A METHODOLOGY FOR IMPROVING MACHINE LEARNING MODELS BASED ON PROCESS KNOWLEDGE AND MODEL ANALYSIS

Felipo Doval Rojas Soares

Doctorate thesis submitted to the Chemical Engineering Program of the Federal University of Rio de Janeiro in partial fulfillment of the requirements for the degree of Doctor of Science.

Advisors: Argimiro Resende Secchi
Maurício Bezerra de Souza Júnior

Rio de Janeiro
July of 2023

A METHODOLOGY FOR IMPROVING MACHINE LEARNING MODELS BASED ON PROCESS KNOWLEDGE AND MODEL ANALYSIS

Felipo Doval Rojas Soares

DOCTORATE THESIS SUBMITTED TO THE CHEMICAL ENGINEERING PROGRAM OF THE FEDERAL UNIVERSITY OF RIO DE JANEIRO IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF SCIENCE.

Advisors: Argimiro Resende Secchi
Maurício Bezerra de Souza Júnior

Aproved by: Prof. Argimiro Resende Secchi
Prof. Maurício Bezerra de Souza Júnior
Prof. Príamo Albuquerque Melo Junior
Dr. Ricardo Emanuel Vaz Vargas
Prof. Flavio Vasconcelos da Silva

RIO DE JANEIRO, RJ – BRASIL
JULY OF 2023

Rojas Soares, Felipo Doval

A methodology for improving machine learning models based on process knowledge and model analysis/Felipo Doval Rojas Soares. – Rio de Janeiro: UFRJ/COPPE, 2023.

XXI, 148 p.: il.; 29,7cm.

Orientadores: Argimiro Resende Secchi

Maurício Bezerra de Souza Júnior

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Química, 2023.

Referências Bibliográficas: p. 113 – 123.

1. Process control. 2. Machine learning. 3. Process monitoring. 4. Industrial data. 5. Fault detection. 6. Oil & gas. I. Secchi, Argimiro Resende *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Química. III. Título.

Acknowledgments

I would like to thank my family. Father, mother, brother, and sister, for the support they have given me, even if we are far now. I would also like to thank Braskem, Radix, and Petrobras, for the projects opportunities and the data.

I also thank my academic supervisors Argimiro and Maurício, without their guidance and support this work would not be possible. And a shout out for my colleagues at LADES, and LASAP, good luck in your future defenses.

In memoriam Dona Lúcia e Seu Soares. You will be missed.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A METHODOLOGY FOR IMPROVING MACHINE LEARNING MODELS BASED ON PROCESS KNOWLEDGE AND MODEL ANALYSIS

Felipo Doval Rojas Soares

July/2023

Advisors: Argimiro Resende Secchi
Maurício Bezerra de Souza Júnior

Department: Chemical Engineering

Data analytics for industrial processes are currently in high demand. A family of non-linear algorithms called machine learning is used to process the data. Deployment of machine learning algorithms is not trivial, and depends on knowledge from both an engineering and a statistical perspective to achieve desirable results. This work seeks to develop a methodology to facilitate and improve data analysis and machine learning applications in chemical processes. This methodology is an iterative procedure validated and guided by data and model analysis, until resulting in a sufficiently good model. Four case studies are presented, one simulation, one using a public dataset and two with industry partners. The simulation case study is the control of a gas lift oil well. The public dataset one is about fault detection in offshore oil wells, while the industrial ones are inference of melt flow index in a polyethylene plant and detection of faults in a compressor dry gas seal. The gas lift oil well was successfully controlled, despite suffering from disturbances never seen before. The fault detection in offshore oil wells achieved better results after applying the methodology, with a median improvement in the F1-score of 44%. The melt flow index inference performed well in validation, in production it had a lower efficacy. However, analysis using operators' knowledge helped the models to achieve good efficacy, with up to 92% "accuracy". The gas seal fault detection worked well in validation, achieving up to 95% accuracy, but the dataset was not sufficient to ensure a good product. Therefore, a framework for building a new more comprehensive

dataset was developed. Overall, the methodology lead to significant improvements on the tasks it was applied to and the insights generated helped in the acceptability of the models by the operators.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

METODOLOGIA PARA MELHORA DE MODELOS DE MACHINE LEARNING BASEADO EM CONHECIMENTO DE ENGENHARIA E ANÁLISE DE MODELO

Felipo Doval Rojas Soares

Julho/2023

Orientadores: Argimiro Resende Secchi
Maurício Bezerra de Souza Júnior

Programa: Engenharia Química

Há uma grande demanda para análise de dados em processos industriais. Para analisar esses dados utiliza-se uma família de algoritmos não-lineares chamada de *Machine Learning*. A aplicação desses algoritmos não é trivial e eles dependem de conhecimento tanto de engenharia quanto de estatística pra se obter resultados desejáveis. Esse trabalho busca desenvolver uma metodologia para facilitar e melhorar a análise de dados e aplicação de *Machine Learning* em processos químicos. A metodologia se baseia em um processo iterativo guiado e validado por análise de dados e de modelos, repetido até criar de um modelo bom o suficiente. Quatro estudos de caso foram realizados, um de simulação, um usando uma base de dados públicos e dois com parceiros da indústria. O estudo de caso de simulação é o controle de um poço de petróleo produzido com *gas lift*. O estudo de caso com dados públicos é a detecção de falha em poços de petróleo. Um dos estudos de caso industriais é a inferência do índice de fluidez de polietileno a partir de dados de planta e o outro é a detecção de falhas em um selo a gás de um compressor. O poço de *gas lift* foi controlado, mesmo contra distúrbios nunca antes vistos. A detecção de falhas em poços mostrou uma melhora mediana de 44% no *F1-score* após a aplicação da metodologia. A estimativa do índice de fluidez teve bons resultados na validação, mas piorou consideravelmente durante a produção. Porém uma análise usando o conhecimento dos operadores mostrou uma boa eficácia, chegando a alcançar uma "acurácia" de 92%. O detector de falhas do selo a gás funcionou bem na validação, alcançando acurácias maiores de 95%. Porém os dados foram insuficientes para uma boa

avaliação da solução, induzindo ao desenvolvimento de um procedimento para a construção de um novo dataset. A metodologia gerou melhoras no desempenho e explicabilidade dos modelos, construindo uma maior aceitação por parte dos operadores.

Contents

List of Figures

List of Tables

List of Symbols

$\alpha^m_{G,b}$	Gas mass fraction at the tubing bottom, p. 135
$\alpha^m_{G,t}$	Gas mass fraction at top of tubing, p. 138
$\alpha_{L,b}$	Liquid volume fraction at bottom of tubing, p. 138
$\alpha_{L,b}$	Liquid volume fraction at top of tubing, p. 138
$\bar{U}_{l,b}$	Liquid velocity at bottom-hole, p. 137
$\bar{U}_{m,t}$	Average mixture velocity of gas phase in tubing, p. 136
$\bar{U}_{sg,t}$	Average superficial velocity of gas phase in tubing, p. 136
$\bar{U}_{sl,t}$	Average superficial velocity of liquid phase in tubing, p. 136
$\bar{\alpha}_{mix}$	Average liquid volume fraction inside tubing, p. 135
$\bar{\rho}_{mix}$	Average density inside tubing, p. 135
ϵ	Rugosity, p. 136
λ	L2 regularization parameter, p. 13
λ_b	Friction factor at bottom-hole, p. 137
λ_t	Friction factor at the tubing, p. 136
$\rho_{G,tb}$	Density of gas at bottom of tubing, p. 138
$\rho_{mix,t}$	Mixture density at top of the tubing is, p. 138
$\hat{y}$	Model output, p. 10
$\varphi()$	Nonlinear expansion, p. 15
$\rho_{G,ab}$	Gas density at the annulus bottom, p. 135
$\rho_{G,in}$	Gas density entering the annulus, p. 135

$\rho_{G,t}$ Gas density at the tubing top, p. 135

ρ_L Oil density, p. 135

b Bias, p. 10

D_t Tubing diameter, p. 136

$f()$ Function, p. 10

F_b Pressure loss from bottom-hole to injection point, p. 137

F_t Pressure loss due to friction, p. 136

g Gravity acceleration, p. 134

$k()$ Kernel function, p. 15

K_{pr} Production valve constant, p. 138

L_a Annulus length, p. 134

L_{bh} Bottom hole length, p. 135

m_1 Gas mass in the annulus, p. 31

m_2 Gas mass in the tubing, p. 31

m_3 Oil mass in the tubing, p. 31

m_4 Exponentially filtered reservoir mass flow, p. 31

M_G Natural gas molecular weight, p. 134

P_{ab} Pressure at the annulus bottom, p. 134

P_{at} Pressure at the annulus top, p. 134

P_{bh} Pressure at bottom-hole, p. 137

P_{GS} Gas source pressure, p. 135

p_i Probability of a sample inside a node belonging to class i, p. 17

P_{res} Reservoir pressure, p. 30

P_{tb}	Pressure at gas injection point, p. 136
P_{tt}	Pressure at the tubing top, p. 135
Q_{out}	Volumetric flow rate of production valve, p. 138
Re_b	Reynolds number of flow at bottom-hole, p. 137
Re_t	Reynolds number of flow in tubing, p. 136
S_{bh}	Bottom hole area, p. 135
T_a	Annulus temperature, p. 134
T_t	Temperature at the tubing top, p. 135
v	Layer output, p. 10
V_a	Annulus volume, p. 134
W	Model weights, p. 10
$w_{G,inj}$	Gas mass flow rate injected into the tubing, p. 31
$w_{G,in}$	Gas mass flow rate injected into the annulus, p. 31
$w_{G,out}$	Gas mass flow rate ejected out of the tubing, p. 31
$w_{G,res}$	Gas mass flow rate produced in the reservoir, p. 31
$w_{L,out}$	Oil mass flow rate ejected out of the tubing, p. 31
$w_{L,res}$	Oil mass flow rate produced in the reservoir, p. 31
w_{res}	Mass flow rate from reservoir to tubing, p. 137
x	Model input, p. 10
y	Target, p. 11

List of Abbreviations

ADAM	Adaptive Moment Estimation, p. 11
BFGS	Broyden–Fletcher–Goldfarb–Shanno, p. 11
DHSV	Downhole Safety Valve, p. 63
EKF	Extended Kalman filter, p. 34
GLOW	Gas Lift Oil Well, p. 29
GOR	Gas Oil Rate, p. 30
HDPE	High Density Polyethylene, p. 74
KDD	Knowledge Discovery in Database, p. 8
LDPE	Low Density Polyethylene, p. 74
ME	Measurement Error, p. 86
MFI	Melt Flow Index, p. 4
MPC	Model Predictive Control, p. 29
NN	Neural Network, p. 9
PCA	Principal Component Analysis, p. 4
PCK	Production Choke, p. 58
PDG	Permanent Downhole Gauge, p. 58
PI	Productivity Index, p. 30
PR_{EoS}	Peng Robinson Equation of State, p. 31
RMSE	Root Mean Squared Error, p. 86
ROC-AUC	Receiver Operating Characteristic, Area Under Curve, p. 126

R^2	R-squared, p. 124
SGD	Stochastic Gradient Descent, p. 11
SMO	Sequential Minimal Optimization, p. 14
TPT	Temperature and Pressure Transducer, p. 58

Chapter 1

Introduction

1.1 Motivation

We are presently living in the 4.0 revolution. There is a huge increase in information available in the industries due to cheaper sensors, improved data storage capabilities and better information technology. This data is called big data. In process system engineering, big data usually means laboratory analysis and sensor data, but can include maintenance reports, economic forecasts, pictures of microorganisms (VON CHAMIER *et al.*, 2021), spectral analysis (SUN and BROCKHAUSER, 2022), etc.

In this new world full of information, it became clear for chemical industry players in Brazil and abroad the potential such data brings. For example: in Petrobras 2022 "connection for innovation" public call, in which start-ups proposed solutions for challenges the company currently struggles with 23 of 35 challenges were related to data science, machine learning and/or artificial intelligence [1]. This shows the demand for these solutions.

This global context leads to an increased demand for data analysis, so engineers and operators can extract useful knowledge from the data. A family of algorithms called machine learning is employed to analyze these huge amounts of data. Machine learning algorithms have non-linear modeling capabilities, few assumptions about the data and ability to handle a considerable features and data samples (HASTIE *et al.*, 2001).

Despite this demand, the chemical industry is lagging behind the implementation of machine learning (SCHWEIDTMANN *et al.*, 2021). Many industrial initiatives face challenges that hinder progress and create subpar products. Knowing how to solve these challenges is important and is the primary focus of this thesis.

[1] Information found in https://www.worldlabs.org/opportunity/petrobras-conexoes-para-inovacao-modulo-startups, acessed March 30th, 2023.

Business solutions for the process industry already exist like Hysys's multivariable interpolators or Microsoft's Azure, but they are not ready-to-use tools and demand statistical expertise alongside process expertise for reliable results. In this work, common pitfalls of data analysis are investigated and it is demonstrated through case studies how to improve process monitoring and control with machine learning.

