

MACHINE LEARNING PATHWAY FOR HARNESSING KNOWLEDGE AND DATA IN MATERIAL PROCESSING

Ning Sun

Metso, Shrewsbury, MA 01545, USA

Adam Kopper 

Mercury Marine, Fond du Lac, WI 54935, USA

Rasika Karkare

Data Science, WPI, Worcester, MA 01609, USA

Randy C. Paffenroth

Mathematical Sciences, Computer Science, Data Science, WPI, Worcester, MA 01609, USA

Diran Apelian 

Materials Science and Engineering, UCI, Irvine, CA 92967, USA

Copyright © 2020 American Foundry Society
<https://doi.org/10.1007/s40962-020-00506-2>

Abstract

Artificial intelligence (AI) is integral to Industry 4.0 and the evolution of smart factories. To realize this future, material processing industries are embarking on adopting AI technologies into their enterprise and plants; however, like all new technologies, there is always the potential for misuse or the false belief that the outcomes are reliable. The goal of this paper is to provide context for the application of machine learning to materials processing. The general landscapes of data science and materials processing are presented, using the foundry and the metal casting industry as an exemplar. The challenges that exist with typical foundry data are that the data are unbalanced, semi-supervised, heterogeneous, and limited in sample size.

Data science methods to address these issues are presented and discussed. The elements of a data science project are outlined and illustrated by a case study using sand cast foundry data. Finally, a prospective view of the application of data science to materials processing and the impact this will have in the field are given.

Keywords: Industry 4.0, machine learning, smart factory, IoT, artificial intelligence, classification models, random forest, XGBoost, unbalanced, semi-supervised, dimension reduction, principal component analysis, feature importance, data standardization

Introduction

The fourth industrial revolution that ushered the Internet of Things (IoT) and the Internet of Services (IoS) has come to be known as Industry 4.0. At the Hannover Messe in 2011, Germany launched a project called “Industrie 4.0” designed to fully digitize manufacturing. The larger vision of Industry 4.0 is the digital transformation of manufacturing, leveraging advanced technologies, and innovation accelerators in the convergence of IT (Information Technology) and OT (Operational Technology). The purpose is

to integrate connected factories within industry, decentralized and self-optimizing systems, and the digital supply chain in the information-driven cyber-physical environment of the fourth industrial revolution.^{1,2} The evolution toward Industry 4.0 is given in Figure 1.

The initial goals of Industry 4.0 typically have been automation, manufacturing process improvement, and productivity optimization. The more advanced goals are innovation and the transition to new business models and revenue sources using information technologies and

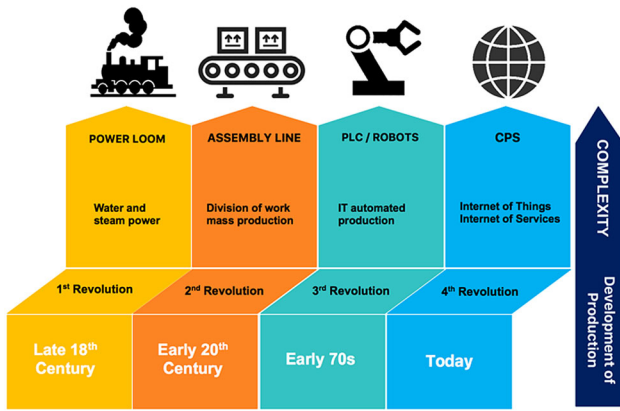


Figure 1. Industrial revolutions¹

services as cornerstones. These developments will transform manufacturing plants into smart factories or foundries. Three keystone digital technologies will enable the transformation to smart factories: (i) connectivity, which implies executing industrial IoT to collect data from various segments of the plant; (ii) intelligent automation which includes advanced robotics, machine vision, digital twins, and distributed control; and (iii) cloud-scale data management and analytics (AI and machine learning).³

In the metal processing field, particularly in the metal casting industry, whether it be ferrous or non-ferrous foundries, many data are collected at various locations within the plant. However, these data are usually siloed within operational departments without an intentional strategy for data fusion and transformation into knowledge. It is a fact that many of our plants and plant infrastructures were built prior to the rise of data science capabilities and tools. The time is now to make the transformation of our plants into smart factories in the context of Industry 4.0.

In this paper, our goal is to establish some context of AI and machine learning and how it can be appropriately utilized in materials processing where physical laws govern the process. We use metal casting as an example in this work as it is a well-established industry from which we have access to process data via the industrial membership of the Advanced Casting Research Center at UCI. In metal casting, the quality of the final product is influenced by many factors: metal composition, processing conditions, the solidification journey where transport phenomena influence the resultant microstructure, post-processing treatments, etc. Even with our understanding of the materials processing world, working with its manufacturing data is not without challenges.⁴ Because the industry is well established, the foundries do not produce components with many defects. In other words, scrap rates are low, making it difficult to utilize algorithms, based on supervised learning, which learn from successes and failures.⁵ This is where the need for using machine learning algorithms that can treat unbalanced data arises. Moreover, real-world

manufacturing processes are complex and appropriate data may not always be available for all parts. Accordingly, the need for advanced unsupervised or semi-supervised machine learning algorithms also exists.⁵ “The Landscape of Machine Learning” section describes these types of algorithms in detail. In this work, we want to show how these techniques can be used to answer the questions: *How can we develop algorithms and apply AI/machine learning to processes where one does not have many defective, or otherwise labeled, parts to teach and learn from?* It should be noted that the fundamentals and the principles presented here are applicable to a host of manufacturing processes.

The Landscape of Machine Learning

What is Machine Learning?

Machine learning is a branch of artificial intelligence (AI) where one constructs computer algorithms intended to mimic tasks commonly performed by humans. Algorithms for image recognition, health analytics, natural language processing, and self-driving vehicles are all examples of AI that have transformed industries that affect our daily lives.⁶ More specifically, AI clearly has a role to play in advanced manufacturing where there are myriad of tasks that could be automated by algorithms such as defect detection, process optimization, and new materials development.²

For many years, classical philosophers have attempted to describe human thinking as a symbolic system. Babbage in the 1830s realized that punched cards used in the Jacquard loom could control operations.⁶ Alan Turing in England (1935–1940 era) developed a machine that could compute using a set of rules transitions/states to solve mathematical functions.^{7,8} Subsequently, Turing went on to expand his view by posing the question: “Can a machine think”? The term AI was formally established in 1956 at a conference at Dartmouth College, Hanover, NH USA, by pioneers John McCarthy and Marvin Minsky. McCarthy challenged the community to make machines that “behave in ways that would be called intelligent if a human were so behaving,” whereas Minsky focused on making machines that would do things that “would require intelligence if done by men.”⁹

AI and machine learning are closely related to fields such as pattern recognition (an umbrella term that covers many different approaches), statistics and statistical learning (where the focus tends to be on formal mathematical relationships), and neural networks (a field which has seen great advancements in the past few years).¹⁰ For example, one class of approaches that was common in AI’s early years was that of rules-based systems. In a manufacturing plant, the convention has been that engineers develop a knowhow enabling them to detect defects in the final product; in turn, they pass on this knowledge to those who

follow their footsteps. It is tempting to distill how a human performs such tasks by enumerating a set of rules for defect recognition. Once such a collection of rules is developed, they can then be encoded in a computer language to allow a machine to mimic what a human does. Unfortunately, such rules-based systems tend to be quite fragile as the interactions in the system can be subtle. As a result, rules-based systems do not achieve human-level performance. The field of machine learning takes a different perspective by developing algorithms that learn by example. Rather than constructing handcrafted rules, in machine learning, one designs systems that can construct their own rules given a collection of examples where the desired task is performed correctly.

