A methodology to support online monitoring and control of complex processes in the foundry industry using machine learning

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Maschinenbau und Verfahrenstechnik der
Universität Duisburg-Essen
zur Erlangung des akademischen Grades

eines

Doktors der Ingenieurwissenschaften

Dr.-Ing.

genehmigte Dissertation

von

Saad Bashir Alvi

aus

Rawalpindi, Pakistan

Gutachter:

Univ.-Prof. Dr. rer. nat. Johannes Gottschling, Universität Duisburg-Essen
Prof. Dr. rer. nat. Robert Martin, Universität Duisburg-Essen

Tag der mündlichen Prüfung: 06. Marz 2023

Abstract

Small and medium foundry industries strive to produce defect free parts and are planning to use the technological advancement to incorporate automation in their production processes. These targets require online monitoring and control of a foundry process. In case of an expected defect, the goal is to know about it in advance. If a monitoring system is in place and informs about the problem before occurring, corrective measures can be taken in time to prevent producing defected parts. This results in high production quality, continuous running of these processes, and minimization of production time and scrap rate. One of the possible ways is to capture the process related data and implicitly learn the unknown hidden model using machine learning methods. Technological advancement has enabled to produce the vast amount of data related to the processes and many promising machine learning methods are available. Utilizing these resources can help learn a data-driven model of the target process. This model can be used for monitoring and control of the target process. Hence in the current work, a novel framework based on data-driven machine learning model has been introduced and is applied on the different datasets of the foundry industry. The proposed framework is composed of three stages. In the first stage, a good data-driven machine learning model is obtained which can be used to monitor the target process by predicting the label of the current inputs. In the second stage, a knowledge-base is created using the model learnt to control running process. In the third stage, the learnt model is used to predict the part quality. If the quality meets the requirements, the production process continues and in case the prediction shows faulty part, then changes are proposed using the knowledge-base. The promising verification results obtained for the foundry industry datasets confirm that the proposed unified framework can enable the production processes to run more efficiently by reducing the manufacturing defects and down time. This can be achieved by preempting a potential problem and proposing a cost effective solution.

Kurzfassung

Kleine und mittlere Gießereien streben danach, fehlerfreie Teile zu produzieren und planen, den technologischen Fortschritt zu nutzen, um Automatisierung in ihre Produktionsprozesse zu integrieren. Diese Ziele erfordern eine Online-Überwachung und -Kontrolle des Gießereiprozesses. Das Ziel ist es, im Falle eines zu erwartenden Fehlers diesen im Voraus zu erkennen. Wenn ein Überwachungssystem vorhanden ist und über das Problem informiert, bevor es auftritt, können rechtzeitig Korrekturmaßnahmen ergriffen werden, um die Produktion fehlerhafter Teile zu verhindern. Dies führt zu einer hohen Produktionsqualität, einem kontinuierlichen Ablauf der Prozesse und einer Minimierung der Produktionszeit und Ausschussrate. Eine Möglichkeit besteht darin, die prozessbezogenen Daten zu erfassen und das unbekannte verborgene Modell mit Hilfe von Methoden des maschinellen Lernens implizit zu erlernen. Der technologische Fortschritt hat es ermöglicht, große Mengen an prozessbezogenen Daten zu erzeugen, und es stehen viele vielversprechende Methoden des maschinellen Lernens zur Verfügung. Die Nutzung dieser Ressourcen kann helfen, ein datengesteuertes Modell des Zielprozesses zu erlernen. Dieses Modell kann für die Überwachung und Steuerung des Zielprozesses verwendet werden. Daher wurde in der vorliegenden Arbeit ein neuartiges Framework auf der Grundlage eines datengesteuerten maschinellen Lernmodells eingeführt und auf verschiedene Datensätze der Gießereiindustrie angewandt. Das vorgeschlagene Framework besteht aus drei Stufen. In der ersten Phase wird ein datengesteuertes maschinelles Lernmodell erlernt, das zur Überwachung des Zielprozesses durch Vorhersage der abhängigen Variablen anhand der aktuellen Eingaben verwendet werden kann. In der zweiten Phase wird mit Hilfe des erlernten Modells eine Wissensdatenbank zur Steuerung des laufenden Prozesses erstellt. In der dritten Stufe wird das gelernte Modell zur Vorhersage der Teilequalität verwendet. Wenn die Qualität den Anforderungen entspricht, wird der Produktionsprozess fortgesetzt, und wenn die Vorhersage ein fehlerhaftes Teil ergibt, werden anhand der Wissensdatenbank Änderungen vorgeschlagen. Die vielversprechenden Verifizierungsergebnisse, die anhand von Datensätzen aus der Gießereiindustrie erzielt wurden, bestätigen, dass das vorgeschlagene einheitliche Framework einen effizienteren Ablauf der Produktionsprozesse ermöglichen kann, indem er die Herstellungsfehler und Ausfallzeiten reduziert. Dies wird dadurch erreicht, dass ein potenzielles Problem bereits im Vorfeld erkannt und eine kosteneffiziente Lösung vorgeschlagen wird.

Contents

List of Figures

List of Tables

List of Abbreviations

ANFIS	Adaptive-Network-based Fuzzy Inference System
ANN	Artificial Neural Networks
ANOVA	Analysis of Variance
CGI	compacted graphite iron
CMNB	Combination Method Naive Bayes
CNC	Computer Numerical Control
DoE	Design of Experiments
FFT	Fast Fourier Transform
GBDT	Gradient Boosting Decision Trees
GJL	lamellar graphite cast iron
LR	Logistic Regression
GJS	nodular graphite cast iron
H test	Kruskal-Wallis test
IID	Independently and Identically Distributed
IIoT	Industrial Internet of Things
IoT	Internet of Things
Kappa	Cohens Kappa statistic
KNN	k-Nearest Neighbours
KPCA	Kernel Principal Component Analysis
LCL	Lower Control Limit
MCC	Matthews correlation coefficient
ML	Machine Learning
MLR	Multiple Linear Regression
MPF	Meta Prediction Function
PaaS	Platform as a Service
PCA	Principal Component Analysis
RFID	Radio Frequency IDentification
RMSE	Root Mean Square Error

ROC	Receiver Operating Characteristic
RRMSE	Relative Root Mean Square Error
SaaS	Software as a Service
SMAPE	Symmetric Mean Absolute Percentage Error
SMEs	Small and Medium-sized Enterprises
SMOTE	Synthetic Minority Oversampling Technique
SPC	Statistical Process Control
SVD	Singular Value Decomposition
SVC	Support Vector Classification
SVM	Support Vector Machine
SVR	Support Vector Regression
UCL	Upper Control Limit

1 Introduction

1.1 Motivation

Nowadays, many companies strive to include optimized green and smart manufacturing techniques into their production processes. The central target of green manufacturing is to mitigate environmental damage and climate change by employing energy efficient techniques, by recycling and reusing materials, and by controlling the exhaustion level of CO_2 emissions [23]. Smart manufacturing, on the other hand, introduces various cutting-edge information and communication technologies and combines them with existing production techniques to provide real-time decision making support in engineering applications [56], leading to improved adaptability of industrial processes as well as rapid design changes [71].

The foundry industry, in particular, has long been striving to achieve these targets. The production of a casting part consist of a complex process composed of several sub-processes (core manufacturing, melting, molding, casting). These interdependent sub-processes work in a sequence to produce the required casting. Due to growing global competition, casting companies are seeking ways to automate this complex process and to achieve high quality products with lowered production time, all while reducing the environmental impact of the casting process. These high demands can only be met with well-controlled, highly optimized process parameters which, in particular, need to ensure a sufficiently low scrap rate.

In order to achieve these goals, a comprehensive understanding of materials and their properties as well as extensive knowledge of process-related casting defects are required. The casting process needs to be continuously monitored in order to detect deviations and, ideally, to identify possible casting errors before they occur so that the process parameters can be changed accordingly. To this end, it is necessary to employ a continuous *prediction* of the outcome, i.e. of the casting part's quality, throughout the process. This prediction needs to be based on the current state of the process, including both the control variables and measurable external parameters which might influence the result of the casting. Of course, such a prediction needs to be highly reliable in recognizing defects or other quality issues sufficiently early.

Two important types of models can be used to realize such a prediction function: *Analytical* and *data-driven* process models. Analytical process models, i.e. classical mathematical functions or numerical algorithms describing the process, are not always suitable for online monitoring of complex processes because they do not allow for a holistic process description: While companies like the MAGMA Gießereitechnologie GmbH offer numerical

simulations of foundry processes, such classical simulations are limited in their applicability to real-world foundries due to the large number of (known or unknown) parameters which influence the casting in a particular foundry environment and which, in practice, cannot be fully reflected in a generic analytical simulation. On the other hand, data-driven process models allow for a more holistic online process diagnosis. Such models are obtained by applying machine learning techniques to a particular foundry's actual process data. Although detailed expert knowledge is not required to obtain these models, an expert can supervise and rate the trained model to assess its reliability. A major limitation of data-driven methods is their dependence on sufficiently large amounts of data which accurately represents the target process; in particular (cf. Section 2.3.1), the application of supervised learning requires not only an extensive record of process parameters and the quality of produced parts, but the latter needs to be assigned reliably to the former. In practice, the lack of recorded data is therefore often a limiting factor for the applicability of machine learning methods in industrial settings.

Fortunately, recent technological advancements have led to the automated capturing of huge amounts of data from all aspects of life. Due to this recent development, data-driven methods have become more prominent objects of research, as have combinations between data-driven and analytical models. These novel methods have shown a tremendous success in various industrial fields. While some advances have already been experienced in foundry industry as well, data-driven techniques are not yet widely in use for the optimization of casting processes, leaving room for potential major improvements. The German foundry industry, in particular, is facing major challenges related to employing such *smart* methods: Since most German companies in this field are *Small and Medium-sized Enterprises* (SMEs), they often do not have the expertise necessary for developing their own machine learning solutions, or even lack a dedicated IT department all together. These companies have often not incorporated extensive data collection techniques in their processes yet; even in foundries where process data is measured systematically, there is often no centralized data management available, with data often being stored manually within local spreadsheets.

One way to provide German foundries with a cost effective way to incorporate data-driven monitoring and control into their processes would be the development of a simple methodology and its software implementation, which should contain all the machine learning functionality required for the task of continuous process monitoring and quality prediction in a foundry, presented in a user-friendly way which allows for easily understanding the implicit knowledge available in the data. In that case, extensive machine learning expertise is no longer required from the foundries, which could instead focus on the task of collecting sufficient amounts of data from the casting process.

1.2 Problem specification

This research work intends to support online monitoring and control of complex processes in the foundry industry using *Machine Learning* (ML) methods by providing a unified framework that can easily be employed even by SMEs without extensive expertise in the field of data science. Within this framework, data-driven predictive analytic methods are employed which summarize the patterns and trends of a foundry process in the form of a data-driven model in order to control the foundry process (or sub-processes) effectively and to produce high quality casting parts efficiently. The proposed approach involves analyzing data and extracting knowledge using different ML methods and predicting the expected outcome of a casting process as well as advanced guidance in case the prediction shows that the monitored qualitative property would leave an optimal range under current process parameters.

In particular, we consider the following research questions:

- Is there a single ML method which can be considered fully suitable for foundry industry processes, or is a unique learning method required for different sub-processes?
- Can a unified approach be proposed which can be employed in any foundry to monitor and control its work, without requiring extensive background knowledge about data processing or machine learning?

2 Theoretical background

In order to successfully apply machine learning techniques within the foundry industry, it is important to understand the particular challenges which are presented by industrial casting processes. Some of these challenges are due to the high complexity of the casting process itself, while others arise from the operational context of the foundry industry, such as the prevalence of SMEs among German foundry companies and the associated lack of extensive digital infrastructure.

Of course, both the possibility of employing data-driven methods in industrial processes and the necessity to do so in order to stay competitive are rather recent developments. This rapid technological progress is often called the *4th industrial revolution*.

2.1 Industry 4.0 and its key technologies

Figure 2.1 shows the major revolutionary developments of industrial processes throughout history. During the 1st revolution (1784-1870), water and steam powered engines were used for the production of mechanical systems. The 2nd revolution (1871-1969) brought mass production and assembly lines due to electricity, whereas the 3rd revolution (1970-2010) was the era of IT systems to further automate the production lines. In the current 4th revolution (2011-today), the so-called *Internet of Things* (IoT) and cloud technology have provided further means to automate even more complex tasks.

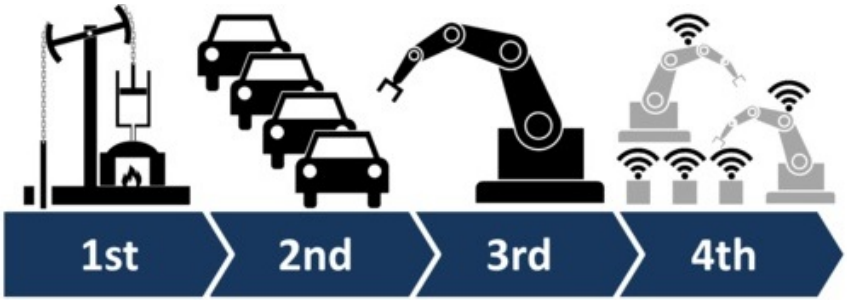


Figure 2.1: Industrial revolutions [108]

The term “Industry 4.0”, which is also used for the current state of industrial development, originated in Germany and originally described a strategic initiative presented by the German government at the 2011 Hannover Fair, which aims to transform the industry by digitizing and harnessing the potential offered by new technologies [36, 105]. With this strategy, Germany intends to maintain its leading position as one of the most competitive

countries in engineering, electronics and the automotive industry. It can be considered one of the reasons Germany was ranked third in 2018 in the global competitiveness rankings published by the World Economic Forum, which measures how a country uses its resources to provide its residents with a high standard of well-being [146]. "Industry 4.0" describes a future industrial scenario, characterized by the real-time availability of relevant information from the connection between people, objects and systems. The platform seeks solutions to the problems associated with the future consequences of industrial developments through standardization and the creation of a legal framework. Business, science and politics work together to make Industry 4.0 a reality by digitizing manufacturing processes and creating new products and services [143]. Industry 4.0, in addition to being a natural consequence of new technologies and digitization, provides a new way of increasing industrial profits: companies which successfully adopt this approach are expected to be able to cut their costs by up to 40% [37, 108].

Smart factory concepts have already been realized by major companies such as Deutsche Telekom [19]. As shown in figure 2.2, processes are automated, secured and highly optimized in smart factories, and predictive maintenance is being employed for their smooth operation. At any point in time, the current state of such a production process and its configuration can be checked and modified remotely. In order to achieve such a degree of automation, a number of recent technological advances need to be fully utilized.

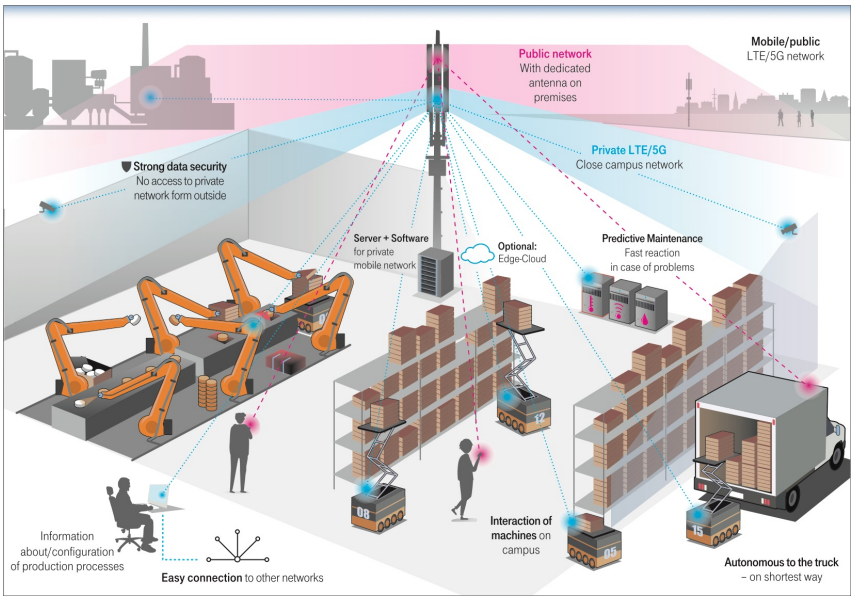


Figure 2.2: Smart factories concept from Deutsche Telekom using 5G [19]

Cloud computing is the provision of computing resources on demand over the Internet. It arises from the need for companies to quickly obtain flexible and powerful computing

services by paying only for what is used [77]. Cloud computing services can be divided into three categories: *Infrastructure-as-a-Service* (IaaS) provides access to storage, servers and networks for the users who use their own applications and platforms. *Platform as a Service* (PaaS), in addition, provides a cloud environment for developing and testing applications. Finally, *Software as a Service* (SaaS) consists of a cloud software subscription from a vendor that is available over the Internet [45]. Due to cloud computing, extensive IT infrastructure is no longer required even by larger companies; instead access to the network space is leased from a cloud provider. The companies providing these services take care of both data security and the maintenance of networks and equipment. Thereby, it is possible to centralize data in the cloud, which is easily accessible from various devices, and exchange information with third parties [103].

The term IoT describes a network of interconnected objects and devices that can receive and transmit data over the Internet [72]. It is an important driver for customer-centric innovation, data-driven automation and optimization, digital transformation and business models across all industries. Thus, the IoT is considered the basic technology to realize the ideas of Industry 4.0. The IoT is based on *Radio Frequency IDentification* (RFID) technology, which allows each product or device to be assigned a code that serves as a unique identifier [83]. Objects connected to the IoT typically carry sensors capable of detecting real-world conditions and actuators with which they can perform actions. In short, the IoT focuses information and decision making on each device, and then transmits that data to the network over the internet.

While these technologies have been adapted in some industrial areas already, their application in the foundry industry presents some additional challenges.

2.2 Manufacturing processes in the foundry industry

Metal casting is a primary forming process in which molten metal is poured into a mold with a cavity to produce an object of the desired shape. Once the metal is solidified, the mold is removed and the casting part is obtained. These castings can be processed to be used, for example, in automotive, spacecraft, industrial and domestic components.

In Germany, SMEs play a particularly important role in the foundry industry. SMEs, with employee numbers ranging from one to several hundred, are often considered to be the backbone of the German economy. As of 2019, Germany was the world fifth largest producer of metal castings (as shown in Figure 2.3), with an annual production of approximately 5 million tons.

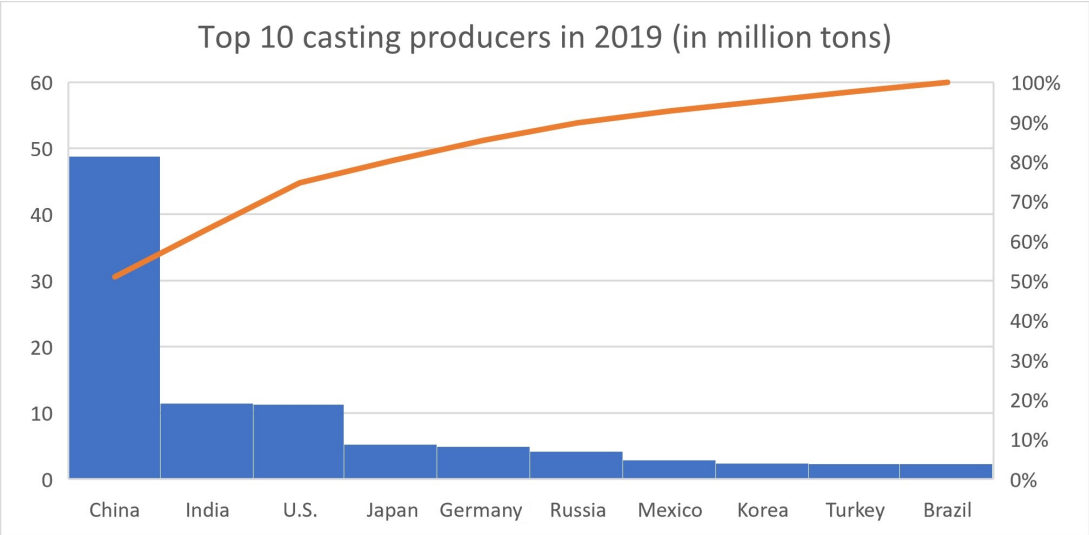


Figure 2.3: Top 10 casting producers in 2019 [130]

Figure 2.4 shows an abstract representation of a very basic foundry process which takes raw material as its input and passes it through multiple sub-processes, thereby transforming the raw input into a finished product (the casting). For the sake of simplicity, we assume that the whole casting process works in a sequence where sub-processes are interdependent, with each of the sub-processes' output becoming the input to the next sub-process. As these processes are inter-related, the final casting quality depends on the optimal operation of each individual sub-process. The operational conditions are influenced not only by controllable parameters, but also by environmental factors, such as the temperature and humidity, which can influence, for example, the chemical properties of the casting.

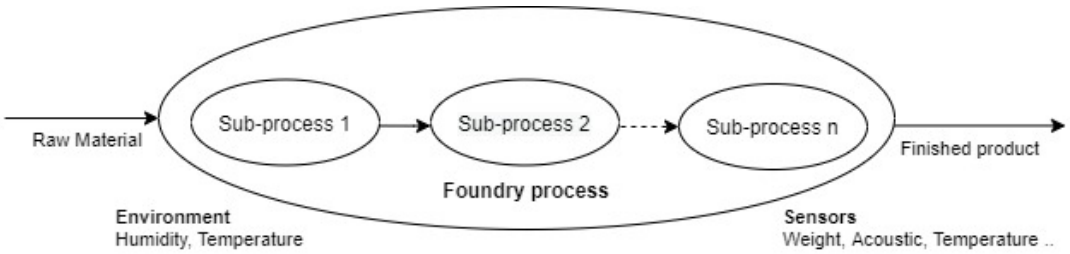


Figure 2.4: Abstract foundry process

In order to optimize the outcome of the casting process, each sub-process therefore needs to be operated in a highly controlled manner, which requires continuous monitoring of the whole process. Optimal process parameters need to be determined, and any deviation from the acceptable range of parameters must be recognized sufficiently early to allow for corrections. Due to the technological advancements of the 4th industrial revolution, it is now indeed possible to continuously monitor casting processes in terms of load, acoustics, temperature and other sensor data.

These sensors can not only be used to monitor and control the casting processes, but also to extract and store operating information in the form of a relation between inputs and outputs for each of the sub-processes as well as for the whole process.

Due to complex nature of the foundry process, such a recording of operations presents a number of challenges. The variety of possible environmental factors which can influence the process means that some relevant information might be missing from the recorded data might or, vice versa, that a large amount of unnecessary information is recorded. Similar information might also be recorded multiple times in the form of interrelated (e.g. correlated) data, leading to additional redundancies.

In order to fully utilize the possibilities offered by Industry 4.0 technologies, obtaining high-quality data representative of the actual process is an essential prerequisite. In particular, the use of *data-driven models* in the casting industry – which is the central subject of the present work – heavily relies on the availability of data which is sufficient not only in quantity, but also in quality.

2.2.1 Categorization of casting processes

Casting processes can be classified according to the type of the target product. If the target of the casting process is a semi-finished product for further processing, it is referred to as semi-finished casting. Typical semi-finished castings are strands, tubes, profiles and strips produced by the continuous casting process. Mold casting can be subdivided based on whether molds are destroyed when the castings are removed (*expendable mold casting*) or whether they can be used several times (*permanent mold casting*). Examples of expendable mold casting include sand casting, investment casting and shell molding [40]. In permanent mold casting, a reusable mold mostly made of metals is used. In these castings, mold is mostly filled by gravity, but gas or vacuum can also be used. Types of permanent mold casting include die-casting, low pressure casting and centrifugal casting. An overview of the different casting categories described in the following is shown in Figure 2.5.

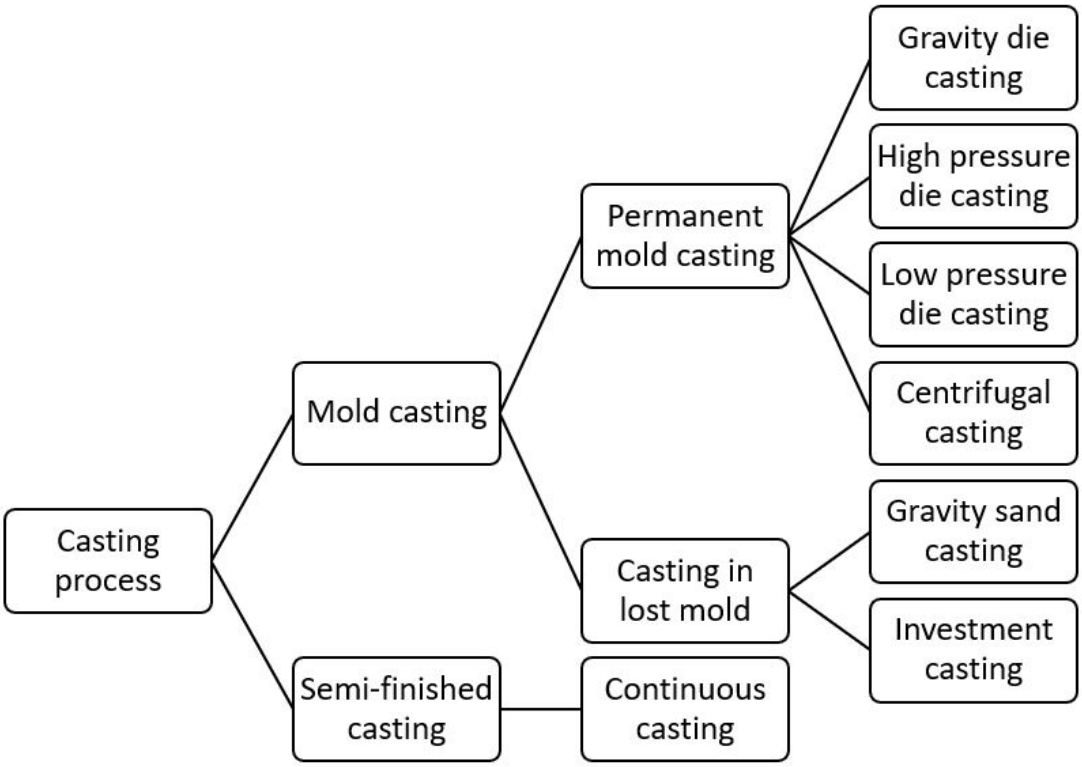


Figure 2.5: Casting process classification

Gravity casting

Gravity casting is a casting process in which the molten metal enters the mold under the effect of gravity. It is considered the standard casting process and is therefore not usually mentioned specifically. An exception is gravity die casting shown in Figure 2.6, which needs to be distinguished from other high and low pressure die casting processes [24].

In gravity casting, the molds have an opening at the top to allow the air in the mold to escape before casting. The melt can be poured directly into the mold from above or a separate gating system is used in which the melt first falls down the gate and then flows laterally into the cavity [141].

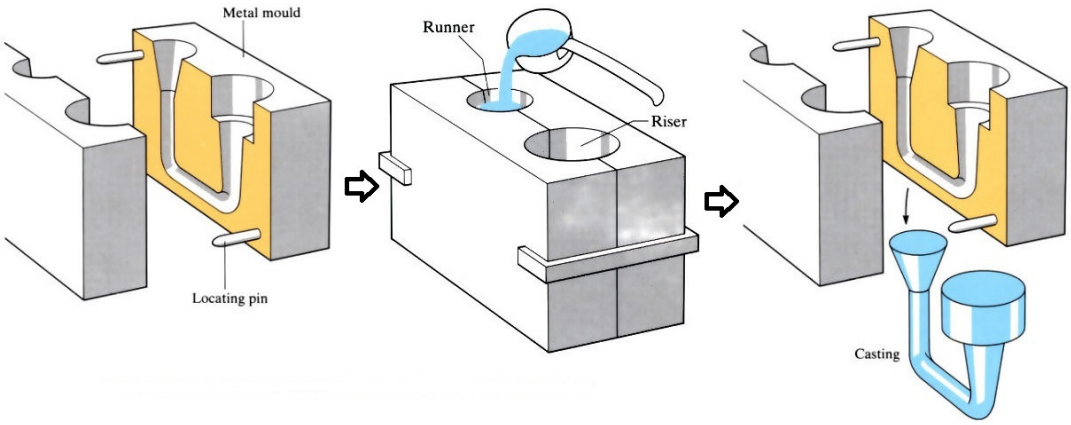


Figure 2.6: Gravity die casting (adapted from [89])

High-pressure die casting

Die casting is a casting process for series or mass production. Casting alloys with a low melting point are usually used for this purpose.

In high-pressure die casting, the liquid melt is forced into a die casting mold under high pressure of approx. 10 to 200 MPa and at a very high die filling speed of up to 20 m/s, where it then solidifies. The special feature of the die casting process is that it works with a permanent mold, which means that for a series of identical components, mold production is only required once, albeit at a much higher manufacturing cost [141].

The die casting process is further divided into hot-chamber and cold-chamber processes. In cold chamber machines, the melt is supplied from an external furnace for each casting cycle. This is suitable for Al, Zn and Cu alloys. In contrast, in hot-chamber machines, the melt feed is integrated into the machine. However, only alloys with lower melting temperature can be used in hot chamber machines, for example Mg alloys [141].

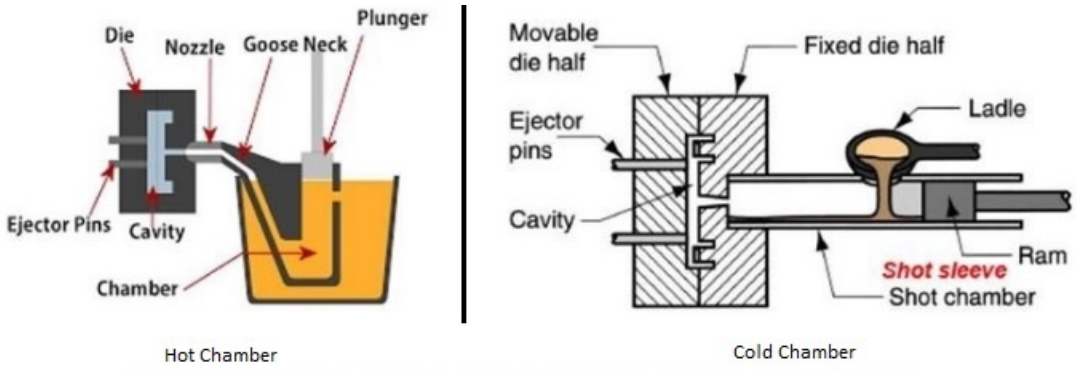


Figure 2.7: Hot/cold chamber die casting (adapted from [128])

The die casting process consists of three main phases. First, the liquid metals are advanced into the casting chamber at low velocity. This is the preliminary phase. The velocity in this phase is relatively low so that the air in the casting chamber can escape. In the subsequent filling phase, the speed of the melt is accelerated. The final phase is the post-compression: the liquid metals are pressed into the molds at a higher speed to compensate for the effect of liquid shrinkage in order to reduce the pores and air pockets.

Low-pressure die casting

Low-pressure casting refers to casting arrangements in which the molten metal (primarily aluminum, but also magnesium, copper, iron and steel) is forced from below into the mold cavity of the attached casting mold, usually a permanent mold, but also a sand mold or an investment casting mold (shell mold), usually by means of a riser tube. In this process, the upward movement of the liquid metal against gravity is effected preferably according to the gas pressure principle [141]. The structure of low-pressure casting is shown in Figure 2.8. The basic process sequence is described as follows:

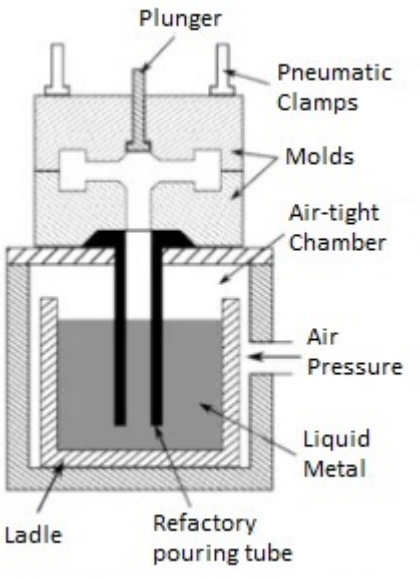


Figure 2.8: Low-pressure casting process [62]

Using gas pressure, the metal rises and enters the mold cavity. After mold filling, gas pressure is maintained during solidification to allow replenishment to compensate for the volume deficit (blowholes) during the transition from the liquid to the solid state.

Centrifugal casting

Centrifugal casting is a casting process for the production of rotationally symmetrical components (e.g. cast iron pipes for water and wastewater). For this purpose, as shown in Figure 2.9, liquid metal is poured into a casting mold rotating about its central axis. Friction-induced thrust forces cause the melt to also rotate, and centrifugal force presses it against the mold wall. The outer contour of the component is determined by the inner geometry of the mold. After the mold has been filled, it continues to rotate until the component has solidified completely [141].

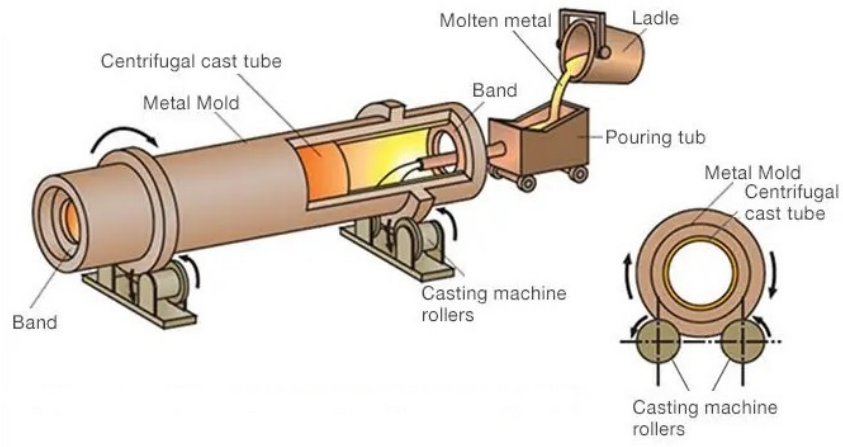


Figure 2.9: Centrifugal casting [126]

Sand casting

Sand molding or sand casting is a casting process for metal and other materials that uses molds made of sand. It works on the principle of lost molds, which means that the mold is destroyed after being used once in order to remove the casting [141].

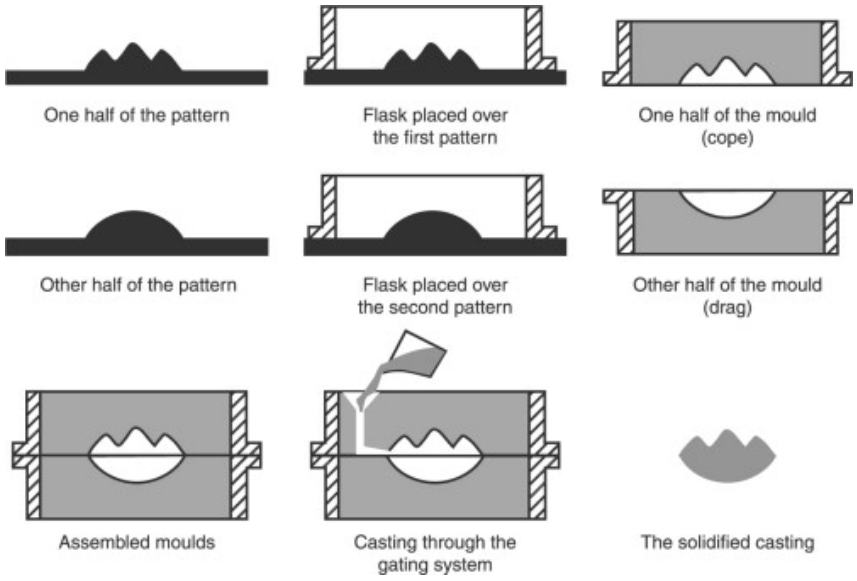


Figure 2.10: The process sequence of sand casting [90]

In sand casting, as shown in Figure 2.10, a model is used to create two separate mold halves from a mixture of sand and binder. After the sand is compacted to its required

strength, cores are placed on the lower half mold and the two halves are joined. The melt is then poured into the mold cavity. After solidification, the molding sand is emptied, the cores are removed and the casting is cleaned from any remaining sand. The gating and feeder system is then removed and later remelted.

An important step in the sand-casting process is the creation of the cores. Two different procedures can be used for this step: In the *cold box process*, the sand mixed with binder and hardener fills the mold cavity. The core is gassed, and the gassing cures the flowable core sand in the core box. In the *Hot-box process*, on the other hand, the core is cured (as the name suggests) in a hot core box equipped with a heater. No gassing of the core is necessary in this case [141].

Over time, there have been a number of advances of the sand casting process. In modern manufacturing, the sand molds and cores can be *Computer Numerical Control* (CNC)-milled directly from compacted sand blocks consolidated by means of resins. In some cases, sand molds are even produced directly using a 3D printing process.

Investment casting

Investment casting is a process suitable for manufacturing small parts. It is classified as a casting process from lost patterns and lost molds because both the pattern, which is usually made of wax or plastic, and the mold are no longer available after casting [141].

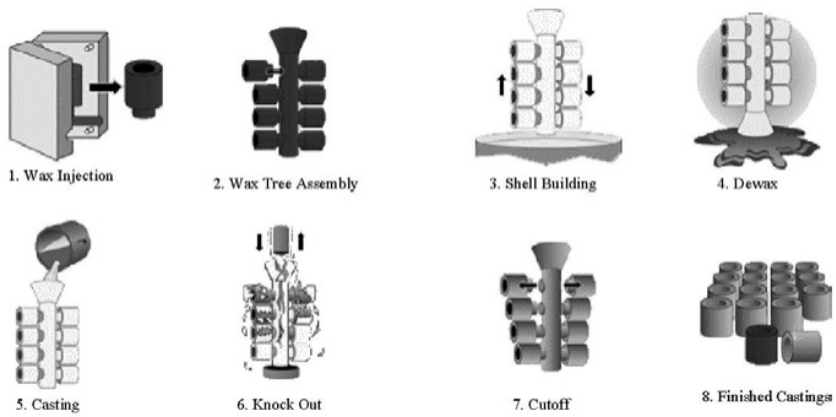


Figure 2.11: The process sequence of investment casting [96]

As shown in Figure 2.11, the investment casting process begins with the production of wax patterns, most commonly in metallic molds. The individual models are then attached to a casting system, which is the complete model of the later casting. A solid ceramic mold shell is then built around the model in several layers. Once the shell is completed,

the model wax is melted out and the ceramic mold is fired in a furnace. The metal melt is then poured into the preheated mold. Finally, after the melt solidifies, the mold is quickly destroyed to avoid shrinkage cracks.

2.2.2 Casting alloys

Metallic casting materials are traditionally divided into the two groups of ferrous and non-ferrous materials. Ferrous casting materials, i.e. iron-carbon alloys, are subdivided into cast steel and cast iron materials based on their carbon content. Cast steel materials are divided into two categories: low alloy steels which contain less than 8% alloying content and high alloy steels which contain 8% or more alloying content. These alloys are produced from different alloy grades like CF-8M, CA-15, HC, HD. For example ZG25MnNi and ZG30Mn2 are two low-alloy steels. [88]. Important categories of cast iron ($> 2\%$ carbon) include lamellar graphite cast iron (GJL), compacted graphite iron (CGI) and nodular graphite cast iron (GJS). Non-ferrous metal casting materials include copper, lead, tin, zinc and nickel as well as light metals such as aluminium, magnesium, titanium and their alloys. Finally, non-metallic materials such as plastics, clay, ceramics or plaster can also be produced by casting [141].

2.2.3 Casting defects

During a casting process, many different types of defects – including surface, internal or dimensional ones – can negatively influence the quality of the casting part [40]. One of the main goals of applying monitoring and control methods in the foundry industry is the prevention of such defects. However, the high complexity of the casting process is reflected by the wide variety of possible defects. Some of the most common defect types are listed in Table 2.1.

Table 2.1: Casting defects [12]

Defect Type	Description
Gas porosity	Dissolved gas in the liquid material forms Gas bubbles inside the final casting.
Shrinkage	Contraction in the volume of casting when molten metal solidifies.
Mold material	
Pouring metal	*Misruns* occur when the mold cavity is not filled completely. *Cold shut* happens when two parts of liquid donot fuse well and leave a weak spot. *Inclusion* is a metal contamination, known as *dross* if it occurs in solid form and *slag* if it occurs as a liquid.
Metallurgical	While cooling down, hot metal can be affected by residual stresses in the material, a defect known as called *hot cracking*. *Hot spots* occur when parts of a casting cool down faster than the rest of the melt.

In particular, defects induced by gas and shrinkage porosity formed during solidification, misruns, a cold shut, inclusion, hot cracking and hot spots can make the tensile behaviour of casting alloys unpredictable. Avoiding such defects is therefore vitally important in order to control the quality of casting products in the foundry industry.

2.2.4 Process parameters and control

Of course, preventing significant defects is not the only objective in the casting process: cost reduction as well as the precise and accurate production of parts play an equally important role, with environmental concerns leading to an ever increasing focus on energy efficiency as an additional objective. In order to achieve these compound objectives, different techniques of control are employed throughout the foundry industry, including the methods of *Statistical Process Control* (SPC), *Design of Experiments* (DoE) and *acceptance sampling* [81].