Currently, most applications of machine learning focus in system monitoring with few applications to process control (VENKATASUBRAMANIAN, 2019). There is a mistrust of automation solutions due to the safety and operability issues that can come with an unreliable control system (MUIR and MORAY, 1996). Most successful applications are in monitoring and fault detection and diagnosis.

Most of the scientific literature on application of machine learning for industrial processes use simulated data or public datasets (NOR *et al.*, 2020), since most of the data, results and models used in these studies are proprietary and economically valuable, therefore not something companies are willing to share publicly due to the risk of losing competitive advantages. Sometimes this information is partially found in patents, for example, KURAMOTO *et al.* (2021). But even in these instances they cannot report on the whole process.

The union of first principle knowledge and machine learning models is widely recognized by many researchers as one of the most significant and promising research areas in chemical engineering for the future. Researchers like SANSANA *et al.* (2021), QIN and CHIANG (2019) and VENKATASUBRAMANIAN (2019) share the viewpoint that knowledge integration presents a compelling challenge and opportunity.

Pure data-based models lack domain knowledge and can make incorrect predictions in not imbued with process knowledge (VENKATASUBRAMANIAN, 2019). For first principle models, assumptions are inherent to them, and these assumptions can give rise to gaps in our understanding of the original system (SANSANA *et al.*, 2021). Models using process knowledge help bridging the gap between both types of models (BIKMUKHAMETOV and JÄSCHKE, 2020).

1.2 Objectives

The objectives of this work are to develop a methodology to facilitate and improve the implementation of machine learning techniques to chemical process monitoring and control, noting and avoiding common pitfalls and delivering meaningful results and insights on the processes. Another objective is to show how to solve some chemical engineering problems with machine learning.

Common pitfalls include data leakage, overly optimistic results due to multimodality, not taking account the measurement noise, among others (HASTIE *et al.*, 2001).

The central hypothesis guiding this thesis is that including engineering and process knowledge in machine learning projects improve the results of the project. This hypothesis is generally accepted by modeling and simulation engineers and researchers (RAWLINGS *et al.*, 2017), but testing it and developing a methodology for its application is valuable for future researchers.

1.3 Thesis structure

In Chapter 2, machine learning and statistical analysis concepts are introduced, along with potential applications. Their limitations and strengths are also presented. There is also a review of currently used machine learning methodologies. In Chapter 3, the methodology is presented, with emphasis on how to use it in process systems. Due to the structure of this thesis, the literature review for each case study will be presented in their respective sections. In Chapter 4, the case study for gas lift oil well control is presented. In Chapter 5 the benchmark 3W dataset (VARGAS *et al.*, 2019) case study, fault detection in offshore oil wells is presented. In Chapter 6 the estimation of the melt flow index of polyethylene is presented and, in Chapter 7, the detection of faults in a dry gas seal system case study is presented. Finally, in Chapter 8, the results are summarized and the conclusions are presented. Appendix A contains a more in-depth explanation of several statistical techniques that appear along the text, while Appendices B to D contain equations, intermediate results and analysis that are important for a more comprehensive thesis.

Chapter 2

Literature review

2.1 Machine learning

There are several overlapping definitions for machine learning, depending on the background or intention of the authors. MITCHELL (1997), wanting to give a more general definition, wrote: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.". MURPHY (2013), using a probabilistic approach and in a big data context, wrote "we define machine learning as a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty (such as planning how to collect more data!)". The machine learning definition used here is "the collection of using various algorithms to teach computers to find patterns in data to be used for future prediction and forecasting or as a quality check for performance optimization", found in BELYADI and HAGHIGHAT (2021).

The difference between machine learning and standard statistical models like Principal Component Analysis (PCA) is up to debate, especially as many machine learning algorithms are extensions from standard statistical models, or have been around for decades, but the lack of quality data and computational power limited their use. There are 3 main types of machine learning problems: supervised, unsupervised, and reinforcement learning (HASTIE *et al.*, 2001).

Supervised problems consist in learning the relationship between two datasets, e.g., correlate polymer melt flow index to operational conditions. It is the most common problem in engineering, as it is easily formulated and has straightforward metrics. Transforming other problems in supervised problems is a common procedure when feasible. The most common supervised problems are regression and classification. Regression is estimating a number from a dataset

and classification is estimating a category from a dataset (HASTIE *et al.*, 2001).

Unsupervised problems consist in finding underlying structures in one dataset, e.g., finding abnormalities in process operation. While less common than supervised problems, unsupervised analysis is useful to develop a better understanding of a problem. Traditional statistical analysis like autocorrelation gives a better understanding of process dynamics, and machine learning approaches, like kernel PCA, can be used to identify unexpected correlation between sensors. However, unsupervised problems are rarely straightforward, demanding specialist confirmation of results and more throughout tests. The most common unsupervised problems are dimensionality reduction and clusterization. Clusterization is aggregating the samples and dimensionality reduction is aggregating/removing the features (HASTIE *et al.*, 2001). Recently there have been developments on **semisupervised** problems, that is a bit of both supervised and unsupervised learning. They are classification problems with plenty of unlabeled samples, and use iterative procedures to automatically label the data.

Reinforcement learning is a type of problem in which the algorithm interacts with an environment and learns to minimize a cost function. It is usually used for control in which the inputs, disturbances and transients are stochastic and not tabular, e.g., self-driving cars where inputs are cameras recording, disturbances are pedestrians and the whole process is transient. It generally lacks many assurances that normal control theory provides like optimality, constraint adherence and stability (SUTTON and BARTO, 2018), and demands much more data than available for industrial applications. Standard procedure to reinforcement learning problems is to transform them into supervised learning problems (HASTIE *et al.*, 2001).

Training or **fitting** is adjusting the parameters of a model to the data. **Validation** or **testing** is presenting new data to the model to evaluate its ability. In this work validation will be used when evaluating the model to available data and testing when using data only available during implementation or never seen before data. **Hyperparameters** are the model parameters that are not fitted to the data, e.g., regularization penalties (BAUGHMAN and LIU, 1995). **Tuning** is hyperparameter optimization and is a difficult task, there is no standard approach on how to do it. Recent works use Bayesian optimization, grid or random search or manual tuning (HUTTER *et al.*, 2018). **Overfitting** is when the machine learning model memorizes the data and not learn the underlying relationship in the data. **Generalization** is how well the model works when presented with new data. **Information leakage** or **data leakage** is when the training dataset contains information about the test or validation set, resulting in overly optimistic models. **Regularization** is a penalty or restraint to model complexity, and is added to

machine learning models to avoid overfitting and improve generalization. **Features** or **dimensions** are the characteristics of the data and in a table are usually presented as columns. Rows of the table are known as **samples** or **data points** (HASTIE *et al.*, 2001) (BAUGHMAN and LIU, 1995).

In process engineering the first data based models were empirical correlations. Empirical correlations have been used for decades and have assumptions about the general shape of the relationship between variables. For example the famous Antoine equation and its extensions assume that the log of the saturation pressure is proportional to the inverse of the temperature, an assumption that can be derived from thermodynamic equations (WAGNER, 1973).

Empirical correlations work well when the relationship between variables is smooth, not many features are used and general process uncertainty is higher than the correlation error. For many properties, empirical correlations are unavailable (JAMES *et al.*, 2014), and developing new empirical correlations or improving known ones are an active area of research.

Another popular and well established class of data-based models are linear dynamic models, like transfer functions, state space models and autoregressive models. They have been used for decades by several fields like econometrics and social studies, as they are easy to interpret and have a robust statistical background, and have important results about several topics of interest in process engineering, like sampling time and informative excitation (THIL and GILSON, 2005), which is not commonly found in machine learning literature.

The main limitation of data-based models is their inability to extrapolate outside the training region. For empirical correlations, while accuracy is not guaranteed outside their validity region, it can be assumed that the property being estimated at least follows a trend. For machine learning models, especially black box models, model behavior outside training region cannot be predicted and is not reliable (HASTIE *et al.*, 2001).

Testing the performance of machine learning models demands special attention. Since they have more capacity for bias in the bias-variance trade-off and therefore capacity for overfitting, most machine learning models will present better metrics in the train dataset than in the test or validation dataset, therefore results should be presented in terms of the latter datasets. There are couple of ways to estimate those metrics, the most popular are holdout or K-fold.

Holdout is straightforward. It consists in just separating the dataset in train, test and validation. The main disadvantage is that there is the possibility of picking "well-behaved" regions of the dataset for test, or vice-versa, creating inaccurate estimations of effectiveness of the model. On the other hand it is relatively fast. It is usually recommended for big data and deep learning, where

training can take days.

K-fold and its variants are by far the most popular method. It consists in separating the dataset like holdout, train and test the model effectiveness, then retrain the model and retest the model, but using different data in each set. This procedure is done K times, each different repetition is called a fold, and the final metric is given by the folds mean. It gives a better view of the model effectiveness in general as it evaluates the whole dataset. However it has high time costs. As most tasks take less than a couple of minutes fit, it is usually fine, but it can build-up if testing several models (HASTIE *et al.*, 2001).

2.2 Machine learning methodologies

There are plenty of researches regarding application of data-based modeling in engineering problems. Most of them are dedicated to system identification of linear models. Methodologies for system identification generally follow as: generate a dataset through a specially designed identification experiment; select a set of candidate models, fit and choose the best model, and finally validate the model (LJUNG, 1986). There is a greater focus on the fitting algorithms and convergence; Fourier and spectral analysis; and using prior knowledge.

Regarding more black-box modeling methodology, some include only basic procedures like normalization and standardization, without which even some classical algorithms, like PCA, does not work. CHIANG *et al.* (2001) is a seminal book regarding data-based modeling applied to industrial processes. It focuses on linear and time-delayed models with some mentioning of data treatment and analysis. For the case studies, it used the Tennessee Eastman model as it is a published problem easily available for any reader. However no data treatment was applied in the case study and it used no feature selection.

HARROU *et al.* (2021) is a more recent book and while it does not do an in-depth explanation of data analysis, it includes more advanced techniques. For an abnormal ozone measurements detection problem, it used the correlation coefficients matrix for feature aggregation and autocorrelation functions to detect cyclic behavior. In a decentralized wastewater treatment plant monitoring problem, the variables were selected by expert recommendation. Descriptive statistics and correlation matrix were used for data analysis.

BELYADI and HAGHIGHAT (2021) focused on applying machine learning using standard Python libraries, but partially dedicates a section to machine learning workflow and data treatment, including data gathering and integration. The authors highlight the importance of institutional data infrastructure and availability, which is essential for model deployment and acceptance. They

also recommend feature selection by expert recommendation. The book uses several datasets from the oil and gas industry to illustrate their results, and those datasets are available online.

Regarding application of first principle engineering knowledge in Machine learning frameworks, BIKMUKHAMETOV and JÄSCHKE (2020) introduces several methodologies that combine engineering modeling and machine learning, with the purpose of improved explainability and accuracy. Some methodologies used feature engineering, others used a residual model given by the difference of a phenomenological model and the data available and some even used both. They also recommended a feature analysis method, partial dependency plot to improve the understanding of the model inner workings.

This type of methodology study is more common in the data science field. Several methodologies were developed over the decades. Their general workflow are similar to the one presented here, in fact most of them are similar with each other but with significant differences on the emphasis of the workflow. The work of FAYYAD *et al.* (1996) is one of the first methodology studies, and focused in data mining and Knowledge Discovery in Database (KDD), which are related fields to machine learning in which the goal is extracting information from the dataset, not generating a model; although generating a model can be a deliverable of the process.

The framework works as following: the first step is understanding the application domain and the relevant prior knowledge and with those identify the client's KDD objective. Then, there is a data selection and sampling step, in which the relevant data is extracted from the database forming a target dataset. On this dataset, a data cleaning and pre-processing step is performed to remove outliers, and treat noise and either input or eliminate missing data. The next step is data reduction and projection, to find the best ways to represent the data using transformations and dimensionality reduction techniques. The fifth step is matching the goals found in the first step to an appropriate method like classification or regression. The sixth step is finding a particular algorithm(s) and hypothesis that can be used to find the data patterns. The seventh step is the data mining itself, followed by a interpreting mined step, in which the extracted patterns are visualized and evaluated if they make sense. The final step is acting on the mined knowledge, which may be generating a report, developing new business practices or utilizing a model.

The framework presented in FAYYAD *et al.* (1996) introduces the interactive and iterative nature of the methodology, with previous steps constantly being revisited depending on the knowledge and results acquired from later steps. All nine steps indicated in this framework are found in later ones, with differ-

ent names or sometimes one step is fused with another. For example: Cross-Industry Standard Process for Data Mining, presented in CHAPMAN *et al.* (2000), expands the business understatement step with specific questions that should be answered before proceeding and add an exploratory analysis before the cleaning and pre-processing of the data. It also fuses the fifth and sixth step from KDD methodology in a step simply called "Modeling".

