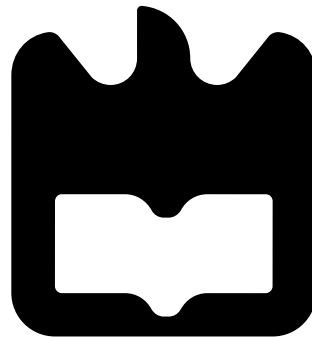




**Manuel Costa
Andrade Silva Xarez**

**Extração de Variáveis de Controlo a partir de
Processos de Fabrico utilizando Técnicas de
Aprendizagem Automática**

**Extracting Control Variables from Manufacturing
Processes with Machine Learning Techniques**





**Manuel Costa
Andrade Silva Xarez**

**Extração de Variáveis de Controlo a partir de
Processos de Fabricação utilizando técnicas de
Aprendizagem Automática**

**Extracting Control Variables from Manufacturing
Processes with Machine Learning techniques**

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica de Ana Tomé, Professora do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri / the jury

presidente / president

Professor Doutor Luís Filipe de Seabra Lopes

Professor Associado, Universidade de Aveiro (por delegação da Reitora da Universidade de Aveiro)

vogais / examiners committee

Professora Doutora Ana Maria Perfeito Tomé

Professora Associada, Universidade de Aveiro (orientadora)

Professor Doutor Joel P. Arrais

Professor Auxiliar Convidado, Universidade de Coimbra

**agradecimientos /
acknowledgements**

I would like to thank both Professor Ana Tomé and Professor Elmar Lang for all their guidance and help,
my supervisor, Peter Weiderer, for always pointing me in the right direction and for all his support throughout my thesis,
my family, the cornerstone for my growth
and finally, my friends with whom I shared all these years.

Resumo

A monitorização e optimização de processos industriais é um tópico complexo. Uma ampla quantidade de dados é capturada e as suas medidas são afetadas por várias interacções complexas. Devido a isto, existe uma necessidade crescente para métodos que reduzam tanto complexidade como dimensão destes dados de forma a ser interpretável por humanos. Nesta tese estudamos a extracção de regras interpretáveis derivadas de components extraídas por técnicas de decomposição de matrizes, nomeadamente Principal Component Analysis (PCA), Independent Component Analysis (ICA) e Non-negative Matrix Factorization (NMF), usando dados reais. As regras extraídas são usadas como guias para encontrar os melhores parâmetros para optimizar um processo industrial.

Abstract

Process monitoring and optimization in an industrial setting is a complex topic, a large amount of data is captured and its measurements are affected by many complex interactions. There is, therefore, a growing necessity for methods to reduce its complexity and dimension in order to be interpreted by humans. In this work we study the extraction of interpretable rules derived from components of matrix factorization techniques, namely Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Non-negative Matrix Factorization (NMF), using real data. We found that the rules can serve as guidelines to find the optimal parameters under which an industrial process can run.

Contents

Contents	i
List of Figures	iii
List of Tables	v
1 Introduction	1
1.1 Context	1
1.2 Motivation	1
1.3 Objective	2
1.4 Related Work	2
1.5 Thesis structure	3
2 Methods and Algorithms	5
2.1 Matrix Factorization in Industrial Data	5
2.1.1 Singular Value Decomposition and Principal Component Analysis .	6
2.1.2 Independent Component Analysis	7
2.1.3 Non-negative Matrix Factorization	9
2.2 Rule Fit in Industrial Data	11
2.2.1 Rule Fit	11
2.2.2 Logistic Regression	13
2.2.3 Random Forest Classifier	13
2.3 Summary	14
3 Data Pipeline Tools	15
3.1 Casting processes	16
3.1.1 Gravity Casting	16
3.1.2 High Pressure Casting	17
3.1.3 Process Chain	19
3.2 Sensors	20
3.3 Data	20
3.4 Gathering Data	21
3.5 Storing Data	22

3.6	Data Visualization and Preparation	25
3.7	Rule Creation	27
3.8	Summary	28
4	Industrial Case Study	31
4.1	Overview	31
4.1.1	Metrics	31
4.2	Case I: Gravity Casting - Temperature Analysis	31
4.2.1	Data Overview	31
4.2.2	Preparing Data	34
4.2.3	Feature Extraction	34
4.2.4	Random Forest Parameters	43
4.2.5	Rule Extraction	46
4.2.6	Results	48
4.2.7	Summary	48
4.3	Case II: High Pressure Casting - Temperature Analysis	48
4.3.1	Data Overview	48
4.3.2	Preparing Data	52
4.3.3	Feature Extraction	52
4.3.4	Random Forest Parameters	57
4.3.5	Rule extraction	58
4.3.6	Results	58
4.3.7	Summary	59
5	Concluding Remarks	61
5.1	Future Work	61
	Glossary	63
	Bibliography	65
	Appendices	69
A	Software	69

List of Figures

2.1	Illustrative example of two principal components.	8
2.2	Illustrative example of two independent components.	9
2.3	Illustration of the NMF where matrix \mathbf{W} indicates the coefficient of each factor defined by matrix \mathbf{H}	10
2.4	Illustration of the connection between supervised and unsupervised methods.	11
2.5	Illustration for the definition of a rule for a specific node.	12
3.1	Gravity casting work steps.	16
3.2	Placement of the sand core.	17
3.3	Pouring of the liquid aluminum.	18
3.4	High pressure casting schematic. [1]	18
3.5	The two different information courses.	22
3.6	Entity relationship diagram of the relational database.	24
3.7	Visualization dashboard: Example of maximum values over time.	26
3.8	Visualization dashboard: Example of time series.	26
3.9	Rule extraction application.	28
3.10	Rule extraction application with rules from a sample training set.	28
3.11	Rule extraction application RFC costumizeable settings.	29
3.12	Data pipeline.	29
4.1	Typical temperature plot for the gravity casting process.	32
4.2	Similar sensors plotted together.	33
4.3	One day of processes, comparison between OK and NOK time series. . . .	35
4.4	Mean and standard deviation of the average temperature of a timeseries for OK and NOK time series.	36
4.5	Visualization of entire data including gross sensor errors.	37
4.6	Explained variance ratio depending on number of principal components. . .	38
4.7	PCA bases.	38
4.8	Time series projected into the 3 new PCA bases.	39
4.9	ICA bases.	40
4.10	Time series projected into the 3 new ICA bases.	40
4.11	Mean squared reconstruction error depending on number of components. .	41
4.12	Normalized NMF bases with PCA, ICA and random matrix initialization. .	42

4.13	Three physical factors for NMF initialization with amplitude = 1.	42
4.14	Normalized NMF bases with knowledge and random initialization.	43
4.15	NMF coefficients for the knowledge initialization.	44
4.16	Recall, Precision and their Geometric Mean of Rule Fit depending on the maximum depth of the Random Forest (KFold = 5).	45
4.17	Processes that comply with 2nd, 4th or neither rule.	47
4.18	Second NMF coefficients a week before and after the change was applied to the process.	49
4.19	Typical temperature plot for the high pressure casting process.	50
4.20	One day of processes, comparison between OK and NOK time series. . . .	51
4.21	Comparison between the mean temperature of OK and NOK time series. .	51
4.22	Time series of sensor 7 within the relevant time frame.	52
4.23	Mean squared reconstruction error depending on number of principal com- ponents.	53
4.24	PCA bases.	54
4.25	Time series projected into the 3 new PCA bases.	54
4.26	ICA bases.	55
4.27	Time series projected into the 3 new ICA bases.	55
4.28	Normalized NMF bases with knowledge and random initialization.	56
4.29	NMF coefficients for the knowledge initialization.	56
4.30	Recall, Precision and their Geometric Mean of Rule Fit depending on the maximum depth of the Random Forest (KFold = 5).	57
4.31	Processes that match rule 1, 3 or both.	59

List of Tables

3.1	Atomic data.	21
3.2	Disk space for each data type in PostgreSQL.	24
3.3	Disk space for each data type in PostgreSQL.	24
4.1	5-Fold Cross Validation average Recall for OK and NOK processes.	46
4.2	Gravity Casting: Scoring of the rules on the testing set.	47
4.3	5-Fold Cross Validation average Recall for OK and NOK processes.	58
4.4	High Pressure Casting: Scoring of the rules on the testing set.	58

Chapter 1

Introduction

1.1 Context

In the manufacturing and processing industries, there has been a push for higher quality in parts and the decrease of their rejection rates in production, all the while obeying increasingly restrictive market regulations. A proper strategy used to achieve these goals is fault detection and process monitoring.

Fault detection and process monitoring have seen significant growth over the years. Classically these problems were approached using Univariate and later Multivariate Control Charts. However, as manufacturing systems become more sophisticated, these approaches are not sufficient anymore since they fail to recognize the more complex relations in the data.

Meanwhile, the *Industrie 4.0* revolution [2] has provided the necessary tools to collect massive amounts of data of manufacturing processes in real-time. In the process monitoring setting this gave rise to history-based or data-driven methods for anomaly detection.

Various IT systems are used in manufacturing, such as for process control and product tracking. From these, a considerable amount of data is stored. The next step is then to transform the data into knowledge that can be used to optimize the processes.

We will be looking at data gathered during gravity and high pressure casting processes at the BMW plant in Landshut, and identify optimal settings for these processes. This data is gathered periodically when a part is produced in a furnace. This means that a time series is effectively built by the sensors over time, in particular: temperature data.

1.2 Motivation

The motivation behind the thesis is to reduce the complexity of analyzing raw data gathered by sensors in the respective machines. As of today, a process expert would only be able to look at the raw information, perhaps in the form of a time series, and try to extract knowledge from there. This extraction of knowledge is a challenging process since

much of the information is mixed or hidden behind a large amount of data that is gathered daily. Furthermore, manual analysis requires a sizable amount of time.

Additionally, there exists also an economic motivation: Being able to determine how a process can be optimized will incur in a lower scrap rate of parts, which in turn reduces costs.

1.3 Objective

This thesis' goal is to provide insight into the process that a human may find difficult to obtain alone. We will study algorithms and methods that can serve as an aid for the process workers.

To this end, we will study matrix factorization methods which extract lower-dimensional features from the data. In conjunction with this, we will also explore a supervised method that extracts rules according to labels.

With the aid of matrix factorization and rule creation, complex data can be reformulated to provide insight into a physical process, events related to the routine of the process or even correlation between extracted data and faulty produced pieces.

1.4 Related Work

This area, from producing clear rules to extracting interpretable information from data sets, has produced some interest in the research community and, consequently, has a substantial amount of research. The next paragraphs provide a short summary of some of the most relevant papers.

Holte devised a straightforward algorithm that obtains the best rule from a data set for a classification problem. Holte named this algorithm One Rule [3]. The paper demonstrated that simple rules perform well on commonly used data sets. However, the simple rules do not consider multiple features at once and only work for categorical features. As such, numeric features have to be categorized. It is most commonly used as a benchmarking tool.

Another algorithm that provides rules for a labelled data set is the Bayesian Rule Lists (Letham et al., 2015 5 [4]). It first extracts frequent values or a combination of frequent values in the data and then uses labels to learn rules for those features. However, it also suffers from the numerical feature shortcoming of One Rule.

The Isolation Forest method was proposed by Liu and Ting in 2008 [5] with the goal of identifying outliers in a data set. It uses an ensemble of trees to decide an anomalous score of a data point. This score bases itself on the assumption that anomalous points should be, on average, identified closer to the root of the trees. Even though it provides a way to identify anomalous data it does not offer any rules for the overall data.

Related to extracting rules from tree ensembles Sirikulviriyā and Naphaporn (2011) considered a method to integrate conflicting rules [6]. More recently (2019) a method

titled inTrees was proposed by Deng to extract, prune and select rules from tree ensembles [7].

Research over the decomposition of temperature time series into its known physical components with matrix decomposition techniques has been carried out by Weiderer (2019) [8].

Lastly, an extended abstract on the case study of rule extraction from NMF's component profiles for process optimization was presented in 2019 [9].

1.5 Thesis structure

After this introductory chapter, the thesis will be organized in a bottom-up fashion, first giving the reader all the essential information necessary on the methods, algorithms and industrial process, then moving on to the specifics of the application and its results.

Chapter 2 provides an overview of the algorithms and methods utilized throughout the thesis. After this chapter the reader should have an intuition over how the outlined methods and algorithms work.

Chapter 3 describes the whole process of transforming raw data in knowledge. The industrial processes from which data is gathered, the gathering and storage of the data, its visualization and rule creation are all described here.

Chapter 4 regards the application of the theoretical concepts in the industrial case as well as its results. Furthermore, it explains what insights were able to be extracted.

Finally, chapter 6 presents some of the conclusions achieved by this thesis and describes possible future work to improve the research.

Chapter 2

Methods and Algorithms

In the industrial setting, a potentially large amount of data can be gathered. This is why, one of the biggest goals and obstacles to data analysis in these cases is the gathering and organization of useful data. In the age of digitalization, the transition to digital techniques is still being done over time. As such there is still no defined way to deal with this. Some problems may appear due to the clash with how things used to be done, such as writing down information on paper. This creates big difficulties to then analyze this data in a efficient way. Backlogging takes too long and the proper way to move forward is to change the way things are stored to a digital process.

In this chapter, we will provide an overview of the methods and algorithms used in this thesis. The common theme for these methods is that they provide ways to be interpreted and, as such, can be used as an auxiliary tool for induction reasoning by a process expert.

2.1 Matrix Factorization in Industrial Data

A sizable portion of the industrial data comes in the form of time series: sensor measurements over time. These measurements can be the result of a mixture of several inter-related sources. If the time series is analyzed only by itself much of the information from the sources is lost.

The sensed data can be organized as a matrix, \mathbf{T} of dimension $M \times N$ with $M > N$, where each row is a time series. Each time series can be registered by different sensors or by one sensor in different phases of an industrial process. Matrix Factorization Techniques can, in these cases, separate the different sources, thus providing more transparent insight into measured processes while also reducing the dimension of the data efficiently.

There are different approaches, each following a different assumption, to separating the sources. Some of the most famous methods are:

- Singular Value Decomposition (SVD) or Principal Component Analysis (PCA) [10] which assumes orthogonality:

$$\mathbf{T} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{B}\mathbf{V}^T \quad (2.1)$$

Where \mathbf{U} and \mathbf{V} are orthogonal matrices and the inner dimension of $\mathbf{B}\mathbf{V}^T$ is K , where $K < N$

- Independent Component Analysis (ICA) [11] assumes independence. With $\mathbf{T}_1 = \mathbf{T}^T$, the decomposition of the $N \times M$ matrix is:

$$\mathbf{T}_1 = \mathbf{A}\mathbf{S} \quad (2.2)$$

Where the rows of \mathbf{S} are independent.

- Non-negative Matrix Factorization (NMF) [12] assumes non-negativity. It decomposes \mathbf{T} as:

$$\mathbf{T} = \mathbf{W}\mathbf{H} \quad (2.3)$$

Where the elements of both factors are all non-negative. The inner dimension K of the matrices is $< N$.

2.1.1 Singular Value Decomposition and Principal Component Analysis

Singular Value Decomposition (SVD) is a computational method that tries to find the uncorrelated sources of a mixed-signal. Many of the PCA algorithms use it in one way or another since it is often more accurate and efficient.

SVD has been dated back to several mathematicians, from the areas of linear algebra or integral equations. Eugenio Beltrami in 1873 [13] was the first to derive SVD's followed by C. Jordan in 1874 [14]. Much later in 1889 J. J. Sylvester [15] derived it independently as well and eventually Erhard Schmidt [16] in 1907.

Computational methods started being developed in 1954 [17]. The most used method was developed in 1970 as a variant of the Gene Golub and William Kahan method [18] by Golub and Christian Reinsch [19].

SVD decomposes a matrix with dimensions $M \times N$ into two unitary matrices \mathbf{U} and \mathbf{V}^T of dimensions $M \times M$ and $N \times N$ and a rectangular diagonal matrix $\mathbf{\Sigma}$ of singular values $M \times N$.

$$\mathbf{T} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (2.4)$$

Principal Component Analysis (PCA) was invented for the field of mechanics in 1901 by Karl Pearson [20] and later also created and named in a paper by Hotelling in 1933 [21]. Its goal is to reduce the number of coordinates of the original data to fewer uncorrelated coordinates while still retaining most of its variability.

Its components are determined iteratively by picking the unit vector whose direction maximizes the variance of the projected data. This means that the first component will explain the most variance of the original data, the second component will explain the second most variance and so forth, while being orthogonal to all previously picked vectors,

decorrelating the data. An example of two principal components defined by their vectors is found in figure 2.1.

The algorithm defines vector \mathbf{v}_1 by maximizing the variance of the data projected onto the vector, with the constraint $\|\mathbf{v}_1\| = 1$.

Consider \mathbf{C} , the matrix of the samples covariance around the mean:

$$\mathbf{C} = \frac{\mathbf{T}^\top \mathbf{T}}{n - 1} \quad (2.5)$$

Then the variance is calculated as:

$$variance = \mathbf{v}_1^\top \mathbf{C} \mathbf{v}_1 \quad (2.6)$$

Every other principal components is calculated in the same manner, except it must also follow the orthogonality restriction: $\mathbf{v}_i \perp \mathbf{v}_{i-1}, \dots, \mathbf{v}_1$.

This is essentially calculating the eigen decomposition of matrix \mathbf{C} and finding the eigen vectors which are the columns of \mathbf{V} :

$$\mathbf{C} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top \quad (2.7)$$

To clarify the relationship between PCA and SVD we substitute 2.4 into 2.5 and obtain:

$$\mathbf{C} = \frac{\mathbf{V} \mathbf{\Sigma} \mathbf{U}^\top \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top}{n - 1} \quad (2.8)$$

$$= \mathbf{V} \frac{\mathbf{\Sigma}^2}{n - 1} \mathbf{V}^{-1} \quad (2.9)$$

Finally, comparing 2.9 and 2.7, the relation between the singular values of SVD and eigen values of PCA is described as:

$$\frac{\mathbf{\Sigma}^2}{n - 1} = \mathbf{\Lambda} \quad (2.10)$$

2.1.2 Independent Component Analysis

ICA tries to find the independent components of a mixed-signal, for it to separate the sources they must be non-Gaussian.

The idea behind ICA was first described by Jeanny Hérault and Bernard Hans in 1984 [22] and compiled by Christian Jutten in 1985 [23]. In 1994 Pierre Comon [11] conclusively defined it.

One of its most common algorithms is the FastICA algorithm published by Aapo Hyvärinen and Erkki Oja in 2006 [24].

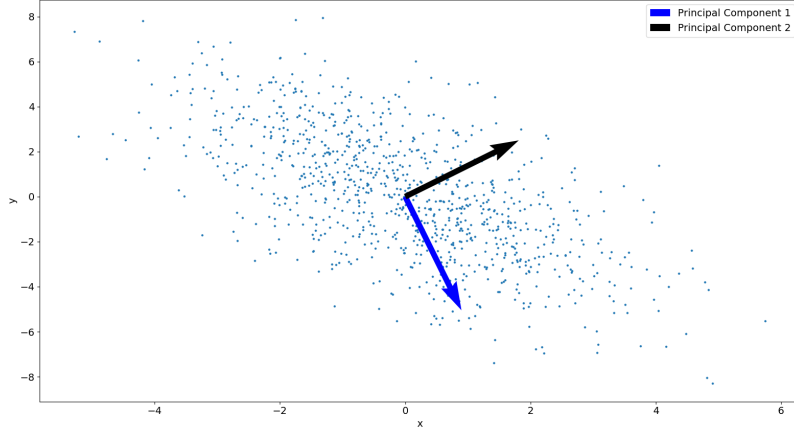


Figure 2.1: Illustrative example of two principal components.

ICA calculates \mathbf{B} , the unmixing matrix of the equation 2.11 where the data is now arranged in the columns: $\mathbf{T}_1 = \mathbf{T}^T$. Conversely, the mixing matrix \mathbf{A} relation to \mathbf{B} is shown in equations 2.12 through 2.15, where \mathbf{A}^* is the pseudo-inverse of \mathbf{A} .

$$\mathbf{S} = \mathbf{B}\mathbf{T}_1 \quad (2.11)$$

$$\mathbf{T}_1 = \mathbf{A}\mathbf{S} \quad (2.12)$$

$$\mathbf{A}^*\mathbf{X}_1 = \mathbf{A}^*\mathbf{A}\mathbf{S} \quad (2.13)$$

$$\mathbf{A}^*\mathbf{T}_1 = \mathbf{S} \quad (2.14)$$

$$\mathbf{A}^* = \mathbf{B} \quad (2.15)$$

While there are many ICA algorithms, most of them calculate \mathbf{B} , and consequently the mixing matrix \mathbf{A} in two steps:

1. First, the data is whitened. Whitening is also sometimes referred to as sphering the data and is a linear transformation such that the correlation between the signals is zero, and their variances are equal. This can be done through PCA using $\mathbf{\Lambda}^{-1/2}\mathbf{V}^T$
2. The second step is to rotate the matrix by optimizing a non-gaussianity cost function.

The independent component basis extracted from the ICA on the same data points as in figure 2.1 are shown in figure 2.2.

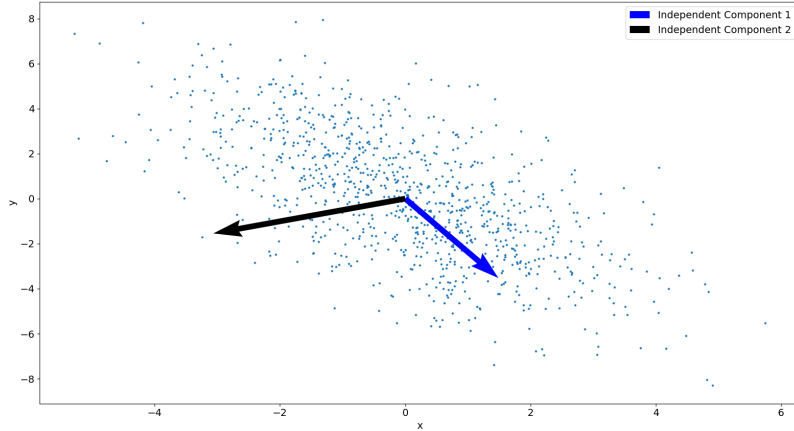


Figure 2.2: Illustrative example of two independent components.