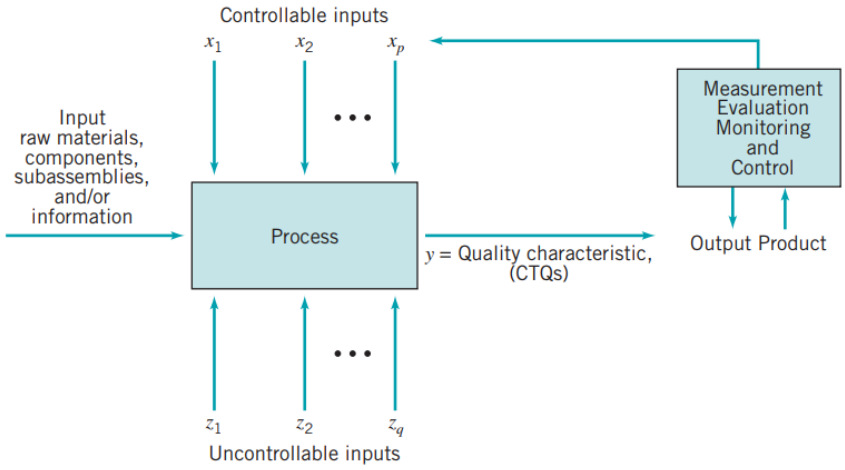


Figure 2.12: Inputs and outputs of the manufacturing processes [81]

Figure 2.12 shows a typical control chart used in SPC, with a distinction between *controllable* and *uncontrollable* inputs which can affect the quality characteristics of the produced parts. These characteristics are monitored and controlled with a control chart (cf. Figure 2.13) which shows the average quality characteristics of the produced parts in relation to the Upper Control Limit (UCL) and the Lower Control Limit (LCL). These quantities can be particularly useful for diagnostic purposes in process monitoring.

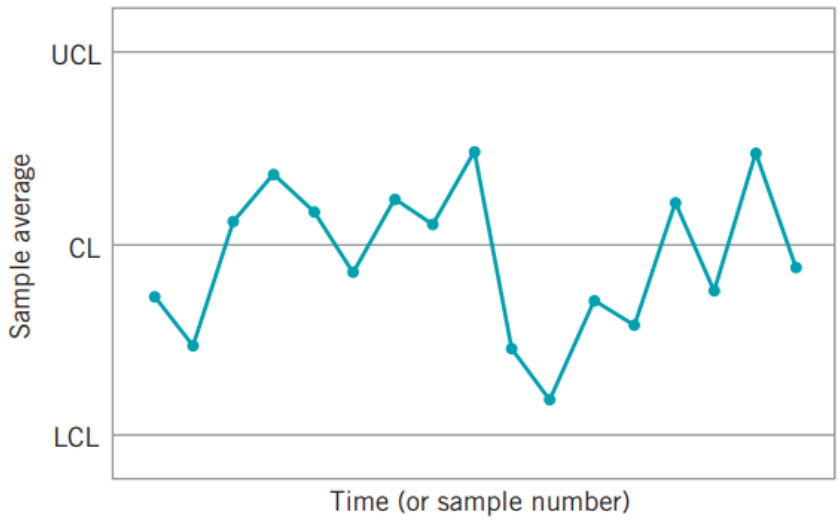


Figure 2.13: A typical control chart [81]

DoE methods [145] are helpful in identifying the key controllable inputs influencing the

quality characteristics of the produced parts. To this end, inputs are systematically changed and the effect of the changes on the quality of the produced part is investigated. Thereby, it is possible to find the relationship of these inputs with the quality characteristics. Once the key input variables are identified, online SPC methods can be used to monitor and control the production processes.

Other techniques for identifying and analysing parameters influencing a process were proposed by Ishikawa, including fishbone diagrams, check sheets, histogram charts, Pareto charts, scatter diagrams, control charts and stratification diagrams [82]. The fishbone diagram, also known as *Ishikawa diagram* or cause-effect diagram, is still widely in use today.

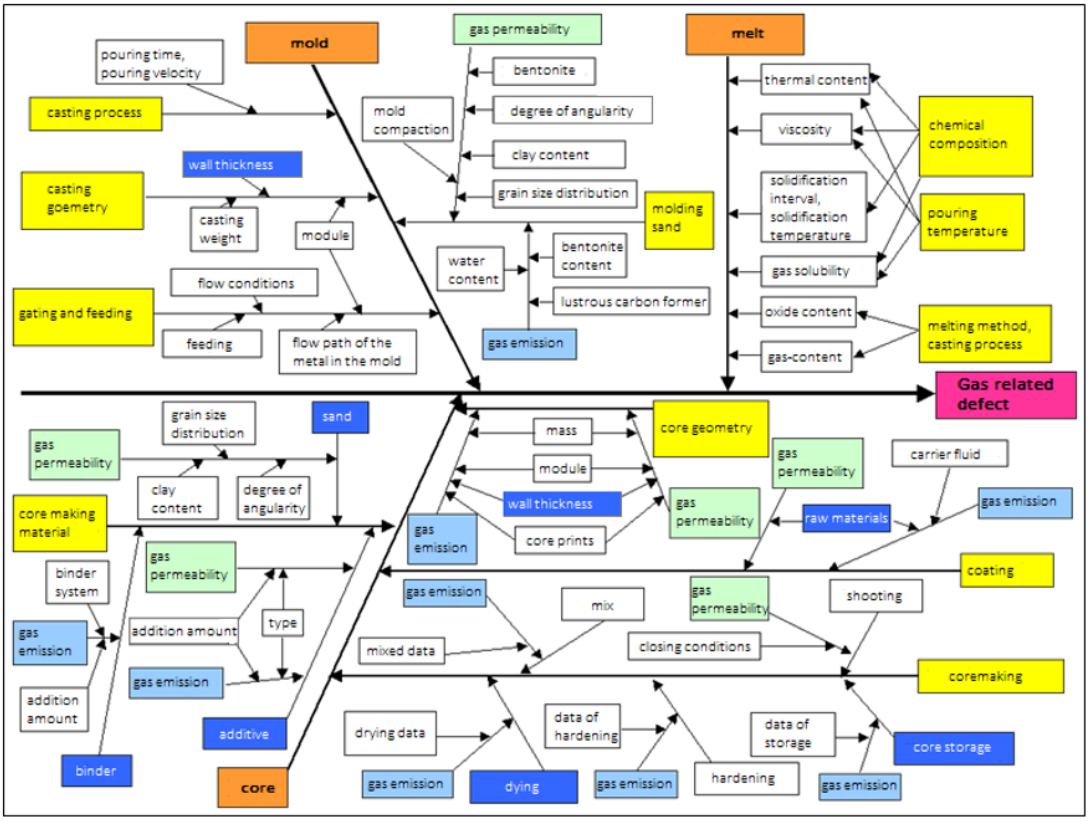


Figure 2.14: Ishikawa diagram: Complex influences in casting of copper-based alloys [111]

Figure 2.14 shows an Ishikawa diagram for the casting of copper-based alloys, which demonstrates the complexity of the various influences of casting sub-processes on gas porosity in cast copper-alloy components. For example, the casting quality depends on the qualities of the core, the mold and the melt; the core itself is influenced by its geometry, the molding material, the coating and the core-making process. Each of these processes are,

in turn, dependent on multiple other parameters. Note that the diagram only shows the influencing factors for gas-related defects – similar interdependencies are to be expected for any other objective in the casting process as well.

Another class of techniques for controlling production processes and finding the relationship measured variables is offered by the methods of *machine learning*. Such methods have been employed for the monitoring and control of industrial processes for more than thirty years. For example, knowledge-based systems have been used in the foundry industry to understand and learn from measured data and to optimise production capacity, for diagnosis and evaluation of the casting defects and for mold preparation and planning. An early knowledge-based expert system called DEFCHAR [127] was developed by Sudesh et al., while Sreenivas et al. proposed a knowledge based expert system called MODCAS [124] for post-casting defect analysis. Kim et al. successfully applied the Artificial Neural Networks in metal forming processes using the back propagation algorithm to find the initial billet size for axisymmetric rib-web products and to design the die geometry for cylindrical pulleys [61]. A more extensive overview of prior applications of machine learning to metal casting processes will be given in Section 3.

2.3 Machine learning and performance evaluation measures

A general definition of machine learning as a "Field of study that endows computers with the ability to learn without being explicitly programmed" can be traced back to Arthur Samuel in 1959. [112] Tom Mitchell, in 1997, defined machine learning as follows: "A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." [80]

2.3.1 Classification of machine learning problems

Machine Learning problems can broadly be classified into a range between *supervised*, *unsupervised* and *reinforcement learning*. In *supervised learning*, the measured data is composed of inputs known as *features* and outputs known as *labels*. The task is to find a suitable model that maps the feature vector to the corresponding label. *Classification* and *regression* are the subtypes of *supervised learning*. In *classification learning*, the dependent variable can take finitely many different values (called *classes*), whereas in *regression learning*, the dependent variable is a real number. In *unsupervised learning*, the measured data is not endowed with labels. This type of learning includes *clustering*, where the task is to group similar data in the form of clusters. Finally, in *reinforcement*

learning, an autonomous agent learns to act optimally by sensing its environment and acting in it to achieve its goals. To this end, an efficient policy needs to be set up which prescribes a sequence of actions.

In the last years, a great deal of research has been focused on developing tools which can learn a system model without having an explicit mathematical function to describe it. The implicit model identification techniques build a model for dynamic systems using only the measured data by adjusting the parameters until the system output label coincides as closely as possible with the measured one. Machine learning has the same goal, using different algorithms to learn from data and find hidden insights. When machine learning tools are exposed to new data, they are able to adapt independently. They learn from previous computations to produce reliable, repeatable decisions and results.

2.3.2 Formal description of supervised learning

For some sample space Ω, let $X : \Omega \to \mathbb{R}$, $Y : \Omega \to \mathbb{R}$ be measurable functions representing random variables, where $X(\omega) = x \in \mathbb{R}$ and $Y(\omega) = y \in \mathbb{R}$ for $\omega \in \Omega$. In supervised learning, we consider a vector $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_N)$ of N random *input* variables, also known as *features*, as well as a vector $\boldsymbol{Y} = (\boldsymbol{Y}_1, \boldsymbol{Y}_2, ..., \boldsymbol{Y}_K)$ of K random *ouput* variables known as *labels*. Note that $\boldsymbol{X} : \Omega \to \mathbb{R}^N$ and $\boldsymbol{Y} : \Omega \to \mathbb{R}^K$ are random vectors.

For a single *outcome* $\omega_i \in \Omega$ we obtain the *observation*

$$o^i = (\boldsymbol{X}(\omega_i), \boldsymbol{Y}(\omega_i)) = (x_1^i, x_2^i, \ldots, x_N^i, y_1^i, y_2^i, \ldots, y_K^i), \tag{2.1}$$

which is the i-th row of the *matrix of observations* O. Assuming there are M observations, then we get a $(M \times (N + K))$-matrix

$$O = (o_j^i)_{1 \le i \le M, 1 \le j \le N+K} = (\boldsymbol{X^i}(\omega_j), \boldsymbol{Y^i}(\omega_j)) = (\boldsymbol{x}^i, \boldsymbol{y}^i)_{1 \le i \le M}\,. \tag{2.2}$$

Each row of this matrix is a realization of the random vector $(\boldsymbol{X}, \boldsymbol{Y})$, whereas each column of O consists of M realizations of a single random variable $\boldsymbol{X}_{k,1 \le k \le N}$ or $\boldsymbol{Y}_{j,1 \le j \le K}$. For sufficiently large $M \in \mathbb{N}$, the observation $(\boldsymbol{x}^i, \boldsymbol{y}^i)_{1 \le i \le M}$ is assumed to be *representative* of the given problem; one expects that when observations are generated again (e.g. if the experiment is performed under similar circumstances), the obtained values will follow a similar distribution.[1] We will assume that the observed data are *stochastically independent* of each other, i.e. that the values of the random quantities $(\boldsymbol{X}^i, \boldsymbol{Y}^i)$ do not influence each other during the observation. For example, if the $\boldsymbol{X}^i$ correspond to several measured

[1] Note that this assumption is violated if one has made an arbitrary selection of training data based on certain characteristics or if there has been any systematic influence on the measured values during the measuring process.

values over time (at time $i = 1, ..., M$), it must be ensured that values obtained closely adjacent in time do not coincide more strongly than those that are far apart in time. Overall, these assumptions mean that the values of $\boldsymbol{X}^i$ are *Independently and Identically Distributed* (IID).

An example of an observation matrix O is shown in table 2.2. Here, the feature variable $\boldsymbol{X}_1(w_i) = (x_1^i)_{1 \leq i \leq M}$ represents carbon (C) with values in the range $3.44 \leq \boldsymbol{X}_1 \leq 3.64$. The values of a feature lying outside the acceptable range are known as *outliers*.

In supervised machine learning, for a selected label variable $\boldsymbol{Y_j}$, $j \in (N+1, N+2, ..., N+K)$, the target is to find a deterministic prediction function $f_i^* : D \subset \mathbb{R}^N \to \mathbb{R}$ with

$$y_j = f_j^*(x_1, x_2, ..., x_N)\,,$$

where $x_1, x_2, ..., x_N$ feature variable values, y_j is the corresponding j-th label variable value and D is a subset of all the possible values for the features.

To find such a prediction function f^*, observations $(\boldsymbol{x}^i, \boldsymbol{y}^i)_{1 \leq i \leq M}$ are required which are realisations of the random variables $\boldsymbol{X}, \boldsymbol{Y}$.

An exact computation of the prediction function f^* based on the observation matrix O is in general not possible. Instead, we want to find a good estimation $\hat{f}_{O,j} : D \subset \mathbb{R}^N \to \mathbb{R}$ of f^*. Such a function $\hat{f}_{O,j}$, which can be called a *data-driven model*, depends on the observation data O. There are two major quality criteria for the evaluation of $\hat{f}_{O,j}$:

- Interpretability (qualitative): Can the results from $\hat{f}_{O,j}$ be used to gain a deeper understanding of the relationship between $\boldsymbol{X}$ and $\boldsymbol{Y_j}$?
- Predictability (quantitative): Does $\hat{f}_{O,j}(\boldsymbol{X})$ predict the correct $\boldsymbol{Y_j}$ for new observations $\boldsymbol{X}$, or how good is this prediction? The quality of the prediction $\hat{f}_{O,j} : \boldsymbol{X} \to \boldsymbol{Y_j}$ is obtained by means of a *loss function* L, which measures the distance between $\hat{f}_{O,j}(\boldsymbol{X})$ and $\boldsymbol{Y_j}$. An example of a loss function for regression learning is the *Root Mean Square Error* (RMSE)

$$L_{RMSE,j} \;=\; \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(|y_j^i - \hat{f}_{O,j}(\boldsymbol{x^i})|^2\right)}\,, \tag{2.3}$$

  where $j \in \{N+1, N+2, ..., N+K\}$.

To get an overview of what such a matrix of observations looks like, let us consider the dataset shown in Table 2.2. This sample dataset shows the chemical composition of an alloy as the features and the resulting Tensile Strength (TS) as the label. Since it is, in practice, too cumbersome or not possible to measure all the possible combination of values for all the random variables, we assume that the selected sample follows the same unknown hidden distribution as the actual population.

Table 2.2: Sample of measured chemical properties of a metal

	Features (All in %)						**Labels**		
Count	1	2	3	4	5	6	1	2	3
	C	**Si**	**Mn**	**Ni**	**Cu**	**Al**	**TS (MPa)**	**Is OK**	**TS Class**
1	3.64	2.15	0.097	0.011	0.051	0.022	400.4	0	2
2	3.6	2.53	0.205	0.007	0.35	0.018	576	0	4
3	3.46	2.38	0.249	0.014	0.766	0.016	739.1	1	3
					⋮				
M	3.54	2.41	0.283	0.297	0.821	0.017	851	1	1

As shown in Table 2.2, a number M of total observations is measured. There are $N = 6$ chemical properties of the metal – the features – and $K = 3$ TS measurements, i.e. the labels. All features are real-valued, while the type of the three dependent variables are real, binary and multiclass, respectively. We split the dataset into $(x^i, y_1^i)_{1\leq i\leq M}$, $(x^i, y_2^i)_{1\leq i\leq M}$ and $(x^i, y_3^i)_{1\leq i\leq M}$ as shown in Tables 2.3, 2.4 and 2.5 respectively.

Table 2.3: Regression dataset

	Features (All in %)						**Labels**
Count	1	2	3	4	5	6	1
	C	**Si**	**Mn**	**Ni**	**Cu**	**Al**	**TS (MPa)**
1	3.64	2.15	0.097	0.011	0.051	0.022	400.4
2	3.6	2.53	0.205	0.007	0.35	0.018	576
3	3.46	2.38	0.249	0.014	0.766	0.016	739.1
				⋮			
M	3.54	2.41	0.283	0.297	0.821	0.017	851

Table 2.3 shows a *regression* problem, since the dependent variable (**TS**) is real-valued. In Table 2.4, the dependent variable (**IsOK**) is binary, i.e. can only take two values; this type of machine learning problem is called *(classical) classification*. Finally, in Table 2.5, the dependent variable (**TS Class**) takes (> 2) finitely many integer values. This type of machine learning problem is known as *multi-classification*.

Table 2.4: Classification dataset

	Features (All in %)						**Labels**
Count	1	2	3	4	5	6	2
	C	**Si**	**Mn**	**Ni**	**Cu**	**Al**	**Is OK**
1	3.64	2.15	0.097	0.011	0.051	0.022	0
2	3.6	2.53	0.205	0.007	0.35	0.018	0
3	3.46	2.38	0.249	0.014	0.766	0.016	1
				⋮			
M	3.54	2.41	0.283	0.297	0.821	0.017	1

Table 2.5: Multiclassification dataset

	Features (All in %)						**Labels**
Count	1	2	3	4	5	6	3
	C	**Si**	**Mn**	**Ni**	**Cu**	**Al**	**TS Class**
1	3.64	2.15	0.097	0.011	0.051	0.022	2
2	3.6	2.53	0.205	0.007	0.35	0.018	4
3	3.46	2.38	0.249	0.014	0.766	0.016	3
				⋮			
M	3.54	2.41	0.283	0.297	0.821	0.017	1

There are a lot of parametric and non-parametric machine learning algorithms available, all of which have their respective weaknesses and strengths. To determine a suitable model function f^* using ML methods, three selections must be made:

1. The ML algorithm $\mathcal{A}$.
2. The loss function L.
3. Given *hyperparameters* $(\boldsymbol{\lambda}_{i,1\leq i\leq P})$ of the algorithm $\mathcal{A}$, an optimization procedure that finds $\boldsymbol{\lambda^*}$ with minimal approximation error.

To this end, a *model assumption* is made for the relationship f^* between $\boldsymbol{X}$ and $\boldsymbol{Y}$ [102]. This is done, for example, by making assumptions about the functional Structure of f^*. Such an assumption reduces the number of possible expressions of f^* and thus enables a systematic identification of $\hat{f}_O$ by maximizing the *predictability* on the training data. If a model assumption does not correspond to reality, the determined $\hat{f}_O$ can deliver arbitrarily bad results; conscientious selection is therefore of great importance. Over the

last decades there has been a paradigm shift in the development of algorithms $\hat{f}_O$: In the past, even before the algorithm was determined, experts spent a considerable amount of time to extract the essential components of $\boldsymbol{X}$ for the prediction of $\boldsymbol{Y}$ (i.e. the so-called feature extraction was performed with the help of expert knowledge). For the model assumption and the calculation of the algorithm based on it, it could then be assumed that all components of $\boldsymbol{X}^i_{1\leq i\leq M}$ are highly important for the prediction of $Y^i_{1\leq i\leq M}$. The dimension of the vector $\boldsymbol{X} \in \mathbb{R}^N$ was only moderately large (often $N < 20$). The number of training data M was generally much larger than N .

Nowadays, however, supervised learning problems are often highly dimensional, i.e. the given $\boldsymbol{X}$ are vectors of high dimension. The preprocessing of $\boldsymbol{X}$ by means of expert knowledge is either no longer available or results in an insufficient reduction of the components of $\boldsymbol{X}$. Accordingly, not every component of $\boldsymbol{X}$ is important for the prediction of $\boldsymbol{Y}_i$. A machine learning algorithm must therefore recognize the components of $\boldsymbol{X}$ that are important for $\boldsymbol{Y}_i$ and then, based on this, make predictions for $\boldsymbol{Y}_i$.

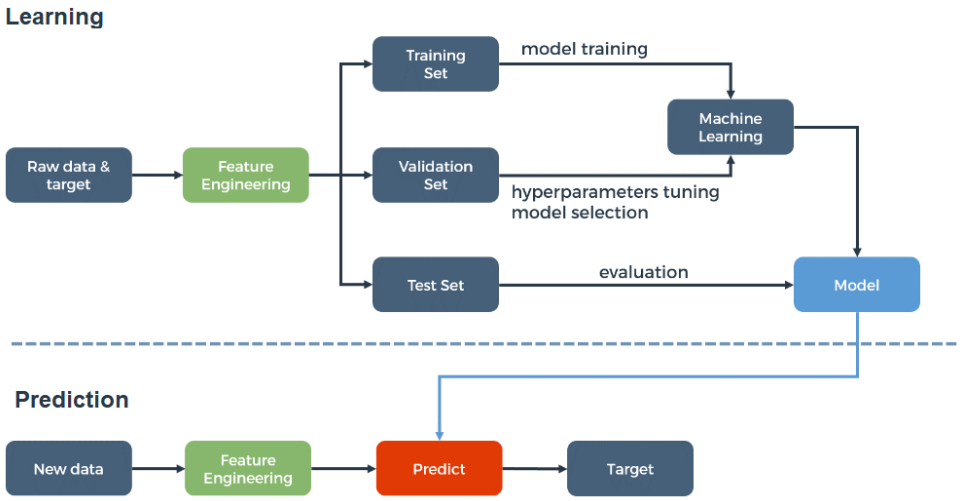


Figure 2.15: Machine learning process [54]

As shown in Figure 2.15, the complete process of ML can be divided into two steps, *Learning* and *Prediction*. In the learning step, the observation matrix $(\boldsymbol{x}^i, y^i)_{1\leq i\leq M}$ of features $(\boldsymbol{x}^i)_{1\leq i\leq M}$ and labels $(y^i)_{1\leq i\leq M}$ is passed to a *Feature Engineering* (FE) module. In the FE module, the data is processed and transformed; missing values, outliers, categorical data encoding and dimensionality reduction methods are applied to transform O. Then, in the simple validation case, the data is split into 3 sets: *Training* $(\boldsymbol{x}^i, y^i)_{1\leq i\leq n}$, *Validation* $(\bar{\boldsymbol{x}}^i, \bar{y}^i)_{1\leq i\leq m_1}$ and *Test* $(\tilde{\boldsymbol{x}}^i, \tilde{y}^i)_{1\leq i\leq m_2}$, for example in the ratio $n : m_1 : m_2 = 70 : 20 : 10$. For the selected machine learning algorithms, their respective model is trained on the training set and the hyperparameters of the model are optimised using the validation set.

Later, the unbiased evaluation of the trained models is performed on the test set. This procedure can be summarized more formally as follows.

1. Select a machine learning algorithm $\mathcal{A}$, limits $l_{\text{tr}} \geq 0$ and $l_{\text{val}} \geq 0$ for the training and validation error, and algorithm specific hyper parameter values $\lambda_k \in \mathbb{R}$.
2. Determine $\hat{f}_{n,\boldsymbol{\lambda}}$ for training data $(\boldsymbol{x}^i, y^i)_{1 \leq i \leq n}$ for different combinations of λ_k.
3. Determine $\hat{f}_n := \hat{f}_{n,\hat{\boldsymbol{\lambda}}^{\text{std}}}$ using the validation data, where
$$\hat{\boldsymbol{\lambda}}^{\text{std}} \in \arg\min_{\boldsymbol{\lambda}} L(\bar{y}^i, \hat{f}_{n,\boldsymbol{\lambda}}(\bar{\boldsymbol{x}}^i)) \text{ and } L(\bar{y}^i, \hat{f}_{n,\boldsymbol{\lambda}}(\bar{\boldsymbol{x}}^i)) < l_{\text{val}}\,. \tag{2.4}$$
4. An estimate of the empirical loss $L(\hat{f}_n)$ is calculated using the test error:
$$L(\hat{f}_n) = L(\tilde{y}^i, \hat{f}_n(\tilde{\boldsymbol{x}}^i))\,. \tag{2.5}$$

If the learning step is finished with good evaluation results, then the selected models are used in the prediction step to predict the output for new input data.

Let us focus on simple validation for determining the hyperparameter values $\boldsymbol{\lambda}$ for any selected ML algorithm in order to select a model $\hat{f}_{n,\boldsymbol{\lambda}}$. In simple validation, training data of size n is used for the calculation of $\hat{f}_n = \hat{f}_{n,\hat{\boldsymbol{\lambda}}}$ and validation data of size n_1 is used for the selection of the parameter vector $\boldsymbol{\lambda}$. To find the algorithm $\hat{f}_n$ with low generalization error, the number of training data n is crucial. On the other hand, a good validation data size n_1 to find and select a good $\boldsymbol{\lambda}$ is very important as well. When the size of the observed data is small, then dividing it further for training and validation can impede each of these steps. There are other methods, including k-fold cross validation, leaveone-out cross-validation and Akaike Information Criteria, which make use of complete training and validation data to determine $\hat{f}_n$. Kohavi et al. [63] used cross-validation for model selection and showed that 10-fold cross-validation can be a good choice in comparison to more expensive leave-one-out cross-validation.

Since noise cannot be excluded from the measured data, following the principle of Occam's Razor, simpler models which approximate the data well should be favoured over more complex models which fit the data perfectly. The target is to match the complexity of the learnt model with the complexity of the true function from which the data was generated. If the complexity of the learnt model is lower than that of the true function, underfitting will occur; if the complexity of the learnt model is higher, the result will be an overfitting to the data. This tradeoff is related to the so-called *bias-variance dilemma*: In parameter

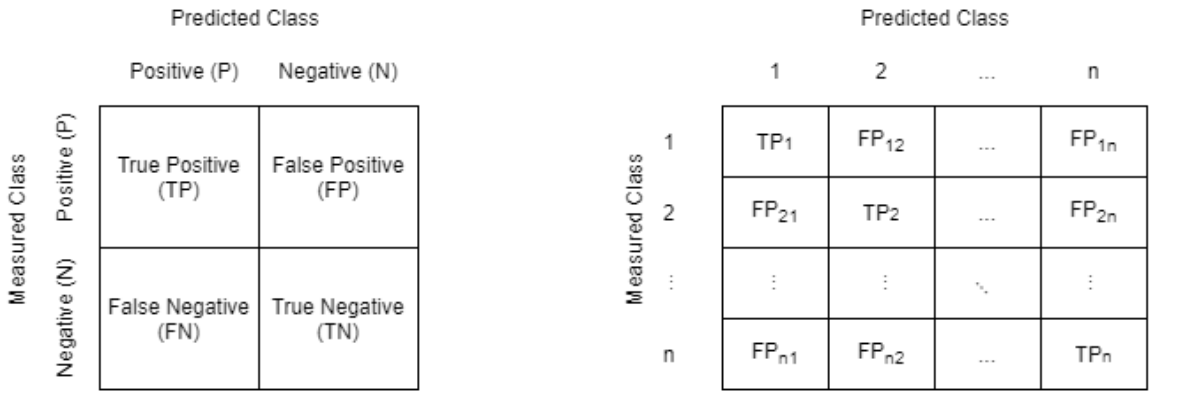
estimation for the models, lower bias results in higher variance and vice versa, which makes it challenging to minimize both.

Note that the process described above and in Figure 2.15 shows the basic functionality of learning using a *single* machine learning algorithm. However, there are many learning algorithms available, each with their own "areas of expertise", which can analyse and learn from data in their own specialized way. Most importantly, there is no single machine learning algorithm which is *best* for modeling *all types* of complex real world problems. In Section 4, we will approach this issue by utilizing multiple learning algorithms, selecting the well-performing ones and combining their outputs into a single result.

2.3.3 Performance evaluation measures

Performance evaluation is an important step for the selection of an optimal algorithm. Depending on the type of the learning problem, different evaluation measures are used.

Performance evaluation measures for classification problems
For a classification learning problem, a *confusion matrix* stores the count of correct and false classifications for each class. In case of a binary classification learning problem with target classses, *Positive*(P) and *Negative*(N), the confusion matrix is shown in Fig. 2.16 (a). The correct classifications count for both classes are on main diagonal, and summing them up results in the total count *True Positive* (TP) + *True Negative* (TN) of correct classifications. The counts in the off diagonal places show the number of incorrect classifications for each class. The sum of these numbers represents the total count of incorrect classifications, *False Positive* (FP) + *False Negative* (FN). The confusion matrix of an n-class learning problem is shown in Fig. 2.16 (b). In this matrix, main diagonal again contains the correct classification count for each class (TP_1, TP_2, ..., TP_n), and all the off-diagonal entries represent incorrect classification counts; for example, the entry at position $(1, 2)$ is FP_{12}, which is the count of misclassifications of class 1 as class 2.



(a) Confusion matrix for binary classification problems.

(b) Confusion matrix for multi-classification problems.

Figure 2.16: Confusion matrices for binary and n-class problems

For binary classification problems, based on the confusion matrix from Fig. 2.16 (a), different performance measures can be calculated, as described in Table 2.6.

Table 2.6: Binary classification measures

Measure	**Formula**	**Evaluation Focus**
Accuracy	$\frac{tp+tn}{tp+fn+fp+tn}$	Fraction of correct predictions by the classifier.
Precision	$\frac{tp}{tp+fp}$	Proportion of positive predictions being correct.
Recall	$\frac{tp}{tp+fn}$	Proportion of measured positives predicted correctly.
F1-Score	$\frac{(\beta^2+1)tp}{(\beta^2+1)tp+\beta^2 fn+fp}$	Consider precision and recall together.
Area Under Curve (AUC)	$\frac{1}{2}(\frac{tp}{tp+fp}+\frac{tn}{fp+tn})$	Measure how good classifier can avoid incorrect classification.

Accuracy measures the overall effectiveness of a classifier and is calculated by dividing the total number of correct predictions by the total number of measurements. ***Precision*** is computed by dividing the count of correctly predicted positive measurements by the total count of the positive predictions. ***Recall*** measures the effectiveness of a classifier to predict positive class labels and is calculated by dividing the count of correctly predicted positive measurements by the number of all positive measurements. The ***F1-Score*** measures the relation between positive class measurements and the predictions by

the classifier. It is calculated by computing the weighted harmonic mean of *Precision* and *Recall*, with the coefficient β serving as a weighting factor. Finally, the ***Area Under the Curve*** measures the area under the *Receiver Operating Characteristic* (ROC) curve from $(0, 0)$ to $(1, 1)$ and is used to check the ability of a Classifier to avoid wrong classification.

For multi-class classification problems, based on the confusion matrix from Fig. 2.16 (b), different performance measures can be calculated, as described in Table 2.7.

Table 2.7: Multi-class classification measures

Measure	**Formula**	**Evaluation Focus**
Average Accuracy	$\frac{\sum_{i=1}^{n} \frac{tp_i+tn_i}{tp_i+fn_i+fp_i+tn_i}}{n}$	The average per-class effectiveness of a classifier
Precision_M	$\frac{\sum_{i=1}^{n} \frac{tp_i}{tp_i+fp_i}}{n}$	An average per-class agreement of the data class labels with those of a classifiers
Recall_M	$\frac{\sum_{i=1}^{n} \frac{tp_i}{tp_i+fn_i}}{n}$	An average per-class effectiveness of a classifier to identify class labels
F1-score_M	$\frac{(\beta^2+1)\text{Precision}_M\text{Recall}_M}{\beta^2\text{Precision}_M + \text{Recall}_M}$	Relations between data's positive labels and those given by a classifier based on a per-class average

Matthews correlation coefficient (MCC) [157] and *Cohens Kappa statistic* (Kappa) [140] can be used as unified performance quality measures of binary and multi-class problems.

Performance evaluation measures for regression problems

The performance of the regression learning problems can be measured using the RMSE, the *Relative Root Mean Square Error* (RRMSE) or the *Symmetric Mean Absolute Percentage Error* (SMAPE), among others. The RRMSE and SMAPE are defined, respectively, via

$$RRMSE = \sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}\frac{(|O_i - P_i|^2)}{|O_i|^2}\right)}, \tag{2.6}$$

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|O_i - P_i|}{(|O_i| + |P_i|)/2}, \tag{2.7}$$

where n is the total number of observations, O_i is the given ("true") output and P_i is the predicted output from the machine learning algorithm.

Comparison between individual learning methods
Dietterich [27] compared learning algorithms using five approximate statistical tests to determine if one algorithm performs better than the other: a test for the difference of two proportions; two tests based on the paired-differences t-test using either several random train-test splits or 10-fold cross-validation; a method based on McNemar's test and, finally, a test based on 5 iterations of 2-fold cross validation. Other statistical algorithm comparison methods include an *Analysis of Variance* (ANOVA) and the *Kruskal-Wallis test* (H test). If the performance of each algorithm follows a normal distribution, then ANOVA is the preferred choice; if no particular distribution is assumed, then the H test is suitable.

To compare the performance of different ML methods on a single dataset, one-way ANOVA method can be used to compare the SMAPE values of different learning methods (single factor). For a given matrix $\boldsymbol{X}$, where each column represents a learning method, we can test the hypothesis that the values in the columns of $\boldsymbol{X}$ are drawn from populations with the same mean against the alternative hypothesis that the population means are not the same. Since we have considered two factors (learning methods, datasets) in our work, we use two-way ANOVA method in Section 5. It is used to determine effect of the two factors in order to test the hypotheses that all the learning methods error have same mean, all datasets used in this work result in same mean error and that learning methods and dataset errors do not interact with each other. If the results from two-way ANOVA method shows that the performance of algorithms is not same, then *post hoc tests* can be used to perform one-to-one comparisons between groups; Tukey's honesty difference criterion, the Bonferroni method, Scheffe's method and other pairwise comparison methods can be used to determine which methods are not significantly similar to each other.

2.4 Selected machine learning methods

In the following, some of the most common machine learning algorithms and methods related to pre- and post-processing are briefly introduced.

2.4.1 Kernel Principal Component Analysis

The *Kernel Principal Component Analysis* (KPCA) is a nonlinear dimension reduction method which maps data from the input space to a lower dimensional feature space while retaining the maximum possible variance in the data [117, 150].

Consider a function $\phi\colon \mathbb{R}^N \to F, \boldsymbol{x} \mapsto \boldsymbol{X}$ which maps the input data $\boldsymbol{x}_k$, $k = 1, \ldots, l$, $\boldsymbol{x}_k \in \mathbb{R}^N$ non-linearly into a feature space F such that $\sum_{k=1}^{l} \phi(\boldsymbol{x}_k) = 0$. The covariance matrix $\bar{C}$ is given by

$$\bar{C} = \frac{1}{l} \sum_{j=1}^{l} \phi(\boldsymbol{x}_j)\, \phi(\boldsymbol{x}_j)^T \,. \tag{2.8}$$

To perform a principal component analysis on $\bar{C}$, the eigenvalues $\lambda \geq 0$ and eigenvectors $\boldsymbol{v} \in F \setminus \{0\}$ which satisfy $\lambda \boldsymbol{v} = \bar{C}\boldsymbol{v}$ need to be determined. Since every eigenvector $\boldsymbol{v}$ lie in the span of $\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_l)$, we can consider the equation

$$\lambda(\phi(\boldsymbol{x}_k) \cdot \boldsymbol{v}) = (\phi(\boldsymbol{x}_k) \cdot \bar{C}\boldsymbol{v}) \quad \text{for all } k = 1, \ldots, l \tag{2.9}$$

and assume there exists coefficients $\alpha_1, \ldots, \alpha_l$ such that

$$\boldsymbol{v} = \sum_{i=1}^{l} \alpha_i \phi(\boldsymbol{x}_i). \tag{2.10}$$

By substituting equations 2.8 and 2.10 in 2.9 and defining a matrix $K \in \mathbb{R}^{l \times l}$ via $K_{ij} = (\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j))$, we get an eigenvalue problem of the form

$$l\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha}\,, \tag{2.11}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)^T$ is column vector. Problem 2.11 is solved for nonzero eigenvalues and the corresponding eigenvectors are normalized in F. By using equations 2.9 and 2.10, we get

$$1 = \sum_{i,j=1}^{l} \alpha_i^k \alpha_j^k (\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)) = (\boldsymbol{\alpha}^k \cdot K\boldsymbol{\alpha}^k) = \lambda_k(\boldsymbol{\alpha}^k \cdot \boldsymbol{\alpha}^k)\,. \tag{2.12}$$

Then for any test point $\phi(\boldsymbol{x})$, its corresponding principal component is extracted by projecting it onto the Eigenvectors $\boldsymbol{v}^k$ in F:

$$(\boldsymbol{v}^k \cdot \phi(\boldsymbol{x})) = \sum_{i=1}^{l} \alpha_i^k (\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}))\,, \tag{2.13}$$

where $k = 1, \ldots, l$..

2.4.2 Singular Value Decomposition

The *Singular Value Decomposition* (SVD) is a matrix decomposition method which can be used to remove noise from the data or reduce the data size. Given a real matrix $A \in \mathbb{R}^{m \times n}$ with m rows and n columns, the SVD performs a factorization $A = U\Sigma V^T$, where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ are real orthogonal matrices and $\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix. The diagonal entries $\sigma_i = \Sigma_{i,i}$ are known as the *singular values* of A, and the rank of the matrix A is equal to the number of non-zero values in Σ.

2.4.3 Fourier Transform

A physical process that can be described in time domain via a time-dependent function $f(t)$ can equivalently be expressed in the *frequency domain* with a function $F(\omega)$. These two expressions are interrelated via the *Fourier transformation*:

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{i\omega t} dt \,, \tag{2.14}$$

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{i\omega t} dt \,. \tag{2.15}$$

When a finite number of sample points is considered, then the discrete Fourier transform of the N points denoted by F_n is given by

$$F_n = \sum_{k=0}^{N-1} f_k e^{2\pi i k n / N} \,, \tag{2.16}$$

$$f_k = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{-2\pi i k n / N} \,. \tag{2.17}$$

A discrete Fourier transform of length N can be decomposed into even-numbered and odd-numbered points. Then by using recursive computations, the complexity for computing the discrete Fourier transform can be reduced from $O(N^2)$ to $O(N \log N)$ by an algorithm known as *Fast Fourier Transform* (FFT).

2.4.4 Multiple Linear Regression

Consider a vector of features $\boldsymbol{X}^i = (X_1^i, X_2^i, ..., X_p^i)$ with the corresponding labels Y^i. *Multiple Linear Regression* (MLR) uses the following simple model to predict Y given a

column vector[2] $\boldsymbol{X} = (1, X_1, \ldots, X_p)^T$:

$$\hat{Y} = (\boldsymbol{X})^T \hat{\beta}\,, \tag{2.18}$$

where $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_p)$, a constant variable and 1 is included in $\boldsymbol{X}$ and is multiplied with the bias $\hat{\beta}_0$. Let us suppose $\boldsymbol{X}$ is an $N \times p$–matrix with each row as a feature vector, and $\boldsymbol{y}$ is the vector of the labels in the training set. The method of least squares is a popular choice to fit the linear model to this data: the coefficients β_k are selected to minimize the residual sum of squares (RSS)

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - x_i^T \beta)^2\,. \tag{2.19}$$

If $\boldsymbol{X}^T\boldsymbol{X}$ is non-singular, then the unique solution for $\hat{\beta}$ is given by

$$\hat{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{y}) \tag{2.20}$$

2.4.5 K-Nearest Neighbours

The *k-Nearest Neighbours* (KNN) method is an instance-based non-parametric machine learning algorithm. For a given input vector, the output value is computed by averaging or majority voting of the labels of its k nearest neighbours within the training data set. More specifically, the KNN prediction $\hat{Y}$ for a given input x is defined as

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i\,, \tag{2.21}$$

where $N_k(x)$ is the set of the k nearest neighbours $x_1 \ldots, x_k$ to the point x according to a pre-selected distance metric. An optimised value for k is examined in the training phase of the algorithm. For example, if $k = 1$, the input vector is simply assigned the label value of its nearest neighbour within the training data. It is useful to use a weighted average, giving more weight to closer neighbours. The response time of the naively implemented algorithm depends linearly on the size of the training data: for each prediction, the selected distance metric from the new input is computed for the entire training data set, which is computationally very intensive for large amounts of data.

[2]Note that the constant value 1 is assumed to be included in the features, which allows for the constant bias $\hat{beta}_0$ to be added to the prediction in (2.18).

2.4.6 Support Vector Machine

The *Support Vector Machine* (SVM) is a supervised learning method [118] which computes optimal hyperplanes in order to separate and thereby classify data. In cases where the data is not linearly separable, a nonlinear transformation into a higher-dimensional space can be used to achieve separability. However, instead of explicitly mapping the data into the higher-dimensional space, the so-called *kernel trick* can be employed: using *kernel functions*, the algorithm can operate directly in the higher-dimensional space using dot products. During the training, the optimal hyperplanes are selected based on the maximum margin of separation between any training point and the hyperplane determined by solving a corresponding optimization problem. For regression problems, these hyperplanes are described by the coordinates of the support vectors.

In this work, for the regression problems ϵ-*Support Vector Regression* (SVR) and for the classification problems ϵ-*Support Vector Classification* (SVC) algorithms, are used [15]. Following kernel functions are used for the learning process: The *polynomial kernel*

$$k(X, Y) = (X^T Y + r)^p \,, \tag{2.22}$$

where p is the power of the polynomial and r is a shifting parameter; the *Gaussian kernel*

$$k(X, Y) = e^{\frac{(-(||X-Y||)^2)}{(2\sigma^2)}} \,, \tag{2.23}$$

where σ is an adjustable parameter; and the *sigmoid kernel*

$$k(X, Y) = \tanh(\rho X^T Y + r) \,, \tag{2.24}$$

where ρ is the scaling parameter of the input data and r is the shifting parameter controlling the threshold.

2.4.7 Artificial Neural Networks

Artificial Neural Networks (ANN) [104, 75] are a non-linear supervised learning method based on a network of so-called neurons which are interconnected by weighted links. An ANN "learns" by adjusting the weights to optimal values based on the given training and validation data.

In this work, all ANNs are trained using a backpropagation learning algorithm for the multilayer perceptron topology. The used feedforward ANNs consist of three layers: an input layer, one hidden layer and an output layer. The number of neurons in the input layer is equal to the total number of independent variables and the number of neurons

in the output layer is equal to the number of dependent variables, whereas the hidden layer contains eight neurons for all considered datasets. The input layer receives the input from the independent variables and forwards it to all the neurons in the hidden layer. The neurons in the hidden layer apply their activation function to the weighted sum of their inputs and compute an output. The output layer then computes the predicted value for the dependent variable(s).