Elements of Machine Learning

Algorithms

Algorithms are the “machines” that can learn and generate the rules. It is reasonable to ask whether it is easier to write down explicit rules for a task or to create an algorithm that can generate its own rules. Perhaps counterintuitive, the latter is often much easier, more effective, and less error prone. Rulemaking algorithms abound, from simple linear regression, to the more complicated support vector machines, to cutting edge neural networks.^{5,11,12} As expected, algorithms are imperfect if the training data are inadequate. In machine learning, and specially in semi-supervised learning mode, one requires a large set of training data; without this, the algorithms developed may be unreliable.

Training Data

For algorithms to be effective, they require many examples to learn from. The question is: *Given the amount of training data, which algorithm is the most suitable?* The choice of algorithms is dependent upon the size of training data available. Using techniques such as cross-validation, we can test the performance of different algorithms in terms of generalizing on unseen test data.¹³ This can be done by training the algorithms on different subsets of the data and then testing on the rest. Choosing between algorithms should be based on comparing their performance on the test set.

Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create features that optimize machine learning algorithms.^{14,15} This is where technical prowess provided by the practitioner or the engineer plays

an important role. Feature engineering increases the predictive power of the algorithms by selecting specific features or creating new features from the data that assist in the learning process.

Feature engineering determines what information is given as input to the machine learning algorithm. The danger, however, is that one may carry this out and overengineer, in the engineering parlance, and overfit, in the machine learning parlance; there is a sweet spot for feature engineering. An example may be useful to explain the concept. Many parameters are collected during metal casting: alloy composition, environmental conditions in the foundry, superheat, temperature changes during the solidification process, etc. It is not unusual to have 40 columns of data for a given cast part. *Feature engineering helps us address how these parameters are communicated to the algorithm.* A close collaboration is needed in generating appropriate training datasets and the appropriate feature engineering by experts in both manufacturing and machine learning. To a large extent, the authors of this paper have formed such a team.

Data Pre-processing

Many machine learning algorithms require that their input data be numeric. In the example above, how should the chemical composition be represented numerically for a fair comparison with melt temperature and foundry environmental conditions? In the original training data, the amount of Si is 0.07 weight fraction, the melt temperature is 704 °C, and the temperature of the foundry is 24 °C. *Many machine learning algorithms depend on an appropriate definition of distance, and the rules they generate hinge on the distances between the training examples.* By setting $Si = 0.07$, $T_{melt} = 704$, and $T_{floor} = 24$, one is implicitly informing the algorithm that melt temperature is a more important parameter as compared to the composition of silicon or the temperature of the foundry. However, such inferences may be neither intended nor correct. In order to avoid such inferences, we pre-process the data such as normalizing the dataset so that all the columns are on the same scale.¹⁶ Details of how we normalize using a Z-transform are given in “[Process Cognition and Harnessing of Knowledge in Metal Casting](#)” section of this paper.¹⁷

Cross-Validation

One important part of machine learning that we have not yet touched upon is the evaluation of the performance of the algorithms we construct. Cross-validation is a technique that is used for algorithm evaluation on unseen data. *Cross-validation can be thought of as testing the algorithm*

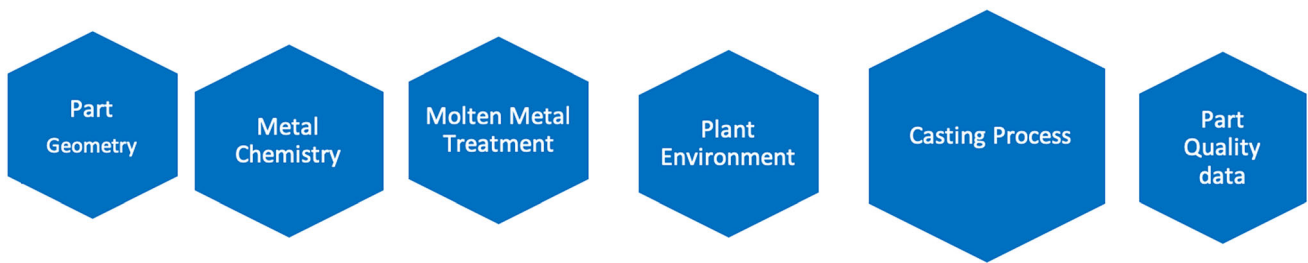


Figure 2. Data from various sources throughout the casting operation are kept in silos

in an environment that is faithful to how it will be used during the manufacturing process. For example, given a set of training data (e.g., labeled X-ray images of parts), one can train the algorithm in the task of detecting defective parts. When the algorithm is utilized on the factory floor, one would be interested in knowing how well it performs on images of parts as they roll off the assembly line, when the true label is not yet known.

There are important differences between how a machine learning algorithm performs on its training data and how it might perform in practice on the factory floor. For example, consider a machine learning algorithm that merely memorizes all the X-ray images in its training data and whether the image corresponds to a good part. Such an algorithm would be able to perfectly label every image in its training set but would have no ability to correctly label new images. In data science terms, it would not be able to generalize from its training data to new examples, such as the current production parts shipping from the foundry.

The machine learning terminology for such an algorithm that performs well on training data but fails to generalize is known as overfitting.¹⁸ Avoiding overfitting is an essential part of machine learning and a place where the expertise of machine learning practitioners can play a pivotal role. Constructing a machine learning algorithm that appears to be quite effective during training but fails in the field can be surprisingly easy to do. However, such situations are clearly to be avoided and require a measure of machine learning expertise.

The Landscape of Materials Processing: Metal Casting

Machine learning promises to have a transformative impact on the advanced manufacturing landscape, where applications of machine learning alongside the IoT is projected to generate \$1.2 to \$3.7 trillion of value globally by 2025.¹ In the metal casting industry, which is one of the core building blocks of advanced manufacturing industries, machine learning has only seen negligible adoption to date. Thus, there exists a huge opportunity to utilize AI and machine learning in the metal casting industry.^{19–24} The metal casting industry is at the cusp of its data revolution.