2.1.3 Non-negative Matrix Factorization

NMF is especially interesting for non-negative data, which is commonly present in real-life applications. The non-negativity constraint often produces interpretable results. Some examples are its application in Astronomy and Astrophysics [25], Biology [26], Audio Signal Processing [27] and Image Processing [28].

This method is defined as the factorization of a matrix $\mathbf{T} \in \mathbb{R}^{M \times N}$ into matrix $\mathbf{W} \in \mathbb{R}^{M \times K}$ and $\mathbf{H} \in \mathbb{R}^{K \times N}$ losing the least amount of information possible while assuming non-negativity of both \mathbf{W} and \mathbf{H} . The \mathbf{H} matrix lines can be seen as the equivalent of the ICA's mixing matrices' (\mathbf{A}) columns and their coefficients found in matrix \mathbf{W} as seen in figure 2.3. In this case the lines of \mathbf{H} are the basis for the new representation \mathbf{W} .

$$\mathbf{T} \approx \mathbf{WH} \quad (2.16)$$

One way of addressing NMF's results lack of uniqueness in scaling and permutation is through the initialization of the right hand side matrices with different methods such as matrix \mathbf{T} 's principal components shifted to the first quadrant (using SVD) or even knowledge based on the underlying physical processes [8].

One of the first published articles describing NMF, by Paatero *et al.* published in 1994 [29], named it Positive Matrix Factorization and aimed at unmixing the main chemical components in a collection of chemical samples. Still, it was a paper published in 1999 [12] by D. Lee and H. Seung that sparked an interest for NMF in the research community. This paper factored human faces into face components such as nose, ears, and eyes in the columns of \mathbf{W} . These results brought to focus the interpretability of the results obtained by NMF.

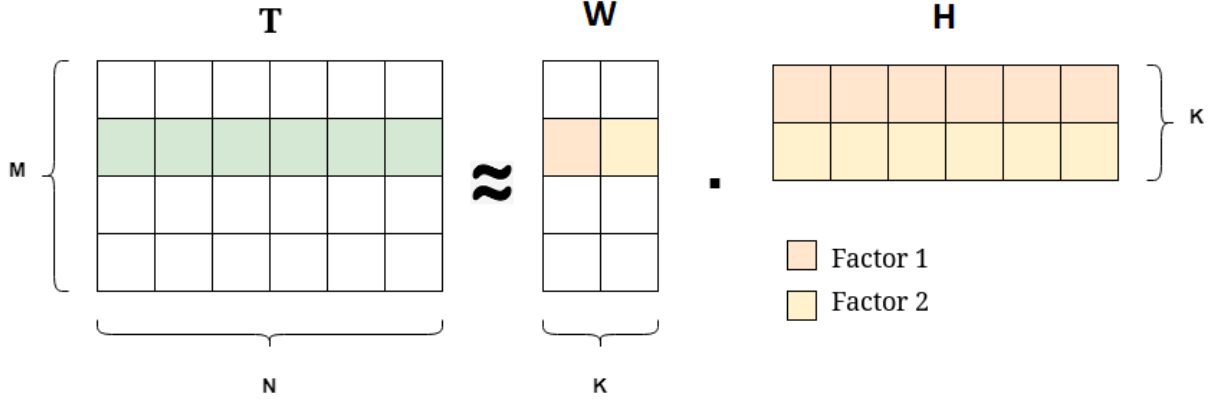


Figure 2.3: Illustration of the NMF where matrix \mathbf{W} indicates the coefficient of each factor defined by matrix \mathbf{H} .

Hierarchical Alternating Least Squares (HALS) is used to calculate both \mathbf{W} and \mathbf{H} . It differs from the normal Alternating Least Square approaches since it does not calculate the whole matrices \mathbf{W} and \mathbf{H} in each step but rather, respectively, individual columns and rows.

Let $\mathbf{T}^{(k)}$ be the residual matrix calculated as:

$$\mathbf{T}^{(k)} = \mathbf{T} - \mathbf{W}\mathbf{H} + \mathbf{w}_{*,k}\mathbf{h}_k \quad (2.17)$$

Where $\mathbf{w}_{*,k}$ is the k -th column of \mathbf{W} .

Then, HALS minimizes iteratively the following cost function, for a fixed \mathbf{h}_k and then for a fixed $\mathbf{w}_{*,k}$:

$$\frac{1}{2} \|\mathbf{T}^{(k)} - \mathbf{w}_{*,k}\mathbf{h}_k\|_F^2 \quad (2.18)$$

Where $\|\dots\|_F^2$ is the Frobenius Norm [30] of a matrix.

To update $\mathbf{w}_{*,k}$ and \mathbf{h}_k we calculate when the gradient of equation 2.18 (respectively in regards to the columns of \mathbf{W} and rows of \mathbf{H}) is zero.

$$\mathbf{w}_{*,k}\mathbf{h}_k\mathbf{h}_k^T - \mathbf{T}^{(k)}\mathbf{h}_k^T = 0 \quad (2.19)$$

$$\mathbf{w}_{*,k}^T\mathbf{w}_{*,k}\mathbf{h}_k - \mathbf{w}_{*,k}^T\mathbf{T}^{(k)} = 0 \quad (2.20)$$

From equation 2.19 and 2.20 we get the update steps:

$$\mathbf{w}_{*,k} \leftarrow \frac{1}{\mathbf{h}_k\mathbf{h}_k^T} \mathbf{T}^{(k)}\mathbf{h}_k^T \quad (2.21)$$

$$\mathbf{h}_k \leftarrow \frac{1}{\mathbf{w}_{*,k}^T\mathbf{w}_{*,k}} \mathbf{w}_{*,k}^T\mathbf{T}^{(k)} \quad (2.22)$$

2.2 Rule Fit in Industrial Data

While extracting different sources from a mixture is relevant and by itself already interesting for the goal of this thesis, it does not make use of any labels available to us. This is where a supervised learning method is useful. We can use the matrix factorization results and their respective labels with a supervised learning method. In this case Rule Fit is studied, a supervised learning algorithm that extracts human-readable rules from features and their respective labels.

Now the input variables for the supervised technique, denoted as \mathbf{X} , are the coefficients of the bases found with the unsupervised matrix factorization techniques. This is illustrated in figure 2.4.

$$\mathbf{X} = \mathbf{W} \quad (2.23)$$

Or

$$\mathbf{X} = \mathbf{S}^T \quad (2.24)$$

Matrix \mathbf{X} coupled with the respective labels are used by Rule Fit to induce rules for the process optimization.

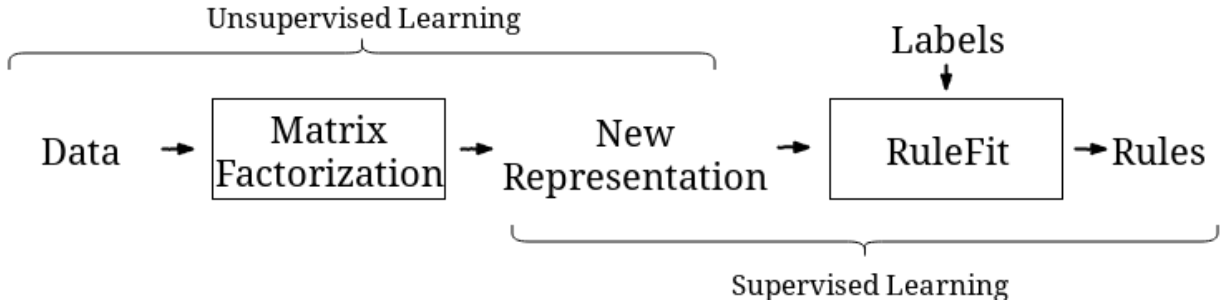


Figure 2.4: Illustration of the connection between supervised and unsupervised methods.

2.2.1 Rule Fit

Rule Fit is an algorithm by Friedman and Popescu published in 2008 [31] that learns sparse linear models from features and their relations in the form of rules.

This algorithm has the advantage of being interpretable like a linear regression model while still modelling relations between features.

There are two stages in the rule fit algorithm. First, it creates rules from the original features through a tree ensemble. To each node, with the exception of the root node, corresponds a rule. A rule is defined by the conjunction of the several conditions that are

$$r_3(\mathbf{x}_i) = I(x_{i3} > 10) \cdot I(x_{i1} \leq 1)$$

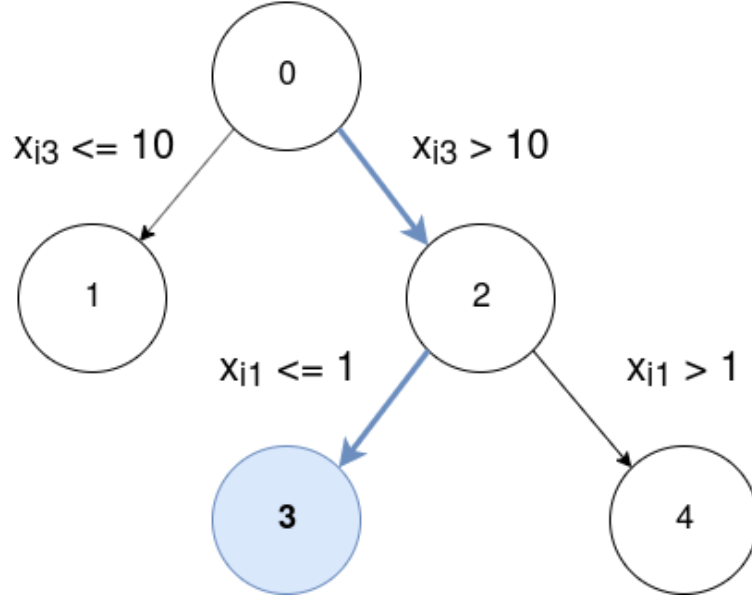


Figure 2.5: Illustration for the definition of a rule for a specific node.

used to split the tree down to the respective node. Let matrix \mathbf{X} be the new representation of the original data (either \mathbf{W} or \mathbf{S}^T):

$$r_k(\mathbf{x}_i) = \prod_{j=1}^n I(x_{ij} \in s_{jk}) \quad (2.25)$$

Where $I(\cdot)$ indicates the truth of the argument, n the dimension of the data and s_{jk} a subset of the set of all possible values for the input variable x_{*j} . This means that a rule is a binary feature which is one if and only if every condition in the rule is true for a certain \mathbf{x}_i . As an example, in figure 2.5 the rule defined in node 3 is only true when $x_{i3} > 10$ and $x_{i1} \leq -1$.

The second step is to calculate the sparse linear model using the original features and rules. Let $l_j(t_{ij})$ be the winsorization [32] of a feature, a_k the weight of the k -th rule, b_j the weight of the winsorized and normalized j -th feature:

$$F(\mathbf{x}_i, \mathbf{t}_i) = a_0 + \sum_{k=1}^K a_k r_k(\mathbf{x}_i) + \sum_{j=1}^n b_j l_j(t_{ij}) \quad (2.26)$$

These weights (a_k and b_j) and are calculated by minimizing a loss function with a LASSO [33] constraint which forces many weights to zero. Let \mathbf{y} be the vector of labels

and $\tilde{y}(\mathbf{x}_i)$ its estimation:

$$\min \left(\sum_{i=1}^N L(\tilde{y}(\mathbf{x}_i, \mathbf{t}_i), y_i) + \lambda \left(\sum_{k=1}^K |a_k| + \sum_{j=1}^n |b_j| \right) \right). \quad (2.27)$$

Finally, one can also calculate the importance of each rule and feature in the data set. For the original features this importance is calculated as in equation 2.28 and for the rules as in equation 2.29

$$I_j = |b_j| \cdot \text{std}(l_j(x_{ij})) \quad (2.28)$$

$$I_k = |a_k| \cdot \sqrt{s_k(1 - s_k)} \quad (2.29)$$

Where s_k is the support of the rule in the training data:

$$s_k = \frac{1}{N} \sum_{i=1}^N r_k(\mathbf{x}_i) \quad (2.30)$$

2.2.2 Logistic Regression

Due to the binary classification nature of our problem the Logistic Regression algorithm is used to calculate the sparse linear model of RuleFit.

Logistic Regression calculates the parameters, \mathbf{a} and \mathbf{b} in equation 2.26, of a generalized linear model function which measures the probability of a given input belonging to one class of a binary classification [34]

$$\tilde{y}(\mathbf{x}_i, \mathbf{t}_i) = \frac{1}{1 + e^{-\tilde{y}(\mathbf{x}_i, \mathbf{t}_i)}} \quad (2.31)$$

In this algorithm the positive label takes the value 1 and the negative label -1 . Since y_i is binary ($y_i \in \{1, -1\}$), the probability of the given input belonging to the opposite class is equal to $1 - \tilde{y}(\mathbf{x}_i, \mathbf{t}_i)$.

The loss function, used in equation 2.27, is [35]:

$$L(\tilde{y}(\mathbf{x}_i, \mathbf{t}_i), y_i) = \log(1 + e^{-y_i \tilde{y}(\mathbf{x}_i, \mathbf{t}_i)}) \quad (2.32)$$

2.2.3 Random Forest Classifier

While in this chapter a single tree was shown for illustrative purposes in practice an ensemble of trees is used to extract rules. For the ensemble learning method used in the first stage of the Rule Fit algorithm, we use a Random Forest Classifier (RFC) considering that our labels are binary.

This method was first introduced by Ho in 1995 [36] and further enhanced by his proposal in 1998 of selecting a subspace of features to be considered at each decision node

[37]. Other improvements were designed by Amit and German in 1997 [38] who proposed to select a random subspace of the data to decide on each node and Dietterich in 2000 who explored the randomization of the internal decisions at each node [39].

A RFC uses an ensemble of decision trees trained with the bagging technique (sampling with replacement) [40], to reduce the overfitting and high variance that a single tree suffers from. Its classification is decided by majority vote of the trees. Let u be the number of trees in the Random Forest then its classification is defined as:

$$C(\mathbf{x}_i) = \text{Majority}(t_1(\mathbf{x}_i), \dots, t_u(\mathbf{x}_i)) \quad (2.33)$$

While tree ensemble classifiers are powerful in classification tasks and one of the most famous ensemble methods today, its results are difficult to interpret due to the combination of a significant amount of decisions being made throughout each tree and a large amount of trees.

2.3 Summary

Each method described in this chapter has the common trait of being interpretable.

The matrix factorization methods provide manners to separate the several sources that contribute to the original time series. This separation gives us new bases and their respective coefficients for each time series. Both bases and coefficients can be visualized to create insight over the industrial process.

Alongside, Rule Fit uses labelled data to create rules. These rules model the relation between features and labels while providing interpretability.

Finally, matrix factorization methods, coupled with RuleFit can be used to extract new knowledge from the data in the form of rules. Consequently, a new understanding of how the process can be optimized to reduce its scrap rate can be achieved.

Chapter 3

Data Pipeline Tools

While we can consider that we are currently living the era of digitalization, the reality in industrial environments that precede it, is that the transition from outdated methods of data gathering, storage and analysis, such as writing information on paper or even analysis through simple excel sheets is underway and requires a significant period of time.

Methodologies to the digitalization of manufacturing have been an active research topic. The lifecycle of manufacturing data and its paradigm change due to big data [41], its challenges in the industrial automation [42] and its use in cost reduction and intuitive fault tracing [43] have been explored in research.

In this chapter the tools used for several steps in the data lifecycle, from its gathering to its transformation to knowledge, are described.

1. To start the chapter the process where the data is captured is defined. An overall understanding of the processes is given as well as what can be running in the background that could influence it.
2. Following, the data acquisition is described: Which tools capture the information and how it is gathered.
3. Subsequently, the approach to the data storage in this industrial setting is defined while also giving an insight to the reader on how the data is organized.
4. Afterward, the data exploration and analysis will begin for a couple of industrial cases. Generally, it is a good idea to visualize the data that will be worked on before applying any machine learning methods to it. The first step for every case will be to look at the data, learn any useful information one might find and prepare it.
5. Next, we explain how we choose the features to extract and how we extract them.
6. Finally, the process of creating rules from features is explained.

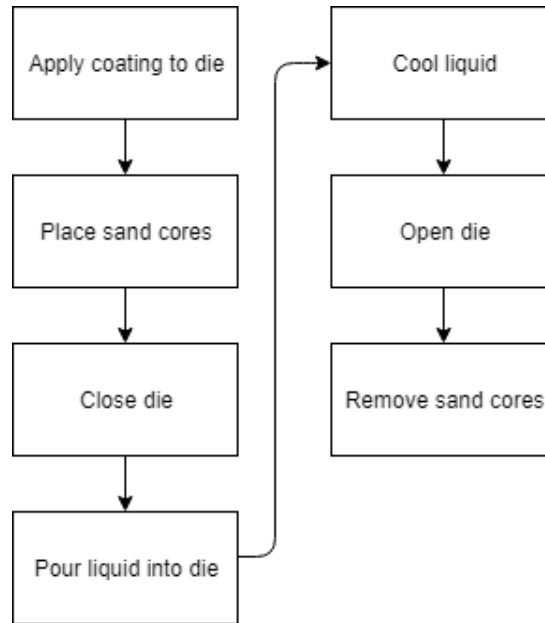


Figure 3.1: Gravity casting work steps.

3.1 Casting processes

Casting processes are not always stable; a considerable percentage of parts are discarded due to a defect being detected in them. While the defects can be detected at the end of the pipeline, it is normally difficult to understand what the source of the issue is due to the complex interrelations in the process. This is why more sophisticated methods of analysis are required to reduce the scrap rates.

In this section, the casting processes that are the focus of the analysis in the thesis will be looked at as well as its surrounding elements such as the pipeline wherein they lie and how they interact with it.

In the BMW Landshut plant's light metal foundry, there are four different kinds of casting processes: gravity casting, low-pressure die casting, high-pressure die casting, and Styrofoam casting. The processes studied in this thesis are the gravity casting of engines' cylinder heads and the high pressure-casting of tailgates.

3.1.1 Gravity Casting

Gravity casting, while relatively slow, is a good compromise when creating parts with complex geometry as the process itself can cast thinner walls. Additionally, it incurs in less trapped gas inside the die which results in higher integrity of the parts.

To produce any part with gravity casting, some procedures happen before and after the casting itself. See figure 3.1

Firstly the die should consist of two separate halves, which are then heated and sprayed

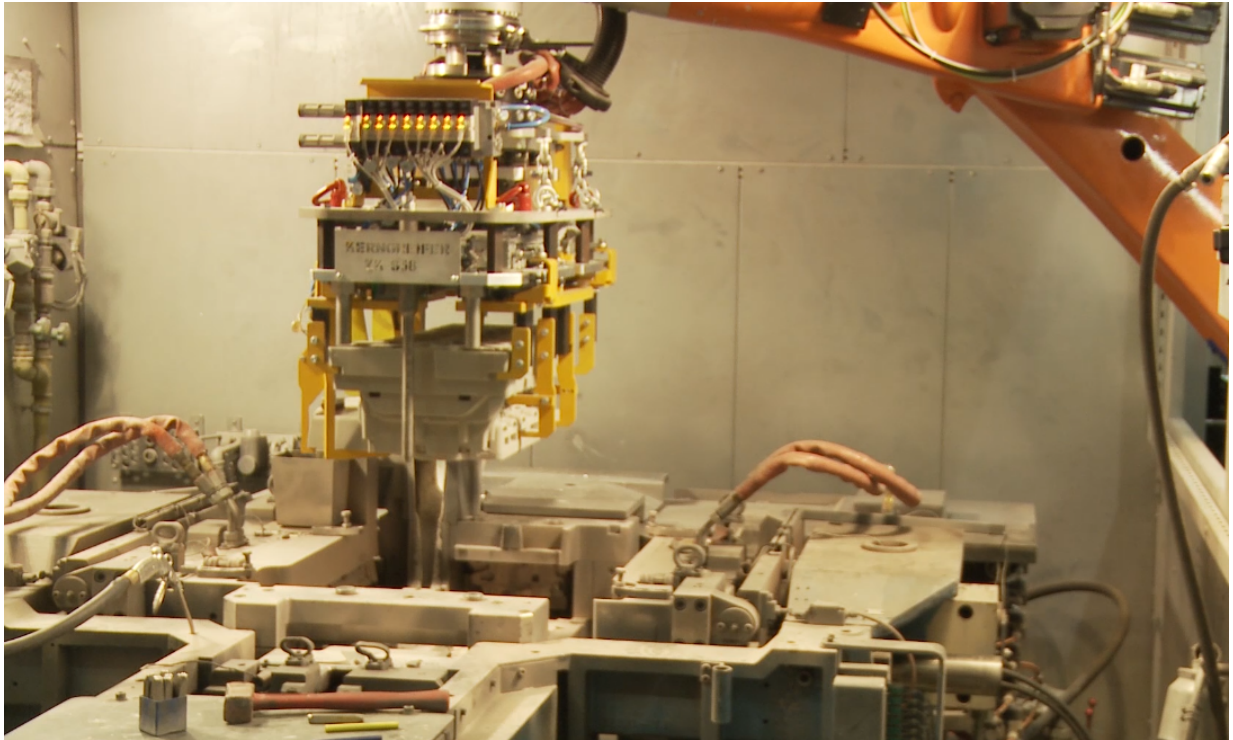


Figure 3.2: Placement of the sand core.

with a coating before being pressed together. This step exists to prevent premature solidification as well as to ease the process of the cast removal.

Afterwards, since the part geometry is complex enough that it cannot be made using only a die, sand cores are placed within the die, as illustrated in figure 3.2. These sand cores consist of the parts' negative geometry as sand and are glued together by adhesive, resin, and binder.

The next step is to clamp the die together and subsequently pour the liquid aluminum alloy into its cavity as seen in figure 3.3.

Eventually, the liquid cools down with the aid of a cooling channel that wraps around the die. This cooling channel contains running low-temperature water or air. When the part solidifies, the dies are unclamped, and there remain only the produced part and the sand cores.

