminant analysis to establish a function to classify sixty-six firms as bankrupt or not. Expanding on this, [84] identifies 79 financial ratios, to be divided into three groups, namely profitability, managerial performance, and solvency ratios.

As the first works focused on financial ratios, [85] stressed the fact that, indeed, many external factors could influence the survival and performance of the firm, such as national, international, and environment conditions. Moreover, different conditions could lead to very different prediction models.

Starting from this assumption, many other studies tried to include, beside financial ratios, qualitative variables. Indeed, this approach is not new to the literature, for example [86] tried to include explanatory variables such as management quality, level of research and development expenditures and market trend. [87] gave more emphasis on the social importance of the firm and the strength of the relationship with the bank. The idea seems to have been abandoned for a while, but in more recent years, probably thanks to easier access to datasets and computing power, new contributions have been made to investigate the importance of non-financial ratios, both in terms of interpretability and in performance increase. For example, [88] explores productivity, profitability and growth as additional

variables, [89] inspect size and age, [90] examines corporate governance indicators, and finally [91] probes an institutional set of predictive variables.

This work, to the best knowledge of the author, is the first to examine the significance of industrial and regional indicators in addition to more traditional financial ratios.

The two most critical variables added refer to Industrial District membership and to a measure of firms' mark-up, which are considered to be decisive predictors for bankruptcy events. The former indicates whether a firm belongs to an industrial district or not. Industrial districts, enabling local cooperative environment, help small innovative firms to be highly competitive despite an ever-growing global arena.

Membership to an industrial district can give rise to short-term and long-term benefits. The latter are more studied. Indeed, the seemingly disadvantage of the small scale of such firms is counterbalanced by other dynamics, such as reduced transportation costs within the district, high availability of specialized workers and suppliers, technological and know-how spillovers within the district. Indeed, [92] states that industrial districts enable small firms to exploit external-scale economies, in the same way as large firms exploit internal-scale economies.

Specifically, within the Italian ecosystem, more emphasis is given to the role of cooperation and the link between social and economic forces, which in turn are fostered by the trust that industrial districts help to create, as [93] highlights. Indeed, [94] shows that belonging to industrial districts give rise to agglomeration advantages. However, no unanimous conclusion can be drawn from recent works which cover different countries ([95], [96], [97], [98]).

Moreover, also short-term effects can be taken into consideration, meaning that membership of an industrial district can benefits the business cycle, in particular during recessions ([99], [100]). According to [99] the intense social interactions within the ID are likely to amplify the responses to negative shocks acting as a social multiplier. In fact, belonging to an industrial district can worsen the performances of firms during recession, compared to similar environment. Conversely, industrial district membership can increase trust and thus lead to better access to credit though relationship lending.

Another critical variable which is considered in this work is a measure of mark-up, namely the Price Cost Margin defined as Total Sales/(Labour Cost + Nominal Materials)) – 1. While more traditional measures of profitability have been widely taken into consideration,

the proposed formulation tries to measure profits relatively only to the core business of the firm. Moreover, it also identifies the firm market power, since it assesses the amount of mark-up that the firm is able to get from its core customers. Indeed, the higher the market competition, the lower the PCM should be. Without barriers to entry, prices should be equal to marginal cost. A certain degree of market power can be derived from a positive and persistent PCM. From the firm perspective, a high mark-up, on one hand, might be related to higher profits and thus more financial resources to increase investments and innovative activities that could reduce production costs [101]. On the other hand, a high mark-up might also mean more product diversification (variety) and higher barriers of entry for external firms. These two factors could be potentially important drivers to reduce the firm's probability of going bankrupt.

To summarize, the explanatory variables used in this work are the following, as listed in Table 1.

Variable	Formula
Liquidity	Net Working Capital / Total Assets
Productivity	EBIT / Total Assets
Leverage	Net Worth / Total Debt
Asset Turnover	Total Sales / Total Assets
Operational Margin	EBIT / Total Sales
Growth of Assets	Growth rate of Total Assets
Growth of Sales	Growth rate of Total Sales
Change in return on equity	ROE delta
Sector of affiliation	Sector
Regional location	Region
Membership to Industrial district	Industrial District
Markup	PCM
Market Share	(Firm value added) / (Sector value added)

Table 1 - Predictive variables for the bankruptc prediction

3.4. Models

As already anticipated, given the high imbalance ratio the dataset presents, the main methodological approach used is based on weighted loss functions, in which different costs are associated to prediction error for each group (B and NB). The aim is to avoid obtaining dull classifiers which would simply classify each observation as belonging to the majority class. Moreover, in this way it is possible to mimic the decision process that a credit institution would follow, assigning higher costs to prediction errors of NB observations. A brief detailed explanation of the implementation of the weighted loss function and of the hyperparameters is given for each predictive model here employed.

- **Weighted Logistic Regression:**

  The Weighted Logistic Regression is trained using Maximum Likelihood, with respect to the log-likelihood function, which can be express as two different sums, in terms of the class to which observation belongs to. Moreover, at each sum is associated a corresponding weight. The resulting likelihood function is as follows:

$$L = w_B \cdot \sum_{y_i \in B} y_i \cdot \log(\hat{y}_i) + w_{NB} \cdot \sum_{y_i \in NB} (1 - y_i) \cdot \log(1 - \hat{y}_i) \tag{14}$$

- **Neural Network:**

  The MLP tested has the following topology: three hidden layers with 16 neurons each, using the Relu activation function. To consider the imbalance ration the weighted binary cross-entropy loss function as been used, as follows:

$$-\frac{w_B}{n_B} \sum_{y_i \in B} L(y_i, \hat{y}_i) - \frac{w_{NB}}{n_{NB}} \sum_{y_i \in NB} L(y_i, \hat{y}_i) \tag{15}$$

  In which $L(y_i, \hat{y}_i) = y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)$

- **Random Forest:**

  To cope with the imbalance ratio, at every node the splitting criterion is essentially a weighted Gini Index based criterion, which aims to maximize the following quantity:

$$WID = \frac{n_{node}}{n}\left[G_{node} - \frac{n_{right}}{n_{node}}G_{right} - \frac{n_{left}}{n_{node}}G_{left}\right] \quad (16)$$

In which $n_{node}$is the number of observations in the considered node, and $n_{right}/n_{left}$are the numbers of observation in the right and left branch, respectively. These are weighted sums, with respect to the weights associated to each group.

- **XBGoost:**

The XGBoost algorithm does not allow to specify weighted costs for the loss function. However, it is possible to cope with the imbalance ratio tuning the *scale positive weight* parameters, which essentially adjust the weights associated to the classification errors of the minority class.

3.5. Results

The benchmark statistical model chosen is the Weighted Logistic Regression. The classification performance of this model has been compared to some Machine Learning techniques, namely Neural Networks, Random Forest, and XGBoost. All these methods have been implemented using Python, with Keras and Scikit-Learn packages. Cross-validation has been used for hyper-parameters optimization.

In order to measure and compare performance of the predictive, the overall dataset with 67,077 observations has been split into training set and test set (respectively 75% and 25% of observations). The split has been done in a stratified manner, so to reproduce the original imbalance ratio of 7.12% both in the training set and in the test set. The split has been performed several times, so to obtain averages of performance measures. Indeed, we performed a *repeated random sub-sampling validation*. In particular, the split has been repeated 200 times, and at each split we trained each model and measured performances. Moreover, for the Weighted Logistic Regression we build a confidence interval around the averaged regression coefficients in order to test significance.

Given the classification task, the performance measures used stem from the confusion matrix, in which are reported two type of classification error and two type of correct classification, as exposed in Table 2:

	Classified as NB	*Classified as B*
True NB	True Negatives (TN) = NB correctly classified	False Positive (FP) = NB misclassified
True B	False Negatives (FN) = B misclassified	True Positive (TP) = B correctly classified

Table 2 - Confusion Matrix

Starting from these foundational measures, the performances index that we used are the following:

- Type I error $=\frac{FN}{FN+TP}$, measures the overall percentage of misclassified B.
- Type II error $=\frac{FP}{FP+TN}$, measures the overall percentage of misclassified NB.
- Recall $=\frac{TP}{FN+TP}$, measures the overall percentage of correctly classified B.
- Precision $=\frac{TP}{FP+TP}$, measures the overall percentage of correctly classified NB.
- F-measure $= (1+\beta^2)\frac{precision \cdot recall}{\beta^2 \cdot precision+recall}$, with $\beta = 1$ measures the harmonic mean between precision and recall (F1-score), with $\beta = 2$ weights double the recall w.r.t. precision, and it is a more precise formulation in case of high imbalancement (F2-score).

Given that the task at hand is shows a severe imbalance ratio, these indicators must be considered altogether, as no single performance index is able to correctly measure model performance at once. For example, the F2 score, which places more emphasis on the Recall, is able to counteract the fact that in our case the Precision measure is highly influenced by the imbalance ratio between B and NB. F1 and F2 scores are alternative measures of reporting accuracy, more suitable in unbalanced scenarios. Correspondingly, the accuracy score is not informative, and it is thus not presented, as it could lead to misleading interpretations.

In addition to different algorithms, two different specifications were tested, namely a forecasting scenario in which only traditional financial ratios are used, and another scenario in which also industrial variables are employed. The rationale behind this distinction is to assess the impact of the industrial variables on the classification performance, as well as enabling more sophisticated interpretability of results.

Moreover, the *repeated random sub-sampling validation,* splitting the dataset into training and test set, mimics the way a credit institution would apply such forecasting models for decision support in loan granting activities. The credit institution would, in fact, try to predict insolvency for new clients whose balance sheet were not available during the model training phase. Moreover, also the decision to use weighted loss functions for Logistic Regression, as well as for all Machine Learning model (except for the XGBoost algorithm), mimics the way a credit institute would weight differently Type I error w.r.t Type II error, meaning that, a less aggressive credit institute would prefer to minimize capital loss, thus it would put more weight on B misclassifications. Indeed, the concern of reducing as much as possible the number of NPL (Non-Performing Loans) has been replicated using the weighted loss function approach, using the weights described in the previous section.

As Table 3 shows, from the Type 1 error perspective the best model in terms of predictive performance is XGB followed by WLR and RF, whereas the NN seems not to work as well as the other two computational techniques. On the other hand, RF and NN work much better in reducing the Type 2 error. Moreover, forecasting models can be divided into two groups: those with a more risk averse policy, and those less. The former is composed of the Weighted Logistic Regression model and the XGBoost algorithm, while the latter is represented by the Neural Network method and the Random Forest model. Finally, XGB is also the model with the lowest Type 1 error which translates in the ability to classify correctly around 90% of bankrupt firms, which is a great achievement in line with other results in the literature ([102], [88]).

	T1	T2	F1	F2	Recall	Precision
Financial Ratios						
Logistic Regression	15.26	19.12	37.39	56.24	84.74	23.99
Random Forest	16.22	10.36	50.91	66.57	83.78	36.57
XGBoost	12.31	17.42	40.57	59.87	87.69	26.39
Neural Network	17.02	11.02	49.53	65.15	82.98	35.54
Financial Ratios and Industrial Variables						
Logistic Regression	15.06	19.71	36.79	55.75	84.94	23.48
Random Forest	15.55	11.26	49.32	65.71	84.45	34.84
XGBoost	10.72	19.22	38.89	58.80	89.28	24.86
Neural Network	17.26	10.89	49.72	65.19	82.74	35.79

Table 3 - Predictive performance across models

In relation to the importance of the industrial and regional variables in adding predictive power, Table 1 also shows that with the exception of NN all the other types of models see a reduction in Type 1 error when adding industrial and regional variables. Once again XGB seems to be able to use more efficiently the information coming from firms' industrial structure to reduce the prediction Type 1 error. For this model the advantage of using industrial variables is substantially increasing the capacity of correctly classifying B firms from 87.69% to 89.28%. This result is particularly striking given that the model was already performing very well only with financial ratios.

In order to further investigate the importance of industrial and regional variables for forecasting firms' bankruptcy within the Italian ecosystem, average results of logistic regression coefficient are reported in Table 4, to examine both the coefficient sign and the significance of each variable.

	M1	*M2*
Financial Ratios		
Net working capital / total assets	-3.975***	-4.022***
Net Worth / Total Debt	-0.827***	-0.824***
Total Sales / Total Assets	-1.264***	-1.270***
EBIT / Total Assets	-10.428***	-10.151***
TA growth	0.404***	0.385***
TS growth	0.160***	0.164***
Roe Variation	0.004	0.004
Industrial Variables		
Mark up		-0.113***
Market Share		-6.398***
District dummy		-0.142***
Sector Dummies		yes
Regional Dummies		yes

Note: M1 = only financial vars, M2= financial & industrial vars.

*Significance levels: *=10%, **=5%, ***=1%*

Table 4 - Logistic Regression coefficient

Indeed, industrial variables have a significant influence on bankruptcy probability: both a high markup and a high market share lower the probability of bankruptcy. Furthermore, the food and machinery sectors show lower probability of failure compared to other sectors, as well as southern regions display higher probability compared to other regions.

These results are both not surprising as well as novel. Results about regional disparity and sector inequalities are in line with the increasing dualism within the Italian economy, as well given that the food and machinery sectors have stronger capacity to differentiate their products, compared to other sectors.

On the other hand, the result about the relationship between industrial district membership and insolvency is very novel, as the empirical literature on this argument does not convey any major result describing this dynamic. A possible explanation could be related to the presence of social capital, which increases the level of trust among firms and institutions sharing the same territory. A higher level of trust might in turn translate, for example, into

easier access to credit which could be decisive in curbing the probability of going bankrupt. Furthermore, the result about the mark-up seems to suggest that a high mark-up is associated with an efficient use of the firms' large rent and/or to a greater market power, as well as higher the size of the firm relative to the sector, the lower the bankruptcy probability.

3.6. Conclusions.

Two main outcomes can be drawn from the work, one methodological and one about interpretability of the obtained forecasting models.

For the former result, the Weighted Logistic Regression shows comparable results to Machine Learning techniques, restricted only to Type I errors, while Type II errors are consistently higher. Moreover, Random Forest and Neural Networks are able to reduce Type II errors only. The XGBoost algorithm, instead, is the best performer overall.

As for the latter result, the set of industrial and regional variables seem relevant for determining the forecasting probability correctly, as well as for incrementing forecasting performance of the models.

For future works, more methodological approaches could be taken, for example comparing different techniques for coping with the imbalance problem, such as using generative algorithms to synthetize additional fake observation for the minority class, in order to avoid downsizing the majority class, or to avoid using class weighted loss function, given that no objective formula for setting the weights is available.

4. WLC enhancements through Deep Learning and Statistical Optimization

4.1. Objective

In this section we continue the analysis of the potential of Machine Learning techniques for forecasting tasks. Unlike what was done in the previous paragraph, where the objective was to predict the probability of bankruptcy of a company, in this case we wanted to apply Deep Learning methodologies against a regression task of a typical industrial case. The objective will be, in fact, to forecast the production lead time of orders issued in a Job Shop production system and to use this forecast to formulate reliable delivery dates. This is an ambitious objective, nevertheless one of great operational scope. Indeed, especially small and medium manufacturing companies often find themselves in trouble when, faced with a request for a quotation, they have to decide whether to accept the order and must agree on a specific delivery date. The definition of a reliable delivery date is in fact a factor of strategic importance: too long dates can cause the cancellation of the order, whereas too short dates can lead to late orders, production planning difficulties, rising costs, and penalties.

In particular, to simplify the problem, the forecasting model will be tested in a job-shop controlled using an innovative production planning and control technique called WorkLoad Control (WLC), created specifically for the optimal management of HVLV-type job-shop systems. In addition to this, aiming to obtain greater optimization and stabilization of the system, it will also be introduced an optimization procedure based on simulative approach supported by statistical methodologies, such as Design of Experiments and Response Surfaces Methodology.

4.2. A conceptual background to WLC

Nowadays, lean manufacturing is getting a lot of attention, with the aim of increasing shop floor performances. Undoubtedly, it helps to develop a fully value-added manufacturing process, in which a perfect values stream is obtained, where jobs flow from one value added activity to the next. The focus of lean manufacturing is removing any waste that could be the source of tangible costs for the manufacturing company. Queues and excessive inventories

are one of the most common sources of waste in push systems. Nevertheless, they increase lead-times, both in terms of mean value and variability. Thus, defining Due Dates becomes very hard to accomplish, which make the firm vulnerable to higher customers' market power. Moreover, nowadays flexibility, rapidity and costs' reduction are key elements for competing in the Industry 4.0 arena.