Modern foundries have the capability to capture a vast amount of process data on a daily basis.²⁵ These include molten metal preparation details, casting process data, simulation data, part geometry data (CAD files), nondestructive evaluation/testing data, etc. However, these many types of data from various sources throughout the operation are often kept in departmental silos where their value might have limited utility (Figure 2). Integrated data are the prerequisite for performing machine learning, and it is a lost opportunity for the foundry industry if no effort is made to compile, fuse, and analyze these data to better understand the process factors influencing the quality of the castings.

Implementation of machine learning in the metal casting industry requires *knowledge workers* who are trained in both data science and materials science and engineering domains.²⁶ Unfortunately, most engineers are not trained in data science. Efforts are underway in academia (e.g., WPI, UCI, University at Buffalo, Northwestern U., U. of Wisconsin, etc.) to develop curricula for engineering students who can navigate in both domains.

At the Advanced Casting Research Center (ACRC), a consortium consisting of 35 corporations has made a commitment to study how machine learning and deep learning can yield transformative improvements to metal casting processes. The long-term goal is to develop a framework that can be adopted by foundries to transform their data into process cognition and knowledge. In this project, the research team is multidisciplinary comprising of faculty and graduate students from data science as well as materials science and engineering. The data scientists apply their expertise in seeking or developing the effective and appropriate data analysis techniques. Material scientists and engineers determine how to treat anomalous data points in the raw dataset and can assess whether the predicted results and the feature importance are in line with observations on the shop floor.

Process Cognition and Harnessing of Knowledge in Metal Casting

In the following sections, we review some technical challenges and pitfalls in applying machine learning to

industrial foundry data and cover some potential solutions to resolve these challenges. Subsequently, we navigate the critical steps of machine learning as applied in a case study to showcase the implementation of machine learning to metal casting step by step.

Challenges of Metal Casting Datasets

Our team is in a fortunate position to have access to cast data from the industrial partners of ACRC. All the data are treated confidentially and are collected from three different casting processes—die casting, permanent mold, and sand casting. Though knowledge extracted directly from these front-line datasets can provide meaningful guidance on process and quality control to foundries, the process of converting these data into knowledge is quite challenging to our data scientists due to three notable attributes of foundry data as discussed below.

The Data are Unbalanced

In machine learning, the algorithm is designed to construct its own rules given a collection of examples. The aim is to develop a machine learning model that can predict the quality of cast components. The algorithm is trained by providing it with a large set of processing data, with each or some of the parts being labeled. The algorithm can construct its own set of rules for distinguishing between the labels. Based upon these examples, the algorithm applies the rules to make predictions on parts whose label is unknown. Ideally, the algorithm would learn from an approximately equal number of examples representing each label, however, in reality, the labels are unbalanced. The lifeblood of successful foundries is large-scale production of defect-free products. Accordingly, only a small percentage of defective products are available to train the machine learning algorithm. For example, in metal casting, the defect rate of a mature product can be as low as 2–5%,

which introduces significant challenges in developing and testing a robust predictive solution. Moreover, the generation of the quality data could be further complicated by the fact that it is too expensive to perform quality inspection for all of the products. As a result, while it is possible and straightforward to measure the processing data (the inputs to the machine learning model) of each casting, to generate the quality data (the response variable of the model) can be quite difficult. In sum, metal casting is an unbalanced, semi-supervised learning problem which is challenging for even state-of-the-art machine learning algorithms.

One of the main tasks in developing the machine learning model is to work meaningfully with unbalanced raw datasets supplied by foundries. As shown in Figure 3a, in a dataset containing 500 castings, only 6% of the total production is categorized as Class 3 and is considered defective. The population of good quality parts (Class 1 and Class 2) is much larger than that of the defective parts. Several algorithms were explored for data balancing. These algorithms can learn from the structure of the minority class in the original dataset and construct their own rules for generating new datapoints, or oversample. For illustration, an example of data balancing is shown in Figure 3. Employing such an algorithm can make the population of all three classes nearly equal.

Figure 3a, b shows the original and the oversampled data, respectively. The oversampling is done using a popular data balancing approach known as Synthetic Minority Oversampling Technique (SMOTE).²⁷ This approach is used when the number of samples in one class is significantly higher than the samples in the other classes, as is typical in manufacturing datasets. As the name suggests, this technique generates synthetic samples of the minority class by interpolating between two instances of the minority class. The oversampling is done until a point that the proportion of the minority class matches that of the majority class, and we have a balanced dataset for training. There are also variations of the SMOTE approach that can

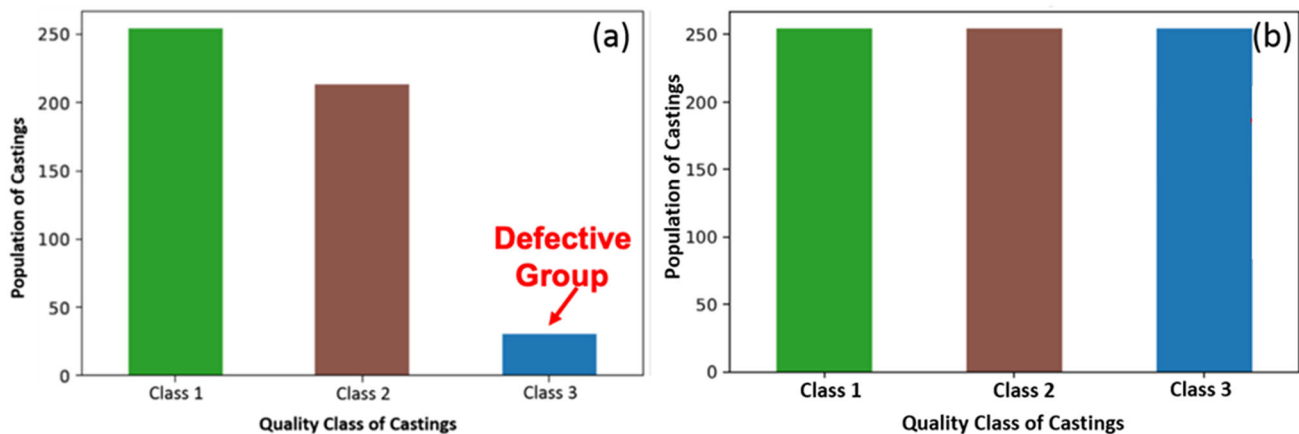


Figure 3. (a) Original and (b) oversampled casting data of each class

be used; for example, Borderline-SMOTE is widely used which focuses on the minority class samples that are at the border of the majority and the minority class, since these samples are more prone to misclassification errors as compared to those that are away from the border.²⁸

The Data are Sporadically Labeled

For example, a classic problem in machine learning would be detecting defects in X-ray images of manufactured parts. The algorithm is trained by providing it with a large set of images of parts, with each image being *labeled* by whether the quality of this particular part is acceptable. From a given set of labeled images, the algorithm learns and constructs its own set of rules for distinguishing between acceptable and non-acceptable parts and subsequently applies these rules to make predictions on *unlabeled* images. Labeled data mean that processing and quality data of the parts manufactured are known. In the machine learning literature, such methods are called supervised machine learning, where supervision arises from the availability of labeled training data. As shown in Table 1, for each individual sample, in the training dataset both the input variable (X) and its corresponding output variable (Y) are known. The algorithm can construct rules (e.g., $Y = f(x; \theta)$) to perform tasks such as predicting the quality of new parts. When labeled training data are not available, the machine learning problem becomes more difficult, and such algorithms are referred to as unsupervised machine learning. Whereas in semi-supervised machine learning only a fraction of the training data is labeled, both labeled and the unlabeled training data are used to develop the appropriate algorithm.