Lastly, the sand cores are hammered by a machine and then shaken out of the part.

3.1.2 High Pressure Casting

High-pressure casting is much faster than gravity casting, and the process has a high level of automation. It also has lower costs when compared to gravity casting. The critical difference between high pressure and gravity casting is that the liquid metal is injected into the mould by a piston, as pictured in figure 3.4.

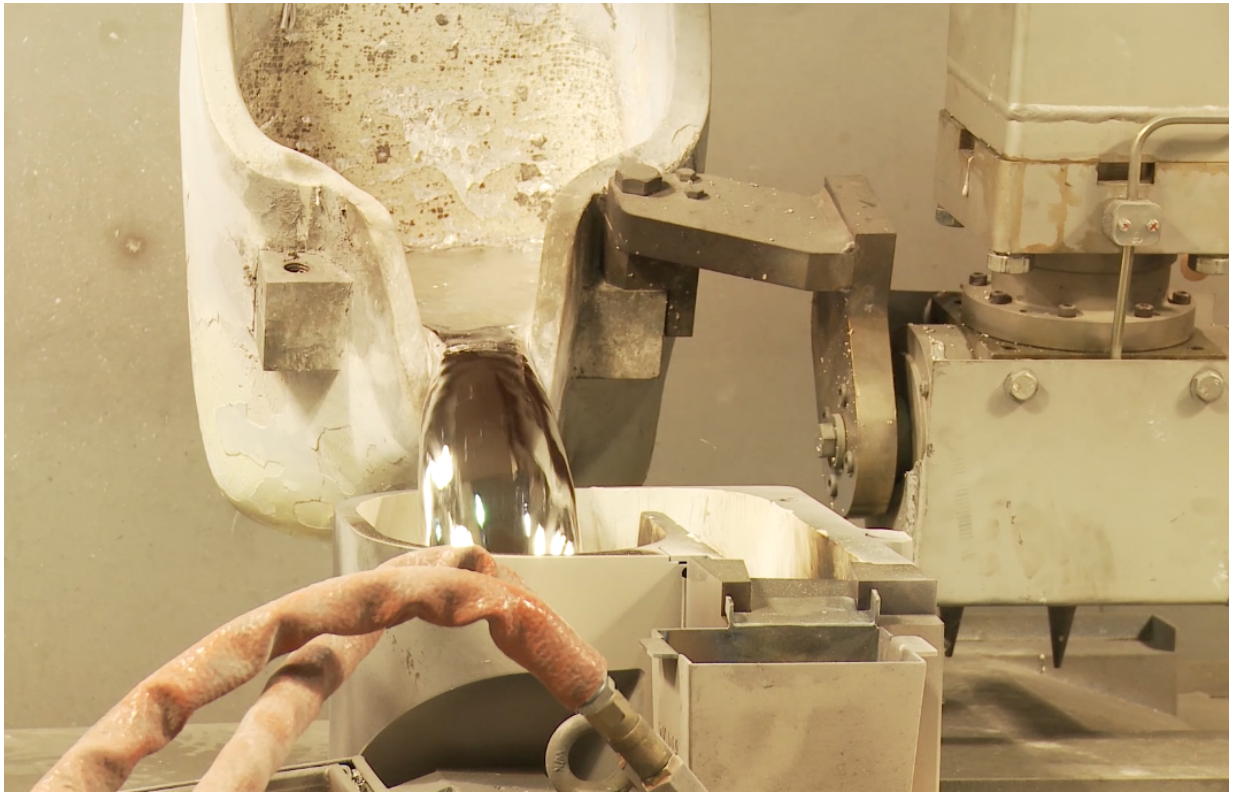


Figure 3.3: Pouring of the liquid aluminum.

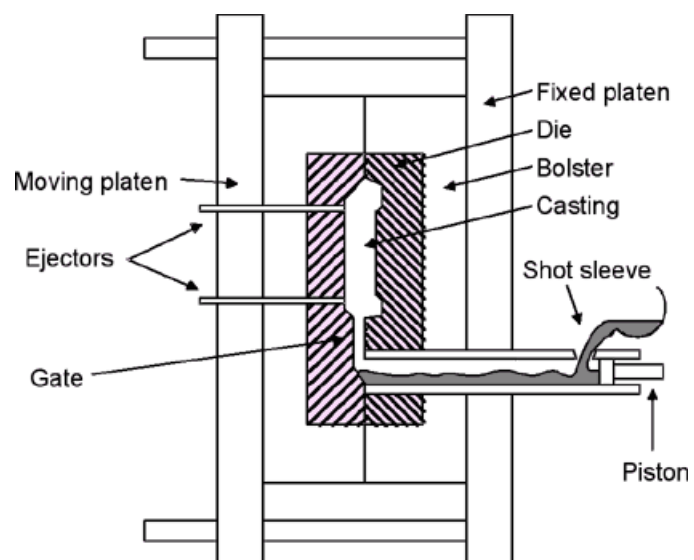


Figure 3.4: High pressure casting schematic. [1]

High-pressure casting has some fundamental differences when compared to gravity casting. However, the process overview is still very similar.

As with gravity casting, the first step should be to apply a coating to the two halves of the die. The halves are fixed to two platens, one of which is movable and the other fixed.

When the die halves are properly lubricated they are clamped together with very high-pressure, this pressure can range from 550 to 5600 tones.

There are two types of high-pressure casting: a hot chamber and a cold chamber. While in the hot chamber process, the piston is placed in the tank with the liquid metal, in the cold chamber process, the piston is located before a shot sleeve, which is only filled with liquid metal when the process starts. Only the cold chamber process is described and analyzed in this thesis.

After the clamping of the die the metal can be poured from a tank to the shot sleeve. This is done automatically. When the metal is poured, the piston will then push it into the die. The piston movement is separated by a "switching point", the point where the system that controls its movement switches from speed to pressure.

Finally when the metal is cooled the die unclamps, and an ejection mechanism pushes the casting out.

3.1.3 Process Chain

For the casting process, there needs to be a surrounding environment that provides it with the necessary materials and tools as well as its maintenance.

The process does not happen in a vacuum, and consequently, the readings are affected by many factors, some of which are:

- Mould (die);
- Sensors;
- Liquid material for casting;
- Heating and cooling pipes;
- Environmental influences.

Moulds, sensors, heating, and cooling pipes are all made by the tool construction department as well as maintained or even replaced. This means that any of these parts may change during the period of the data we collect and analyze.

The liquid material is mostly delivered in liquid form by an external supplier. It can also be in the solid-state and melted in the plant. This is done by the smelting department. The smelting department also checks the quality of the material it delivers using, for example, density measurements. As we have no way of checking ourselves whether this mass is within quality standards or not, we will, throughout this thesis, assume that the process chain surrounding the casting itself incurs in no defects.

3.2 Sensors

The temperature data is collected during the processes by thermocouple sensors. Thermocouples produce a voltage depending on the temperature and use this to calculate the temperature itself.

For them to be placed inside the mould, the tool construction department creates boreholes as close to the surface as possible without them affecting the process itself, the placement of these boreholes is also the responsibility of the tool construction team. For the high-pressure casting, these have to be placed in direct contact with the liquid since the process is so short that any significant delay on the propagation of heat to the sensor means that the sensors' data will not give any useful information on the process. On the other hand, gravity casting sensors are not directly in contact with the liquid, on account of the duration of the process being long enough that the propagation delay is not an issue. Once again we do not have access to this data, but we will assume that they are correctly placed.

3.3 Data

The sensors collect data throughout each process. This data comes in the form of the value measured coupled with its timestamp. This data is collected when the process starts and then periodically, depending on the sensor. The whole process takes one hundred and sixty (160) seconds, and measurements are taken every five (5) seconds in this gravity casting processes and, therefore, we have a total of thirty-three (33) measurements per process. The high-pressure casting, however, is of much shorter duration and only lasts around 12 seconds. Nonetheless, we have a much higher frequency of measurements. During the process, the sensors capture three thousand (3000) values. This means that approximately every 4 ms, there is a new recorded value.

Furthermore, before the parts are handed over to the engine works they are quality checked.

During the casting process, the parts can be marked as NOK by the human operators if there are visible defects. However, the analysed parts are mainly quality checked using computer tomography. If a part is deemed suitable by this tomography, then the cylinder head will be tagged as OK and automatically move on to the next step (engine works). Parts marked as NOK are all then sent to a process expert who will analyze the tomography images and verify if the scan produced a false negative (i.e.: Falsely detected anomaly) in which case the part also moves on to the next step. Otherwise, the expert will determine the type of defect.

Since a false positive (a part marked as OK being defective) is the worst case and to be avoided, the computer tomography is calibrated so that it produces almost no false positives as it would be much more costly to have a bad part marked as OK than to mark some suitable parts as NOK. This means, however, that it produces more false negatives, which the process expert must over-rule.

N x	DMCode	Timestamp	Measurement
1 x	DMCode	Label	

Table 3.1: Atomic data.

Both the sensor data and the quality check data are tagged with the produced part unique identifier (DMCode), this allows us to join the information from the sensors and the quality of the part produced.

So far, to summarize: The data is gathered from the process, either gravity casting or high pressure casting. While both kinds of processes differ in the number and frequency of data both come in the form of tuples. This tuple contains a unique identifier, a timestamp and a temperature measurement. Furthermore, a label regarding the quality of a part is later attached to each identifier when the part has been marked as OK by the computer tomography or, otherwise, when it has been manually checked. The gathered data, in its atomic form is then as defined in the following table 3.3, for each part we have N tuples of the measurement type and one of the status type.

3.4 Gathering Data

Obtaining data in an industrial environment commonly consists of gathering information from several sensors.

For each casting area, numerous sensors are measuring the processes. Every sensor captures a physical attribute periodically, p.e.: temperature, pressure or humidity, and transforms this data into an electrical signal. These electrical signals are captured and interpreted by a programmable logic controller (PLC) located near the casting place. This controller attaches this information to other data such as an identifier of the produced part (DMCode) and its timestamp. It is from this association that timeseries can be constructed for each produced part.

The controllers are also an Open Platform Communications (OPC) Server. OPC is maintained by the *OPC Foundation* and defines standards and standards for communication with a PLC, it serves as an interface so that read/write requests to any PLC are device and vendor agnostic.

An OPC client periodically requests data from the server, processes it and stores it in a database, this will be described in the next section.

Furthermore, quality data is logged in a relational database for each piece, either automatically in case no defect is detected by the computer tomography or manually after it fails this check, and a process expert has to verify the piece for any false negatives. However, this means that while the information of the process itself might be available, the corresponding quality data will be available only after a relatively long period.

There are essentially two different courses that create the information, one comes from the sensors to the PLC server and the other is the assignment of labels to each part

produced, this is illustrated in figure 3.5.

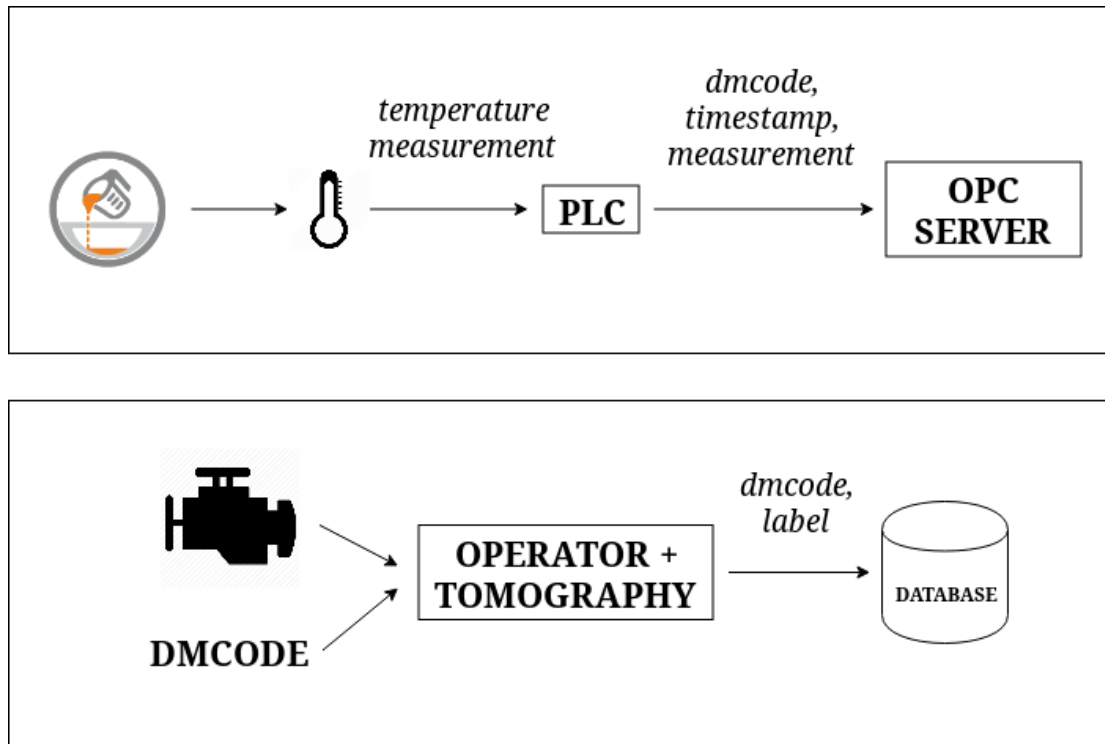


Figure 3.5: The two different information courses.

3.5 Storing Data

After obtaining the data tuples in the OPC server, the next step should be to store it. Otherwise, the data is only contained in volatile-memory and, therefore, no historical data can be accessed as there is no data persistence. This means an OPC client retrieves data from the server and then stores it in a database. This is done through KEPServerEX. It is a platform for communication, management, control and monitoring of automation devices. It supports the OPC Unified Architecture. It can serve as a client and can store the information in any ODBC-compliant database.

There are several ways data can be stored. One must decide the most desirable data model. There are several options.

One of the simplest models is a key-value database. As the name implies, it can only store data in the form of a key and value pair. It is able to retrieve a value given its corresponding key.

The classical approach is the relational model. In this model, data is stored in the form of tables where each row represents an entry of data and each column an attribute. Every column has a well-defined data type. Relations between data are represented by a reference to another row.

Wide-column databases are similar to the relational model in the sense that they also store data in tables. However, the number and types of columns are not well-defined, for any row it may be different.

Object or NoSQL data model is more loosely structured when compared to the relational model. Data is stored in the form of a JSON data blob. This kind of data model is a good fit for when the structure of the data is unknown or varies greatly.

With GraphDBs, data is stored as a node, information on the node comes as a set of nestable properties and the relation between data are edges that connect nodes. This type of data model is useful, for example, in transaction-based data. There are also some databases that are optimized for time-stamped data, usually known as Timeseries Databases, they are used to store values over long periods of time. They excel in time stamp storage as well as handling queries that involve time.

The data that is required is more complex than a simple key-value pair. For example, in a given process, it is necessary to know its state as well as its measurements. Furthermore, the structure of the data is well-defined. Therefore, there is no need for a NoSQL or wide-column database as it would only increase the difficulty in managing and processing the information. Both relational and time-series databases are a good match for the data storage. While the data is well adapted to a timeseries database, in practice, a relational database made more sense. Namely *PostgreSQL*, an open-source relational-database. Most of the work being done used relational databases already, choosing a relational database allows for an easier integration with the surrounding work. Added to this, the database must be able to operate with KEPServerEX. Furthermore, a relational database can also be used to store time-series (although not as efficiently) and has access to the full SQL support. A possible concern in regards to the relational database is the size it requires on storage. This is addressed later in this section after the database entities and relations are defined.

To decide on the structure for the database it is useful to know its use-cases.

In the interest of this thesis some of the use cases are:

- Visualizing a unique time series;
- Filtering by industrial process;
- Filtering information by sensor;
- Filtering information by sensor type;
- Filtering information by date;
- Obtaining time series from measurements;
- Obtaining status/quality data of a process.

The majoring entities are "processes". During every process one part is produced. This process has a starting date, the identification of the produced part as well as the quality

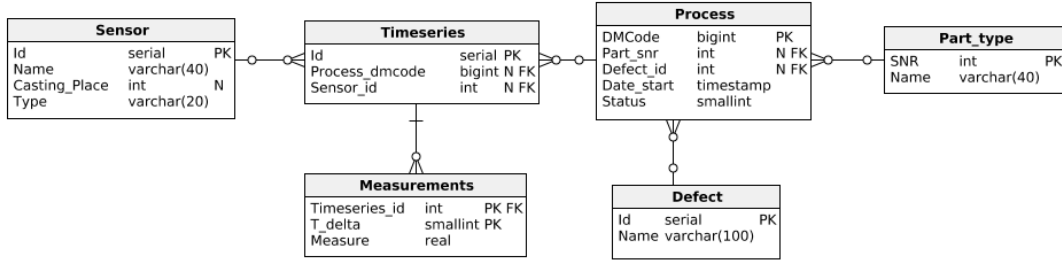


Figure 3.6: Entity relationship diagram of the relational database.

Type x	Disk Space (bytes)
smallint	2
int	4
real	4
serial	4
timestamp	8
bigint	8

Table 3.2: Disk space for each data type in PostgreSQL.

data for this part. Every process has several "time series", one for each sensor that is measuring the process. A time series has N pairs of values, an integer that indicates the order of the measurement coupled with the value that was recorded at the time.

The devised entity-relationship diagram can be visualized in figure 3.6. Regarding the disk space required for storage, the calculation can be done. The *Sensor*, *Part_type* and *Defect* table will all have a negligible amount of rows and therefore occupy an inconsequential amount of disk space. The space required for each type of data in *PostgreSQL* database is defined in table 3.5.

From this, one row for the *Process*, *Timeseries* and *Measurements* tables will occupy a defined amount of space. For a given process, the disk space used is, in bytes:

$$gravity\ casting\ process = 26 + 16 * n_1 * (1 + 33) \quad (3.1)$$

Table	Disk Space for a Row (bytes)
Process	26
Timeseries	16
Measurements	10

Table 3.3: Disk space for each data type in PostgreSQL.

$$high\ pressure\ process = 26 + 16 * n_2 * (1 + 3000) \quad (3.2)$$

Where n_1 is the number of sensors in the gravity casting mould and n_2 in the high pressure casting mould. Finally, to extrapolate the average amount of disk space necessary per time frame an average of the number of processes is done for the respective time frame and multiplied by the respective disk space used per process. Unfortunately, this information cannot be disclosed. However, the amount of space required for one year is in the order of magnitude of tens of Gigabytes and therefore not an issue.

Lastly, it is also interesting for the analysis to have a SQL view that aggregates all the useful information in one single table. It should include the identifier for the part as well as its status, the date for the process start, the identification of the sensor that collected it, the measurement order as well as its value:

DMCode	Status	Date_start	Sensor.id	T_delta	Measure
--------	--------	------------	-----------	---------	---------

3.6 Data Visualization and Preparation

Before any machine learning can be applied the data should be visualized.

Visualization is a handy tool. It allows us to understand a large amount of data that would otherwise be impossible to analyze. It provides intuition for the data, reveals apparent relations or correlations, and exposes outliers or erroneous data points. It is also possible to identify patterns over time as well as relationships between data, for example, how an OK part compares to a NOK part. Lastly, it is also useful to determine if the captured data has at some point errors or inconsistencies. For example, in figure 3.7 it is easy to ascertain that from June 12 to June 15 the data gathered is anomalous since its max value is 0, furthermore clusters are formed some of which can signify badly recorded data.

To this end an application was developed using *python3* and *dash by plotly* that, connected to the database provides a visualization for singular or multiple time series as well as trends over time, such as the average of the time series or its starting values over time. This application allows us to:

- Visualize the timeseries recorded during the production of a specific part (by DM-Code);
- Overlap the timeseries with the plunger position, to locate the measurements relatively to the casting phases;
- Visualize statistics of the time series over time (figure 3.7);
- Visualize several time series at once, allowing comparison (figure 3.8).

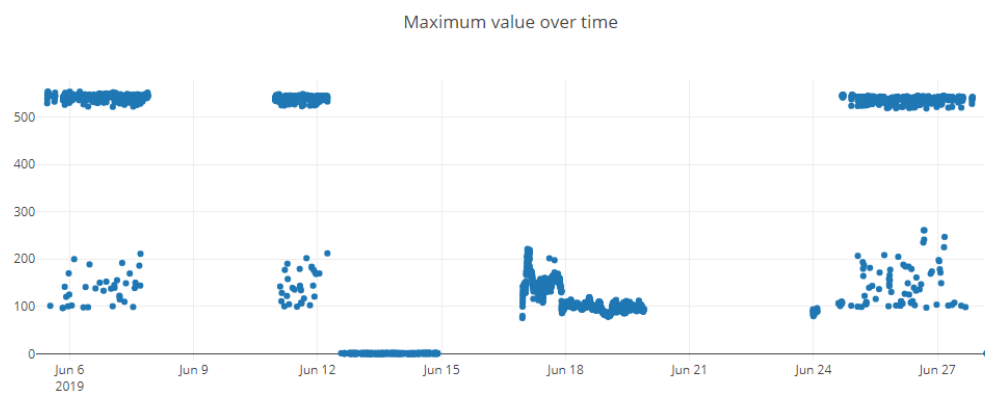
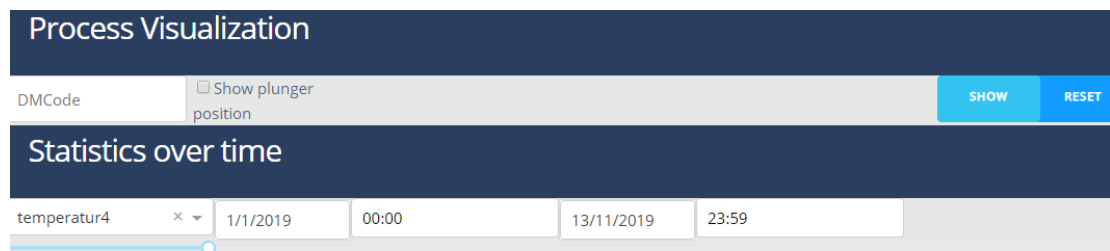


Figure 3.7: Visualization dashboard: Example of maximum values over time.