Another popular framework is the Team Data Science Process, presented in SEVERTSON *et al.* (2017). It is more focused on commercial applications, and resembles an instruction manual for an analyst with a Microsoft Azure toolkit doing a data science project than a scientific paper. It fuses the data acquisition, selection and treatment in one step, and adds a customer acceptance step highlighting the importance of future maintenance and support of the data science solution developed. It is focused in Azure infrastructure, and includes a GitHub page with examples on how to organize documentation and code.

The methodology developed and presented in this work will be similar to the ones presented here. The main difference is the emphasis in model analysis and in the agreement with process knowledge. Additionally, in this methodology operators' and engineers' input and feedback was always sought for and welcomed when available, while this kind of interaction is not as explicitly evident in the other methodologies found in the literature.

2.3 Neural networks

Neural network (NN) is a machine learning algorithm that has the interesting property of being an universal approximator given enough data and an appropriate hidden layer size (BAUGHMAN and LIU, 1995). NNs are algorithms loosely inspired on how the human brain works. They are composed of neurons and layers. Each neuron takes inputs, does a weighted sum of those inputs then a function is applied to this sum, Equation 2.1. The output of the neurons in a layer is used as input to the next layer. Neurons in parallel make a layer, 2 or more layers in series make a neural network. A neural network with a single layer would be equal to a generalized linear model.

$$\bar{v_i} = f(\bar{\bar{W}}^T \times \bar{x} + \bar{b_i}) \tag{2.1}$$

$$\bar{y}_{nn} = f(\bar{\bar{W}}^T \times \bar{v_i} + \bar{b_j}) \tag{2.2}$$

where $\bar{x}$ is the input vector, $\bar{\bar{W}}$ is the weight matrix of the i-th layer, $\bar{b_i}$ is the bias, v_i is the i-th layer output vector, $f()$ is the function applied to the layer, $\bar{y}_{nn}$ is the NN output. An example of neural network, with 3 inputs 4 neurons in the

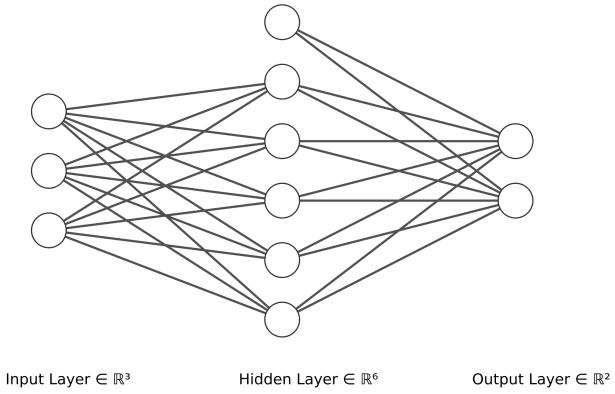


Figure 2.1: Neural network example.

hidden layer, and two outputs can be seen in Figure 2.1.

The bias can also be interpreted as a neuron that is always outputing 1. The function $f()$ is also called an activation function. Popular activation functions include Relu, hyperbolic tangent and sigmoid (MONTAVON *et al.*, 2012). The last layer is known as output layer and the previous layers are known as hidden layers.

$$Relu(x) = max(0, x) \tag{2.3}$$

$$sig(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} = 2sig(2x) - 1 \tag{2.5}$$

During training, the optimization algorithm minimizes an objective function. In modern NN formulations any differentiable objective function can be used, as automatic differentiation algorithms make switching between objectives easy (BERGSTRA *et al.*, 2010). The objective function depends on the NN purpose and data characteristics. For regression with data containing few outliers, mean squared error (MSE), Equation 2.6 is used, while for data with a high amount of outliers mean absolute error, Equation 2.7 is preferred. For classification, cross entropy, Equation 2.8, is generally chosen as it gives a probabilistic interpretation, but hinge loss, Equation 2.9, has been used to create comparisons with Support Vector Machines (MONTAVON *et al.*, 2012).

$$MSE(\hat{y}_{nn}, y) = \frac{1}{n} \cdot \sum_{i=1}^{n} ((\hat{y}_{nn,i} - y_i)^2) \tag{2.6}$$

$$MAE(\hat{y}_{nn}, y) = \frac{1}{n} \cdot \sum_{i=1}^{n} (abs(\hat{y}_{nn,i} - y_i)) \tag{2.7}$$

$$CE(\hat{y}_{nn}, y) = \frac{1}{n} \cdot \sum_{i=1}^{n} (-y_i \cdot ln(\hat{y}_{nn,i}) - (1 - y_i) \cdot ln(1 - \hat{y}_{nn,i})) \tag{2.8}$$

$$Hinge\ loss(\hat{y}_{nn}, y) = \frac{1}{n} \cdot \sum_{i=1}^{n} max(0, 1 - y_i \cdot \hat{y}_{nn,i}) \tag{2.9}$$

There are many regularization techniques that can be added to NNs. The most popular are weight shrinkage, dropout and noise addition. Weight shrinkage adds a regularization term to the objective function that penalizes huge weights, making NN's output smoother. The main types of weight shrinkage are L2, also known as weight decay, and L1, given by Equations 2.10 and 2.11 respectively. L2 regularization tends to make optimization better conditioned, while L1 tends to create sparse weight matrices, which is interesting for computationally efficient implementations. Dropout consists in randomly deactivating some neurons during training, forcing different neurons to learn similar relationships in the data. Noise addition inserts uncertainty in the model, hampering data memorization that leads to overfitting (BISHOP, 1995).

$$L2 = ||\bar{\bar{W}}||_2 \tag{2.10}$$

$$L1 = ||\bar{\bar{W}}||_1 \tag{2.11}$$

Training is done by some kind of gradient following algorithm, as the model is differentiable. Common optimization algorithms are Broyden-Fletcher-Goldfarb-Shanno (BFGS), Adaptive Moment Estimation (ADAM) and Stochastic Gradient Descent (SGD). They are described in more detail in the following paragraphs.

BFGS is an optimization algorithm that uses a rank-2 matrix to update the hessian estimate. As a quasi-Newton method it converges much faster than the others. However, it assumes that the objective function is locally quadratic and it takes huge steps compared to ADAM and SGD, which is not desirable as the objective surface tends to be non-smooth (FLETCHER, 1987). Keeping the inverse hessian approximation in memory has a quadratic cost in BFGS original formulation, it requires a $n \times n$ matrix. Low-memory variants store some vectors that represent the inverse hessian approximation updates, instead of the approximation itself, making the memory requirement linear (LIU and NOCEDAL, 1989).

SGD is gradient descent applied to each training sample individually. It is called "stochastic" because the sample being used for training is selected randomly. It has two advantages: low-memory demand and each gradient update changes considerably the error surface, making landing in a bad local minima difficult. On the other hand, constantly changing the error surface may lead the algorithm to not converge at all. Generally SGD is done in mini batches,

leveraging modern hardware parallelization capabilities and smoothing the error surface (ROBBINS and MONRO, 1951).

ADAM is a gradient descent algorithm with moment that employs an adaptive learning rate. The learning rate is divided by the square root exponential moving average square gradients, and the local gradient is replaced by the exponential moving average of the gradients. It is very popular as the adaptive learning rate makes it more robust to different hyperparameters (KINGMA and BA, 2014).

The main limitation regarding NN is that they demand a lot of quality data, but many times this amount of data is unavailable. Another limitation of NN is that they have too many hyperparameters, and as they become deeper those hyperparameters become increasingly important. It is also not clear how they affect the model, so tuning is almost a brute force effort, with the search methods cited in Section 2.1 having to explore a wide region. Another problem is that behavior outside training region is unpredictable.

2.4 Support vector machines

Support Vector Machine is an algorithm that seeks to construct a hyperplane that separates two classes of data, seeking the maximum margin between classes, as can be seen in Figure 2.2. It was originally developed in 1963 by Vladimir N. Vapnik and Alexey Chervonenkis (HASTIE *et al.*, 2001). It is popular due to the kernel trick, that allows the model to efficiently map the data to a higher dimensional space. It also has the interesting property of ignoring data points that are already correctly classified, creating a robust classification. Its name come from the support vectors, the data points outside the correct margin that are used in fitting the algorithm. Equation 2.12 describes the algorithm.

$$\hat{y}_{svm} = \bar{W}^T \times \bar{x} + b \tag{2.12}$$

$$\begin{cases} x \in class\ 1, \text{if } \hat{y}_{svm} \geq 0 \\ x \in class\ 2, \text{otherwise} \end{cases}$$

The objective function being optimized to fit the SVM is a dual objective unconstrained function given by Equation 2.13. It has one term seeking the separation of classes and the other seeking the size of the margin. A SVM with a margin too wide will have many data points inside it, while a SVM too focused in separating the classes will be sensitive to outliers. The margins' separation is given by the hinge loss term and the size of the margin is given by the L2 regularization term. To balance both objectives the hyperparameter λ is used. λ

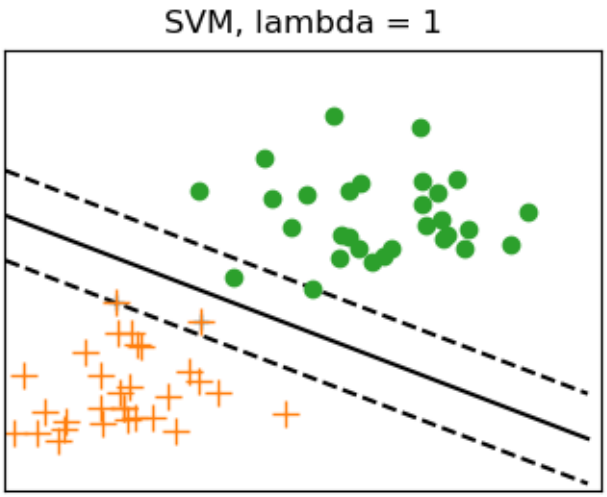


Figure 2.2: Support vector machine example, the dashed lines are the margin.

is a term for how much regularization contributes for the objective function. It must be noted some implementations use a term C instead. This C term is for how much the hinge loss contribute for the objective function. As can be seen in Figure 2.3, a higher λ produced a wider margin for the same data.

$$\min_{W,b} \frac{1}{n} \cdot \sum_{i=1}^{n} max(0, 1 - y_i \cdot (\bar{W}^T \times \bar{x}_i + b)) + \lambda \cdot ||\bar{W}||_2 \tag{2.13}$$

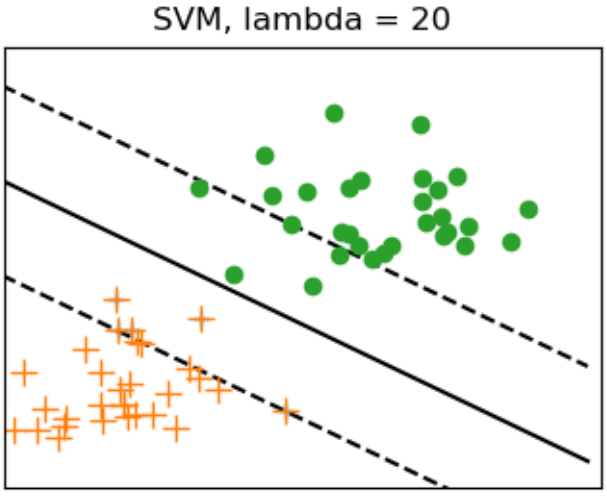


Figure 2.3: Support vector machine example, more regularized and with wider margin.

SVM fitting is rarely done by minimizing Equation 2.13 directly. Instead, it is transformed into a constrained optimization problem by adding the slack

variable $\xi_i = max(0, 1 - y_i \cdot (\bar{W}^T \times \bar{x}_i + b))$, Equation 2.14; then is reformulated in the Lagrangian dual form, which is a quadratic problem, Equation 2.15. In the dual formulation, $c_i = 0$ if the point is at the correct side of the margin. In other words, if the data point is not a support vector it is not used in the optimization. W can be obtained from a linear combination of the support vectors.

$$\begin{aligned} \min_{\bar{W},\xi,b} \quad & \frac{1}{n} \cdot \sum_{i=1}^{n} \xi_i + \lambda \cdot ||\bar{W}||_2 \\ \text{s. t.} \quad & y_i \cdot (\bar{W}^T \times \bar{x}_i + b) \geq 1 - \xi_i \\ & \xi \geq 0 \end{aligned} \tag{2.14}$$

$$\begin{aligned} \max_{c} \quad & \sum_{i=1}^{n} c_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i=1}^{n} y_i \cdot c_i \cdot (\bar{x}_i^T \times \bar{x}_j) \cdot y_j \cdot c_j \\ \text{s. t.} \quad & \sum_{i=1}^{n} c_i \cdot y_i = 0 \\ & 0 \leq c_i \leq \frac{1}{2 \cdot n \cdot \lambda} \\ & \textit{where } \bar{W} = \sum_{i=1}^{n} c_i \cdot \bar{x}_i \cdot y_i \textit{ and } b = y_m - \bar{W}^T \bar{x}_m \end{aligned} \tag{2.15}$$

The dual is usually solved with a specialized algorithm called sequential minimal optimization (SMO). It was developed in 1998 by John Platt while working at Microsoft. While any quadratic programming algorithm works, SMO is more efficient. SMO is based in separating the dual problem into smaller subproblems that can be analytically solved (PLATT, 1998).