Most Metal Casting Data are Not ‘Big Data’

In our dataset, each individual row represents one cast component. The various columns in each row contain the recorded parameters when the cast component was produced. Unlike datasets generated from social media activities, the scale of metal casting dataset is quite small. Although more sensors can be installed to capture additional processing data during casting (to add more columns), the total number of rows in the dataset is still limited by the volume or production capability of the foundry. For instance, we have collected data from three casting manufactures over the past two years, depending upon the casting method and the size of the casting component, the total number of one part produced in one year varies from 300 to 7000 parts per year. Even if the foundry can save and extract 10 years of historical data, there would only be about 70,000 rows in this dataset, which is well short of being considered appropriate in the realm of “Big Data.”

Along with the oversampling techniques such as SMOTE, we can also use generative adversarial networks (GANs), a class of artificial intelligence algorithms, to generate rows of new data by learning the structure of the original data and generating new samples that follow the same distribution.^{29,30} The original application of this technique was to generate photographs with many realistic characteristics that were superficially authentic to human observers. Applying GANs to generate more datapoints in metal casting datasets has shown promise. Mixing synthetic and real data is one way to overcome the drawback of having a small-sized dataset. Synthetic data can be used to increase the volume of the data in case of small size datasets such as

Table 1. Classes of Machine Learning Tasks and Techniques

Supervised	Unsupervised		Semi-supervised				
	X	Y	X	Y			
<i>Training data</i>							
Sample 1	✓	✓	Sample 1	✓	Sample 1	✓	✓
Sample 2	✓	✓	Sample 2	✓	Sample 2	✓	✗
Sample 3	✓	✓	Sample 3	✓	Sample 3	✓	✗
Sample 4	✓	✓	Sample 4	✓	Sample 4	✓	✓
Sample 5	✓	✓	Sample 5	✓	Sample 5	✓	✗
Sample 6	✓		Sample 6	✓	Sample 6	✓	✓
$Y = f(x; \theta)$		Pattern of the data		$Y = f(x; \theta)$			
<i>Predictions</i>							
Sample 7	✓	✓	Sample 7	✓	Sample 7	✓	✓
Sample 8	✓	✓	Sample 8	✓	Sample 8	✓	✓
Sample 9	✓	✓	Sample 9	✓	Sample 9	✓	✓

these, and the real data are used so that it is faithful to the original dataset.

Case Study of Machine Learning in Metal Casting

The following paragraphs provide detail for training and evaluating machine learning algorithms on production foundry data. Analysis begins with an investigation of the training data. The output for prediction is a binary pass or fail rating of the porosity classification. SMOTE is applied to overcome the class imbalance between pass and fail samples, and the newly balanced training data are standardized. Several machine learning algorithms are trained on datasets with and without dimension reduction. Algorithm performance is evaluated with a metric of minimizing false negative classifications on the testing dataset.

The results shown below are generated using the scikit-learn v0.24,^{31,32} matplotlib v2.0.2,³³ and pandas v1.0.4^{34,35} libraries within the Python^{36,37} programming language.

Training Data

Table 2 shows a snapshot of a portion of the dataset collected from a sand cast foundry. This dataset has 510 rows and 28 columns; each individual row represents a cast component. The various columns in each row give the processing parameters captured: component ID, metal chemistry, casting processing details, and quality data (X-ray inspection results). All the cast components were inspected, and the quality results were labeled with varying levels depending upon the appearance of porosity. Class 1 indicates that the casting was porosity free, Class 2 indicates fine porosity, and Class 3 indicates large porosity

voids. A dummy variable was used to divide quality data into a binary quality condition as given in Eqn. 1.

$$Y = \begin{cases} \text{Pass} : & \text{if the quality level of the part is Class 1 or Class 2} \\ \text{Fail} : & \text{if the quality level of the part is Class 3} \end{cases} \quad \text{Eqn. 1}$$

Data Standardization

Since the physical meaning and the scale of all processing parameters incorporated into a given dataset vary significantly, the raw data in each column that represents a particular class need to be standardized to ensure the data are unitless and are of comparable scale. Once all processing parameters are incorporated into a given dataset, the data in each column are standardized using a statistical method, called the Z-transform,^{17,38} which converts the values in each column using the following equation:

$$Z_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j} \quad \text{Eqn. 2}$$

where

- Z_{ij} is the Z-transformed value of the parameter in one data cell
- X_{ij} is the original value of the parameter in the data cell
- μ_j is the mean of the original values of the parameter in the data column
- σ_j is the standard deviation of the original values of the parameter in the data column

Table 3 shows a snapshot of the dataset after normalizing using a Z-transform. Compared with the original dataset shown in Table 2, the values in each cell of the transformed dataset are on the same scale regardless of the physical

Table 2. A Snapshot Showing Portion of One Dataset Containing 510 Rows and 28 Columns

PourID	Temp_Floor	RH_Floor	Gr_Floor	LadleTemp	LadleDensityPP
6750	79	60	88.77	1320	2.639
6756	73	50	60.50	1333	2.644
6758	78	51	72.85	1330	2.637
6766	75	56	72.43	1332	2.638
6768	70	57	62.30	1333	2.635
6770	71	57	64.44	1330	2.636
6773	76	54	72.20	1327	2.635
6835	62	40	33.42	1332	2.641
6837	73	30	36.56	1338	2.644
6839	68	22	22.59	1340	2.642
6841	68	22	22.59	1338	2.639
6844	65	19	17.39	1335	2.644
6849	71	40	45.40	1330	2.640

Table 3. A Snapshot Showing Portion of the Dataset After Applying Z-Transform

PourID	Temp_Floor	RH_floor	Gr_floor	LadleTemp	LadleDensityPP
6750	1.34	1.48	2.03	- 1.69	- 0.48
6756	0.43	0.85	0.78	0.20	0.58
6758	1.19	0.91	1.33	- 0.23	- 0.90
6766	0.73	1.23	1.31	0.06	- 0.69
6768	-0.03	1.29	0.86	0.20	- 1.32
6770	0.12	1.29	0.95	- 0.23	- 1.11
6773	0.88	1.10	1.30	- 0.67	- 1.32
6835	- 1.26	0.21	- 0.42	0.06	- 0.05
6837	0.43	- 0.43	- 0.28	0.93	0.58
6839	- 0.34	- 0.94	- 0.90	1.22	0.16
6841	- 0.34	- 0.94	- 0.90	0.93	- 0.48
6844	- 0.80	- 1.13	- 1.13	0.49	0.58
6849	0.12	0.21	0.11	- 0.23	- 0.26

meaning and the scale of these processing parameters. The data are now unitless.