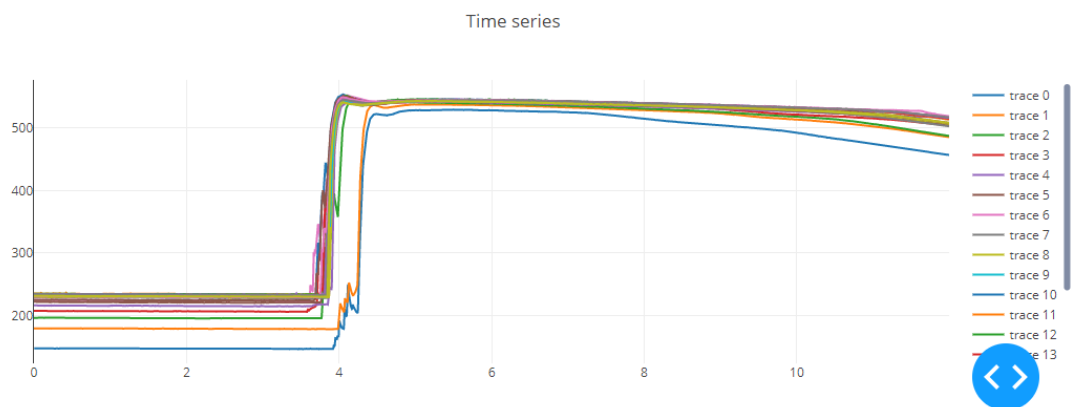


Figure 3.8: Visualization dashboard: Example of time series.

After visualizing the data, one should understand what outliers look like and if any measurements have gross errors. Once anomalous data is detected its handling can be decided, if the amount of anomalous data is not very significant it can be removed. Otherwise data correction can be applied.

This step also helps in deciding what the appropriate algorithms are for the data. For example, since the data collected during these studies are purely positive, we may choose to apply the Non-Negative Matrix Factorization technique to separate its sources.

3.7 Rule Creation

The next step is to extract different features for each time series to be later used for analysis and classification.

The raw data are time series. They are all arranged in a matrix (T) of size $m \times n$, where m is the number of processes, and n the number of measurements per process. The processes are ordered in the matrix by their dates in an ascending fashion.

Since the goal is to define conditions under which the process is optimal every feature we extract must be interpretable so that a process expert can understand the rules and be able to change parameters in the industrial process to uphold them.

We use the NMF matrix decomposition technique, which was overviewed in chapter 2. The library used for this decomposition was the python3 *sklearn* [44] to extract features that are interpretable by using an initialization based on knowledge of the underlying physical processes. Moreover, we will also compare the extracted features with other known matrix decomposition techniques: PCA and ICA, also from the same library. While these are not easy to interpret, they may serve as a base for comparison to the NMF results.

Finally, to extract rules from features a small application was developed. This application takes as input either an excel file (commonly used by the process experts), or the database with both features and labels. The input methods and simple usability were defined as to be readily usable by a process expert without the need to change their workflow. This application then transforms the features and labels in rules that are immediately readable by the user coupled with its FDR and Recall.

It also allows the user to control Rules are created with the Rule Fit algorithm [31]. These rules define thresholds within which the process can be optimal. If every condition is upheld, the value of the rule is 1, associated to an OK process, otherwise a 0 representing a NOK process.

First, a random forest classifier, from the *sklearn* library, using the bootstrapping technique is trained with the extracted features as input samples, and status as target classes. The positive class represents the OK status while the negative class represents the NOK status indicating that the part produced was scrapped.

All rules from this forest are then extracted: A rule is the logical conjunction of every split decision in a tree that is in the path from a node to the root. This means that if a node is situated at depth N , the final rule will be the conjunction of N decisions. However, we use one technique proposed by Sirikulviriyaya and Naphaporn [6] to remove any redundant



Figure 3.9: Rule extraction application.

Importance	Rule	FDR	Recall
6.9513249475	feature 1 > 399.69171142578125 & feature 2 <= 172.29168701171875	0.01	0.21
6.7786314140	feature 3 > 204.9809112548828 & feature 1 <= 404.54026794433594	0.01	0.16
6.2987557101	feature 3 <= 238.38642120361328 & feature 1 > 398.41954040527344	0.02	0.19
5.9830610843	feature 1 > 410.386474609375 & feature 2 <= 191.62733459472656	0.01	0.27
5.9514602527	feature 2 > 219.43950653076172 & feature 3 > 247.50362396240234	0.0	0.14

Figure 3.10: Rule extraction application with rules from a sample training set.

conditions within a rule. Therefore a rule will be the conjunction of at most N decisions.

Once all rules are extracted from the random forest, they are used as the new features for a generalized linear model, a LASSO logistic regression [33], where the targets are once again the status, which means that due to the binary classification nature of the problem in these studies our loss function uses a logistic function of the kind $y = (1/1 + \exp(-F(\mathbf{x})))$.

From here the importance 2.29 of each rule is calculated using the respective coefficient of the logistic regression and the support of the rule in the training data.

While the Rule Fit algorithm could be used to predict defects, we are interested in exploring the most useful rules extracted from the ensemble classifier. These will then serve as supporting guidelines for the process expert to make decisions on the process. The rule creation algorithm extracts rules that would give a true value when applied to features of the non-defective process. These rules will then serve as supporting guidelines for the process expert to make decisions on the process. Therefore the ordering of rules by its importance (defined in equation 2.29 of the second chapter) gives an indication of how to select the most relevant.

The application abstracts all the complexities requiring just the features and labels as input (see figure 3.9 and figure 3.10), while still allowing to change the maximum depth permitted to control overfitting and the number of trees in the ensemble (figure 3.11).

3.8 Summary

Throughout this chapter we discuss what was done to gather and process data acquired. Several human-oriented interfaces were created to allow the understanding of the data at each step.

In summary: First, the raw data is obtained from the industrial machines and processed

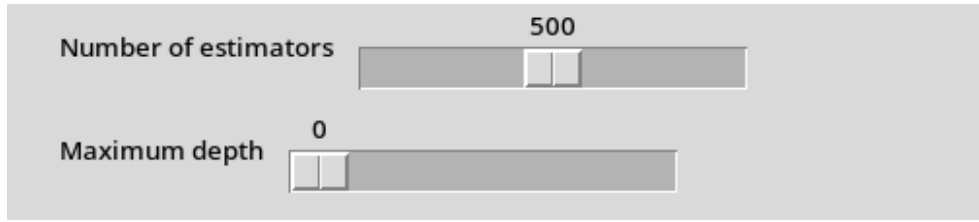


Figure 3.11: Rule extraction application RFC customizable settings.

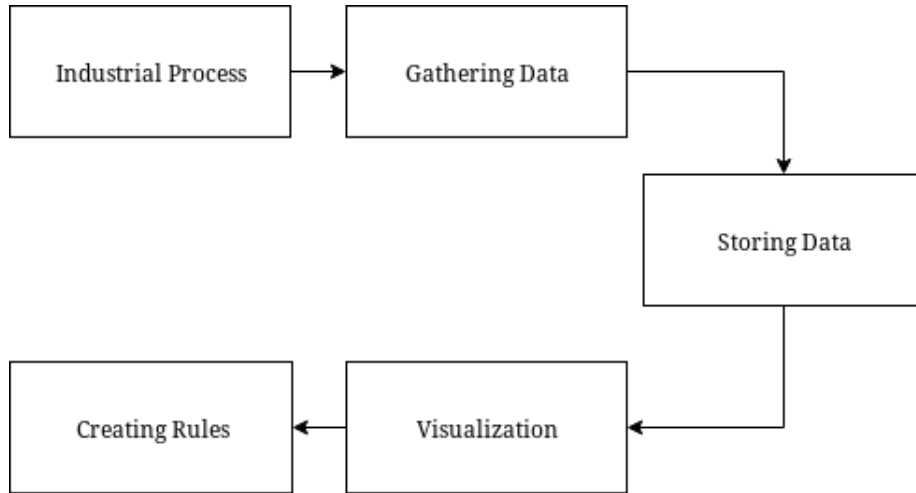


Figure 3.12: Data pipeline.

to be stored in a relational database as time series and their labels along with all other necessary information. After this, the time series are visualized to provide initial intuition and knowledge of the data and pre-processing it accordingly. Next, the time series are decomposed using blind source separation methods, and from these new extracted features, rules are created and evaluated according to scores that are important for this specific case. Figure 3.12 illustrates the main steps of the pipeline.

Chapter 4

Industrial Case Study

4.1 Overview

In this chapter, we will apply our methods and extract results in different case studies pertaining to the foundry at the BMW plant in Landshut. From each step further insight into the process is extracted that can be used to assist in inductive reasoning.

4.1.1 Metrics

Throughout the case studies we will use several metrics, which are defined as:

$$Precision = \frac{TP}{TP + FP}$$

$$False\ Discovery\ Rate = \frac{FP}{FP + TP}$$

$$Recall = \frac{TP}{P}$$

$$Geometrical\ Mean(A, B) = \sqrt{A * B}$$

4.2 Case I: Gravity Casting - Temperature Analysis

4.2.1 Data Overview

The data comprises of several temperature time series taken during the month of April for a single gravity casting machine which contains several operating sensors. During this month 4428 parts were produced of which only 69 were marked as a cold run. This means that our data set is very unbalanced with the negative class happening only around 1.6% of the time. Each time series has 33 measurements, and each measurement is separated by 5 seconds with the first one being taken when the liquid metal is released into the die of the cylinder head as described in section 3.1.1.

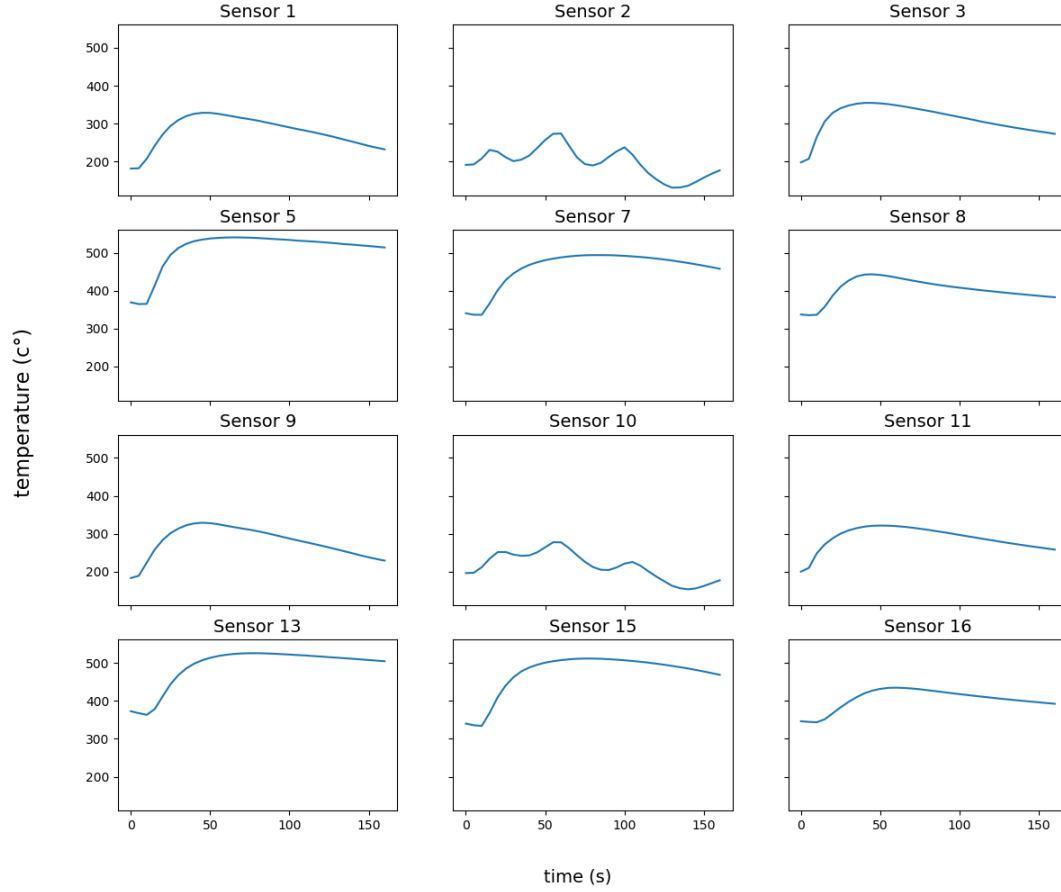


Figure 4.1: Typical temperature plot for the gravity casting process.

It is useful to visualize the typical temperature time series for each available sensor when a part is marked as OK. The plot of a usual time series for each functioning sensor can be seen in figure 4.1.

Immediately, a pattern is observed between the sensors. Every 8th sensor, that is, the first sensor and the ninth, the second sensor and the tenth, and so on seem to have very similar values (see 4.2). The reason becomes apparent once the die itself is considered: The die consists of the negative geometry of two cylinder heads, that is: two cylinder heads are cast during the same process. The implication is that the liquid metal which is released from above should fill both sides of the symmetrical die uniformly. Because of this, to analyze a produced part, one has to consider which side of the die it was cast on and from there which sensors to analyze.

In gross terms, the temperature curves increase exponentially and then decrease linearly. The increase can be attributed to the approximation of the liquid metal to the locals

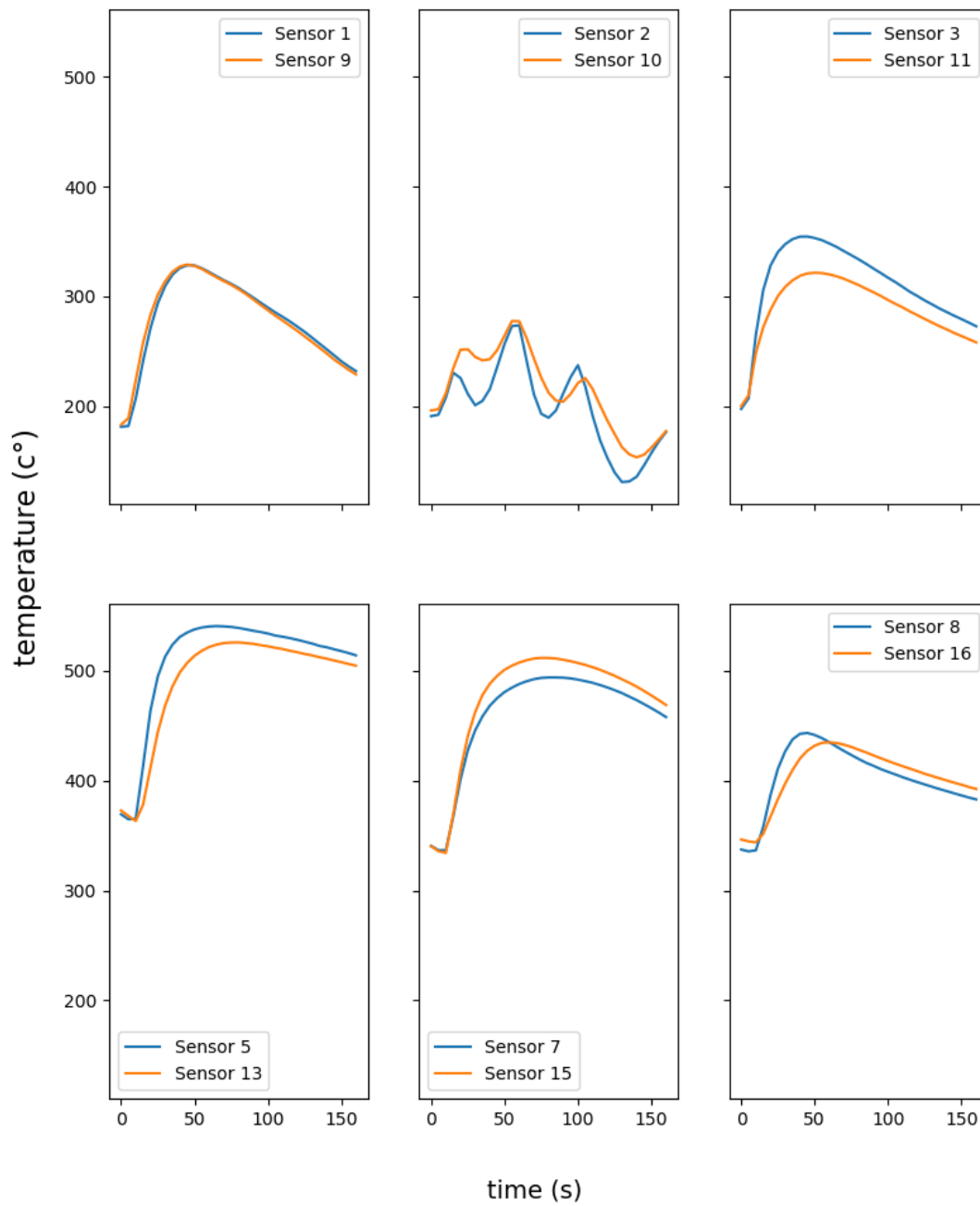


Figure 4.2: Similar sensors plotted together.

where the sensors are placed, as the liquid is poured down the sensors will detect a higher temperature. The peak of the temperature should be close to when the liquid stops being poured and the cooling down step begins.

There is, however, a pair of sensors that seem to create a different kind of time series compared to the other ones. Sensor 2 and sensor 10 still have an overall increase and decrease in temperature as the others, but they also seem to have a mixing of another sinusoidal component. Further investigation on this leads to the conclusion that these sensors are placed near where the cooling channel wraps the die. This cooling channel contains cold running water. However, it does not run water continuously, it releases the water periodically in an automated fashion. This is therefore the source of this extra sinusoidal component.

Finally, the observation of several time-series and their respective process statuses, that is, if the process produced a defect or not, is studied. This observation, figure 4.3, suggests that the time series obtained from any of the available sensors does not offer any linear separation between the OK and NOK parts. To confirm this, we can compare the mean and standard deviation of the average temperature of a time-series for both the OK and NOK parts in figure 4.4, again the conclusion that they are not linearly separable is confirmed. However, one can observe that the mean of the NOK parts is usually lower, and its variation is higher. There is, therefore, some statistical information that can be extracted from the time series.

4.2.2 Preparing Data

While we are interested in outliers, gross sensor malfunction is not of interest as it is not directly correlated to the process. It is necessary, therefore, to correct or remove cases of gross sensor malfunction. In this industrial case, a gross sensor malfunction has the sensor returning extreme values, i.e.: nine hundred ninety-nine (999) or zero (0) °C. Since these values are otherwise never reached in our process, the malfunctions are easy to identify, and we incur in no risk of a false positive (see 4.5).

Since the percentage of processes that have a gross sensor malfunction in any of its sensors is only 3.6% and introducing corrections could form some bias in the analysis, we have decided to remove them from the data set.

4.2.3 Feature Extraction

The processes over one month are going to be analyzed. The time series from sensor 8 will be the focus of the analysis, mainly due to its central position in the casting die. Nonetheless, the other sensors could be analyzed in a similarly.

We will start by extracting principal and independent components using ICA and PCA, respectively. While they are difficult to interpret it is useful to compare them to the NMF components that we will also extract.

The analysis will begin with a PCA. The number of principal components to extract has to be defined. This choice is usually taken by increasing/decreasing the amount of

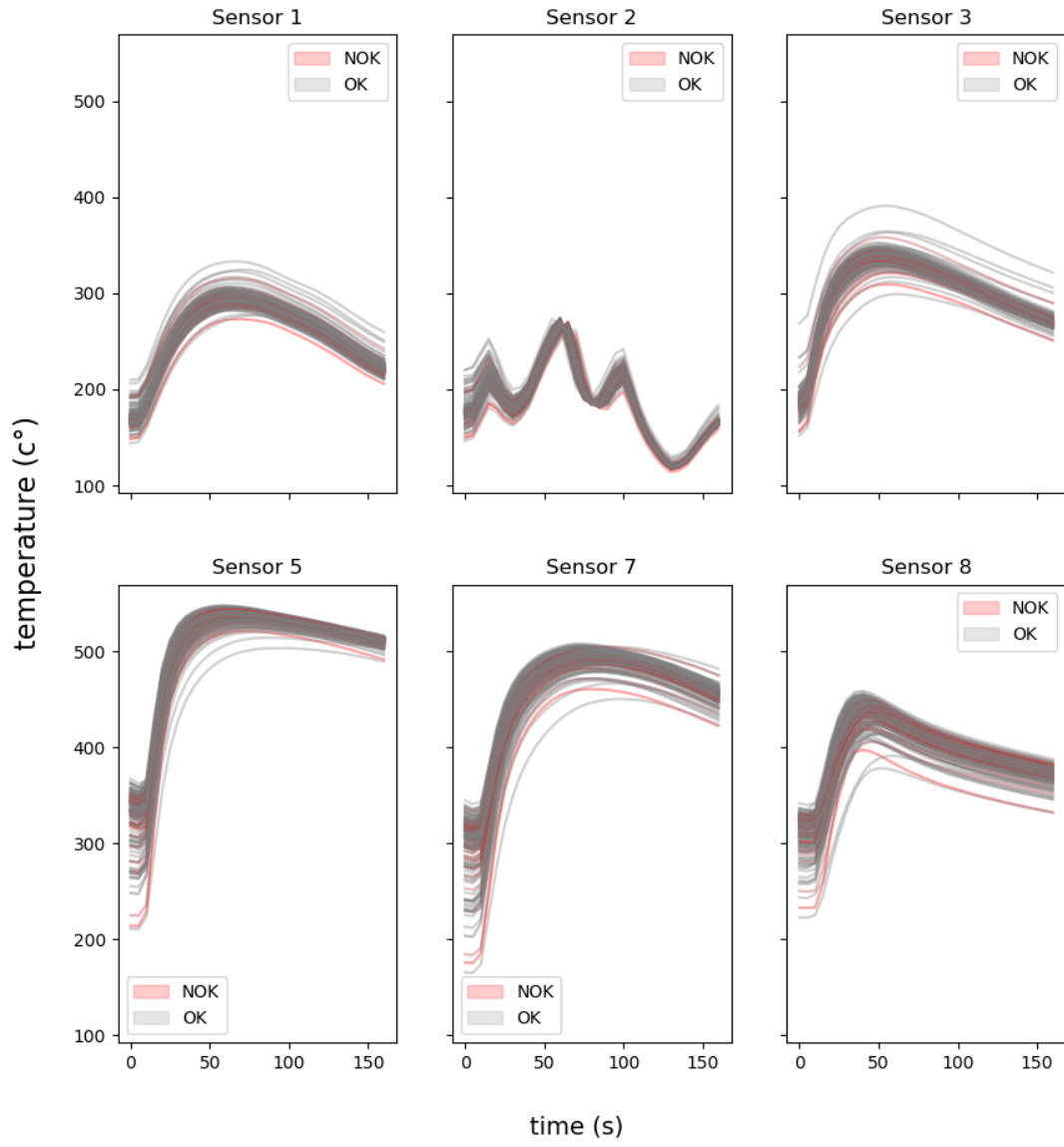


Figure 4.3: One day of processes, comparison between OK and NOK time series.

