Model-Based Engineering for the Integration of Manufacturing Systems with Advanced Analytics

David Lechevalier[1], Anantha Narayanan[2], Sudarsan Rachuri[3], Sebti Foufou[4], and Y. Tina Lee[5(✉)]

[1] Le2i, Université de Bourgogne, Dijon, France
david_lechevalier@etu.u-bourgogne.fr
[2] University of Maryland, College Park, MD, USA
anantha@umd.edu
[3] Advanced Manufacturing Office,
Department of Energy, Office of Energy Efficiency and Renewable Energy,
Washington, DC, USA
sudarsan.rachuri@hq.doe.gov
sudarsan.rachuri@ee.doe.gov
[4] CSE Department, College of Engineering, Qatar University, Doha, Qatar
sfoufou@qu.edu.qa
[5] Systems Integration Division, National Institute of Standards and Technology,
Gaithersburg, MD, USA
yung-tsun.lee@nist.gov

Abstract. To employ data analytics effectively and efficiently on manufacturing systems, engineers and data scientists need to collaborate closely to bring their domain knowledge together. In this paper, we introduce a domain-specific modeling approach to integrate a manufacturing system model with advanced analytics, in particular neural networks, to model predictions. Our approach combines a set of meta-models and transformation rules based on the domain knowledge of manufacturing engineers and data scientists. Our approach uses a model of a manufacturing process and its associated data as inputs, and generates a trained neural network model as an output to predict a quantity of interest. This paper presents the domain-specific knowledge that the approach should employ, the formal workflow of the approach, and a milling process use case to illustrate the proposed approach. We also discuss potential extensions of the approach.

Keywords: Data analytics · Meta-model · Neural network · Manufacturing process · Predictive modeling

1 Introduction

The manufacturing industry generates large amounts of data on products, processes, and resources, among other things. Data analytics provide the capabilities needed to extract insights and make predictions from these data. The potential impacts of data

R. Harik et al. (Eds.): PLM 2016, IFIP AICT 492, pp. 146–157, 2016.
DOI: 10.1007/978-3-319-54660-5_14

analytics on manufacturing-systems efficiency include a reduction of production cost and time across all manufacturing levels [1, 2]. Data scientists and manufacturing engineers often collaborate when using data analytics to solve process-specific problems to improve product quality [3, 4], equipment efficiency [5, 6], and resource efficiency [7, 8]. However, these collaborations require a significant amount of time and effort to merge the expertise from these two domains. In [9], the authors present a domain-specific framework to address this challenge. The framework (1) identifies the main components and interfaces that must be implemented to improve communication between these domains and (2) facilitates the application of data analytics in manufacturing. In this paper, we introduce an implementation of some of the components and interfaces that will be a part of this framework.

Our approach focuses on using data analytics – specifically neural networks (NNs) – for predicting a set of manufacturing-process-related performance metrics. There are three main contributions of this paper. First, we provide meta-models to represent manufacturing processes and NNs. Second, we describe an algorithm to generate a trained NN automatically from a manufacturing process model and data. Third, we discuss a tool to export the NN in two standard formats: the Predictive Model Markup Language (PMML) [10] and the Portable Format for Analytics (PFA) [11].

The paper is organized as follows. Section 2 presents the domain-specific knowledge required from the manufacturing and data-science domains to generate NNs for manufacturing processes. It also introduces the approach to generate NNs automatically. Section 3 describes, in more detail, two components of the proposed approach: a manufacturing meta-model and transformation rules to generate an NN. Section 4 presents a process-level manufacturing use case to illustrate the capabilities of the approach.

2 Domain Specific Knowledge from Neural Networks and Manufacturing Processes

In this section, we discuss the knowledge required from manufacturing engineers and data scientists to apply NNs to manufacturing processes. We review applications of NNs in manufacturing processes, and devise a methodology based on the common practice of data scientists.