The back propagation learning algorithm used here to train the multilayer network consists of two passes. In the forward pass, with the current (initially randomly selected) weights and the input given by the training data, the algorithm produces an output for the dependent variables. An error is then calculated based on the difference between predicted and actual output. In the backward pass, this error is propagated backwards through the network from the output layer to the input layer and weights in the network are modified using the so-called *delta rule*, with

$$\Delta w_{ij}(p) = \beta.\Delta w_{ij}(p-1) + \alpha.x_i(p).\delta_j(p)\,, \tag{2.25}$$

$$\Delta w_{jk}(p) = \beta.\Delta w_{jk}(p-1) + \alpha.y_j(p).\delta_k(p)\,, \tag{2.26}$$

where the indices i, j, k refer to input, hidden and output layers, α is the learning rate, β is the momentum with a value between 0 and 1, $x_i(p)$ and $y_j(p)$ are the output of neuron i in the input layer and j in the hidden layer at iteration p, $\Delta\omega_{ij}(p), \Delta\omega_{jk}(p)$ are the error gradients at the neuron j in the hidden layer and k in the output layer at iteration p.

The output for the test data is computed using the trained neural network as follows: the original input value $x^{(0)} = x \in \mathbb{R}^N$ is used to compute $x^{(l)} = \sigma\left(v^{(1)} + W^{(1)}.x^{(1)}\right)$, where σ is the activation function (applied componentwise) and $W^{(1)}, v^{(1)}$ denote the matrix of input weights and the displacement (or *bias*) vector between the input layer and the hidden layer, respectively. Then the final output value is given by $y = W^{(2)}x^{(2)}$, with $W^{(2)}, v^{(2)}$ denoting the weights and biases between the hidden layer and the output layer.

2.4.8 Naive Bayesian Classifier

The *naive Bayesian classifier* provides a machine learning model based on the principles of Bayesian statistics. The below description follows the introduction of the method by Wu and Coggeshall [148].

For multiple classes y_k, $k = 1, 2, \ldots, K$, let $P(y = y_k|x)$ be the *probability* of the output y being class y_k for a given input x. The value $P(y = y_k|x)$ can be estimated using Bayes' theorem:

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)} \qquad \text{if } P(x) \neq 0\,. \tag{2.27}$$

The probability $P(x|y) = P(x_1, x_2, ..., x_n|y)$ needs to be calculated; note that

$$\begin{aligned} P(x_1, x_2, ..., x_n|y) &= P(x_1|y)P(x_2, ..., x_n|x_1, y) \\ &= P(x_1|y)P(x_2|x_1, y)P(x_3, ..., x_n|x_1, x_2, y) \\ &= P(x_1|y)P(x_2|x_1, y)...P(x_n|x_1, x_2, ..., x_n - 1, y)\,. \end{aligned} \tag{2.28}$$

The Naive Bayesian classifier now assumes that the distribution of each variable x_i is independent of others within the class y:

$$P(x_i|x_1, x_2, ..., x_{i-1}, y) = P(x_i|y) \text{ for all } i \in \mathbb{N}\,.$$

Under this assumption,

$$P(x_1, x_2, ..., x_n|y) = \prod_{i=1}^{n} P(x_i|y)$$

and thus

$$P(y|x) = \frac{P(y)}{P(x)} \prod_{i=1}^{n} P(x_i|y)\,.$$

Finally, for a given input x, the *maximum a posteriori* (MAP) decision rule is used to pick the class with the largest probability:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y|x) = \underset{y}{\operatorname{argmax}}\, \frac{P(y)}{P(x)} \prod_{i=1}^{n} P(x_i|y)\,.$$

Different realizations of the naive Bayesian classifier employ different estimates for the probability distributions $P(x_i|y)$ [148].

2.4.9 Gradient Boosting Decision Trees

Gradient Boosting Decision Trees (GBDT) generalize the ensemble method of *boosting* to arbitrary differentiable loss functions and build an additive model in a forward stage-wise fashion. In each stage, a chosen number of regression trees are fitted to the negative gradient of the binomial or mutinomial logistic regression loss function. For binary classification, only a single regression tree is induced. The method can be used for both regression and multi-class classification problems as well. The hyper-parameters include the number of weak learners (for regression trees), the depth size of each tree, the maximum number of leaf nodes and the learning rate [93].

2.4.10 Adaptive-Network-based Fuzzy Inference System

Adaptive-Network-based Fuzzy Inference System (ANFIS) combines ANN and Fuzzy Logic principles to utilize the benefits of both in a single framework. Since its fuzzy inference system is based on fuzzy rules and can learn approximate nonlinear functions, it is considered a universal estimator. Its architecture is composed of five layers. The first layer is known as fuzzification layer. It takes the input values and computes the membership degree of each function using the premise parameter set. The second layer is known as rule layer. It generates the firing strengths for the rules. The third layer is called normalization layer. It normalizes the computed firing strengths. The fourth layer is called defuzzification layer. It takes the normalized values and consequent parameter set as input and returns defuzzified values. These all values are passed to the last layer which combines them to return the final output [51].

2.4.11 Logistic Regression

Logistic Regression (LR) is a supervised learning classification method which estimates the parameters of a logistic model using maximum-likelihood estimation. When dependent variable has binary values, we use binary LR and in case of multiple values, multinomial LR is used [79].

2.5 Predictive analytics

In recent years, automated data generation tools (sensors, cameras, smart devices, medical instruments etc.) are producing huge amounts of heterogeneous data. Technological advancements make it possible to transmit, store, manage, process and visualize this data, which contains inherent knowledge that can be learned using analytic methods.

The evolution of *predictive* and *prescriptive analytics* through time is shown in Fig. 2.17. The development started in the 1980s, using descriptive analytics by examining the data to determine *what* happened in processes and by showing the static and interactive reporting visually. Diagnostic analytics examines data to determine *why* something happened using drill-down, data discovery, data mining and correlation techniques. Predictive analytics aims to determine what *will* happen using machine learning algorithms on the data. Finally, prescriptive analytics *recommends* actions by applying advanced statistical techniques.

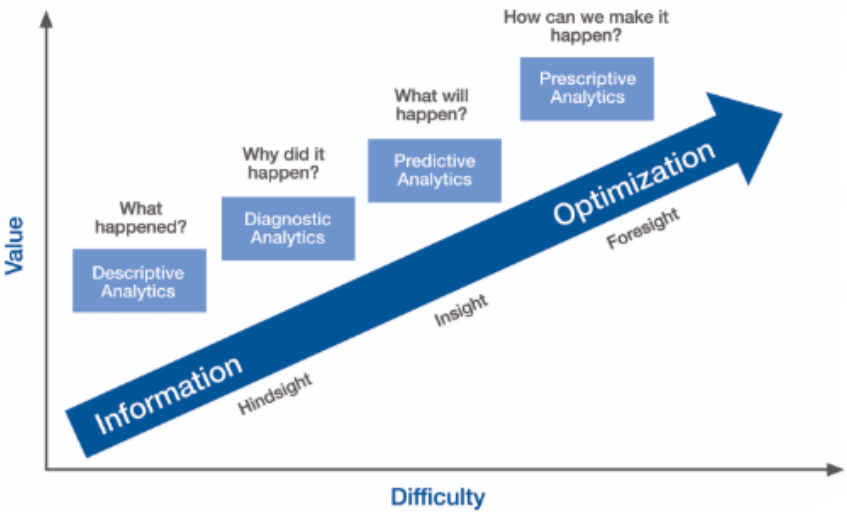


Figure 2.17: Predictive analytics vs other analytics methods [142]

The *CRoss Industry Standard Process for Data Mining* (CRISP-DM) [120] is a European-funded non-proprietary initiative that started in September 1996 with the aim that the knowledge discovery process should be reliable and reproducible even for people with little data mining experience.

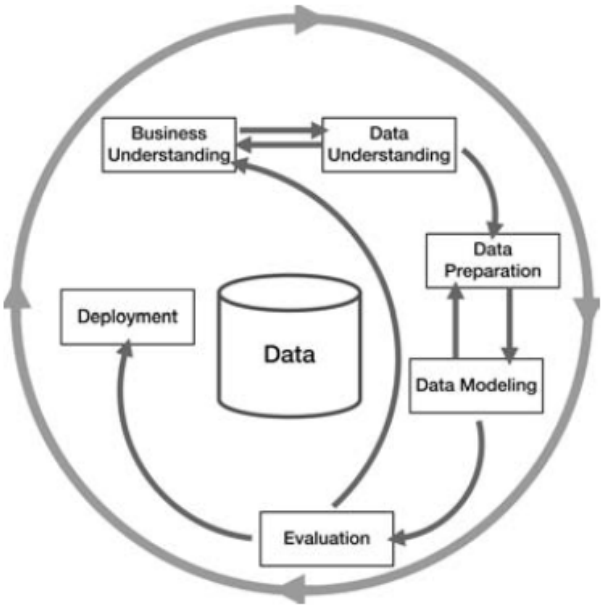


Figure 2.18: Phases of CRISP-DM [47]

It can be used as a guideline and provides analysis templates in the form of Experience Collection, and it is neutral regarding application fields and applications. The life cycle of CRISP-DM process model consists of six phases as shown in the Fig. 2.18.

- In the *business understanding* phase, business objectives and success criteria are determined which are expected to be achieved using data mining. An inventory of resources, requirements, assumptions and constraints is taken; risks, costs and benefits are assessed. Based on this study, a project plan is made.
- In the *data understanding* phase, initial data is collected, described, explored and its quality is verified.
- In the *data preparation* phase, data is collected, selected, cleaned, constructed, integrated and formatted. This phase needs most time and effort of the whole project life cycle.
- In the *modeling phase*, a modeling method is selected based on the data mining target and the collected data, and a test design is created. The model is then built and assessed using varying parameter settings.
- In the *evaluation phase*, the data mining results are assessed with respect to the business success criteria. The evaluation results are used to accept or reject the model.
- In the *deployment phase*, an accepted model is deployed, a monitoring and maintenance plan is made and the final report is produced.

2.5.1 Potential of predictive analytics in manufacturing

Predictive analytics can help to achieve a number of improvements in the manufacturing industry [1]:

- fault Prediction and preventive maintenance,
- demand forecasting and inventory management,
- price forecasting and optimization,
- automation,
- computer Vision Applications,
- managing supply chain risk.

2.5.2 Application of predictive analytics in different fields

As shown in Figure 2.19, predictive analytics has already been applied in a variety of fields. In 2012, 51% of its overall application was in the financial services, business intelligence and the marketing industry [32]. In power engineering, predictive analytics has been used to determine the peak loads and electricity prices in order to determine the efficient distribution of electric power from plants during day and night. It has also been applied in the health sector and to predict the crime rates. The automotive sector is also intensively making use of predictive analytics, with companies like Daimler AG and BMW Group using it to reduce the waste and to perform predictive maintenance.

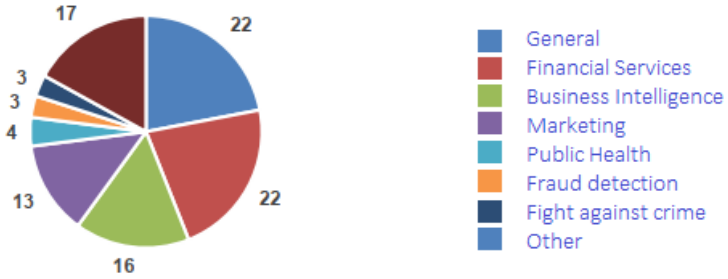


Figure 2.19: Fields of application [32]

3 Literature review

Machine Learning can be effectively used in production environments. It promotes efficient and reliable production and helps manufacturing industries bring greater efficiency to many different areas. In preventive maintenance, for instance, ML algorithms can assess the condition of machines or tools and predict the optimal time to service or replace them [10, 135, 153].

In quality management, ML models can be used to monitor and predict product quality based on process data, and thereby replace checking random samples [9, 57]. The use of ML in process control can lead to greater adaptability in changing conditions, stabilization of product quality and a simultaneous decrease in the level of rejects [103].

ML based object recognition and motion planning have recently led to important advances, especially in the field of robotics [13]. Other exemplary applications include shop floor planning [33], fault diagnosis [66], and power management [16, 2].

As in the case of Industry 4.0, the potential of ML can be derived from technological capabilities ("engine of opportunity") or the existing need for optimization ("engine of problems") [11]. Since ML can only be applied if the identified manufacturing problem or optimization opportunity is suited for it, choosing the right use case is critical. Techniques such as value chain mapping, Ishikawa diagrams, process capability analysis, or expert presentations can help identify areas appropriate for ML applications. The selected use cases are also identified and analysed in terms of cost-benefit ratio.

For ML projects, the availability and quality of data greatly impacts this analysis; however, additional sensors, storage and processing capabilities require additional investment. The availability and quality of data is a crucial prerequisite for using ML [91]. According to RAMI 4.0, a reference framework for digitization, this includes data from product, control and field devices, stations, work centers, the company itself to suppliers and customers. Despite the ambitious implementation of the Industry 4.0 asset management shell in OPC UA [73], aimed at standardizing data exchange, the ability to connect new and old plants remains a serious problem [78, 132]. Unfortunately, it is often not known in advance what data is ultimately needed to train a model with sufficient accuracy [70]. When choosing an algorithm, an important role is played by its reliability, that is, the reaction to unexpected situations, as well as its interpretability [113]. Based on this, prioritizing and choosing the most appropriate use cases, for example energy demand forecasting [94], is critical.

For project execution, the CRISP-DM iteration cycle with its six phases is the most common method [120, 64]. While CRISP-DM provides a rough guide to start ML projects,

specific instructions and methods are still lacking. One approach is to combine CRISP-DM with well-structured Six Sigma methods and tools [30]. Beyond project execution, there is also a lack of a specific method that incorporates industry characteristics to identify appropriate use cases. A more structured presentation of use cases would provide an improved basis for aligning with manufacturing concerns.

The number of connected devices is growing exponentially. In 2015, the number of connected devices was five billion; in 2017 it exceeded eight billion [67] and we currently have a forecast of 29.4 billion for 2030 [125]. This gives us an insight into the large amount of data that must be processed by these devices and the involvement of companies to get the most out of them. The Internet of Things has evolved more in enterprises called the *Industrial Internet of Things* (IIoT) compared to the consumer IoT that is evolving outside of them. IIoT is defined as the collaboration between machines, computers and people that perform manufacturing operations using advanced data analytics for the benefit of the business [49].

With the rapid technological advancement, the Industry 4.0 concept is becoming reality. Production companies are utilizing these advancements to evolve into being smart and interconnected by integrating information and communication technologies. Sensors are being used widely to get real time production process data. This data is being used to learn the intrinsic knowledge in this data using data-driven methods and to make decisions based on it. In the following, we provide a brief chronological overview of prior applications of Machine Learning techniques in industrial settings in general and, more specifically, in the metal industry.

3.1 Machine learning methods applied in industrial settings

[1990] Hansen et. al. used cross-validation techniques to optimise neural network architectures and parameters and employed ensembles of similar neural networks to improve the performance of neural networks for classification problems [43].

[1992] Upadhyaya et al. [138] applied multiple-input multiple-output autoassociative back propagation neural network for sensor and process monitoring in power plants to estimate several process variables.

[1994] Alexander et al. [92] used an adaptive back propagation neural network algorithm to develop a robust fault detection and identification system for incipient faults occurring in process systems. The process investigated was composed of a direct current motor with regulated speed using a proportional-integral type controller, a centrifugal pump and an associated piping system. Boon [50] used artificial neural

networks on synthetic data to train the microcontroller. Simulation results showed that the learnt model could successfully control the operating characteristics of the nuclear power plants.

[1995] Cortes et al. [20] proposed a method to estimate the performance limits based on the data quality for any machine learning algorithm whose parameters can be varied. They showed the similarity of the results by applying neural networks and learning vector quantization on a dataset to predict failure in telecommunication paths. Shukla [121] applied a generalized feedforward neural network model as well as an algorithm based on adaptive optimisation to control a simplified aircraft turbo engine simulation successfully. He showed that the algorithm based on adaptive optimisation worked comparatively better than the backpropogation method. Sjoberg [122] showed the importance of regularization in optimization (minimization) for parameter estimation, related it to early stopping in neural networks using validation dataset and explained its positive effects on the variance of the parameter estimates and in keeping the variance error small for models with many parameters.

[1996] Sharkey [119] gave an overview of the methods used by the research community to combine different neural nets, which includes ensemble methods like bagging, boosting and modular based approaches, such as divide-and-conquer.

[1997] Sola et al. [123] used back propagation neural networks for estimation and identification problems in nuclear power plants and showed that data normalization results in improved prediction accuracy in fewer iterations.

[2010] Yang et al. [151] reviewed the ensemble learning methods with focus on their application in bioinformatic problems. They identified and summarized the future trends of the ensemble methods in this field.

[2011] Kalidindi [55] built efficient microstructure databases to add and capture knowledge from datasets produced by multiple groups and demonstrated fast scale-bridging modeling and simulation of material phenomena.

[2012] Gröger et al. [39] used indication-based and pattern-based approaches provided by an Advanced Manufacturing Analytics Platform to overcome the limitations that existed in analytics in manufacturing. They also demonstrated their usefulness with examples and proposed suitable data mining techniques with implementation guides.

[2013] Gröger et al. [38] presented an Operational Process Dashboard for Manufacturing targeted at shop floor workers. It provided information on process context, performance, knowledge and communication. Karimi et al. [46] developed a new expert

system for statistical process control in the manufacturing industry. This system recognises critical process and quality characteristics of the system. It also extracts rules from different sources as the abstract knowledge in various domains. The efficiency of the system was demonstrated by implementing it in several manufacturing industry processes. Lieber et al. [68] presented a framework based on supervised and unsupervised Machine Learning to identify operational patterns, quality related features and production parameters, and to predict physical quality of intermediate products in interlinked manufacturing processes in a rolling mill case study.

[2014] Zheng et al. [156] developed a data analytics platform Plasma Display Panel (PDP) Miner for process optimisation in PDP manufacturing. This platform can automatically configure and schedule analysis tasks, and balance heterogeneous computing resources.

[2015] Chen et al. [17] used a Principal Component Analysis (PCA) Back Propagation Network approach to estimate factory simulation workload in cloud manufacturing. They tested and showed improved estimated accuracy using data from 90 simulation models. Sahno et al. [110] introduced a framework which identifies the most critical operations in the process influencing Key Performance Indicators (KPIs), which allows engineers to define, measure and analyse failure with less effort and results in decreased production lead time and increased product throughput.

[2016] Niesen et al. [84] presented an integrative big data analysis framework for data-driven risk management in Industry 4.0 to efficiently manage business processes and process risks. Ray et al. [99] used gray relational analysis associated with PCA to optimize process parameters of green electrical discharge machining. Sata [116] used ANN and multivariate regression together with PCA to predict investment casting defects. Wuest et al. [149] presented an overview of machine learning techniques, their advantages, challenges, and applications in manufacturing. Zhang et al. [154] used evolutionary computation for feature selection and feature construction.

[2017] Duarte et al. [29] empirically compared cross-validation techniques and internal metrics for tuning SVM hyperparameters. Liu et al. [69] used wavelet and support vector machine to predict machinery condition. Nikolic et al. [86] gave an overview of predictive manufacturing systems in Industry 4.0 and discussed their trends, benefits and challenges. Preuveneers et al. [95] wrote a survey on emerging trends, research challenges and opportunities in Industry 4.0.

[2018] Bai et al. [4] used two intelligent learning approaches, shallow learning and deep learning, to predict manufacturing quality. For shallow learning, they used feed forward neural networks and least squares support vector machines; for deep learning,

they used a deep restricted Boltzmann machine and a stack autoencoder. They concluded that deep learning outperformed shallow learning in terms of chosen performance measures. Tao et al. [129] wrote about the role of big data to support smart manufacturing. Zacarias et al. [152] proposed a framework to generate analytic solutions in manufacturing based on a systematic profiling so that users can explore relevant alternatives for their specific scenario, and obtain recommendations in terms of quality measures.

[2019] Bai et al. [5] compared three dimension reduction techniques, namely principal component analysis, locally linear embedding and isometric mapping, and used support vector machines for modeling multi-parameter manufacturing to predict manufacturing quality. Bashar [7] used intelligent big data analytics and cloud computing to improve the manufacturing process. Carvalho et al. [13] presented a lterature review of machine learning methods applied to predictive maintenance and presented the performance of state-of-the-art machine learning methods. Cavalcante et al. [14] developed a hybrid technique based on simulation and machine learning and applied it to data-driven decision-making support in resilient supplier selection with on-time delivery as the benchmark. Dai et al. [21] discussed the necessities and challenges of big data analytics in IoT manufacturing and surveyed its enabling technologies. Lahdhiri et al. [65] used the rank-reduced (RR) KPCA for fault detection in online processes and proposed to use partial RR-KPCA to identify the variables correlated to the fault occurred. Nzuva et al. [87] gave a comprehensive overview of the different ensemble classification methods like bagging and boosting and discussed how base learning algorithms are combined together using these methods. Ren et al. [100] used locally weighted partial least squares (LWPLS) for industrial soft sensor modeling and proposed a two-phase bandwidth optimization strategy combining particle swamp optimization and LWPLS. Ren et al. [101] wrote an comprehensive review about - and combined the key technologies of - smart manufacturing. They gave an overview of big data in smart manufacturing and proposed a conceptual framework for product lifecycles. Tian et al. [134] introduced an incremental learning ensemble strategy which aggregates multiple sublearning machines with different weights. When new dataset is collected from the automated industrial processes, this strategy adds a new trained submachine and updates the weights of the existing submachines based on their performance on this data set. They showed its superiority in comparison to other Extreme Learning Machine methods. Weichert et al. [144] wrote a literature review about machine learning and optimization approaches carried out for process improvement in the manufacturing industry during 2008-2018. Wolf et al. [147], using a systematic mapping review, identified seven application areas where huge improvements can be made along with the relevant

advanced analytical techniques for each of them.

[2020] Dai et al. [48] presented a multi-model soft sensor to estimate the floating height of the strip in an air cushion furnace based on state identification and soft transitions. The KNN method and PCA were used for partitioning and a double-random forest model was employed for the vibration state identification.

3.2 Machine learning methods applied in metal industry

[2007–2008] Khanzode et al. [59] investigated cluster analysis and path modeling to improve foundry process control; in 2008, they extended their work by implementing a Mahalanobis-Taguchi system to improve the casting quality in grey iron foundries [60].

[2008] Rai et al. [97] developed and used a neural network based casting process model on data generated by ProCast (an FEM-based flow simulation software) for a high pressure die casting process and used four features, namely inlet melt temperature, mold initial temperature, inlet first phase velocity and inlet second phase velocity, to predict filling time, solidification time and porosity simultaneously. They demonstrated an improved performance in comparison to other models available in the literature. Tsoukalas [136] determined optimum conditions to minimize porosity in AlSi9Cu3 aluminium alloy die castings by using multivariable linear regression and genetic algorithms for model generation on data generated by three-stage experiments by varying holding furnace temperature, die temperature, plunger velocities in the first two stages and multiplied pressure in the third stage using L27 Taguchi orthogonal arrays.

[2009–2010] Santos et al. [115] used machine-learning methods to predict mechanical properties in foundry production processes and, in 2010 [114], used machine-learning methods to predict defects in high-precision foundry production. Zheng et al. [155] proposed an evaluation system to quantify the surface defects in high pressure die casting and used artificial neural networks to generalize the correlation between die-casting parameters and surface defects. The trained neural network was used to optimize the parameter values to achieve acceptable surface quality.

[2011] Tsoukalas [137] created a tool based on an adaptive neuro-fuzzy inference system and generated experimental data by performing selective experiments using orthogonal arrays and casting parameters such as metal and die temperature, piston and die

gate velocity, and solidification pressure to study the effect of these parameters on porosity formation in AlSi9Cu3 pressure die castings.

[2012] Susanta et al. [25] used a neural network based expert system to compute the tensile behaviour of tailor welded blanks made of dual-phase steel. Santos et al. [85] used a combination of machine learning algorithms to predict faults in high-precision foundries.

[2013] Hossam et al. [31] used soft computing techniques to model a hot rolling manufacturing process. Ransing et al. [98] used a coupled penalty matrix approach and a principal component based co-linearity index technique to discover product-specific foundry process knowledge from in-process data in order to reduce defects.

[2015] Batbooti et al. [8] used Principal Coomponent Analysis (PCA) to discover knowledge from foundry data, defined optimal limits for factors and considered interactions among them. Since this approach discovers factor tolerance limits contributing most to the overall variance, process engineers can adjust system parameters with better insight.

[2017] Chokshi et al. [18] successfully used artificial neural networks to train a phase distribution prediction model to control localized microstructures for environment friendly and safe 22MnB5 boron steel with the help of tailored temperature rates for localized zones.

[2018] Dib et al. [26] used machine learning methods including multilayer perceptrons, classification and regression trees, naive-bayes, random forest and support vector machines to predict defects in sheet metal forming processes and compared their performance. Hamouche et al. [42] used a deep convolutional neural network method to identify a suitable manufacturing process in sheet metal forming.

[2019] Fragassa et al. [35] performed pattern recognition analysis using methods including k-nearest neighbours, artificial neural networks and random forests to successfully predict the tensile behaviour of metal cast alloys and discussed the application of these techniques in material design and process quality control. Matos et al. [22] proposed the use of waste foundry sand (WFS) in conventional concrete by WFS calcination and in dry-mix concrete for the production of concrete blocks. They showed that its use resulted in reduction of flow and compressive strength of the mortars. He et al. [44] used neural networks to detect molten steel levels. They learnt the features of the temperature gradients using neural networks and used the convolution of neural networks to detect the flux-steel interface from the temperature gradient distribution. Kenkin et al. [58] used support vector machines, neural

networks and logistic regression together with the dimension reduction techniques of PCA and isometric feature mapping for fault prediction during the molding process of a wheel rim manufacturer. Ruiz et al. [106] used the KNN method, random forests and ANN together with undersampling, oversampling and Synthetic Minority Oversampling Technique (SMOTE) on unbalanced process data to optimize the fabrication of cold drawn steel wire and successfully reduced the rejection rate.

[2020] Ruiz et al. [107] employed basic machine learning algorithms, namely MLR, KNN, Classification and Regression Trees, ANN, as well as ensemble methods, namely Random Forest, Gradient Boosting and Adaboost, in order to predict the strength of steel rods. They identified the important features influencing the material strength using the feature importance and permutation importance algorithms and visually showed their results with partial dependence plots.

4 A unified machine learning framework

In order to address some of the difficulties associated with using machine learning techniques in the metal industry (as described in Sections 1 and 3), we propose a *unified framework* for the application of different ML methods to monitoring and control problems in the foundry industry. This framework aims to simplify the application process

4.1 Overview of the framework

The proposed framework works in 3 three stages, as shown in Fig. 4.1. In stage 1, the *Meta Prediction Function* (MPF) is learnt from the process data. In stage 2, *knowledge* is generated from the learnt Meta Prediction function. In stage 3, the MPF is used to *monitor* the running process by predicting the dependent variable values for the current independent variable values in advance and suggesting changes when the results are not in the acceptable range. In Sections 4.2–4.3, these individual stages will be explained in more detail.

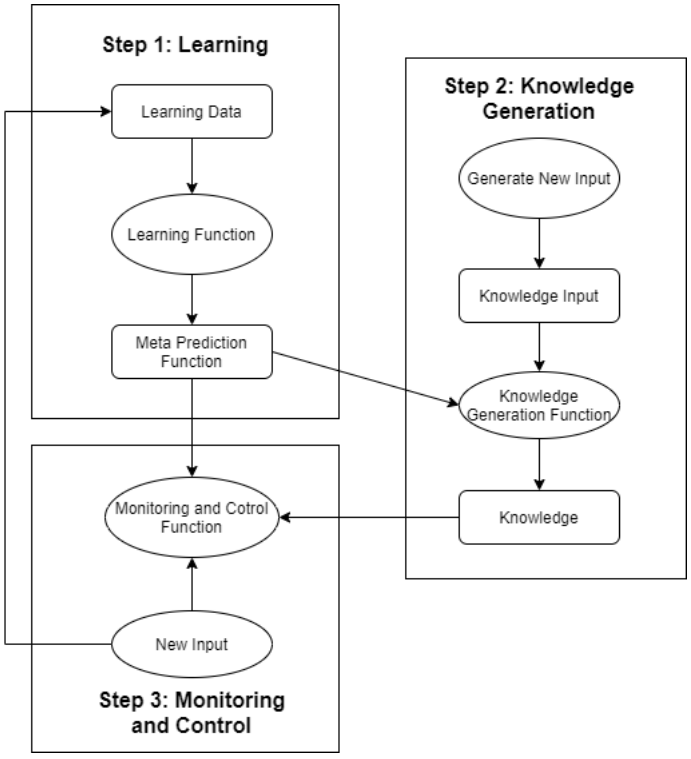


Figure 4.1: Overview of the proposed framework

In the Learning stage, a data-driven non-linear model for the process under consideration is learnt based on labeled data captured from the process. This dataset must be composed of functionally independent variables (features or inputs to the process) and functional dependent variables (features or outputs from the process being monitored as quality characteristics). In the start of the learning stage, if the amount of data is too large, if missing values or outliers exists, or if independent variables are interdependent or the data is unbalanced, then feature mapping techniques are applied to improve the data quality. Once the complete data is of good quality, it is divided into learning data (90%) and testing data (10%). Learning data is used to train and validate the selected ML methods, while the testing data is used to compute the unbiased[3] evaluation of the learnt model. Thereby, it is ensured that performance evaluation is based on test data which had no influence on model selection. Each ML method is trained on z-score normalized data using k-fold cross validation; the z-score normalization is performed to transform all the variables to the same scale, while k-fold cross validation ensures that the complete set of learning data is used for training as well as validation. An error threshold is selected beforehand to consider only those ML methods which perform within an acceptable performance range. Once the learning is complete and a ML method is selected, then its unbiased performance is calculated using the test data.

In ML, there are many learning methods available which might provide good prediction results for a given data set. In the proposed framework, a number of well-performing algorithms are selected, and each of them gives a prediction for a given input. In order to obtain a single prediction output value, these predictions are then combined using ensemble methods. In a previous contribution [3], a novel method called *Meta Prediction Function* (MPF) was proposed for this purpose. As shown in Fig. 4.7, the predictions of different ML methods are considered as new features which are, in turn, the input for the MPF, which combines them into a final, single prediction. A straight-forward approach to achieve such a combination can be to take the average of the predicted labels from the ML methods; however, experiments have shown that this method is generally less accurate than the best-performing individual ML method. Note that, since all involved models are attempting to predict the same unknown value, these variables must be expected to be highly collinear. Therefore, PCA is applied to this data in order to first extract the Principal Components, which are linear combinations of the generated input variables (i.e. of the individual ML algorithms' predictions). Since the dependent variable is already known for the learning data, it is used as a dependent variable again and combined with the calculated Principal Components to train the linear model using MLR. The resulting composite model composed of individual ML methods with post processing by Principal

[3]Here, the term "unbiased" is used in the context of the bias-variance dilemma and is not to be confused with the statistical notion of bias.

Component Regression [53] is stored for later predictions. Using the original test data, an unbiased evaluation of the composite learning method can then be determined.

Once the learning stage is completed and a data-driven non-linear model is set up, it can be used for the Knowledge Generation stage as shown in the Fig. 4.1. In this stage, for each independent variable a minimum and a maximum value is selected; by default, these values are selected from the complete dataset, but they can also be modified based on domain knowledge and expert recommendations. Then, for real-valued independent variables, a number of steps is chosen for data generation, while for discrete variables, the original values are retained. Based on all possible combinations of values for the independent variables within the selected range, a large amount of input data is then generated. The output values for these generated input values are predicted using the composite learning method and the complete data, which represents the generated knowledge, is stored in a database.

Finally, the Monitoring and Control Stage, as shown in the Fig. 4.1, is employed in the actual industrial environment. The current state of the process is continuously monitored via sensor data, which is used as new input data for the MPF. The quality characteristics for the process are then predicted with the MPF and compared to a range of acceptable values, which needs to be specified beforehand based on domain knowledge about the process. If the predicted values lie within this optimal range, then no changes to the process parameters are required. However, if the MPF predicts a quality characteristic to lie outside the acceptable range, the generated knowledge database is searched for the input values which are closest to the current process state, but result in an output within the optimal range. In this way, if the prediction is sufficiently reliable, it can be determined how the input parameters can be changed in order for the process to remain in a proper working state.

4.2 Learning stage

The learning stage is subdivided into two stages, the first of which is shown in Fig. 4.2. First, the data composed of inputs as well as outputs representing the process is loaded into the system and preprocessed. In the next step, statistical and machine learning methods are used to select optimal independent variables. Next, if the distribution of dependent variable is unbalanced, it is balanced using suitable methods such as SMOTE. In the next step, depending on the type of the problem, regression or classification methods are applied and the corresponding learning and test errors are calculated. If the errors found are below a predefined threshold, then the method is deployed; in case all the errors are above the threshold, then the dataset is passed to stage T.

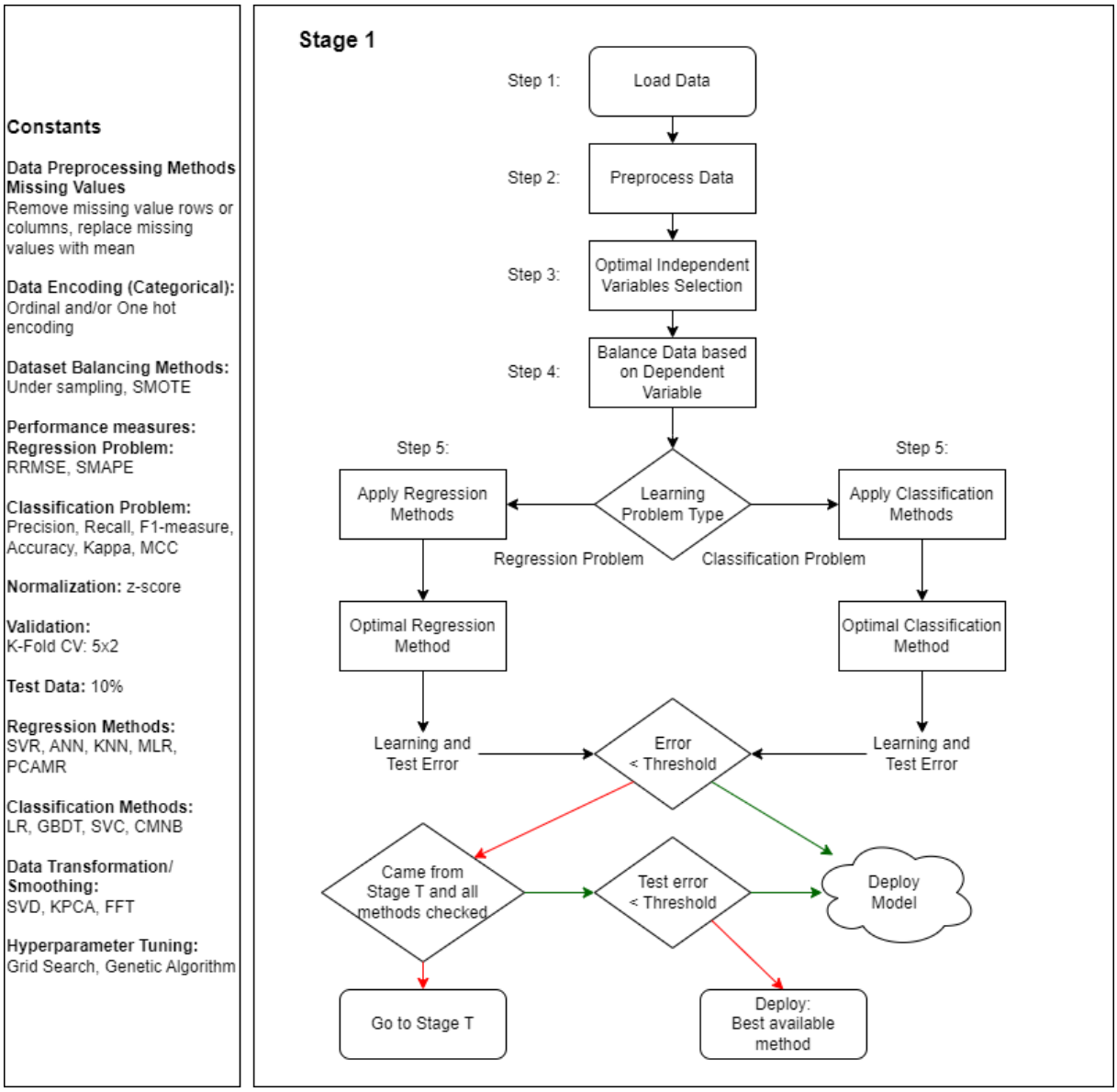


Figure 4.2: Stage one of the learning step

The detailed Machine Learning process is shown in Fig. 4.3. There are two phases in this process, the Learning phase and the Prediction phase. During the Learning phase, the loaded raw data (including the dependent variable) are split into learning and test data. On the learning data, preprocessing and feature engineering methods are applied. The actual Machine Learning step is then applied to the resulting data in order to fit a suitable model. The same transformations are applied to the test data and predictions are made using the learnt model. The learning and test errors are then computed. During

the prediction phase, the model is used to predict output values for new data.

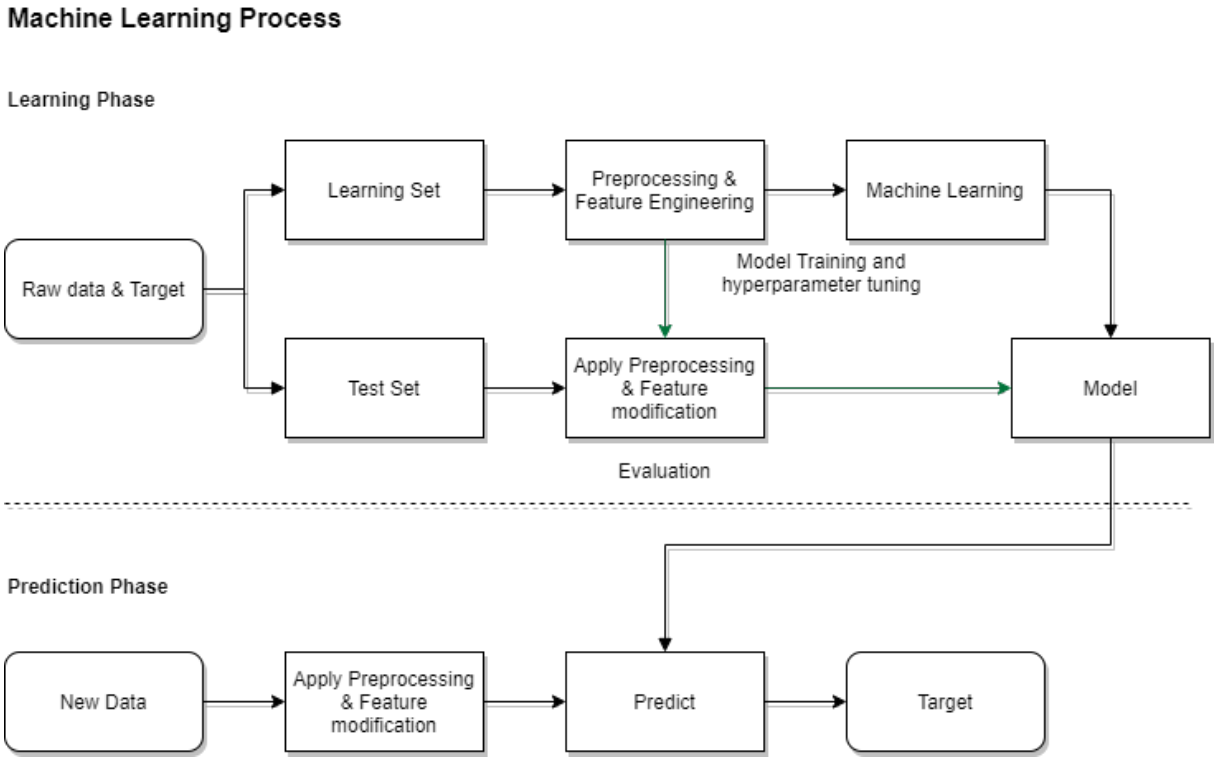


Figure 4.3: Machine learning process during learning Step

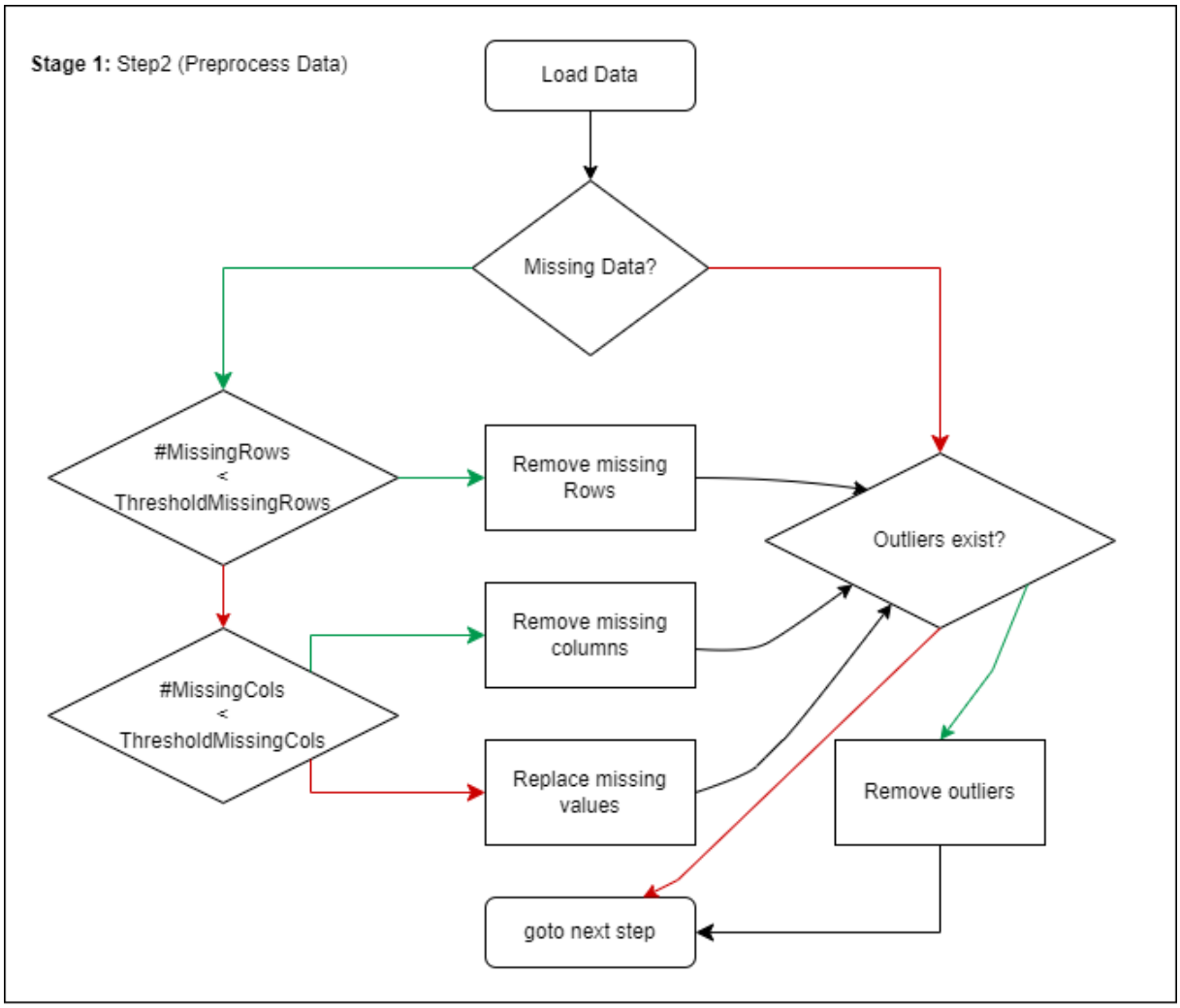


Figure 4.4: Preprocessing step of machine learning during learning step

In stage T of the learning process, data is transformed using SVD and KPCA; thereby,

independent data is transformed into new features. These new transformed features and dependent variables are given to Step 4 to train new models.