Workload Control is a hybrid production and planning control (HPPC) system suitable for HVLV (High Variety Low Volume) manufacturing environments which are managed with a Make-To-Order policy, with a job-shop configuration [103]. Introduced the first time by [104] in 1981, with the aim of resolving the *lead time syndrome*, it has been receiving a great deal of attention, as demonstrated by the vast literature in the field [105]. HPPC systems main aim is to reduce "wasteful" variability to increase manufacturing performances. Their hybrid nature comes from the fact that they operate in a push fashion in terms of scheduling production, and in a pull way in terms of authorizing production. Among pull oriented PPC systems, WLC can be successfully applied even in case of job-shop with shifting bottlenecks, as it will be shown later in chapter 4.4.

In job shop MTO environments, the flexibility of the production system is often counterbalanced by a costly weakness caused by workloads balancing issues and lead times variability. In this context, one of the main operational levers can be the balancing of production and Work in Process (WIP), which is exactly the goal of WLC [106].

With the aim of decoupling the manufacturing system from the order acceptance stage, the WLC system introduces both a Pre-Shop Pool (PSP) for accepted orders with a production order release mechanism. The PSP is a staging environment, which can be implemented totally via software, in which all the accepted orders are gathered waiting for actual release in the manufacturing environment. Within the PSP, orders can be sorted with any given rule, for example using a simple First-In-First-Out (FIFO) rule or an Earliest Due Date (EDD) rule. At specific time interval, orders in the PSP are evaluated for release in the shop-floor. The release of new jobs in production is managed by checking that the value of certain WIP caps do not exceed a certain threshold, known as the system *norm*. The norm regulates the maximum amount of WIP which can be measured at any time in the production environment. The aim is to keep the queues in front of each work center (machine) as lowest as possible, without incurring into starvation, i.e. without reducing production rates, and maintaining

unaltered the throughput rate. Norms are generally quantified as the number of expected processing hours.

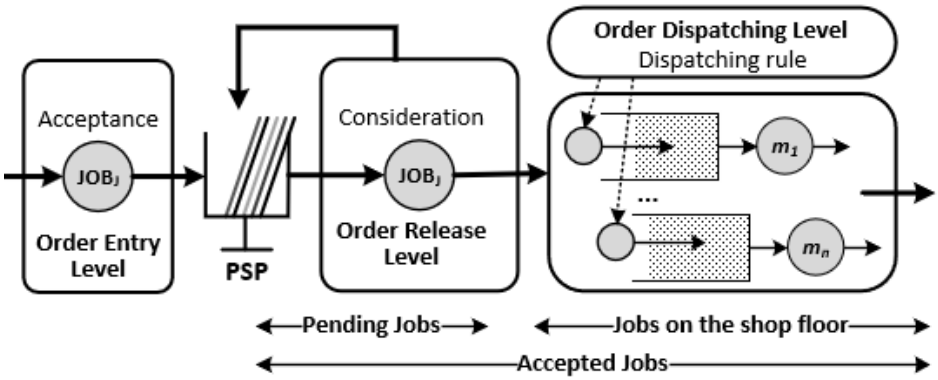


Figure 10 - WLC three levels of control

More precisely, as it can be seen in Figure 10, WLC employs three different control levels, namely *Job Entry*, *Job Release*, and *Job Dispatching*. Within the former, as new orders coming in from customers are evaluated, those that are accepted are placed within the PSP staging area. Due Dates (DD) are also defined. The way in which DDs are defined may depend on the respective bargaining forces of the customer and the manufacturer. As for the second, as jobs pending in the PSP are sorted using a given dispatching rule, either continuously or at discrete time interval jobs are considered to enter the shop-floor. To this aim, taking into consideration a generic job *j*, its workload contribution is calculated. The system workload W and each machine workload W_m are updated accordingly, as can be seen in equations 17 and 18:

$$W_{t+1}^{m} = W_{t}^{m} + w_{j}^{m} \tag{17}$$

$$W_{t+1} = W_{t} + \sum_{m} w_{j}^{m} \tag{18}$$

where the machine and system workload contribution of the job *j* are, respectively, the second addendum in each equation. More specifically, the job workload contribution can be calculated either using an *aggregate* approach or a *probabilistic* approach. In the first case, the workload contribution is given by the sum of the job setup time and processing time on any given machine. Conversely, in the second approach the workload is rescaled, in order to take into consideration time gap needed for the job to actually reach a given machine. Usually,

the workload is rescaled by means of dividing it by the position of the machine within the job routing.

Whenever, after updating the workloads using equations 17 and 18, none of the norms are violated, then the job is released to the shop floor, otherwise it is left within the PSP staging area. Either the control can be at the system's WIP level, referred to as *Shop Load control*, or it can be extended to each machine, which is referred to as *Bottleneck Load control*.

As for the latter control level, each time the job is released, it will follow its routing and it will be pushed from a workstation to the next one. For this very reason, the WLC is considered a hybrid push and pull system. As queues build up in front of each work center, they are sorted using some dispatching rule.

4.3. Deep Learning and WLC: realistic due dates setting in HVLV manufacturing systems

4.3.1. Introduction

As already briefly mentioned in the conceptual background of WLC theory, the definition of Due Dates (DD) is of crucial relevance for the success of a WLC managed system. Indeed, the promise and respect of short DD, in response of customers' enquiries, is a key success factor.

WLC makes possible to dramatically reduce WIP and queuing times preserving the maximum throughput rate (of a comparable push system), and at the same time the Throughput Time (TT) becomes shorter and more predictable. Thus, the definition of realistic DD is made possible.

While the WLC per se does not provide any framework for managing Due Dates, in literature very simple rules, based on average values, are implemented. However, these very simple rules can be hardly thought as the right tool for HVLV job-shops, with complex routings [107].

This work aims at fixing this issue, comparing a multivariate regression model and a Deep Learning model for building an effective forecasting system. The forecasting system, using as input the workload description as the job enters the PSP, aims at predicting the expected Lead Time (LT) of the job. This prediction will be exploited to set realistic DDs, within a scenario of balanced market power.

4.3.2. Simulation model

To test the forecasting system, a pure job-shop simulation environment was developed using SimPy©, a simulation package for Python© 3.7. The job-shop, regulated using a WLC system based on Bottleneck Load, consists of six different work centers, equally loaded, with constant capacity and 100% availability (no breakdowns are modeled). Each job can visit each machine, in any order, provided that each machine is visited only once by each job. Any machine can also be completely skipped. In order to model the variability of a typical low-volume job-shop, processing times are sampled from a truncated 2-Erlang distribution, with a maximum of 4 units of time and a mean processing time $\bar{p}$ of 1.2 units of time. Conversely,

set up times have not been considered. The customer orders generation is modeled after a Poisson process, with inter-arrival time sampled from an exponential distribution with an arrival rate λ of 1.54 units of time. Both $\bar{p}$ and λ have been arbitrarily selected to obtain an average utilization level of 90% at each workstation.

As already pointed out, the use of the WLC framework is of strikingly importance to reduce queuing time, both with respect to its expected value and the variability of it, relatively to pure push systems. Moreover, the choice of the Bottleneck Load approach gives more detailed description of the workload state of the system at any time, enabling a more informed and thus powerful forecasting system. These are key-enabling features for assuring a stable and solid forecasting system.

In the direction of further reducing variability as much as possible, it is needed to optimize the *norm* regulating the system. Being a non-trivial task, as later discussed in chapter 4.4, the norms' optimization procedure has been carried over, using a simple iterative approach. Having a balanced system with an average utilization rate of 90% for each workstation, the most sensible setting is to set each work center norm equal to the others, i.e. $N_i = N_j$, with $i \neq j$ and $i, j \in [1, 6]$. Moreover, this setting effectively reduces the number of parameters that needs to be tuned. Firstly, a 95 % confidence interval of the average throughput rate ρ [jobs/hour] of a purely push operated job-shop has been obtained through extensive simulation. The reference values obtained are as follows: *(i)* an average throughput rate equal to ρ=5585±5 [jobs per unit of time], *(ii)* an average WIP of 132 [jobs], *(iii)* an average LT of 26 [time units] and *(iv)* an average tardy of 19.5%. Next, a set of possible values for the norm *N* are sequentially tested, starting from a plausible high value and progressively reducing it until the difference between the throughput rate of the WLC system and that of the push system becomes statistically significant. For each plausible value of the norm, a simulation trial of 50 runs of 3650 units of time, with warmup, has been performed. The value obtained right before the condition described in the previous step approximates the optimal value for *N*. Thus, the optimal value found, as described in Table 5, is $N = 40$. Table 5 also reports average waiting time in the PSP (W_{PSP}) and in the shop-floor (W_{SF}), and the percentage of tardy jobs, which is obtained computing the amount of jobs that were completed after the agreed DD. The DD setting are generated simulating an allowance factor to be added to the job entry time, sampling from a uniform distribution, as in [108], ranging from

$[(n+1)\cdot p_{max}; 2\cdot(n+1)\cdot p_{max}]$, where p_{max} is the maximum processing time and *n* is the maximum number of workstation that can be visited by a job.

N	ρ	WIP	% Tardy	W_{PSP}	W_{SF}
50	5584	130	19.5%	1.26	25.35
45	5581	125	22.1%	1.98	24.87
40	5582	123	26.4%	6.17	24.68
35	5569	119	31.3%	9.30	23.66
30	5547	111	38.6%	14.76	22.16

Table 5 - Norms' levels tested

The simulation model here described is comparable with those described in literature (e.g. [108]; [107]).

A peculiar phenomenon must be noted here. As it can be seen, transitioning from a pure push system ($N = 50$) to a WLC optimized one ($N = 40$), the percentage of tardy jobs increases. As explained by [109], this is the typical behavior of an unsaturated system, which further prompt for the introduction of a reliable forecasting system for a better regulation of due dates, in order to likewise optimize the percentage of tardy jobs.

4.3.3. Forecasting models

The main approach for engineering the forecasting system, which would enable a reliable DDs setting within a WLC system based on Bottleneck Load, consists in modeling the (probably non-linear) relationship between the work-centers' workloads and the job's waiting time. More precisely, each time a job is accepted and staged at the PSP, the workload for each machine of the job's routing are measured, once for the jobs into the PSP and another for those in the shop floor. Thus, having six work centers, the available explanatory variables will be 12 in total. It must be noted that whenever a given machine does not belong to the job's routing, its corresponding workload contribution (either from the PSP or the shop floor) is set to zero. Conversely, the job's waiting time is obtained as sum of the waiting time occurred at each queue in front of the work centers.

To generate the training and test dataset, we performed 5,000 simulation runs of 3,650 units of time, with warmup period. This approach generated an entire dataset comprising the

description of workloads and corresponding waiting times of 6,717,775 different jobs. 80% of generated data were used as training set, the remaining for the test set.

As for the implementation, both the multivariate regression model and the neural network were realized using Python© 3.7 using Scikit-learn© and Keras© (with TensorFlow© as backend) libraries for machine learning and deep learning. This decision has made it possible to directly integrate the forecasting models within the simulation, also developed in Python, which was needed to control in "real-time" the definition of due dates based on the forecasted waiting time.

The multivariate linear regression model implemented is as follows:

$$Y_{j^*} = \left(\alpha_1 \sum_{j \in J_{PSP}} w_{j,1} + \cdots + \alpha_6 \sum_{j \in J_{PSP}} w_{j,6} \right) + \\ + \left(\sum_{j \in J_{SF}} w_{j,1} \ , \ldots , \sum_{j \in J_{SF}} w_{j,m} \ , \ldots , \sum_{j \in J_{SF}} w_{j,6} \right) \tag{19}$$

where the first group of sums models linearly the workloads of the jobs within the PSP when the job enters it, while the second group of sums models the workloads of the jobs active in the shop-floor when the job enters the PSP.

Relatively to the linear regression we got an $R^2 = 0.65$, with all the α and β coefficients significant with a p-value of 0.05. Also, the Root Mean Square Error (RMSE) computed on the test-set was equal to 10.48 units of time.

The Deep Neural Network proposed presents the following topology: a first input layer with 12 neurons, one for each workload, an output layer with a single neuron which computes the expected waiting time, and three hidden fully-connected layers, with 32 neurons each, using the Relu activation function [3]. Moreover, batch normalization is used after each hidden layer.

The neural network was trained using the well-known back propagation algorithm, using the Adam optimizer, i.e., an algorithm for first-order gradient-based optimization that takes advantage of adaptive estimates of lower-order moments. The training process was based on: (i) 500 epochs, (ii) early stopping with a patience of 50, (iii) validation split of 5%,

(iv) batch size of 2048. After the training, relatively to the test set, we got a RMSE of 7.2 units of time that, as expected, is better than that of the linear regression. This fact confirms a non-linear dependence between response and explanatory variables.

4.3.4. Results

In order to test the multivariate regression model and the neural network forecasting model, both have been integrated in the simulation environment. The integration is done at the stage of the job entering the PSP. During this step, the Lead Time of incoming jobs is estimated, and this estimate is leveraged to generate DDs, accordingly. Moreover, to simulate more realistic scenarios, we modeled the possibility for the company to negotiate the terms with the customer, whenever the proposed customer DDs reveal to be probably too close. The negotiation process is modeled as an equal market power scenario. More precisely, as the customers' Due Date is modeled as described in the previous section, the predicted job's waiting time, coupled with its processing times, is compared to the customer DD and checked for feasibility. Obviously, if the customer DD is later than the predicted LT, then it is accepted as is. Otherwise, a corrected DD, which represents the process of negotiation between the customer and the firm, is generated randomly along the range given by [DD; *min*(1.2DD; LT)]. The allowance factor 1.2 models the customer availability. As benchmark, we use the simple average formula used by [110], which models the job's lead time as a function of the average waiting time within the PSP, $\hat{q}_{j^*,P}$, of the job's routing length k_{j^*}, and of the average throughput time at each workstation, $\bar{q}$, as follows:

$$LT_{j^*} = \hat{q}_{j^*,PSP} + k_{j^*}\bar{q} \qquad (20)$$

Thanks to the stabilization of the throughput time of a WLC system, $\bar{q}$ can be simply calculated as the average of the throughput time at each machine, which in this simulation is equal to 8 units of time. However, $\hat{q}_{j^*,PSP}$ must be estimated considering the ongoing situation at the PSP, meaning that $\hat{q}_{j^*,PSP}$ represents the average waiting time before all workloads fall below the *norm*, as experienced by a job waiting in the PSP. $\hat{q}_{j^*,PSP}$ is computed as follows:

$$\hat{q}_{j^*,PSP} = \max_{m \in R_{j^*}} \left(W_m + \sum_{j \in J_{PSP}} w_j^m - N \right) \tag{21}$$

Moreover, an additional benchmark is considered, where no forecasting is done at all and DDs are negotiated as soon as they are proposed by the customer, sampling from the range [DD; 1.2DD]. This approach has been applied to both the push system and the WLC system. Results are shown in Table 6, which displays the average percentage of tardy jobs for each one of the 5 tested scenarios. Results are statistically confirmed by ANOVA and Bonferroni post hoc test with a p-level of 0.05. The model by [110] is the worst performing, even worse than the base line WLC scenario, which, by no surprise, in turn is worse than the push base line scenario. The surprising result from the thirds scenario could be in part explained given that the assumptions of the underlying formula for the DD generation are over optimistic, thus generating too tight estimates of the LT. This leads to no negotiation of the DD, which are in turn often violated.

Finally, both the linear regression and the neural network forecasting methods allow for the best approach for setting realistic delivery dates in high manufacturing systems, with the neural network powered scenario outperforming by a large margin the linear regression scenario. This could be given by the fact that the neural network is able to recognize and successfully exploit the non-linear relationship between the explanatory variables, work centers' workload, and the response variable, the jobs' waiting time.

Exploiting these two scenarios, it is possible to further enhance the WLC framework potentials, given that is now possible to optimize both the tardiness and the WIP of the systems, and overall lowering operating costs.