The previously mentioned kernel trick was developed in 1992 (BOSER *et al.*, 1992). It is based on efficiently expanding the dataset in another feature space. The kernel trick is possible due to the $(x_i^T \times x_j)$ term. Suppose a $\phi(x_i)$ expansion and a kernel function such $k(x_i, x_j) = \phi(x_i)^T \times \phi(x_j)$. It is only needed to replace it with $k(x_i, x_j)$, so there is no need to actually expand the dataset, only finding the output of the kernel function on the support vectors. Equation 2.15 becomes Equation 2.16.

$$
\begin{aligned}
\max_{c} \quad & \sum_{i=1}^{n} c_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i=1}^{n} y_i \cdot c_i \cdot k(\bar{x}_i \bar{x}_j) \cdot y_j \cdot c_j \\
\text{s. t.} \quad & \sum_{i=1}^{n} c_i \cdot y_i = 0 \\
& 0 \leq c_i \leq \frac{1}{2 \cdot n \cdot \lambda} \\
& where\ b = y_m - [\sum_{i=1}^{n} c_i \cdot y_i \cdot k(\bar{x}_i, \bar{x}_m)] \\
& \hat{y}_{SVM} = \sum_{i=1}^{n} c_i \cdot y_i \cdot k(\bar{x}_i, \bar{x}) + b
\end{aligned}
\tag{2.16}
$$

Popular kernels include Radial Basis Functions (RBF), polynomial and sigmoid, seen in Table 2.1. Each one of them carry additional hyperparameters that increase how well a SVM can be tuned. γ can be interpreted as a scale parameter and r as a bias.

Table 2.1: SVM kernels.

kernel	equation	hyperparameters
RBF	$exp(-\gamma \|\|\bar{x}_i - \bar{x}_j\|\|)$	γ
Polynomial	$(\gamma(\bar{x}_i^T \times \bar{x}_j) + r)^d$	$\gamma,\ r,\ \ d$
Sigmoid	$tanh(\gamma(\bar{x}_i^T \times \bar{x}_j) + r)$	γ, r

SMO is slow when a large quantity of data is fitted. It scales between $O(n_{features} \cdot n_{samples}^2)$ and $O(n_{features} \cdot n_{samples}^3)$ depending on the dataset. Recently subgradient (SHALEV-SHWARTZ *et al.*, 2011) or coordinate descent (JUI HSIEH *et al.*, 2008) methods have been gaining prominence. Subgradient methods solve the original equation and coordinate descent methods solve the dual. They accept modifications of the objective, including squaring the hinge loss for better behaved subgradients and L1 regularization for sparse weights and feature selection. However, neither of them are quadratic programming solvers, so convergence is not guaranteed.

Additional strategies are necessary to apply SVM for native multiclass classification. One-vs-rest is based on using several SVMs, each classifying one class against all others and evaluating which one is further from the hyperplane. For k classes this method creates k classifiers. One-vs-one is based on using several SVMs, each classifying one class against other and evaluating which class is more commonly classified. For k classes this method creates $k(k-1)/2$ classifiers.

SVM can also be extended for regression using Equation 2.17. This algorithm is also called support vector regression. The idea is minimizing the margin while trying to keep most points inside it, given a tolerance ε. ε does not appear in

the classification, as classification is already encoded in [-1,1], and must be set depending on the range of the output. The model ignores the points inside the tolerance. All other algorithm features discussed previously like the kernel trick and optimization techniques also apply for regression.

$$\begin{aligned} \min_{\bar{W},\xi,b} \quad & \frac{1}{n} \cdot \sum_{i=1}^{n} (\xi_i + \xi_i^*) + \lambda \cdot ||\bar{W}||_2 \\ \text{s. t.} \quad & y_i - (\bar{W}^T \times \bar{x}_i + b) \leq \epsilon + \xi_i \\ & -y_i + (\bar{W}^T \times \bar{x}_i + b) \leq \epsilon + \xi_i^* \\ & \xi \geq 0 \end{aligned} \tag{2.17}$$

2.5 Decision trees

Decision trees are algorithms that seek to separate the data using sequential if-else conditionals, as can be seen in Figure 2.18. They are mainly designed for classification, but can be used in regression by joining the continuous numbers into bins. Decision trees have the advantages of being easily explainable, fast to train and having a probabilistic interpretation. On the other hand: they are discontinuous, more specifically: piecewise constant, and prone to overfitting without strong regularization (HASTIE *et al.*, 2001).

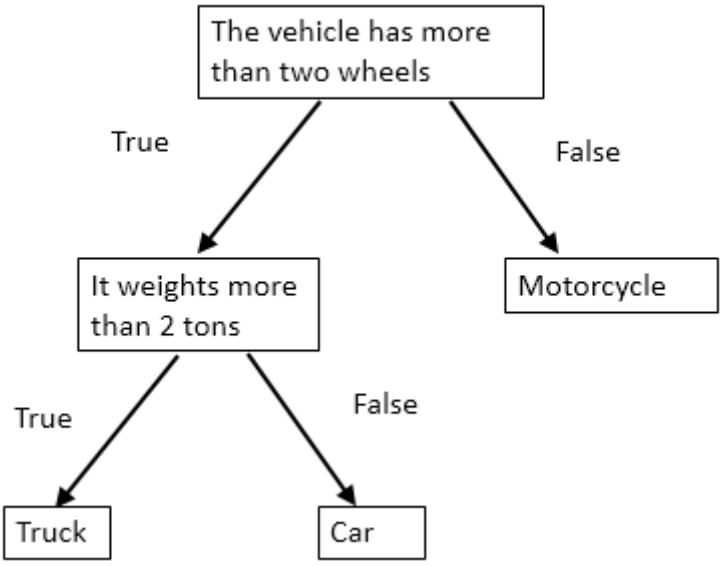


Figure 2.4: Decision tree example (author).

Each step of the tree is called a node. If there is a conditional in the node it is called an interior node and the conditional is called split. If the conditional is true the following node is called the left child, the false following node is called

right child. If there is no conditional inside a node it is called a leaf and the class corresponding to that leaf is the model output. The depth of a tree is the maximum number of sequential splits (ROKACH and MAIMON, 2014).

Decision trees are trained by seeking the best split inside each node that minimizes an impurity metric, according to the subset inside the node. The two most common metrics are Gini impurity and entropy. They produce similar results, although entropy is slightly slower due to the logarithm computation (RAILEANU and STOFFEL, 2004), using one or the other is a matter of personal preference.

Gini impurity is defined by the probability of a sample being randomly drawn from the subset times the probability of the sample being incorrectly classified. It is given by Equation 2.18. Entropy is a less straightforward concept. It is derived from information theory and is given by Equation 2.19.

$$Gini\ impurity = \sum_{i=1}^{J} p_i \sum_{k \neq i} p_k = 1 - \sum_{i=1}^{J} p_i^2 \tag{2.18}$$

$$Entropy = -\sum_{i=1}^{J} p_i log_2(p_i) \tag{2.19}$$

where p_i is the chance of a random sample from class i being drawn. Both approaches are minimum when all data points inside a node belong to the same class (ROKACH and MAIMON, 2014).

One interesting property of decision trees models is feature importance extraction. As each node only evaluates one feature, how much the impurity is reduced can be interpreted as how important is the feature. Unlike feature selection by L1 regularization done with SVMs or generalized linear models, this has no linear assumption and in case of colinearity it splits the relevance between the features.

Regularization in decision trees is done mainly by restricting the tree size, e.g., forcing the number of leaves or the depth of a tree to be below a certain threshold, or demanding that internal nodes or a leaves contain a minimum number of samples.

Most applications do not use regular trees by themselves, but rather forests of trees. A forest of decision trees is the combination of several small/medium sized decision trees, up to dozens of thousands, each trained with a modified version of the training data. Each tree learns a different aspect of the relationship between datasets, and may overfit to specific points, but the assemble of trees reduces the overfitting (HASTIE *et al.*, 2001).

There are different types of forests depending on how the training data is

modified. The two most popular are random forests and boosting forests. Random forests modify the training data by removing samples randomly. Features can also be randomly removed. Later the average of the result is used to predict the model. This procedure is called bagging or bootstrap aggregating. It is not exclusive of decision trees but mostly used by them due to short training time. To give an engineering analogy, it is similar to repeating a laboratory analysis several times with different samples and take the average of results.

Boosting forests modify the data by reinforcing samples that the model gets wrong. They can boost with either the gradient of a loss, or increasing the sample weight. The final prediction is a combination of each individual tree. Boosting forests are more powerful, as sequentially improving prediction eventually will remove any residual of the training dataset, but at the same time more susceptible to overfitting (HASTIE *et al.*, 2001).

A summary of each method and their advantages and disadvantages are given in Table 2.2.

Table 2.2: Summary of each algorithm advantages and disadvantages.

Algorithm	Advantages	Disadvantages
Neural network	Fast evaluation Very powerful Continuous and differentiable	May demand a long time to train Demand many data samples Not interpretable Strong stochastic factor
Support vector machine	Robust to classification errors. Straightforward hyperparameters	Demand a long time to train Only interpretable in the linear case
Decision tree and forest	Fast training and evaluation. Easily interpretable Robust for single outliers	Prone to overfitting Piecewise constant output Regression has a tendency to the mean

Chapter 3

Proposed methodology

The purpose of developing a methodology is to facilitate and improve the application of machine learning in chemical processes, with focus in monitoring and control. It is expected that researchers produce faster and better results following this methodology. The potential for improvement of machine learning solutions by adding process knowledge has been researched before (VENKATASUBRAMANIAN, 2019) (JOE QIN *et al.*, 2021), but a methodology would facilitate their application to new tasks and processes.

Methodology study in Machine learning is not a novelty by itself (MOHRI *et al.*, 2012), however chemical industry has several specific challenges like sensor faults; tight environmental, safety and economical constraints and unstructured manually compiled databases. There are methodology studies for chemical processes, while more focused on more traditional statistical models (FORTUNA *et al.*, 2006). This work develops a methodology more suitable to machine learning, especially as the field advanced considerably in this past decade (QIN and CHIANG, 2019).

The methodology being proposed is summarized in the following steps and in Figure 3.1. The details of each step will depend on the task, dataset and the available computational resources.

- Data treatment.
  - Data reformat.
  - Data quality evaluation.
  - Data cleaning.
- Data analysis.
  - Unsupervised analysis.
  - Data selection.

- Modeling.
  - Initial models.
  - Model analysis.
  - Deployment-refine cycle.
- Follow-up with operators.

All of them build up from the methodologies and frameworks presented in the previous chapter. Comparing with the KDD framework: Data reformat and data quality evaluation are analog to the data selection and sampling step of the KDD framework. The data cleaning step here is similar to the data cleaning and pre-processing step in the KDD framework, although with a enhanced focus in process data. The unsupervised analysis and Data selection steps used here are similar to the the data reduction and projection steps of the KDD framework.

The fifth to seventh steps of the KDD framework do not have an exact and unique correspondence, but are handled in the Deployment-refine cycle, Data cleaning, model analysis and Data section steps. The final step of the KDD framework, acting on the mined knowledge, here is the follow-up with operators step.