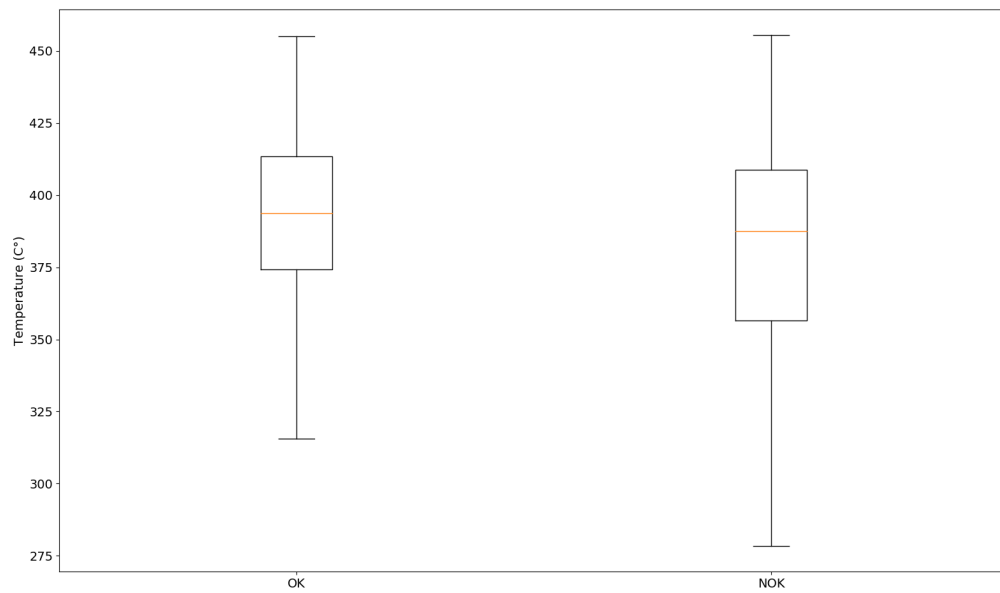


Figure 4.4: Mean and standard deviation of the average temperature of a timeseries for OK and NOK time series.

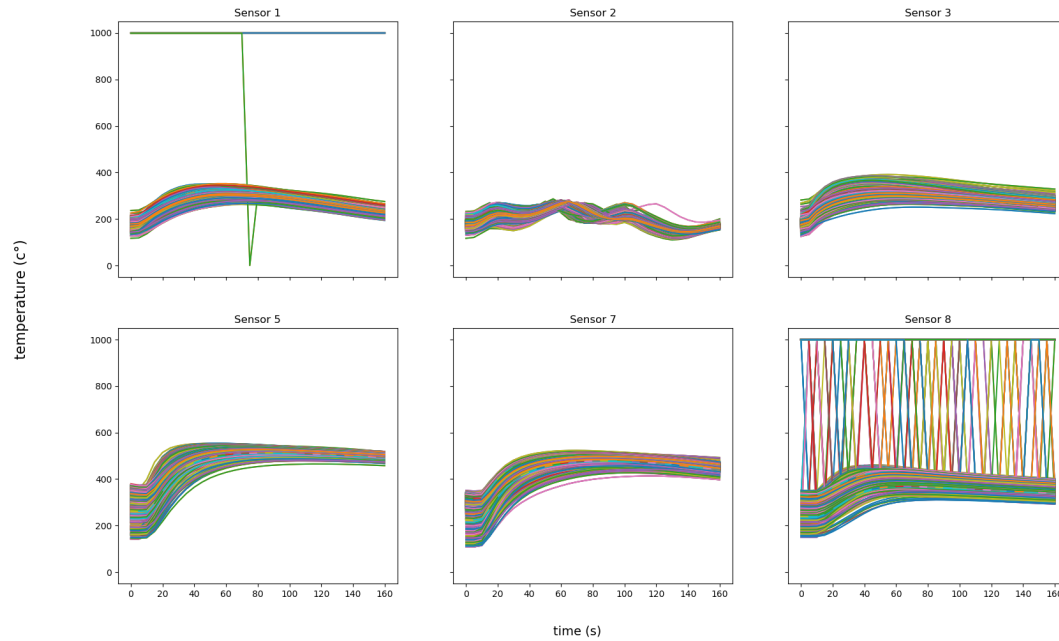


Figure 4.5: Visualization of entire data including gross sensor errors.

components iteratively and comparing the amount of variance that is explained by each principal component. A rule of thumb is to choose the value where an elbow exists.

Figure 4.6 measures these values for up to ten principal components, it can be observed that there is an elbow by three principal components and thus this is the number of principal components that are chosen to be extracted from the matrix.

The obtained components of the principal component analysis are seen in 4.7. By nature of the PCA method, the components should be the ones that explain the most variance. Any factors that are unchanged during production are not extracted by PCA. Furthermore, we can observe that most of the change happens during the first half of the measured period. The second half is mostly stable and therefore the focus of the analysis should be on the first half.

Also interesting are the projections of the time series into these three new base vectors, our new features 4.8. As they are ordered by time of production, one can see that some patterns take form, the biggest of which is in the second component. It seems like something changed in the middle of the month and then changed back by the end. By nature of the method used these patterns should be uncorrelated.

As an attempt to further discriminate patterns, we will impose another constraint to the coefficients. ICA looks for the independent components of the signal. Using the fastICA algorithm, and again looking for 3 components, we acquire the bases seen in figure 4.9. Even though the scale is lost when using ICA due to the whitening phase, similarities can

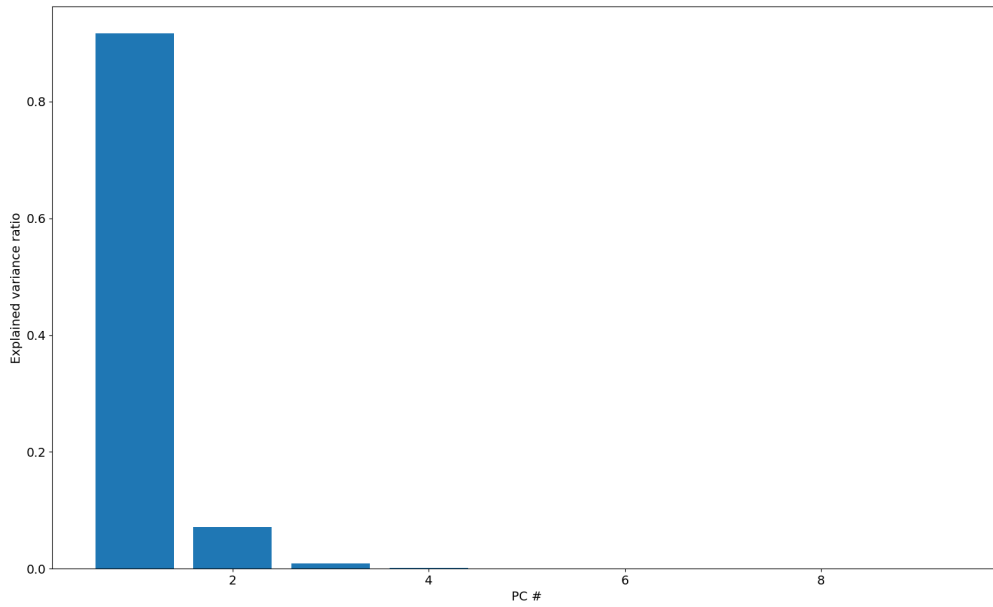


Figure 4.6: Explained variance ratio depending on number of principal components.

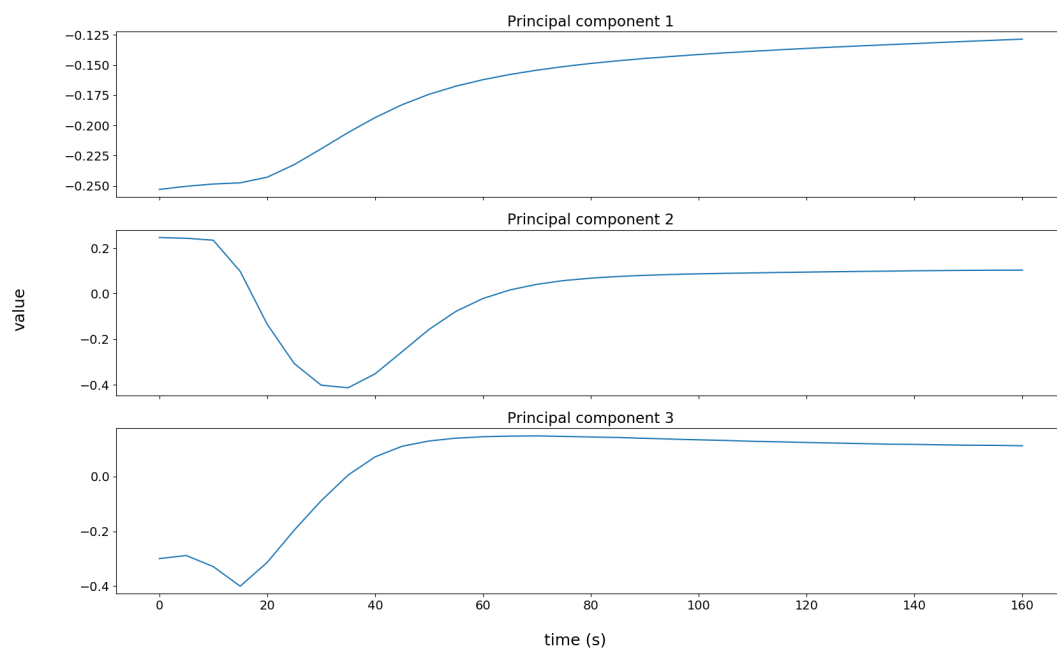


Figure 4.7: PCA bases.

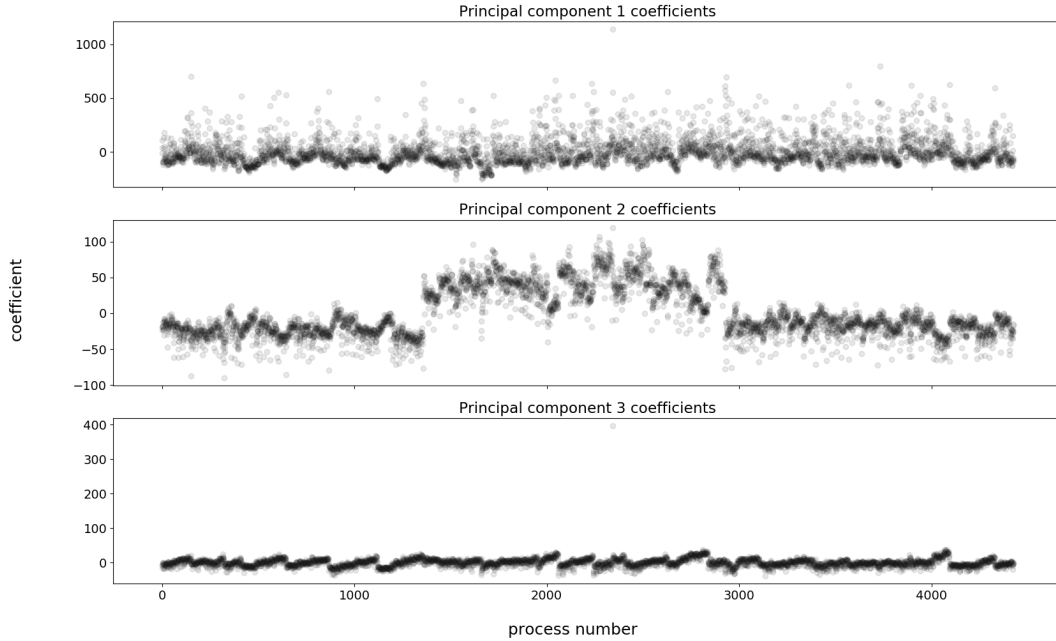


Figure 4.8: Time series projected into the 3 new PCA bases.

be observed between components of PCA and ICA.

More notably, the coefficients of both unmixing methods are similar; they exhibit the same patterns (see 4.10).

While the analysis of both PCA and ICA on the data is interesting and provides some insight, the constraints that define both the methods are not very useful for our industrial case. PCA extracts uncorrelated components; we cannot be sure that they relate to any ongoing effect as most effects in the process will likely be related to one another. The same can be said for ICA since its constraint is the statistical independence of the components and is, therefore, an even more constrained assumption than PCA.

The non-negativity constraint, however, is less severe and intuitively makes sense in the context of temperature measurements. We know that the temperatures never go below zero, and furthermore, the coefficients being non-negative allow us to interpret the components on a scale of its effect in the measurements: Zero means that it is not present and large nonzero coefficients imply a significant impact.

For these reasons the components that can be better interpreted are the NMF's components, therefore we now begin their analysis.

The amount of components to extract has to be defined. Like with the PCA components, in figure 4.11 it can be observed that there is an elbow by three components and thus this is the number of components that are chosen to be extracted from the matrix.

We can initialize the NMF basis matrix using different methods. Naively one might

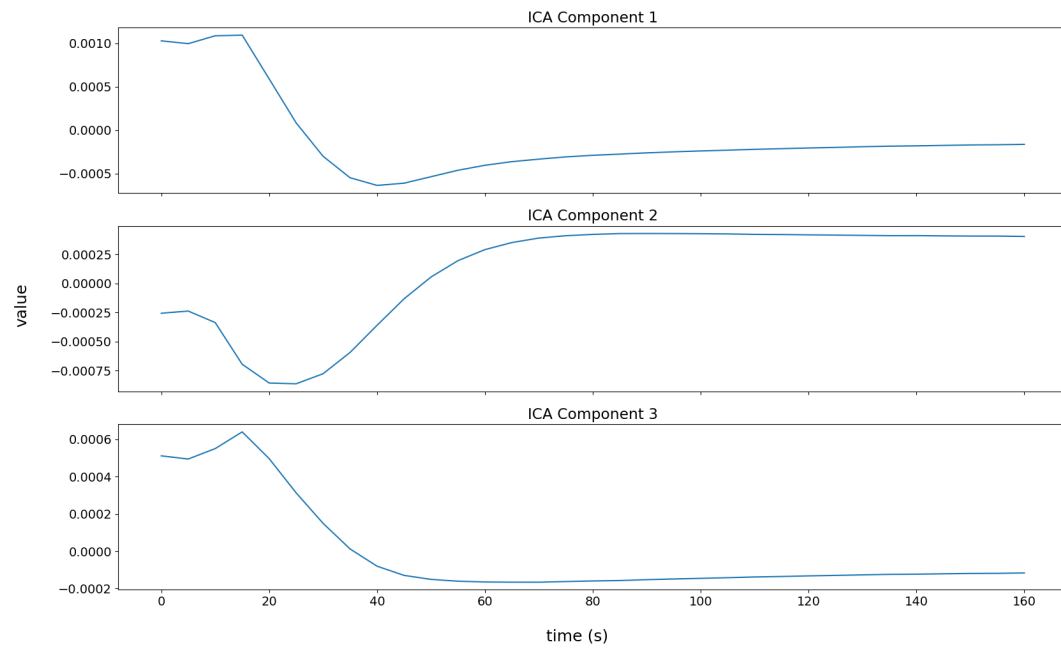


Figure 4.9: ICA bases.

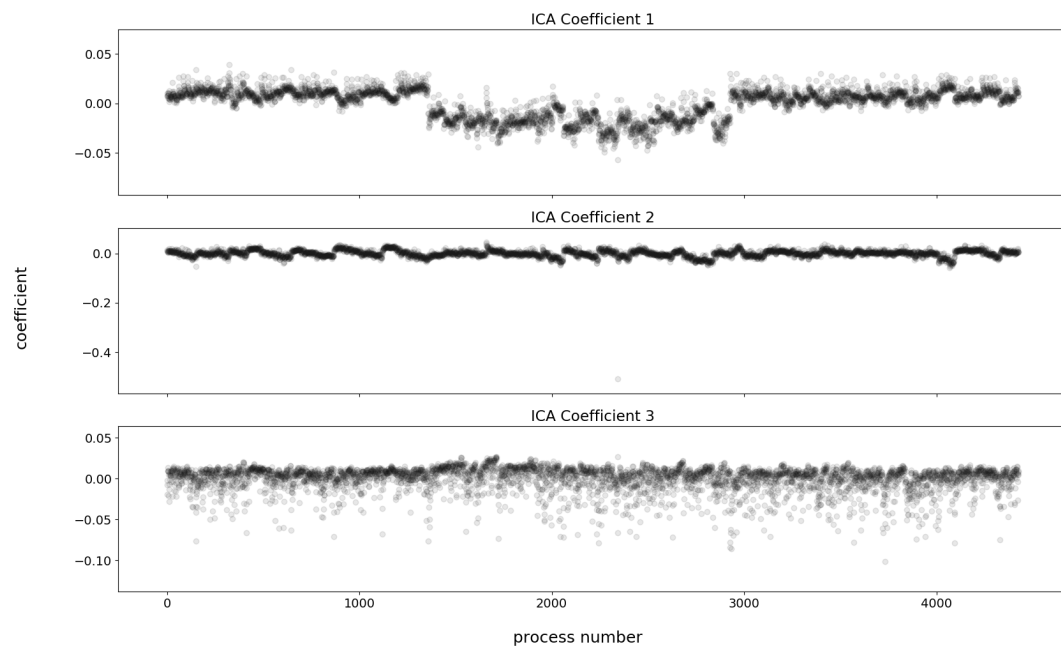


Figure 4.10: Time series projected into the 3 new ICA bases.

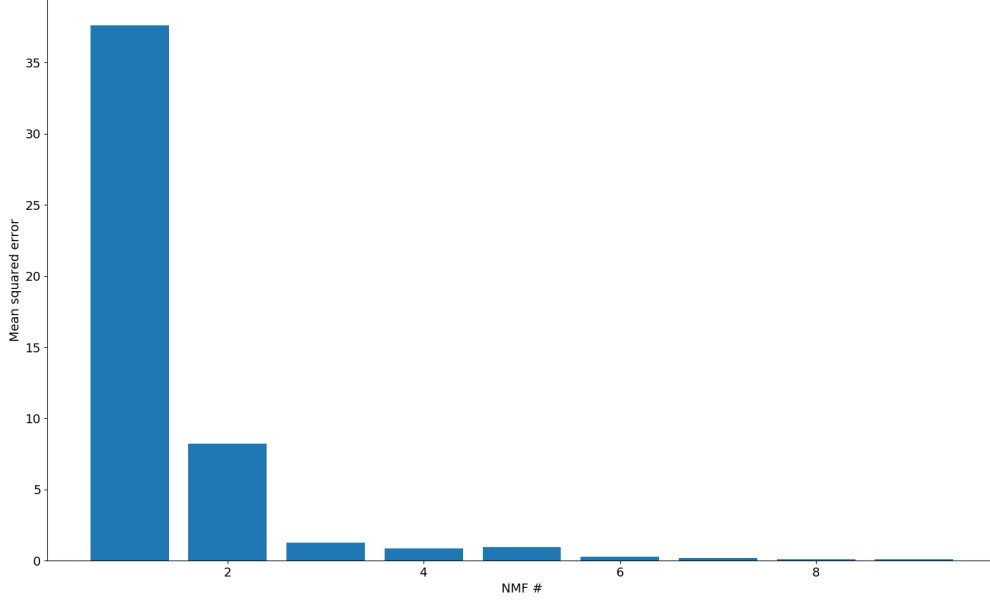


Figure 4.11: Mean squared reconstruction error depending on number of components.

initialize it randomly. We can also initialize this matrix with the respective basis matrix of the PCA and ICA decompositions. Each of these bases is shifted to the first quadrant to comply to the non-negative constraint of the non-negative factorization.

We can observe how the principal and independent components look like in figure 4.7 and figure 4.9 respectively. Using these and the random initialization strategies the respective components are seen in figure 4.12, depending on the initialization the components may converge to different bases.

However, it was proposed by Peter Weiderer [8] while working on the same data that the non-negative matrix basis can be used to model the three main physical processes that affect the process: The initial temperature's impact, represented by an exponentially decreasing time series, the base of the temperature time series, represented by an exponentially increasing time series, and the heat transfer rate, represented by a mixed time series.

This representation is useful for the ease of interpretation, to translate this knowledge into the real process and for the initialization of the NMF matrices with process knowledge.

From this, one might initialize NMF bases as those three factors. An exponential increase $n * (1 - e^{-t/x})$, an exponential decrease $n * e^{-t/x}$ and a mix of increase and decrease $n * (e^{-t/x1} - e^{-t/x2})$ where n is the amplitude of the signals and x , $x1$ and $x2$ pertain to the rate of change, these functions, with $n = 1$ can be seen in figure 4.13.