2.1 Manufacturing Domain Knowledge

To identify the required manufacturing-domain knowledge, we studied several research efforts on the applications of data analytics (DA) to manufacturing processes. In [12], the authors apply analytics to detect faults in the alignment of a cap to the base part of a product. In [13–15], the authors predict product quality using three DA algorithms: Bayesian networks (BNs), linear regression, and NNs. In [16], the authors describe a way to predict the need for equipment repair using BNs. In [15], the authors used NNs to study surface roughness in a milling process. They identified surface roughness as the performance metric of interest. They also identified spindle speed, feed rate, depth of cut, and the vibration average per revolution as the process variables that have the

most impact on surface roughness. They collected 492 data samples to train and validate the NN. Each application followed a similar workflow: (1) identify the performance metric to be studied, (2) identify the variables that impact this target quantity, and (3) use test data to build an analytical model to predict the performance metric from the process variables. We used this same workflow in our work.

2.2 Data Science Domain Knowledge

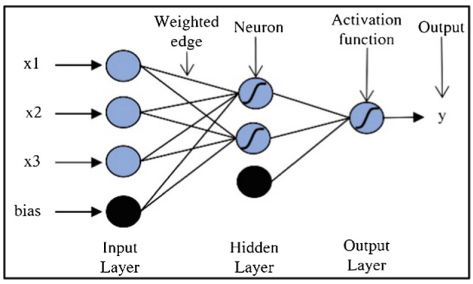

Fig. 1. Structure of a neural network

To understand the knowledge required from a data scientist to apply data analytics techniques to build an NN, it is important to understand how an NN is built. Figure 1 presents the main elements and the structure of an NN. An NN is composed of an input layer, zero or more hidden layers, and an output layer. Each layer contains at least one neuron. All layers except the output layer contain a bias neuron (shown in black). Weighted edges fully connect neurons in different layers. From a mathematical viewpoint, NNs can be viewed as a set of nonlinear basis functions (the activation functions), with free parameters (the adjustable weights). Training the NN is about adjusting the weights to minimize the error between the output value of the NN and the known, real, output value for a given data sample [17].

As noted above, the first step in building the NN involves selecting the input variables relevant to the performance metric. This step is called *feature selection* and defines the number of input neurons of the NN. There is one input neuron for each input variable. The second step is to determine the number of hidden layers and the number of neurons in each layer. In general, one hidden layer is sufficient [18] for the class of problems related to manufacturing processes. The number of hidden neurons has an impact on the NN accuracy, thus data scientists define this number very carefully. Finally, the output neuron represents the variable that we are trying to predict, which we call the quantity of interest. For example, a performance metric such as energy consumption may be the quantity of interest in a manufacturing scenario.

Based on these reviews, our approach needs to define the input neurons based on the process variables, define the optimal number of hidden neurons for an NN with one hidden layer, and finally define the output neuron for the quantity of interest.

2.3 Integration of Manufacturing and Data Science Domain Models

After identifying the required knowledge from manufacturing engineers and data scientists, we describe our approach and how it contributes to the framework defined in [9]. Figure 2 summarizes the workflow of our approach. In this figure, meta-models (Ⓐ and Ⓑ) are in gray, models (①, ③ and ⑤) are in yellow, and software solutions

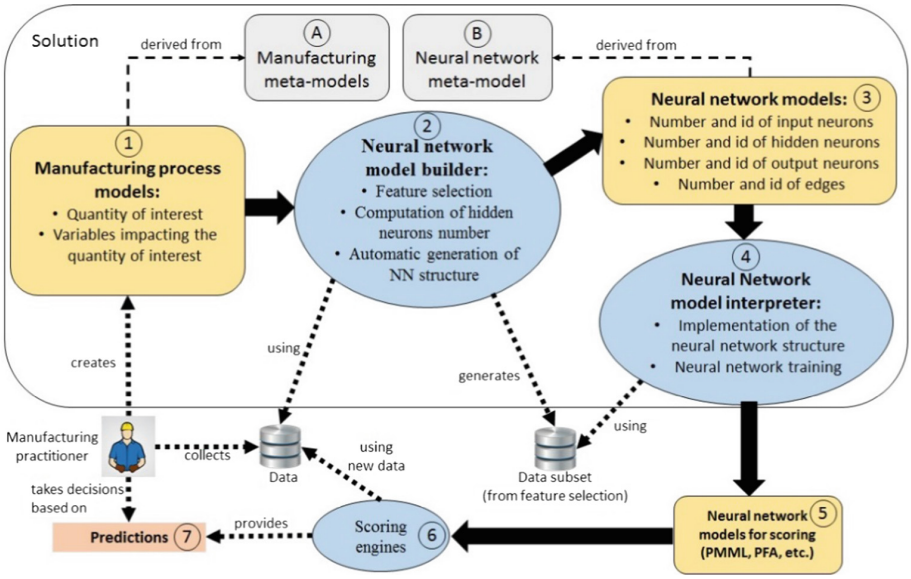

Fig. 2. Formal workflow of the approach

(②, ④ and ⑥) are in blue. The dashed arrows represent actions defined in the related label. The solid arrows show the use of models as input or output of the software solutions.