Dimension Reduction

All datasets collected from foundries are comprised of many columns regardless of the type of casting process. If the whole dataset were to be plotted on a scatter plot, this plot would need to have as many axes as the data have columns. However, the human perceptual system is designed to process three dimensions. As a result, foundry engineers will often have difficulty producing meaningful visual representations of their data. Fortunately, representing high-dimensional data in a low-dimensional space is a well-studied problem. In particular, principal component analysis (PCA) is a classic dimension reduction technique allowing us to blend a large set of correlated variables in the original dataset into a smaller number of newly created representative variables.³⁹⁻⁴¹ This is a powerful tool explored in our study to compress the dimensionality of the original dataset and allow visualization of the complicated dataset on a two- or three-dimensional plot whose axes correspond to the newly created principal components. We can then use these principal components as the predictors in the machine learning model in place of the original larger set of variables. We evaluated this technique against the original high-dimensional data and found that the dimension reduction via PCA was not necessary in this case study. However, PCA is an important method employed in many machine learning projects, so we offer the following detailed description.

Compared with the original dataset which contained 28 columns, the dataset is now represented with three newly created columns, PC1, PC2, and PC3; therefore, the

complicated dataset can be visualized with the three-dimensional plot shown in Figure 4b. The PCA plot is a scatter plot; in other words, PC1 is not a function of PC2 or PC3. These three components were used to display the dataset into several groupings of points. Each point in Figure 4 represents a cast component, and the position of the point is determined by all the input variables describing how the casting was manufactured. The output variable of the casting, in this case, the quality of the part (Class 1, 2, or 3), is marked by color.

The formation of clusters is most likely accounted for by the variations in production conditions. We investigated castings in the small cluster and found that they were manufactured in the last quarter of 2016. The cluster separation, most likely, is related to the seasonal changes when these parts were manufactured. This type of variation can more easily be detected once the data are visualized on a plot.

The singular value decomposition, or SVD, is a computational method often employed to calculate principal components for a dataset. Using SVD to perform PCA is efficient and numerically robust.⁴¹ The singular value plot of the dataset is shown in Figure 5. The *x*-axis of this plot represents the first six principal components, and the *y*-axis shows the singular values of these components. The singular values of these principal components are plotted in the order from largest to smallest. The statistical interpretation of singular values is in the form of variance in the data explained by the various components. It can be interpreted that if a component has a high singular value, it represents a high percentage of variance in the dataset.

As shown in Figure 5, the first principal component (PC1) and the second principal component (PC2), respectively, represent about 27% and 18% of the variance of the

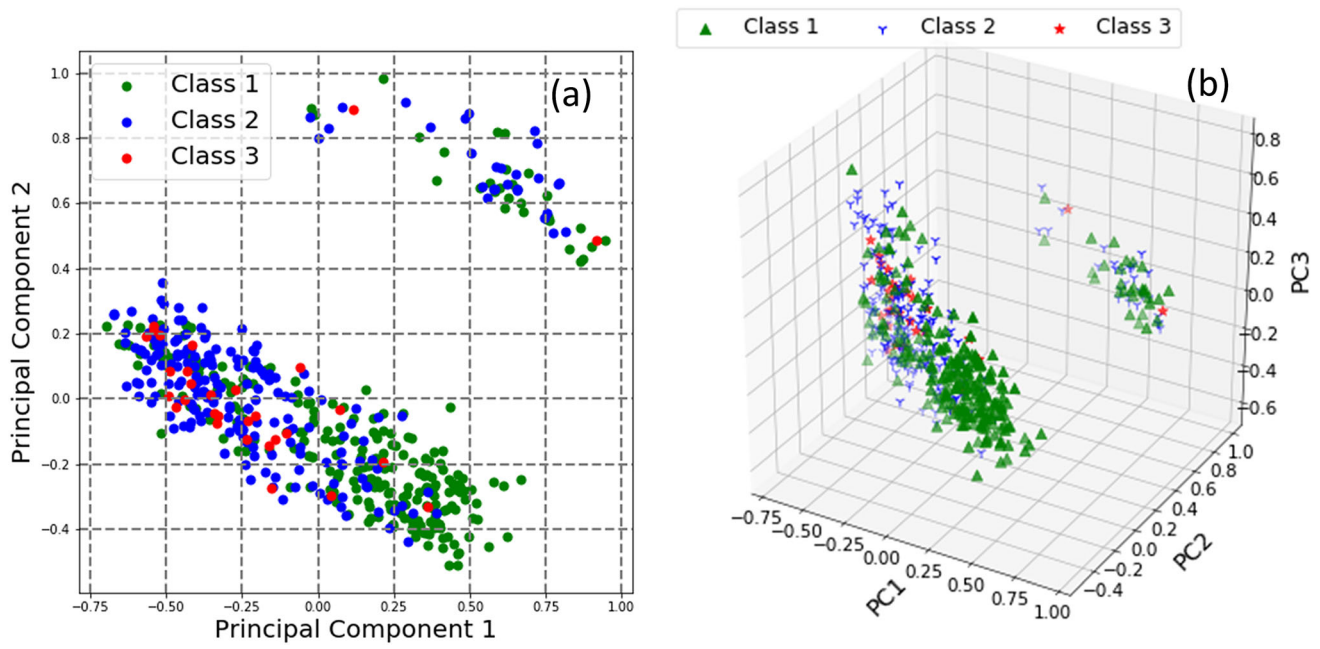


Figure 4. (a) Two-dimensional PCA plot with color-coded quality feature of the original dataset. (b) Three-dimensional PCA plot with color-coded quality feature of the original dataset

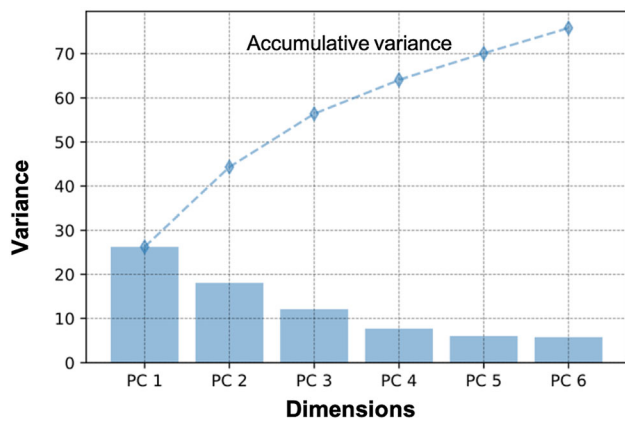


Figure 5. Singular value decomposition plot of the dataset

dataset. Since the first two principal components represent less than 50% of the original data, it is necessary to introduce more principal components to better capture the essence of the original dataset.