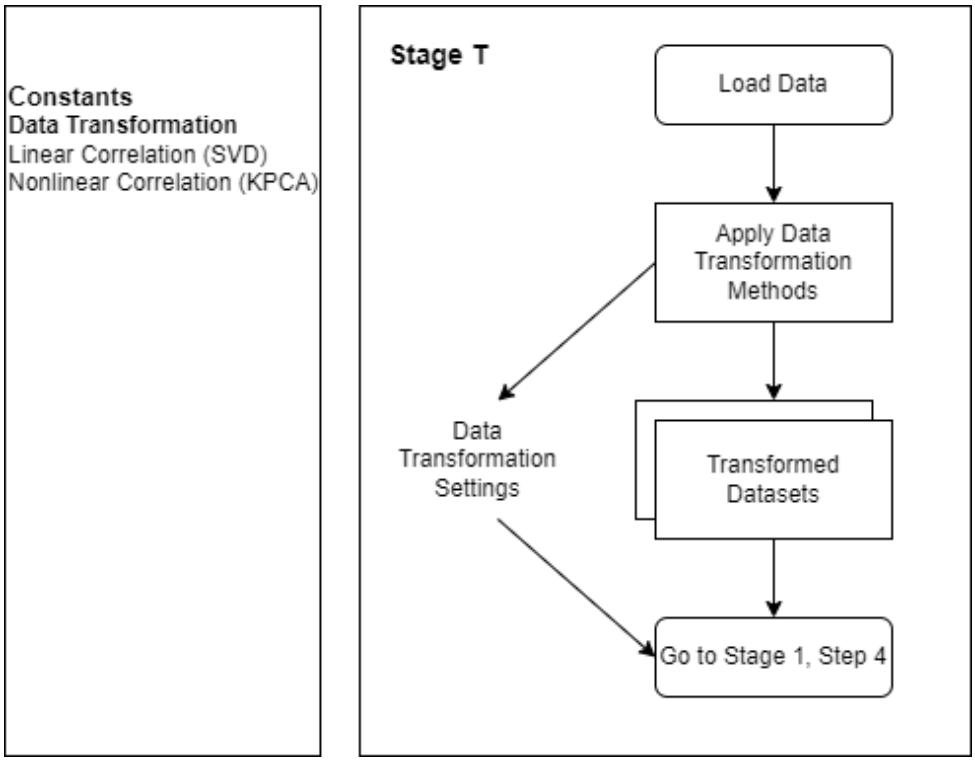


Figure 4.5: Data transformation step of machine learning during learning

4.2.1 Feature selection

The Feature Selection step from stage 1 is shown in Fig. 4.6. While applying machine learning algorithms on data with a high number of variables, the so-called "curse of dimensionality" can lead to major setbacks; the term refers to the phenomenon of data appearing much sparser in high-dimensional spaces, affecting the performance of algorithms designed for computing operations in a lower-dimensional setting. A higher number of features may also lead to overfitting, which can effect the prediction performance on unseen testing data.

Dimension reduction, which can be used to reduce the data size, is categorized into two main approaches:

Feature Extraction converts high-dimensional data to a lower dimension. The features of transformed data are usually linear or nonlinear combination of original data features

based on the transformation method used. Some of the feature extraction methods used are PCA, KPCA, Linear Discriminant Analysis (LDA) and SVD. [41].

Feature Selection is a process of selecting the most significant features out of given feature space. This is done by analysing the features present in the dataset and creating a subset of relevant features which represent the dataset most accurately. Feature selection methods can be further categorized into three distinct categories:

Filter Methods are often used in pre-processing step. These methods are independent of the learning algorithms and depend purely on data to determine the importance of features [41]. They use statistical methods to score the variables based on their correlation to the resulting variable. The variables with the highest scores are then selected and passed to the learning model for training. Among the various correlation methods used are the linear correlation coefficient of Pearson, χ^2 statistics and a family of algorithms known as Relief. These methods are very fast in computation and reduce the data volume significantly, which may result in better performance, but they do not consider multicollinearity between the independent variables [41].

Wrapper Methods work by evaluating the quality of selected features based on their performance using a specific Machine Learning model. To this end, a subset of features is selected and the model is trained and evaluated using only these features. This process is repeated until the required performance or a minimum error is achieved, as shown in the flow diagram shown in Fig. 4.6. The problem faced with wrapper methods is that for n features, the required search space is 2^n, which can require extensive computations for higher values of n, increasing the complexity and the cost of the algorithm and thus making it an inadequate option to be used for feature selection. The search space is therefore often reduced by suitable methods. A commonly used approach is the so-called Forward Search: Variables are added successively and, depending on the change of the model's performance, each variable is either selected or rejected from the main feature set; once a predetermined criterion is reached, the resulting set of selected variables is specified as the new input data set. An alternative approach is given by the Backward Search method, which starts with the complete original dataset and succesively removes features such that the impact on the model's performance remains sufficiently small.

Embedded Methods include qualities of both, filter and wrapper methods: Like wrapper methods, they embed the variable selection procedure using model training. However, embedded methods do not evaluate the features iteratively, which eradicates the problem of exponential search spaces. Commonly used embedded methods are based on regularization models, which focus on minimising the fitting error while reducing feature coefficients to zero, hence removing those features.

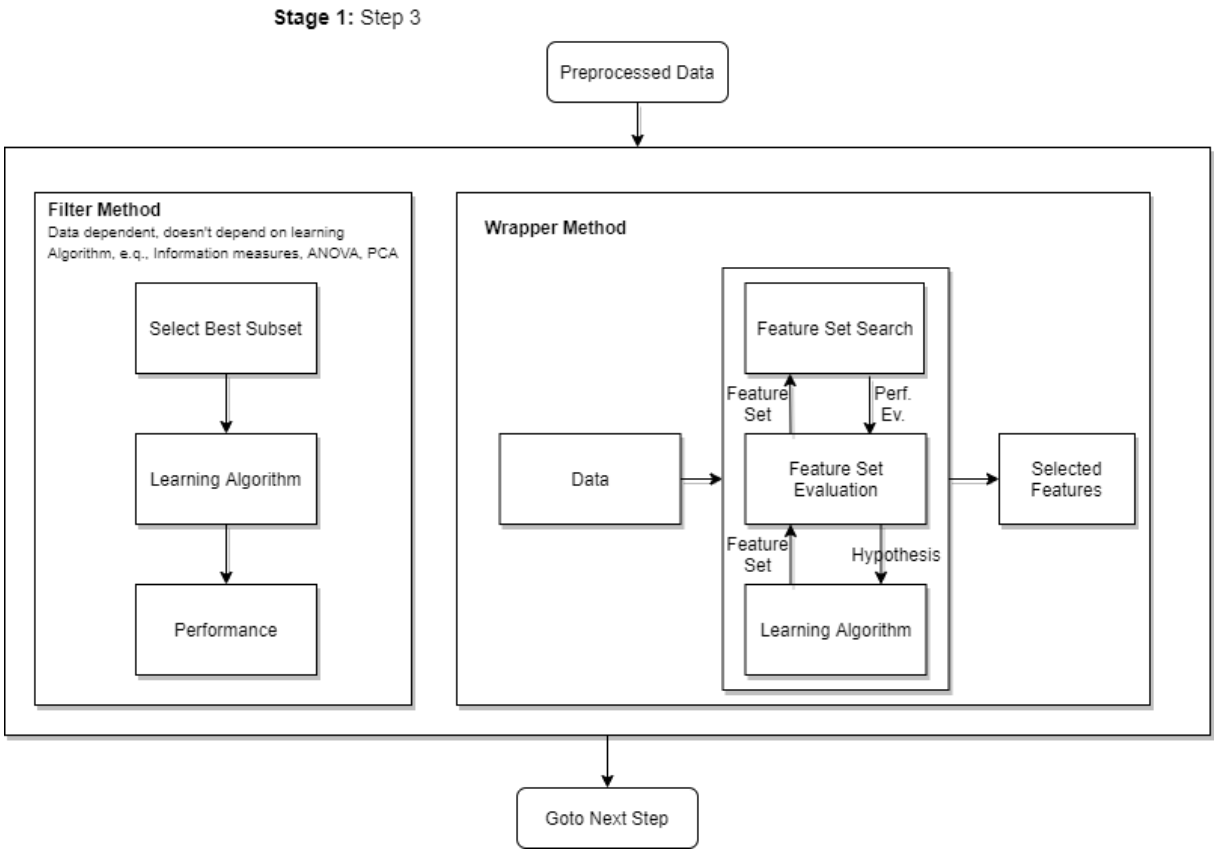


Figure 4.6: Feature selection step during learning

4.2.2 Meta prediction function

The proposed Meta Prediction Function (MPF) is inspired by the principle of ensemble methods [28], which provide an optimized prediction by combining the results obtained from individual ML methods. The main component of our proposed MPF is the combination of KPCA and the accumulator module. This accumulator module is a combination of several base ML algorithms. The learning algorithm for MPF can be summarized as follows:

Step 1: Different ML methods are applied to the training data and every prediction ML_i^k corresponding to the k-th ML method and the i-th entry of the training data set is stored.

Step 2: The predictions from the selected ML methods are pre-processed and multi-collinearity is removed by applying the KPCA to obtain the new intermediate features PC^k:

$$(PC_i^1, PC_i^2, \ldots, PC_i^n) = KPCA(ML_i^1, ML_i^2, \ldots, ML_i^n)\,. \tag{4.1}$$

Step 3: The computed Principal Components obtained from the KPCA are used as the independent variables for a MLR, which is then fitted to the training labels Y_i via

$$Y_i \approx \beta_0 + \sum_{k=1}^{n} \beta_k \, PC_i^k \tag{4.2}$$

to obtain the final prediction function.

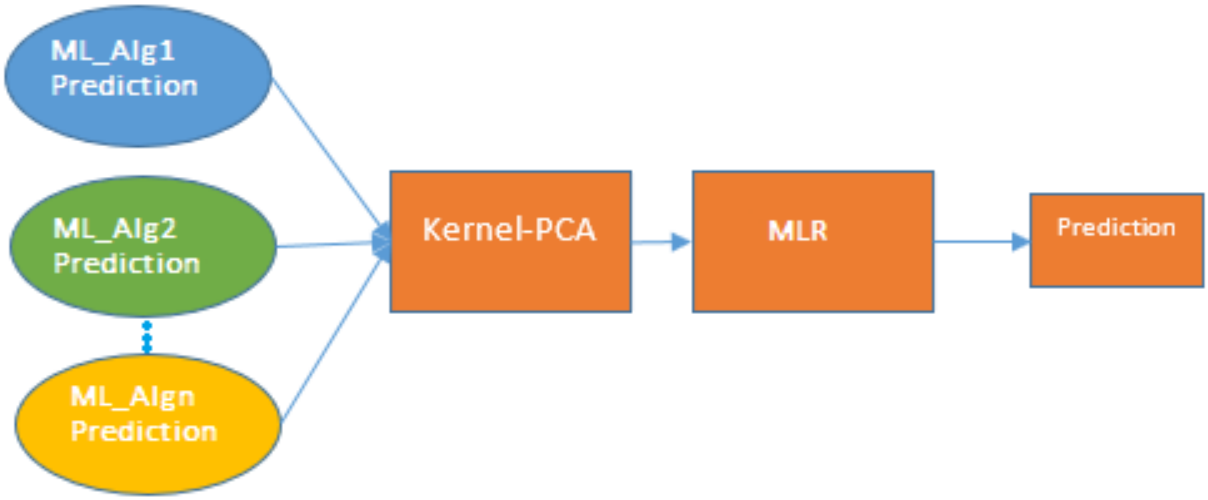


Figure 4.7: Proposed Meta Prediction Function

In order to compute the performance of the MPF on the test data, independent variables of the test data set must be normalized based on the respective mean and standard deviation of training and validation data variables. Its kernel matrix is then determined and its principal components are computed using the Eigenvectors of the training and validation data. The same process is later used to evaluate the MPF on newly obtained data.

4.3 Knowledge generation, monitoring and control

Suppose we have trained a ML algorithm to predict the hardness of a casting part based on its chemical composition[4] and that we want to create castings within a selected range of hardness values. If, during the early stages of the casting process, we can take a sample of the molten steel before solidification and measure its chemical composition),

[4]Note that in practice, the hardness of a casting part is not solely determined by its chemical composition, but that numerous additional process parameters should be monitored during the casting process.

then the hardness resulting from the casting process *under current conditions* can be predicted using the learnt prediction function. If the predicted hardness is not acceptable, then measures can be taken to change the chemical composition such that the achieved hardness is, accoding to the prediction, acceptable. This procedure keeps the process in a running state since potential problems are detected before they occur.

In order to apply ML during the casting process in the way described above, according to the unified framework proposed here, we first need to generate a knowledge database using the trained prediction function (i.e. the MPF) as shown in Fig. 4.9. To this end, first a large set of input data is created by varying the values of each chemical element between a given minimum and maximum, with a specific *number of steps* for each variable. Then, for all combinations of these generated chemical element values, the MPF is employed to predict the dependent variable values (i.e. the hardness of the resulting casting part).[5] This complete *knowledge* is then stored in the database for the later stages.

It should be noted again that the complexity of the created knowledge database depends both on the number of input features and on the number of steps between minimum and maximum for each of the independent variables. More specifically, the number of predictions that need to be computed for k steps and n features is given by k^n. For example, in Section 5, we will consider two datasets (related to the plastic deformation and elongation of steel) with 3 and 17 input features, respectively. For these two datasets, the graph in Fig. 4.8 shows the number of required predictions to generate the knowledge base for up to 20 steps. In particular, for the elongation dataset, 20 steps for each of the 17 features requires predictions for $20^{17} = 1.31072 \cdot 10^{22}$ combinations. To generate this amount of combinations and then to predict and store them in dataset would require millions of years on a regular personal computer. Therefore, the feature selection process described in Section 4.2.1 plays a particularly important role for the proposed framework. The number of input variables can also be further reduced by considering, for example, a coraviance matrix and finding linear relationships between the features and the label. Features with higher SHAP values can also be selected beforehand [74].

[5]Note that this approach is only necessary because the *inverse* of the MPF is not readily available: there is no analytic way to simply find independent variable values which would result in a given dependent variable value.

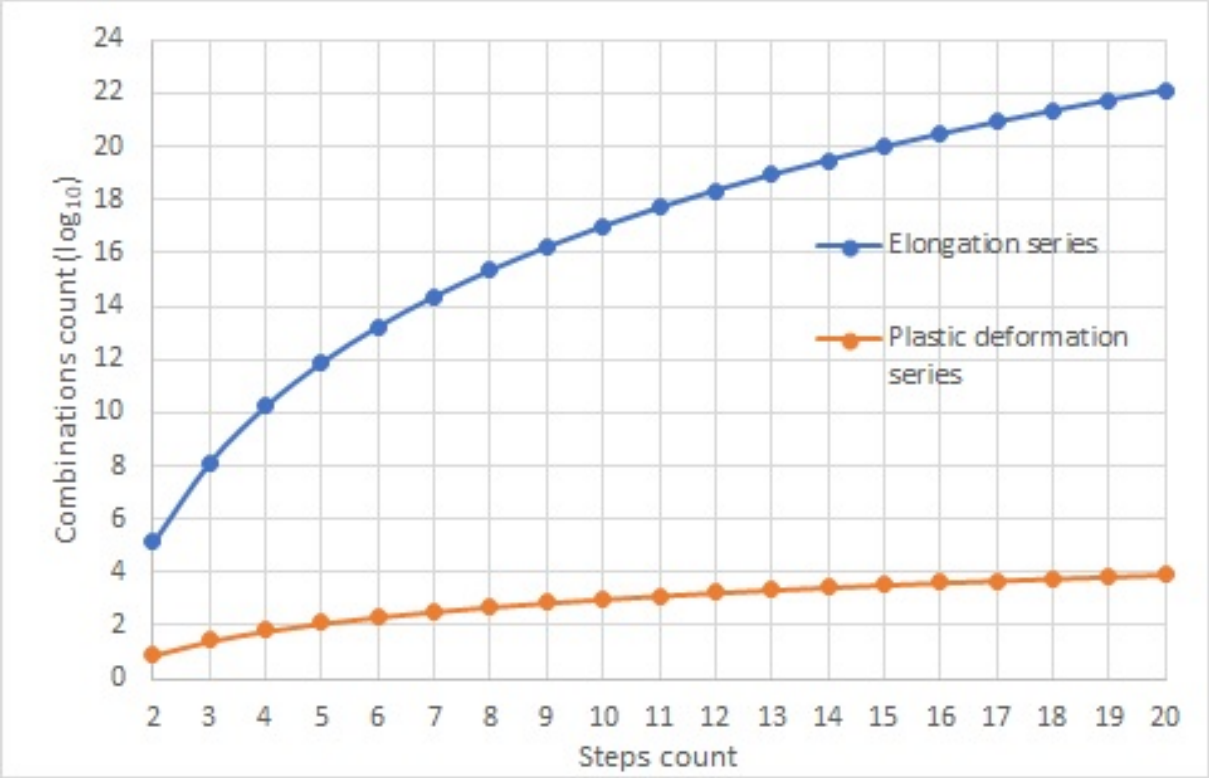


Figure 4.8: Combinations count for plastic deformation and elongation dataset based on steps count

Once the important features are found and based on that the number of steps for each feature are decided, then the knowledge-base is created by generation of each combination, prediction of corresponding label and then storing all these predictions in the database.

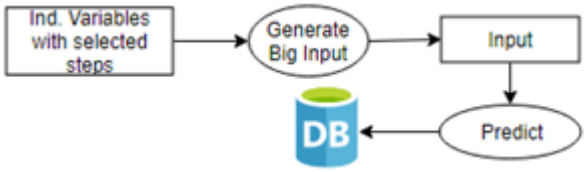


Figure 4.9: Knowledge generation phase

After completing knowledge generation step, the algorithm can be employed for monitoring and control: for each new set of input data observed during the casting, the quality characteristics (e.g. the hardness) are predicted as shown in Figure 4.10, and it is checked whether the predicted value lies within the optimal range. If this is not the case, the knowledge base is searched for the input values closest to the currently observed process

parameters (e.g. the current chemical composition) which, according to the prediction, will result in the dependent variable lying within the optimal range. Once such a set of input process parameters has been found, the changes are proposed accordingly.

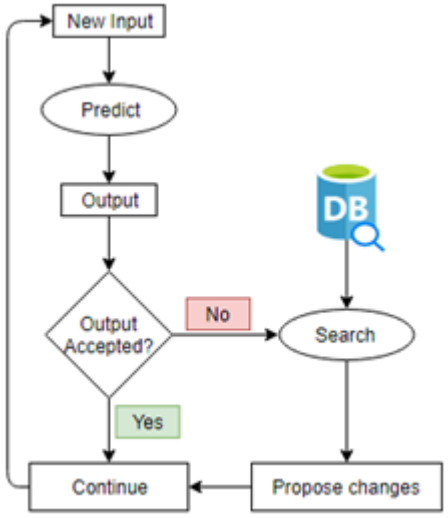


Figure 4.10: Continuous monitoring and keeping process stable

4.4 Evaluation and comparison

For the evaluation in Section 5, the process data is divided into learning and test parts using stratification. The model is learnt using k-fold cross validation which uses the complete learning data as both training and validation data. The trained function's performance is then tested on the test data. To give statistically significant results, the proposed framework was tested by following this procedure and computing the test errors 30 times, followed by computing the mean and standard deviation of the errors obtained. *Analysis of Variance* (ANOVA) and one-to-one statistical comparison tests were performed using the performance results.

5 Experiments and results

In order to evaluate our proposed framework, its performance is measured on several regression as well as classification datasets related to the foundry industry.

All datasets used in this work except the benchmark dataset *Plastic Deformation* are related to the foundry industry and have been obtained from companies which were part of the *EUREKA* research project *Intelligent Process Control in Foundry Manufacturing* (E!5092-IPRO) in cooperation with the University of Duisburg-Essen. During this project, a machine learning software *EIDOminer* was developed to support an optimised parameter selection for a production process through intelligent process data evaluation. The partner companies of the project manufacture safety and precision components for the automotive industry in machine molding casting. The market segments for which these components are made include hydraulics, vehicle technology, general engineering, drive technology and engine construction. For the manufactured foundry materials, some important mechanical properties are inspected before selecting them for an engineering application. These properties include, for example, strength, hardness, resilience, elasticity, plasticity, brittleness, ductility and malleability, which need to satisfy specifications according to customer requirements.

The dependent variables in the considered datasets are related to these quality characteristics, which have been obtained experimentally by the project partners. For example, to measure the elongation of a material as shown in Fig. 5.1, a strip or rod of the casting material with a certain length and a uniform cross-sectional area is fixed at one end. A tensile load is then applied along its axis. The load is increased incrementally and elongation is observed until the material mechanically fractures [76].

For the evaluation of our proposed method, the following mechanical properties are predicted based on different independent variables:

(Ultimate) Elongation is a measure of deformation that occurs before a material eventually breaks when subjected to a tensile load. The ultimate elongation of a material determines its possible use; for example, bumpers require high elongation values so that they can absorb energy by plastic deformation [76].

Tensile Strength is the amount of load or stress that a material can handle until it stretches or breaks down. The tensile strength of alloys is significantly influenced by its chemical composition.

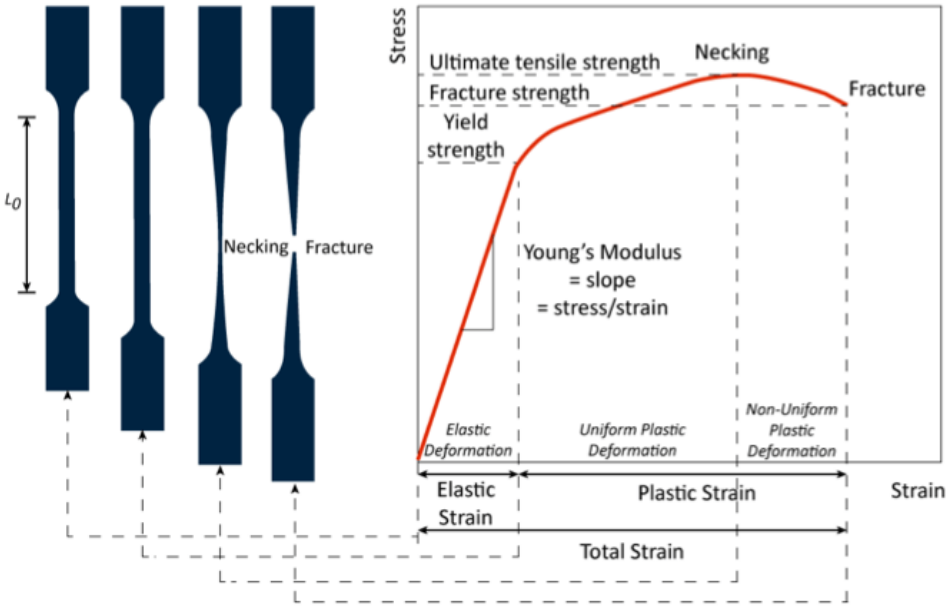


Figure 5.1: Tensile strength test to measure material elongation due to axial force [76]

Yield Strength is the amount of stress beyond which the deformation of a material is no longer elastic, but *plastic*, i.e. beyond which a body changes its shape permanently [76].

Compressive Strength is the ability of a material to withstand applied loads that reduce its size. To test the compressive strength of a material, a force is applied to the top and bottom of the body until it fractures or deforms [76].

Compressibility relates to the reduction in volume of sand bonded with clay and water after undergoing compression applied by squeezing or compaction. Ideally, its value should be in the range of 36-42 [133]. Compressibility is measured by applying three continuous ramming strokes on a 100 mm tube filler filled with a 100 g sand sample and later observing the difference using the compressibility scale [139].

Wet Tensile Strength is the "strength of the saturated condensation zone of a bentonite-bound mold material" [34].

Bulk weight represents the weight of foundry core sand and is measured in kg/m^3. The average value of bulk weight in foundries is around 1200 [131].

During the description of the results related to our experiments, for each variable of a given dataset, we provide the mean, standard deviation (std), minimum (min), first quartile (25%), second quartile (50%), third quartile (75%) and maximum (max) values.

All variables are then min-max normalized using a 0-1 scale, with the spread of the values and possible outliers shown in a boxplot. Note that if Q1 denotes the first quartile and Q3 is the third quartile, then the measurements which are outside 1.5 times the Inter Quartile Range (IQR) above the upper quartile (Q3 + $1.5 \times$ IQR) or below the lower quartile (Q1 - $1.5 \times$ IQR) are considered outliers. Boxplots are also used to show the results of the used machine learning methods. For each dataset, experiments are performed 30 times. For each experiment, the input data is randomly divided into two parts, with 90% of the input data used for learning by performing 5-fold cross validation on the selected machine learning methods to select optimal parameter values and 10% used for unbiased testing. The results of these methods for test data are shown in sections 5.1 and 5.2.

In the last part of section 5.1, performance results of individual learning methods are compared against each other using statistical methods.

5.1 Results for regression problems

The following **regression datasets** were used for the evaluation.

The **Flow stress** dataset is related to the heat flow curve of C15 steel: For plastic deformations, it is well known that the flow stress (kf) mainly depends on the temperature (T) of the material and the strain (ϕ) as well as the strain rate ($\dot{\phi}$) of the deformation. The dataset considered here contains 1248 measurements of these four quantities, which have been obtained experimentally in the controlled environment of the metallurgy and metal forming laboratory at the University of Duisburg-Essen. Although not directly related to the foundry industry, this dataset is used as a benchmark due to its high data quality: Since the dependent variable is well known to depend on the observed features (almost) deterministically, the data can be used to establish a baseline for different ML algorithms and demonstrate their suitability for "simple" applied regression problems.

The first foundry industry dataset investigated here describes the **Elongation** of the two materials EN-GJS- 350-22-LT and 400-18-LT. While the ultimate elongation for these materials should lie in the range of 15–22%, the exact value can differ depending on, for instance, the chemical composition of a specimen. The dataset used here therefore relates the content of 17 chemical elements in the material, which are used as the input features, to the measured elongation. In total, the dataset contains 386 of these measurements. Note, however, that the elongation of a specimen does not solely depend on its chemical composition, but also on additional features such as the nodularity or the content of ferrite and pearlite, which are not included in the dataset under consideration. In practice, however, the latter properties can only be measured *after* the casting has solidified and are therefore not available online in a production process. In situations where only the

basic chemical analysis is available, a prediction of the elongation is still possible (as will be shown in Section 5.1.2), although it needs to be expected that the results are not optimal and could be further improved by performing additional measurements during the production process.

Next, we consider a foundry industry dataset which indeed includes such additional information: here, the properties of **Yield Strength** (YS), **Tensile Strength** (TS) and **Elongation** (Elong) of the materials EN-GJS- 350-22-LT, 400-18-LT, 400-18, 400-15, 500-7, 600-3, 700-2 and 800-2 are given in relation to 25 input features, which include not only the chemical composition of the materials, but also the graphite content, the particle number, the particle density and the proportion of spheres as well as the pearlite and ferrite content and the nodularity index across 466 measurements.

The next dataset contains 180 measurements of, again, the **Yield Strength**, the **Elongation** and the **Tensile Strength** properties as the dependent variables of the materials EN-GJS- 350-22-LT, 400-18-LT, 400-18, 400-15, 500-7, 600-3, 700-2 and 800-2. However, similar to the first foundry dataset listed above, only the results of the chemical analysis – more specifically, the content of 11 elements in the material – are available as input features.

The final three regression datasets we investigate are related to the compressive strength, compressibility, wet tensile strength and bulk weight properties of *green molding sand*. The molding material has a decisive influence on the quality of a casting part: A large number of defect patterns on casting components are caused by properties of the molding material used, which in turn are determined by many influencing variables, including the quantity and quality of the materials as well as the process parameters, both for the preparation of the molding material and in the molding itself. In operational practice, some of these influencing variables can be regarded as constant; for example, the basic machinery and technology used for preparing the molding material do not change even over long periods of time. In order to characterise the condition of the molding material, some material parameters can be determined in the so-called *sand laboratory*. These properties are intended to describe the molding material in terms of its behaviour during molding and casting or after demolding. In practice, the properties of the molding material are mainly controlled by the content of water, bentonite and lustrous carbon formers as well as the particle size distribution of the silicon sand. Today, the basic qualitative influences of these parameters on the molding sand properties are well known from a large number of studies. There are also recommendations for suitable adjustments of the sand system, albeit with large tolerance ranges. Nevertheless, in a foundry environment, the quantitative dependency existing between all these parameters required for molding material control is not available in the literature [6]. For a foundry with a specific product range, it is therefore, on the one hand, still not a trivial task to determine ideal param-

eter settings and to control them in a targeted manner. On the other hand, within the scope of molding sand testing in the sand laboratory, a large number of parameters and properties are recorded over long periods of time, which implicitly contain the knowledge of their interrelationships. The application of ML methods to the control of molding sand processes is therefore seems highly appropriate.

5.1.1 Flow stress benchmark dataset

We begin with a simple dataset in order to test our basic ML algorithms and demonstrate their ability to predict quantities from input data in cases where the relationship between the features and the labels is well known. Here, the functionally dependent variable is the measured flow stress (Kf) during the plastic deformation of C15 steel, while the three functionally independent variables are the temperature (T), the strain (ϕ) and the strain rate ($\dot{\phi}$) during the plastic deformation.

The statistics of the dataset are shown in Table 5.1 and Fig. 5.2. The data is distributed uniformly and there are no outliers present in the dataset.

Table 5.1: Flow stress statistics

	$\dot{\boldsymbol{\phi}}$ [s^{-1}]	$\boldsymbol{\phi}$ [-]	**T** [°C]	**kf** [N/mm^2]
mean	32.902	0.372	593.942	394.442
std	41.162	0.196	384.406	227.454
min	0.100	0.040	20.000	43.725
25%	1.500	0.200	300.000	165.928
50%	1.500	0.375	600.000	412.845
75%	90.000	0.540	1000.000	605.893
max	100.000	0.700	1200.000	794.080

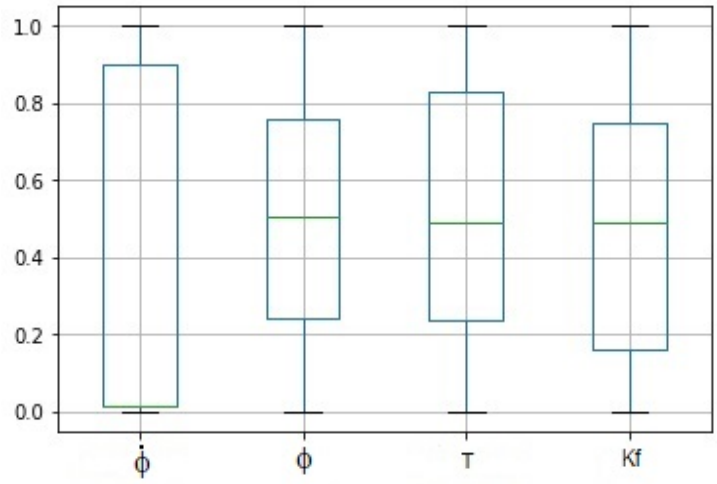


Figure 5.2: Boxplot for plastic deformation dataset normalized variables distribution

According to the correlation matrix shown in Fig. 5.3, there is a strong negative correlation between the temperature and the flow stress, whereas the strain and the force are positively correlated. These relations are, of course, to be expected from well-known physical principles.

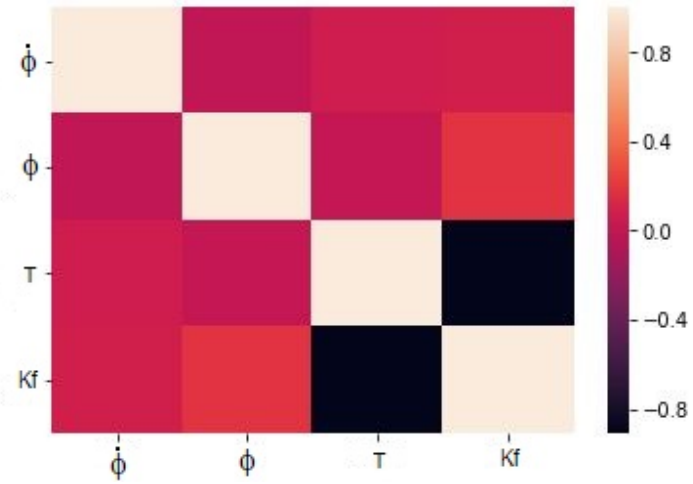


Figure 5.3: Correlation matrix for plastic deformation dataset variables distribution

Learning Phase Learning results for the Plastic Deformation dataset are shown in Fig. 5.4. During the learning phase, without and with preprocesssing methods, the individual learning algorithms and their combinations using the MPF method were applied.

It can be observed that ANFIS performed best among all the learning methods used. Among the other algorithms, KNN individually as well as combinations of KNN with other methods performed best.

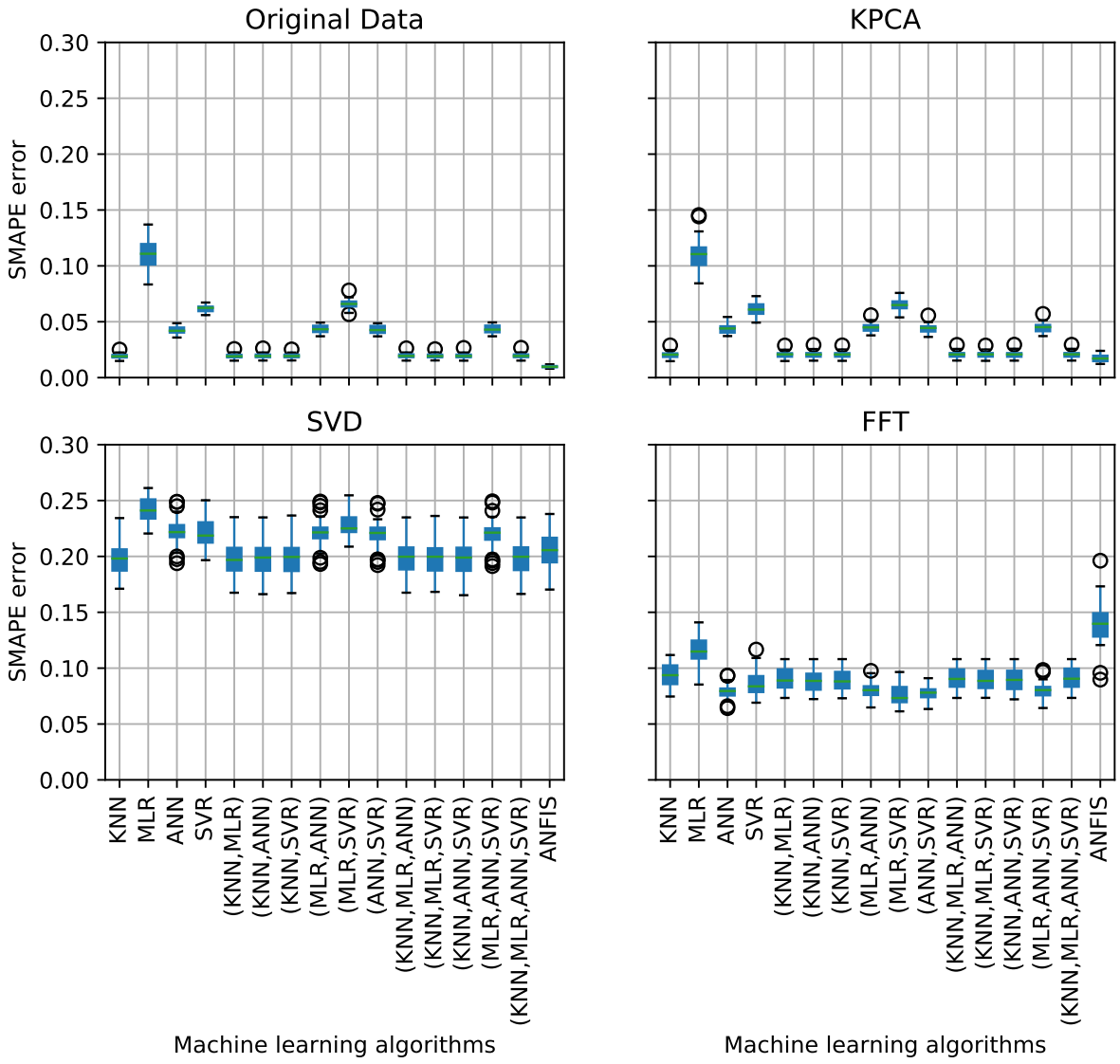


Figure 5.4: SMAPE error for plastic deformation test dataset

Among the eager learning methods, ANN performed best. Also, learning from unprocessed and KPCA-preprocessed data produced almost identical results, while the performance across all methods was significantly reduced by SVD-preprocessing.

It is interesting to note that the comparatively simple KNN algorithm seems to outperform

the more sophisticated methods not only in this case, but (as will be demonstrated later on) for many other datasets as well. However, there are some significant drawbacks to the KNN method: First, as is generally the case for "lazy learners", the prediction on very large datasets is much less performant for KNN compared to other methods considered here. Another important shortcoming, especially in view of the challenges faced by the foundry industry as outlined in Section 2, is the algorithm's susceptibility to redundant input features: Whereas other algorithms are much more robust under the addition of independent variables containing "pure noise", i.e. features unrelated to the dependent variable, the prediction quality exhibited by the KNN method deteriorates strongly in such cases. In order to demonstrates this effect, we added 2 and 5 unrelated variables to the dataset and re-trained all the algorithms. The results are shown in Fig. 5.5.

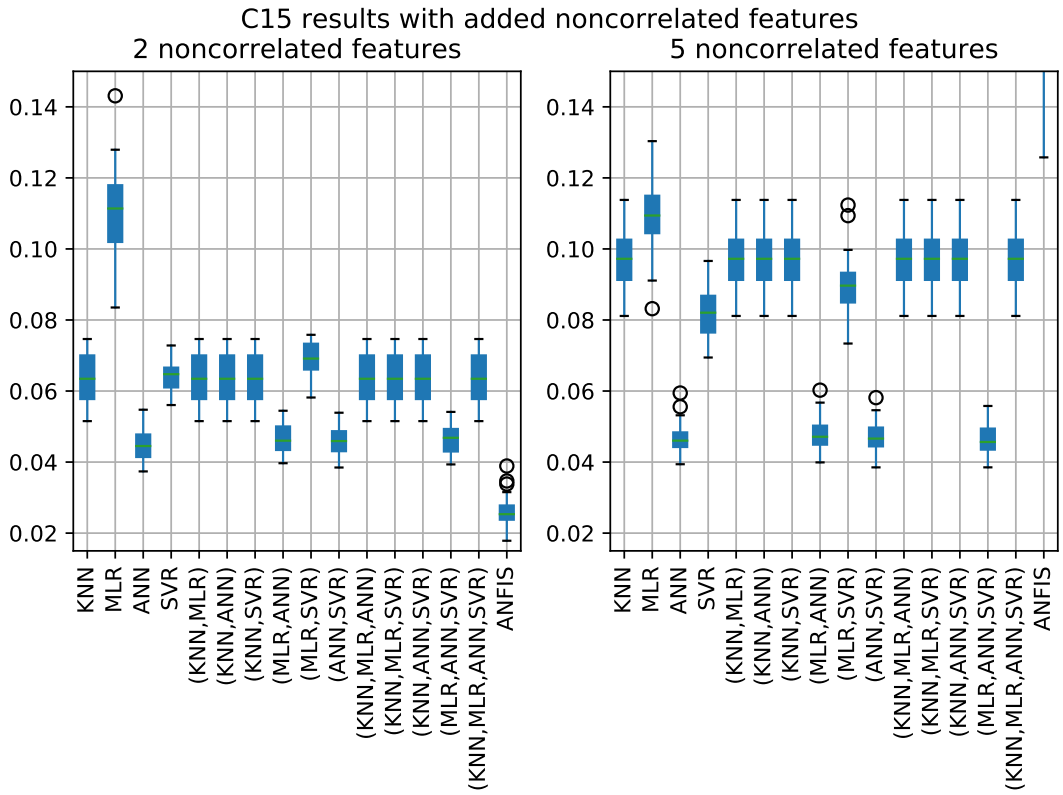


Figure 5.5: SMAPE error for plastic deformation test dataset with added features

Knowledge generation phase The initial data records count in the dataset is 1248. By selecting 100 steps for each of the three independent variables, an input set of size $100^3 = 10^6 = 1{,}000{,}000$ is generated. The corresponding output for each of these input value vectors is predicted using the best learning model and the complete generated

knowledge is stored in the database.