Box Ⓐ represents our manufacturing meta-model that captures the manufacturing knowledge. This meta-model defines the concepts, rules and constraints needed to represent a manufacturing process. Using the meta-model, a manufacturing engineer is able to build a manufacturing process model ① to define the quantities of interest and the variables involved in the manufacturing process. Note, we provide an interface to collect data related to the manufacturing process.

Taking the manufacturing process model and data as inputs, an NN model builder ② embeds a set of algorithms to run a feature selection that (1) optimizes the number of input neurons, (2) computes the optimal number of hidden neurons, and (3) builds the optimal structure of the NN ③. This NN structure is recorded using an NN meta-model contained in the meta-model repository. The NN meta-model and NN model interpreter are presented in [19]. The NN model interpreter ④ generates a trained NN. This NN is exported as a PMML or PFA file ⑤ that is ready to use for prediction with new data. A scoring engine ⑥ provides predictions ⑦ using the PMML file and new data. Scoring is the process of using a model to make predictions about the behavior of a quantity of interest. A manufacturing engineer makes decisions based on these predictions to control the manufacturing process under investigation.

3 Manufacturing Meta-models and Transformation Rules of the Neural Network Builder

In this section, we describe the components, Ⓐ, Ⓑ and ② in Fig. 2, to generate NNs from manufacturing process descriptions automatically. We also describe our implementations of these components.

3.1 Meta-model for Manufacturing Processes

A meta-model is a graphical description of concepts and their relationships, which can be used to describe objects or instances of those concepts in a particular domain. We developed a meta-model for describing manufacturing processes in a way that is helpful to build an NN. A manufacturing engineer builds a manufacturing model using the meta-model to provide the required knowledge identified in Sect. 2.1. Since the purpose of the approach is to use data-driven techniques (in this case NNs), there are no physics-based equations associated with the model. Figure 3 presents the main concepts of the manufacturing meta-model. Please note that this is a simple but a reasonable representation of the domain model. The notation in Figs. 3 and 4 is based on Unified Modeling Language Class Diagrams [20], where the rectangles represent concepts occurring in the domain, and the lines represent relationships between the concepts. A line with a solid diamond represents a containment relationship, with a numerical range at one end denoting the number of allowed instances. For example, in Fig. 3, a *ManufacturingModel* can contain 0 or more instances of *Manufacturing Process*.

The annotation <<Model>> is used to identify first class objects, while the annotation <<Connection>> is used to represent edges, flows, or associations.

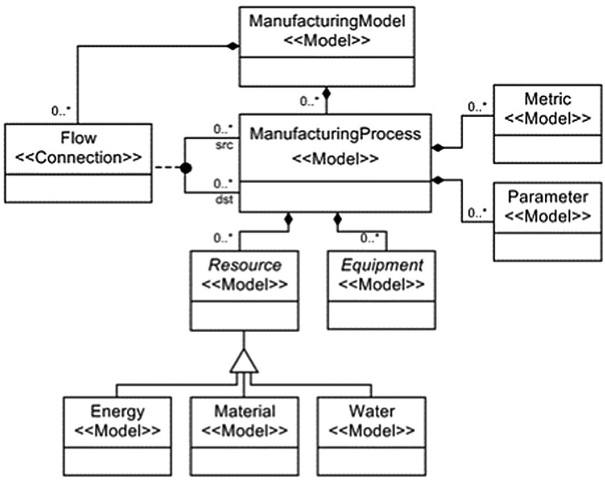

Fig. 3. Manufacturing meta-model

ManufacturingModel is a high level concept that allows the description of a manufacturing model that is composed of *Flows* and *ManufacturingProcess* concepts. The *Flow* concept represents connections between instances of the *ManufacturingProcess* concept. A *ManufacturingProcess* is composed of *Resource* and *Equipment* concepts, which allow the manufacturer to include resource or equipment parameters as variables of the manufacturing process. *ManufacturingProcess* also contains the concepts of *Parameter* and *Metric*. Metric is used to define a quantity of interest in the manufacturing process. Parameters are the variables that can impact the metric for a manufacturing process.