Machine Learning Classifiers for Quality Prediction

The main objective of the data analysis work is to develop a machine learning-based model to be used for part quality prediction. The performance of the models developed is evaluated by cross-validation. The complete dataset was divided into two sets of data, one for training the algorithms and the other set for testing the performance of the algorithms. The testing dataset is about 10% the size of the

original dataset. Since the quality of each casting in the test set is known, and the quality result is simplified into two possible classifications, “pass” or “fail,” the performance of the model is measured via four numbers obtained from applying the algorithm to the testing dataset. These numbers are called True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). They can be presented using a two by two matrix called a *confusion matrix*. In our study, since the “fail” class is more critical to the foundry operation, we call the “fail” class Positives (P) and the “pass” class Negatives (N). The confusion matrix we used to present the output of cross-validation in our study is shown in Table 4. If the model mistakenly predicted a bad part as a good part, it created a False Negative case. A model with good performance should give very few False Negatives, because the cost of this error would be high for the foundry.

Several algorithms were evaluated to predict part quality, and the confusion matrices of these algorithms are shown in Table 5.⁵ We use the SMOTE technique for oversampling and increasing the number of minority class samples in the dataset. We then use the oversampled data for training the classification algorithms such as random forest,⁴² logistic regression,⁴³ and support vector classifier (SVC).¹¹ Specifically, logistic regression is a machine learning algorithm that is used for performing classification tasks based on the logistic function using probabilities. For example, anything above a probability threshold of 0.5 is predicted as one class and anything below 0.5 is predicted as another. Random forest is a decision tree⁴⁴-based machine learning algorithm that is widely used in a number of classification as well as regression applications. Random

Table 4. Confusion Matrix to Visualize Model Performance

	Predicted # of good part	Predicted # of bad part
Actual # of good part	True negative	False positive
Actual # of bad part	False negative	True positive

Table 5. Several Algorithms and Their Confusion Matrix for Performance Evaluation

SMOTE + algorithm	Confusion matrix
Random forest ⁴²	$\begin{bmatrix} 22 & 5 \\ 6 & 14 \end{bmatrix}$
Logistic regression ⁴³	$\begin{bmatrix} 22 & 5 \\ 9 & 11 \end{bmatrix}$
Ensemble learning	$\begin{bmatrix} 23 & 4 \\ 8 & 12 \end{bmatrix}$
SVC ¹¹ (best performance)	$\begin{bmatrix} 23 & 2 \\ 0 & 19 \end{bmatrix}$

forest makes the prediction using the average of the predictions of the trees that build the forest in case of regression tasks and using the majority vote of the trees for label prediction in case of a classification task. SVC is a machine learning algorithm that is defined by a hyperplane that separates the classes in a dataset. It uses a labeled set of data as the training set and then categorizes new data on the correct side of the optimal separating hyperplane. Ensemble learning combines the predictions obtained using all the classifiers mentioned above and then makes a prediction based on the majority vote for a certain class for every sample in the test dataset. Combining SMOTE with SVC performed best as seen in Table 5. No False Negatives were assigned by this model with only two False Positives.

Employing this method in a production environment allows for targeted selection of production parts for detailed quality inspection. Instead of a random sampling of parts, the system can select suspect parts identified by a validated model.

Important Features Influencing Part Quality

In addition to making a quality prediction, another application of machine learning is the identification of critical features which are predicted to have the greatest influence on the quality of the product.⁴⁵ Some algorithms, for instance, random forest and other ensemble methods, can rank the various features (variables) in the dataset in terms of their importance to the predicted quality. Foundry

engineers can benefit from this function to identify parameters to monitor and control product quality.

For this case study, Figure 6 shows the rankings of feature importance relevant to the quality calculated by two machine learning algorithms, namely the XGBoost⁴⁶ and random forest. Like random forest, XGBoost is a decision tree-based algorithm that is used for classification and regression problems. These algorithms are used for finding the most important features in a dataset in terms of the label predictions for many applications. The two algorithms differ in the way that most important features are selected. XGBoost uses a criterion known as F-score to decide which features are the most important in terms of the label prediction. F-score for a feature is defined as the number of times that a feature in the dataset is used for prediction. Higher F-scores represent the most important features. Similarly, for random forest, permutation importance is used as the criterion for selecting the top features in the dataset. Permutation importance permutes, or randomly shuffles, the values of every feature in the dataset by taking one column at a time and checking by how much the predictions change. Moreover, if after permuting the values of a column in the dataset, the predictions change significantly, then that column is deemed as important in terms of the predictions. We check the feature importance using two different algorithms to compare and see if they agree with each other. Figure 6 shows the top four features found by using these two techniques and it can be seen that at least three of the top features found by these algorithms are common. The random forest determined environmental conditions such as the grains of moisture content in the air on the foundry floor (Gr_Floor), relative humidity (RH_Floor), and ambient temperature (Temp_Floor) in addition to the metal temperature in the ladle (LadleTemp) to be important factors in predicting casting quality. XGBoost replaces ambient temperature with the density of the metal inside the riser, or feeder. We validated these predictions using domain expertise of foundry engineers, and these were indeed the top features related to part quality according to domain experts. This is where machine learning can be exploited to target important inputs for better control over the casting process. Further, techniques such as feature importance can be used to drive designs of experiment, identify issues more quickly through targeted monitoring, and improve the overall cognition of the process.