Monitoring and control phase The first step in this phase is to decide the optimal range for the functional dependent variable. Here, we assume for example that, for the process under consideration, the optimal range is as follows:

Optimal range: 500–600 [N/mm^2]

As described in Section 4, during the running process, the value of the input variables is continuously measured, and the corresponding expected output computed and compared with the optimal range. We further assume there is a new input occurring during the process and we predict the output using our composite learning method as follows:

Table 5.2: Output outside acceptable range for new input

$\dot{\boldsymbol{\phi}}$ [s^{-1}]	$\boldsymbol{\phi}$ [-]	**T** [°C]	**kf** [N/mm^2]
0.1	0.3	20	657.18

Since the expected output lies outside the optimal range, the system looks in the generated knowledgebase to find the inputs closest to the current ones that results in an output value within the optimal range. In Table 5.3 below, the closest inputs found are shown. One can observe that if only the amount of ϕ is reduced, the kf value decreases to within the acceptable range.

Table 5.3: Proposed changes

$\dot{\boldsymbol{\phi}}$ [s^{-1}]	$\boldsymbol{\phi}$ [-]	**T** [°C]	**kf** [N/mm^2]
0.1	0.2	20	599.2451
0.1	0.1933	20	593.4819
0.1	0.1867	20	587.4417
0.1	0.18	20	581.1079
0.1	0.1733	20	574.4604

5.1.2 Elongation of different materials

Next, we consider a dataset wich has been recorded directly from applications in the foundry industry: In the *Elongation* dataset, both the chemical composition and the elongation of multiple specimen from the materials EN-GJS- 350-22-LT and 400-18-LT were recorded. The functionally dependent variable is the elongation property (as a

measure of ductility), while the 17 independent variables represent the contents of different elements in the tested material: carbon (C), silicon (Si), manganese (Mn), phosphorus (P), sulphur (S), copper (Cu), magnesium (Mg), chromium (Cr), nickel (Ni), lead (Pb), aluminium (Al), molybdenum (Mo), titanium (Ti), tin (Sn), vanadium (V), zinc (Zn) and cerium (Ce).

Some basic statistics[6] of the Elongation dataset are shown in Table 5.4 and Fig. 5.6. We can observe that the measured data is not uniformly distributed and – for almost all features except C – there are many values which can be considered outliers. These values are either rarely measured or have not been recorded correctly.

Table 5.4: Elongation dataset statistics

	mean	**std**	**min**	**25%**	**50%**	**75%**	**max**
C [%]	3.524	0.114	3.190	3.440	3.530	3.610	3.880
Si [%]	2.799	0.107	2.350	2.740	2.800	2.850	3.990
Mn [%]	0.324	0.034	0.170	0.300	0.320	0.340	0.480
P [%]	0.013	0.002	0.010	0.012	0.013	0.013	0.021
S [%]	0.009	0.001	0.005	0.007	0.008	0.009	0.016
Cu [%]	0.059	0.018	0.030	0.050	0.060	0.070	0.150
Mg [%]	0.049	0.007	0.030	0.045	0.050	0.053	0.085
Cr [%]	0.045	0.010	0.025	0.039	0.044	0.050	0.103
Ni [%]	0.034	0.014	0.019	0.027	0.031	0.038	0.205
Pb [%]	0.193	0.111	0.001	0.097	0.193	0.289	0.385
Al [%]	0.008	0.001	0.005	0.007	0.008	0.009	0.014
Mo [%]	0.008	0.029	0.001	0.002	0.004	0.008	0.564
Ti [%]	0.007	0.002	0.004	0.006	0.007	0.008	0.017
Sn [%]	0.004	0.001	0.002	0.003	0.004	0.004	0.006
V [%]	0.004	0.001	0.002	0.003	0.004	0.004	0.015
Zn [%]	0.006	0.013	0.001	0.001	0.001	0.003	0.110
Ce [%]	0.005	0.002	0.001	0.004	0.005	0.007	0.011
Elong [%]	17.501	1.722	6.900	16.300	17.400	18.600	23.700

[6]Note that some of the values – particularly the lead content – are much higher than expected for the considered materials. While the data presented here was obtained and conveyed to us by a partner in the foundry industry, it is possible that some of the data has been rescaled and does not accurately represent the elemental content in percentage by mass. However, due to the z-score normalization used in our unified framework (cf. Section 4.1), such a rescaling would not influence any of the prediction results presented in the following.

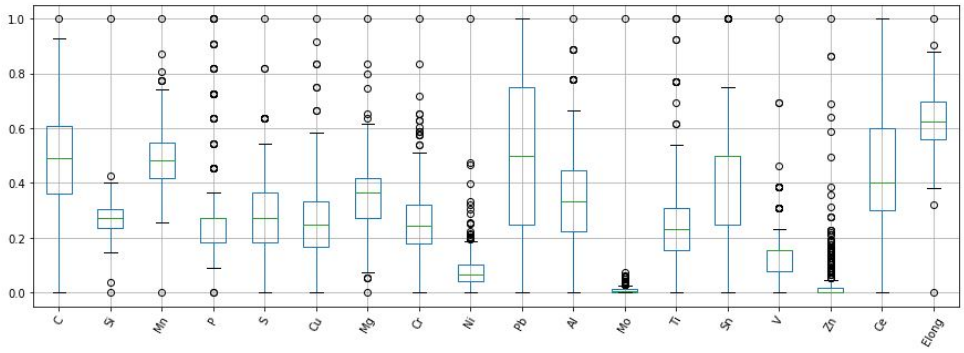


Figure 5.6: Boxplot of elongation dataset normalized variables distribution

As the correlation matrix shown in Fig. 5.7 shows, the elongation property of the metal has a strong negative correlation to the lead content (Pb), which is to be expected from basic metallurgical considerations.

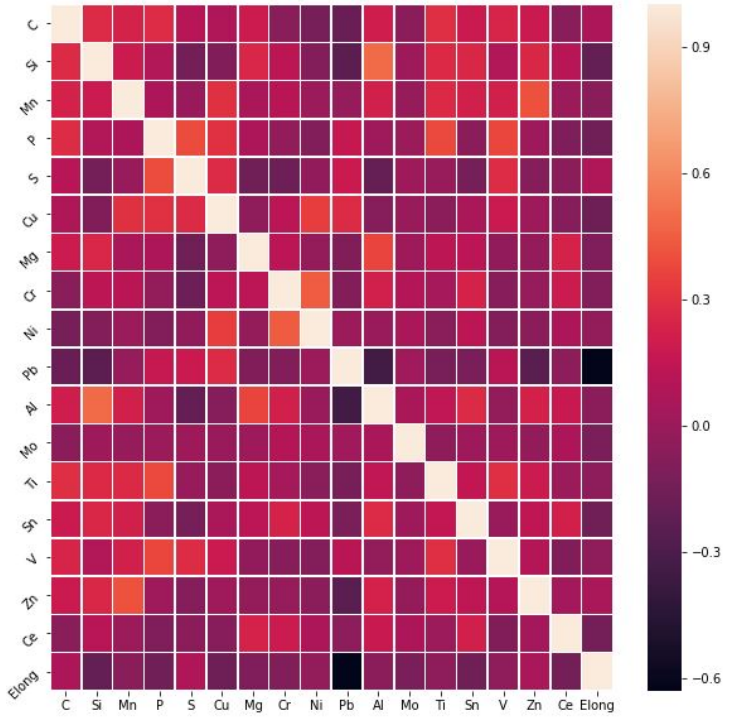


Figure 5.7: Correlation matrix for elongation dataset

The prediction results for the Elongation dataset are shown in Fig. 5.8. We can observe that MLR and SVR worked best among the base algorithms. Among the combination

methods, combinations of (KNN, SVR) and (MLR, SVR) performed best.

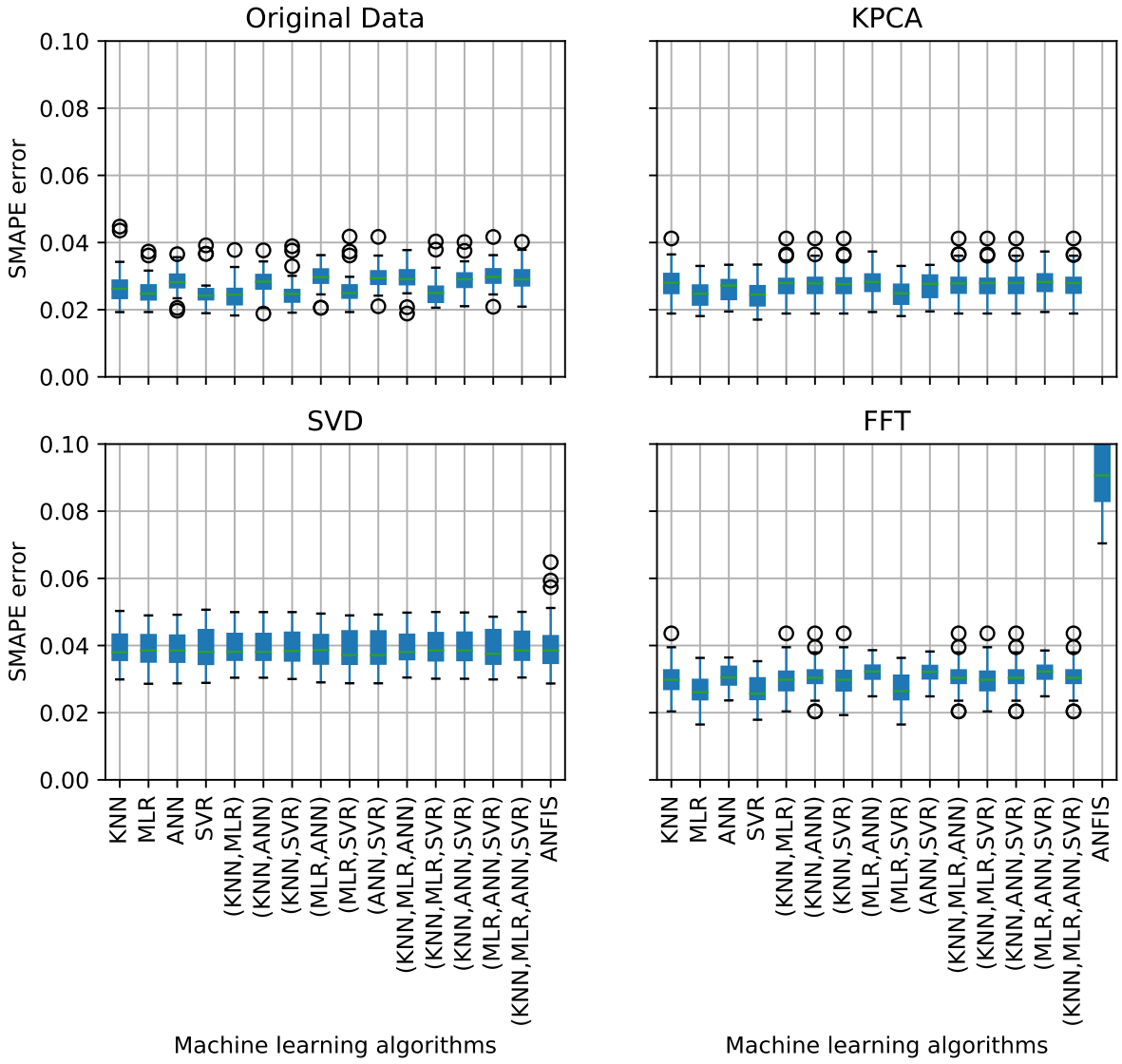


Figure 5.8: SMAPE error for elongation dataset

In this case, SVD preprocessing also reduced the learning performance for all the learning algorithms. With KPCA preprocessing, the performance of ANN improved and was comparable to SVM; otherwise there is not much difference noted for other algorithms. Applying FFT preprocessing reduces the variance of the ANFIS method.

5.1.3 Yield strength, tensile strength and elongation of different GJS alloys

The second foundry industry dataset concerns the *yield strength (YS)*, the *tensile strength* (TS) and (again) the *elongation* (El) of specimen from the materials EN-GJS- 350-22-LT, 400-18-LT, 400-18, 400-15, 500-7, 600-3, 700-2 and 800-2. Those three properties represent the dependent variables, whereas the functionally independent variables are the graphite content (Gr), particle number (PN), particle density (PD), proportion of spheres (FS), pearlite (Pr), ferrite (Fr) and nodularity index (FN) as well as the contents of carbon (C), silicon (Si), manganese (Mn), phosphorus (P), sulphur (S), copper (Cu), magnesium (Mg), chromium (Cr), nickel (Ni), lead (Pb), aluminium (Al), molybdenum (Mo), titanium (Ti), tin (Sn), vanadium (V), zinc (Zn), cerium (Ce) and scandium (Sc) in the specimen.

The statistics of the dataset are shown in Table 5.5 and Fig. 5.9.

Table 5.5: Statistics of the dataset

	mean	**std**	**min**	**25%**	**50%**	**75%**	**max**
Gr [%]	10.219	2.666	2.100	8.400	10.300	12.200	17.700
PN [%]	140.118	56.888	23.000	106.000	133.000	163.000	636.000
PD [%]	488.052	287.552	47.000	283.750	441.000	658.000	3382.000
FS [%]	79.561	10.411	39.800	74.300	81.700	87.100	98.800
Pr [%]	15.402	26.228	0.400	2.000	4.000	9.000	98.000
Fr [%]	84.598	26.228	2.000	91.000	96.000	98.000	99.600
FN [%]	85.152	8.233	51.700	81.100	86.800	91.100	99.400
C [%]	3.588	0.124	3.000	3.520	3.610	3.670	3.960
Si [%]	2.758	0.183	2.000	2.723	2.790	2.870	3.890
Mn [%]	0.336	0.104	0.000	0.260	0.310	0.370	0.630
P [%]	0.017	0.003	0.000	0.015	0.017	0.019	0.026
S [%]	0.009	0.002	0.000	0.008	0.009	0.010	0.015
Cu [%]	0.110	0.142	0.000	0.050	0.060	0.080	0.800
Mg [%]	0.045	0.006	0.000	0.042	0.045	0.048	0.070
Cr [%]	0.046	0.011	0.000	0.040	0.044	0.050	0.104
Ni [%]	0.032	0.012	0.000	0.026	0.030	0.035	0.191
Pb [%]	0.001	0.000	0.000	0.001	0.001	0.001	0.005
Al [%]	0.008	0.001	0.000	0.007	0.008	0.009	0.012
Mo [%]	0.010	0.041	0.000	0.005	0.007	0.009	0.894
Ti [%]	0.009	0.002	0.000	0.007	0.009	0.010	0.016
Sn [%]	0.004	0.001	0.000	0.003	0.004	0.004	0.006

V [%]	0.006	0.002	0.000	0.005	0.006	0.008	0.014
Zn [%]	0.003	0.005	0.000	0.001	0.001	0.001	0.057
Ce [%]	0.005	0.002	0.000	0.004	0.005	0.006	0.010
Sc [%]	1.064	0.050	0.921	1.034	1.077	1.099	1.249
YS [N/mm^2]	347.805	42.916	257.000	323.000	335.000	351.000	535.000
TS [N/mm^2]	517.762	104.648	411.000	464.000	478.000	498.750	911.000
El [%]	17.119	4.301	3.600	16.600	18.600	19.700	23.700

Fig. 5.9, in particular, shows that the measured data distribution for the independent variables *Fr*, *Fn*, *Cu*, *Mo*, *Zn* and for three all dependent variables of this dataset includes a large number of outliers.

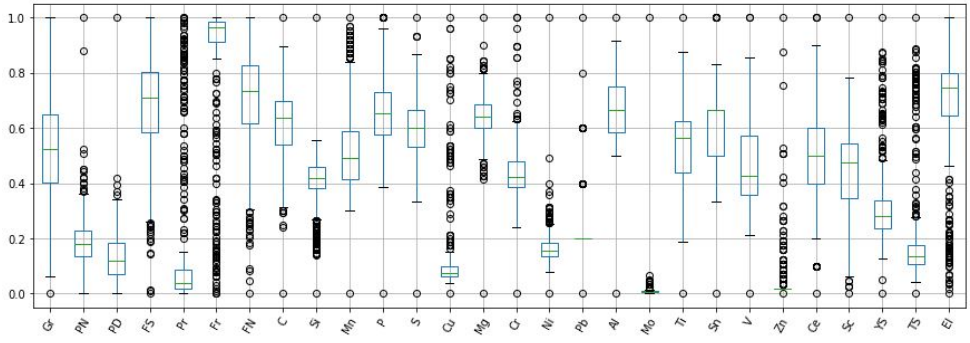


Figure 5.9: Boxplot of normalized variables distribution

The correlation matrix of the dataset is shown in Fig. 5.10. From the matrix, we can observe that *Fr*, *C*, *Si*, *S* and *Sc* are negatively correlated with *yield strength* and *tensile strength*, with a particularly strong correlation between the dependent variables and the Ferrite content (*FR*).

Prediction results for the *yield strength* property of the materials are shown in Fig. 5.11. We can observe that the best results were achieved when we used original data without preprocessing and with data preprocessed by KCPA. All the learning algorithms produced good results, with the exception of ANFIS. However, the ANFIS results improved with SVD preprocessing. The overall quality of the prediction demonstrates the high quality of the dataset.

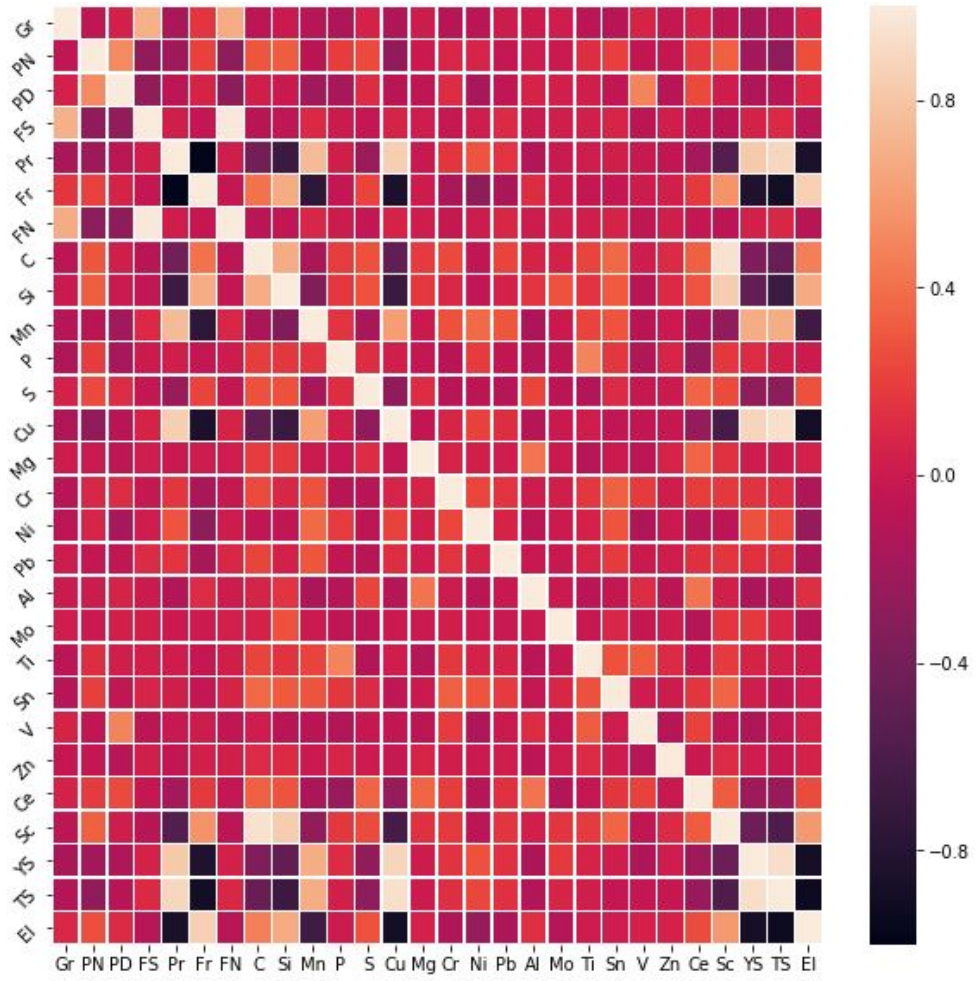


Figure 5.10: Correlation matrix of the dataset

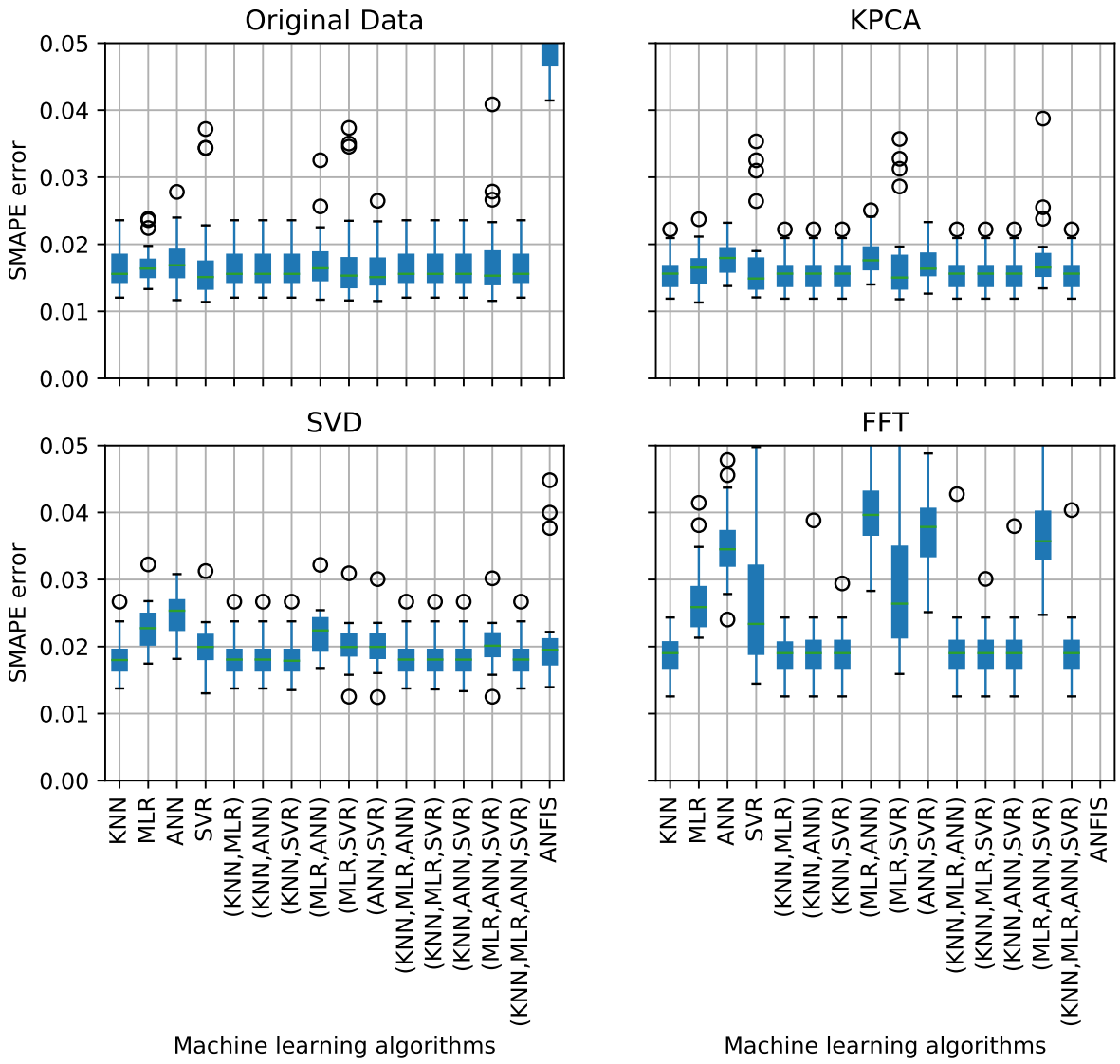


Figure 5.11: SMAPE errors for yield strength

The results for the *tensile strength* property of the materials are shown in Fig. 5.12. For this property, altogether results were even better than for the *yield strength*. Again, using no preprocessing provided the best results, although KPCA and FFT preprocessing showed good results as well. Among the base algorithms, KNN performed best, both as a stand-alone method and in combinations with other algorithms. The second best base method was SVM, whereas ANFIS produced the worst results on the original data, although its results improved and were comparable with other learning methods when applying SVD preprocessing.

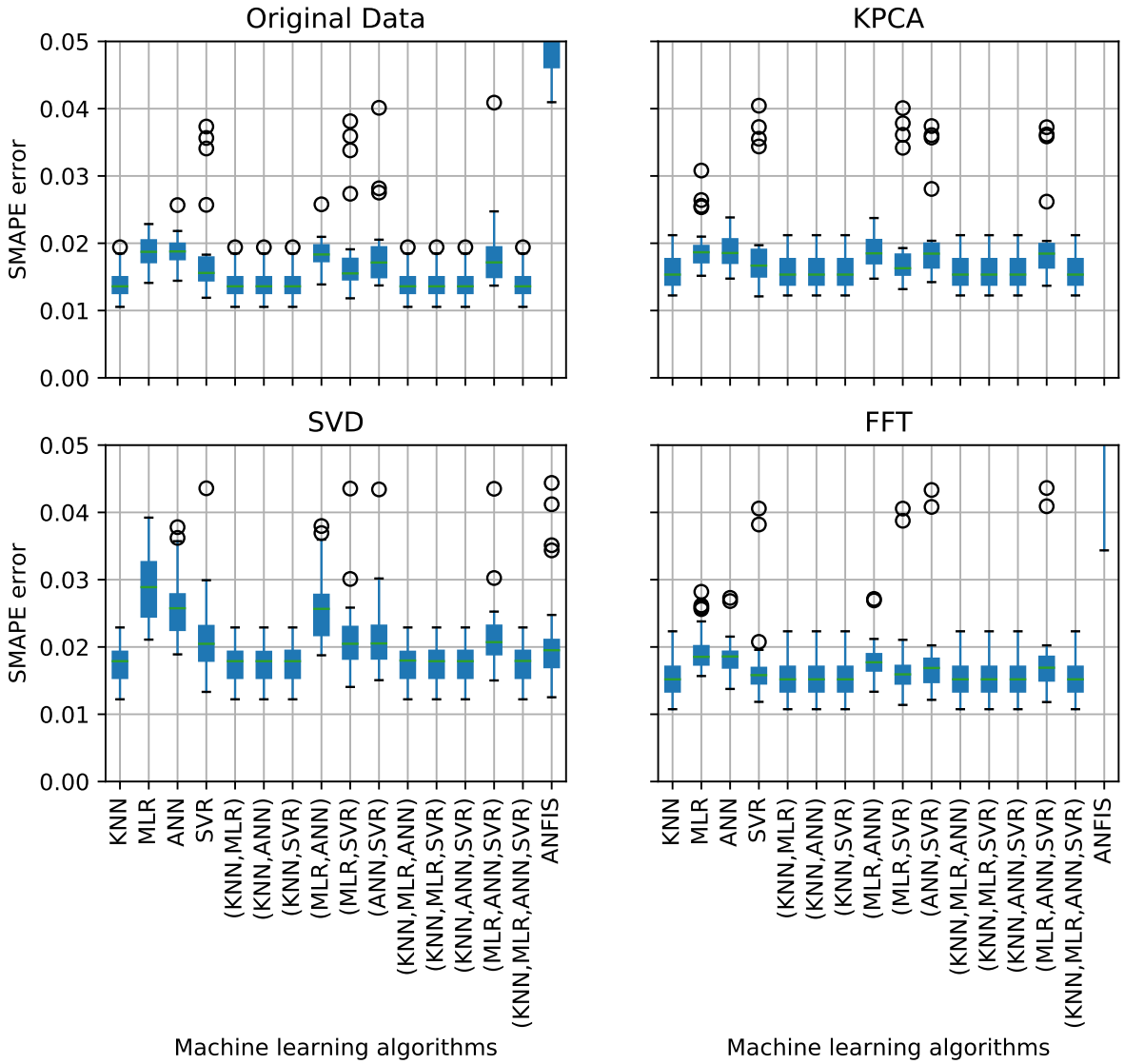


Figure 5.12: SMAPE errors for tensile strength

Finally, the results for the *elongation* property of the materials are shown in Fig. 5.13. Again, we can observe that prediction results are good overall, with considerably less accuracy by the ANFIS method unless SVD preprocessing is performed. Like for *tensile strength*, KNN and its combination method produced the best results. FFT preprocessing slightly improved results.

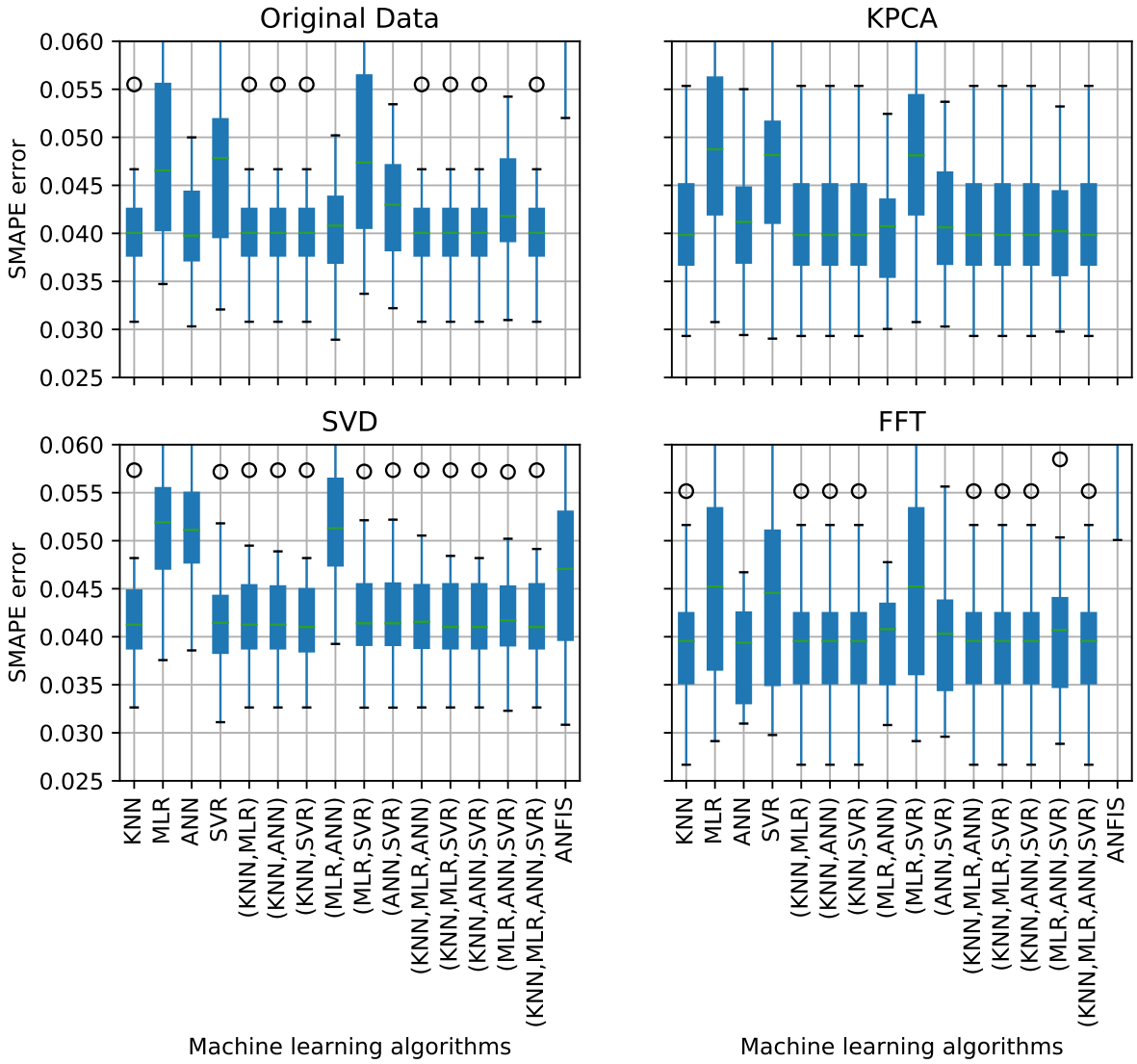


Figure 5.13: SMAPE errors for elongation

A comparison between the results for the elongation property and the results obtained in the previous section, we can observe that there is not much improvement in the prediction when we consider material properties in addition to the chemical composition; recall that using only the chemical composition has already produced very good prediction results.

5.1.4 Elongation, yield strength and tensile strength of different GJS alloys

We consider another dataset from the foundry industry, relating the functionally dependent variables **yield strength (YS)**, **elongation (Elong)** and **tensile strength (TS)** with functionally independent variables representing the carbon (C), silicon (Si), manganese (Mn), phosphorus (P), sulphur (S), magnesium (Mg), chromium (Cr), nickel (Ni), molybdenum (Mo), copper (Cu) and aluminium (Al) content. The statistics of the dataset are shown in Table 5.6 and Fig. 5.14.

Table 5.6: Statistics of elongation, yield strength and tensile strength properties of different materials

	mean	**std**	**min**	**25%**	**50%**	**75%**	**max**
C [%]	3.577	0.074	3.440	3.520	3.575	3.620	3.820
Si [%]	2.320	0.174	2.040	2.130	2.355	2.460	2.730
Mn [%]	0.140	0.056	0.064	0.102	0.111	0.174	0.284
P [%]	0.028	0.002	0.025	0.028	0.028	0.029	0.045
S [%]	0.009	0.002	0.005	0.008	0.009	0.010	0.014
MG [%]	0.046	0.005	0.036	0.042	0.045	0.049	0.065
Cr [%]	0.022	0.007	0.000	0.019	0.023	0.026	0.033
Ni [%]	0.013	0.022	0.000	0.009	0.012	0.014	0.297
Mo [%]	0.001	0.001	0.000	0.001	0.001	0.001	0.004
Cu [%]	0.173	0.222	0.000	0.053	0.074	0.113	0.845
Al [%]	0.018	0.006	0.000	0.017	0.018	0.020	0.041
Elong [%]	19.272	6.385	3.600	14.300	22.250	24.025	26.500
YS [N/mm^2]	313.744	52.707	258.400	280.875	296.450	343.500	475.800
TS [N/mm^2]	489.174	117.174	400.400	417.825	429.300	565.575	851.000

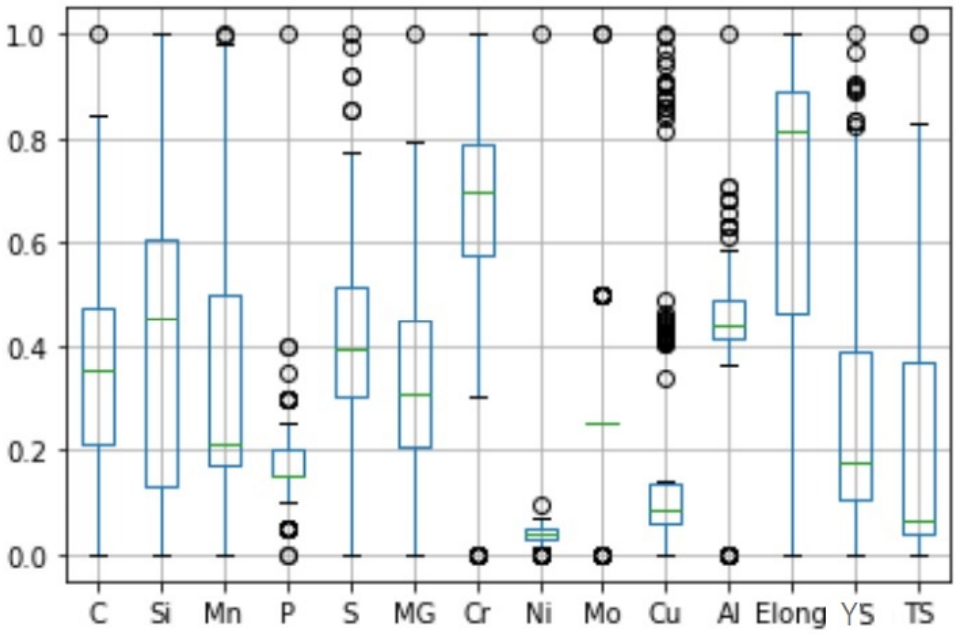


Figure 5.14: Boxplot of normalized variables distribution

This dataset is more uniformly distributed than the measured data in the previous section, with only few features (namely *P*, *S*, *Cu* and *Al*) exhibiting some outlier values. We can also observe from the correlation shown in Fig. 5.15 that the features *Mn* and *Cu* are strongly negatively correlated with the elongation and positively related with the labels *YS* and *TS*.

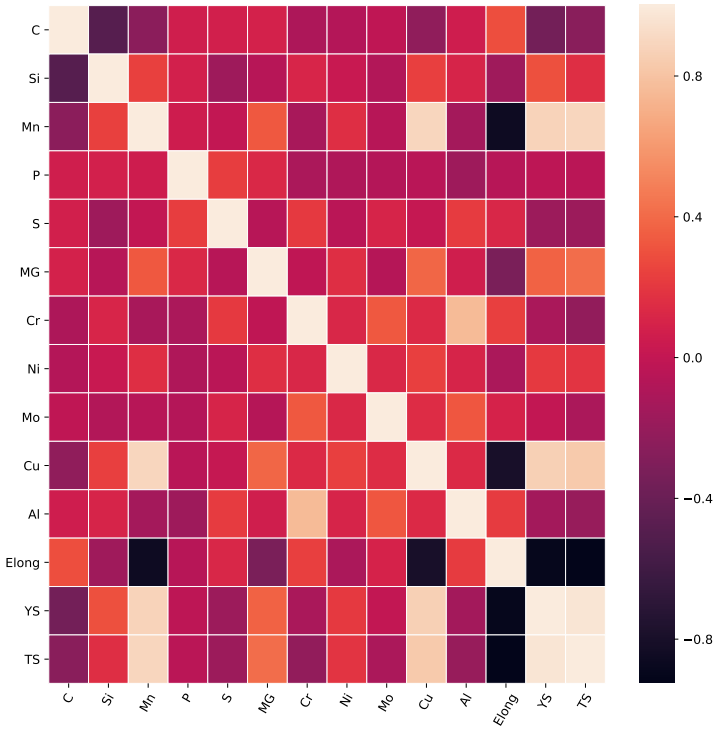


Figure 5.15: Correlation matrix of the dataset

The learning results for the *yield strength* property of the casting are shown in Fig. 5.16. The best performing base algorithms in this case are KNN and SVR. Among the combination methods, the methods involving KNN again performed best.

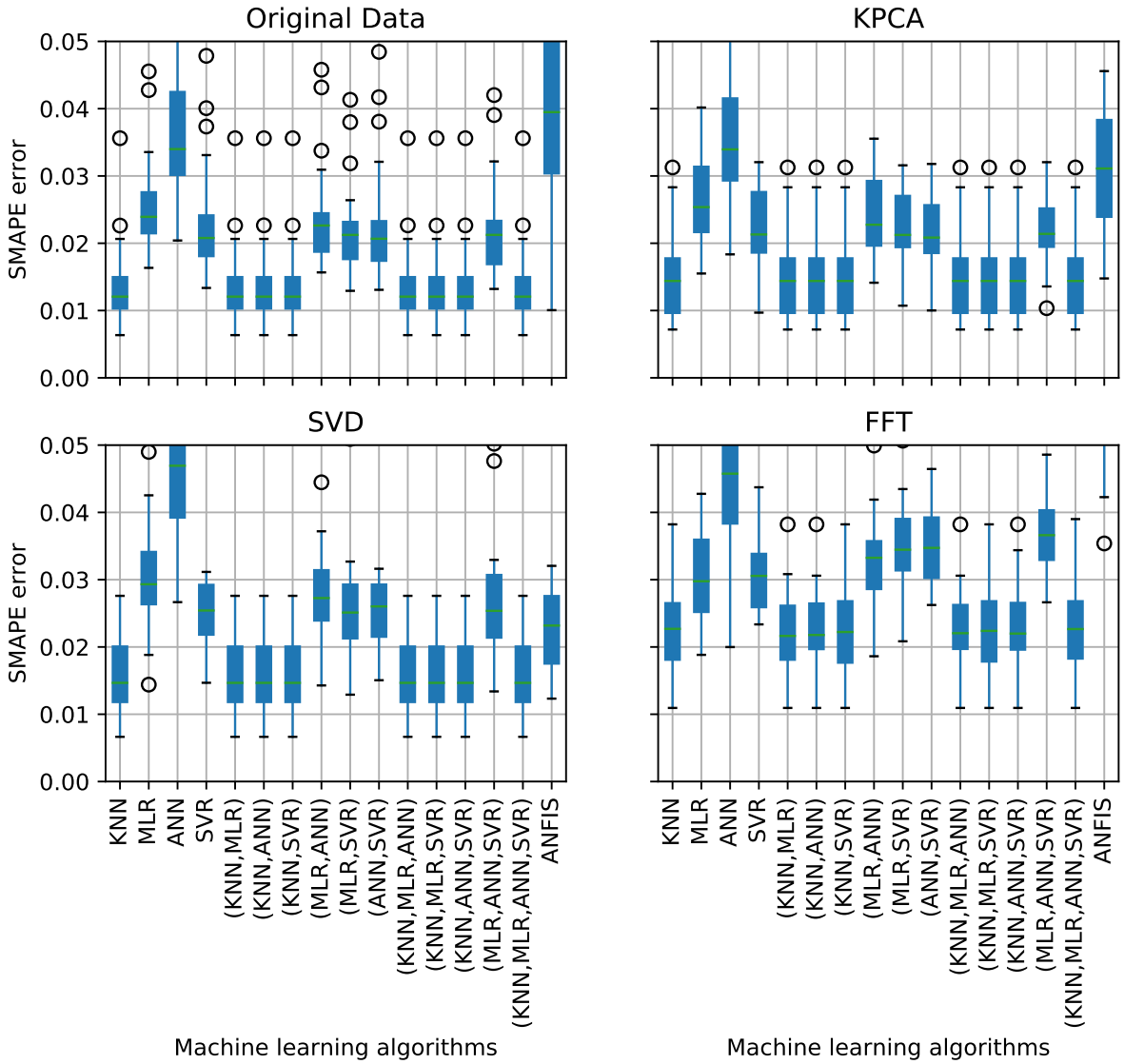


Figure 5.16: SMAPE errors for yield strength

Applying SVD preprocessing did not improve the results; in fact, the performance decreased for all the learning algorithms when SVD was applied. A similar decline in the prediction quality was observed when applying KPCA, although the results were better than with SVD overall. Using FFT preprocessing reduced the variance of the ANFIS predictions compared to other methods, but overall, the results were still worse than the basic predictions with no preprocessing at all.