The knowledge and random initialization seem to converge to somewhat the same bases

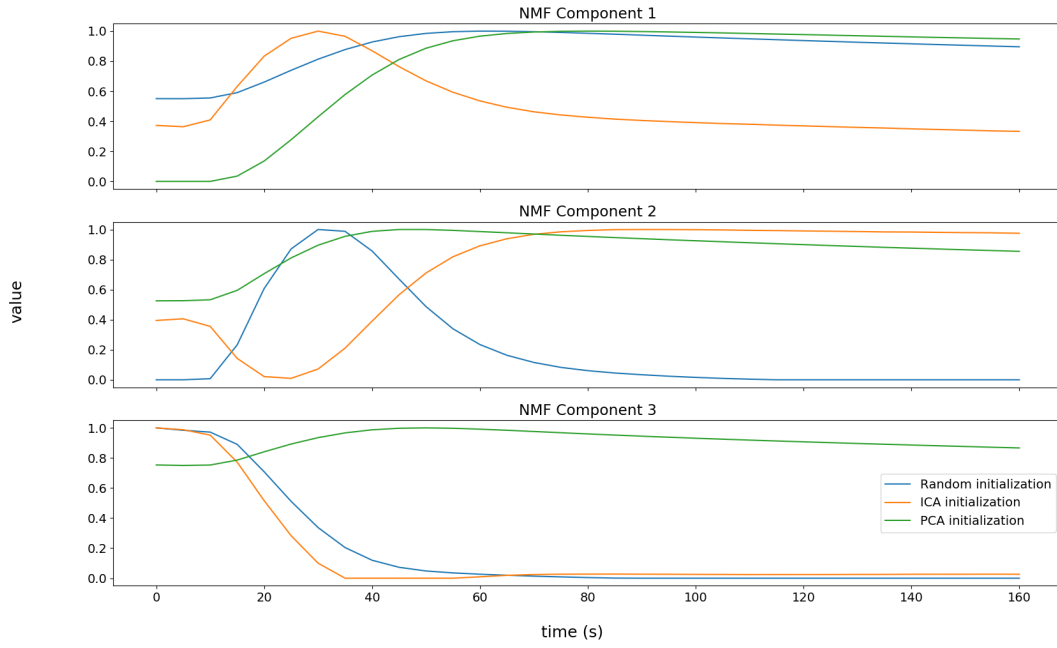


Figure 4.12: Normalized NMF bases with PCA, ICA and random matrix initialization.

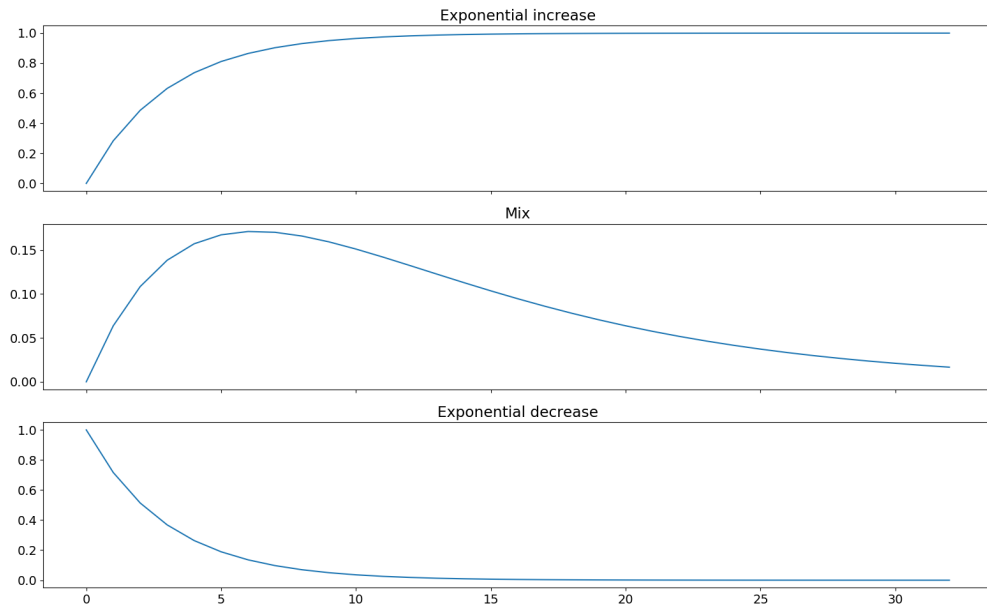


Figure 4.13: Three physical factors for NMF initialization with amplitude = 1.

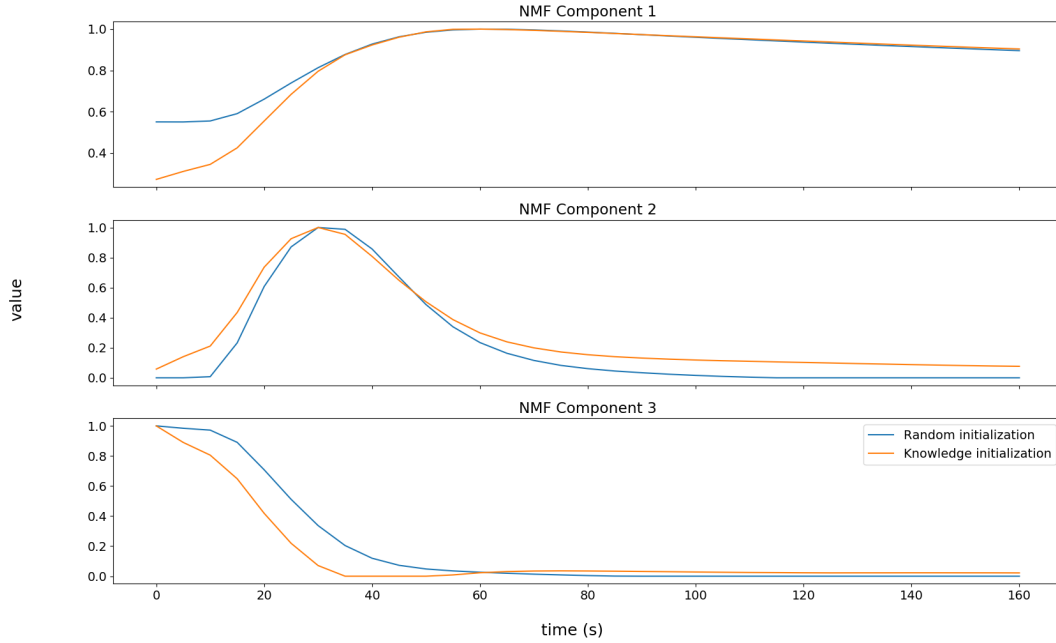


Figure 4.14: Normalized NMF bases with knowledge and random initialization.

4.14. This leads us to believe that those physical factors can indeed be used to modulate the temperature time series. And, indeed, the reconstruction error is very low with differences only in the limits of plus or minus two degrees Celsius.

If one looks at the projected coefficients of the knowledge initialization 4.15 the patterns can now be tied to a physical factor. Much like the PCA and ICA coefficients we still see one of the components drastically change in the middle of the month and then go back to normal. This same component also appears to show some periodical ups and downs. With this physical interpretation, the analysis of the projections of the time series becomes more natural. For example, an increase in the second component could mean that the heat transfer rate increased, and a decrease means the opposite.

From the feature extraction process we have achieved 3 main milestones:

- Interpretable features;
- Reduction of the original data dimension from 33 points to 3;
- Low reconstruction error.

4.2.4 Random Forest Parameters

The new data representation must now be combined with the label information. To this end, we apply a Random Forest Classifier. This tree ensemble can then be used by

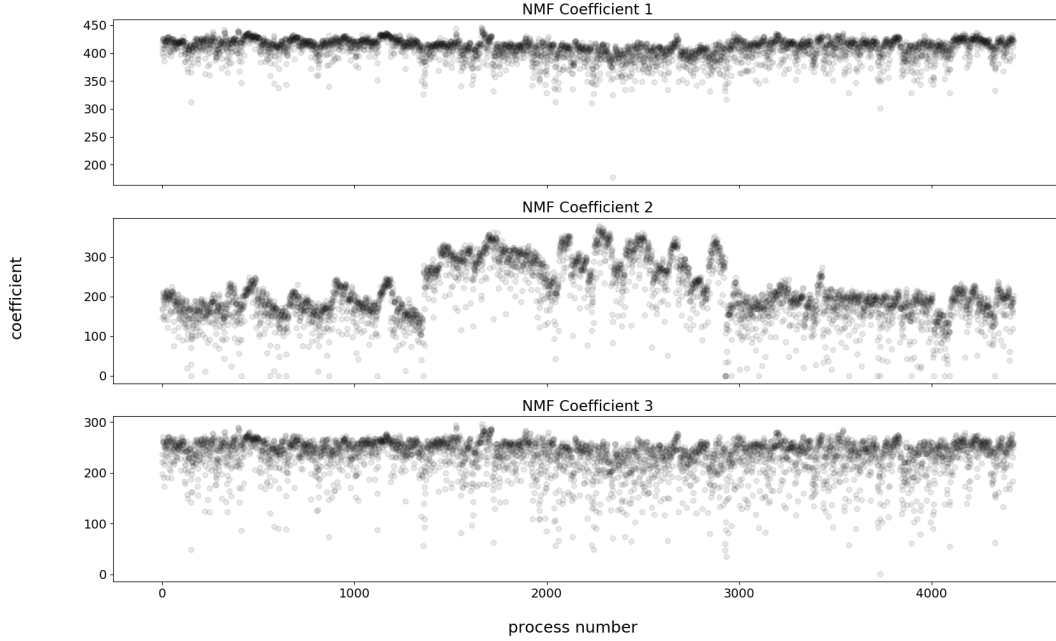


Figure 4.15: NMF coefficients for the knowledge initialization.

the Rule Fit method to extract several rules. Note that the data set is highly unbalanced: 4428 processes were captured from which only approximately 1.6% (69) were marked as NOK, for this reason, the Random Forest Classifier will use a cost-sensitive technique to balance the weights for each class [45].

Before the rules are extracted however, we want to determine the parameters for the Random Forest Classifier.

Generally speaking, the number of estimators and the performance of the Random Forest are directly related. However, there is a diminishing return effect, and one also has to balance the computational effort required. For these reasons, the random forest classifier is trained with 100 estimators to reduce the intrinsic variance of the tree classifiers but maintain the computational effort relatively low.

At each node, only \sqrt{N} of the total amount of features are randomly selected to be considered to create a new split. This is the norm in most literature. To measure the quality of a split the Gini impurity measure is calculated.

Lastly, we must decide on the depth of the tree. We measure the OK recall, OK precision, and the geometric mean of the RuleFit algorithm depending on the maximum depth of the Random Forest in 4.16. The performance increases with the maximum depth allowed until depth 8 and then sees no significant improvements afterward.

Using a 5-fold cross-validation technique where the percentage of classes is maintained for each fold (stratified K-Fold) we can observe how the largely imbalanced dataset affects

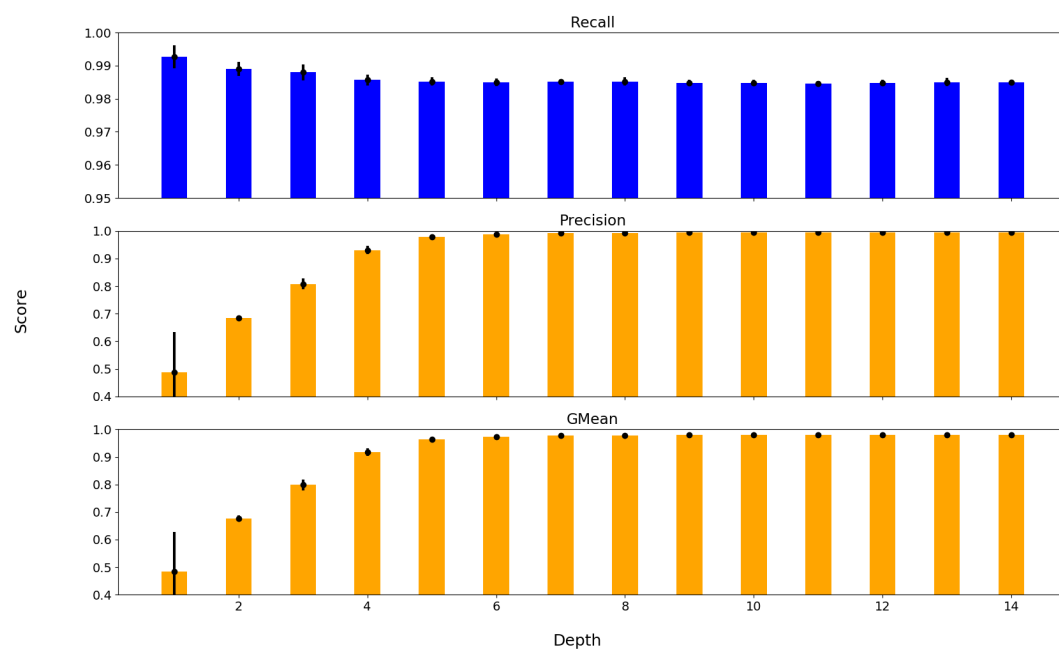


Figure 4.16: Recall, Precision and their Geometric Mean of Rule Fit depending on the maximum depth of the Random Forest (KFold = 5).

the predictive power of the class of NOK processes (see table 4.1).

OK Recall	NOK Recall
0.995	0.015

Table 4.1: 5-Fold Cross Validation average Recall for OK and NOK processes.

4.2.5 Rule Extraction

With a Random Forest classifier trained we can now begin the process of extracting their most useful rules.

The four most important 2.29 rules are:

Rule 1: $w_{*,1} \leq 415.06 \ \& \ w_{*,2} > 197.56 \ \& \ w_{*,1} > 397.07 \ \& \ w_{*,2} \leq 251.52 \ \& \ w_{*,3} > 226.10$

Rule 2: $w_{*,2} > 255.26 \ \& \ w_{*,1} > 404.62 \ \& \ w_{*,3} \leq 251.20 \ \& \ w_{*,2} \leq 344.09$

Rule 3: $w_{*,1} \leq 419.69 \ \& \ w_{*,2} > 94.17 \ \& \ w_{*,3} > 242.56 \ \& \ w_{*,1} > 416.31$

Rule 4: $w_{*,3} > 230.96 \ \& \ w_{*,2} \leq 226.56 \ \& \ w_{*,1} \leq 416.30 \ \& \ w_{*,1} > 397.07$

One might notice that some rules can contradict each other. This happens since the rules might model different subsets of the data such as different time periods or configurations of the processes. One of the rules might model for example a minimal subset of outliers and therefore have a large coefficient in the logistic regression making its importance higher. To illustrate this point see figure 4.17 where we compare the processes that are within the restrictions of two rules that seem to contradict each other: rule 2 and rule 4. It becomes evident that the rules model different periods. Rule two is modelling a specific period where some configuration of the process was changed and as a result, the second NMF coefficients increase.

Each rule extracted from the RuleFit algorithm can be interpreted as a set of conditions under which the process is optimized for minimizing its NOK outcomes. Following this interpretation, if the OK false discovery rate of the rule is lower than the percentage of negative processes. We should also see a decrease in the defects when the rule is applied. However, even if the miss rate is 0%, the rule can be too restrictive and, therefore, not be applicable in the real process. To this end the OK recall should be maximized, secondary to the OK false discovery rate (FDR).

The scores from the four most important (see equation 2.29) rules extracted by RuleFit are seen in table 4.2, along with the score of OneRule for comparison.

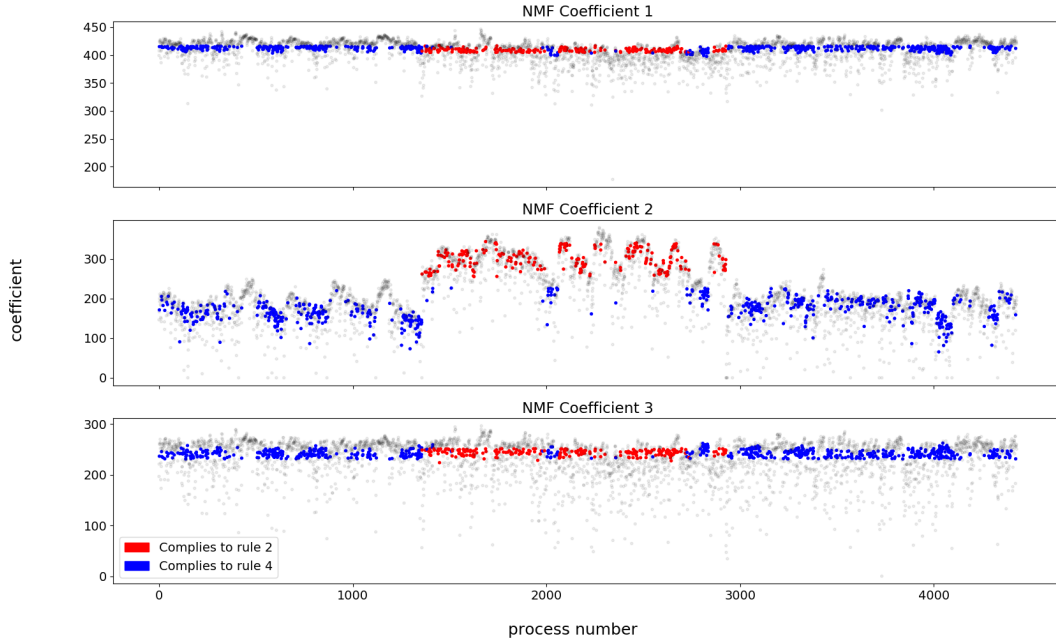


Figure 4.17: Processes that comply with 2nd, 4th or neither rule.

Table 4.2: Gravity Casting: Scoring of the rules on the testing set.

	OK FDR	OK Recall
One Rule	1.6%	100.0%
Rule 1	0.0%	3.8%
Rule 2	0.0%	5.9%
Rule 3	2.4%	9.4%
Rule 4	0.0%	14.5%

The One Rule algorithm performs poorly in this data set. It is unable to classify a NOK class. This performance suggests that the relation between the features is essential in the distinction between both classes. Only rule 3 performs worse, as its false discovery rate is higher than the original percentage of 1.6 % of defects.

Of the other rules which all have a 0.0 % miss rate in the testing data the fourth rules has the most significant positive rate and should, therefore, be the least restrictive rule to apply in the process:

$$w_{*,3} > 230.96 \ \& \ w_{*,2} \leq 226.56 \ \& \ w_{*,1} \leq 416.30 \ \& \ w_{*,1} > 397.07$$

4.2.6 Results

To use the rule in controlling, one has to understand how they translate to the industrial processes. Since the meaning of each NMF component was already described in the feature extraction section, the components must now only be paired with the selected rule.

The chosen rule indicates that to optimize the process for its scrap rate:

- The baseline temperature should be held between a certain threshold ($w_{*,1}$);
- The heat transfer rate ($w_{*,2}$) should be lower than a certain threshold;
- The initial temperature of the mould should be maintained above a threshold ($w_{*,3}$).

If all those conditions are followed, one can expect fewer defects.

From this knowledge, a process expert can now define concrete actions to improve the process. One such example is increasing the frequency of the coat application as it is known to be inversely correlated to the second NMF coefficient.

This change was performed in the process: Coating was now applied when the second NMF coefficient was too high. This contrasted with before where the coating was applied at random intervals by decision of the process worker. The result was a 68% defect drop: From 1.6% to 0.5%. See figure 4.18 for a visualization of the coefficient one week before and after the change was applied.

4.2.7 Summary

Using unsupervised learning, we extracted interpretable components from the original data, separating the mixture into 3 main components. This extraction also has the effect of reducing the dimension to 3 from the original 33 features.

Furthermore, we combined this approach with a rule extraction technique which can provide guidelines for the process expert to follow.

This approach allowed us to take a large amount of complex data and create knowledge that can be interpreted and applied to the process to reduce the scrap rate of parts.

4.3 Case II: High Pressure Casting - Temperature Analysis

4.3.1 Data Overview

This data was collected during a pre-production phase of the casting machine. This means that the machine was not continuously producing during these weeks. Due to this, the collected data is limited and, therefore, any results obtained from statistical analysis are not reliable. Furthermore, the parameters of the casting machine were changed several times during this testing period which may create inconsistency issues with the data.

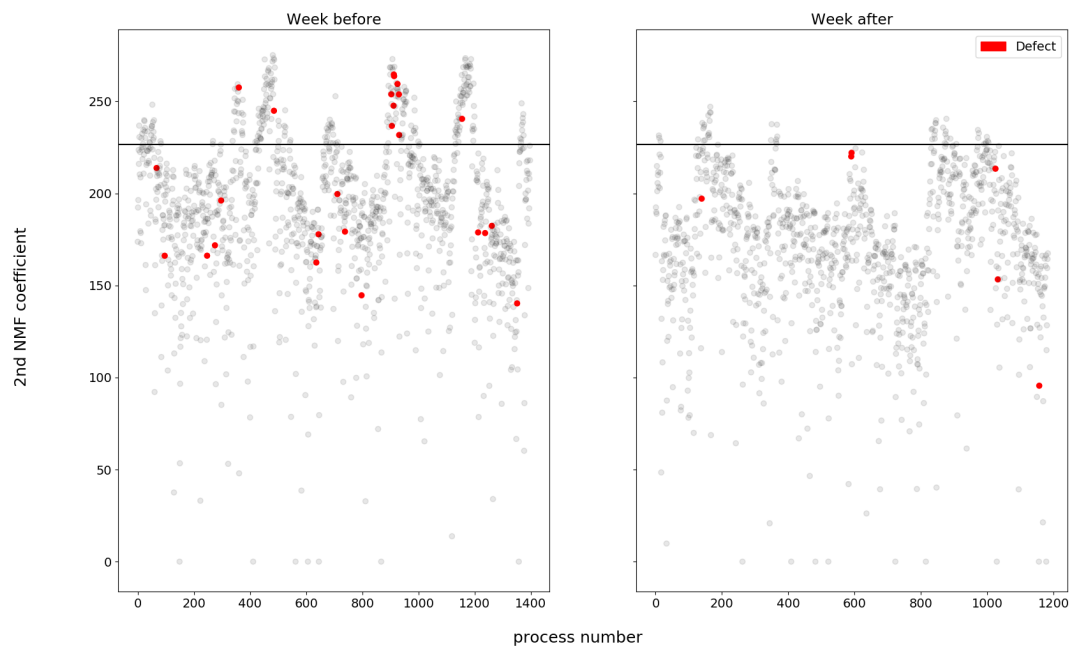


Figure 4.18: Second NMF coefficients a week before and after the change was applied to the process.