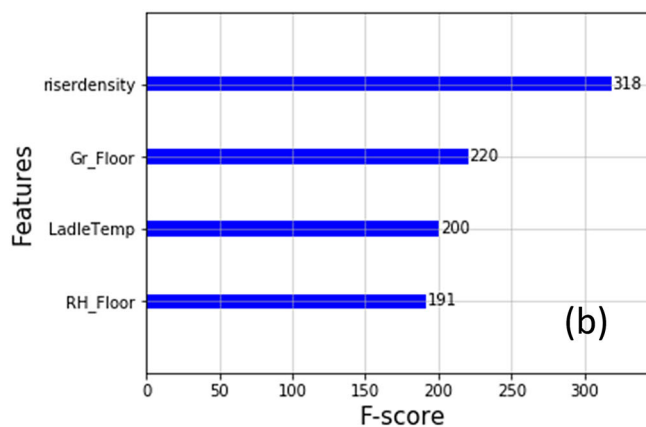
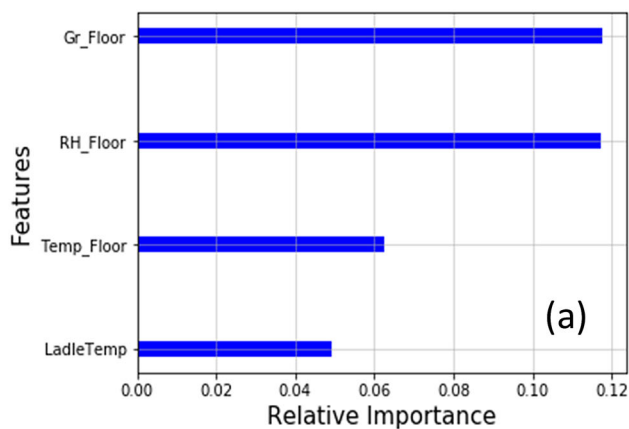


Figure 6. Feature importance relevant to quality by rankings: (a) generated by random forest and (b) generated by XGBoost

Prospective View of Machine Learning for Manufacturing

The abilities to gather data, mine it for knowledge, and apply the insights that we gain are transforming almost everything that we do as a society and manufacturing is no exception. On the other hand, modern automated factories are well-springs of data where more information can be measured about manufacturing processes than ever dreamed before. These two ideas make the manufacturing industry ripe for a data revolution in which quality can be improved, production can be accelerated, and waste can be minimized, if only the right collaborations between data scientists, material engineers, and the industrial sector can be achieved.

Modern machine learning, and deep learning in particular, provides tantalizing opportunities for progress, but the very power and generality of such data science methods bring along an important measure of responsibility. If used wisely, then such techniques allow for unprecedented improvements in the full flowering of Industry 4.0. If used unwisely, then the unbalanced, semi-supervised, and partially observed data that naturally arise in manufacturing problems, if not treated correctly from a data science and statistical perspective, can lead us astray. Perhaps as put best by the National Research Council of the National Academies⁴⁷:

Overlooking this foundation may yield results that are not useful at best, or harmful at worst. In any discussion of massive data and inference, it is essential to be aware that it is quite possible to turn data into something resembling knowledge when actually it is not. Moreover, it can be quite difficult to know that this has happened.

However, through the opportunity that we have had working with so many industrial partners of the ACRC,

who have generously worked with us and shared their data, we have at least helped begin a conversation on how best to use data science, machine learning, and deep learning in manufacturing problems. We see tremendous opportunities in improving the quality of cast components via the enabling tools we are developing, and there is also an implicit opportunity for major advances in planning for manufacturing and supply chain management. We are at the beginning of a revolution.

Concluding Comments

Almost over six decades ago, C. P. Snow wrote a critical essay titled “The Two Cultures”⁴⁸ that not only pointed out the gap between the sciences and the arts, but also the opportunities if we could cross the bridge between the two cultures. If he were alive today, C.P. Snow may have written about the “Three Cultures”—the Arts, Sciences, and Engineering/Manufacturing. The authors of this paper are a good example of individuals from “three cultures” who have worked together and learned much from each other. However, to do so required emotional as well as time commitments and investments. The dividends those investments have paid for the authors have been invaluable and impactful. In a similar way, the execution of the Fourth Industrial Revolution will require a cultural diffusion and much discourse between data scientists and manufacturing engineers. There is no question that future of work will be transformed in the twenty-first century, as well as the future of the worker. But as has been stated before, the future is for us to make.

Acknowledgements

The authors would like to thank the ACRC consortium members for their support and data for this project.