The learning results for the *elongation* property of the materials are shown in Fig. 5.17. Here, the best performing base algorithms are KNN and ANN, whereas the best combi-

nation methods are the ones involving the KNN algorithm as well as the combination of MLR and ANN. Once again, applying SVD and KPCA preprocessing did not improve the results, although using SVD reduces the variance of error for most of the prediction methods.

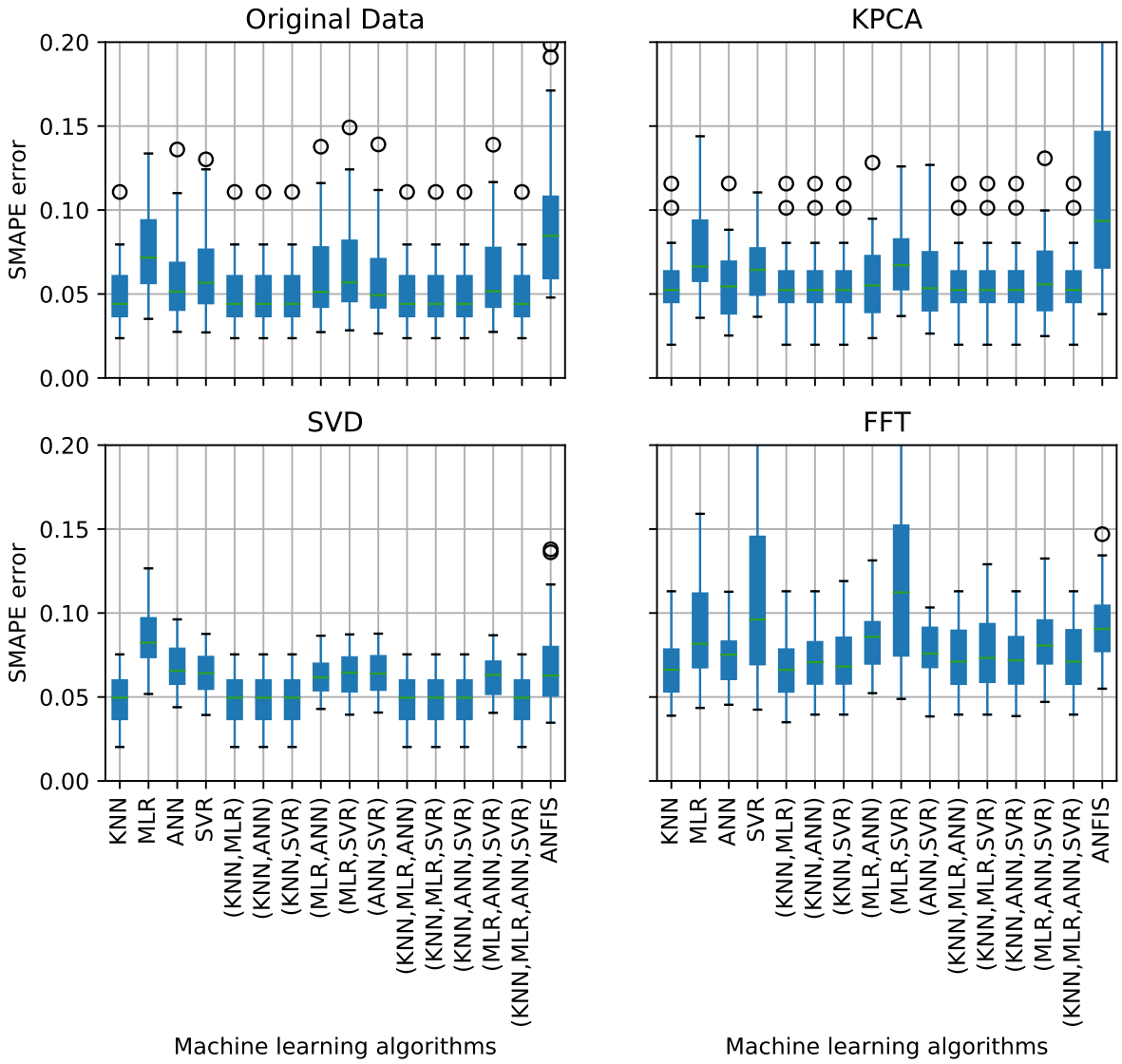


Figure 5.17: SMAPE errors for elongation

The learning results for the *tensile strength* property of the materials are shown in Fig. 5.18. The base algorithms KNN and SVR performed well again, whereas ANN led to a significantly higher prediction error in this case. Note that the ANN and ANFIS

prediction was improved by KPCA preprocessing, although the other base methods still performed better.

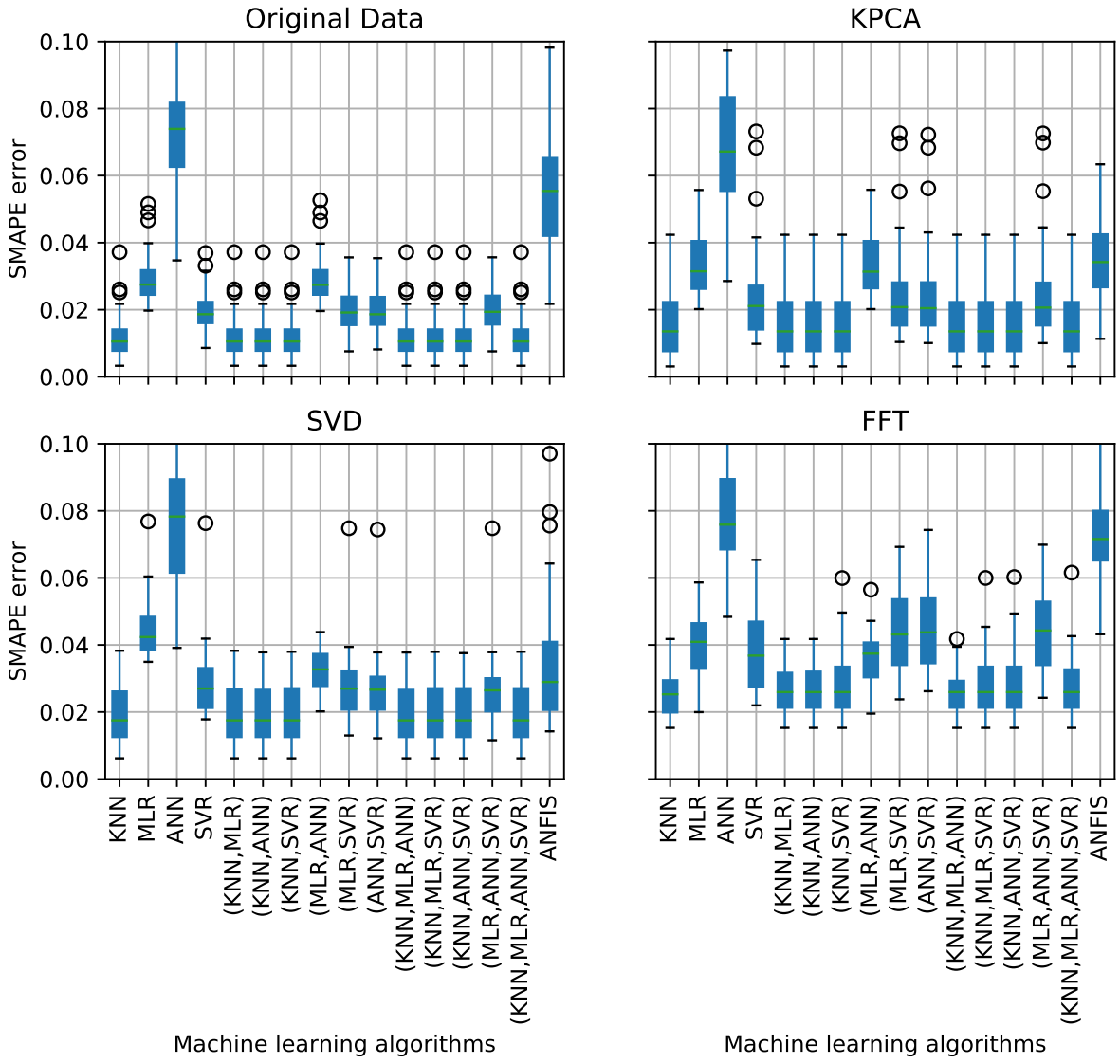


Figure 5.18: SMAPE errors for tensile strength

5.1.5 Compressive Strength and Compressibility of green molding sand

The last three regression datasets we consider are related to properties of *molding sand* used in metal casting. The first of these datasets consists of 1076 measurements and relates the **compressive strength** (CS) and the **compressibility** (Compr) of green

molding sand to its weight (W) and the content of water (H_2O), carbon (C), active clay (AcCley) and slack (Slk).

Some basic statistics of the dataset are shown in Table 5.7 and Fig. 5.19. From the correlation matrix shown in Fig. 5.20, we can observe that the independent features *Weight*, H_2O, *AcCley* and *Slk* and the label *CS* are positively correlated. Furthermore, the compressibility is correlated negatively with the weight and positively with the water content.

Table 5.7: Statistics of green molding sand dataset for compressiveStrength (CS), compressibility (Compr)

	Weight	**H_2O** [%]	**C** [%]	**AcCley** [%]	**Slk** [%]	**CS** [kN/m]	**Compr** [%]
mean	150.650	3.169	2.388	7.540	11.169	20.943	34.322
std	1.380	0.198	0.239	0.684	0.691	1.701	2.995
min	147.500	2.550	1.130	5.200	8.000	15.800	27.000
25%	150.000	3.040	2.230	7.100	10.720	19.800	32.000
50%	150.500	3.160	2.380	7.400	11.135	20.800	34.000
75%	151.500	3.300	2.540	7.900	11.610	22.000	36.000
max	155.000	3.850	3.100	11.680	14.400	27.500	48.000

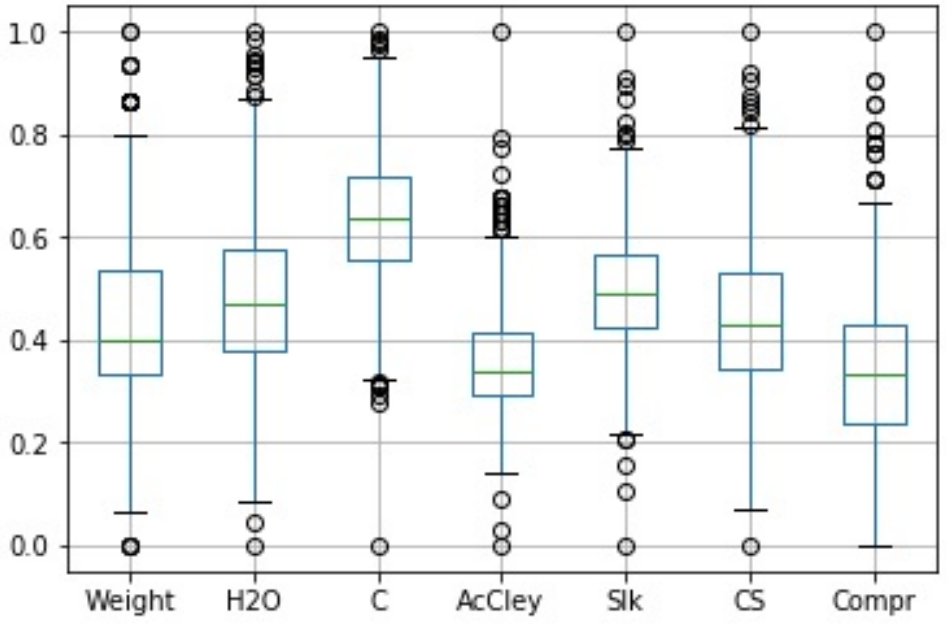


Figure 5.19: Boxplot of green molding sand dataset normalized variables distribution

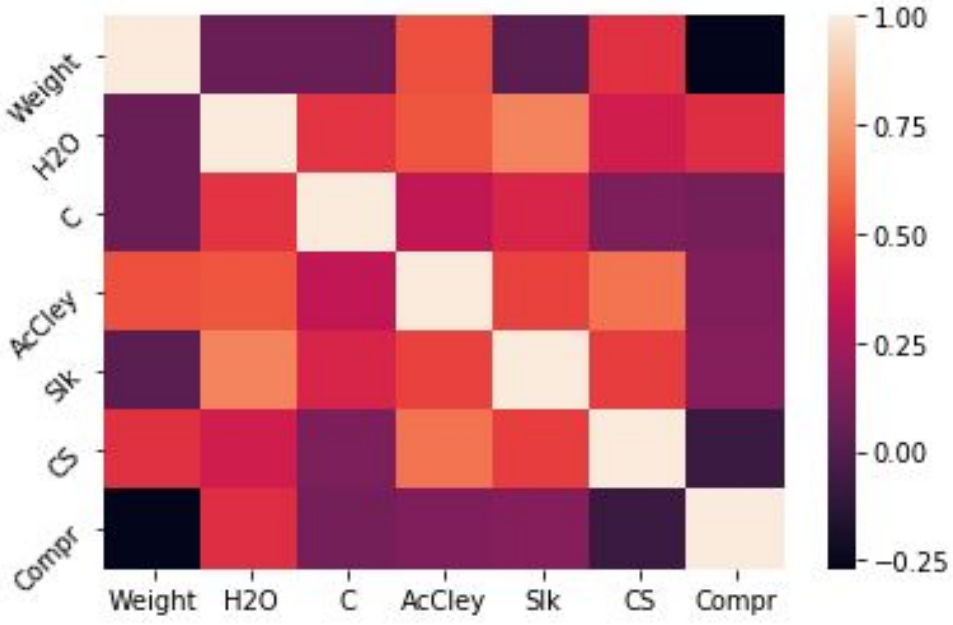


Figure 5.20: Correlation matrix for the green molding sand dataset

As with the plastic deformation dataset discussed in Section 5.1.1, we will describe not only the prediction results for the present dataset, but once more demonstrate how the unified framework described in Chapter 4 could be applied in an industrial setting in order to supervise the quality of molding sand with respect to its compressibility and compressive strength.

Learning phase The prediction results for the compressive strenth and the compressibility of the sand are shown in Figs. 5.21 and 5.22, respectively. In both cases, the SVR method and combination methods which include SVR show the best results. No significant improvement in the performance was observed after applying additional preprocessing methods.

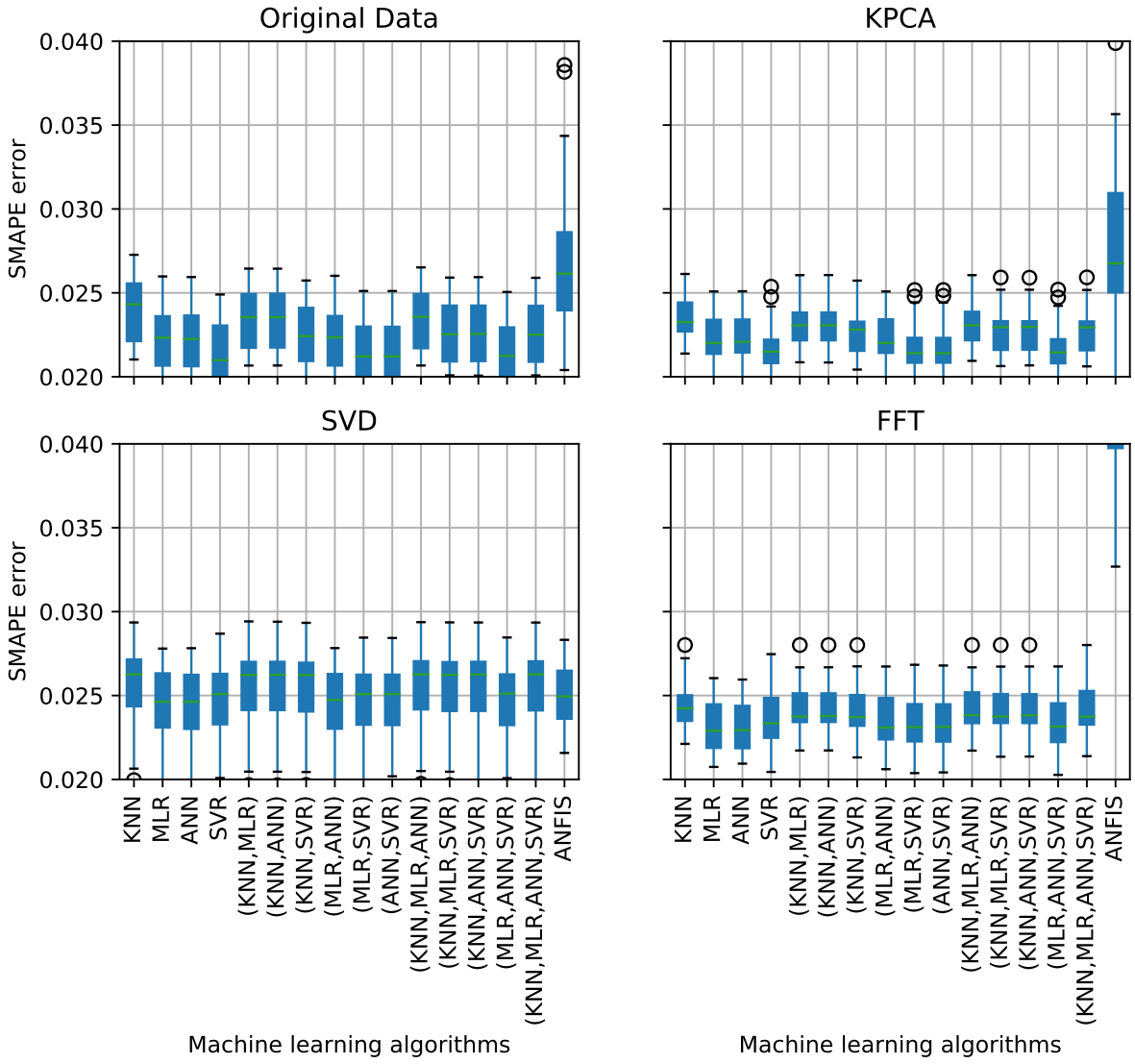


Figure 5.21: SMAPE Errors for compressive strength

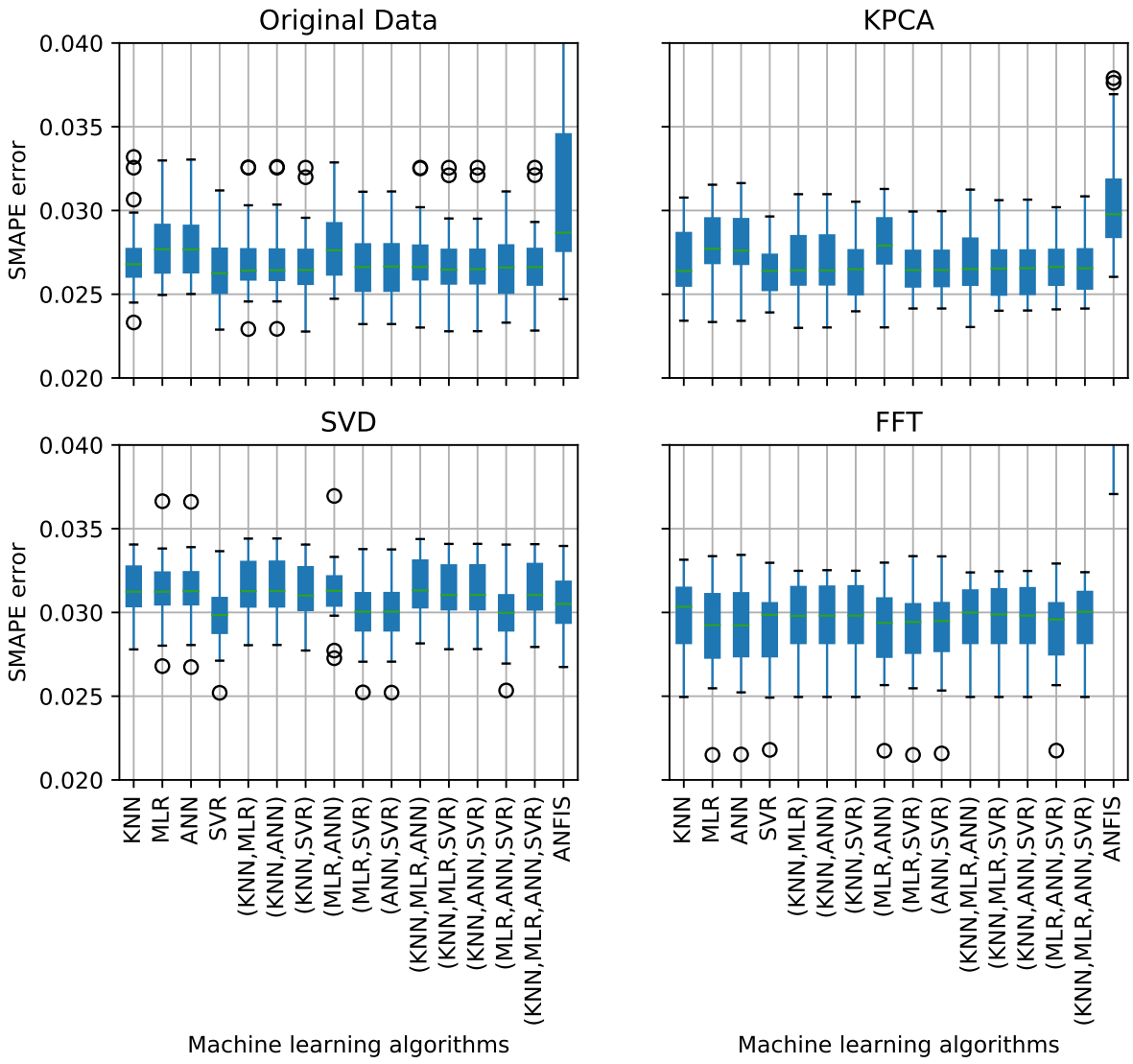


Figure 5.22: SMAPE errors for compressibility

Knowledge generation phase Selecting 20 steps for each of the five independent variables, an input size of $20^5 = 3{,}200{,}000$ is generated and the best selected learning model is applied to predict the corresponding output values. The entire generated dataset is stored in the database.

Monitoring and control phase For this process, we assume an optimal range of 24 – 26 kN/m for the compression strength.

Now, assume that during the continuous running of the process, the expected output of each input is computed (using the best performing algorithm again) and compared with

the optimal range. Assume further that there is a new input measurement during the process, and that we predict the output shown in Table 5.8.

Table 5.8: Output outside acceptable range for a new input

Weight	**Water** [%]	**Carbon** [%]	**Bentonite** [%]	**Clay** [%]	**Compression strength** [kN/m]
150.65	3.35	2.38	7.6	11.73	21.42

Since the expected output lies outside the optimal range, the system uses the knowledge-base to find the closest inputs to the current ones that, according to the prediction, still result in an optimal output value. In Table 5.9, the closest such inputs are listed. The user can now select one of the recommendations; for example, a decrease in the water and carbon content combined with an increase in bentonite and clay content is expected to improve the compression strength to above the selected threshold.

Table 5.9: Proposed changes

Weight	**Water** [%]	**Carbon** [%]	**Bentonite** [%]	**Clay** [%]	**Compression strength** [kN/m]
150.65	3.3026	2.063158	8.6105	12.0421	24.01943
150.65	3.2342	2.063158	8.6105	12.0421	24.00217
150.65	3.3710	1.959474	8.6105	12.0421	24.08666
150.65	3.3026	1.959474	8.6105	12.0421	24.12926
150.65	3.3026	1.855789	8.6105	11.7052	24.0067

5.1.6 Wet Tensile Strength of green molding sand dataset

The next dataset contains 183 measurements of the **wet tensile strength** (MTS) of green molding sand – which is considered the dependent feature – as well as the *grain size* (GrainSz), *water content* (H_2O), *carbon content* (C), *active clay content* (AcCley) and *slack content* (Slk) of the sand. The dataset's statistics are shown in Table 5.10 and Fig. 5.23; the measured data is evenly distributed for most of the variables and there are few outliers. From the correlation matrix in Fig. 5.24, we can observe that AcCley and the water content are positively correlated to the MTS.

Table 5.10: Statistics of wet tensile strength of green molding sand

	GrainSz [%]	**H_20** [%]	**C** [%]	**AcCley** [%]	**Slk** [%]	**MTS** [N/cm^2]
mean	0.270	3.153	2.377	7.533	11.162	0.333
std	0.005	0.184	0.215	0.636	0.643	0.019
min	0.254	2.730	1.750	5.800	9.790	0.299
25%	0.266	3.025	2.250	7.100	10.705	0.319
50%	0.270	3.160	2.340	7.400	11.130	0.332
75%	0.274	3.295	2.525	7.900	11.545	0.345
max	0.278	3.740	2.910	9.500	13.290	0.390

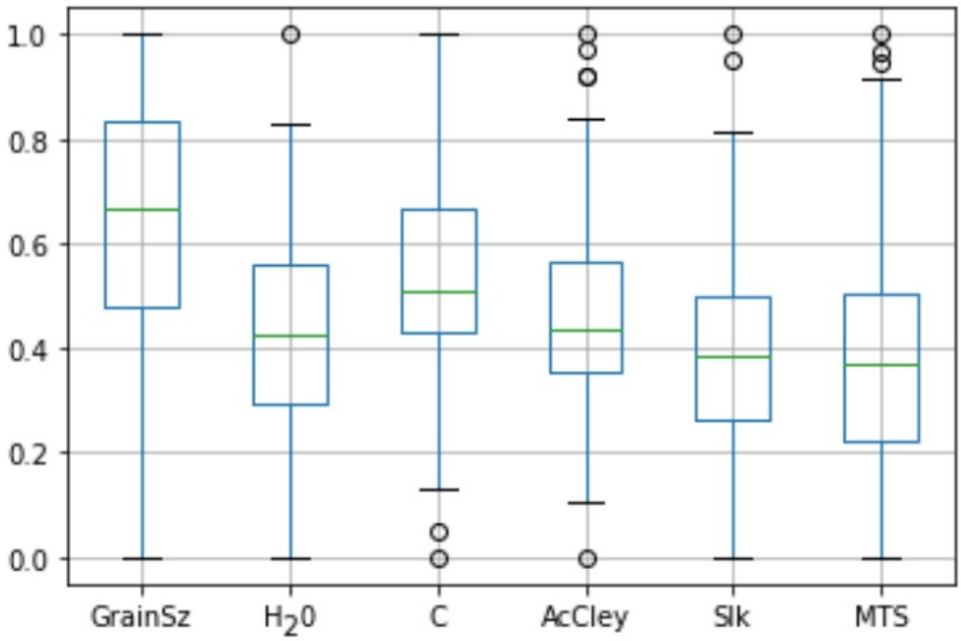


Figure 5.23: Boxplot of the green molding sand dataset normalized variables distribution

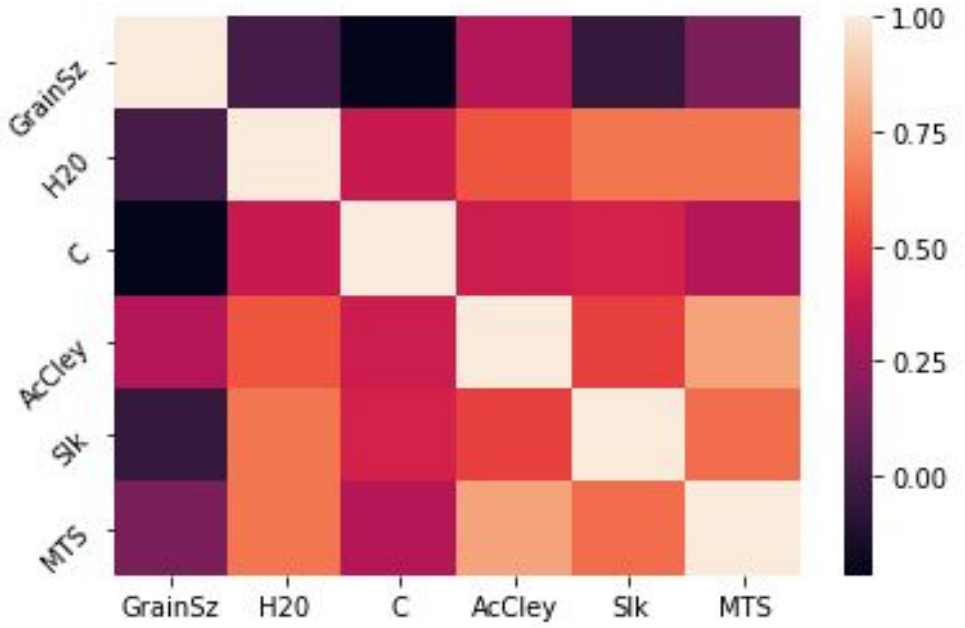


Figure 5.24: Correlation matrix for the green molding sand dataset

The prediction results for the tensile strength property of a material are shown in Fig. 5.25. Clearly, ANN and ANFIS are performing worst, while KNN and MLR are the best performing base algorithms. Again, the performance of the learning methods does not improve with the preprocessing methods.

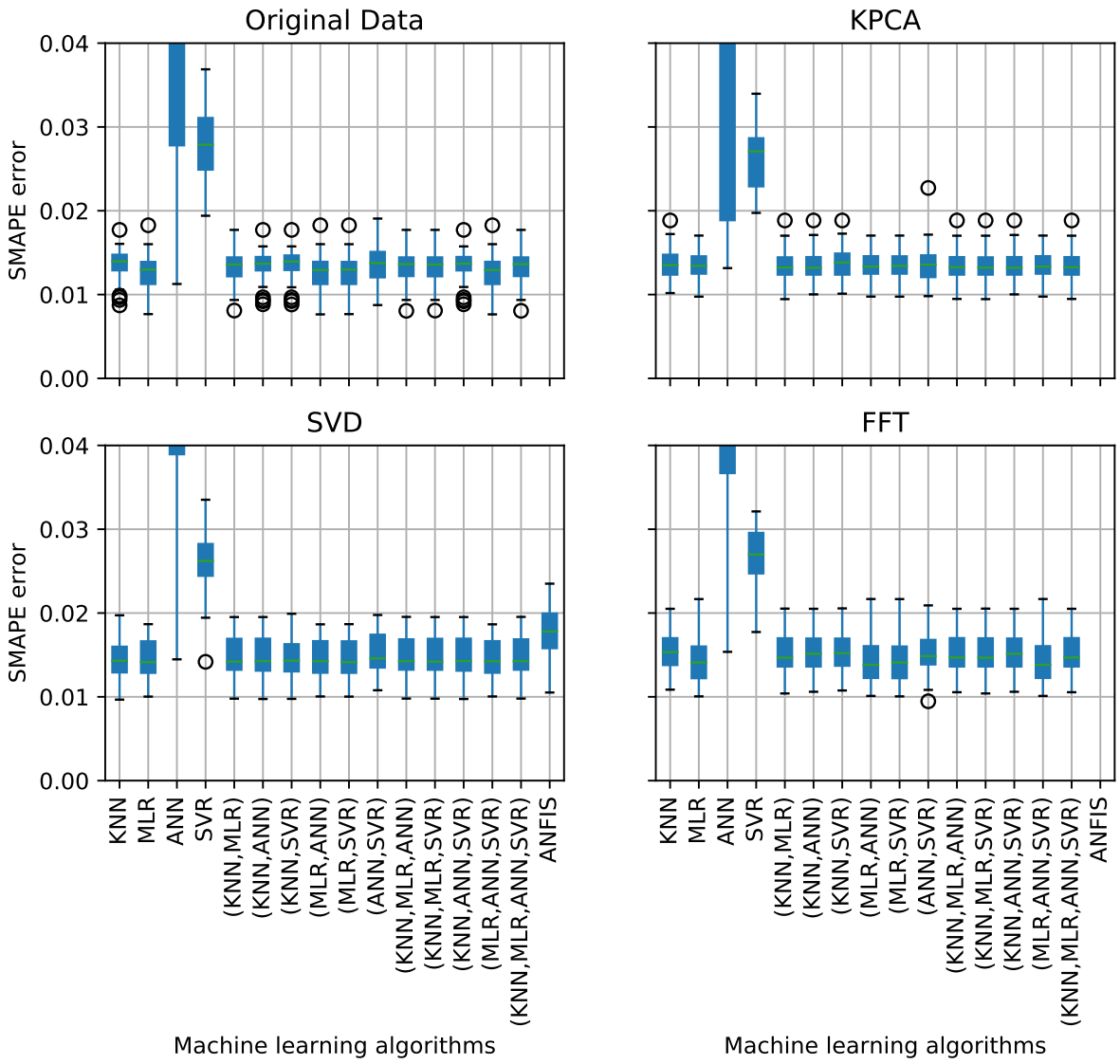


Figure 5.25: SMAPE errors for tensile strength

5.1.7 Bulk Weight of the green molding sand

The final regression dataset we consider consists of 1072 measurements of the **bulk weight** (BW) of green molding sand and the five independent variables weight (Weight), water (H_2O), carbon (C), active clay (AcCley) and slack (Slk) content. Basic properties of the dataset are shown in Table 5.11 and Fig. 5.26. The correlation matrix in Fig. 5.27 shows that weight and water content H_2O are positively and negatively correlated with the bulk weight, respectively.

Table 5.11: Statistics of bulk weight of the green molding sand

	Weight	**H_20** [%]	**C** [%]	**AcCley** [%]	**Slk** [%]	**BW** [kg/m^3]
mean	150.649	3.168	2.386	7.539	11.167	1023.748
std	1.376	0.198	0.238	0.682	0.691	49.573
min	147.500	2.550	1.130	5.200	8.000	815.000
25%	150.000	3.040	2.230	7.100	10.718	994.000
50%	150.500	3.160	2.380	7.400	11.130	1024.000
75%	151.500	3.300	2.540	7.900	11.602	1056.000
max	155.000	3.850	3.100	11.680	14.400	1223.000

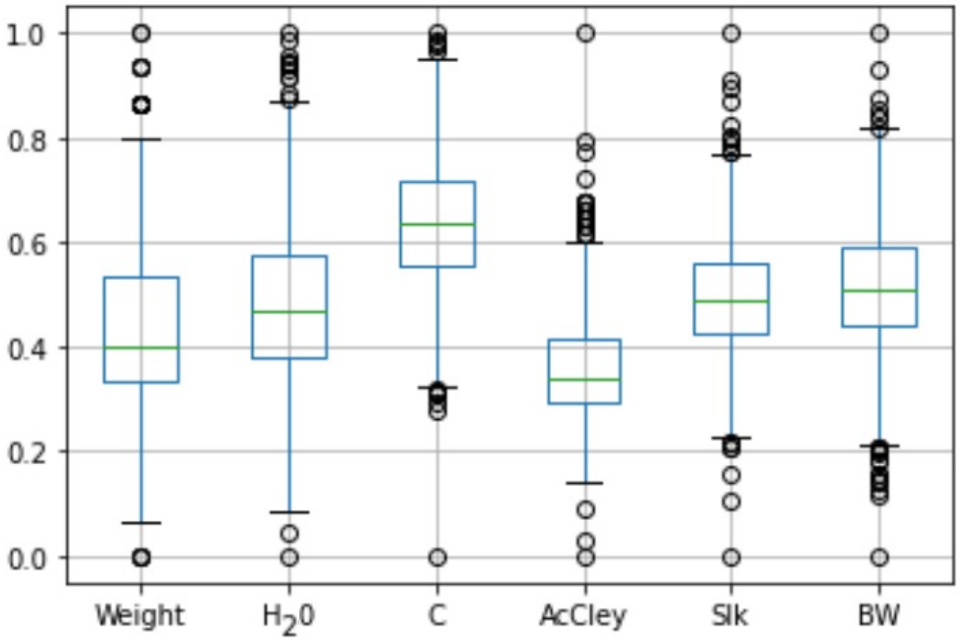


Figure 5.26: Boxplot of green molding sand dataset normalized variables distribution

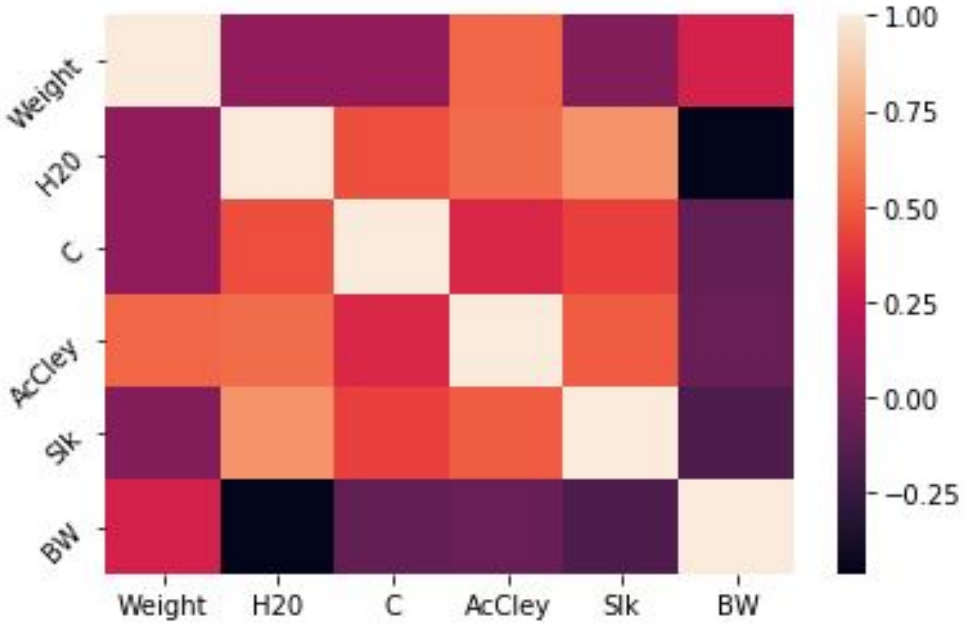


Figure 5.27: Heatmap of the correlation matrix for green molding sand dataset

The prediction results for the bulk weight from the independent variables are shown in Fig. 5.28. In this case, the best performing algorithms for this dataset are MLR, ANN and SVR as well as the combinations (MLR, ANN), (MLR, SVR) and (ANN, SVR).

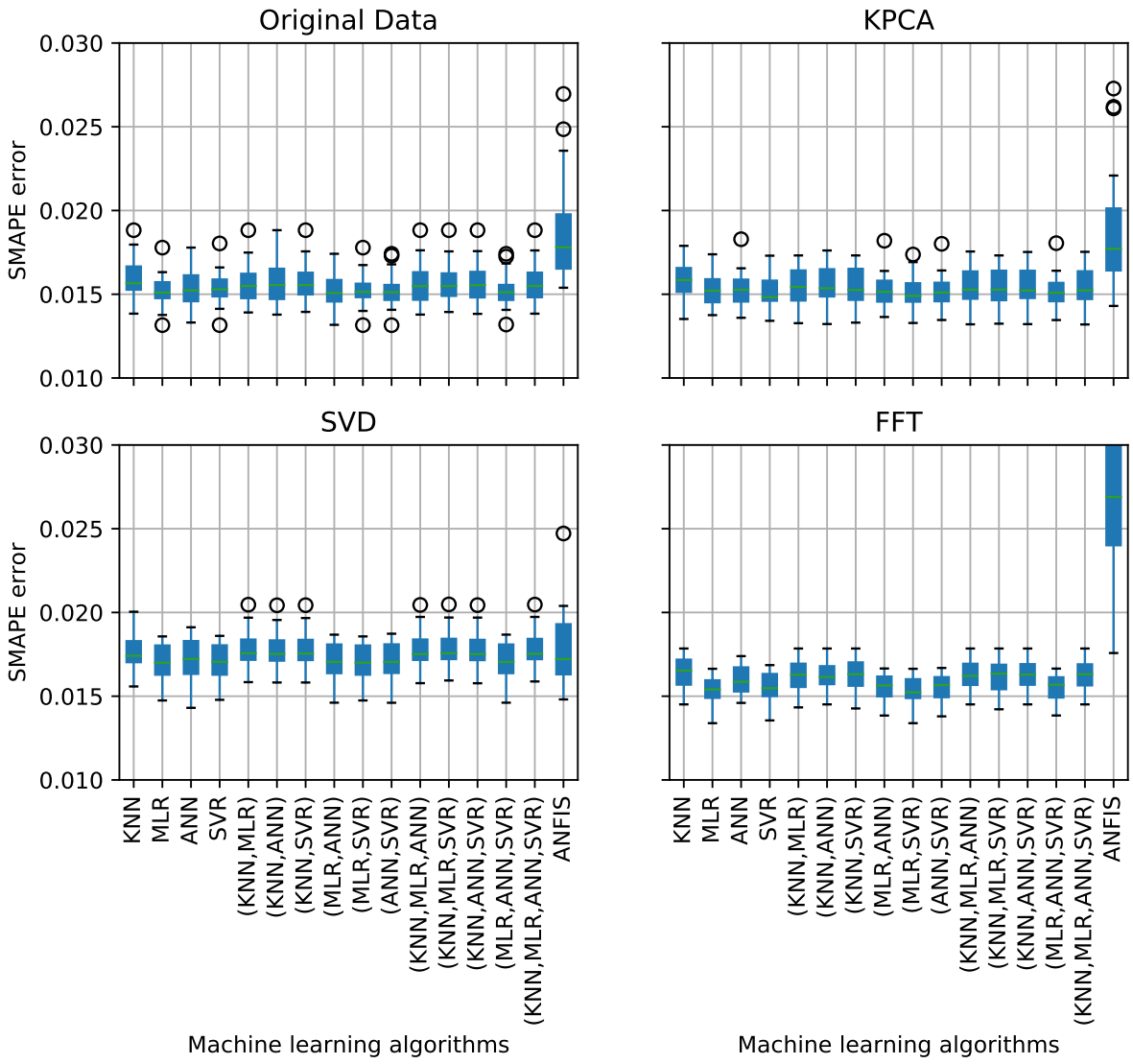


Figure 5.28: SMAPE errors for bulk weight

5.1.8 Comparing performance of the used learning methods, datasets and their interaction

To compare the performance of all the considered learning algorithms on all the datasets described above – and to determine if there is any interdependence between them – the two-way *Analysis of Variance* (ANOVA) method is applied, with the results shown in Table 5.12. With p-values of zero up to four decimal places, it clearly follows from these results that the ML algorithms do not perform equally well and that there is a significant

difference in prediction quality across datasets.