	Push Base	**WLC Base**	**WLC Land**	**WLC Regr.**	**WLC N.N.**
% Tardy	17.5%	22%	23.5%	13%	9.5%

Table 6 – Final Results

4.3.5. Conclusions

This work showed the enhancements of a WLC system obtained using a forecasting system based on statistical and Deep Learning methodologies to set reliable delivery dates, based on the system workload state. Indeed, as a WLC managed shop floor shows optimized WIP, on the other hand, almost paradoxically, tardiness is increased, which is due to higher waiting time in the PSP causing higher LT than those of a push system.

At an implementation level, the proposed approach can be rather uncomplicated and cheap to deploy in real industrial applications, using open source software, as done in the simulation approach, coupled with existing MES applications, prerequisite of any WLC system and further promoted by Industry 4.0 oriented initiatives.

4.4. WLC with shifting bottlenecks: norms optimization through Design of Experiments

4.4.1. Introduction

As stated in the previous section, the optimization procedure of *norms* regulating a WLC system can be rather troublesome. While in the previous work, as well as in technical literature, the shop floor model is simplified, the present section deals with a more realistic scenario, in which the assumption of highly balanced systems, and thus of equally characterized norms, is abandoned in favor of a more reasonable and convoluted scenario in which the active bottleneck vary in time between different machines. Moreover, to further enable a more realistic scenario, the incoming jobs are differentiated in three families, each with its own processing time distribution, so to better model a High-Variety-Low-Volume (HVLV) production environment. This section describes an innovative approach to tackle the norm optimization problem which has the objective to introduce a practical and general approach for unbalanced systems. Indeed, practical evidences have shown that machines with different utilization levels are a common operating condition ([111]; [112]), and that WLC can be effectively used also in case of unbalanced manufacturing systems ([113]; [114]; [115]). Bottleneck shiftiness, which is defined as either the dynamic change within the routing sequence or a physical change in location (Craighead et al., 2001; Fredendall et al., 2010), requires to drop the over simplistic approach to use the same norm for each work center, in order to avoid high deterioration of system performance. Indeed, recent literature has become more and more aware of the compelling urgency to address the problem. However, it has mainly worked around the optimization problem. For example, [114] and [116] studied a 7-machines flow-shop characterized by one bottleneck machine, shifting from machine 1 to machine 7, while [117] considered a 6-machines general flow-shop, where the second machine acts as the long run bottleneck. Both drawn identical conclusions, arguing that bottleneck shiftiness has more severe impacts when it occurs downstream. However, none of them focused on the norms' optimization problem. Only two works handle the optimization problem. The first one, [118], adopts a commercial optimization package (OptQuest©) coupled with a simulation developed using Simul8©, to model a shop with 12 work centers. As norms were iteratively fine-tuned, authors demonstrated that in case of high unbalance, it

is mandatory to use different norms levels, in response to different utilization rate. The second work, by [112], modeling a 6-machine job-shop with random routing sequences and two potential bottlenecks, used the following procedure to level the system WIP: two norms' levels N_B and N_{NB} are used to control bottleneck and non-bottleneck machines, respectively. As the norms are optimized in a straightforward fashion, iteratively starting from a high value and lowering it in a stepwise fashion, the ratio of the two norms $N_R = N_B / N_{NB}$ is kept constant. The conclusion of this study concerns the fact that within scenarios with high bottleneck shiftiness and low protective capacity, both bottlenecks and non-bottlenecks machines should be controlled, with the latter's norms tighter than for the former.

The two study findings could be summarized as follows: norms should be different and should be determined as a function of actual workload of work centers, and all norms' levels can interact with each other influencing the overall system performance. As a consequence, norms must be jointly optimized. These findings, however, highlight and stress the absence of a practical procedure that could be successfully used on the field, as one work relies on a commercial software whose internal logic is unknown, and the other requires a cumbersome lengthy iterative procedure, worsened as the number of potential bottlenecks increase.

The proposed novel method for integrated norms optimization leverages two statistical method, namely Design-Of-Experiments (DOE) and Response Surfaces Method (RSM), which makes it possible to explore and highlight interdependences between norms' levels and one or more response variable, such as WIP level, percentage of tardiness, lateness, or total throughput time. The suggested technique can be successfully exploited both by researches and practitioners, further enabling industry adoption of WLC, especially in Industry 4.0 contexts.

4.4.2. Simulation model

As in the previous section, a general flow shop simulation model was developed using the Python© 3.7 open source package for discrete event simulations called SimPy©. The simulated shop comprises six sequentially arranged work centers, and each machine is modeled as a single resource with constant capacity. As most of the machines are non-crucial, designed with an 80% of average utilization level, two work centers, the third (M3) and the sixth (M6) are modeled as potential bottlenecks, in order to obtain the desired two bottleneck

shiftiness dynamic. The two potential bottlenecks have a utilization rate of 92% and 95%, respectively. The design of two bottlenecks shiftiness is motivated by the fact that from a practical industrial point of view it is difficult to find more than two bottlenecks, as stated by [119]. Nonetheless, the choice of restricting the number of potential bottlenecks to two aids the make more understandable the result of the design of experiment procedure. Moreover, the supposedly main bottleneck is at the bottom of the flow-shop, which effectively restrict the simulation to model only upstream bottleneck shiftiness. This design decision was made to effectively comply with the indication that upstream shifts pose more severe challenges than downstream ones. While the utilization level differs from bottleneck and non-bottleneck, the coefficient of variation of the utilization level, namely the ratio between the average utilization level and the standard deviation of it, is the same for all work centers, equal to 0.2. This setup allows to achieve the desired protective capacity for non-bottlenecks machines and a very high-frequency bottleneck shiftiness between M3 and M6, meaning that at any time anyone of the two machines can be the current bottleneck influencing the job-shop.

Order entry is modeled as a Poisson process, with inter-arrival time sampled from an exponential distribution with an arrival rate λ of 0.295. Each generated job can belong to three different families, namely F1, F2, and F3, with a probability of 0.5, 0.3, and 0.2, respectively. The family of which jobs belong defines both the processing times and the routings. Within each family, processing times are identically distributed and modeled as truncated k-Erlang distributions, with shape k and scale parameter β dependent on the family, but not on the machine, to comply with other works found in literature ([111], [112], [114]). Table 7 shows parameters values as a function of each family.

	k	β	μ	CV
F_1	11	0.4	4.40	0.30
F_2	6	0.4	2.40	0.40
F_3	4	0.4	1.60	0.50

Table 7 – Parameters' settings for each family

The parameters of the three k-Erlang distributions (one for each family) and that of the inter-arrival rate (λ) of the orders were chosen to obtain the aforementioned utilization levels, both at the bottlenecks and non-bottlenecks machines.

Routings are obtained as defined in Table 8, in which, for each family, the probabilities $p_{f,m}$ that each machine would be part of the job routing belonging to a given family are shown.

	F₁ (50%)	***F₂ (30%)***	***F₃ (20%)***
M1	$p_{1,1} = 1.0$	$p_{2,1} = 0.5$	$p_{3,1} = 0.40$
M2	$p_{1,2} = 1.0$	$p_{2,2} = 0.5$	$p_{3,2} = 0.40$
M3	$p_{1,3} = 1.0$	$p_{2,3} = 0.9$	$p_{3,3} = 0.85$
M4	$p_{1,4} = 1.0$	$p_{2,4} = 0.5$	$p_{3,4} = 0.40$
M5	$p_{1,5} = 1.0$	$p_{2,5} = 0.5$	$p_{3,5} = 0.40$
M6	$p_{1,6} = 1.0$	$p_{2,6} = 1.0$	$p_{3,6} = 1.0$

Table 8 - Routings Probabilities

As a result of the probabilities shown in Table 8, the study models a general flow shop system, in which jobs may visit less than six machines, routing can skip altogether some machines, and the first machine in the job routing could not correspond to the first machine in the job-shop.

The settings described here define a pure push flow shop with a degree of bottleneck shiftiness $\gamma = 0.297$, as in formula (22) [120]:

$$\gamma = 1 - \frac{CV}{\sqrt{M}} \tag{22}$$

in which M is the number of work centers and CV is the coefficient of variation of bottleneck probabilities, which is defined as the long-run proportion of time a certain work center has a lengthier queue than any other work center. Please note that, for a 6-machine flow shop, as the one here described, with two shifting bottlenecks, the maximum value of bottleneck shiftiness equals 0.368. Thus, the job-shop here simulated has a degree of bottleneck shiftiness at around 80% of the maximum.

The procedure which has been used to fine tune the reported parameters is based on the following set of equations. Considering a single machine which processes jobs belonging to *n* different families, the workload contribution *X* of a single job is a random variable with probability density function *f(x)* obtained as a finite mixture distribution, based on the

probabilities w_i of a job belonging to each family i, and the corresponding processing times probabilities distribution $p_i(x)$, as in eq (23):

$$f(x) = \sum_i w_i \cdot p_i(x) \tag{23}$$

The mean and variance can thus be computed as follows ([121]):

$$\mu = E[X] = \sum_i w_i \mu_i \tag{24}$$

$$\sigma^2 = E[(X-\mu)^2] = \int_{-\infty}^{+\infty} (X-\mu)^2 \cdot \sum_i w_i \cdot p_i(x) = \tag{25}$$

$$= \sum_i w_i \cdot \int_{-\infty}^{+\infty} (X_i - \mu_i)^2 \cdot p_i(x)$$

$$= \sum_i w_i \cdot E[(X_i - \mu_i)^2] =$$

$$= \sum_i w_i \cdot ((\mu_i - \mu)^2 + \sigma_i^2) =$$

$$= \sum_i w_i \cdot (\mu_i^2 + \sigma_i^2) - \mu^2$$

Where μ_i and σ_i^2 are the mean and the variance of $p_i(x)$, respectively.

It follows that the linkage between the workload contribution X and the total workload $W_{\Delta t}$ observed by the single machine in the time interval Δt is given by:

$$f(w_{\Delta t}) = \sum_k \gamma_k \cdot f_k(x) \tag{26}$$

Where· $f_k(x)$ is the workload contribution due to k jobs and γ_k is the probability that exactly k jobs enter the system during the interval Δt.

Note that, since the workload contribution due to k jobs is the sum of the workload contribution of each single job, $f_k(x)$ is the k-fold convolution of $f(x)$, with mean $k\mu$ and

variance $k\sigma^2$. Also, if the jobs' interarrival times are exponentially distributed with parameter λ, γ_k can be computed as in Eq. (27):

$$\gamma_k = \frac{ke^{-\lambda\Delta t}}{k!} \tag{27}$$

As Eq. (26) shows, also $f(w_{\Delta t})$ is a mixture distribution. So, by replacing w_i with γ_k, μ_i with k, and σ_i^2 with $k\sigma^2$, both in Eq. (24) and in Eq. (25), we can finally write that:

$$E[W_{\Delta t}] = \sum_k \gamma_k k\mu = \mu \cdot \sum_k \gamma_k k = \mu\lambda\Delta t \tag{28}$$

$$\begin{aligned} Var[W_{\Delta t}] &= \sum_k \gamma_k k^2\mu^2 + \sum_k \gamma_k\, k\sigma^2 + (\mu\lambda\Delta t)^2 \\ &= \mu^2 \cdot (\lambda\Delta t + (\lambda\Delta t)^2) + \sigma^2\lambda\Delta t - \mu^2 \cdot (\lambda\Delta t)^2 \\ &= \lambda\Delta t \cdot (\mu^2 + \sigma^2) \end{aligned} \tag{29}$$

In this way, through equations 23 to 29, we have defined a set of relations that link processing times $p_i(x)$, productive mix, (i.e., the probabilities w_i) and the job's interarrival rate λ, with the mean and the variance of the workload of the machine.

Given the high protective capacity of M1 and M2, they have little influence on the potential bottleneck M3. Thus, M3 can be considered as a single machine. On the other hand, M6 is both the last machine in the job shop and the potential bottleneck with the highest utilization rate, subjected to downstream shiftiness.

4.4.3. Norm's optimization procedure

4.4.3.1. Overall procedure

Generally, studies on WLC focus on optimization of the shop floor with regards to tardiness, total Lead time, or gross throughput time, which considers also the time spent by the job in the PSP. However, as the production capacity of the flow shop here simulated is under saturated (the most downstream bottleneck has an average utilization level of 95%), performance of WLC, measured in terms of lateness or of Lead Time, are comparable to those

of a purely push system, either regulated by EDD or SPT dispatching rules [109]. Moreover, given the famous Little's Law [122], if the optimization objective is the reduction of WIP, the resulting reduction of total throughput time can be much more substantial [109].

Indeed, the optimization goal of the present study is to minimize the overall WIP in the shop floor, within the constraint of keeping the average throughput time at its maximum level, as measured in a pure shop floor. In this case the maximum throughput rate coincides, exactly, with the job's arrival rate $\lambda = 0.295$ [jobs per unit of time].

As the problem at hand has two free parameters to be finetuned, the norms' levels for Machine 3 and Machine 6, a single optimization criteria, the overall WIP, and a single constraint, the throughput rate, the present study implemented a joint procedure consisting of a Design Of Experiments [123] coupled with Response Surface Methodology and the Steepest Descent algorithm. Similarly to what was made by [124] and by [125] for a robust optimization (via DOE and simulation) of industrial problems in the field of production management, this procedure allows to avoid costly blind trial-and-error search of the optimum, or worse exhaustive search. Moreover, as the techniques are based on solid statistic theory, statistical inference concerning possible interaction effects among the norms.

The procedure can be roughly summarized as follows:

1. Perform a preliminary analysis over the entire search space where the optimal solution can lie;
2. Initialize the problem choosing a starting point, and construct a suitable DOE (i.e., a suitable grid of points to be tested) in a restricted range around it;
3. Perform experiments and test results to see if, on this restricted area, they are satisfactorily described by a first-order surface or by a second-order surface;
4. Estimate the parameters of the corresponding first or second order surface;
5. Apply the steepest descent gradient technique to identify a search path leading to the minimum.
6. Follow this direction to select a new center for an updated DOE;
7. Repeat points 3 to 5 until the constraint (on the maximum throughput rate) is not violated or a stationary point has been reached.

In doing so, we considered experiments with two factors, the norms' levels at M3 and M6, evaluated at two levels (high and low), for a total of 2^2 combinations, each one repeated n = 100 times. A centre point, corresponding to the mean level between high and low levels of both factors, was also considered, as this allowed us to assess the opportunity to fit data with a second order polynomial and to consider quadratic effects, too. Furthermore, accordingly to the center composite design method [126], any time a second order polynomial was needed, aiming to get a better fit, four additional points, located on the circumcircle of the square formed by the original factorial points, were also considered.

4.4.3.2. Preliminary study

As already introduced, the proposed approach requires to get preliminary insight of the functional relationship between norms and WIP over a large search space, following a sparse and rather scattered grid of parameters' combinations. Hence, to analyze the relationship between norms and WIP, 5 evenly spaced points were needed to generate a grid of 5^5 different combination of the norms. The range definition for each norm required to find a minimum and a maximum value for each norm level. The minimum value was found considering the lowest possible setting for each norm, as if the system was controlled using a single norm approach, before the average throughput rate would drop below the given constraint. The maximum value was found as the average workloads at M3 and M6, when the system operates in a purely push way.

Hence, the values considered for the preliminary analysis are the following:

- $WC3 \in [17, 45]$
- $WC6 \in [30, 90]$

Where $WC3$ and $WC6$ denote the Workload Caps (i.e., the norms) at machine M3 and M6, respectively.

5 evenly spaced points were chosen both for *WC*3 {17, 24, 31, 38, 45} and for *WC*6 {30, 45, 60, 75, 90}, to generate a grid of 5^5 different combination of the norms. For each norm combination, the flow-shop was simulated using runs of 4000 units of time (of which 500 were used as warm up, enough to reach a steady state condition), with *n* = 100 repetitions. To visually convey the relationship between norms on the overall WIP, a 3D-scatterplot of

WIP versus *WC*3 and *WC*6, and the interaction graph of the mean WIP on the *n* repetitions are shown in Figure 11.