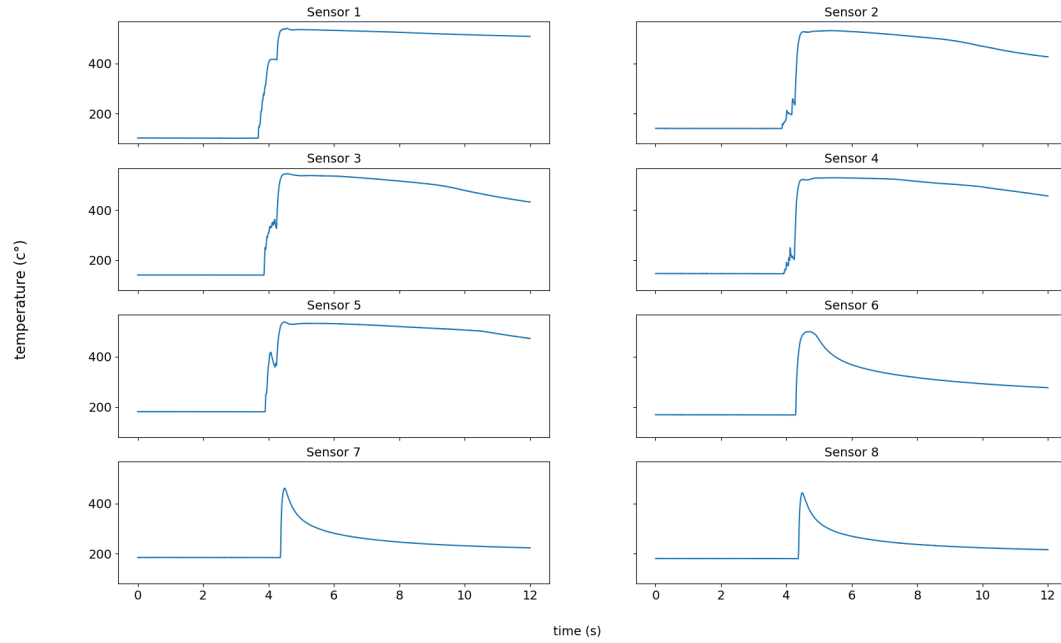


Figure 4.19: Typical temperature plot for the high pressure casting process.

In total 909 processes were captured of which 128 were labelled as NOK, this means that 14.1% of the data set is labeled as the NOK class.

Once again, data of temperature over time will be analyzed. However, the frequency of the recording is much higher in this process, at around one measurement every 0.004 seconds over 12 seconds. This frequency means that each time series will contain three thousand values. The process is described in more detail in section 3.1.2.

A normal time series is divided in three phases (see 4.19):

- Stable until the piston is fired and pushes the liquid metal into the cast;
- Exponential increase of the measured temperature while the liquid fills the cavity;
- Cool down period.

As in the previous industrial case we will look at the differences between OK and NOK processes (see 4.20). Even though there is much overlap between the time series with both status, there seems to be a correlation between the time at which the piston is fired and defects.

Also, when comparing the mean of the average temperatures of the OK and NOK processes, it is observable that the defective mean is lower than the normal mean 4.21, this means that processes with lower temperature have a higher chance of producing defective parts.

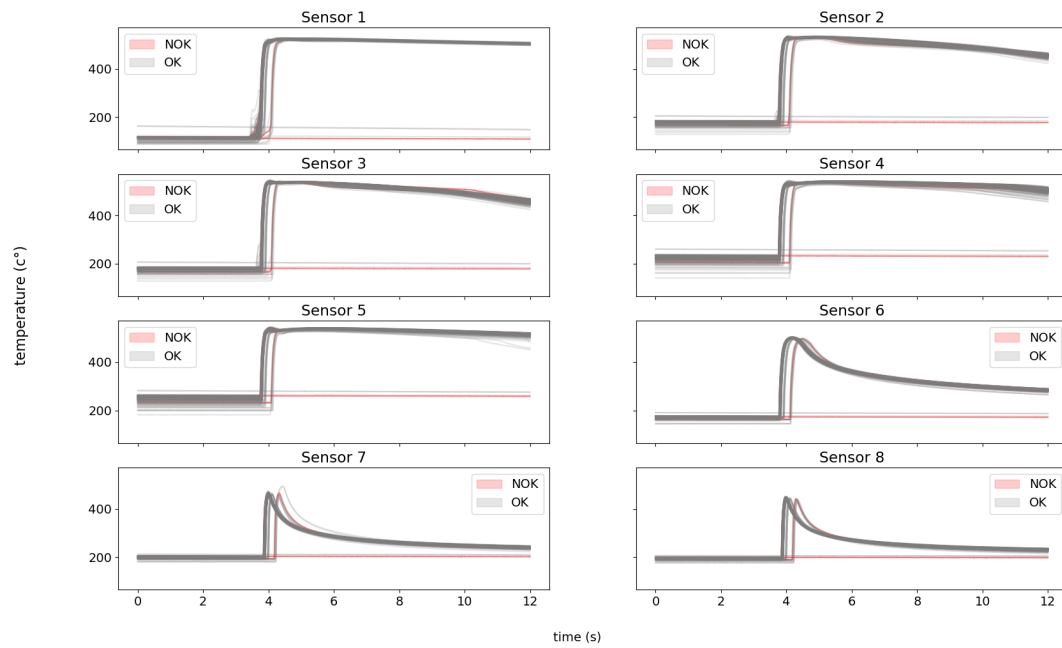


Figure 4.20: One day of processes, comparison between OK and NOK time series.

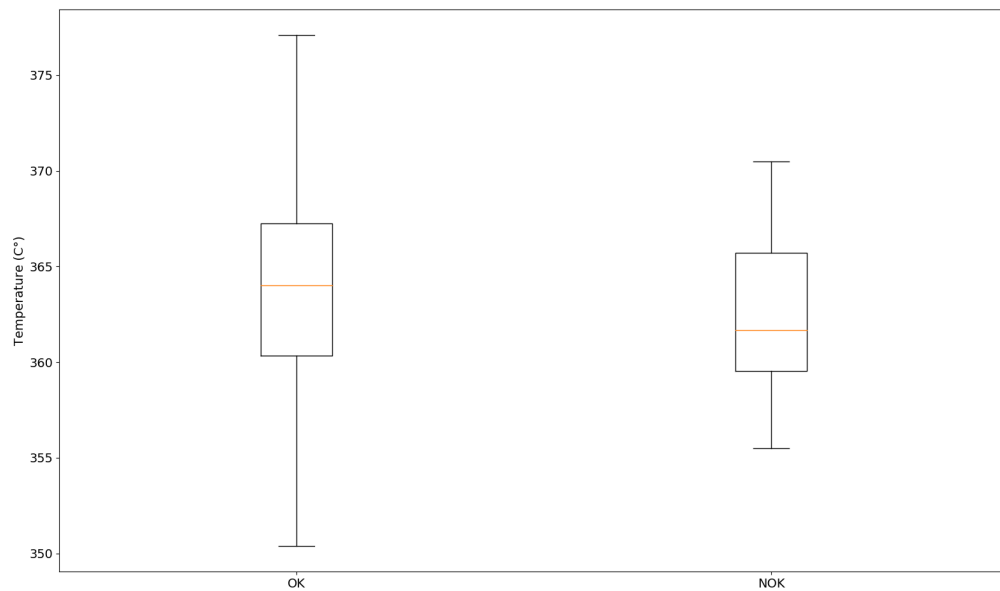


Figure 4.21: Comparison between the mean temperature of OK and NOK time series.

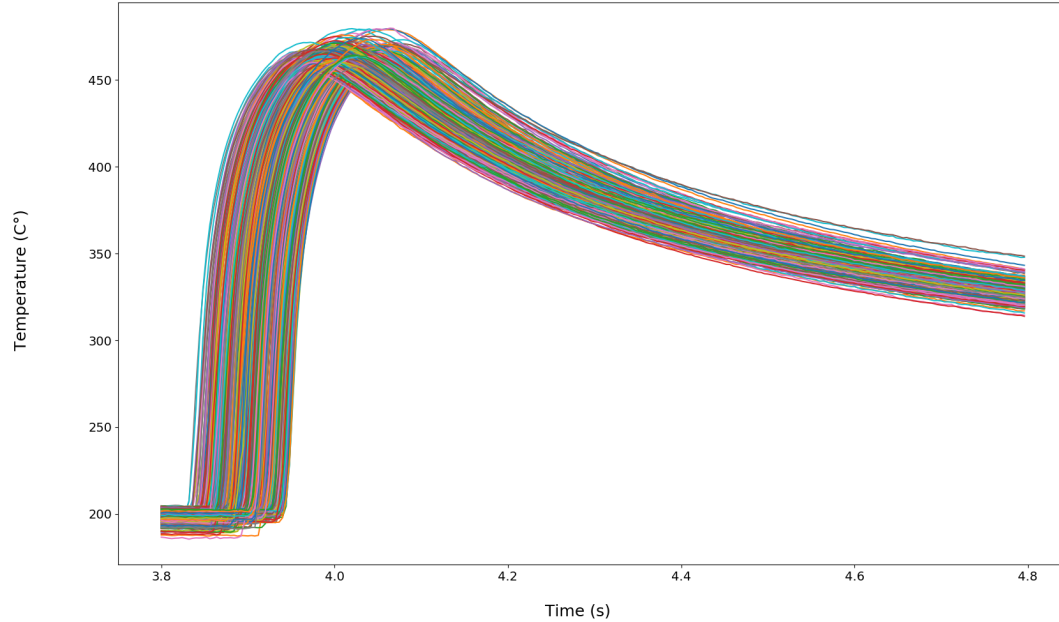


Figure 4.22: Time series of sensor 7 within the relevant time frame.

4.3.2 Preparing Data

Before any processing of the data is applied the data must be prepared.

In this setting, there are two unmistakable signs of gross sensor malfunction: The time series having all their measurements below 400 degrees or containing temperatures of 999 degrees Celsius that is impossible to achieve in this machine. Any time-series that contains one or two of these symptoms is discarded.

With the gross sensor malfunctions already discarded, it is also possible to reduce the amount of measurements per time series: Although the whole process takes around 12 seconds, we are interested in the filling of the cavity. Before the piston is shot, the data is at a constant value, and therefore not particularly interesting. Additionally, the cool-down period is also very stable and not going to be a focus of the analysis. Figure 4.22 shows how the data looks like when only the relevant time frame is observed.

4.3.3 Feature Extraction

We will extract features from the temperature measurements of sensor 7 using the same approach outlined in section 4.2.3 of the previous industrial case. Many of the discussions from the previous case also apply here since the nature of the process is very similar.

We have to again define the amount of components to extract from the matrix decomposition techniques. There an elbow can be observed at 3 components for the explained

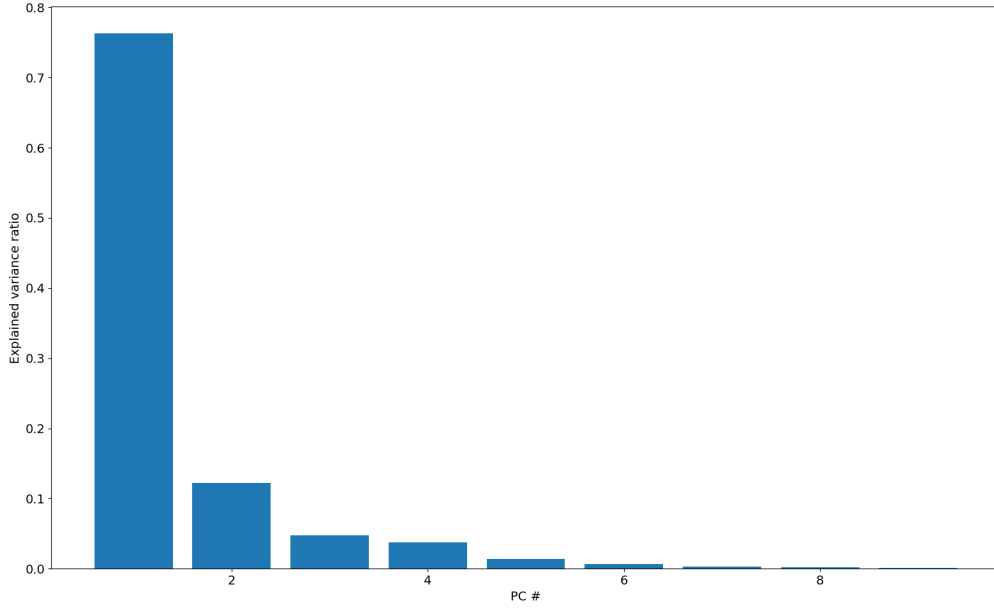


Figure 4.23: Mean squared reconstruction error depending on number of principal components.

variance in figure 4.23.

The PCA components in 4.24 and ICA components 4.26 are again very difficult to interpret due to the nature of their constraints and as in the last case appear similar. Furthermore, looking at the coefficients there are some clear periodical increases and decreases (see 4.25 and 4.27) which seem to appear similarly in both methods. Since this data was collected during a pre-production phase of the casting machine, it was not continuously producing parts. These stoppages may explain the pattern since the starting temperature of the processes is lower when the cast is allowed to cool-down during production breaks.

The NMF components were initialized using the same strategy as in the high gravity casting process. However, this time the components 4.28 don't converge to the expected factors (refer to figure 4.13 of the last industrial case). Even so, both the initialization based on knowledge of the physical processes and randomized initialization of the NMF matrices converge to the same bases with scale and permutation variations. This indicates that the components that most affect the process are not the expected factors but something else. Looking at figure 4.28, the first component, as expected, models the base of the time-series. The second and third components are similar but shifted in time. One interpretation is that both these components model the switching point of the piston movement (see section 3.1.2).

To summarize, after the feature extraction we are able to:

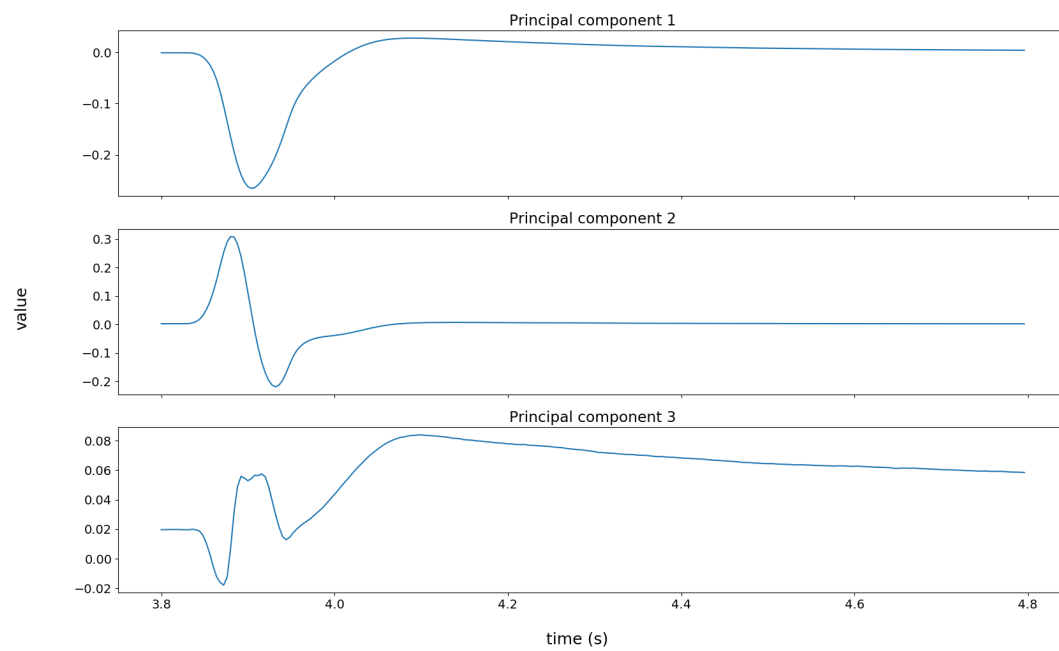


Figure 4.24: PCA bases.

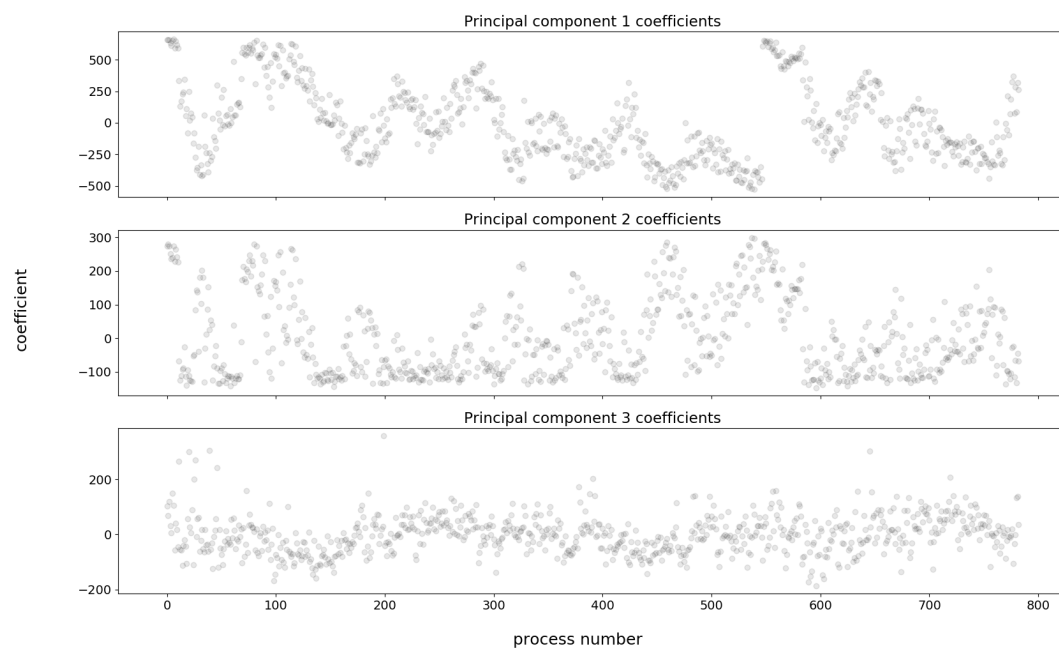


Figure 4.25: Time series projected into the 3 new PCA bases.

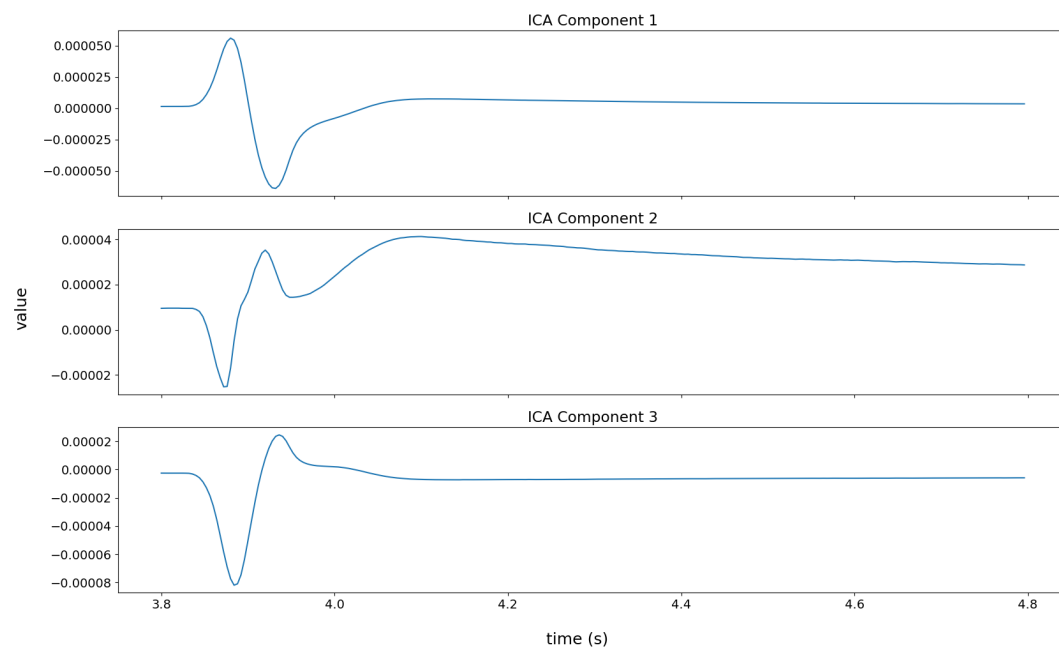


Figure 4.26: ICA bases.

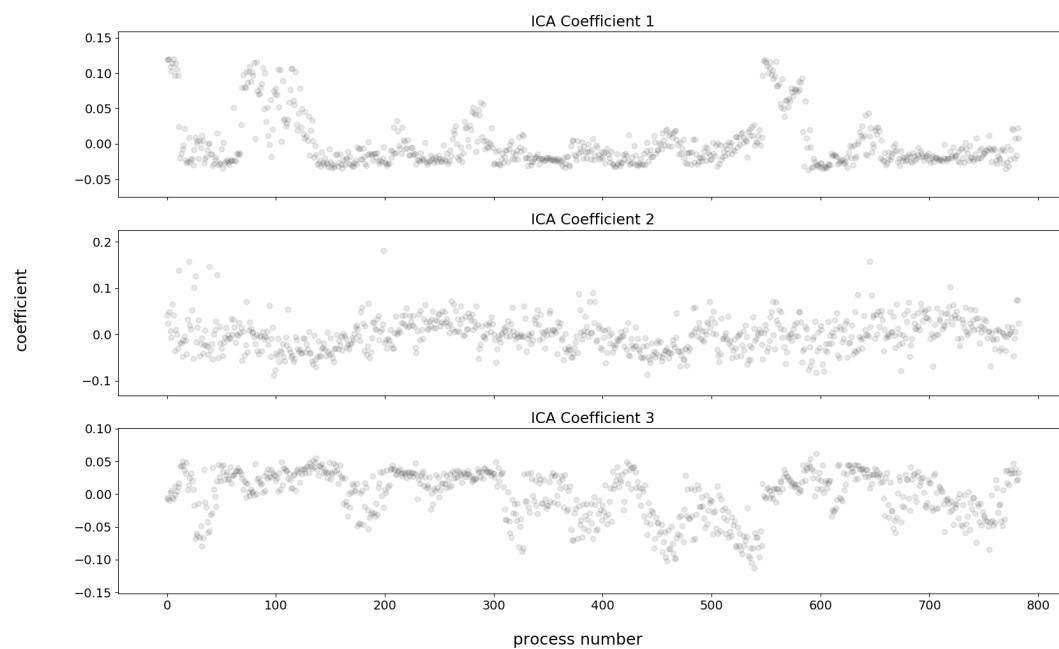


Figure 4.27: Time series projected into the 3 new ICA bases.