Most interestingly, the p-value for the *interaction* between the ML algorithm and the dataset is close to zero as well. Therefore, the results show (with a very high significance) that the performance of specific ML methods and the dataset under consideration are *interdependent*; in other words, whether an ML algorithm performs well can differ from dataset to dataset.

Source	**SS**	**df**	**MS**	**F**	**Prob>F**
ML Algorithms	0.6400	15	0.0427	319.58	0
Datasets	0.8261	11	0.0751	562.52	0
Interaction	1.2242	165	0.00742	55.58	0
Error	0.7433	5568	0.00013		
Total	3.4336	5759			

Table 5.12: Two-way ANOVA results

In order to obtain further information about the performance of the different ML methods, we use the Tukey, Bonnferroni and Scheffe tests to identify pairs of algorithms with differences in performance. The results of this pairwise comparison are shown in Table 5.13. For example, based on all three considered pairwise comparison tests, the performance of KNN significantly differs from the performance of MLR, ANN, SVR, (MLR,ANN), (MLR,SVR), (ANN,SVR), (MLR, ANN, SVR) and ANFIS across datasets.

Table 5.13: One-One comparison of learning methods using Tukey, Bonnferroni and Scheffe's tests

	One-One Comparison Methods		
Learning Method	**Tukey**	**Bonnferroni**	**Scheffe**
1 - KNN	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-10,14,16
2 - MLR	1,3-16	1,3-16	1,3-16
3 - ANN	1,2,4-16	1,2,4-16	1,2,4-16
4 - SVR	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16
5 - (KNN,MLR)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-10,14,16
6 - (KNN,ANN)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-9,14,16
7 - (KNN,SVR)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-10,14,16
8 - (MLR,ANN)	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16
9 - (MLR,SVR)	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16
10 - (ANN,SVR)	1-3,5-7,12-13,15,16	1-3,5-7,12-13,15,16	1-3,5,7,12-13,16
11 - (KNN,MLR,ANN)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-9,14,16
12 - (KNN,MLR,SVR)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-10,14,16
13 - (KNN,ANN,SVR)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-10,14,16
14 - (MLR,ANN,SVR)	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16	1-3,5-7,11-13,15,16
15 - (KNN,MLR,ANN,SVR)	2-4,8-10,14,16	2-4,8-10,14,16	2-4,8-9,14,16
16 - ANFIS	1-15	1-15	1-15

5.2 Results for classification problems

In addition to the regression problems discussed above, we also consider two *classification* datasets. For these datasets, the machine learning methods LR, SVC and GBDT are used as the base learners. Their results are then combined using the *Combination Method Naive Bayes* (CMNB).

5.2.1 Elongation classification dataset

The first classification dataset contains the chemical composition of a material as the independent features; more specifically, the contents of Carbon (C), Silicon (Si), Manganese (Mn), Phosphorus(P), Sulfur(S), Copper(Cu), Magnesium (M), Gallium (G), Chromium (Cr), Nickel (Ni), Lead (Pb), Aluminium (Al), Molybdenum (Mo), Titanium (Ti), Tin (Sn), Vanadium (V), Zinc (Zn) and Cerium (Ce) are given. The two-class label (ElongOK) represents the acceptability of the *elongation* according to requirements on the mechanical properties of the material.

The results of each of the base learners as well as the final results from the CMNB are shown in Table 5.14. From these results, it can be observed that all the methods including the combined one are working very well on this dataset and that the obtained prediction quality is indeed very high.

Table 5.14: Elongation: Precision and recall of base learners and naive bayes for raw data

	LR			**SVC**			**GBDT**			**CMNB**			
	precision	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**support**
0	**1.00**	**0.91**	**0.95**	**1.00**	**0.91**	**0.95**	0.98	0.91	0.95	0.98	0.91	0.95	57
1	**0.89**	**1.00**	**0.94**	**0.89**	**1.00**	**0.94**	0.89	0.97	0.93	0.89	0.97	0.93	40

Table 5.15: Elongation - Confusion matrix for naive Bayes method

		Predicted	
		0	**1**
Measured	**0**	52	5
	1	1	39

5.2.2 Internal mircoshrinkages in the production of callipers and anchors

Our final dataset, which contains 936 measurements in total, is related to internal *microshrinkages* that occur during the production of callipers and anchors for wind turbines. In order to prevent such microshrinkages it is necessary to predict their occurence reliably

based on the data available during the production process. In the present dataset, the independent variables represent the chemical composition of the material as well as some additional sand and melting parameters, whereas the resulting microshrinkages are distinguished into 5 different *risk* classes. The 23 independent features from which the risk class is to be determined include the contents of Carbon (C), Silicon (Si), Magnesium (Mg), Phosphorus (P), Sulfur(S), Copper (Cu),Chromium (Cr), Manganese (Mn), Aluminium (Al), Cerium (Ce), Tin (Sn) and Zinc (Zn) as well as some additional properties related to the molding sand and the casting temperature.

Table 5.16: Statistics of microshrinkages dataset

	median	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
C	3.83	3.83	0.07	3.48	3.79	3.83	3.87	4.08
Si	2.27	2.28	0.09	2.00	2.21	2.27	2.33	2.69
Mg	0.04	0.04	0.00	0.03	0.03	0.04	0.04	0.05
P	0.02	0.02	0.00	0.01	0.02	0.02	0.02	0.02
S	0.01	0.01	0.00	0.00	0.00	0.01	0.01	0.01
Cu	0.14	0.14	0.03	0.04	0.12	0.14	0.17	0.38
Cr	0.07	0.07	0.02	0.04	0.06	0.07	0.08	0.18
Mn	0.35	0.35	0.04	0.25	0.33	0.35	0.37	0.58
Al	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.04
Ce	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.01
Sn	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.01
Zn	0.09	0.09	0.02	0.01	0.07	0.09	0.10	0.17
Ceq	4.60	4.59	0.07	4.25	4.55	4.60	4.64	4.81
Tliq	1162.90	1164.36	11.67	1143.40	1154.40	1162.90	1170.50	1206.00
Te min	1148.30	1147.78	2.30	1136.10	1146.73	1148.30	1149.30	1153.40
Recal	1.20	1.31	1.05	0.00	0.70	1.20	1.70	12.00
K	0.85	0.84	0.04	0.54	0.83	0.85	0.86	0.91
MouldComp	27.90	28.00	1.87	18.40	26.50	27.90	29.50	34.60
Campos	2.00	2.49	1.45	1.00	1.00	2.00	4.00	4.00
Ppressing	53.00	51.72	4.74	25.00	50.00	53.00	53.00	90.00
Tblowing	1.00	1.04	0.37	0.80	1.00	1.00	1.00	3.50
Inoculant	95.00	97.03	12.18	75.00	85.00	95.00	105.00	130.00
Tpouring	7.60	7.60	0.47	6.40	7.30	7.60	7.90	10.50
Tapouring	1398.00	1396.60	11.12	1323.00	1391.00	1398.00	1404.00	1424.00

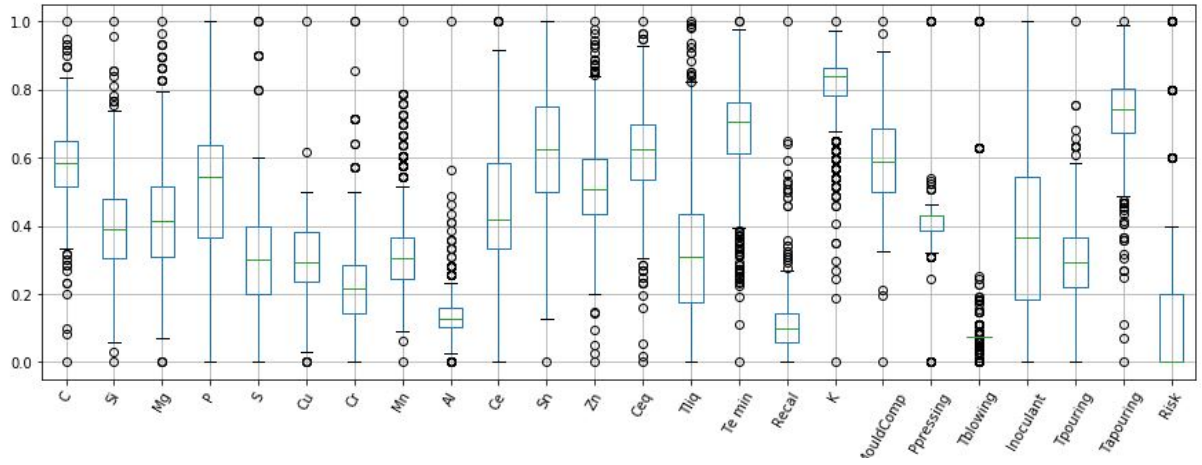


Figure 5.29: Boxplot of data distribution in the mircroshrinkage dataset

The heatmap of the correlation matrix for the dataset is shown in Fig. 5.30. Note that there is very little correlation between the risk and any of the independent variables, which means that no immediate identification of a major cause for microshrinkages can be identified without further analysis.

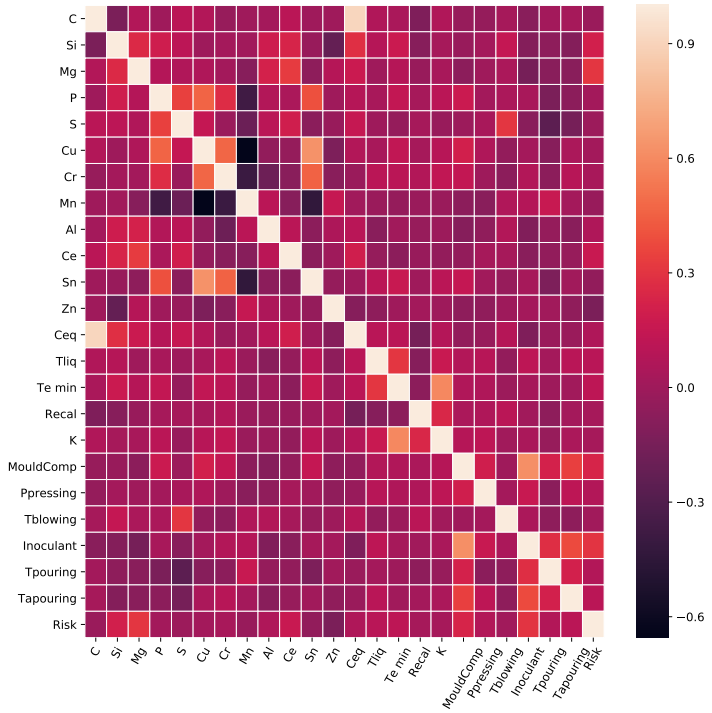


Figure 5.30: Heatmap of the correlation matrix for mircroshrinkage dataset

From the class frequencies of the dependent variable, as shown in Table 5.17, it is immediately clear that the dataset is heavily unbalanced (i.e. that the classes are not uniformly distributed within the available data). In the following, we will therefore use this dataset to investigate specific pre-processing methods for unbalanced classification data. Such preprocessing methods can easily be integrated into the unified framework described in Chapter 4, since an imbalance in the class distribution is easy to identify in an automated fashion.

Table 5.17 lists the frequencies of each label in the dataset. We can observe that majority of the samples are from class 0. In particular, 10% test data corresponds to only 3 samples in the minority classes. Therefore, we perform the same experiment with 25% to ensure slightly more significant results.

Table 5.17: Microshrinkages label frequencies

Class	**Complete Data**	**Learning with 90% data**				**Learning with 75% data**			
		Original	**Undersampling**	**SMOTE**	**Test Data**	**Original**	**Undersampling**	**SMOTE**	**Test Data**
0	682	614	27	614	68	512	22	512	171
1	90	81	27	614	9	68	22	512	22
2	68	61	27	614	7	51	22	512	17
3	35	31	27	614	4	26	22	512	9
4	30	27	27	614	3	22	22	512	7
5	31	28	27	614	3	23	22	512	8

When we apply our machine learning methods to 85% of the data from this unbalanced dataset, we obtain the results shown in table 5.18. The values of macro average and weighted average of f1-score shows that method CMNB performed best among all compared methods. However, from the results, it is obvious that for this severely unbalanced dataset, the prediction does not perform well on the minority classes.

Table 5.18: Microshrinkages: Classification results for original data using 10% test data

	LR			**SVC**			**GBDT**			**CMNB**			
Classes	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Support**
0	0.77	0.97	0.86	0.72	1.00	0.84	0.77	0.98	0.86	0.87	0.86	0.86	68
1	0.23	0.11	0.14	0.00	0.00	0.00	0.24	0.06	0.10	0.27	0.37	0.30	9
2	0.16	0.07	0.10	0.01	0.01	0.01	0.16	0.07	0.09	0.30	0.24	0.25	7
3	0.00	0.00	0.00	0.00	0.00	0.00	0.09	0.03	0.05	0.06	0.03	0.03	4
4	0.08	0.03	0.05	0.00	0.00	0.00	0.09	0.04	0.06	0.15	0.14	0.14	3
5	0.23	0.11	0.14	0.03	0.01	0.02	0.36	0.18	0.22	0.26	0.27	0.25	3
Macro average	0.24	0.22	0.21	0.13	0.17	0.15	0.28	0.23	0.23	0.32	0.32	0.30	94
Weighted average	0.60	0.72	0.65	0.53	0.72	0.61	0.61	0.72	0.65	0.69	0.69	0.68	94
Accuracy	0.72			0.72			0.72			0.69			94

We attempt to overcome this challenge of unbalancedness by applying under-over sampling techniques to transform the dataset into a balanced one. When we apply the undersampling technique, we randomly reduce the size of the majority classes to the size of the class with minimum representation; for oversampling, we apply the SMOTE method to synthetically increase the size of the minority classes until it equals the size of the majority class. The resulting class frequencies are shown in Table 5.17.

After balancing, the same machine learning methods as before were applied to the modified dataset. The results for the undersampling and the SMOTE-based oversampling techniques are shown in Tables 5.19 and 5.20, respectively.

Table 5.19: Microshrinkages: Classification results for undersampled data using 10% test data

	LR			SVC			GBDT			CMNB			
Classes	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Support**
0	0.78	0.42	0.53	0.72	0.85	0.76	0.46	0.14	0.19	0.70	0.44	0.46	68
1	0.09	0.16	0.11	0.02	0.01	0.01	0.09	0.09	0.06	0.04	0.06	0.04	9
2	0.04	0.10	0.06	0.03	0.04	0.03	0.05	0.31	0.08	0.01	0.03	0.02	7
3	0.02	0.10	0.03	0.02	0.03	0.02	0.02	0.15	0.03	0.05	0.20	0.06	4
4	0.07	0.31	0.11	0.03	0.06	0.03	0.03	0.11	0.04	0.01	0.07	0.02	3
5	0.07	0.07	0.05	0.01	0.11	0.02	0.05	0.19	0.05	0.03	0.24	0.05	3
Macro average	0.18	0.19	0.15	0.14	0.18	0.15	0.12	0.17	0.08	0.14	0.18	0.11	94
Weighted average	0.58	0.34	0.40	0.52	0.62	0.56	0.35	0.15	0.15	0.52	0.35	0.34	94
Accuracy	0.34			0.62			0.15			0.35			94

Table 5.20: Classification results for SMOTE data using 10% test data

	LR			SVC			GBDT			CMNB			
Classes	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Support**
0	0.79	0.20	0.31	0.72	1.00	0.84	0.73	0.93	0.81	0.73	0.92	0.81	68
1	0.16	0.10	0.11	0.00	0.00	0.00	0.05	0.06	0.04	0.11	0.09	0.07	9
2	0.14	0.16	0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	7
3	0.02	0.10	0.03	0.00	0.00	0.00	0.03	0.01	0.01	0.00	0.00	0.00	4
4	0.04	0.23	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3
5	0.04	0.23	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3
Macro average	0.20	0.17	0.12	0.12	0.17	0.14	0.13	0.17	0.15	0.14	0.17	0.15	94
Weighted average	0.60	0.18	0.25	0.52	0.72	0.61	0.53	0.68	0.59	0.54	0.68	0.59	94
Accuracy	0.18			0.72			0.68			0.68			94

From the results, we can observe that there is no performance improvement due to under-over sampling. Precision and recall results from the learning algorithms show that both for original as well as for the SMOTE augmented dataset, SVC always predicted the majority class. It is interesting to note that in case of undersampling, SVC performance improved.

Using 25% test data, we repeated the experiments on all three datasets. The results are shown in Table 5.21 for the original data, in Table 5.22 for under-sampled data and in Table 5.23 for over-sampled data using SMOTE.

Table 5.21: Classification results for original data using 25% test data

	LR			SVC			GBDT			CMNB			
Classes	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Support**
0	0.78	0.95	0.86	0.73	1.00	0.84	0.77	0.97	0.86	0.87	0.84	0.85	171
1	0.26	0.15	0.18	0.00	0.00	0.00	0.25	0.08	0.11	0.22	0.36	0.27	22
2	0.19	0.08	0.11	0.00	0.00	0.00	0.23	0.08	0.11	0.21	0.17	0.17	17
3	0.12	0.02	0.03	0.00	0.00	0.00	0.09	0.03	0.05	0.04	0.02	0.03	9
4	0.25	0.09	0.13	0.00	0.00	0.00	0.19	0.06	0.08	0.18	0.14	0.15	7
5	0.39	0.19	0.25	0.00	0.00	0.00	0.41	0.13	0.19	0.34	0.35	0.33	8
Macro average	0.33	0.25	0.26	0.12	0.17	0.14	0.32	0.22	0.23	0.31	0.31	0.30	234
Weighted average	0.64	0.73	0.67	0.53	0.73	0.61	0.62	0.73	0.65	0.69	0.68	0.68	234
Accuracy	0.73			0.73			0.73			0.68			234

Table 5.22: Classification results for undersampled data using 25% test data

	LR			SVC			GBDT			CMNB			
Classes	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Precision**	**Recall**	**f1-score**	**Support**
0	0.77	0.35	0.46	0.74	0.83	0.76	0.47	0.13	0.18	0.65	0.29	0.36	171
1	0.12	0.19	0.13	0.04	0.04	0.03	0.06	0.14	0.08	0.06	0.15	0.07	22
2	0.10	0.15	0.09	0.02	0.03	0.02	0.04	0.20	0.06	0.05	0.08	0.05	17
3	0.04	0.15	0.05	0.01	0.01	0.01	0.04	0.17	0.05	0.03	0.27	0.04	9
4	0.07	0.34	0.11	0.03	0.06	0.04	0.03	0.12	0.03	0.04	0.13	0.05	7
5	0.10	0.18	0.11	0.03	0.13	0.05	0.06	0.30	0.06	0.07	0.22	0.07	8
Macro average	0.20	0.23	0.16	0.14	0.19	0.15	0.12	0.18	0.07	0.15	0.19	0.10	234
Weighted average	0.59	0.31	0.36	0.55	0.62	0.56	0.36	0.14	0.15	0.49	0.25	0.28	234
Accuracy	0.31			0.62			0.14			0.25			234

Table 5.23: Classification results for SMOTE data using 25% test data

	LR			SVC			GBDT			CMNB			
	precision	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**precision**	**recall**	**f1-score**	**support**
0	0.80	0.22	0.33	0.73	1.00	0.84	0.73	0.97	0.83	0.73	0.98	0.84	171
1	0.16	0.14	0.13	0.00	0.00	0.00	0.02	0.01	0.01	0.04	0.01	0.02	22
2	0.19	0.12	0.11	0.00	0.00	0.00	0.02	0.01	0.01	0.02	0.00	0.01	17
3	0.03	0.14	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	9
4	0.05	0.29	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7
5	0.04	0.25	0.07	0.00	0.00	0.00	0.02	0.00	0.01	0.03	0.00	0.01	8
Macro average	0.21	0.20	0.13	0.12	0.17	0.14	0.13	0.17	0.14	0.13	0.17	0.14	234
Weighted average	0.62	0.21	0.27	0.53	0.73	0.61	0.54	0.71	0.61	0.54	0.72	0.61	234
Accuracy	0.21			0.73			0.71			0.72			234

Again, the results obtained from the original data are better than those for the under-oversampled datasets. Among the learning methods, CMNB performance was better in comparison to others. SVC showed a similar trend as before and performed better for undersampled in comparison to original or oversampled data.

To summarize the results, we calculated the Kappa and MCC values to find which dataset and algorithm performed best altogether. The Kappa scores are shown in Figures 5.31 and 5.32 whereas the MCC scores are shown in Figures 5.33 and 5.34 for 10% and 25%, test data respectively. These measures can attain a maximum value of 1, which would indicate a perfect prediction. In general, ML algorithms with higher values performed better than others.

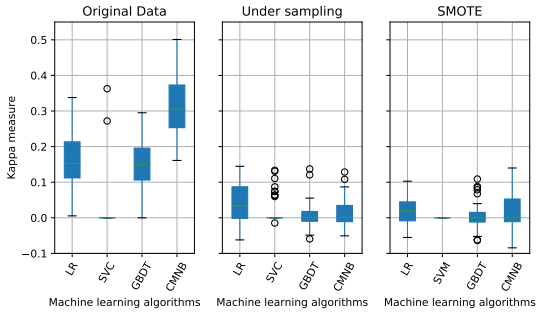


Figure 5.31: Kappa results for 10% test data

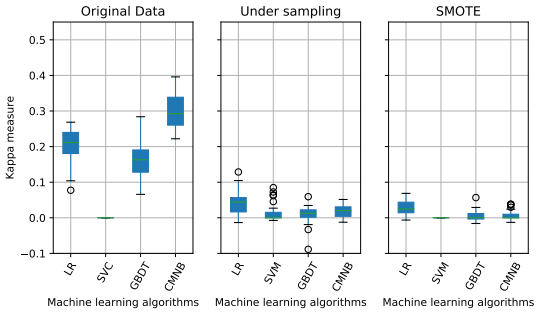


Figure 5.32: Kappa results for 25% test data

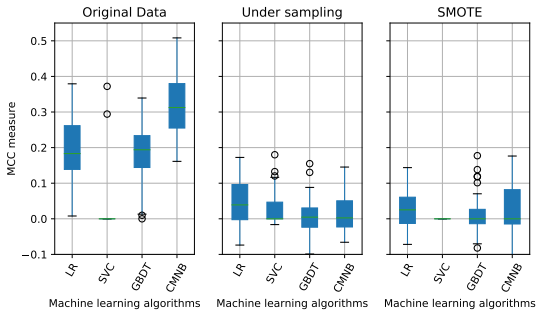


Figure 5.33: MCC results for 10% test data

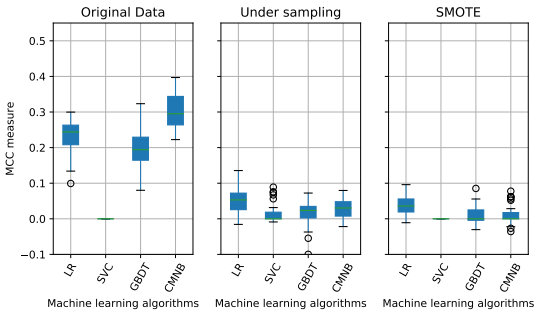


Figure 5.34: MCC results for 25% test data

The results demonstrate again that learning algorithms performed best with the original data and were not able to improve performance with balanced data using the selected undersampling and oversampling methods. For the original dataset, CMNB performed best in comparison to the base algorithms, followed by LR and GBDT.

6 Practical application

EIDOminer is the data analysis and machine learning software that resulted from the cooperation of a consortium working on the EU project *IPRO*. The consortium included a joint cooperation of the University of Duisburg-Essen, the Kemtpen University of Applied Sciences and multiple foundries. The software, which is designed to analyze and learn from the measured data to optimize processes, improve quality and save costs, consists of four main sub-parts, namely EIDOlearner, EIDOconsole, EIDOanalyser and EIDOpredictor, as shown in Fig. 6.1 and 6.2.

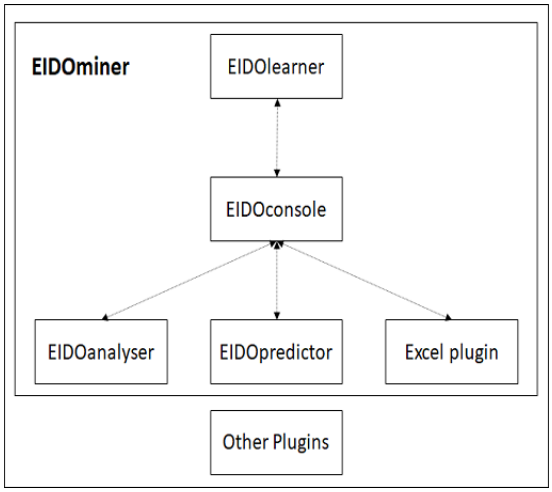


Figure 6.1: Abstract overview

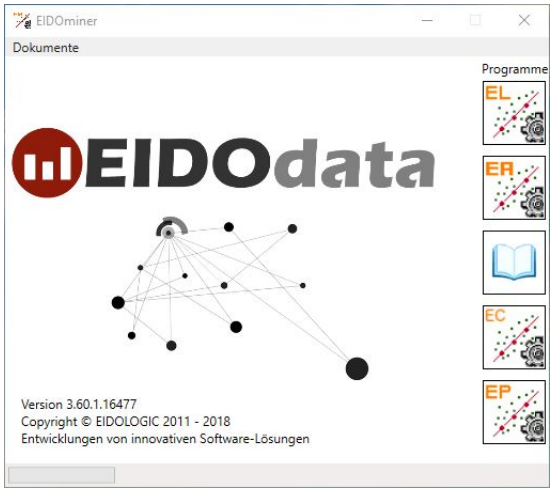


Figure 6.2: EIDOminer: Main window

The Intelligent Analysis Manager (IAM) is the core module of the EIDOlearner. It consists of 3 parts: the preprocessor, the function box and the postprocessor with the supervisor. The preprocessor provides the filtering and data smoothing functionality. Individual machine learning methods are then trained in the function box. Based on the performance of the individual learning methods, well-trained algorithms are selected in the functon box for the postprocessor. In this last stage, the supervisor combines the individual learning methods using various combination methods such as *weighted averaging* and MPF (labeled as PCAMR in EIDOminer) and evaluates which method offers the best solution. For different processes, the supervisor thereby selects suitable method to solve the specific problem at hand.

The EIDOminer provides access to the trained prediction function in the other sub-parts EIDOanalyser and EIDOpredictor using the EIDOconsole. As the name suggests, this module is a console application which, as input parameters, takes the name of the trained process in the form of the *Analysis History* and the *Analysis* as well as new unlabeled

input and returns a prediction of the output. A plugin to use the functionality of the EIDOlearner directly in Microsoft Excel® is also part of the EIDOminer suite. Similarly, the EIDOconsole can be used by other third party applications like Matlab® to make use of the learnt prediction function.

The EIDOanalyser provides the functionality to analyse the data, to determine its quality and to show how the input is related to the output. It can also be used to predict new data based on the model trained in the EIDOlearner. We can observe the data distribution by dividing the label values from minimum to maximum into pre-defined classes of equal range.If we are not able to train a good prediction function, then one of the possible reasons can be that some rows with significantly different label values are contained in the dataset for which the input values are very close or even (almost) identical; note that such an occurence would suggest multiple distinct label values to be predicted for (almost) the same input, which cannot be expected from a (regular) ML algorithm.[7] We can find such rows using a *Predictability* function available inside the EIDOanalyser. We can also create two dimensional charts in the EIDOanalyser using the original data, calculation data or prediction data; here, "original data" refers to the values from the EIDOlearner, while "calculation data" is newly predicted in the EIDOanalyser. Normally, we use calculation data to observe the nonlinear relationship between a selected feature and a label. To this end, we generate data by varying the selected feature value between a minimum and maximum value, keeping all the other feature values constant, predicting the output corresponding to this generated input and creating a two dimensional plot between this feature and the label to observe the relationship.

The EIDOpredictor provides the functionality for prediction, knowledge generation and backward analysis. It accepts the current input of the running process and passes it to the EIDOminer console to obtain the prediction. For knowledge generation, the EIDOpredictor creates a huge input, passes it to EIDOminer console for prediction and saves both the generated input and the predicted output inside a database. Once the knowledge generation phase is complete, the EIDOpredictor starts monitoring the running process by checking the prediction for each input. As soon as a predicted label value lies outside the permitted range, the backward analysis is performed to propose a possible change to the input. The suggested minimal change to keep the process production within the acceptable range is based on the knowledge database. This application therefore implements all three stages of the unified framework introduced in Section 4.

In the following, we consider the Elongation dataset described in Section 5.1.2 as an example to demonstrate the use of this software.

[7] In fact, this phenomenon strongly suggests that parameters which influence the dependent variable are missing from the dataset or, more generally, that the label is not fully determined by the input features alone.

6.1 Learning stage

In the learning stage, the dataset is loaded into the software EIDOlearner and is by default split into two parts, for learning (90%) and testing (10%); the split ratio can also be changed if so desired. Then preprocessing techniques are selected from the choice of outlier removal, convolution, FFT, moving average, SVD and KPCA, which are then applied to the learning data; of course, it is also possible to include the raw data means without any preprocessing. Afterwards, each of the preprocessing techniques produces a new representation of the dataset. Each of these datasets are then supplied to the selected machine learning algorithms, which can include Bayesian networks, two variants of decision trees (C4.5 and CART), KNN, MR, NN and SVM, as shown in Fig. 6.3.

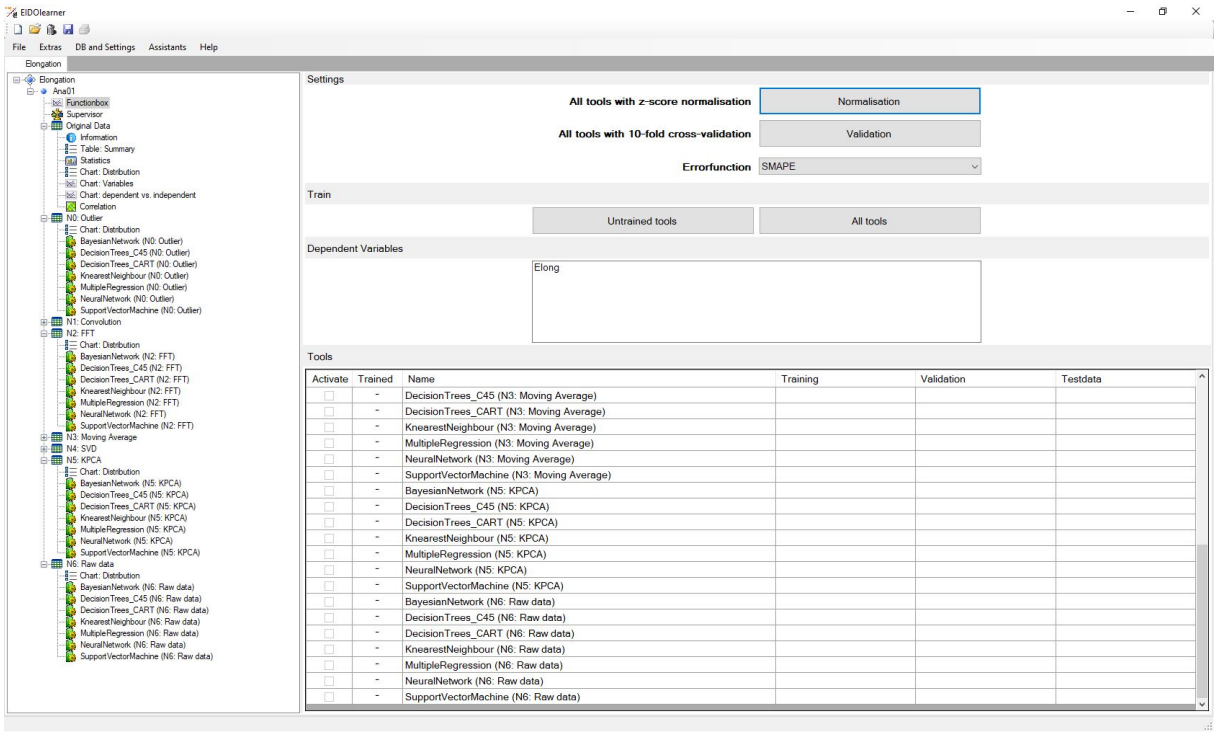


Figure 6.3: EIDOlearner: Functionbox window

With z-score normalization, 10-fold cross validation and a predefined acceptance threshold for each of the learning methods, all the base algorithms are trained as shown in Fig. 6.4. Here, we can observe that more than one algorithm is trained successfully.

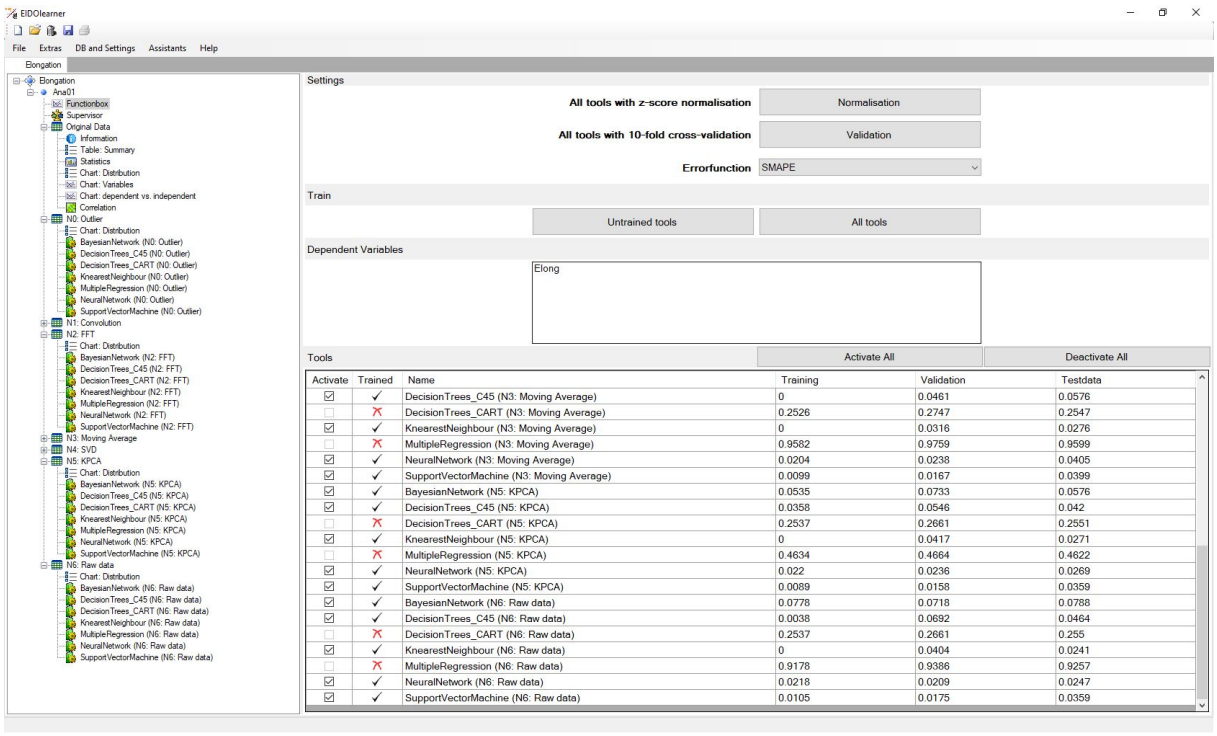
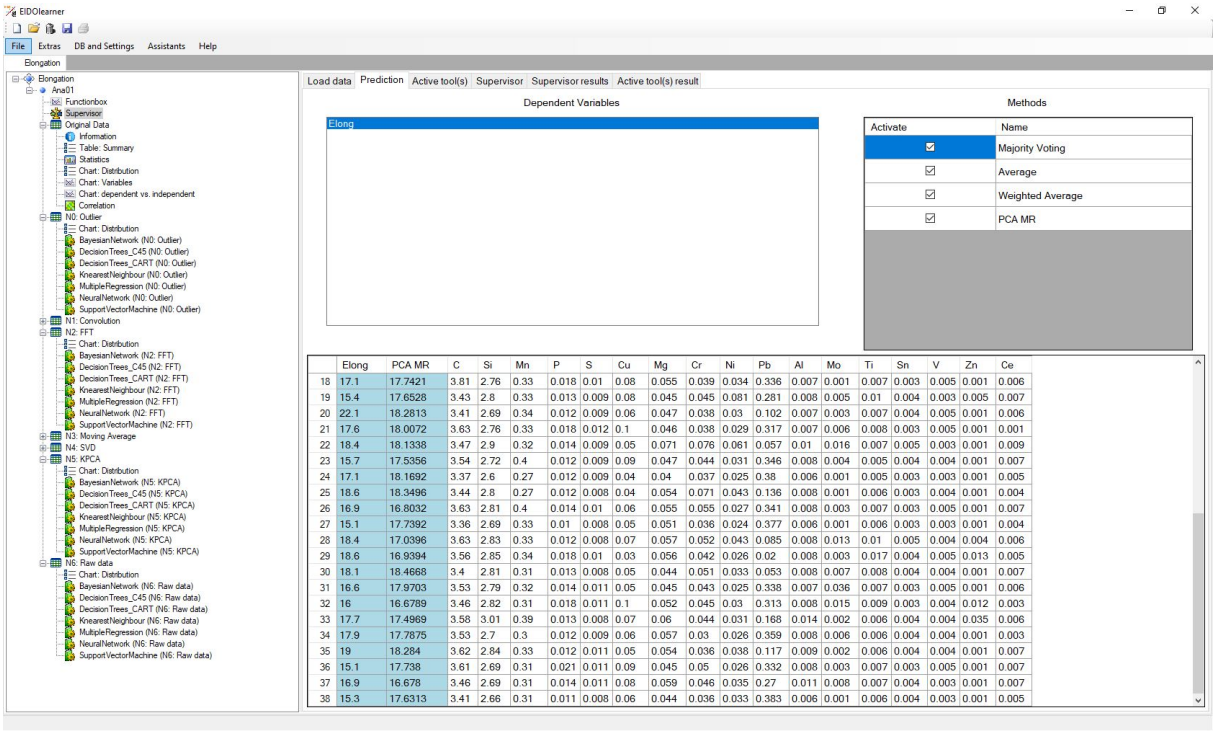


Figure 6.4: EIDOlearner: Functionbox trained window

Since we would like to utilize the expertise of each of the trained algorithms, the supervisor provides four combination methods: majority voting, average, weighted average and PCAMR. The combination method with least error on the learning data is selected as shown in Fig. 6.5. In this case, the combination method PCAMR produced the best results and is therefore selected. Once the prediction function has been trained, it is saved in the software to be used in the EIDOpredictor.



	Elong	PCA MR	C	Si	Mn	P	S	Cu	Mg	Cr	Ni	Pb	Al	Mo	Ti	Sn	V	Zn	Ce
18	17.1	17.7421	3.81	2.76	0.33	0.018	0.01	0.08	0.055	0.039	0.034	0.336	0.007	0.001	0.007	0.003	0.005	0.001	0.006
19	15.4	17.6528	3.43	2.8	0.33	0.013	0.009	0.08	0.045	0.045	0.081	0.281	0.008	0.005	0.01	0.004	0.003	0.005	0.007
20	22.1	18.2813	3.41	2.69	0.34	0.012	0.009	0.06	0.047	0.038	0.03	0.102	0.007	0.003	0.007	0.004	0.005	0.001	0.006
21	17.6	18.0072	3.63	2.76	0.33	0.018	0.012	0.1	0.046	0.038	0.029	0.317	0.007	0.006	0.008	0.003	0.005	0.001	0.001
22	18.4	18.1338	3.47	2.9	0.32	0.014	0.009	0.05	0.071	0.076	0.061	0.057	0.01	0.016	0.007	0.005	0.003	0.001	0.009
23	15.7	17.5356	3.54	2.72	0.4	0.012	0.009	0.09	0.047	0.044	0.031	0.346	0.008	0.004	0.005	0.004	0.004	0.001	0.007
24	17.1	18.1692	3.37	2.6	0.27	0.012	0.009	0.04	0.04	0.037	0.025	0.38	0.006	0.001	0.005	0.003	0.003	0.001	0.005
25	18.6	18.3496	3.44	2.8	0.27	0.012	0.008	0.04	0.054	0.071	0.043	0.136	0.008	0.001	0.006	0.003	0.004	0.001	0.004
26	16.9	16.8032	3.63	2.81	0.4	0.014	0.01	0.06	0.055	0.055	0.027	0.341	0.008	0.003	0.007	0.003	0.005	0.001	0.007
27	15.1	17.7392	3.36	2.69	0.33	0.01	0.008	0.05	0.051	0.036	0.024	0.377	0.006	0.001	0.006	0.003	0.003	0.001	0.004
28	18.4	17.0396	3.63	2.83	0.33	0.012	0.008	0.07	0.057	0.052	0.043	0.085	0.008	0.013	0.01	0.005	0.004	0.004	0.006
29	18.6	16.9394	3.56	2.85	0.34	0.018	0.01	0.03	0.056	0.042	0.026	0.02	0.008	0.003	0.017	0.004	0.005	0.013	0.005
30	18.1	18.4668	3.4	2.81	0.31	0.013	0.008	0.05	0.044	0.051	0.033	0.053	0.008	0.007	0.008	0.004	0.004	0.001	0.007
31	16.6	17.9703	3.53	2.79	0.32	0.014	0.011	0.05	0.045	0.043	0.025	0.338	0.007	0.036	0.007	0.003	0.005	0.001	0.006
32	16	16.6789	3.46	2.82	0.31	0.018	0.011	0.1	0.052	0.045	0.03	0.313	0.008	0.015	0.009	0.003	0.004	0.012	0.003
33	17.7	17.4969	3.58	3.01	0.39	0.013	0.008	0.07	0.06	0.044	0.031	0.168	0.014	0.002	0.006	0.004	0.004	0.035	0.006
34	17.9	17.7875	3.53	2.7	0.3	0.012	0.009	0.06	0.057	0.03	0.026	0.359	0.008	0.006	0.006	0.004	0.004	0.001	0.003
35	19	18.284	3.62	2.84	0.33	0.012	0.011	0.05	0.054	0.036	0.038	0.117	0.009	0.002	0.006	0.004	0.004	0.001	0.007
36	15.1	17.738	3.61	2.69	0.31	0.021	0.011	0.09	0.045	0.05	0.026	0.332	0.008	0.003	0.007	0.003	0.005	0.001	0.007
37	16.9	16.678	3.46	2.69	0.31	0.014	0.011	0.08	0.059	0.046	0.035	0.27	0.011	0.008	0.007	0.004	0.003	0.001	0.007
38	15.3	17.6313	3.41	2.66	0.31	0.011	0.008	0.06	0.044	0.036	0.033	0.383	0.006	0.001	0.006	0.004	0.003	0.001	0.005

Figure 6.5: EIDOlearner: Superviser window

6.2 Knowledge generation stage

Fig. 6.6 shows the main window of the EIDOpredictor. Analysis history *Elongation* and analysis *Ana01* are selected in the left pane and, on top, the tab *Data table* is active. The original data is shown in the middle pane and the acceptable elongation limits (15–19) are provided in the right pane.