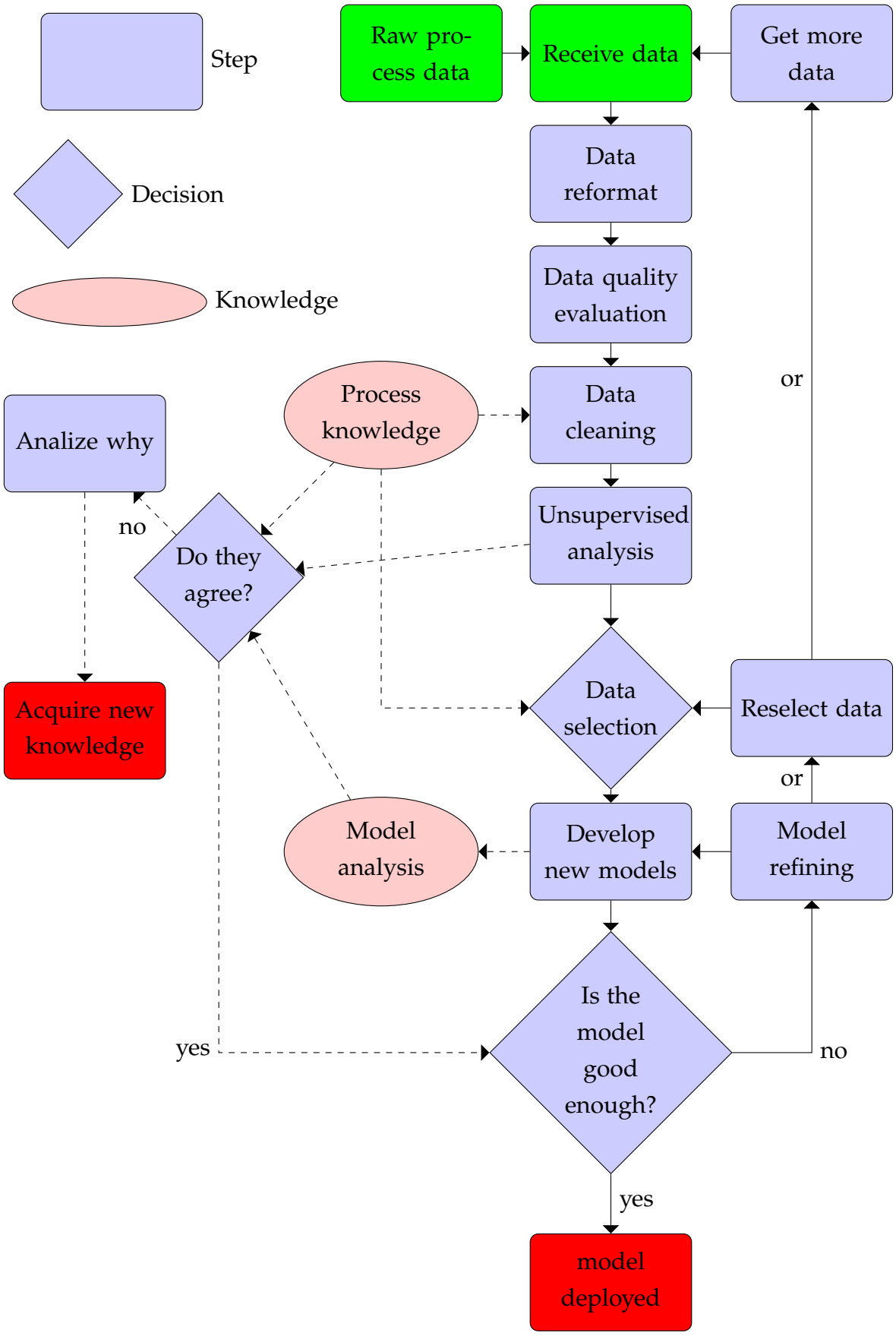


Figure 3.1: Flowchart of the proposed methodology. The solid lines represent the usual steps found in the literature, the dashed lines represent the analysis recommended in this work.

3.1 Data treatment

The general idea is: Receive the data, in whatever format the data management system outputs. If necessary, reformat it in such a way that less memory is consumed, and visualization and data transfer is easier. Data reformat may include joining different datasets, eliminating some columns, etc. Usually the desired result is a single table. It is recommended to save the resulting dataset in a human-readable format, like a text file or Excel spreadsheet if the dataset is small, but saving in computer-friendly formats like binary files is also possible to facilitate loading the data in the memory and processing.

The data reformat step is given additional importance in this methodology compared to the other methodologies in literature because of the lessons learned during the case studies with huge industrial datasets. Reformatting the data was a recurring and time consuming task, so time should be allocated during planning to account for it.

With a properly formatted dataset, an initial quality assessment can be made. Check if all sensors have the same timestamp and if they were concatenated properly. Check if any sensor has too many missing values or unfeasible values, e.g., negative pressures. Ideally the data management system will have a sensor quality descriptor, but this is not always set up. Check for noise or lack of noise. In summary:

- Aspects to check during data quality evaluation.
  - If the data is properly concatenated.
  - Consistent timestamps.
  - Quantity of missing data.
  - Physical consistency of the data.
  - Frozen/interpolated values.
  - Quantity of noise.

Once the data quality is properly evaluated, some technique for data cleaning may need to be applied: Data cleaning may mean removing sensors and data periods too damaged to be useful, but it can also mean data imputation or noise reduction technique. If a sensor is deemed exceptionally important for operation, like pressure in a gas phase reaction, but appears to have spent some weeks not working, it may be better to input the data for this specific period rather than removing it from the dataset. Particularly noisy measurements like gas flow may need to be smoothed. Because of these details, data cleaning should be carried steered by process knowledge.

- Techniques for data cleaning.
  - Smoothing noisy data.
  - Removing sensor with too much missing values.
  - Data imputation for sparse missing data.
  - Outlier removal.

There are several techniques for data imputation. The simplest is replacing the missing values for the mean of the sensor. More complex ones include linear interpolation if the missing values are not consecutive. Using the k-nearest neighbors algorithm to correlate working sensors to the missing one is a popular choice if the missing values are consecutive (BATISTA *et al.*, 2002). All techniques depend on the data missing sparsely or in a predictable manner. In chemical processes, many missing values happen in time blocks, because for example sensor failure, therefore inference of these missing values is nonviable.

Data treatment is an important steps in all methodologies reviewed in this thesis, and was an essential part in all case studies with real data. Separating data treatment in data quality evaluation and data cleaning, it became easier to revisit and find the rationale for each step and which information could have been lost in the data cleaning. In the data quality evaluation, the physical consistency was added to the list of aspects that should be checked up because some data in the real datasets appear to have numerical value, but the value is from a sensor fault.

In the case of synthetic datasets, acquired from simulated processes, the previous steps are not necessary, as the stochastic nature of noise is generally known and the data is already generated in the desired format. In a real world dataset these steps are essential before doing any modeling work. Depending on the size of the dataset and the available computational resources they may take a good share of the man-hours worked, so the schedule must accommodate these steps.

3.2 Data analysis

The objective of the unsupervised analysis is to find underlying relationships inside the data. It usually comes to two aspects: redundant and/or irrelevant features and different modes of operation. Some amount of redundant information is expected, as the upstream affects the downstream and vice-versa, therefore many variables in an operation are correlated with each other. Depending on the degree of collinearity between features they may cause issues in some

models (BABALOLA and OBUBU, 2019). Recycles and control loops also contribute to the correlation between features.

In modern plants many modes of operation can be found, as different products demand different operational conditions. These modes of production generate interesting properties on the dataset like multimodality that more traditional statistical methods do not deal well with (JOLLIFFE and CADIMA, 2016). e.g. PCA and correlation analysis. Creating different models for different modes of operations is a viable strategy. For detecting different modes of operation clustering methods can be used, like K-means, mean shift or Gaussian mixture models with special care to analyze the behaviour of the model over time.

- For detecting feature redundancy.
  - Correlation analysis.
  - PCA.
- For detecting operation modes.
  - K-means.
  - Mean shift.
  - Gaussian mixture models.

The purpose of this step is to develop knowledge for the data selection step, and to complement previously established process knowledge. The outcome of this step should not be seen without context. For example: Lack of correlation between 2 variables that should be correlated may indicate sensor fault or labeling error.

This step exists in the methodologies reviewed in this thesis, but with a focus in feature reduction or sample selection. Here it was given a interpretation more geared towards process engineering and developing a dataset understanding. The outcomes of this step were also used for feature selection in the next step, the results also lead to a better understanding of the processes studied.

The next step is data selection. More specifically: Which samples and features should be used for training, test and validation. The data cleaning step gives an initial amount of samples and features to discard. Sample selection is very problem dependent, e.g., depending of the scope of the project, operation mode transitions may be considered out of scope, or the most important part to be modeled.

- Methods for feature selection.

- Linear models with L1 regularization.
- Decision-tree based feature importance.
- Permutation based performance decrease.
- Successive training.

Feature selection must be balanced between increasing model reliability, and removal of information. Fewer features make the system more reliable as a sensor breaking will not jam it, but removing variables may also remove important information. For feature selection there are many techniques. The previously mentioned L1 regularization does internal feature selection. Feature importance given by decision trees can be used to decide which features to remove.

For pure black-box models, Permutation Feature Importance, also called mean decrease in accuracy (HAN *et al.*, 2016), can be used to evaluate feature importance. It consists in keeping most features constant and shuffling one in the test dataset, then evaluating the score compared with the original test dataset. The more important features will show a bigger performance reduction, some shuffled features may even improve performance, showing they can be discarded.

Successive training is a family of feature selection techniques. One of them is stepwise regression (JOHNSSON, 1992), in which several linear models are fitted in succession, removing or adding the variables each new fitting according to a determined metric, for example F-test. If the chosen metric is a validation score the method is called Sequential Feature Selection.

Plain operator knowledge can also be used. It has the advantage of increasing trustness in the model, and given sufficient regularization, it does not decrease performance significantly. Some operator recommendations may have been discarded in the unsupervised analysis due to collinearity or other phenomena. Some recommendations are given due to sensor reliability, as some sensors are so important to the process that there are extra measurements or constant maintenance checks and calibration.

Feature selection has demonstrated to be one of the central steps is developing a good model, therefore is separated in its own step on this methodology. One issue that happened in the case studies is that some sensors may be informative, but if they receive outliers or break, the whole model breaks with them. So fewer and reliable sensors should be selected.

3.3 Modeling

The initial models are built on the data after all processing, and tested in a separated environment; ideally with more recent data. Then feedback from engineers and operators regarding the quality and behavior of the model should be received and noted, whether it works as expected, or if not, what is the expected behavior. For example: if the model oscillates too much between samples, it may be better to remove relevant but noisy features. Another point is to analyze the metrics. Choose one as the main metric, but keep another for better understanding of the results. Example: Unbalanced datasets can have high accuracy but low F1-score, as ccan be seen in Appendix A.2, by having the model only predict the most common class. Inference in a stable process can have a low R^2 but be within the reproducibility of the lab test. A list of regression metrics can be found is Appendix A.1.

- Metrics for regression.
  - RMSE.
  - R^2.
  - ISO/ASTM reproducibility.
- Metrics for classification.
  - Accuracy.
  - F1-score.
  - Precision and recall.
  - ROC-AUC.

This issue of metrics arose during the case studies because the industry is a much more open environment than a laboratory or a simulation. In industry, uncertainties in measurements and unmeasurable disturbances are commonplace. So pursuing concepts like "95% of the explained variance" for R^2 are not realistic goals. Additionally, in the industrial case studies the operators valued metrics they could easily interpret, like RMSE that has the same unit as the measurement, or accuracy.

When possible, a model analysis is done, e.g., analyze the internal weights to see any important pattern, check if multiple feature importance procedures agree with each other. When the training data is sampled every hour, see how it behaves with data from every minute. Another possible analysis is uncertainty propagation to see how to reduce variance in an overly noisy output. If the

model properties matches with previously known process knowledge and with the unsupervised analysis, it assures that it will behave properly in production. Otherwise there may be an issue in any step of the process. How to deal with this mismatch is still being developed.

Operators do not trust models that do not follow expected behaviour (MUIR and MORAY, 1996). Model analysis helped to build trust with the operators by showing how the model would behave in unseen circumstance and how its output would look like in case of a fault or outlier.

Then a refinement cycle is done until desired performance is achieved. Usually refinement consists in acquiring more data, developing new features and hyperparameter optimization. In practice, industrial data is more complex than simulated data, so the desired performance may not be achievable, so a "good enough" line must be drawn.

The refinement part came from the cooperative and business side of projects with the industrial partners. They wanted regular updates to evaluate how the project is progressing. It was also desired by the rest of the development team to have a model so they could develop the rest of the framework in which the model would operate. "Good enough" line is drawn because diminishing returns were noticed during the projects, and it was eventually necessary to move on to the next phase of the projects.

3.4 On the implementation of the proposed methodology

To validate the methodology, four case studies will be used. A simulated gas lift oil well control problem, a melt flow index (MFI) estimation problem, an offshore oil well fault detection problem, and a dry gas seal fault detection problem. The gas lift oil well problem was chosen because it is a control problem with an interesting observability issue, with no measurements on the bottom of the oil well. The 3W dataset was chosen because it is a large public dataset, allowing for a good analysis of real industrial data without having to anonymize the data and the results. The MFI estimation problem was chosen because it was an industrial regression problem with well structured datasets. The dry gas seal problem was chosen because it was an industrial classification problem with common issues like fault overlap.

For the simulation gas lift oil well case study, Matlab and its neural network toolbox were used. It was chosen because Matlab is well suited for dynamical simulation and control studies, with state-of-the-art numerical integrators and

constrained optimizers. For the industrial case studies and the offshore oil well problem, Python was chosen because it is a very popular open-source language (ROSSUM, 1995). Python also has good machine learning and data processing libraries like Scikit-learn, also known as SKlearn (PEDREGOSA *et al.*, 2011) and Pandas (MCKINNEY, 2010). An advantage of using a single library for modeling is faster prototyping. SKlearn was developed in such a way that it is possible to develop and implement several different models with minimal changes in the supporting code.