In the UML notation, an empty triangle is used to denote specialization, where one concept may be specialized into many sub-concepts. As shown in Fig. 3, the *Resource* concept is extended to define different types of resources: energy, water, and material. The manufacturing meta-model can also be extended to define other kinds of resources such as labor.

3.2 Meta-model for Neural Networks

Figure 4 shows the neural network meta-model (NNMM) presented in [19]. The NNMM represents different types of NNs through various abstractions. A *NeuralNetworkModel* concept is composed of *Neuron* and *Edge* concepts. A *Neuron* can be one of four types: *InputNeuron*, *HiddenNeuron*, *BiasNeuron*, and *OutputNeuron*. An *Edge* can be a *VisibleEdge* or a *HiddenEdge*. A *VisibleEdge* is used to represent an edge between an input neuron and a hidden neuron, between a hidden neuron and an output neuron, or between a bias neuron and an output neuron. Edges between two hidden neurons, or between a bias neuron and a hidden neuron are represented using *HiddenEdge*.

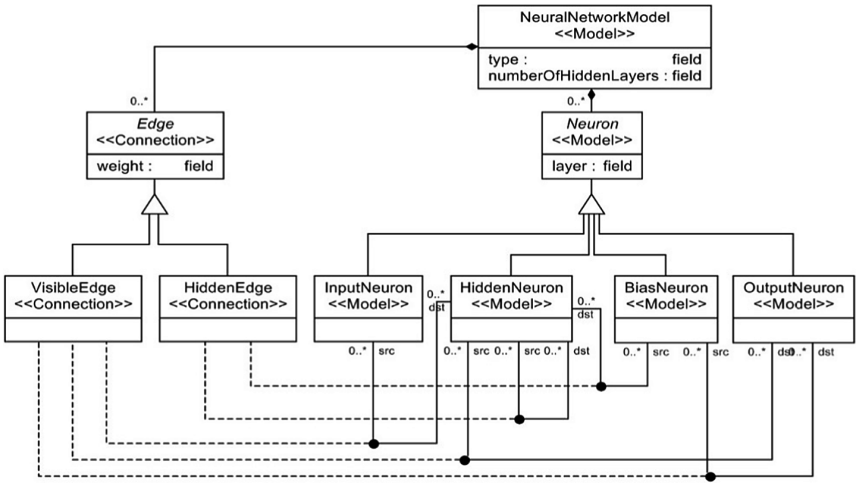

Fig. 4. Neural network meta-model [19]

3.3 Transformation Rules to Generate an NN from a Manufacturing Model

We developed a set of transformation rules to generate an NN model from a manufacturing model. Together, these rules represent a step-by-step process to build an NN from the input model and the data provided by a manufacturing engineer. We embedded these rules into the NN model builder, so that they can be applied to any type of manufacturing process model. The result of applying these rules is an untrained NN model, which is built based from the NN meta-model described above, and an input data set for training. Figure 5 presents the workflow and the transformation rules of the NN model builder, identified as ② in Fig. 2.

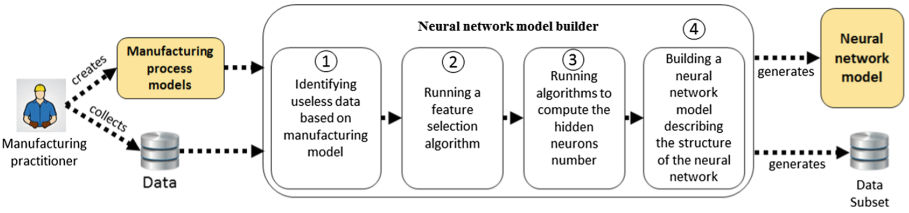

Fig. 5. Workflow of the NN model builder

The NN model builder takes the manufacturing process model and data provided by the manufacturing engineer as inputs. In the builder, Step 1 identifies those variables that the manufacturer listed as impacting the quantity of interest in the manufacturing model. The identified variables are compared with the variables present in the data set. The variables that are not identified in the manufacturing model are then removed. Step 1 takes advantage of the manufacturing expertise that the manufacturing engineer provides in the model.