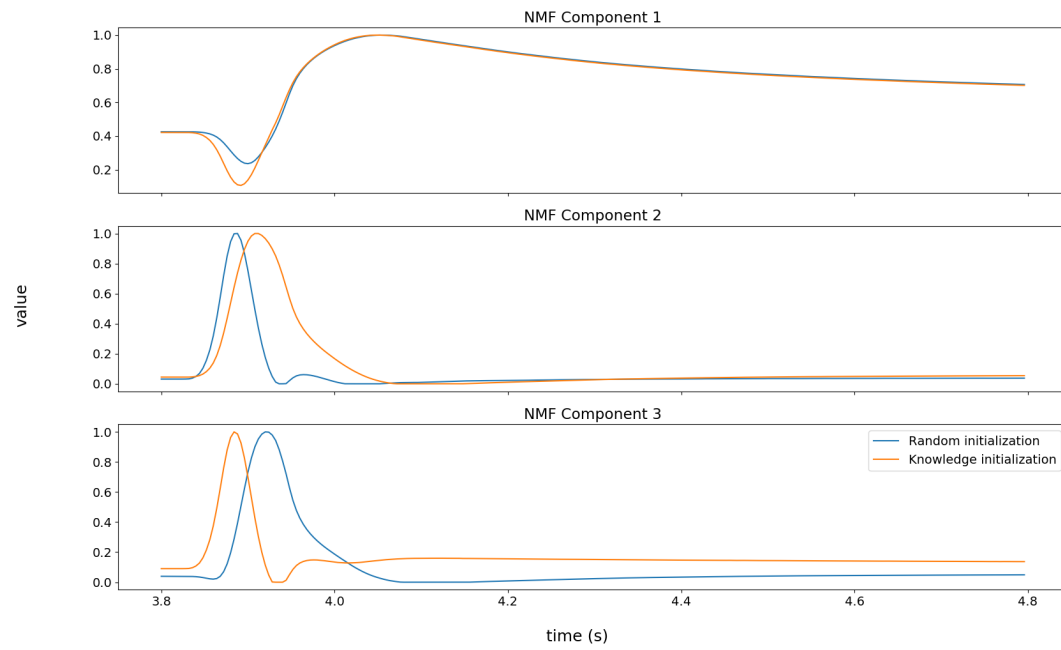


Figure 4.28: Normalized NMF bases with knowledge and random initialization.

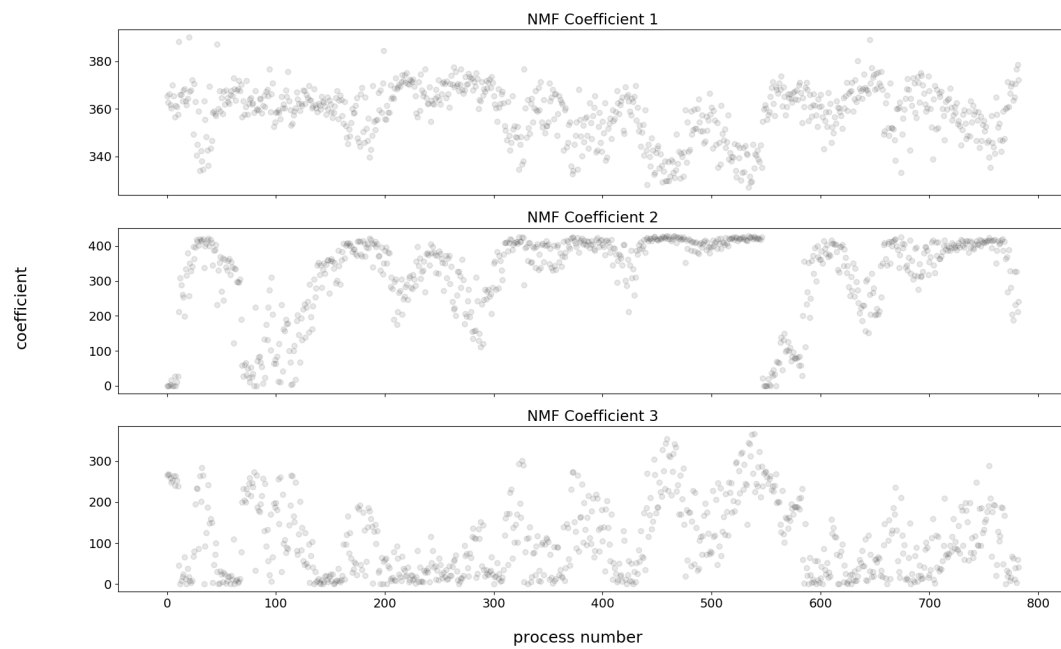


Figure 4.29: NMF coefficients for the knowledge initialization.

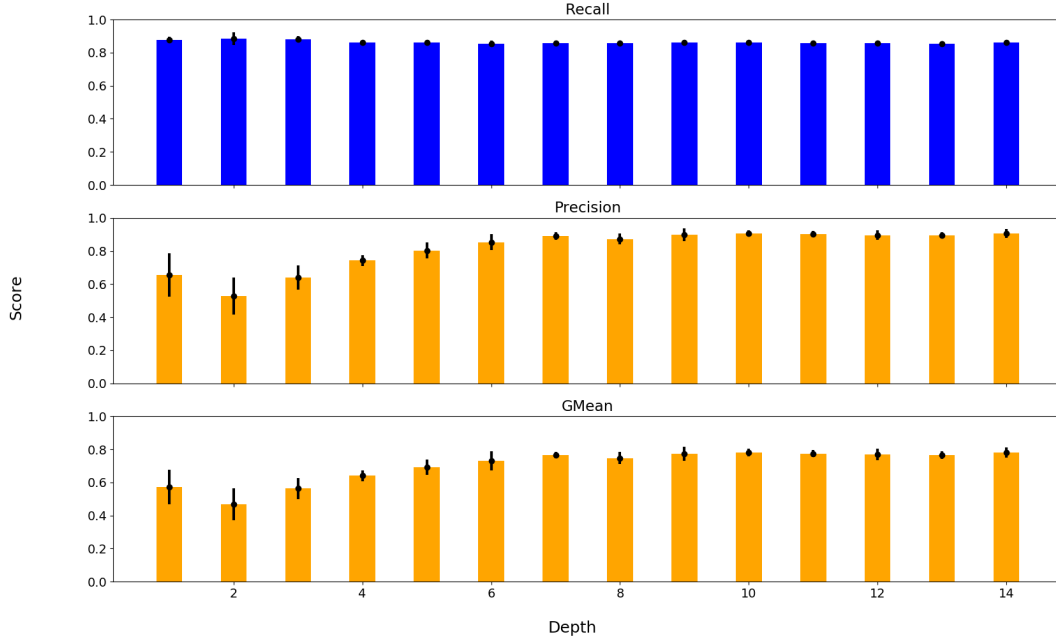


Figure 4.30: Recall, Precision and their Geometric Mean of Rule Fit depending on the maximum depth of the Random Forest (KFold = 5).

-
- Unmix different sources from the original time series;
 - Interpret the sources;
 - Reduce the dimension of the feature space from 250 to 3;
 - Achieve a low reconstruction error.

4.3.4 Random Forest Parameters

To combine the labels with the extracted knowledge initialization features, we use a Random Forest Classifier with the same parameters as studied in the previous case (see section 4.2.4). The classes are again balanced [45] to accommodate for their disequilibrium in the data set.

To define the optimal depth for the RFC, we study how well Rule Fit performs depending on the depth of RFC using a 5-fold cross-validation technique that maintains the ratio of classes for each fold. It seems it has an elbow at depth 7 (see figure 4.30).

Again the stratified 5-fold cross-validation technique averages of the predictive power of both classes indicates a lack of predictive power for the NOK class. (see table 4.3).

OK Recall	NOK Recall
0.822	0.204

Table 4.3: 5-Fold Cross Validation average Recall for OK and NOK processes.

4.3.5 Rule extraction

From the Random Forest Classifier, we extract the best four rules as calculated through their support on the data and their coefficients on the logistic regression classification.

Rule 1: $w_{*,3} \leq 142.72 \ \& \ w_{*,3} > 25.34 \ \& \ w_{*,2} > 172.56$

Rule 2: $w_{*,3} \leq 126.16 \ \& \ w_{*,2} > 352.66 \ \& \ w_{*,2} \leq 389.19 \ \& \ w_{*,1} \leq 363.28$

Rule 3: $w_{*,1} \leq 370.97 \ \& \ w_{*,2} \leq 352.62 \ \& \ w_{*,2} > 305.81 \ \& \ w_{*,1} > 364.87$

Rule 4: $w_{*,1} > 341.35 \ \& \ w_{*,1} \leq 359.75 \ \& \ w_{*,3} > 31.44 \ \& \ w_{*,2} > 334.01$

Following the established interpretation for the OK false discovery rate (FDR) and the OK recall of a rule we want to choose the best rule (see the previous industrial case rule extraction 4.2.5). The scores are presented in table 4.4.

	OK FDR	OK Recall
One Rule	13.8%	100.0%
Rule 1	9.0%	44.2%
Rule 2	12.0%	14.1%
Rule 3	0.0%	5.8%
Rule 4	12.5%	26.9%

Table 4.4: High Pressure Casting: Scoring of the rules on the testing set.

4.3.6 Results

Once again, the One Rule algorithm can only predict OK classes in the testing set. All rules offer a better FDR than the original percentage of negative classes. Rule 3 has a 0% FDR but a low recall rate, which might be too restrictive to the process. On the other hand Rule 1 has the second-best FDR and a very high recall rate .

Matching the processes which comply with one or both rules, we arrive to figure 4.31. It is interesting to note that despite rule 3 having no condition for the third coefficient, it seems to only model processes where this is low. Also, looking at the processes that

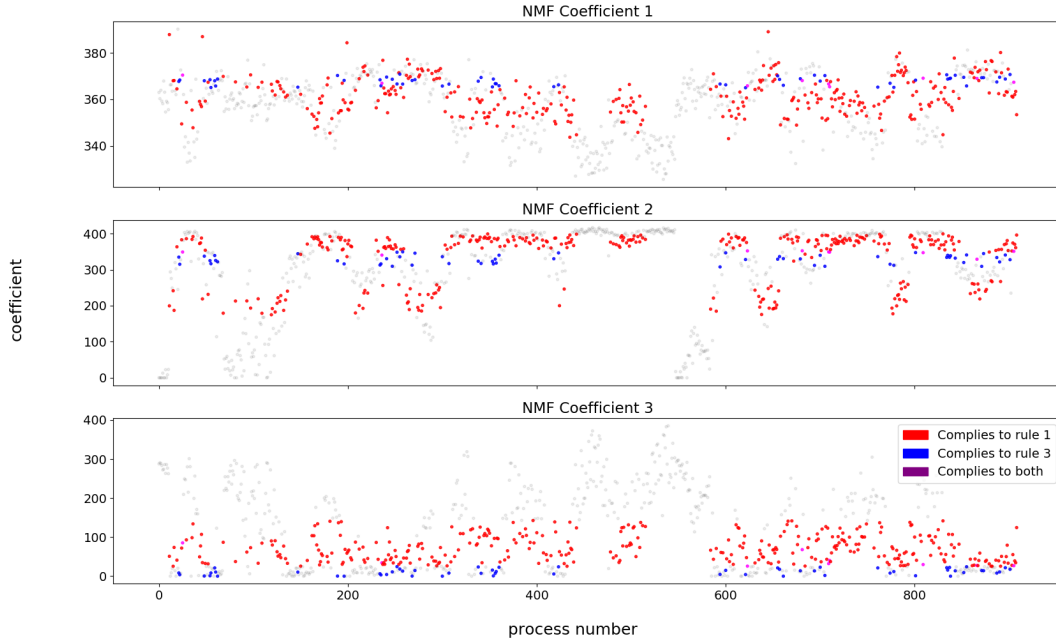


Figure 4.31: Processes that match rule 1, 3 or both.

comply to rule 1 it is clear that almost half of the length of measurements in coefficient 3 is too high. Both rules seem to point to the same but in different subsets of the data.

Using the interpretation that this coefficient models the switching point: a low coefficient for the third base would mean that the switching point happened later in time. A wave may be formed by the liquid if the switching point happens too soon, which changes the way the cavity is filled, producing more defects.

4.3.7 Summary

The original data was represented through a large number of dimensions. Due to this, the data was first reduced to the time frame that was considered useful. Nonetheless, the dimension was still high, at 250 time points. With the non-negative matrix decomposition, we were able to reduce it to 3 dimensions with low reconstruction error.

While we were able to use the non-supervised method in conjunction with the rule extraction, the meaning of the non-negative bases is challenging to understand. The interpretation might be incorrect.

Nonetheless, it provides a possible route for reduction of the defects.

Chapter 5

Concluding Remarks

This thesis has studied a multi-step approach that provides relevant insight to process workers and experts from industrial measurements.

A pipeline for the data was created so that in the end it can be transformed into knowledge. Data was acquired, stored and analyzed. Part of the analysis requires the visualization of the time series and its decomposition into several components with an intuitive meaning. To this end, the non-negative matrix factorization convergence with different initialization matrices was studied. This factorization reduced the data dimension and separated different sources while maintaining interpretable bases. From the new bases, the most essential factors of a process were tied to their physical meaning.

Furthermore, a rule extraction application was created so that from data the process experts already have they can easily extract optimization rules for the processes being analyzed. The Rule Fit algorithm was used in conjunction with the interpretable bases from the decomposition to extract rules under which the process has an optimal scrap rate.

Two case studies on industrial data were run. We were able to collect valuable information on the otherwise largely complex data that is gathered. It was found that even in data with high superposition between classes, feasible rules can be extracted which provide a lower scrap rate.

Even though black-box machine learning has seen an extraordinary amount of research and progress throughout recent history. When the goal is inference, it is also necessary to study methods where the internal work can be interpreted by humans. These methods allow for a symbiotic relationship between humans and algorithms.

5.1 Future Work

Improving this work can be done from the standpoint of the captured data. One approach is to stabilize the process (i.e., no changes to the process parameters while data is captured), to offer stable data to analyze. Conversely, changes may still be done but digitally cataloged to provide the option of separating the data *a posteriori*.

Another interesting approach for future work is the study of tensor matrix factorization

techniques: This makes it so the analysis of data allows for the input of several sensors at once while possibly still maintaining interpretability if all sensors reflect the same physical processes.

Glossary

DMCode Unique identifier of a part. 21

HALS Hierarchical Alternating Least Squares. 10

ICA Independent Component Analysis. iii, iv, 6, 7, 8, 9, 27, 34, 37, 39, 40, 41, 42, 43, 53, 55

NMF Non-Negative Matrix Factorization. iii, iv, 3, 6, 9, 10, 27, 34, 39, 41, 42, 43, 44, 46, 48, 49, 53, 56

NOK Part marked as faulty. iii, iv, v, 20, 25, 27, 34, 35, 36, 44, 46, 47, 50, 51, 57, 58

OK Part marked as good. iii, iv, v, 20, 21, 25, 27, 32, 34, 35, 36, 46, 47, 50, 51, 58

OPC Open Platform Communications. 21, 22

PCA Principal Component Analysis. iii, iv, 5, 6, 7, 8, 27, 34, 39, 41, 42, 43, 53, 54

PLC Programmable Logic Controller. 21

RFC Random Forest Classifier. 13, 14, 57

SVD Singular Value Decomposition. 5, 6, 7

Bibliography

- [1] H. M. Mahendra, G. S. Prakash, Prasad KSK, and Rajanna. Mechanical properties of al6061 - al2o3 metal matrix composite using die casting technique. 2018.
- [2] Mario Hermann, Tobias Pentek, and Boris Otto. Design principles for industrie 4.0 scenarios. *Hawaii International Conference of System Sciences*, 49, 2016.
- [3] Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993.
- [4] Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. 9(3):1350–1371, 2015.
- [5] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou, editors. *Isolation forest*. IEEE, 2008.
- [6] Naphaporn Sirikulviriyaya and Sukree Sinthupinyo, editors. *Integration of rules from a random forest*, volume 6, 2011.
- [7] Houtao Deng. Interpreting tree ensembles with intrees. *International Journal of Data Science and Analytics*, 7(4):277–287, 2019.
- [8] Peter Weiderer, A. M. Tomé, and Elmar Lang. Decomposing temperature time series with non-negative matrix factorization, 2019.
- [9] M. Xarez, P. Weiderer, A. M. Tomé, and E. W. Lang. Extracting rules for process optimization: a case study. *Proc. of RECPAD 2019*, 25:73–74, 2019.
- [10] P. Comon and G. H. Golub. Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE*, 78:1327–1343, 1990.
- [11] P. Comon. Independent component analysis. *Signal Processing*, 36:287–314, 1994.
- [12] D. Lee and H. Seung. Learning the parts of objects by non-negativematrix factorization. *Nature*, 401:788–791, 1999.
- [13] Eugenio Beltrami. Sulle funzioni bilineari. *Giornale di Matematiche ad Uso degli Studenti Delle Universita*, 11(2):98–106, 1873.

- [14] Camille Jordan. Mémoire sur les formes bilinéaires. *Journal de mathématiques pures et appliquées*, 19:35–54, 1874.
- [15] James J. Sylvester. On the reduction of a bilinear quantic of the n th order to the form of a sum of n products by a double orthogonal substitution. *Messenger of Mathematics*, 19(42-46):340, 1889.
- [16] Erhard Schmidt. Zur theorie der linearen und nicht linearen integralgleichungen zweite abhandlung. *Mathematische Annalen*, 64(2):161–174, 1907.
- [17] E. Kogbetliantz. Diagonalization of general complex matrices as a new method for solution of linear equations. *Proc. Intern. Congr. Math. Amsterdam*, 2:356–357, 1954.
- [18] Gene Golub and William Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.
- [19] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, 1970.
- [20] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.
- [21] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(7):498–520, 1933.
- [22] Jeanny Hérault. Réseaux de neurones à synapses modifiables: décodage de messages sensoriels composites par une apprentissage non supervisé et permanent. *CR Acad. Sci. Paris*, pages 525–528, 1984.
- [23] J. Hérault, C. Jutten, and B. Ans. Détection de grandeurs primitives dans un message composite par une architecture de calcul neuromimétique en apprentissage non supervisé. *GRETSI*, pages 1017–1022, 1985.
- [24] Zbynek Koldovsky, Petr Tichavsky, and Erkki Oja. Efficient variant of algorithm fastica for independent component analysis attaining the cramér-rao lower bound. *IEEE Transactions on neural networks*, 17(5):1265–1277, 2006.
- [25] O. Berné, C. Joblin, Y. Deville, J. D. Smith, M. Rapacioli, J. P. Bernard, J. Thomas, W. Reach, and A. Abergel. Analysis of the emission of very small dust particles from spitzer spectro-imagery data using blind signal separation methods. *Astronomy and Astrophysics*, 469:575–586, 2007.
- [26] Karthik Devarajan. Nonnegative matrix factorization: An analytical and interpretive tool in computational biology. *PLoS Comput Biol*, 4, 2008.

- [27] Tuomas Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1066-1074), 2007.
- [28] Zhirong Yang, Zhijian Yuan, and Jorma Laaksonen. Projective non-negative matrix factorization with applications to facial image processing. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(08):1353–1362, 2007.
- [29] Paatero and U. Trapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.
- [30] Ben Noble and James W. Daniel. Applied linear algebra. 2nd ed. Englewood Cliffs, New Jersey: Prentice-Hall, Inc. XVII, 477 p. \$ 11.85 (1977)., 1977.
- [31] Jerome H. Friedman and Bogdan E. Popescu. Predictive learning via rule ensembles. 2(3):916–954, 2008.
- [32] Cecil Hastings, Frederick Mosteller, John W. Tukey, and Charles P. Winsor. Low moments for small samples: A comparative study of order statistics. *Ann. Math. Statist.*, 18(3):413–426, 09 1947.
- [33] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- [34] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.
- [35] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
- [36] Tin Kam Ho, editor. *Random decision forests*, volume 1. IEEE, 1995.
- [37] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
- [38] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.
- [39] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.
- [40] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, Aug 1996.
- [41] Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018.

- [42] Marek Obitko, Václav Jirkovský, and Jan Bezdiček. Big data challenges in industrial automation. In *Industrial Applications of Holonic and Multi-Agent Systems*, pages 305–316. Springer, 2013.
- [43] K. Herzog, G Winter, and et al. The digitalization of steel production. 2017.
- [44] 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.
- [45] Chao Chen, Andy Liaw, and Leo Breiman. Using random forest to learn imbalanced data. *University of California, Berkeley*, 110(1-12):24, 2004.
- [46] Christoph Molnar. Rulefit. <https://github.com/christophM/rulefit>, 2016.

Appendix A

Software

- Python 3
- PostgreSQL 10.5
- KepServerEX
- NMF, PCA and FastICA from sklearn [44]
- RandomForestClassifier, LogisticRegressionCV and LassoCV from sklearn [44]
- RuleFit [46]
- Dash by plotly