Regarding the metrics, it was expected to use well established metrics like visual inspection for control, R^2 for regression and accuracy for fault detection. During the development of this work, metric formulation became an important part of the workflow. Custom metrics, F1-score, pre-fault accuracy and other metrics were evaluated to capture interesting characteristics of the models.

Physically, the models in the industrial case studies were implemented on the data management software of the plants. Some plant data management programs like Honeywell's Aspen (AL-MALAH, 2016) already have compatibility with Python. The models are expected to return a floating number with a timestamp that can be plotted over time. The simulation case study was not implemented on a real plant, but may be used as the starting point in the implementation of a similar control scheme.

To assure actual process improvement the follow up with operators is essential. They can tell how day to day operations changed with these processes and if the model should or should not follow the changes. MFI estimation case study can be directly evaluated as polymers with MFI out of specification are discarded resulting in economic loss. Dry gas seal fault detection is trickier, since faults can take months to happen. Therefore, weekly meetings with the operators helped evaluating the model, to see if the process spent some time in a given week at fault or not.

A current limitation of the proposed methodology is what to do if process knowledge, unsupervised analysis and model analysis do not agree with each other. There is no unified approach, the researcher must use their own insights to solve them, as some examples are given on the industrial case studies. Another limitation is how to deal with process drift and model update. Process drift happens in every process, and take years to be seen. A model update methodology is a theme worth studying. A constant retraining of the models could be done, or look for a semi-deterministic approach, that can be updated through traditional engineering modeling, to guide the model are possible approaches that should be studied in future works.

Chapter 4

Gas lift oil well control

4.1 Problem description

Gas lift oil wells (GLOW) are oil wells that use high pressure gas to improve oil production. In these wells, natural gas is used to reduce the density of the mixture in the tubing, lowering the pressure in the bottom hole, and increasing the production rate of the reservoir, as can be seen in Figure 4.1.

The purpose of this case study is to develop a model predictive control (MPC) for a gas lift oil well. The challenges of this problem are:

- unknown internal states.
- noisy measurements.
- unmeasurable disturbances.
- slugging.
- model mismatch.

Slugging occurs when the gas injection flow rate is too low. In this case the annulus loses more gas than enters, decreasing pressure, until the pressure in the bottom of the annulus is lower than the bottom hole pressure. Natural gas stops flowing into the tubing and accumulates in the annulus, increasing the pressure. The output flow also reduces and both pressure and density of the mixture in tubing increase. When the pressure in the annulus gets higher than the bottom hole pressure, gas resumes flowing into the tubing, generating a huge peak in the output flow and the density of the tubing rapidly decreases. Eventually the annulus starts losing more gas than is injected; both the pressure in the annulus and in the tubing decreases to the point of the bottom hole pressure becomes higher than the pressure at the annulus bottom, so the process restart (EIKREM *et al.*, 2004).

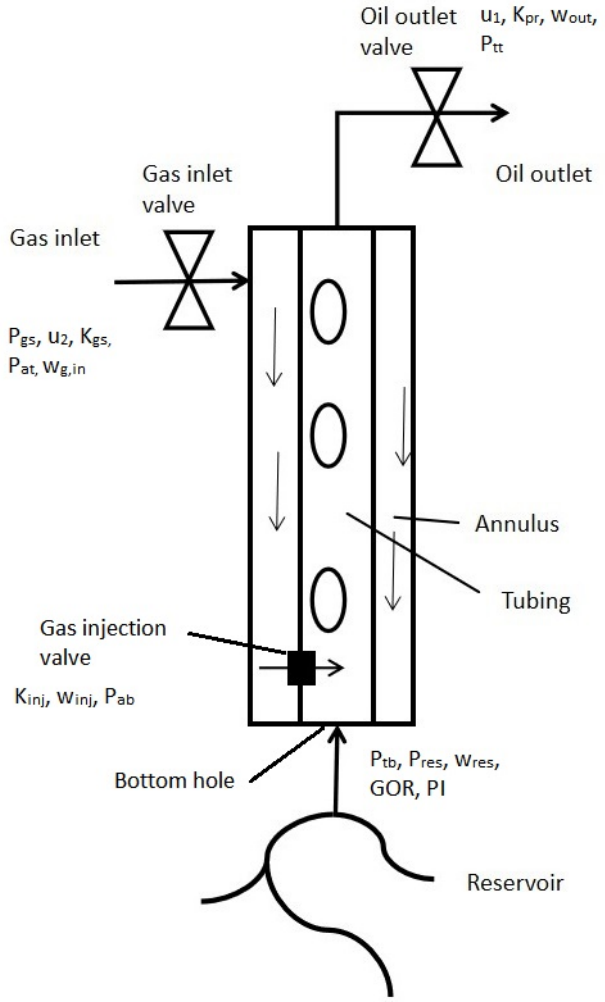


Figure 4.1: Gas Lift Oil Well.

The gas lift oil well in this case is supposed to be a mature well. It is also considered that sensors installed in the bottom of the well have broken or discalibrated over the years, so those measurements do not exist or are unreliable. The internal states are the mass of gas in the annulus, the mass of gas in the tubing and the mass of oil in the tubing. It is not possible to directly measure any of those states (JAHANSHAHI *et al.*, 2012).

Sources of disturbances are the natural gas pressure in the gas source, reservoir pressure (P_{res}), productivity index (PI) and gas oil rate (GOR). The natural gas source pressure depends on upstream processes, while the other disturbances are due to the inconstant nature of mature oil wells. As these disturbances occur in the reservoir no measurement is available.

MPC utilizes a model of the process to find the optimum set of control actions to make the process behave as desired. In this work a simplified first principles internal model was used. MPC model assumes ideal gas, no pres-

sure drop from friction, homogeneous oil gas mixture inside the oil well and no disturbances, while the process model uses Peng-Robinson Equation of State (PENG and ROBINSON, 1976) (PR_{EoS}), assumes pressure drop from friction, linear profile of liquid fraction between the bottom and the top of the column and suffers from disturbances.

The process model comes from JAHANSHAHI *et al.* (2012). Which is an improvement of the model found in EIKREM *et al.* (2004). JAHANSHAHI *et al.* (2012) added pressure drop calculations in the tubing, which are important as the tube has more than 2 km, they also added a valve in the production choke, turning a SISO problem into a MIMO problem and assumed that a gas lift valve is responsible for the natural gas inflow, creating a variable upper limit for gas injection, while EIKREM *et al.* (2004) assumed natural gas inflow as the manipulated variable. The process model was improved here by changing the equation of state from ideal gas to Peng-Robinson, as the pressure varies from 20 bar to 90 bar, a pressure in which ideal gas behavior is not a reasonable approximation. In this region at the working temperature the compressibility factor ranges between 0.98 and 0.92 (PERRY *et al.*, 1997). An exponential filter was used to estimate the reservoir mass flow used in pressure drop calculations, Equation 4.4.

The other process model differential equations are based on mass balance; Equation 4.1 describes the gas mass balance in the annulus, Equation 4.2 describes the gas mass balance in the tubing and Equation 4.3 describes the oil mass balance in the tubing.

$$\dot{m}_1 = w_{G,in} - w_{G,inj} \tag{4.1}$$

$$\dot{m}_2 = w_{G,res} + w_{G,inj} - w_{G,out} \tag{4.2}$$

$$\dot{m}_3 = w_{L,res} - w_{L,out} \tag{4.3}$$

$$\dot{m}_4 = m_4 - w_{res} \tag{4.4}$$

w are mass flow rates. Subscripts G and L mean gas and liquid flow rates respectively, if neither are present both phases are combined. Subscripts *in*, *inj*, *res* and *out* mean the fluid is injected into the system, is injected from the annulus to the tubing, comes from the reservoir or is ejected from the tubing, respectively. The equations that govern the dynamics of the oil well and the model constants are given in Appendix B.2.

4.2 Literature review

4.2.1 Slugging control

RIBEIRO *et al.* (2016) connected different wells in a network and to a three phase separator, while adding different pressure drop calculations to EIKREM *et al.* (2004) model. They controlled the process using a MPC based on a set of linear models. They also assumed that gas injection was a manipulated variable instead of the gas injection valve opening. They also tried to keep quality specification inside the desired setpoints.

DIEHL *et al.* (2018) controlled a model containing a GLOW connected to a flowline and a raiser using an NMPC. The process was modeled in OLGA while the NMPC used a more simplified first-principles model. The states were estimated with a Hybrid Extended Kalman Filter and managed to suppress slugging. In a follow-up study DIEHL *et al.* (2019) managed to increase the oil production of a real-life GLOW by 10% using the predictive controller BR-NMPC, which was used in the linear option.

PEIXOTO *et al.* (2015) used the same process and controlled it using Extremum Seeking Control. However their work was focused on the gain inversion caused by high gas injection flow rate and optimizing oil production. They managed to find an optimum of oil production using a precompensation based on a reduced-order dynamic model with similar behavior of the model of RIBEIRO *et al.* (2016).

Regarding data-based modeling, JORDANOU *et al.* (2018) used a new kind of recurrent neural network, called Echo State Network, to build an NMPC of GLOW. They used a linearized forced response of the Echo State Network at the current process region as the internal model of the NMPC, and managed to control the gas lift oil well despite significant modeling errors. In a follow up work JORDANOU *et al.* (2019) used a novel framework to add online learning to the ESN and tested it to control a more complex system including 2 gas lift oil well, a pipeline-riser and a manifold.

DIAS *et al.* (2019) also tested an Echo State Network for this process, but focused on prediction, trying to evaluate if the model can predict the topside measurements over longer time periods and at different process regions. They evaluated the stability of the Echo State Network in making longer time prediction, indicating the model would be useful in applications that demand a longer prediction horizon like RTO.

GEREVINI *et al.* (2018) used an NMPC formulated using the local linearization of the process model and an Extended Kalman Filter as state estimator to

control both a GLOW and a raiser. They showed that disturbances can permanently destabilize the process and the Extended Kalman filter control helps compensate for it.

KRISHNAMOORTHY *et al.* (2016) used CasADI to model the process and to optimize two GLOW connected through a manifold under uncertainty. While not dealing with control itself it demonstrates the importance of the unmeasured disturbances down the oil well, more specifically the deviation of Gas-Oil ratio.

The purpose of the new modeling is to create a pair of mismatched models with different degrees of complexity and different estimated parameters, as real life processes tend to be more complex than the models derived from them and suffer from dynamically inaccurate parameter estimation and process drift. Many of the previous works cited here used OLGA simulator for that, but it is a proprietary software that researchers may not be able to access. Other works either used linearized models in the MPC or a Machine learning based model.

4.2.2 State estimation

The parameters being estimated are K_{inj}, PI and P_{res} and GOR. Geometric parameters are known. To control the process, the internal NMPC model is integrated; for this integration an estimate of the internal states is needed. To estimate them a neural network was trained on the available sensor data and the estimated internal states obtained during parameter estimation.

In the first iteration, an optimization problem was solved to estimate the model parameters. More specifically, minimization of the mean squared error between the measurements and the integrated NMPC model. Parameter estimation is given by Equation 4.5. As the sensors have different scales they were normalized using the variance. The constraint $y^{est} = g(par)$ means that the estimated data comes from the integration of the NMPC model which is a function of the parameters.

$$\begin{aligned} \min_{par} \quad & \frac{1}{S \cdot T} \cdot \sum_{i=1}^{T} \sum_{j=1}^{S} \frac{(y_{i,j}^{est} - y_{i,j})^2}{var(y_j^{meas})} \\ \text{s. t.} \quad & y^{est} = g(par) \end{aligned} \tag{4.5}$$

This optimization problem is considerably time expensive to solve. In the second iteration a parameter estimation based on the Extended Kalman Filter (EKF) was used. The EKF is better explained on the next subsection.

4.2.3 Extended Kalman Filter

Extended Kalman filter is an algorithm that uses measurements to estimate the state of a nonlinear system (WELCH and BISHOP, 1995). The Extended Kalman Filter has several formulations, the one used here is called Hybrid Extended Kalman Filter as it uses a continuous model and discrete measurements. The formulation of the EKF is given in Appendix B.4.

Extended Kalman filters can also be used for parameter estimation (SIMON, 2006). First an augmented state vector is created by concatenating a parameter vector and rewriting Equation B.36 as Equation 4.6.

$$\begin{bmatrix} \dot{x} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} f(x,u,t) + q \\ q_p \end{bmatrix} \tag{4.6}$$

where q_p is a small artificial noise to allow the parameter vector p to move. In this formulation the parameters are not exactly assumed constant but are treated as a variable taking a random walk. It should be noted that this parameter estimation is dynamic and the parameters may not converge to a stable solution if the process is unstable. It also gives additional degrees of freedom to the EKF, which allows it to reduce the measurements and modeling errors by adjusting the parameters even if they are supposed to be stationary.