Step 2 uses the feature-selection algorithm and a real data set to identify those variables that do not contribute to the quantity of interest based on a data set. This algorithm takes the variables provided from Step 1 and removes the variables that do not contribute from the list.

During Step 3, the builder runs an algorithm to optimize the number of hidden neurons for the NN. Several reports document the studies associated with optimizing the number of hidden neurons and putting them all into a single hidden layer. Sheela and Deepa [21], for example, analyzes the performance of the different optimization methods described in different reports – that is, their ability to predict the actual optimal number of hidden neurons. Our algorithm applies these different methods and computes the number that appeared most frequently. That number is the one selected for the NN. As we mentioned previously, one hidden layer is sufficient for manufacturing process-related problems, and the algorithm is implemented to build one hidden layer. This algorithm, however, can easily be modified to build NNs with more than one hidden layer.

In Step 4, the builder generates the NN instance model and a data set as outputs. The NN instance model describes the structure of the NN, and must be trained in order

to predict the quantity of interest. The output data set is a subset of the input data set. The input data set variables that do not impact the quantity of interest are not included in the output data set.

Using the output data set, the NN model interpreter [19] performs the NN training and updates the weights on the NN instance model. It also generates a PMML or PFA file containing the trained NN model.

4 Use Case

In this section, we show how our implementation of the proposed approach is used in a typical manufacturing scenario. For our approach, manufacturing engineers build a model of the process they wish to study using the meta-model described earlier. Next, they collect test data by conducting experiments or from other sources. Finally, they use the automated tools described in Fig. 2 to generate an NN for their process. This NN can be used to make future decisions without having to conduct physical experiments to determine target values.

4.1 Scenario Description

This case study focuses on predicting the energy consumed by a milling machine tool. The data set used in this case study was generated in [22] from a total of 18 parts machined with 196 face milling, 108 contouring, 54 slotting and pocketing, 16 spiraling and 32 drilling operations. We focused on face milling in this use case. The series of machining operations were performed. Data was collected using power meters and different sensors, and then stored in a database. We use the collected data as test data to build an NN model to predict power consumption for different combinations of machining parameters.

The test data includes the timestamp, power demand, feed rate, spindle speed, depth of cut, cutting direction, cutting strategy, cutting ratio, cutting volume, and length of cut in the 3 axis, referred as cutX, cutY and cutZ. As described below, we first built a manufacturing process model based on our case study, then identified the optimal process parameters and the metric of interest.

4.2 Building the Manufacturing Process Model

Figure 6 shows the manufacturing process model that we built for the milling process. This model contains the parameters that the manufacturing engineer has specified as contributing to the quantity of interest. In this model, we defined power as the quantity of interest. We defined feed rate, spindle speed, depth of cut, cutting ratio, cutting volume, cutX, cutY, and cutZ as parameters that impact the power consumption. During this step,

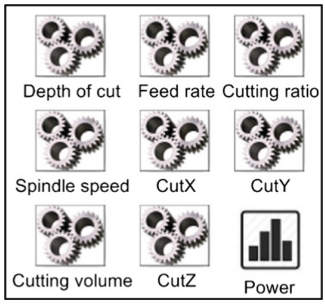

Fig. 6. Manufacturing model

the manufacturers use their domain expertise to list only those parameters that they think will have a significant impact on their power consumption. The test data and the manufacturing model are the inputs to our next step, which performs feature selection and generates the NN.

4.3 Generating the Neural Network for Prediction

In the next step, the manufacturing engineer executes the NN model builder using both the manufacturing model (in Fig. 6) and the test data as inputs. Our algorithm, takes those inputs and generates a trained NN. It does this using two pieces of software: the NN model builder and the NN model interpreter.