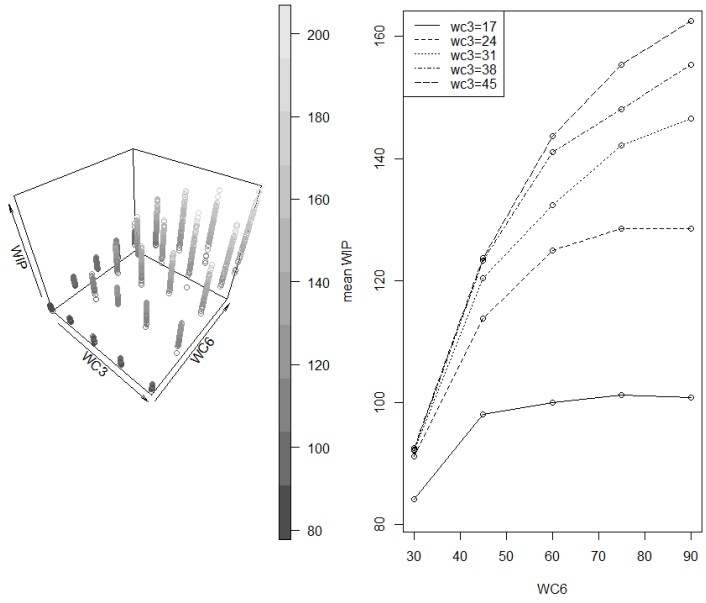


Figure 11 - 3D-scatteplot and interaction graph of WIP relatively to WC3 and WC6

A couple of intermediate results can be already pointed out. The interaction graph clearly shows Little's Law dynamics, as the overall WIP decreases non-linearly as norms are tightened. Moreover, a high level of heteroskedasticity can be retraced in the increasing dispersion of the simulated WIP, as *WC*3 and *WC*6 increase. Such heteroskedasticity cannot be overcome, as it is inherent to the nature of the problem. Indeed, as the tested combinations span a wide range of operating conditions, variability is measured as the preliminary analysis basically compares purely push system operating conditions and, vice versa, very tightened operating conditions with very constrained parameters. Truly, variability will be higher in the first case, while it is significantly reduced in the second one.

Fitting the 5^5 points with a second order model, a good coefficient of determination R^2 = 0.83 was obtained; also, all the terms of the model, including the interaction one, were highly significative, as clearly shown by the *F* values of Table 9.

Response WIP	Df	Sum Squares	Mean Squares	F value	Pr(>F)
FO(WC_3, WC_6)	2	1,094,135	547,067	5,047.512	<2.2e-16
TWI(WC_3, WC_6)	1	110,053	110,053	1,015.4	<2.2e-16
PQ(WC_3, WC_6)	2	138,426	69,213	638.594	<2.2e-16
Residual	2,494	270,309	108		
Lack of fit	19	19,913	1,048	10.359	<2.2e-16
Pure error	2,475	250,395	101		

First Order (FO), Two-Way Interaction (TWI), Pure Quadratic (PQ)

Table 9 - Analysis of Variance Table for the second order model with interaction

However, given a lack-of-fit which is significantly non null (p-value < 0.01), the preliminary 5^5 grid cannot be used to estimate the relationship between *WC*3, *WC*6, and the overall WIP. Moreover, heteroskedasticity is confirmed, as shown in the residual plot in Figure 12

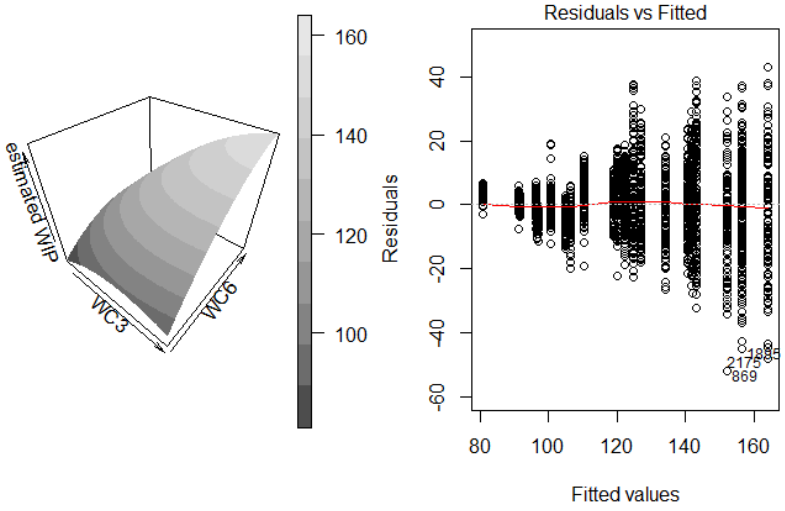


Figure 12 - Estimated second-order model and residuals versus WIP's fitted values

4.4.3.3. Gradient descent

Given the issues shown in the first preliminary analysis, a recursive approach based on the Steepest Descent algorithm is needed. Starting from a restricted region relative to the one studied in the preliminary phase, a 2^2 DOE will be performed, and the response surface is estimated. Thus, the steepest gradient descent algorithm will provide the direction in which to move to proceed recursively with the analysis. The procedure will halt as soon as the regulating throughput constraint is violated. In more detail, the iterative procedure will, at each step, perform a new DOE, such that the center point is located on the path given by the steepest descent obtained from the RSM obtained at the previous step. Moreover, the 4 points laying on the rectangle are chosen so that the search region is half-overlapped with the previous one, as in Figure 13. Finally, a first-order model is estimated and in case of poor lack-of-fit, axial points are added and a second-order model is estimated.

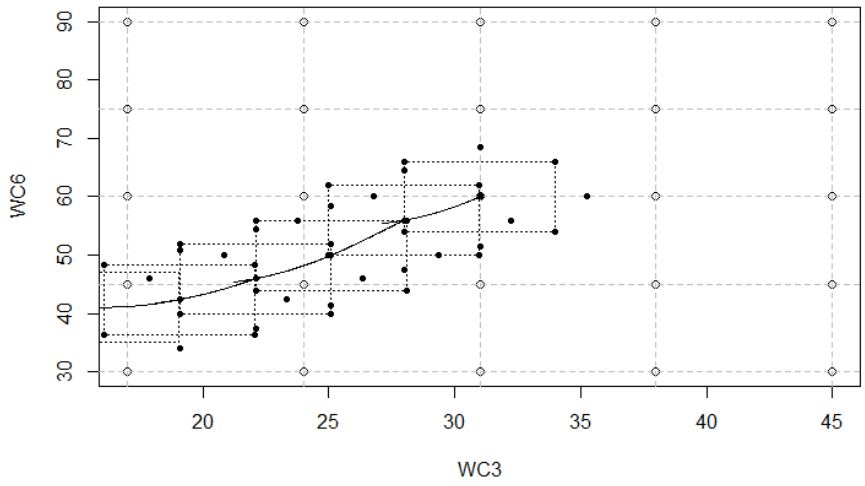


Figure 13 - Iterative steepest descent gradient path

The iterative approach is initialized using a 2^2 design, centered on (31, 60), in the middle of the original grid. The height and width of the design have been chosen to avoid both the heteroskedasticity issue and to reveal possible interactions between parameters. In consideration of these requirements, the starting design of experiment consisted in these following factorial points: (28, 54), (28, 66), (34, 54), and (34, 66), plus the center point (31, 60). Each configuration was simulated n = 100 times, and corresponding WIP results were

recorded. Moreover, the center point was simulated 4 times the other points, in order to estimate the presence of quadratic effects, as reported by [126].

Indeed, as a common decision, as the design makes use of 2 factors each evaluated using 2 possible levels (high and low), repeated *n* times, if the aim is to investigate whether quadratic effects could exist and a second order polynomial model could be required, then the following statistic should be calculated and an F-test should be performed:

$$SS_Q = \frac{n_f n_c}{n(n_f + n_c)} \left[\bar{y}_f - \bar{y}_c\right]^2 \tag{30}$$

where: n_f and n_c is the number of factorial and center points, n is the number of replications of the experiment in every point, $\bar{y}_f$ and $\bar{y}_c$ are the mean response on the factorial and on the center points.

If SS_Q is significatively not null, quadratic effects should be considered and the first-order model must be enlarged to be suitable for the estimation of a second order polynomial. To this aim one can use the so-called center composite design, obtained by adding four additional points located on the circumcircle of the square formed by the original factorial points.

The DOE described above was employed to estimate parameters of a first-order surface, as shown in Table 10 and in Table 11:

	Estimate	**Std. Error**	**t-value**	**Pr(>|t|)**
Intercept	134	0.35	381.4	< 2.20e-16
WC3	4.35	0.496	8.75	< 2.20e-16
WC6	3.10	0.496	6.25	6.74e-10
R^2 = 0.127				

Table 10 - Linear surface estimated on the central standard first order DOE

Response WIP	Df	Sum Squares	Mean Squares	F value	Pr(>F)
FO(WC_3, WC_6)	2	11430	5715.2	57.87	<2.2e-16
Residual	779	78709	98.8		
Lack of fit	2	860	430	4.39	0.0127
Pure error	795	77849	97.9		

Table 11 - Analysis of Variance Table for the first order model

As it can be seen, the determination coefficient R^2 is not adequate, as well as the lack-of-fit, which is not statistically null (with a p-value of 0.0127). As described just above, we fitted a second-order model, using a central composite design with 4 additional points and 8 repetitions of the center point. Results are shown in Table 12 and Table 13.

	Estimate	Std. Error	t-value	Pr(>|t|)
Intercept	134.62	0.358	375.71	< 2.20e-16
WC3	4.01	0.358	11.19	< 2.20e-16
WC6	3.85	0.358	10.75	< 2.20e-16
WC3xWC6	0.73	0.51	1.44	0.15
$(WC3)^2$	-1.05	0.358	-2.94	0.0033
$(WC6)^2$	-0.48	0.358	-1.34	0.18
$R^2 = 0.137$				

Table 12 - Quadratic surface estimated on the central composite second order DOE

Response WIP	Df	Sum Squares	Mean Squares	F value	Pr(>F)
FO(WC_3, WC_6)	2	24758	12378.9	120.53	<2.2e-16
TWI(WC_3, WC_6)	1	213	213.2	2.07	0.148
PQ(WC_3, WC_6)	2	1074	536.9	5.23	0.005
Residual	1594	163708	102.7		
Lack of fit	3	543	181.0	1.76	0.15
Pure error	1591	163165	102.6		

Table 13 - Analysis of Variance Table for the second order model

However, only the lack-of-fit were statistically null, while the R^2 is only slightly increased. Nonetheless, as the objective of this step in the optimization procedure is not to find the perfect model to describe the relationship between factors and response variable, but rather to obtain a first indication of the direction to follow to further investigate the search space, the obtained results are satisfactory enough to be used as input to the steepest gradient descent algorithm. Indeed, as it will described, the fitting will improve along the search path, as an indication of a relationship of increasingly second-order nature.

4.4.4. Results

Obtained results are summarized in Table 14.

Step	Centre point	Order	R^2	*p*-value
1	(31, 60)	Second	0.1345	0.1519
2	(27.979, 55.974)	Second	0.1988	0.4178
3	(25.093, 49.956)	Second	0.3114	0.3891
4	(22.090, 45.978)	Second	0.4701	0.1782
5	(19.066, 42.408)	Second	0.7948	0.0002

Table 14 - The obtained gradient descent research path

The coefficient of determination increases rather dramatically as the steepest descent guides the exploration. This further strengthens the procedure, as the first results were only needed to kickstart the steepest descent, and thus the poor fitting of the fitted model were not so problematic in this regard.

Moreover, to find the optimal combination of norms, 100 simulation runs of the flow-shop were performed at 116 equally spaced WLC operating settings laying on the search path retraced by the five consecutive response surfaces obtained at each step. As it can be seen in Figure 14, the predicted WIP using the 5 response surfaces found at each step of the gradient descent arguably perfectly fit the average simulated WIP.

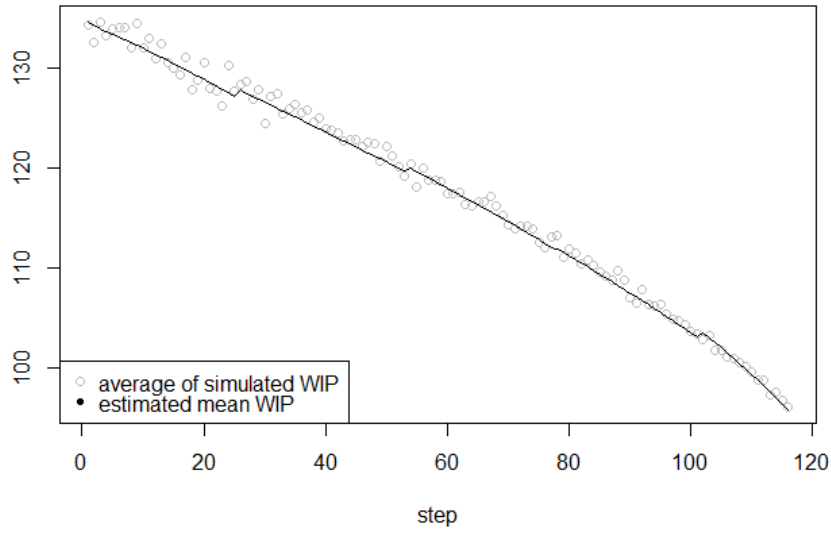


Figure 14 - Simulated and estimated WIP, via recursively updated second-order RSM

As already mentioned, the search for the optimal WIP is constrained to respect the same throughput rate of a pure flow-shop. As a result, along the steepest descent path the search was halted as soon as the constraint were violated, and the last conforming step is selected as the optimum point. More precisely, the 95th step correspond to the combination of norms given by {WC3 = 20; WC6 = 43}, which lays on the DOE corresponding to the 4th step. As the lack-of-fit is significantly high, the point considered the required optimal point, and the optimization procedure comes to a halt.

To validate the obtained result, a comparison with four classical WLC configuration has been carried over. The same WLC configuration that was used for the DOE and RSM procedure was implemented for the benchmark configuration, namely the periodic review, with aggregate workload with continuous update and Earliest Due Date dispatching rule. The four benchmark configurations progressively alter both the number of parameters to be tuned and the location of the bottleneck. Each benchmark configuration was optimized in a stepwise iterative fashion, starting from a high level of norms and progressively reducing it until the average throughput constraint is met. In the fourth configuration, the procedure by [112] has been implemented, iteratively testing different norms' levels with respect to a fixed norm ratio NR = (WC3/WC6), with steps of size 0.1, moving from NR = 0.2 to NR = 2.

Also, to make a fair comparison, we randomly generated 500 jobs sequences, each one containing about 1180 jobs, created using the same interarrival rate $\lambda = 0.295$ and the same

rules, both for family and routing assignment, described in chapter 4.3. Next, for each generated job sequence, we computed and compared the performance of the five investigated WLC configurations. Table 15 reports the benchmark configurations, while Table 16 and Table 17 report the obtained results.

#	Configuration	*WC3*	*WC6*
1.	Single norm at the main bottleneck	[-]	40
2.	Single norm at the first bottleneck	20	[-]
3.	Two equal norms, one for each bottleneck	40	40
4.	Two different norms, one for each bottleneck	20	50

Table 15 - Benchmark configurations

Groups	*WIP Mean*	*WIP Variance*
1. WC6 40	117.76	35.97
2. WC3 20	115.18	28.74
3. WC3 40 - WC6 40	115.13	28.47
4. WC3 20 - WC6 50	110.36	32.01
5. WC3 20 - WC6 43	106.25	19.80

Table 16 - Results of different configurations

Source of Variation	*SS*	*Df*	*MS*	*F*	*P-Value*	*F crit.*
Between Groups	42040.93	4	10510.23	206.07	6.88e -153	2.375
Within Groups	127254.86	2495	51.00			
Total	169295.80	2499				

WLC configuration is the single factor, WIP is the response variable

Table 17 - ANOVA results of comparison

A post-hoc test was made to further examine the results obtained, as the ANOVA test is significant. The test objective was to highlight for which pair of WLC configuration the average WIP significantly differs. Dunnett's test was employed, to compare the DOE-RSM solution with the benchmark solutions, since it compares the means of several experimental group against the mean of a control group [127].