The disadvantages of EKF is that the model may diverge if the process is incorrectly modeled and that tuning is difficult. There are several strategies for tuning an EKF (SALAU *et al.*, 2009) but there are no universal standards, with some authors even suggesting trial and error (SIMON, 2006). It may also diverge if the process is too nonlinear between the updates (JULIER and UHLMANN, 2004). For the GLOW model specifically, near and during slugging the discontinuous functions make the Jacobians matrix ill-conditioned, leading to numerical issues during estimation.

The motivation for replacing the EKF for a NN is that the NN encodes previous information while the EKF uses mostly the local information about the process. So regions were the EKF does not work well can be interpolated by the NN.

4.2.4 Nonlinear Model Predictive Control

The NMPC formulation is given by Equation 4.7 (RAWLINGS *et al.*, 2017). It minimizes the error given by the term $(y_k^{sp} - y_k)$, and contains a penalty for control action, Δu, to reduce sudden movement of the manipulated variables. The manipulated variables are restricted from 0 to 1 and the increments of the

control actions are bounded. In this problem y_k and y_k^{sp} are a 2-element column vector, and so is u_k.

$$\begin{aligned} \min_{u} \quad & \sum_{i=1}^{H_p} (y_i^{sp} - y_i)^T \times \gamma \times (y_i^{sp} - y_i) + \Delta u_i^T \times \phi \times \Delta u_i + \Delta y_i^T \times \lambda \times \Delta y_i \\ \text{s. t.} \quad & y = g(x, u, d) \\ & 0 < u < 1 \\ & \Delta u_i^{min} < u_i < u_i^{max} \\ & \Delta u_i = 0, i > H_c \end{aligned} \tag{4.7}$$

The outputs estimated by the NMPC model are given by $y = g(x, u, d) + e$, where x is the internal state of the model, here a 3-element column vector. e is a zero order corrector to reduce the process-NMPC model mismatch given by the difference between the current measurement y_0^{meas} and the measurements estimated by the model, y_0. d is the NMPC model measured disturbance, P_{gs}, which is a scalar.

γ and ϕ are weights to balance the setpoint error and the manipulated variable movement terms, respectively, of the objective function. H_p is the prediction horizon and H_c is the control horizon.

In this work the manipulated variables are the oil outlet valve and gas inlet valves, and the controlled variables are the gas inlet flow rate and oil outlet flow rate. They are all coupled, as the flows depend on the pressure balance which in turn is dependent on the internal states.

The proposed methodology for the creation of the state estimator is summarized in Figure 4.2. In this work a factorial design of experiments was used to run a series of step changes, but a train of random excitations or any other methods can be used to generate the data. In real life this process can be replaced with historical data. Once the data is generated, parameter estimation is run to fit the model to the data.

The internal states generated with the optimal estimated parameters for all input data are stored for further usage in the neural network training. A combination of the data and knowledge about the simplified model is used to evaluate which sensors would be relevant, since multivariate data-based models work better when irrelevant or redundant inputs are removed. In this study, correlation analysis was used for data-based selection, but any feature selection method can be used. For exemplification, feature selection by L1 regularization and random forest feature importance was also performed. Once the estimated internal states and the selected sensor data are available, a neural network is fit to the

data.

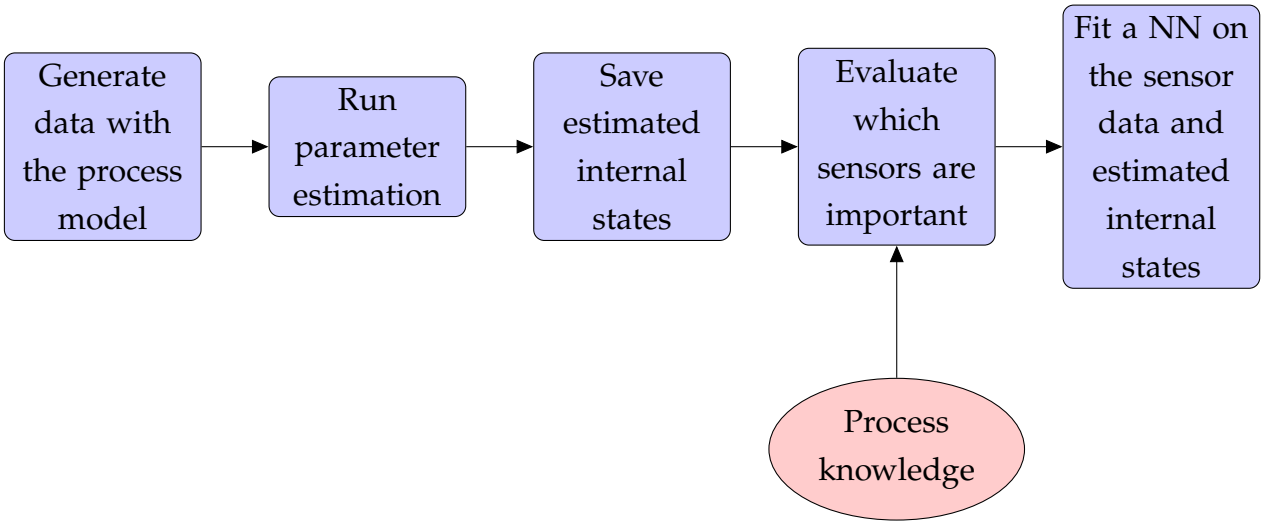


Figure 4.2: Flowchart of the proposed methodology for creating the state estimator.

Part of the methodology is data selection. The NMPC model parameter estimation and NN fitting would have worked much smoother if the slugging region were ignored. However one of the NMPC's task is to minimize slugging. NN cannot be expected to extrapolate well, therefore this data must be included. NN was chosen mainly due to evaluation speed and continuous output, as NMPC has a time constraint for computing the control actions in real time and should have a smooth response.

4.3 Materials and methods

The NMPC and well models were implemented in MatLab 2018a using the CasADI V. 3.5.5 framework (ANDERSSON *et al.*, 2019). CasADI is an automatic differentiation framework that includes numerical integration support and C-file generation. CasADI allowed for easy generation of the relevant Jacobians for the process integration, Extended Kalman Filter, including Equations B.38 and B.39 and objective functions and for the generation of mex-files that significantly increased the speed of the simulation.

The process model was integrated with Matlab's solver ode15s, as the process model is stiff in the slugging region, and the NMPC model was integrated with CasADI's RK4 integrator, as the RK4 integrator allows for automatic differentiation. The optimization problem was solved with Matlab's fmincon solver. The parameter estimation was done in the first iteration by optimization and in

the second iteration by the method described on the EKF section. Feedforward Neural Networks were implement using Matlab's Neural Network toolbox.

The data used in the model training and parameter estimation was generated by applying several step changes on the manipulated variables and allowing the process to run for 4 hours before the next step. The step changes were selected by 4-level factorial design as 0.88, 0.64, 0.40 and 0.17. The simulations ran for 2 hours when the process was on a stable region and for 6 hours during slugging. A total of 48 hours of the simulated process data was collected. Random Gaussian noise was applied to the data, with zero mean and standard deviation of 0.05 kg/m^3 for density measurements, 0.02 MPa for the pressure measurements, 0.001 kg/s for gas flow measurements, and 0.01 kg/s for oil flow measurements. This noise is also present in the control tests. The step changes are in Appendix B.2, Table B.2.

The data was sampled from the step changes experiments in 5 seconds intervals, later downsampled to 40 seconds. No analysis of sensor overall cleanliness was done because the data is simulated data. Data reformat was also unnecessary as the model already outputs the data in a tabular easy to read format with constant intervals.

For the parameter estimation the variables were scaled to improve convergence and numerical stability. The initial point was randomly selected but centered around $GOR = 4.4 \times 10^{-3}$, $P_{res} = 17$ MPa, $K_{inj} = 2.5$ cm^2 and $PI = 10$ kg/(MPa s). GOR initial point was given by subtracting all gas injected from all gas leaving the well, and dividing by the total oil production, all the other points were randomly selected. In the first iteration the optimization algorithm was Nelder-Meads simplex (NELDER and MEAD, 1965). For the second iteration the parameters final values were given by the median of the values calculated by the augmented EKF.

4.4 Results

4.4.1 Parameter estimation

The results of the parameter estimation are shown on Table 4.1. The results shown for the first iteration are the best of five trials. To know how much the noise and model mismatch affect the parameter estimation a new estimation starting from the known parameters was performed, and the results presented in the last row of Table 4.1. In a noiseless and perfectly matched system the optimization would not move from the starting point. However it can be seen that while the estimation from known parameters was better than the best found by

randomized starting points, it still deviated significantly from the model. They also deviated in the same direction, with PI and P_{res} lower than the known parameters but K_{inj} and GOR higher than known parameters. The parameter estimation given by the EKF had a better agreement with the known parameters compared with the optimization. The inverse relationship between P_{res} and PI is also seen in the EKF estimation. In general the EKF estimation was faster, however it was less stable, specially during slugging, in which the discontinuities of the model make matrices H_k and F ill-conditioned, therefore making $(H_k \widetilde{P}_k H_k^T + R)^{-1}$ numerically unstable.

Table 4.1: Parameter estimation results.

Parameter	GOR	P_{res}	PI	K_{inj}
Unit	10^{-3}	MPa	mg/(Pa· s)	10^{-4}
Known parameters	0	16	2.47	1.4
Parameter estimation starting from known parameters	1.3	15.36	2.59	1.34
Estimated parameters, first iteration	5.82	14.45	3.29	1.2
Estimated parameters, second iteration	2.812	17.019	2.143	1.390

The stability of the operational region was investigated in terms of valves opening, in order to the determine where slugging occurs, as can bee seen in Figure 4.3. It happens mainly when the gas lift valve opening is below 0.31 in both models. This agrees with the results presented by JAHANSHAHI *et al.* (2012), however in their paper the stability region was given in terms of gas inlet flow rate and oil output flow rate. There is some divergence when the higher the oil outlet valve opening is but it is at most a difference of 0.02 of gas inlet valve opening. In terms of the flow rates, the difference is more pronounced, with the NMPC model predicting slugging at a higher oil outlet flow rate for the same gas inlet flow rate.

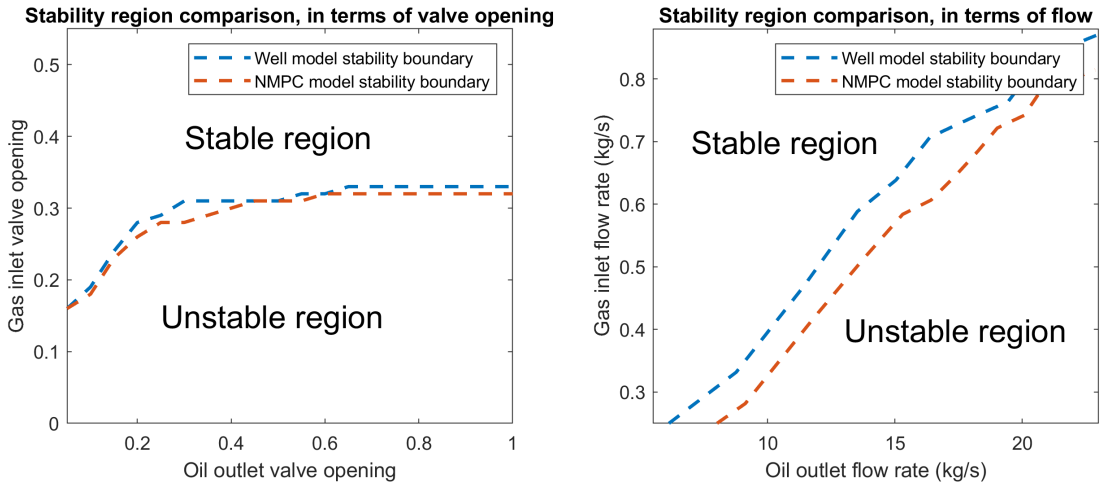


Figure 4.3: GLOW stability region.

One interesting analysis made possible by the EKF estimation is to see how the parameter estimation change over time. As can be seen in Figure 4.4, the estimation of P_{res} depends on the current steady state of GLOW, while K_{inj} is mostly consistent when there is no slugging. This is a good result, since while P_{res} can present some variation over time, K_{inj} should be constant. However as one can see in Figure 4.5 the estimation varies strongly during slugging. This is expected due to the instability of the system and is a good highlight on why care should be taken in modeling a parameter as a variable taking a random walk. The median value found in Figures 4.4 and 4.5 is the numerical value found in Table 4.1.