The NN model builder identifies the quantity of interest (the selected performance metric) from the manufacturing model. In this case, it is power. The NN model builder prunes the data set by removing the data that were omitted in the manufacturing model. In our case, it removes the cutting direction and cutting strategy variables from the data set since these are not present in the manufacturing model in Fig. 6. Next, the feature selection algorithm is executed. It uses the test data to identify and remove parameters that have an insignificant impact on the target variable. In our example, feed rate, depth of cut, cutX, and cutY were found not to have a significant effect on power; therefore, these parameters are not considered when building the NN.

The NN model builder then displays which variables were removed (1) based on the manufacturing model and (2) using the feature selection algorithm. Then the builder saves the new data set in a location identified by the manufacturing engineer. Figure 7 shows the resulting NN model, an automatically generated instance of the NNMM which is shown in Fig. 4.

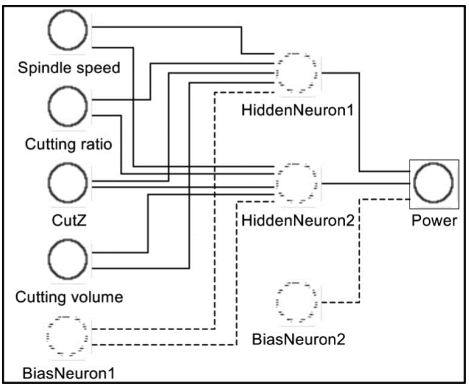

Fig. 7. Neural network model

In this NN model, the NN model builder keeps four variables: spindle speed, cutting ratio, cutting volume and cutZ. They are defined as input neurons in the NN. The algorithm computes that two hidden neurons are optimal in this model. Power is defined as the output neuron in the NN. Finally, the builder adds a bias neuron for every layer except the output layer to build a correct NN.

The NN model builder generates the structure model the NN. It still needs to be trained (i.e. weights must be assigned to the edges to make correct predictions). To generate the weights, the structural NN must be trained with the test data. The NN model and the test data are inputs to the NN model interpreter. The interpreter generates a trained NN based on the structure described in the NN model and the test data. The NN is generated as a standard PMML file. Several off-the-shelf data analytics tools can read this PMML file.

The manufacturer can now use this NN to predict the energy consumption of the milling machine under different conditions. This allows the manufacturer to perform different tests and make decisions, without having to physically execute experiments on the machine.

5 Summary and Future Work

In this paper, we proposed an approach to generate an NN to predict performance metrics for manufacturing processes. This approach provides capabilities to collect the required manufacturing knowledge and to use that knowledge to build NN models to predict the performance metrics for different values of the process parameters. This can be used to optimize performance by finding the best values for the process parameters.

We first reviewed the applications of data analytics to manufacturing processes for identifying the steps taken by data scientists to create NNs. We then developed and implemented the components needed to provide the capabilities required by this approach. Part of that approach is developing a manufacturing meta-model. The meta-model allows manufacturing engineers to provide a set of the most important process parameters – those have the most impact on performance – in a manufacturing model. In addition to this meta-model, we implemented an NN model builder to automatically build an NN model from a manufacturing model and data provided by manufacturing engineers. The NN model builder provides (1) a feature-selection algorithm based on the test data and (2) an NN model generator that generates the structure of the NN. From the generated NN structure, an NN model interpreter produces a trained NN in a standard format. Using a scoring engine, the trained NN can then be used to predict the quantity of interest.

We illustrated the capabilities of our implementation using a realistic manufacturing scenario. In this scenario, an NN is trained to predict energy use during a particular milling process. A manufacturing engineer provides a manufacturing model used as input to the NN builder. The implemented algorithms finally generate a trained NN that can be used with new data for predicting energy consumption.

This paper presented an initial description and implementation of an approach to generate predictive models for manufacturing applications. We implemented a translator (the NN model builder) to generate neural networks automatically. More translators will be implemented in future work to generate other types of predictive models. In practice, manufacturing processes and their interactions with their surrounding environment are complex. In order to generate reliable prediction models for practical scenarios, our meta-models and translators must be extended to account for other parameters and constraints that affect manufacturing processes. Future work lies in four directions. The first is to extend the manufacturing meta-model to enable the representation of problems in greater detail, and at different manufacturing levels such as assembly. Next, add new steps to the NN model builder to improve its accuracy. Third, include a scoring engine. Fourth, extend the framework to include different analytical techniques such as Bayesian networks. Capabilities to build BN models could enable the application of uncertainty quantification in manufacturing [23].