Briefly, the test is based on the following statistic:

$$D = t_{\alpha,k,d} \cdot \sqrt{\frac{2 \cdot MS_{wg}}{n}} \tag{31}$$

where: $t_{\alpha,k,d}$ is the Dunnett critical value at level α (with k groups and d-degree of freedoms of the within group output of the ANOVA table), MS_{wg} is the within group mean square of the ANOVA analysis, and n is the sample size.

Table 18 shows, for each WLC configuration, the difference between the mean of each group with respect to the mean of the control group. If the difference exceeds a critical value, then it is stated as significant at level α. In the presented case, $d \approx \infty$, k = 5, and using α = 0.01 we have that $t_{0.01,5,\infty} = 2.84$. So, since MS_{wg} = 51, the critical distance D equals 1.283.

WLC Configuration	Difference
1. WC6 40	8.9
2. WC3 20	11.5
3. WC3 40 - WC6 40	8.8
4. WC3 20 - WC6 50	4.1

Table 18 - Difference in means w.r.t control group

Results showed in Table 18 confirm the primacy of the proposed approach, as all differences exceed the threshold. These results confirm that in case of strongly unbalanced systems, neither the use of a single norms nor the use of the same norms at each bottleneck is enough to push the system toward its optimal operating condition. Moreover, the strong improvement obtained with respect to the optimization procedure based on the fixed norm ratio NR, demonstrates how a careful optimization allows achieving higher benefits.

4.4.5. Conclusions

This work proposed an optimization framework for WLC system based on Design of Experiment, Response Surface Methodology, and Steepest Descent. The optimization objective is to minimize the overall WIP, adjusting norms' levels in a flow-shop with shifting bottlenecks, which makes the problem very difficult to deal with operationally. Specifically, we considered a general flow-shop made of 6 machines, with two potential bottlenecks and a high bottleneck shiftiness. To control the system, we used two norms (one for each potential

bottleneck) and we optimized their level to minimize the average WIP, keeping the average throughput rate at its maximum level. The optimal norms' combination was obtained performing five iterations of the gradient descent algorithm and as many estimates of second-order response surfaces. Furthermore, the throughput rate constraint is considered via simulation, meaning that at each point along the search path, the average throughput rate is computed and verified against the constraint. To fully automate the procedure, the throughput rate, as done for the overall WIP, should be modeled as a non-linear function of the WIP, reformulating the gradient descent procedure as a constrained optimization model. Moreover, as optimization was limited to WIP minimization, other criteria could be taken into consideration, effectively posing the optimization problem as a multi-objective one.

5. Demand forecasting framework for the Fashion Industry: a case study.

5.1. Introduction

Given recent trends, as discussed in Section 2.2, many companies feel the need to integrate Machine Learning and Deep Learning algorithms into their production and decision-making processes. Nevertheless, many enterprises are aware of the lack of know-how within the company itself needed to develop and effectively deploy solutions based on Artificial Intelligence. Nonetheless, they recognize that such solutions, once successfully deployed, could give them competitive advantages over the competition. Such competitive advantage could stem from the exploitation of knowledge buried within their data-warehouses [19], for example building forecasting systems, which leverage historical data, to predict future sales and manage production and procurement processes, accordingly. These are the main reasons that induced a renowned Italian fashion company to seek to undertake a research project with the aim of updating and improving their demand forecasting system. The objective is two-fold: to verify if a new demand forecasting system based on Machine Learning is possible, and to disseminate theoretical and practical knowledge.

While operating with a Make-To-Order logic, the need for a new demand forecasting system emerged to study the possibility of improving the current forecasting system of sales orders, to anticipate as much as possible the supply of raw materials.

In fact, the company operates with a sales campaign logic, which takes place twice per year. After a first phase of design of the new collection, the company determines a time window, during which business customers can review the collection and place orders accordingly. Orders are collected as soon as the client send them and can be cancelled or modified at any time during the sales campaign. Only at the end of the sales campaign final quantities to be sold are computed for each product. Indeed, the company works accordingly to an MTO dynamic. Nonetheless, the company is interested in forecasting final quantities for each product as soon as possible and as best as possible.

This would give a strategic advantage to the company, increasing the guarantee of a timely delivery of the finished product at the retailers. Indeed, anticipating as much as possible raw materials' supply, to streamline the production planning and scheduling, is a

critical success factor. Hence, an accurate forecasting system is strongly required. For such purpose, a more evolved system of orders forecasting is proposed, which integrates historical data with the real-time data of the ongoing sales campaign.

We stress that, one of the main application fields of Machine Learning, and Neural Networks in particular, is demand forecast, which is even more the case within the fashion retail ecosystem, where a high demand uncertainty creates many hurdles, for an efficient logistics management [128]. In literature there are many works that faced the problem of forecasting demand in the fashion sector where, to cope with fashion trend and market response, demand is highly variable, erratic and subjected to unpredictable peaks [129]. However, as reported by Gutierrez [130], traditional statistical demand forecasting systems do not always capture non-linear patterns present in historical data of fashion products.

It must be noted that, in contrast to usual problem tackled in literature [131], which focused on the problem of forecasting demand at the retail level, that is the demand of end consumers, the aim is to predict the final quantity sold to business customers, at the end of a sales campaign.

The work, divided into three phases, is hindered by many technical and theoretical issues, here listed:

- **Product similarities** - The main issue is that, as a fashion industry, each year the products offering changes a lot to respond to fashion trends: this leads to a lack of time-series, since no product is sold for more than one year. The first step of the work is then about the analysis of products and the conception of a similarity model between products of different years: this will be exploited to retrace chains of similarities to build, for each new product never sold before, pseudo-time series which will be used for the forecasting model.
- **Core customer selection** - The second phase deals with customer base analysis, which is needed to find homogeneous clusters of customers with the aim of filtering out some “noise”. The noise is reduced identifying the most representative customers for each product class. This core subset of customers will be leveraged both for the analysis of historical data and during the currently ongoing sales campaign.

- **Forecasting model** - The third stage, which takes advantage of the results of the previous package, presents the demand forecasting system. The model, a complex neural network architecture based on a Recurrent Neural Network module, analyzes reconstructed pseudo-time series, together with current ongoing sales campaign data, to produce the final forecast. The following sub-sections will describe each stage in more details.

5.2. Products' similarities recovering methods.

The first phase of the work presents three different methodologies aimed at measuring similarities between products from different years based on historical purchase behavior. These models are needed because, pushed by ever-changing market trends, the company at study creates different products every year, either modifying or removing old ones, or introducing completely new apparel. This implies that new collections are always composed of completely new products, for which a time-series of sales is consequently not available. This is obviously a very impeding issue, for which no statistical techniques nor Machine Learning algorithms can be successfully employed to generate accurate forecasts. It is true, on the other hand, that human experts are able to trace, based on physical appearance, similarities between products offerings which span many years. These similarities are then processed in some way to produce a forecast based on human intuition, experience, and other commercial reasonings. The main hypothesis, then, is that physically similar looking products exhibit similar sales patterns. Thus, these similarities can be exploited to build *pseudo* time-series for each current product, based on past sales of those most similar products.

The first methodology is based on a two-step algorithm based on partitioning rules and weighted distances. This methodology has been developed to act as benchmark reference for the remaining two methodologies, as it is rule-based, fixed, and impose a metric in the product space that is based only on physical attributes and is not learned from data.

The second methodology is based on a Deep Learning model called Autoencoder, applied to the products' technical drawing. Technical drawings are flat, detailed sketches of just the garment, with all of the details. It is generally accompanied by specifications which provides measurements, construction details, fabric and trim information. As more precisely

described below, the Autoencoder maps each product, compared to its technical drawing, into a product space in which physically similar products are embedded close to each other.

Finally, a Deep Learning model based on Entity Embedding layers aims at, indirectly, learning the best vector representation for each product, in terms of purchase history and behavior.

Thus, in all methodologies objective products are embedded into a product space, where both present and past products are projected. Within this new learned space, distances between products can be calculated, hence similarities based on these distances can be used to search for the most akin products, given a reference. Then, given a current product never sold before and looking for the most similar products in the preceding years, *pseudo* time-series are built. The forecasting model will then exploit this information to tackle the prediction task.

5.2.1. Rule-based methodology

The rule-based methodology to track down similarities between products of the current sales campaign and past ones is based on two steps: the first is concerned of partitioning the product collection in different non-overlapping subsets of products which share same categorical attributes (i.e. brand, collection, season), while the second step imposes a weighted metric to the subset products' based on continuous attributes (i.e. textile composition, color).

More specifically, *i)* based on a group of selected product categorical attributes, given a product, all past products that share the same combination of these partitive categorical attributes are filtered and considered in the next step. More formally, given a product *i*, the subset of products is defined as in formula (30):

$$subset_i = \{j \in P | j_k = i_{k,} \forall\, k \in CA\} \quad (32)$$

where *CA* is the set of categorical variables used to partition the products' set *P*.

Then, *ii)* within the partitive subset identified at the previous step, four different attributes are used to measure similarities between products. The four different attributes are the following: fabric composition, fabric texture, color, and a subset of Boolean flags, all

related to describe physical product characteristics. Euclidean distances (d_{fc}, d_{ft}, d_c, d_f) are computed with respect to the first three of these product attributes, while the Jaccard distance is correctly used to measure the similarity between products with respect to the subset of Boolean flags variables. It must be noted that for the fabric texture and color attributes, a manually imposed metric for each attribute has been set, accordingly to human expertise, as these discrete attributes, while very important to describe the physical appearance of the product, cannot be directly used to compute Euclidean distances. Details of these imposed metrics are not reported for the sake of conciseness. Then, a final weighted Euclidean distance *wd* is computed, in which each product coordinate is one of the previously calculated distances, as in formula (31):

$$wd = \sqrt{w_{fc} \cdot {d_{fc}}^2 + w_{ft} \cdot {d_{ft}}^2 + w_c \cdot {d_c}^2 + w_f \cdot {d_f}^2} \qquad (33)$$

Finally, based on this weighted Euclidean distance, similarities are obtained accordingly.

5.2.2. Autoencoder

The second method, which is based on a Deep Learning model, leverages an AutoEncoder topology to embed each product in a space accordingly to its technical drawing. An AutoEncoder is a Neural Network model used to learn efficient data representations, which is trained in an unsupervised manner. The Neural Network topology of the AutoEncoder is composed of two main modules, the Encoder stage and the Decoder stage. The first part is composed of several hidden layers, which shrinks in terms of number of neurons, as the layers progress away the input layer. The second part, conversely, is made of many hidden layers which gradually increase the dimensionality of the processed data. The output layer of the AutoEncoder has the same dimensionality of the input layer, given that the Neural Network is trained to reconstruct the input received. The Encoder and the Decoder stages share a common hidden layer, where the dimensionality reaches the minimum within the topology. Such hidden layer is usually referred to as the bottleneck of the AutoEncoder. The key idea behind this peculiar topology is that, during the training of the network, the Decoder stage should learn effectively out to reconstruct the input of the network based on

the compressed representation passed to it by the Encoder stage through the bottleneck. One of the main applications is to proceed, once the training is finished, to use only the Encoder part to compress the data into a much lower dimensional space than the original.

Indeed, in this work, the main scope of applying a Convolutional AutoEncoder to products' technical drawings has been to embed each product within a learned product space in which products which look similar in terms of their physical appearance are clustered closer together. Similarities are then computed in terms of the Euclidean distance between the learned compressed products' representations.

5.2.3. Entity Embedding MLP

This Deep Learning model aims at learning the most optimal embedded product space, based on the product attributes available, and it is based on a multitask MultiLayer Perceptron (MLP) which employs Entity Embeddings layers in front of the input layer. The neural network is trained in a supervised multi-task fashion, processing the order history, which consists of data-points where for each tuple (product, client) the quantity sold is given. Both the product and the customer are described by their respective attributes. Moreover, since the dataset is augmented using a Negative Sampling technique, which is an adaptation of the technique used in [132], the neural network will produce a double output, in which, for a given (product, customer) tuple it will predict both the quantity sold and the probability that the event will occur.

More precisely, the dataset, extracted from the company database, obviously does not contain records which report the lack of purchase of a certain garment by any customer, which could be interpreted as a signal of lack of interest for the product. The Negative Sampling technique here adopted consists of analyzing the whole order history to list, for each product, all customers which never bought that product. Then, for each product, with a uniform probability, a subset of these customers is sampled to create the negative samples of missing purchase and then inserted into the original dataset. Moreover, while the negative sampling technique adds more data-points to the dataset, a new column is also added to the design matrix, in which these newly synthetized rows are marked with a zero (i.e. missing purchase), and those originally present are marked with a one (i.e. real purchase, actually occurred). In this way, while maintaining the original regressive task of predicting the quantity sold for

each (product, customer) tuple, a new classification task is added, in which the neural networks is trained to predict the probability of purchase of a product by a customer. This enables to train the neural network in a multi-task fashion, which helps speed up learning the best vector representation of each product.

However, while the lack of time-series is the fundamental obstacle to overcome to be able to synthesize a demand forecasting system, another problem further complicates the task at hand. The representation learning task tackled by the MLP model is further hindered by the large number of categorical variables that describe the product attributes, which are difficult to reconcile with similarity metrics. Some of the categorical variables describe the type of product (class), the color, its brand, and other descriptive characteristics. While a first approach could exploit these categorical and binary variables to build a partition-based system, it could prove to be too restrictive. Easily, the products could be separated into bins corresponding to different combinations of the categorical variables' values, and for each bin a different forecasting model could be built. Nonetheless, it could be argued that similarities should also be traced between clothes with different categorical attributes, such as different color. Furthermore, the forecasting system should be fed with input data which should somehow correlates in terms of purchasing behavior, not simply on arbitrary pre-fixed partitions, whose structure could be not optimized for the task at hand. Another classical approach to encode categorical variables is to use the One-Hot Encoding method. The methodology consists in mapping each discrete value of the categorical variable to a vector of n entries of Boolean values, where n is the cardinality of the categorical variable, and with only one entry equals to one. However, such representation would be hardcoded, sparse (most entries would be zero), and high-dimensional (forced to be equal to the cardinality of the categorical variable it encodes). Moreover, all possible values would be orthogonal equispatiated vertices on a high-dimensional hypercube. Thus, the encoding would not carry any specific meaning and relationships about the original values encoded. To obtain the best vector representation for the information encoded in the categorical variables (both product and customer attributes), we leveraged the concept of Entity Embeddings [133], which maps categorical variables values into Euclidean spaces. Entity Embeddings are compact, lower-dimensional and dense representations, learned directly from data and optimized for the task(s) at hand. Moreover, the learning process should give meaningful representations, in which relationships and intuitive distances between original discrete values could be traced.

Entity Embeddings map each discrete categorical value to a *d*-dimensional real valued vector. Intuitively, they are equivalent to a hidden linear layer with *d* neurons which follows the input layer of the neural network, which is fed with one-hot encoded inputs. At the implementation level, however, the one-hot encoding step is not needed, and the weight matrix ***E*** of the first hidden linear layer is treated as a look-up table. The matrix ***E*** is of dimension *Cxd*, where C corresponds to the cardinality of the categorical variable embedded by the matrix. Instead, the number of columns *d* of the matrix is instead a hyper-parameter which can be fine-tuned by the user, and which usually is greatly lower than the cardinality, to have a more compact data representation.

The architecture of the Entity Embeddings Neural Network, illustrated in Figure 15, is the following: the input layer has 15 neurons for categorical variables, each followed by its corresponding Entity Embedding matrix, 9 inputs for Boolean variables and 2 inputs for continuous variables. After, two concatenation layers exists: one concatenates products' input variables, the other concatenates customers' input variables. Then, the topology split into two Siamese modules which are placed after the two concatenation layers: each Siamese module has three hidden layers of 128, 64, 32 neurons each. Then, the output of one module from the product's channel and the output of one module from the customer's channel are concatenated together and elaborated by a final hidden layer of 32 neurons, to output the predicted quantity. The same occurs to produce the second output, namely the probability of purchase event.

It must be stressed out that the forecast made by the Entity Embedding Neural Network is not the one that will be used by the company. It is, in fact, just a "feedback signal" needed to train the network to recognize similarities (both physical and commercial) among the products.