REFERENCES

1. “Industry 4.0: the fourth industrial revolution- guide to Industrie 4.0.” <https://www.i-scoop.eu/industry-4-0/>. Accessed May 26, 2020.
2. K.-D. Thoben, S. Wiesner, T. Wuest, BIBA – Bremer Institut für Produktion und Logistik GmbH, the University of Bremen, Faculty of Production Engineering, University of Bremen, Bremen, Germany, and Industrial and Management Systems Engineering, “‘Industrie 4.0’ and Smart Manufacturing—A Review of Research Issues and Application Examples. *Int. J. Autom. Technol.* **11**(1), 4–16 (2017). <https://doi.org/10.20965/ijat.2017.p0004>
3. Capgemini Consulting Group, Industry_4.0_The_Capgemini_Consulting_V.pdf. Capgemini, 2014, [Online]. Available: https://www.capgemini.com/consulting/wp-content/uploads/sites/30/2017/07/capgemini-consulting-industrie-4.0_0_0.pdf.
4. T. Prucha, From the Editor - Big Data. *Int. J. Met.* **9**(3), 5 (2015)
5. J. Friedman, R. Tibshirani, T. Hastie, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer, Berlin, 2001)
6. L. Hauser, Internet encyclopedia of philosophy, *Artificial Intelligence*. <https://www.iep.utm.edu/art-inte/>. Accessed 26 May 2020.
7. A. M. Turing, I.—COMPUTING MACHINERY AND INTELLIGENCE, *Mind*, vol. LIX, no. 236, pp. 433–460, 1950, <https://doi.org/10.1093/mind/LIX.236.433>.
8. C. Bernhardt, *Turing’s Vision—The Birth of Computer Science* (MIT Press, Cambridge, 2016)
9. J. McCarthy, M. Minsky, N. Rochester, C.E. Shannon, A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. Aug. 31, 1955, Accessed 17 Feb 17 2020. [Online]. <https://wvww.aaai.org/ojs/index.php/aimagazine/article/view/1904>.
10. K.P. Murphy, *Machine Learning: A Probabilistic Perspective* (MIT Press, Cambridge, MA, 2012)
11. Y. Zhu, Y. Zhang, The study on some problems of support vector classifier, *Comput. Eng. Appl.*, 2003, [Online]. Available: https://en.cnki.com.cn/Article_en/CJFDTotal-JSGG200313011.htm.
12. M.W. Craven, J.W. Shavlik, Using neural networks for data mining. *Data Min.* **13**(2), 211–229 (1997). [https://doi.org/10.1016/S0167-739X\(97\)00022-8](https://doi.org/10.1016/S0167-739X(97)00022-8)
13. J.D. Rodriguez, A. Perez, J.A. Lozano, Sensitivity analysis of k-Fold cross validation in prediction error estimation. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(3), 569–575 (2010). <https://doi.org/10.1109/TPAMI.2009.187>
14. C. Reid Turner, A. Fuggetta, L. Lavazza, A.L. Wolf, A conceptual basis for feature engineering. *J. Syst. Softw.* **49**(1), 3–15 (1999). [https://doi.org/10.1016/S0164-1212\(99\)00062-X](https://doi.org/10.1016/S0164-1212(99)00062-X)
15. A. Zheng, A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists* (O-Reilly, Beijing, 2018)
16. I. Gibson, C. Amies, Data normalization techniques, 6259456, 10 Jul 2001.
17. Z-Transform, *Wolfram MathWorld*. <https://mathworld.wolfram.com/Z-Transform.html>. Accessed 26 May 2020.
18. W.M.P. van der Aalst, V. Rubin, H.M.W. Verbeek, B.F. van Dongen, E. Kindler, C.W. Günther, Process mining: a two-step approach to balance between underfitting and overfitting. *Softw. Syst. Model.* **9**(1), 87 (2008). <https://doi.org/10.1007/s10270-008-0106-z>
19. J.K. Kittur, G.C. ManjunathPatel, M.B. Parapagoudar, Modeling of pressure die casting process: an artificial intelligence approach. *Int. J. Met.* **10**(1), 70–87 (2016). <https://doi.org/10.1007/s40962-015-0001-7>
20. E. Kocaman, S. Şirin, D. Dispınar, Artificial neural network modeling of grain refinement performance in AlSi10Mg alloy. *Int. J. Met.* **20**, 20 (2020). <https://doi.org/10.1007/s40962-020-00472-9>
21. P.K.D.V. Yarlagadda, E. Cheng Wei Chiang, A neural network system for the prediction of process parameters in pressure die casting. *J. Mater. Process. Technol.* **89–90**, 583–590 (1999). [https://doi.org/10.1016/S0924-0136\(99\)00071-0](https://doi.org/10.1016/S0924-0136(99)00071-0)
22. J.K. Rai, A.M. Lajimi, P. Xirouchakis, An intelligent system for predicting HPDC process variables in interactive environment. *J. Mater. Process. Technol.* **203**(1–3), 72–79 (2008). <https://doi.org/10.1016/j.jmatprotec.2007.10.011>
23. A. Krimpenis, P.G. Benardos, G.-C. Vosniakos, A. Koukouvitaki, Simulation-based selection of optimum pressure die-casting process parameters using neural nets and genetic algorithms. *Int. J. Adv. Manuf. Technol.* **27**(5–6), 509–517 (2006). <https://doi.org/10.1007/s00170-004-2218-0>
24. J. Zheng, Q. Wang, P. Zhao, C. Wu, Optimization of high-pressure die-casting process parameters using artificial neural network. *Int. J. Adv. Manuf. Technol.* **44**(7–8), 667–674 (2009). <https://doi.org/10.1007/s00170-008-1886-6>
25. D. Blondheim, Artificial intelligence, machine learning, and data analytics: understanding the concepts to find value in die casting data, presented at the 2020 NADCA Executive Conference, Clearwater Beach, FL, 25 Feb 2020.
26. T. Prucha, From the editor: AI needs CSI: common sense input. *Int. J. Met.* **12**(3), 425–426 (2018). <https://doi.org/10.1007/s40962-018-0235-2>
27. R. Blagus, L. Lusa, SMOTE for high-dimensional class-imbalanced data. *BMC Bioinform.* **14**(1), 106 (2013). <https://doi.org/10.1186/1471-2105-14-106>
28. H. Han, W.-Y. Wang, B.-H. Mao, Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning, in *Advances in Intelligent Computing*, vol. 3644, D.-S. Huang, X.-P. Zhang, and G.-B. Huang, Eds. Berlin: Springer, 2005, pp. 878–887.

29. A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: an overview. *IEEE Signal Process. Mag.* **35**(1), 53–65 (2018). <https://doi.org/10.1109/MSP.2017.2765202>
30. I. Goodfellow, NIPS 2016 Tutorial: Generative Adversarial Networks, *ArXiv170100160 Cs*, Apr. 2017. Accessed 27 May 2020. [Online]. Available: <https://arxiv.org/abs/1701.00160>.
31. F. Pedregosa et al., Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011). <https://doi.org/10.1016/j.patcog.2011.04.006>
32. A. Geron, *Hands-On Machine Learning with Scikit-Learn and Tensor Flow*, 1st edn. (O'Reilly, Beijing, 2017)
33. J.D. Hunter, Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.* **9**(3), 90–95 (2007). <https://doi.org/10.1109/MCSE.2007.55>
34. The pandas development team, *pandas-dev/pandas: Pandas*. Zenodo, 2020.
35. W. McKinney, Data structures for statistical computing in Python, in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 51–56, Accessed 09 Jan 2020. [Online]. Available: <https://conference.scipy.org/proceedings/scipy2010/pdfs/mckinney.pdf>.
36. G. Van Rossum, F.L. Drake, *Python 3 Reference Manual* (CreateSpace, Scotts Valley, CA, 2009)
37. T.E. Oliphant, Python for scientific computing. *Comput. Sci. Eng.* **9**(3), 10–20 (2007). <https://doi.org/10.1109/MCSE.2007.58>
38. T. Wuest, D. Weimer, C. Irgens, K.-D. Thoben, Machine learning in manufacturing: advantages, challenges, and applications. *Prod. Manuf. Res.* **4**(1), 23–45 (2016). <https://doi.org/10.1080/21693277.2016.1192517>
39. C. Eckart, G. Young, The approximation of one matrix by another of lower rank. *Psychometrika* **1**(3), 211–218 (1936). <https://doi.org/10.1007/BF02288367>
40. H. Abdi, L.J. Williams, Principal component analysis: principal component analysis. *Wiley Interdiscip. Rev. Comput. Stat.* **2**(4), 433–459 (2010). <https://doi.org/10.1002/wics.101>
41. S. Wold, K. Esbensen, P. Geladi, Principal component analysis. *Chemom. Intell. Lab. Syst.* **2**, 37–52 (1987). [https://doi.org/10.1016/0169-7439\(87\)80084-9](https://doi.org/10.1016/0169-7439(87)80084-9)
42. M. Pal, Random forest classifier for remote sensing classification. *Int. J. Remote Sens.* **26**(1), 217–222 (2005). <https://doi.org/10.1080/01431160412331269698>
43. R.E. Wright, Logistic regression, in *Reading and Understanding Multivariate Statistics*, Washington, DC, US: American Psychological Association, 1995, pp. 217–244.
44. D. Dietrich, B. Heller, B. Yang, *Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*, 1st edn. (Wiley, Hoboken, 2015)
45. A. Altmann, L. Toloşi, O. Sander, T. Lengauer, Permutation importance: a corrected feature importance measure. *Bioinformatics* **26**(10), 1340–1347 (2010). <https://doi.org/10.1093/bioinformatics/btq134>
46. T. Chen, C. Guestrin, XGBoost: a scalable tree boosting system. in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 785–794, Aug. 2016, <https://doi.org/10.1145/2939672.2939785>.
47. National Research Council, *Frontiers in Massive Data Analysis* (National Academies Press, Washington, D.C., 2013)
48. C.P. Snow, *The Two Cultures* (Cambridge University Press, London, 1959)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.