Acknowledgement. The research in this paper was supported by National Institute of Standards and Technology's Foreign Guest Researcher Program, and Cooperative Agreement No. 70NANB14H250.

References

1. Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Byers, A.H.: Big Data: The Next Frontier for Innovation, Competition, and Productivity. McKinsey (2011)
2. Brown, B., Chui, M., Manyika, J.: Are you ready for the era of 'big data'. McKinsey Q. **4**(1), 24–35 (2011)
3. Erzurumlu, T., Oktem, H.: Comparison of response surface model with neural network in determining the surface quality of moulded parts. Mater. Des. **28**(2), 459–465 (2007)
4. Zhai, L.Y., Khoo, L.P., Fok, S.C.: Feature extraction using rough set theory and genetic algorithms—an application for the simplification of product quality evaluation. Comput. Ind. Eng. **43**(4), 661–676 (2002)
5. Dabbas, R.M., Chen, H.N.: Mining semiconductor manufacturing data for productivity improvement—an integrated relational database approach. Comput. Ind. **45**(1), 29–44 (2001)
6. Chien, C.F., Diaz, A.C., Lan, Y.B.: A data mining approach for analyzing semiconductor MES and FDC data to enhance overall usage effectiveness (OUE). Int. J. Comput. Intell. Syst. **7**(sup2), 52–65 (2014)
7. Shin, S.J., Woo, J., Rachuri, S.: Predictive analytics model for power consumption in manufacturing. Procedia CIRP **15**, 153–158 (2014)
8. Gupta, D., Gopalakrishnan, B.: Energy sensitive machining parameter optimisation. Int. J. Ind. Syst. Eng. **5**(4), 405–423 (2010)
9. Lechevalier, D., Narayanan, A., Rachuri, S.: Towards a domain-specific framework for predictive analytics in manufacturing. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 987–995. IEEE (2014)
10. PMML v4.2.1. http://dmg.org/pmml/pmml-v4-2-1.html. Accessed 1 May 2016
11. PFA v0.8.1. http://dmg.org/pfa/index.html. Accessed 1 May 2016
12. Wolbrecht, E., D'ambrosio, B., Paasch, R., Kirby, D.: Monitoring and diagnosis of a multistage manufacturing process using Bayesian networks. Ai Edam **14**(01), 53–67 (2000)
13. Correa, M., Bielza, C., de Ramirez, M.J., Alique, J.R.: A Bayesian network model for surface roughness prediction in the machining process. Int. J. Syst. Sci. **39**(12), 1181–1192 (2008)
14. Abouelatta, O.B., Madl, J.: Surface roughness prediction based on cutting parameters and tool vibrations in turning operations. J. Mater. Process. Technol. **118**(1), 269–277 (2001)
15. Tsai, Y.H., Chen, J.C., Lou, S.J.: An in-process surface recognition system based on neural networks in end milling cutting operations. Int. J. Mach. Tools Manuf. **39**(4), 583–605 (1999)
16. Kurz, D., Kaspar, J., Pilz, J.: Dynamic maintenance in semiconductor manufacturing using Bayesian networks. In: 2011 IEEE Conference on Automation Science and Engineering (CASE), pp. 238–243. IEEE (2011)
17. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall (2004)
18. Heaton, J.: Introduction to Neural Networks with Java. Heaton Research Inc., St. Louis (2008)

19. Lechevalier, D., Hudak, S., Ak, R., Lee, Y.T., Foufou, S.: A neural network meta-model and its application for manufacturing. In: 2015 IEEE International Conference on Big Data (Big Data), pp. 1428–1435. IEEE (2015)
20. Unified Modeling Language. http://www.uml.org. Accessed 1 May 2016
21. Sheela, K.G., Deepa, S.N.: Review on methods to fix number of hidden neurons in neural networks. Mathematical Problems in Engineering (2013)
22. Park, J., et al.: A generalized data-driven energy prediction model with uncertainty for a milling machine tool using Gaussian Process. In: ASME 2015 International Manufacturing Science and Engineering Conference (2015)
23. Nannapaneni, S., Mahadevan, S.: Uncertainty quantification in performance evaluation of manufacturing processes. In: 2014 IEEE International Conference on Big Data (Big Data). IEEE (2014)