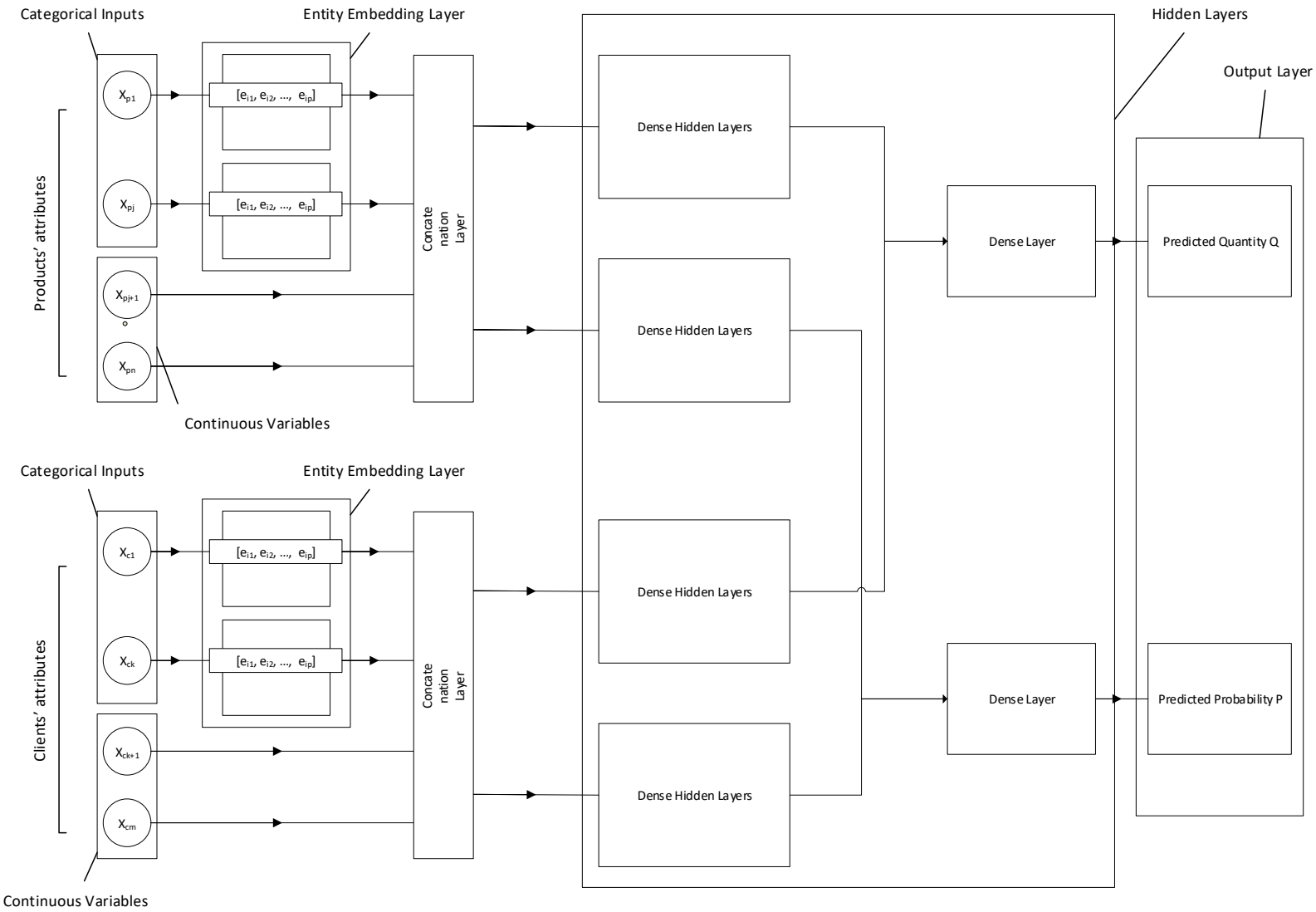


Figure 15 - Entity Embedding Topology

5.3. Core customers selection

The second phase of the work tackled the problem of establishing a methodology to find the most representative (stable) subsets of customers, with the aim of feeding the forecasting model with the data generated by those subsets, in order to filter out noise generated by very different purchase behaviors within the customer base. In addition, the customer groups identified in this way will also be used operationally: during the early stages of the sales campaign, efforts will be made to collect data from these representative customers first, so that, basically, a customer survey is carried out. These data will be the input of the forecast model that concerns the real-time trend of the sales campaign.

The reasoning behind this phase stems from a first phase of exploratory data analysis, during which two dynamics were traced in the customer data: first, a limited group of customers was active during the overall time window considered here, ten years. Conversely, many customers have been active for a limited number of sales campaigns, sometimes in a rather overt way. Moreover, regardless of how many sales campaign they have been active, some customers have tastes in line with average market behavior, whereas others have interests in small groups of very peculiar products, sometimes in a very erratic way.

Given these observations, the methodology here presented aims at finding those core subsets of customers, whose purchase behavior has been constantly aligned with the average market performance. To this aim, for each product class the overall average market performance is calculated, along each year. Then, within specific geographical areas, after filtering out the least important customers in terms of volume, by means of a Pareto Analysis, for each customer a score is calculated, measuring the divergence with respect to the overall average behavior, for each product class. The output will be, for each product class, a ranking of the customers most aligned with the market.

More specifically, *i)* customers are divided into three geographical areas (namely Europe, Asia, and North America), with respect to where they sell to end customers. Then, *ii)* for each area, only those customers which cumulatively sold 80% of the total quantities sold overall are kept. Finally, *iii)* the last ten years are considered and for each geographical area A the total quantities $Q_{A,t}$ and the total quantities $Q_{A,C,t}$ for each product class C sold in year t

$\in[1, 10]$ are computed. Next, dividing $Q_{A,C,t}$ by $Q_{A,t}$, a reference vector $\vec{\boldsymbol{r}}_{A,C}$ is defined for each geographical area *A* and for each product class *C*.

$$\vec{\mathbf{r}}_{\mathrm{A,C}} = \langle \mathrm{r}_{\mathrm{t}}^{\mathrm{A,C}} \rangle, \quad \mathrm{t} \in [1, 10] \tag{34}$$

with $\mathrm{r}_{\mathrm{t}}^{\mathrm{A,C}}$ defined as follows:

$$\mathrm{r}_{\mathrm{t}}^{\mathrm{A,C}} = \frac{\sum_{\{\mathrm{i} \in \mathrm{A}, \mathrm{j} \in \mathrm{C}\}} \mathrm{q}_{\mathrm{i,j,t}}}{\sum_{\{\mathrm{i} \in \mathrm{A}\}} \mathrm{q}_{\mathrm{i,t}}} = \frac{\mathrm{Q}_{\mathrm{A,C,t}}}{\mathrm{Q}_{\mathrm{A,t}}} \tag{35}$$

The vector represents the proportion of each product class *C* with respect to total quantity sold in a given area *A*

Furthermore, *iv)* for each stable customer a similar analysis is conducted, to obtain the following reference vector $\vec{\mathbf{r}}_{\mathrm{i,C}}$:

$$\vec{\mathbf{r}}_{\mathrm{i,C}} = \langle \mathrm{r}_{\mathrm{t}}^{\mathrm{i,C}} \rangle, \quad \mathrm{t} \in [1, 10] \tag{36}$$

where:

$$\mathrm{r}_{\mathrm{t}}^{\mathrm{i,C}} = \frac{\sum_{\{\mathrm{j} \in \mathrm{C}\}} \mathrm{q}_{\mathrm{i,j,t}}}{\mathrm{q}_{\mathrm{i,t}}} \tag{37}$$

which characterize the volume of products of class *C*, purchased by customer *i*, with respect to the overall volume purchased by the same customer, in a given year *t*.

Finally, the Euclidean distance between each reference vector $\vec{\mathbf{r}}_{\mathrm{i,C}}$ and $\vec{\mathbf{r}}_{\mathrm{A,C}}$ is computed as in Eq. (36) to obtain a score for each customer *i* in each class *C*.

$$\mathrm{score}_{\mathrm{i,C}} = \left\| \vec{\mathbf{r}}_{\mathrm{A_i,C}} - \vec{\mathbf{r}}_{\mathrm{i,C}} \right\|_2 \tag{38}$$

The lower is the score (distance) the more customer *i* is aligned with the purchasing behavior of the overall customer base. Thus, for each area and product class, customers are sorted by score in ascending order, and only the first 50 percent is held and aggregated among

geographical area. By operating in this way, it is possible to identify, for each product class, a set of reference customers, that will be used in the following phases.

5.4. An RNN-based Deep Learning Demand Forecasting Model

The output of the final stage of the work is the demand forecasting model which takes advantage of the Entity Embedding neural network and of the core customer selection method. As for the latter, in the following considerations all quantities are limited to those generated by the core customer subset identified.

The Entity Embedding neural network is used to output, for each current and past product, the learned vector representation. This produces a final Products Embedding matrix ***PE*** in which, for each row corresponding to a product, the matching learned vector representation is indicated. From this matrix a Euclidean distance matrix ***D*** can be calculated. Based on this distance matrix ***D*** a similarity search can be carried over: for each current product P_0, from the corresponding row in the matrix ***D***, distances from all others products are partitioned by year, meaning that for each of the three preceding years only the distances of those products belonging to each year are taken into consideration. Then, for each year $t \in [-1,-3]$ the most similar, aka the closest, product P_t is selected. The search for the most similar product in the past is done for the three preceding years, which was indicated by the company at study as a reasonable amount of time in which fashion trends can be tracked.

For each similar past products, the following variables are collected: *i)* the learned vector representation; *ii)* the quantity sold at the end of the first week $q_{i,-t}$; *iii)* the quantity sold at the end of the sales campaign $Q_{i,-t}$. Additionally, the initial quantity q_0 sold of the current product P_0 in the first week of sales campaign is used, along with its learned vector representation. These data are organized into one multidimensional *pseudo* time-series, in which, for each preceding year and corresponding most similar product, the vector representation, the quantity sold at the end of the first week, the final quantity sold, are merged together. This *pseudo* time-series is then fed to a Recurrent Neural Network (RNN) module. The RNN then outputs the sequence of hidden states produced, which are processed by a dense hidden layer. The output of the dense hidden layer is concatenated with information about the relative current sales campaign, namely the vector representation and the quantity

sold up to the end of the first week of sales campaign. The concatenation is then processed by a final hidden layer which outputs the predicted final quantity. The network topology is described in Figure 16, while Figure 17 illustrates how the RNN-based network takes in input the different data sources.

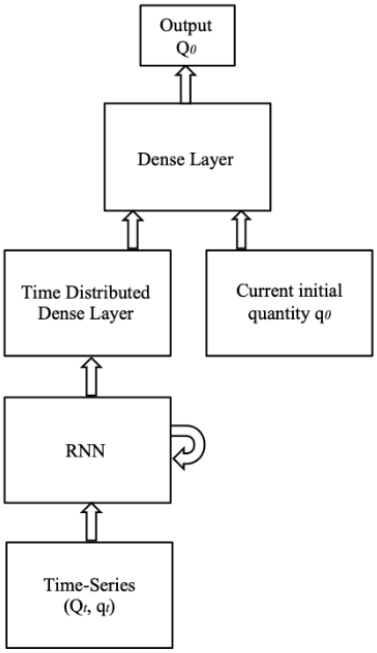


Figure 16 - Forecasting RNN-based neural network topology

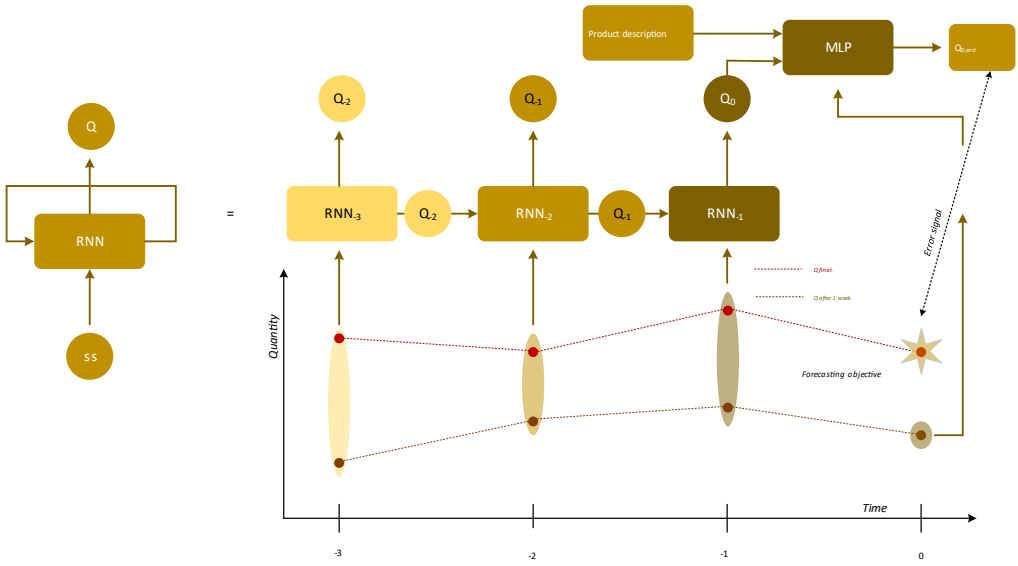


Figure 17 - RNN data flow

5.5. Results

Experiments have been conducted training the Entity Embedding Neural Network and the Demand Forecasting RNN-based Neural Network on the second-last sales campaign. The last sales campaign has been used as test set. The neural networks were implemented in Python© 3.6 with the Keras© package, with TensorFlow© as backend.

The Entity Embeddings Neural Network was trained using the Relu activation function [3] in the hidden layers, a linear activation for the output of the predicted quantity, and a sigmoid activation for the output of the probability of purchase. The batch size, to allow for a better convergence behavior, was set to 2048. The neural network was trained for 10 epochs, with the Adam optimizer. The loss function is a weighted sum of the binary cross-entropy loss for the probability output and mean absolute error for the continuous output.

The RNN-based Neural Network was trained for 100 epochs, with a batch size of 64 and with a Mean Absolute Percentage Error, to cope with the extreme variation in the final output (ranging from as low as 1 to as high as 5000). The training set consists of data of the penultimate sales campaign available, and the *pseudo* time series are reconstructed with a time window of three years, that is, for each current product, the most similar product for the

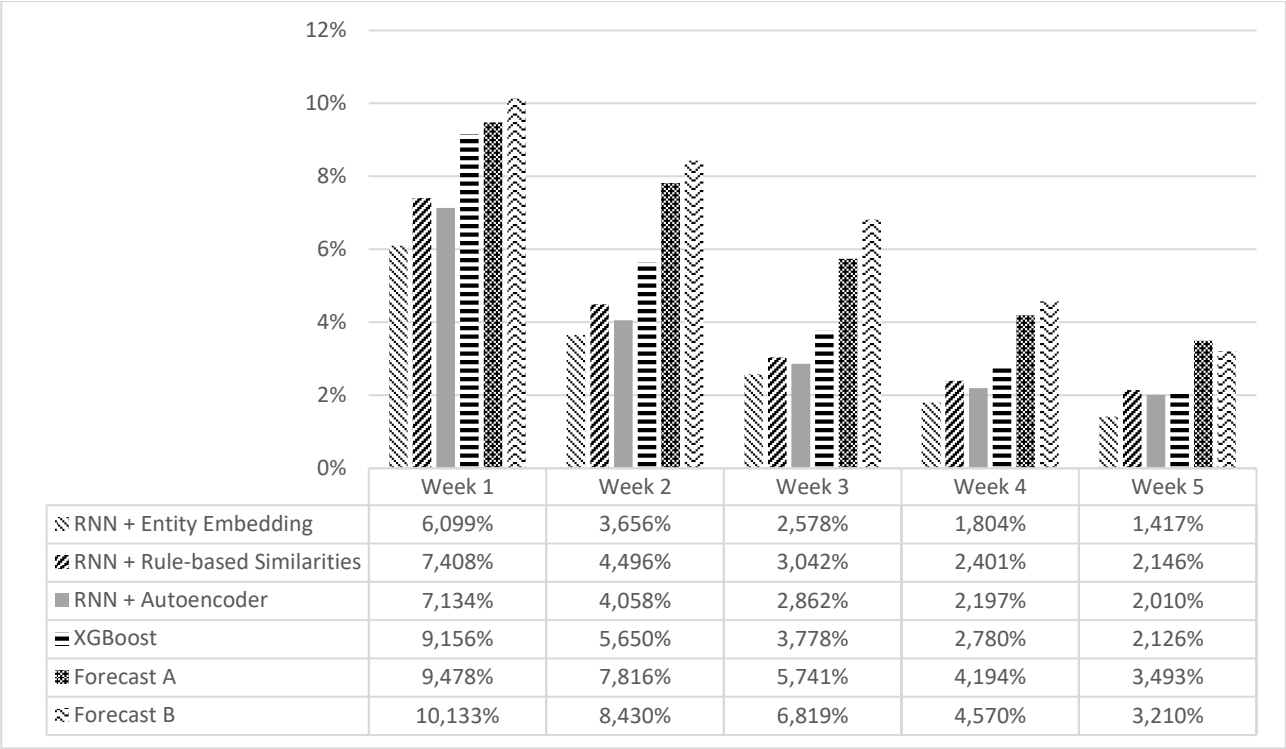
three preceding years is identified and its attribute are used to build the time series. The test set consists of the most recent sales campaign available.