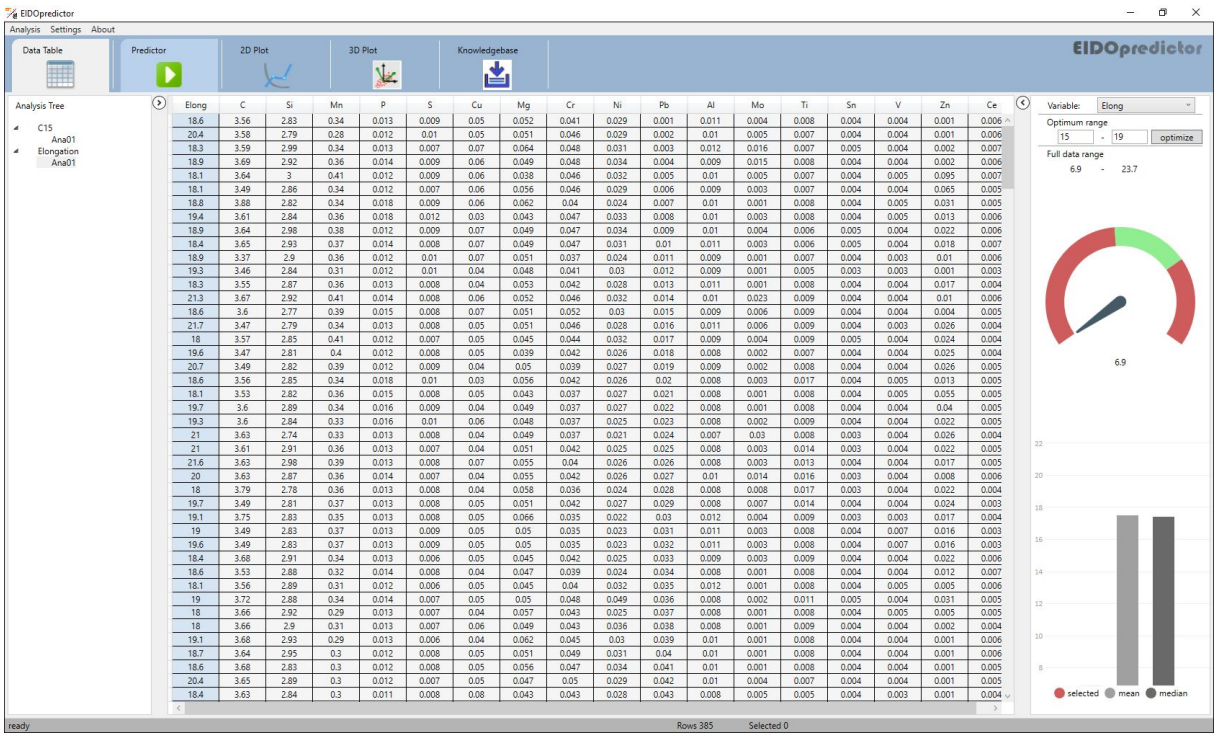


Figure 6.6: EIDOpredictor: Setting limits

To generate the knowledge for the selected analysis, we select the tab *Knowledgebase* as shown in Fig. 6.7. Here, for each of the features, lower and upper limits as well as the number of steps are selected. Based on these settings, knowledge is now generated. Since there are 17 features in this dataset, choosing 10 steps for each feature will result in 17^{10} instances, which would by far exceed the available storage capacity. However, it is possible that not all the features are important. In fact, changing some of the input values might not have any significant influence on the value of the predicted output. Furthermore, some features might be too expensive or practically not feasible to change. In such a scenario, we need to determine which features are important to consider as controllable parameters, which should be the only ones included in the variation for the knowledge generation. There are some methods which can help us determine the important features. For example, using a coraviance matrix, we can find linear relationships between the features and the label. Features with higher SHAP values are also considered more important. Wrapper methods such as forward selection, backward elimination or recursive feature elimination using a good prediction function can also be used to determine the most important independent variables. Once we have determined these features, the number of steps can be adjusted accordingly, while the thresholds can be chosen by keeping in mind their costs and othr practical limitations. In this case, since we observed from

the covariance matrix that the feature *Pb* is strongly related to *elongation*, thus 10 steps are selected for this particular feature and 2 steps are selected for the remaining ones to generate the knowledge.

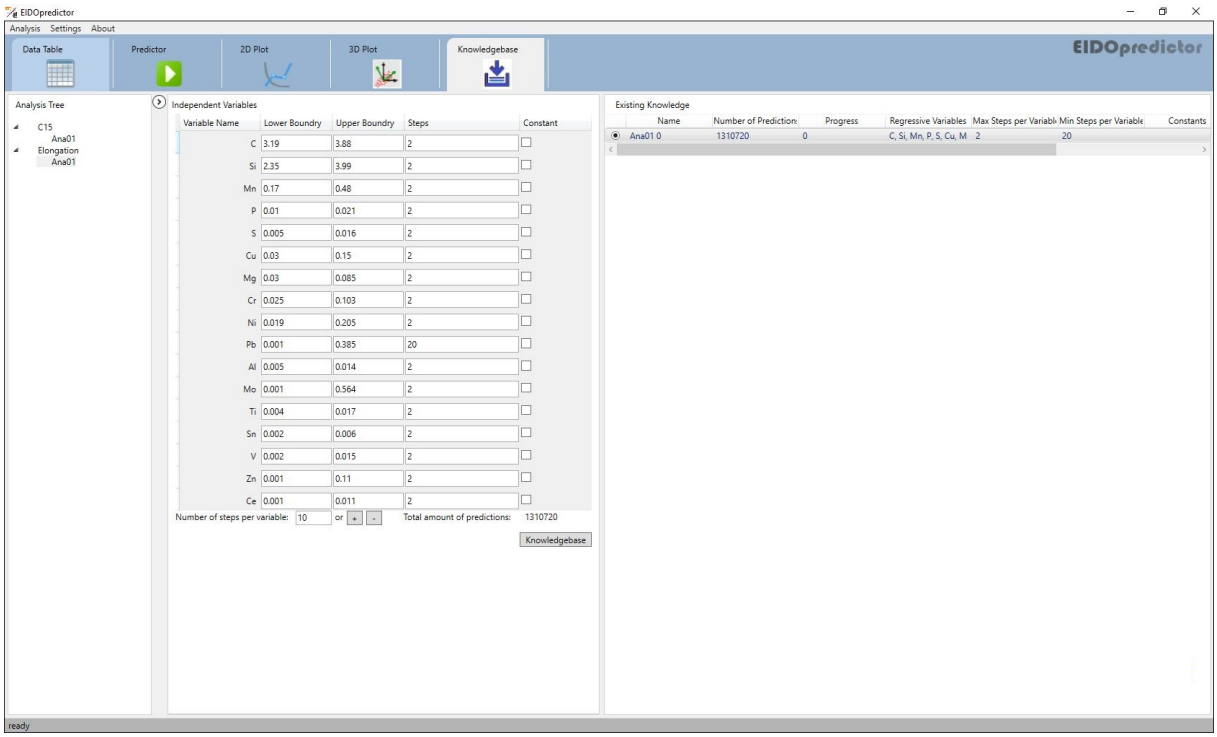


Figure 6.7: EIDOpredictor: Generating knowledge

6.3 Monitoring and control stage

In the monitoring and control stage, as soon the input leads to a predicted elongation value outside the limit, possible changes to the inputs are suggested as shown in Fig. 6.8. Here, we observe that the value of *elongation* is 18.45. Therefore, the optimization method is called, which searches the top 5 inputs closest to the current input but generates *elongation* values inside the acceptable range. We can then select one of these proposed changes to our parameters in order to keep the *elongation* value between the acceptable thresholds.

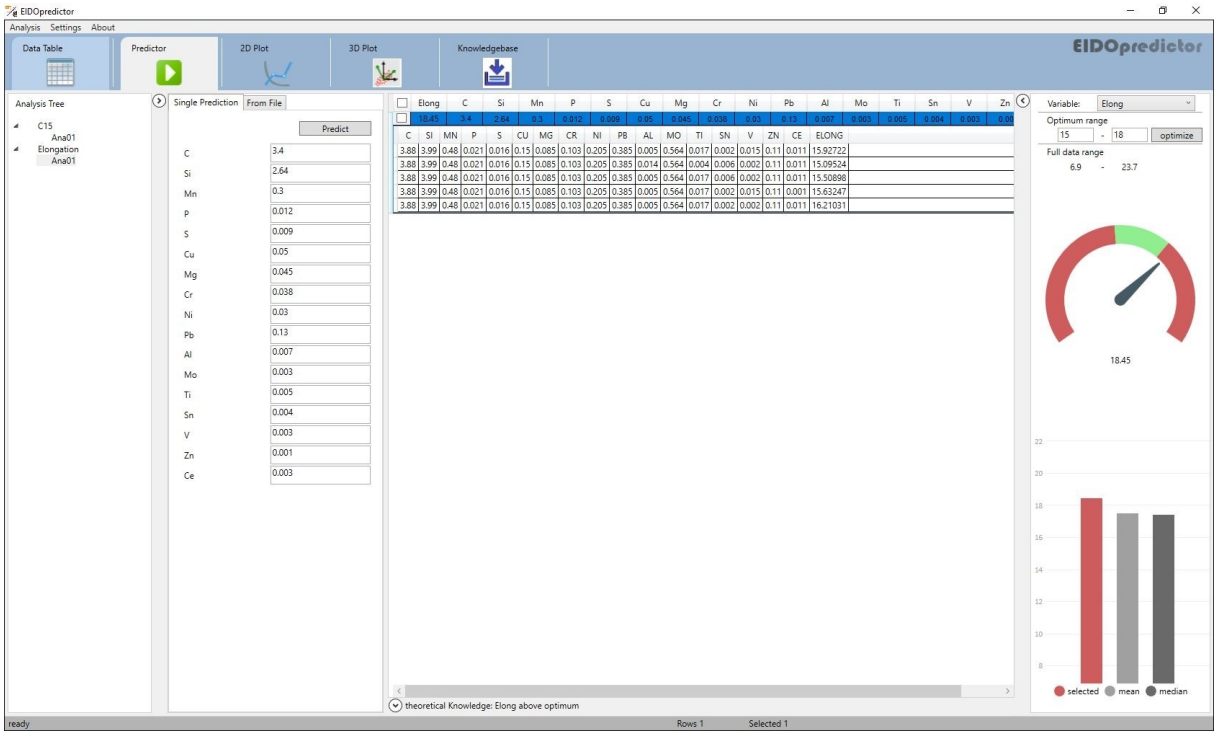


Figure 6.8: EIDOpredictor: Suggested changes

7 Conclusion and outlook

The work outlined in this thesis contributes to the availability of advanced machine learning techniques for the foundry industry by providing a simple unified framework which can directly be applied to the process monitoring and control of casting processes. The proposed framework consists of an algorithm for automatically training a suitable prediction function on a given dataset, a method for generating a knowledge database from the trained prediction function and, finally, a monitoring and control technique which, based on the prediction function, suggests changes to the control variables in order to improve the outcome of the process.

Using multiple datasets obtained directly from foundry companies, it has been demonstrated that among a variety of classical machine learning algorithms, it is not possible to select a single method which is optimal for all possible applications related to the casting process. The more extensive procedure employed in our framework – which consists of training multiple algorithms with a number of pre- and postprocessing techniques and combining the ones which performs best in a particular situation – therefore provides a major advantage over more direct applications of a single pre-selected machine learning method. It was also shown that for datasets of sufficient quality, an accurate prediction of different characteristics of casting parts is indeed possible using the algorithm described in the learning stage. The remaining two stages, i.e. the knowledge generation and the monitoring and control, have been successfully implemented as well. While these implementations have not yet been validated in a production environment, their functionality has been tested extensively in the context of teaching and research projects using the *EIDOminer* software, which has been developed in parts by the author.

In particular, the application *EIDOminer* has been used and extended in two ZIM supported research projects, namely *IPROguss* and *AUTOTemp*, both of which were related to applications in the foundry industry: While the *IPROguss* extended the *IPRO* project that originally motivated the development of the *EIDOminer* software, the aim of the project *AUTOTemp* was to automatically control a grid surface temperature on a metal die casting using cooling channels to keep a homogeneous surface temperature. Here, the data collected and used for model generation during the experiments included the initial and final surface temperature, the time lapse and the cooling channel settings, namely flow speed and temperature of the liquid flowing through the channel.

7.1 Future work

While the proposed framework has demonstrated its applicability on a number of datasets, the availability of high-quality process data remains the foundation of any machine learn-

ing technique. In order to implement our framework in an small and medium-sized enterprise, data collection needs to become an integral part of the foundry's IT infrastructure. Once comprehensive data is available for a casting process, our method could easily be implemented, which would allow for a full validation of the proposed framework. Of course, the availability of extensive process data would provide other benefits as well, such as a digital overview of the production state with detailed real-time quantities of the materials for improved process control.

The framework might also be adapted to include hybrid predictions based on analytical as well as data driven modeling. Theoretical expertise could thereby be incorporated into the model more directly, and thus foundry-specific knowledge of the production processes could ensure that the impact of low data quality or sensor errors are minimized.

The Meta Prediction Function (MPF) in the EIDOlearner can also easily be extended to include addtional machine learning methods like variants of decision trees [109] and Bayesian networks [52]. Other feature selection techniques can be incorporated as well to improve the quality of the dataset or to reduce its size. Deep learning techniques have also shown very good results in datasets related to the identification and classification of casting parts and can be included directly into the EIDOlearner. Furthermore, the usage of the proposed MPF is not limited to the prediction of casting defects, but can also be used for forecasting cost, expenditure or sales prices related to the production process as well. Finally, the proposed framework is not limited to the foundry industry and can be applied in other industrial settings as well.

References

[1] ActiveWizards. Top 8 data science use cases in manufacturing. `https://www.kdnuggets.com/2019/03/top-8-data-science-use-cases-manufacturing.html`, 2019.

[2] T. Ahmad, H. Chen, R. Huang, G. Yabin, J. Wang, J. Shair, H. M. A. Akram, S. A. H. Mohsan, and M. Kazim. Supervised based machine learning models for short, medium and long-term energy prediction in distinct building environment. *Energy*, 158:17–32, 2018.

[3] S. B. Alvi, R. Martin, and J. Gottschling. Efficient use of hybrid adaptive neuro-fuzzy inference system combined with nonlinear dimension reduction method in production processes. In *Proceedings of the 4th International Conference on Information Technology, Control, Chaos, Modeling and Applications*, pages 29–43, 2017.

[4] Y. Bai, Z. Sun, J. Deng, L. Li, J. Long, and C. Li. Manufacturing quality prediction using intelligent learning approaches: A comparative study. *Sustainability*, 10(1):85, 2018.

[5] Y. Bai, Z. Sun, B. Zeng, J. Long, L. Li, J. V. de Oliveira, and C. Li. A comparison of dimension reduction techniques for support vector machine modeling of multi-parameter manufacturing quality prediction. *Journal of Intelligent Manufacturing*, 30(5):2245–2256, 2019.

[6] C. Bartels and J. Gottschling. Anwendung intelligenter datenanalyse in gießereien. Tagungsband Formstofftage, 2014.

[7] A. Bashar. Intelligent development of big data analytics for manufacturing industry in cloud computing. *Journal: Journal of Ubiquitous Computing and Communication Technologies September*, 2019(01):13–22, 2019.

[8] R. S. Batbooti and R. S. Ransing. Data mining and knowledge discovery approach for manufacturing in process data optimization. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 203–208. Springer, 2015.

[9] J. Bell. *Machine Learning: Hands-On for Developers and Technical Professionals*. Wiley, Indianapolis, IN, 2014.

[10] A. Binding, N. Dykeman, and S. Pang. Machine learning predictive maintenance on data in the wild. *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pages 507–512, 2019.

[11] Z. Birkner. Industry 4.0 - opportunity or challenge? 2018.

[12] J. Campbell. *Complete Casting Handbook: Metal Casting Processes, Techniques and Design*. Complete Casting Handbook: Metal Casting Processes, Metallurgy, Techniques and Design. Elsevier Science, 2011.

[13] T. Carvalho, F. Soares, R. Vita, R. Francisco, J. P. Basto, and S. G. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Comput. Ind. Eng.*, 137, 2019.

[14] I. M. Cavalcante, E. M. Frazzon, F. A. Forcellini, and D. Ivanov. A supervised machine learning approach to data-driven simulation of resilient supplier selection in digital manufacturing. *International Journal of Information Management*, 49:86–97, 2019.

[15] C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[16] N. Chawla, A. K. Singh, H. Kumar, M. Kar, and S. Mukhopadhyay. Securing IoT Devices using Dynamic Power Management: Machine Learning Approach. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[17] T. Chen. A PCA-BPN approach for estimating simulation workload in cloud manufacturing. In *2015 Seventh International Conference on Ubiquitous and Future Networks*, pages 826–830. IEEE, 2015.

[18] P. Chokshi, Dashwood R., and D. J. Hughes. Artificial Neural Network (ANN) based microstructural prediction model for 22MnB5 boron steel during tailored hot stamping. *Computers & Structures*, 190:162 – 172, 2017.

[19] T. Cole. 5G partnerships for smart-factories from Deutsche Telekom. `https://www.smart-industry.net/5g-partnerships-for-smart-factories-from-deutsche-telekom/`, 2019.

[20] C. Cortes, L. D. Jackel, and W. P. Chiang. Limits on learning machine accuracy imposed by data quality. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, KDD'95, page 57–62. AAAI Press, 1995.

[21] H. N. Dai, H. Wang, G. Xu, J. Wan, and M. Imran. Big data analytics for manufacturing internet of things: opportunities, challenges and enabling technologies. *Enterprise Information Systems*, pages 1–25, 2019.

[22] P. R. de Matos, M. F. Marcon, R. A. Schankoski, and L. R. Prudêncio Jr. Novel applications of waste foundry sand in conventional and dry-mix concretes. *Journal of environmental management*, 244:294–303, 2019.

[23] A. M. Deif. A system model for green manufacturing. *Journal of cleaner production*, 19(14):1553–1559, 2011.

[24] Der Verein Deutscher Giessereifachleute. Bdguss: Verfahren und werkstoffe des giessenverfahrens (casting process and materials), 2010.

[25] S. K. Dey, R. G. Narayanan, and G. S. Kumar. Computing the tensile behaviour of tailor welded blanks made of dual-phase steel by neural network-based expert system. *International Journal of Computer Integrated Manufacturing*, 25(2):158–176, 2012.

[26] M. Dib, B. Ribeiro, and P. Prates. Model prediction of defects in sheet metal forming processes. In *International Conference on Engineering Applications of Neural Networks*, pages 169–180. Springer, 2018.

[27] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10(7):1895–1923, oct 1998.

[28] T. G. Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[29] E. Duarte and J. Wainer. Empirical comparison of cross-validation and internal metrics for tuning svm hyperparameters. *Pattern Recogn. Lett.*, 88(C):6–11, March 2017.

[30] A. F. Fahmy, H. K. Mohamed, and A. Yousef. A data mining experimentation framework to improve six sigma projects. *2017 13th International Computer Engineering Conference (ICENCO)*, pages 243–249, 2017.

[31] Hossam Faris, Alaa Sheta, and Ertan Öznergiz. Modelling hot rolling manufacturing process using soft computing techniques. *International Journal of Computer Integrated Manufacturing*, 26(8):762–771, 2013.

[32] C. Felden, C. Koschtial, and J. Buder. *Predictive Analytics in der Strategischen Anlagenwirtschaft*, pages 519–537. 03 2012.

[33] M. Fischer, M. Pourbafrani, M. Kemmerling, and V. Stich. A framework for online detection and reaction to disturbances on the shop floor using process mining and machine learning. 2020.

[34] FOUNDARY LEXICON. Wet tensile strength. `https://www.giessereilexikon.com/en/foundry-lexicon/Encyclopedia/show/wet-tensile-strength-4418/?cHash=92102c63853dfd3b89af0b60381657ff`, 2022.

[35] C. Fragassa, M. Babic, C. Bergmann, and G. Minak. Predicting the tensile behaviour of cast alloys by a pattern recognition analysis on experimental data. *Metals*, 9:557, 05 2019.

[36] H. Gao, T. J. Struble, C. W. Coley, Y. Wang, W. Green, and K. Jensen. Using machine learning to predict suitable conditions for organic reactions. *ACS Central Science*, 4:1465 – 1476, 2018.

[37] M. Ghobakhloo. Industry 4.0, digitization, and opportunities for sustainability. *Journal of cleaner production*, 252:119869, 2020.

[38] C. Gröger, M. Hillmann, F. Hahn, B. Mitschang, and E. Westkämper. The operational process dashboard for manufacturing. *Procedia Cirp*, 7:205–210, 2013.

[39] C. Gröger, F. Niedermann, and B. Mitschang. Data mining-driven manufacturing process optimization. In *Proceedings of the world congress on engineering*, volume 3, pages 4–6, 2012.

[40] M. P. Groover. *Fundamentals of Modern Manufacturing: Materials, Processes, and Systems*. John Wiley & Sons, New York, 5th edition edition, 2012.

[41] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature extraction: foundations and applications*, volume 207. Springer, 2008.

[42] E. Hamouche and E. G. Loukaides. Classification and selection of sheet forming processes with machine learning. *International Journal of Computer Integrated Manufacturing*, 31(9):921–932, 2018.

[43] L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

[44] Q. He, H. Wu, H. Meng, Z. Hu, and Z. Xie. Molten steel level detection by temperature gradients with a neural network. *IEEE Access*, PP:1–1, 05 2019.

[45] C. N. Höfer and G. Karagiannis. Cloud computing services: taxonomy and comparison. *Journal of Internet Services and Applications*, 2(2):81–94, 2011.

[46] K. Hosein, K. Atashgar, and K. S. M. Saeed. Development of a new expert system for statistical process control in manufacturing industry. 2013.

[47] A. Hotho, A. Nürnberger, and G. Paaß. A brief survey of text mining. Jan 2005.

[48] S. Hou, X. Zhang, W. Dai, X. Han, and F. Hua. Multi-model- and soft-transition-based height soft sensor for an air cushion furnace. *Sensors*, 20(3), 2020.

[49] H. Hui, C. Zhou, S. Xu, and F. Lin. A novel secure data transmission scheme in industrial internet of things. *China Communications*, 17:73–88, 2020.

[50] B. C. Hwang. Intelligent control for a nuclear power plant using artificial neural networks. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 4, pages 2580–2584 vol.4, 1994.

[51] J. S. R. Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, 23(3):665–685, 1993.

[52] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, N. J. Krogan, S. Chung, A. Emili, M. Snyder, J. F. Greenblatt, and M. Gerstein. A bayesian networks approach for predicting protein-protein interactions from genomic data. *science*, 302(5644):449–453, 2003.

[53] I. T. Jolliffe. A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 31(3):300–303, 1982.

[54] J. Jouganous, R. Savidan, and A. Bellec. A brief overview of automatic machine learning solutions (automl). May 2018.

[55] S. R. Kalidindi, S. R. Niezgoda, and A. A. Salem. Microstructure informatics using higher-order statistics and efficient data-mining protocols. *JOM*, 63(4):34–41, Apr 2011.

[56] H. S. Kang, J. Y. Lee, S. Su Choi, H. Kim, J. H. Park, Ji Y. Son, Bo H. Kim, and S. Do Noh. Smart manufacturing: Past research, present findings, and future directions. *International Journal of Precision Engineering and Manufacturing-Green Technology*, 3(1):111–128, 2016.

[57] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda. Machine learning and data mining methods in diabetes research. *Computational and Structural Biotechnology Journal*, 15:104–116, 2017.

[58] F. D. Keskin and İ. Kabasakal. Application of machine learning methods with dimension reduction techniques for fault prediction in molding process. *Akademik Platform Mühendislik ve Fen Bilimleri Dergisi*, 8(2):371–378, 2019.

[59] V. V. Khanzode and J. Maiti. Improving foundry process control: an investigation of cluster analysis and path model. *International Journal of Productivity and Quality Management*, 2(4):404–422, 2007.

[60] V. V. Khanzode and J. Maiti. Implementing mahalanobis-taguchi system to improve casting quality in grey iron foundry. *International Journal of Productivity and Quality Management*, 3(4):444–456, 2008.

[61] D.J. Kim and B.M. Kim. Application of neural network and FEM for metal forming processes. *International Journal of Machine Tools and Manufacture*, 40(6):911 – 925, 2000.

[62] Knords Learning. Low Pressure Casting Process (All you need to know). `https://www.knordslearning.com/low-pressure-casting-process/`, 2022.

[63] R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[64] D. Krishnamurthy, H. Weiland, A. Farimani, E. Antono, J. Green, and V. Viswanathan. Machine learning based approaches to accelerate energy materials discovery and optimization. *ACS energy letters*, 4:187–191, 2019.

[65] H. Lahdhiri, M. Said, K. B. Abdellafou, O. Taouali, and M. F. Harkat. Supervised process monitoring and fault diagnosis based on machine learning methods. *The International Journal of Advanced Manufacturing Technology*, 102(5-8):2321–2337, 2019.

[66] Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, and A. Nandi. Applications of machine learning to machine fault diagnosis: A review and roadmap. *Mechanical Systems and Signal Processing*, 138:106587, 2020.

[67] S. Li, L. Xu, and S. Zhao. The internet of things: a survey. *Information Systems Frontiers*, 17:243–259, 2015.

[68] D. Lieber, M. Stolpe, B. Konrad, J. Deuse, and K. Morik. Quality prediction in interlinked manufacturing processes based on supervised & unsupervised machine learning. *Procedia Cirp*, 7:193–198, 2013.

[69] S. Liu, Y. Hu, C. Li, H. Lu, and H. Zhang. Machinery condition prediction based on wavelet and support vector machine. *Journal of Intelligent Manufacturing*, 28(4):1045–1055, 2017.

[70] Y. Liu, T. Zhao, W. Ju, and S. Shi. Materials discovery and design using machine learning. *Journal of Materiomics*, 3:159–177, 2017.

[71] Y. Lu, X. Xu, and Wang; L. Smart manufacturing process and system automation – a critical review of the standards and envisioned scenarios. *Journal of Manufacturing Systems*, 56:312–325, 2020.

[72] T. J. Lukka, T. Tossavainen, J. Kujala, and T. Raiko. Zenrobotics recycler - robotic sorting using machine learning. 2014.

[73] W. Mahnke and S. H. Leitner. OPC Unified Architecture The future standard for communication and information modeling in automation. 2009.

[74] W. E. Marcílio and D. M. Eler. From explanations to feature selection: assessing shap values as feature selection mechanism. In *2020 33rd SIBGRAPI conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 340–347. Ieee, 2020.

[75] W. C. Mark and W. S. Jude. Using neural networks for data mining. *Future Generation Computer Systems*, 13(2):211 – 229, 1997. Data Mining.

[76] Matmatch. Make better material decisions. `https://matmatch.com`, 2022.

[77] P. Mell and T. Grance. The NIST Definition of Cloud Computing. 2011.

[78] P. F. S. D. Melo, E. P. Godoy, P. Ferrari, and E. Sisinni. Open Source Control Device for Industry 4.0 Based on RAMI 4.0. *Electronics*, 10:869, 2021.

[79] S. Menard. *Applied logistic regression analysis*. Number 106. Sage, 2002.

[80] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[81] D. C. Montgomery. *Introduction to Statistical Quality Control*. John Wiley & Sons, New York, 6th edition edition, 2009.

[82] B. Neyestani. Seven basic tools of quality control: The appropriate quality techniques for solving quality problems in the organizations, mar 2017.

[83] L. Ni, Y. Liu, Y. C. Lau, and A. Patil. Landmarc: Indoor location sensing using active rfid. *Wireless Networks*, 10:701–710, 2003.

[84] T. Niesen, C. Houy, P. Fettke, and P. Loos. Towards an integrative big data analysis framework for data-driven risk management in industry 4.0. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 5065–5074. IEEE, 2016.

[85] J. Nieves, I. Santos, and P. G. Bringas. Combination of machine-learning algorithms for fault prediction in high-precision foundries. In Stephen W. Liddle, A. Min Schewe, Klaus-Dieterand Tjoa, and Xiaofang Zhou, editors, *Database and Expert Systems Applications*, pages 56–70, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[86] B. Nikolic, J. Ignjatic, N. Suzic, B. Stevanov, and A. Rikalovic. Predictive manufacturing systems in industry 4.0: Trends, benefits and challenges. *Annals of DAAAM & Proceedings*, 28, 2017.

[87] S. Nzuva and L. Nderu. The superiority of the ensemble classification methods: A comprehensive review. 08 2019.

[88] E. Oberg. Machinery's handbook. 2012.

[89] OpenLearn. Gravity die casting. `https://www.open.edu/openlearn/science-maths-technology/engineering-technology/manupedia/gravity-die-casting`, nov 2017.

[90] S. Otarawanna and A. K. Dahle. 6 - casting of aluminium alloys. In Roger Lumley, editor, *Fundamentals of Aluminium Metallurgy*, Woodhead Publishing Series in Metals and Surface Engineering, pages 141–154. Woodhead Publishing, 2011.

[91] N. Paltrinieri, L. Comfort, and G. Reniers. Learning about risk: Machine learning for risk assessment. *Safety Science*, 118:475–486, 2019.

[92] A. G. Parlos, J. Muthusami, and A. F. Atiya. Incipient fault detection and identification in process systems using accelerated neural network learning. *Nuclear Technology*, 105(2):145–161, 1994.

[93] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[94] M. Piltan, H. Shiri, and S. Ghaderi. Energy demand forecasting in iranian metal industry using linear and nonlinear models based on evolutionary algorithms. *Energy Conversion and Management*, 58:1–9, 2012.

[95] D. Preuveneers and E. Ilie-Zudor. The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in industry 4.0. *Journal of Ambient Intelligence and Smart Environments*, 9(3):287–298, 2017.

[96] R. H. Puffer Jr. Energy and waste minimization in the investment casting industry, 2003.

[97] J. K. Rai, A. M. Lajimi, and P. Xirouchakis. An intelligent system for predicting hpdc process variables in interactive environment. *Journal of Materials Processing Technology*, 203(1):72–79, 2008.

[98] R. S. Ransing, C. Giannetti, M. R. Ransing, and M. W. James. A coupled penalty matrix approach and principal component based co-linearity index technique to discover product specific foundry process knowledge from in-process data in order to reduce defects. *Computers in Industry*, 64(5):514–523, 2013.

[99] A. Ray et al. Optimization of process parameters of green electrical discharge machining using principal component analysis (pca). *The International Journal of Advanced Manufacturing Technology*, 87(5-8):1299–1311, 2016.

[100] M. Ren, Y. Song, and W. Chu. An improved locally weighted PLS based on particle swarm optimization for industrial soft sensor modeling. *Sensors*, 19(19):4099, 2019.

[101] S. Ren, Y. Zhang, Y. Liu, T. Sakao, D. Huisingh, and C. M. V. B. Almeida. A comprehensive review of big data analytics throughout product lifecycle to support sustainable smart manufacturing: a framework, challenges and future research directions. *Journal of cleaner production*, 210:1343–1365, 2019.

[102] S. Richter. *Statistisches und maschinelles Lernen*. 2019.

[103] Y. Roh, G. Heo, and S. E. Whang. A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33:1328–1347, 2021.

[104] R. Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag, Berlin, Heidelberg, 1996.

[105] D. Rolnick, P. Donti, L. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, A. Luccioni, T. Maharaj, E. D. Sherwin, S. Mukkavilli, K. Körding, C. P. Gomes, A. Y. Ng, D. Hassabis, J. C. Platt, F. Creutzig, J. Chayes, and Y. Bengio. Tackling climate change with machine learning. *ArXiv*, abs/1906.05433, 2019.

[106] E. Ruiz, M. Cuartas, D. Ferreño, L. Romero, V. Arroyo, and F. Gutiérrez-Solana. Optimization of the fabrication of cold drawn steel wire through classification and clustering machine learning algorithms. *IEEE Access*, 7:141689–141700, 2019.

[107] E. Ruiz, D. Ferreño, M. Cuartas, López A., Arroyo V., and F. Gutiérrez-Solana. Machine learning algorithms for the prediction of the strength of steel rods: an example of data-driven manufacturing in steelmaking. *International Journal of Computer Integrated Manufacturing*, 0(0):1–15, 2020.

[108] S. Sader, I. Husti, and M. Daróczi. Total quality management in the context of industry 4.0. 10 2017.

[109] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[110] J. Sahno, E. Shevtshenko, T. Karaulova, and K. Tahera. Framework for continuous improvement of production processes. *Engineering Economics*, 26(2):169–180, 2015.

[111] M. Saleem, S. Malik, J. Gottschling, D. Hartmann, and H. Gemming. Intelligent process control in foundry manufacturing. *International Foundary Research*, (66):18 – 31, 2014.

[112] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, 3(3):210–229, July 1959.

[113] B. Sánchez-Lengeling and A. Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361:360 – 365, 2018.

[114] I. Santos, J. Nieves, P. Bringas, and Y. Penya. *Machine-Learning-Based Defect Prediction in High-Precision Foundry Production*, pages 259–276. 01 2010.

[115] I. Santos, J. Nieves, Y. K. Penya, and P. G. Bringas. Machine-learning-based mechanical properties prediction in foundry production. In *2009 ICCAS-SICE*, pages 4536–4541, 2009.

[116] A. Sata. Investment casting defect prediction using neural network and multivariate regression along with principal component analysis. *International Journal of Manufacturing Research*, 11(4):356–373, 2016.

[117] B. Schölkopf, A. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[118] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[119] A. J. C. Sharkey. On Combining Artificial Neural Nets. *Connection Science*, 8(3-4):299–314, 1996. cited By 272.

[120] C. Shearer. The CRISP-DM Model: The New Blueprint for Data Mining. *J Data Warehousing*, (5):13 – 22, 2000.

[121] K. K. Shukla. A neurocomputer based learning controller for critical industrial applications. In *Proceedings of IEEE/IAS International Conference on Industrial Automation and Control*, pages 43–48, 1995.

[122] J. SJÖBERG and L. LJUNG. Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, 62(6):1391–1407, 1995.

[123] J. Sola and J. Sevilla. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 44(3 PART 3):1464–1468, 1997. cited By 217.

[124] K. Sreenivas, S. Sengupta, and M. Chakrabarty. Modcas – a knowledge based expert system for defect diagnosis in castings. *Indian Foundry Journal*, (6):1 – 8, 1994.

[125] statistica. Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030. `https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/`, 2022.

[126] StudentLesson. Different Types of Centrifugal Casting and their Working Principle). `https://studentlesson.com/different-types-of-centrifugal-casting-and-their-working-principle/`, dec 2019.

[127] K. Sudesh, G. Prakash, and H. Roshan. Knowledge-based expert system for analysis of casting defects. *Transactions of the American Foundrymen's Society*, pages 145–150, 1989.

[128] SUNRISE. Hot chamber die casting vs cold chamber die casting. `http://www.sunrise-metal.com/hot-chamber-die-casting-vs-cold-chamber-die-casting-2/`, 2022.

[129] F. Tao, Q. Qi, A. Liu, and A. Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018.

[130] Clariant Technology. Census of world casting production. `https://www.thewfo.com/census/`, 2021.

[131] THEmeter. Bulk specific weight and density. `http://www.themeter.net/pesi-muc_e.htm`, 2022.

[132] M. R. Thomsen, P. Nicholas, M. Tamke, S. Gatz, Y. Sinke, and G. Rossi. Towards machine learning for architectural fabrication in the age of industry 4.0. *International Journal of Architectural Computing*, 18:335 – 352, 2020.

[133] Thors. How to control Compactability of Sand? `https://thors.com/how-to-control-compactability-of-sand/`, 2022.

[134] H. Tian, M. Shuai, K. Li, and X. Peng. An Incremental Learning Ensemble Strategy for Industrial Process Soft Sensors. *Complexity*, 2019:5353296, May 2019.

[135] T. Tossavainen, J. Kujala, and T. Raiko. Robotic sorting using machine learning. 2014.

[136] V. D. Tsoukalas. Optimization of porosity formation in AlSi9Cu3 pressure die castings using genetic algorithm analysis. *Materials & Design*, 29(10):2027–2033, 2008.

[137] V. D. Tsoukalas. An adaptive neuro-fuzzy inference system (ANFIS) model for high pressure die casting. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 225(12):2276–2286, 2011.

[138] B. R. Upadhyaya and E. Eryurek. Application of neural networks for sensor validation and plant monitoring. *Nuclear Technology*, 97(2):170–176, 1992.

[139] VERSATILE. Right Way to Check Compactability. `https://sandtesting.com/2020/01/right-way-of-checking-compactability-for-green-sand/`, 2022.

[140] Susana M Vieira, Uzay Kaymak, and João MC Sousa. Cohen's kappa coefficient as a performance measure for feature selection. In *International conference on fuzzy systems*, pages 1–8. IEEE, 2010.

[141] S. Viswanathan, D. Apelian, R. J. Donahue, B. DasGupta, M. Gywn, J. L. Jorstad, R. W. Monroe, M. Sahoo, T. E. Prucha, and D. Twarog. *Casting*. ASM International, 12 2008.

[142] J. Walker. Don't be afraid of predictive analytics. May 2016.

[143] Prof. Dr. D. Wegener. *German Standardization Roadmap Industrie 4.0*. DIN and DKE ROADMAP, 03 2020.

[144] D. Weichert, P. Link, A. Stoll, S. Rüping, S. Ihlenfeldt, and S. Wrobel. A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology*, 104(5-8):1889–1902, 2019.

[145] S. A. Weissman and N. G. Anderson. Design of Experiments (DoE) and Process Optimization. A Review of Recent Publications. *Organic Process Research & Development*, 19(11):1605–1633, 2015.

[146] K. Whiting. These are 2018's most competitive economies. `https://www.weforum.org/agenda/2018/10/most-competitive-economies-global-competitiveness-report-2018/`, 2018.

[147] H. Wolf, R. Lorenz, M. Kraus, S. Feuerriegel, and T. Netland. *Bringing Advanced Analytics to Manufacturing: A Systematic Mapping*, pages 333–340. 08 2019.

[148] J. Wu and S. Coggeshall. *Foundations of Predictive Analytics (Chapman & HallCRC Data Mining and Knowledge Discovery Series)*. Chapman & HallCRC, 1st edition, 2012.

[149] T. Wuest, D. Weimer, C. Irgens, and K. D. Thoben. Machine learning in manufacturing: advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1):23–45, 2016.

[150] J. Yang, A. F. Frangi, J. Yang, D. Zhang, and Z. Jin. KPCA plus LDA: a complete kernel Fisher discriminant framework for feature extraction and recognition. *IEEE Transactions on pattern analysis and machine intelligence*, 27(2):230–244, 2005.

[151] P. Yang, J. Yang, B. Zhou, and A. Zomaya. A review of ensemble methods in bioinformatics. *Current Bioinformatics*, 5, 12 2010.

[152] A. G. V. Zacarias, P. Reimann, and B. Mitschang. A framework to guide the selection and configuration of machine-learning-based data analytics solutions in manufacturing. *Procedia CIRP*, 72:153–158, 2018.

[153] W. Zeng, B. Zhang, and A. Davoodi. Analysis of security of split manufacturing using machine learning. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 27(12):2767–2780, 2019.

[154] M. Zhang and B. Xue. Evolutionary computation for feature selection and feature construction. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pages 861–881, 2016.

[155] J. Zheng, Q. Wang, P. Zhao, and C. Wu. Optimization of high-pressure die-casting process parameters using artificial neural network. *The International Journal of Advanced Manufacturing Technology*, 44(7):667–674, 2009.

[156] Li Zheng, C. Zeng, L. Li, Y. Jiang, W. Xue, J. Li, C. Shen, W. Zhou, H. Li, L. Tang, T. Li, B. Duan, M. Lei, and P. Wang. Applying data mining techniques to address critical process optimization needs in advanced manufacturing. In *KDD '14*, 2014.

[157] Q. Zhu. On the performance of matthews correlation coefficient (mcc) for imbalanced dataset. *Pattern Recognition Letters*, 136:71–80, 2020.

DOI: 10.17185/duepublico/77522
URN: urn:nbn:de:hbz:465-20230505-133240-0