Moreover, to assess the effectiveness of the similarities recovered using the Entity Embeddings Neural Network, both the rule-based method and the AutoEncoder method are tested for reconstructing the *pseudo* time series.

Three different models were used as benchmark to assess the validity of the proposed RNN approach. Two of them (Forecast A and B) consist of the existing forecasting system currently employed at the firm. While the details of these two forecasting methods cannot be fully disclosed, they are based on human expertise and on expectations about cumulative selling trends.

The third benchmark model is based on the XGBoost algorithm [15], described in section 2.1.5.5. This model is not based on the *pseudo* time series, since it is only trained with products' attributes in order to obtain the final prediction. Categorical variables are one-hot encoded.

Test have been conducted at different time interval of the sales campaign, namely with sales data gathered after one week of sales campaign up to five weeks. Errors are measured in terms of Mean Absolute Percentage Error (MAPE) and result are shown in Figure 18.



	Week 1	Week 2	Week 3	Week 4	Week 5
RNN + Entity Embedding	6,099%	3,656%	2,578%	1,804%	1,417%
RNN + Rule-based Similarities	7,408%	4,496%	3,042%	2,401%	2,146%
RNN + Autoencoder	7,134%	4,058%	2,862%	2,197%	2,010%
XGBoost	9,156%	5,650%	3,778%	2,780%	2,126%
Forecast A	9,478%	7,816%	5,741%	4,194%	3,493%
Forecast B	10,133%	8,430%	6,819%	4,570%	3,210%

Figure 18 - Comparison of forecasting results

As it can be seen, the three forecasting models based on product similarities and *pseudo* time-series consistently outperformed both the benchmark internal forecast and the XGBoost method. Indeed, just after one week of sales campaign the RNN forecasting model based on the Entity Embeddings similarities has a MAPE which is comparable to the error of the Forecast An internal model after three weeks of sales campaign. Furthermore, the RNN fed with *pseudo* time series obtained through the Entity Embedding neural network outperformed also other methods for similarities reconstruction. Thus, the similarities obtained embedding products based on historical purchase behavior can be effectively exploited to trace back chain of similarities between current and past product, even if the current product has never been sold before.

5.6. Conclusions

The work presented a three-stage forecasting system for the fashion industry, based on Deep Learning methodologies.

The proposed approach focused on three main part. The first one proposes three different methods for tracing similarities between products. This step is needed because new products are offered at each sales campaign, thus no time-series are available and must be

reconstructed through a chain of similarities between current offerings and past ones. Three methods are proposed to measure similarities between products, two based on Deep Learning models and one rule-based. The second step proposes a method to focus the analysis on core subsets of customers, so to reduce variability and noise within the historical dataset available, caused by market and fashion trend dynamics. The last step proposes an RNN based forecasting system which analyzes the reconstructed *pseudo* time series and the status of the current sales campaign to forecast the final quantity sold. It is compared with existing forecasting system and a XGBoost based benchmark approach.

Results show a valuable gain with respect to the existing forecasting system, effectively enabling a more accurate and timelier forecast.

Future works could evaluate more complex RNN topologies, as well as more deep architectures. Moreover, also CNN could be used to replace the RNN-based topologies, for faster inference. Furthermore, forecasting models which take into account cannibalistic effect among current offering could be explored.

6. Conclusions

This thesis focused on the application of Machine Learning algorithms, and in particular Deep Learning, to address and solve problems typical of the industrial field. Four applications in particular have been studied, one of which is a business case study. Specifically, innovative Machine Learning algorithms, which have shown great developments in recent years, have been tested against the following challenging industrial problems: *i*) bankruptcy forecast, *ii*) management of hybrid systems for production planning and control, and *iii*) demand forecast in the fashion industry.

Interesting results have been achieved and indeed, in all occasions, the tested models improved the results achievable with more classic statistical techniques. Moreover, as in the case study of chapter 5, the Machine Learning techniques have proved to be essential to be able to face problems otherwise difficult to face with classical methodologies.

Surely, for several years now, there is a lot of interest and hype towards these methodologies, especially given the results in different areas, such as speech and image recognition. In addition, initiatives such as Industry 4.0 enable companies to take these technologies seriously.

Nevertheless, although studies on the applicability of these algorithms in the industrial field have always existed, as described in chapter 2.2, it is only recently that, perhaps a little late with respect to similar application for the IT field, the innovations of recent years in the industrial field are now being applied. Certainly, there is still the need to understand well how far these methodologies can be pushed, and how much they can be useful. Their high complexity, in fact, can sometimes make the use of simpler and more comprehensible methods more attractive. However, it is opinion of the author that these methodologies will find more and more applications and success in the industrial field, in line with the results obtained within each of the studies presented in this thesis. Moreover, in order to better promote a faster spread of these methodologies and other Machine Learning algorithms, the author suggests that it could be helpful to establish benchmark datasets against which to compare the different algorithms, so as to improve the state-of-the-art and facilitate the implementation in real industrial case studies.

This could prove to be particularly helpful with respect to Reinforcement Learning algorithms, which are still shallowly explored in the industrial field, especially for decision support systems. This is probably due to the complexity of their implementation that still requires a lot of effort. Nevertheless, considering the rapid and impressive results obtained against complex board games (i.e. AlphaGo), their industrial use can prove to be fruitful, and thus, more research efforts should be spent in the near future.

Lastly, we note that one of the most critical aspect that could hinder the diffusion of these method concerns the interpretability of results, because these methodologies can sometimes be treated and used as black-boxes. Although this point is not always seen as disadvantage, especially when employed in decision-support systems, it would be preferable that the user could be able to interpret and thoroughly understand the algorithm and the mechanisms that led to certain decisions. Thus, the definition of new ways to better explore the underlying logic of Machine Learning and Deep Learning algorithms and to gain interpretability of the results could be another very interesting research field, which in case of success could further accelerate the adoption of these techniques in the industrial field.

7. Bibliography

[1] K. Murphy, Machine Learning: a probabilistic perspective, Cambridge: MIT Press, 2012.

[2] R. Sutton and A. Barto, Reinforcement Learning: An introduction, Cambridge: MIT press, 1998.

[3] X. Glorot, A. Bordes and Y. Bengio, "Deep sparse rectifier neural networks.," *Proceedings of the fourteenth international conference on artificial intelligence and statistics. ,* pp. 315-323, 2011.

[4] D. Rumelhart, R. Durbin, R. Golden and Y. Chauvin, Backpropagation: The basic theory. Backpropagation: Theory, architectures and applications, 1995.

[5] K. Fukushima, "Neural network model for a mechanism of pattern recognition unaffected by shift in position-Neocognitron.," *IEICE Technical Report,* vol. 62, no. 10, pp. 658-665, 1979.

[6] Y. LeCun, L. Bottou and Y. Bengio, "Gradient-based learning applied to document recognition.," in *Proceedings of the IEEE 86.11*, 1998.

[7] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT press, 2016.

[8] P. Werbos, "Backpropagation through time: what it does and how to do it.," in *Proceedings of the IEEE 78.10*, 1990.

[9] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions.," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 6, no. 02, pp. 107-116, 1998.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory.," *Neural computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[11] J. Chung, C. Gulcehre, K. Cho and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling.," *arXiv preprint arXiv:1412.3555,* 2014.

[12] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001.

[13] G. James, D. Witten, T. Hastie and R. Tibshirani, An Introduction to Statistical Learning, New York: Springer, 2013.

[14] L. Breiman, "Bagging predictors," *Machine Learning,* vol. 24, no. 2, pp. 123-140, 1996.

[15] C. Tianqi and C. Guestrin, "Xgboost: A scalable tree boosting system.," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.*, 2016.

[16] J. Friedman, "Greedy function approximation: a gradient boosting machine.," *Annals of statistics,* pp. 1189-1232, 2001.

[17] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues.," *Journal of Industrial Information Integration,* vol. 6, pp. 1-10, 2017.

[18] S. Mittal, M. A. Khan and T. Wuest, "Smart manufacturing: characteristics and technologies," in *IFIP International Conference on Product Lifecycle Management*, 2016.

[19] A. Kusiak, "Smart manufacturing must embrace big data," *Nature,* vol. 544, no. 7648, pp. 23-25, 2017.

[20] M. Waller and S. Fawcett, "Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management.," *Journal of Business Logistics,* vol. 34, no. 2, pp. 77-84, 2013.

[21] L. Monostori, "AI and machine learning techniques for managing complexity, changes and uncertainties in manufacturing.," *Engineering applications of artificial intelligence,* vol. 16, no. 4, pp. 277-291, 2003.

[22] J. Harding, M. Shahbaz and A. Kusiak, "Data mining in manufacturing: a review.," *Journal of Manufacturing Science and Engineering,* vol. 128, no. 4, pp. 969-976, 2006.

[23] A. McAfee and E. Brynjolfsson, "Big data: the management revolution.," *Harvard Business Review,* vol. 90, no. 10, pp. 60-68, 2012.

[24] D. Pham and A. Afify, "Machine-learning techniques and their applications in manufacturing.," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* vol. 219, no. 5, pp. 395-412, 2005.

[25] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever and Lillic, "Mastering the game of Go with deep neural networks and tree search," *Nature,* vol. 529, no. 7587, p. 484, 2016.

[26] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan and D. Hassabis, "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," *arXiv preprint,* 2017.

[27] Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning," *Nature,* vol. 521, no. 7553, p. 436, 2015.

[28] S. LaValle, E. Lesser, R. Shockley, M. Hopkins and N. Kruschwitz, "Big data, analytics and the path from insights to value.," *MIT sloan management review,* vol. 52, no. 2, pp. 21-32, 2011.

[29] D. Wolpert and W. Macready, "No free lunch theorems for optimization.," *IEEE transactions on evolutionary computation ,* pp. 67-82, 1997.

[30] A. Widodo and . B.-S. Yang, "Support vector machine in machine condition monitoring and fault diagnosis.," *Mechanical systems and signal processing,* vol. 21, no. 6, pp. 2560-2574, 2007.

[31] F. Jia, Y. Lei, J. Lin., X. Zhou and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery.," *Mechanical Systems and Signal Processing,* vol. 72, pp. 303-315, 2016.

[32] P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification.," *Reliability Engineering & System Safety,* vol. 115, pp. 124-135, 2013.

[33] S. Li, G. liu, X. Tang, J. Lu and J. Hu, "An Ensemble Deep Convolutional Neural Network Model with Improved D-S Evidence Fusion for Bearing Fault Diagnosis.," *Sensors,* vol. 17, no. 8, p. 1729, 2017.

[34] A. Saxena and A. Saad, "Evolving an artificial neural network classifier for condition monitoring of rotating mechanical systems.," *Applied Soft Computing,* vol. 7, no. 1, pp. 441-454, 2007.

[35] X. ùZhang, W. Chen, B. Wang and X. Chen, " Intelligent fault diagnosis of rotating machinery using support vector machine with ant colony algorithm for synchronous feature selection and parameter optimization.," *Neurocomputing,* vol. 167, pp. 260-279, 2015.

[36] P. Kankar, S. Sharma and S. Harsha, "Fault diagnosis of ball bearings using machine learning methods," *Expert Systems with applications,* vol. 38, no. 3, pp. 1876-1886, 2011.

[37] C. Sobie, C. Freitas and M. Nicolai, "Simulation-driven machine learning: Bearing fault classification.," *Mechanical Systems and Signal Processing,* vol. 99, pp. 403-419, 2018.

[38] M. D. Prieto, G. Cirrincione, A. Espinosa, J. Ortega and H. Henao, "Bearing Fault Detection by a Novel Condition-Monitoring Scheme Based on Statistical-Time Features and Neural Networks.," *EEE Transactions on Industrial Electronics,* vol. 60, no. 8, pp. 3398-3407, 2012.

[39] A. Azadeh, M. Saberi, A. Kazem, V. Ebrahimipour, A. Nourmohammadzadeh and Z. Saberi, "A flexible algorithm for fault diagnosis in a centrifugal pump with corrupted data and noise based on ANN and support vector machine with hyper-parameters optimization.," *Applied Soft Computing,* vol. 13, no. 3, pp. 1478-1485, 2013.

[40] G. Susto, A. Schirru, S. Pampuri, S. McLoone and A. Beghi, "Machine Learning for Predictive Maintenance: A Multiple Classifier Approach.," *EEE Transactions on Industrial Informatics,* vol. 11, no. 3, pp. 812-820, 2014.

[41] A. Kusiak and C. Kurasek, "Data Mining of Printed-Circuit Board Defects.," *IEEE transactions on robotics and automation,* vol. 17, no. 2, pp. 191-196, 2001.

[42] S. Tan, J. Watada, Z. Ibrahim and M. Khalid, "Evolutionary Fuzzy ARTMAP Neural Networks for Classification of Semiconductor Defects," *IEEE transactions on neural networks and learning systems,* vol. 26, no. 5, pp. 933-950, 2014.

[43] F. Adly, O. Alhussein, P. D. Yoo, Y. Al-Hammadi, K. Taha, S. Muhaidat, Y.-S. Jeong, U. Lee and M. Ismail, "Simplified Subspaced Regression Network for Identification of Defect Patterns in Semiconductor Wafer Maps," *IEEE Transactions on Industrial Informatics,* vol. 11, no. 6, pp. 1267-1276, 2015.

[44] S. Ravikumar, K. Ramachandran and V. Sugumaran, "Machine learning approach for automated visual inspection of machine components.," *Expert systems with applications,* vol. 38, no. 4, pp. 3260-3266, 2011.

[45] B. Gao, W. Woo, G. Tiam and H. Zhang, "nsupervised Diagnostic and Monitoring of Defects Using Waveguide Imaging With Adaptive Sparse Representation," *IEEE Transactions on Industrial Informatics,* vol. 12, no. 1, pp. 405-416, 2015.

[46] U. Çaydaş and S. Ekici, "Support vector machines models for surface roughness prediction in CNC turning of AISI 304 austenitic stainless steel.," *Journal of Intelligent Manufacturing,* vol. 23, no. 3, pp. 639-650, 2012.

[47] B. Ribeiro, "Support vector machines for quality monitoring in a plastic injection molding process.," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 35, no. 3, pp. 401-410, 2005.

[48] N. El-Bendary, E. El Hariri, A. Hassanien and A. Badr, "Using machine learning techniques for evaluating tomato ripeness.," *Expert Systems with Applications,* vol. 42, no. 4, pp. 1892-1905, 2015.

[49] P. Mohammadi and Z. Wang, "Machine Learning for Quality Prediction in Abrasion- Resistant Material Manufacturing Process.," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2016.

[50] T. Tušar, K. Gantar, V. Koblar, B. Ženko and B. Filipič, "A study of overfitting in optimization of a manufacturing quality control procedure.," *Applied Soft Computing,* vol. 59, pp. 77-87, 2017.

[51] D. Kim, P. Kang, S. Cho, H. Lee and S. Doh, "Machine learning-based novelty detection for faulty wafer detection in semiconductor manufacturing.," *Expert Systems with Applications,* vol. 39, no. 4, pp. 4075-4083, 2012.

[52] F. Ye, Z. Zhang, K. Chakrabarty and X. Gu, "Board-level functional fault diagnosis using artificial neural networks, support-vector machines, and weighted-majority voting.," *EEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 32, no. 5, pp. 723-736, 2013.

[53] T. Lee, K. Lee and C. Kim, "Performance of Machine Learning Algorithms for Class-Imbalanced Process Fault Detection Problems.," *EIEE Transactions on Semiconductor Manufacturing,* vol. 29, no. 4, pp. 436-445, 2016.

[54] K. Nakata, R. Orihara, Y. Mizuoka and K. Takagi, "A Comprehensive Big-Data-Based Monitoring System for Yield Enhancement in Semiconductor Manufacturing," *IEEE Transactions on Semiconductor Manufacturing,* vol. 30, no. 4, pp. 339-344, 2017.

[55] Z. Liu, Z. Jia, C. Vong, S. Bu, J. Han and X. Tang, "apturing High-Discriminative Fault Features for Electronics-Rich Analog System via Deep Learning," *IEEE Transactions on Industrial Informatics,* vol. 13, no. 3, pp. 1213-1226, 2017.

[56] T. Wuest, C. Irgens and K. Thoben, "An approach to monitoring quality in manufacturing using supervised machine learning on product state dat," *Journal of Intelligent Manufacturing,* vol. 25, no. 5, pp. 1167-1180, 2014.

[57] T. Ko, J. Lee, H. Cho, S. Cho, W. Lee and M. Lee, "Machine learning-based anomaly detection via integration of manufacturing, inspection and after-sales service data," *Industrial Management & Data Systems,* vol. 117, no. 5, pp. 927-945, 2017.

[58] M. Saucedo-Espinosa, H. Escalante and A. Berrones, "Detection of defective embedded bearings by sound analysis: a machine learning approach.," *ournal of Intelligent Manufacturing,* vol. 28, no. 2, pp. 489-500, 2017.

[59] B. Lenz, B. Barak, J. Mührwald and C. Leicht, "Virtual metrology in semiconductor manufacturing by means of predictive machine learning models.," in *2013 12th International Conference on Machine Learning and Applications.*, 2013.

[60] Y. Chen, C. Fan and K. Chang, "Manufacturing intelligence for reducing false alarm of defect classification by integrating similarity matching ap," *Computers & Industrial Engineering,* vol. 99, pp. 465-473, 2016.

[61] W. Yang and W. Zhou, "Autoregressive coefficient-invariant control chart pattern recognition in autocorrelated manufacturing processes using neural network ensemble.," *Journal of Intelligent Manufacturing,* vol. 26, no. 6, pp. 1161-1180, 2015.

[62] P. Priore, D. De La Fuente, A. Gomez and J. Puente, "Dynamic scheduling of manufacturing systems with machine learning.," *International Journal of Foundations of Computer Science,* vol. 12, no. 06, pp. 751-762, 2001.

[63] P. Priore, J. De la Fuente and J. Parreño, "A comparison of machine-learning algorithms for dynamic scheduling of flexible manufacturing systems.," *ngineering Applications of Artificial Intelligence,* vol. 19, no. 3, pp. 247-255, 2006.

[64] P. Priore, D. De La Fuente and R. Pino, "Learning-based scheduling of flexible manufacturing systems using case-based reasoning," *Applied Artificial Intelligence,* vol. 15, no. 10, pp. 949-963, 2001.

[65] B. Csáji, L. Monostori and B. Kádár, "Reinforcement learning in a distributed market-based production control system," *Advanced Engineering Informatics,* vol. 20, no. 3, pp. 279-288, 2006.

[66] M. Gaham and B. Bouzouia, "Intelligent product-driven manufacturing control: A mixed genetic algorithms and machine learning approach to product intelligence synthesis.," in *XXII International Symposium on Information, Communication and Automation Technologies*, 2009.

[67] F. Arredondo and E. Martinez, "Learning and adaptation of a policy for dynamic order acceptance in make-to-order manufacturing," *Computers & Industrial Engineering,* vol. 58, no. 1, pp. 70-83, 2010.

[68] Y. Shiue, R. Guh and K. Lee, "Study of SOM-based intelligent multi-controller for real-time scheduling," *Applied Soft Computing,* vol. 11, no. 8, pp. 4569-4580, 2011.

[69] J. Palombarini and E. Martinez, "SmartGantt–An intelligent system for real time rescheduling based on relational reinforcement learning," *Expert Systems with Applications,* vol. 39, no. 11, pp. 10251-10268, 2012.

[70] J. Heger, J. Branke, T. Hildebrandt and B. Scholz-Reiter, "Dynamic adjustment of dispatching rule parameters in flow shops with sequence dependent setup times," *International Journal of Production Research,* vol. 54, no. 22, pp. 6812-6824, 2016.

[71] M. Drakaki and P. Tzionas, "Manufacturing Scheduling Using Colored Petri Nets and Reinforcement Learning," *Applied Sciences,* vol. 7, no. 2, p. 136, 2017.

[72] S. Doltsinis, P. Ferreira and N. Lohse, "Reinforcement learning for production ramp-up: A Q-batch learning approach," in *2012 11th International Conference on Machine Learning and Applications*, 2012.

[73] H. Li, "An approach to improve flexible manufacturing systems with machine learning algorithms," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016.

[74] C. Kim, I. Kwon and J. Baek, "Asynchronous action-reward learning for nonstationary serial supply chain inventory control," *Applied Intelligence,* vol. 28, no. 1, pp. 1-16, 2008.

[75] I. Kwon, C. Kim, J. Jun and J. Lee, "Case-based myopic reinforcement learning for satisfying target service level in supply chain," *Expert Systems with applications,* vol. 35, no. 1-2, pp. 389-397, 2008.

[76] C. Jiang and Z. Sheng, "Case-based reinforcement learning for dynamic inventory control in a multi-agent supply-chain system," *Expert Systems with Applications,* vol. 36, no. 3, pp. 6520-6526, 2009.

[77] A. Kara and I. Dogan, "Reinforcement learning approaches for specifying ordering policies of perishable inventory systems," *Expert Systems with Applications,* vol. 91, pp. 150-158, 2018.

[78] S. Chaharsooghi, J. Heydari and S. Zegordi, "A reinforcement learning model for supply chain ordering management: An application to the beer game," *Decision Support Systems,* vol. 45, no. 4, pp. 949-959, 2008.

[79] J. Kruppa, A. Schwarz, G. Arminger and A. Ziegler, "Consumer credit risk: Individual probability estimates using machine learning," *Expert Systems with Applications,* pp. 5125-5131, 2013.

[80] R. Pal, K. Kupka, A. Aneja and J. Militky, "Business health characterization: a hybrid regression and support vector machine analysis.," *Expert Systems with Applications,* pp. 48-59, 2016.

[81] G. Wang, M. Jian and S. Yang, "An improved boosting based on feature selection for corporate bankruptcy prediction.," *Expert Systems with Applications,* pp. 2353-2361, 2014.

[82] G. King and L. Zeng, "Logistic regression in rare events data," *Political analysis,* vol. 9, no. 2, pp. 137-163, 2001.

[83] E. Altman, "Financial ratios, discriminant analysis and the prediction of corporate bankruptcy.," *The Journal of Finance,* vol. 23, no. 4, pp. 589-609, 1968.

[84] J. Courtis, "Modelling a financial ratios categoric framework," *Journal of Business Finance & Accounting,* vol. 5, no. 4, pp. 371-386, 1978.

[85] Y. Mensah, "An examination of the stationarity of multivariate bankruptcy prediction models: A methodological study," *Journal of Accounting Research,* pp. 380-395, 1984.

[86] C. Zopounidis, "A multicriteria decision-making methodology for the evaluation of the risk of failure and an application," *Foundations of Control Engineering,* vol. 12, no. 1, pp. 45-64, 1987.

[87] S. Suzuki and R. Wright, "Financial structure and bankruptcy risk in Japanese companies," *Journal of International Business Studies,* vol. 16, no. 1, pp. 97-110, 1985.

[88] G. Bottazzi, M. Grazzi, A. Secchi and F. Tamagni, "Financial and economic determinants of firm default," *Journal of Evolutionary Economics,* vol. 21, no. 3, pp. 373-406, 2011.

[89] S. Mueller and J. Stegmaier, "Economic failure and the role of plant age and size," *Small Business Economics,* vol. 44, no. 3, pp. 621-638, 2015.

[90] D. Liang, C. Lu, C. Tsai and G. Shih, "Financial ratios and corpo- rate governance indicators in bankruptcy prediction: A comprehensive study," *European Journal of Operational Research,* vol. 252, no. 2, pp. 561-572, 2016.

[91] J. Eklund, N. Levratto and G. Ramello, "Entrepreneurship and failure: two sides of the same coin?," *Small Business Economics,* pp. 1-10, 2018.

[92] A. Marshall, Principles of Economics, London: Macmillan, 1890.

[93] G. Beccatini, "The Marshallian industrial district as a socio-economic notion. In Industrial Districts and Inter-Firm Co-operation in Italy," *International Institute for Labour Studies,* pp. 37-51, 1990.

[94] L. Signorini, "The price of Prato, or measuring the industrial district effect," *Papers in Regional Science,* vol. 73, no. 4, pp. 369-392, 1994.

[95] E. Glaeser, H. Kallal, J. Scheinkman and A. Shleifer, "Growth in cities," *Journal of Political Economy,* vol. 100, no. 6, pp. 1126-1152, 1992.

[96] J. De Lucio, J. Herce and A. Goicolea, "The effects of externalities on productivity growth in Spanish industry," *Regional Science and Urban Economics,* vol. 32, no. 2, pp. 241-258, 2002.

[97] F. Cingano and F. Schivardi, "Identifying the sources of local productivity growth," *Journal of the European Economic Association,* vol. 2, no. 4, pp. 720-742, 2004.

[98] P. Martin, T. Mayer and F. Mayneris, "Spatial concentration and plant-level productivity in France," *Journal of Urban Economics,* vol. 69, no. 2, pp. 182-195, 2011.

[99] L. Guiso and Schivardi F., "Spillovers in industrial districts," *The Economic Journal,* vol. 117, no. 516, pp. 68-93, 2007.

[100] G. Brunello and M. Langella, "Local agglomeration, entrepreneurship and the 2008 recession: Evidence from Italian industrial districts," *Regional Science and Urban Economics,* vol. 58, pp. 104-114, 2016.

[101] B. Cassiman and S. Vanormelingen, Profiting from innovation: Firm level evidence on markups, 2013.

[102] F. Barboza, H. Kimura and E. Altman, "Machine learning models and bankruptcy prediction," *Expert Systems with Applications,* vol. 83, pp. 405-417, 2017.

[103] P. Henrich, M. Land and G. Gaalman, "Semi-interchangeable machines: implications for workload control," *Production Planning and Control,* pp. 91-104, 2007.

[104] J. Bertrand and H. Van Ooijen, "Workload based order release and productivity: a missing link.," *Production Planning and Control,* pp. 665-678, 2002.

[105] M. Thürer, M. Stevenson and C. Silva, "Three decades of workload control research: a systematic review of the literature.," *International Journal of Production Research,* pp. 6905-6935, 2011.

[106] G. Amaro, L. Hendry and B. Kingsman, "Competitive advantage, customisation and a new taxonomy for non make-to-stock companies.," *International Journal of Operations & Production Management,* pp. 349-371, 1999.

[107] M. Thürer, M. Land, M. Stevenson and L. Frendendall, "Card-based delivery date promising in high-variety manufacturing with order release control.," *International Journal of Production Economics,* pp. 19-30, 2016.

[108] M. Land, "Parameters and sensitivity in workload control.," *International journal of production economics,* pp. 625-638, 2006.

[109] M. Bertolini, G. Romagnoli and F. Zammori, "Simulation of two hybrid production planning and control systems: A comparative analysis.," *2015 International Conference on Industrial Engineering and Systems Management (IESM),* 2015.

[110] M. Land, "Cobacabana (control of balance by card-based navigation): A card-based system for job shop control.," *International Journal of Production Economics,* pp. 97-103, 117.

[111] M. Stevenson, Y. Huang, L. Hendry and E. Soepenberg, "The Theory and Practice of Workload Control: A Research Agenda and Implementation Strategy," *International Journal of Production Economics,* vol. 131, no. 2, pp. 689-700, 2011.

[112] N. Fernandes, M. Land and S. Carmo-Silva, "Workload control in unbalanced job shops," *International Journal of Production Research,* vol. 52, no. 3, pp. 679-690, 2014.

[113] M. Hüsig, "Online workload control with shifting bottlenecks and due date constraints.," in *2009 IEEE International Symposium on Assembly and Manufacturing*, 2009.

[114] M. Thürer, M. Stevenson, C. Silva and T. Qu, "Drum-buffer-rope and workload control in High-variety flow and job shops with bottlenecks: An assessment by simulation.," *International Journal of Production Economics,* vol. 188, pp. 116-127, 2017.

[115] M. Bertolini, G. Romagnoli and F. Zammori, "2MTO, a new mapping tool to achieve lean benefits in high-variety low-volume job shops," *Production Planning and Control,* vol. 28, no. 5, pp. 444-458, 2017.

[116] M. Thürer, T. Qu, M. Stevenson, C. Li and G. Huang, "Deconstructing bottleneck shiftiness: the impact of bottleneck position on order release control in pure flow shops.," *Production Planning and Control,* vol. 28, no. 15, pp. 1223-1235, 2017.

[117] S. Carmo-Silva and N. Fernandes, "Bottleneck-Oriented Order Release: An Assessment by Simulation," in *Proceeding of World Conference on Information Systems and Technologies World CIST*, Madeira, 2017.

[118] M. Thürer, C. Silva and M. Stevenson, "Optimising workload norms: the influence of shop floor characteristics on setting workload norms for the workload control concept.," *International Journal of Production Research,* vol. 49, no. 4, pp. 1151-1171, 2011.

[119] T. Corbett and J. M. Csillag, "Analysis of the effects of seven drum-buffer-rope implementations.," *Production & Inventory Management Journal,* pp. 17-25, 2001.

[120] S. Lawrence and A. Buss, "Shifting production bottlenecks: causes, cures, and conundrums," *Production and Operations Management,* vol. 3, pp. 21-37, 1994.

[121] S. Frühwirth-Schnatter, Finite mixture and Markov switching models., Springer Science & Business Media, 2006.

[122] W. Hopp and M. L. Spearman, "To pull or not to pull: what is the question?.," *Manufacturing & service operations management,* vol. 6, no. 2, pp. 133-148, 2004.

[123] D. Montgomery, Design and Analysis of Experiments, New York: John Wiley & Sons, 2012.

[124] P. Lavoie, J. Kenné and A. Gharbi, "Production control and combined discrete/continuous simulation modeling in failure-prone transfer lines.,"

International Journal of Production Research, vol. 45, no. 24, pp. 5667-5685, 2007.

[125] G. Dellino, J. Kleijnen and C. Meloni, "Robust optimization in simulation: Taguchi and response surface methodology.," *International Journal of Production Economics,* vol. 125, no. 1, p. 5259, 2010.

[126] A. Dean, D. Voss and D. Draguljić, Design and analysis of experiments. Vol. 1, New York: Springer, 2017.

[127] C. W. Dunnet, "A multiple comparison procedure for comparing several treatments with a control.," *Journal of the American Statistical Association,* vol. 50, no. 272, pp. 1096-1121, 1955.

[128] J. Hammond, Quick Response in the Apparel Industries, Cambridge, 1990.

[129] M. E. Nenni, L. Giustiniano and L. Pirolo, "Demand forecasting in the fashion industry: a review.," *International Journal of Engineering Business Management,* vol. 5, p. 37, 2013.

[130] R. Gutierrez, A. Solis and S. Mukhopadhyay, "Lumpy demand forecasting using neural networks.," *International Journal of Production Economics,* vol. 111, no. 2, pp. 409-420, 2008.

[131] Y. Yu, T.-M. Choi and C.-L. Hui, "An intelligent fast sales forecasting model for fashion products," *Expert Systems with Applications,* vol. 38, no. 6, pp. 7373-7379, 2011.

[132] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality.," *Advances in neural information processing systems,* pp. 3111-3119, 2013.

[133] C. Guo and F. Berkhahn, "Entity Embeddings of Categorical Variables," *arXiv preprint,* vol. arXiv:1604.06737, 2016.