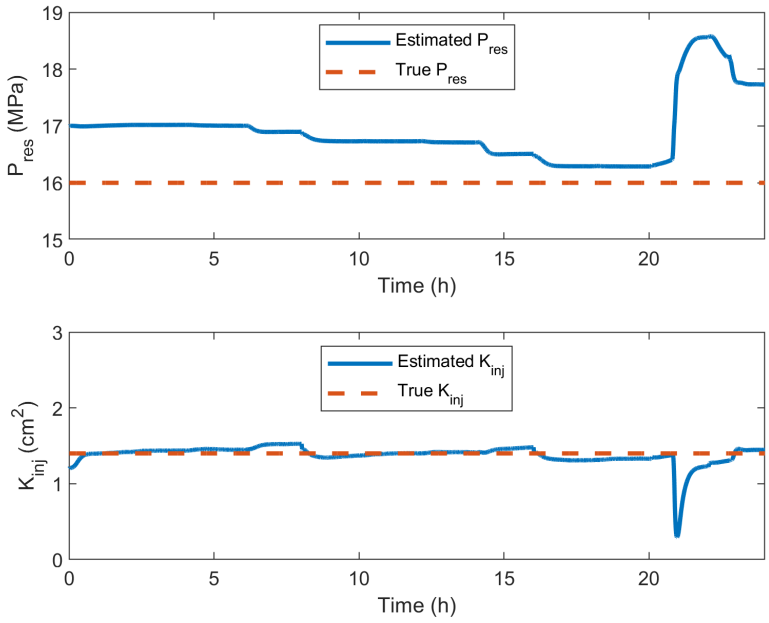


Figure 4.4: Parameter estimation results for the GLOW in stable condition.

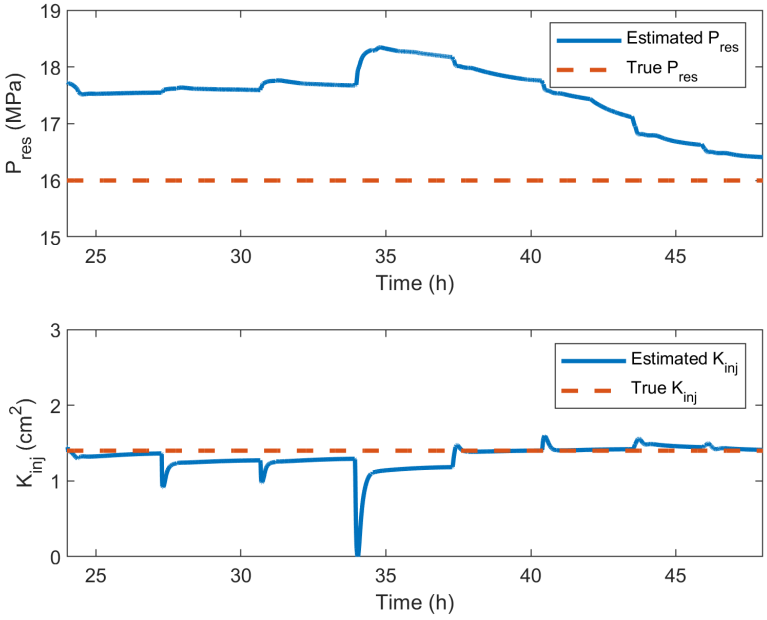


Figure 4.5: Parameter estimation results for the GLOW during slugging.

4.4.2 Modeling

In order to prepare the dataset for neural network training, process knowledge from the NMPC model supported by unsupervised analysis was used. The unsupervised analysis in this case was correlation coefficient matrix, seen in Figure 4.6. The idea is to check which variables are the most correlated with both m_2 and m_3, while not being correlated between each other. For example: $P_g s$ has no correlation with either internal state, so it is not used. w_G, in has strong correlation with the internal states, so it is used.

It was known from the NMPC model that the pressure at the top and mass flow ratio between oil and gas were relevant, however α_{gt} was not shown to be much relevant. Additionally gas source pressure only affects ρ_{ab}, so both were removed. Total output flow is very close to the oil output flow, so the former was removed. The selected features were:

- $w_{G,in}$
- P_{tt}
- u_2
- $\rho_{mix,t}$
- $w_{L,out}$
- P_{at}
- $\rho_{G,ab}$

How to do feature selection with L1 regularization and random forests is also shown here for illustration purposes. Although they are not used in this case study, they are used in other case studies. The general idea is that the results should agree, even though each method carries different assumptions.

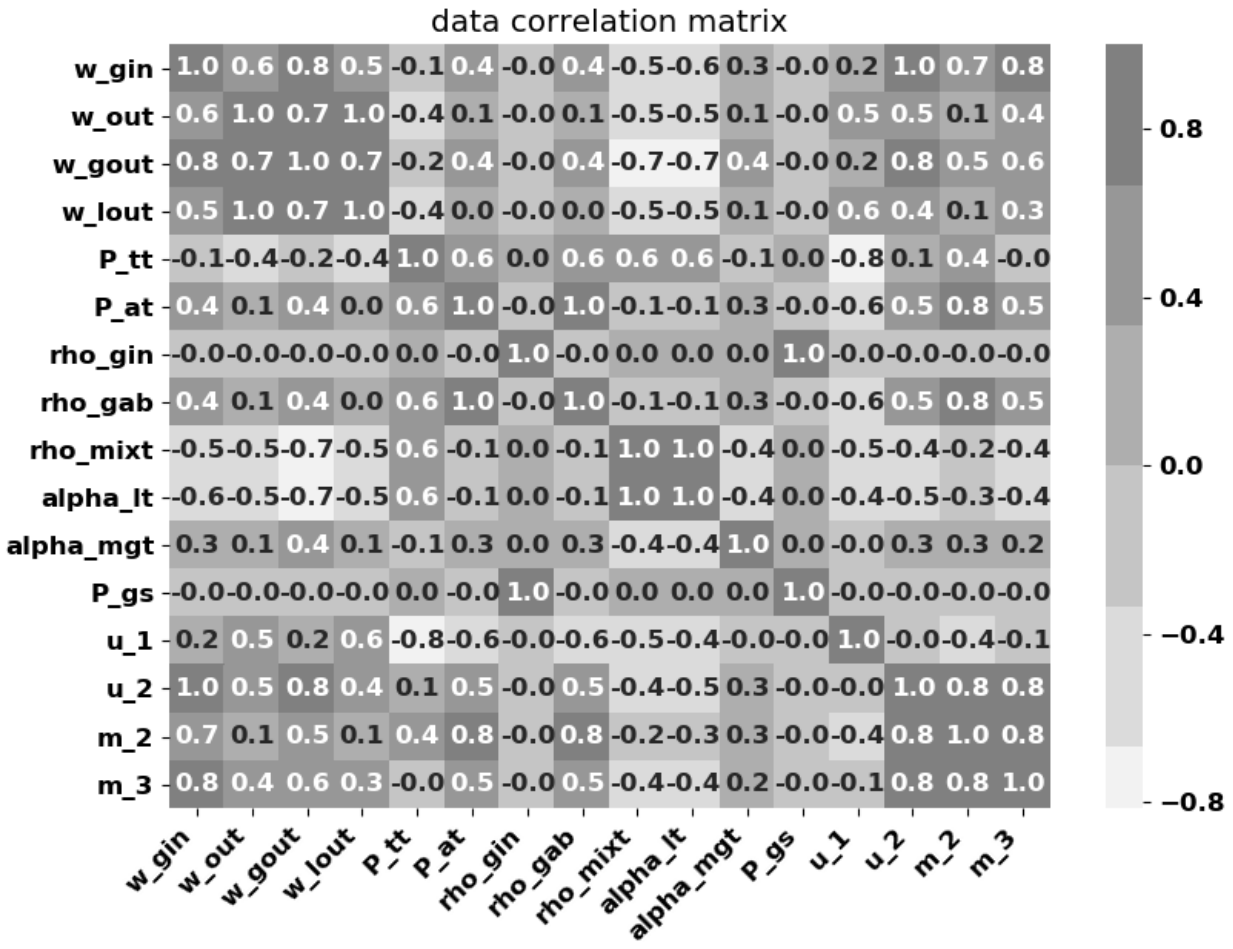


Figure 4.6: Correlation coefficient of the dataset.

In L1 regularization feature selection, a simple linear model with L1 regularization is fitted to the data using K-fold. The resulting weights at each folder are saved. It is plotted whether a weight is zero or not, as seen in Figure 4.7. Ideally each fold should agree with the others, so straight vertical lines are expected. This does not always happen for two reasons. 1) The model deals with colinearity by erasing one of the colinear features, which one depends on feature scale and initialization of the model. 2) A relevant variable in one fold may not be important in another due to distribution shift.

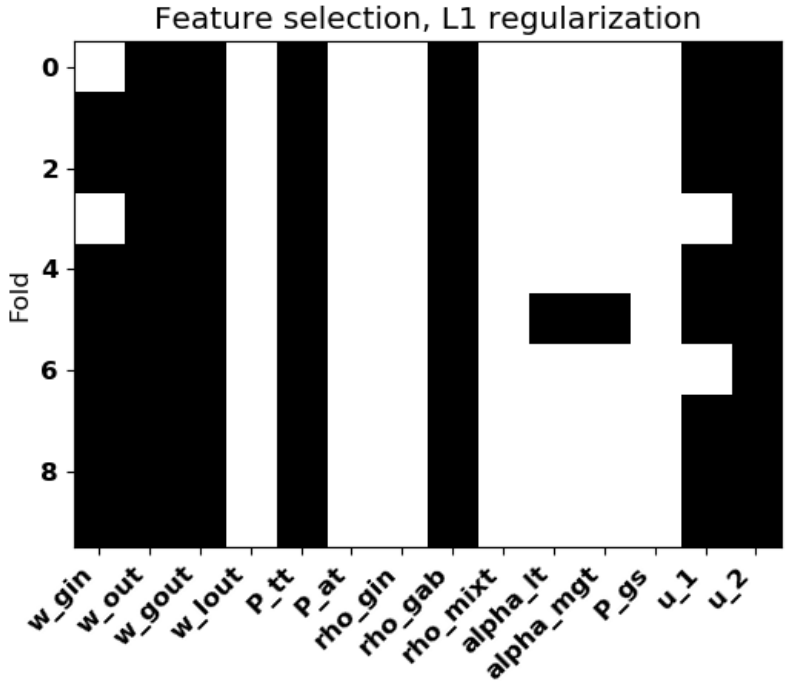


Figure 4.7: Plot for feature selection with L1 regularization.

In random forest feature selection, a random forest is fitted to the data using again K-fold. The resulting feature importances at each folder is saved. Then the mean of the feature importances is plotted, as seen in Figure 4.8. Plotting the error bar is optional but it is a good measure to see how accurate is the feature selection. The error bar in this case is given by the standard deviation of the feature importances. Ideally the error bar should be much smaller than the mean feature importance.

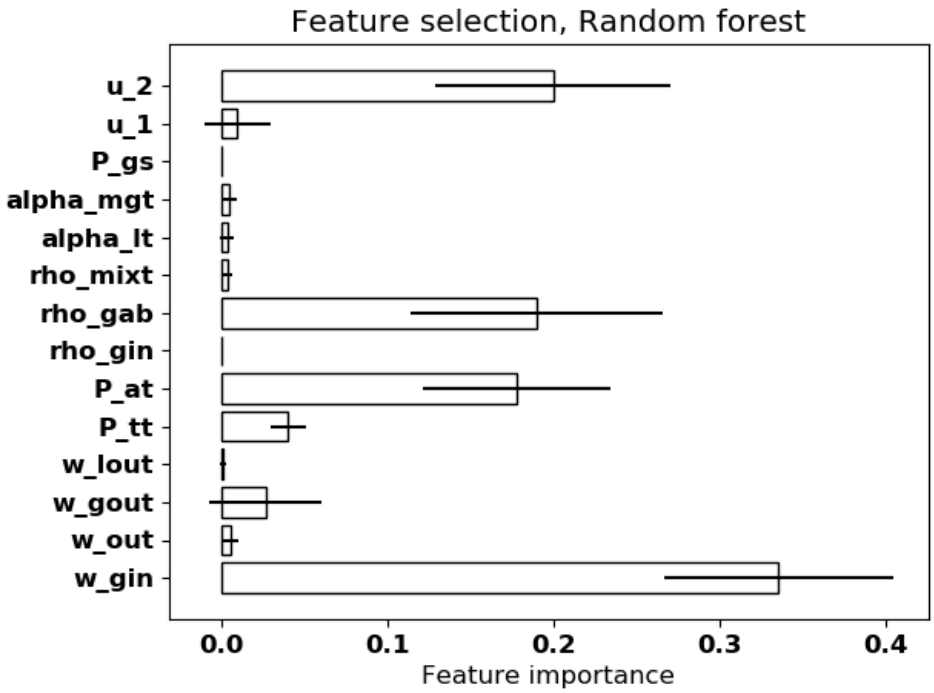


Figure 4.8: Feature selection using random forest.

Regarding the NN, training was successful, with a correlation coefficient of 89% in the first iteration and 93% in the second iteration, as can be seen in Figures 4.9 and 4.10. Most of the noise is due to slugging, where the internal states estimation becomes worst. Without slugging R^2 increases to 99.6%. Data based modeling is as good as the available data. Given the data during slugging is an inaccurate estimation, it is expected that the NN will not achieve a great inference capability. Table B.4 contains NN hyperparameters, while the NMPC setup is given in Table B.3, both found in Appendix B.3.

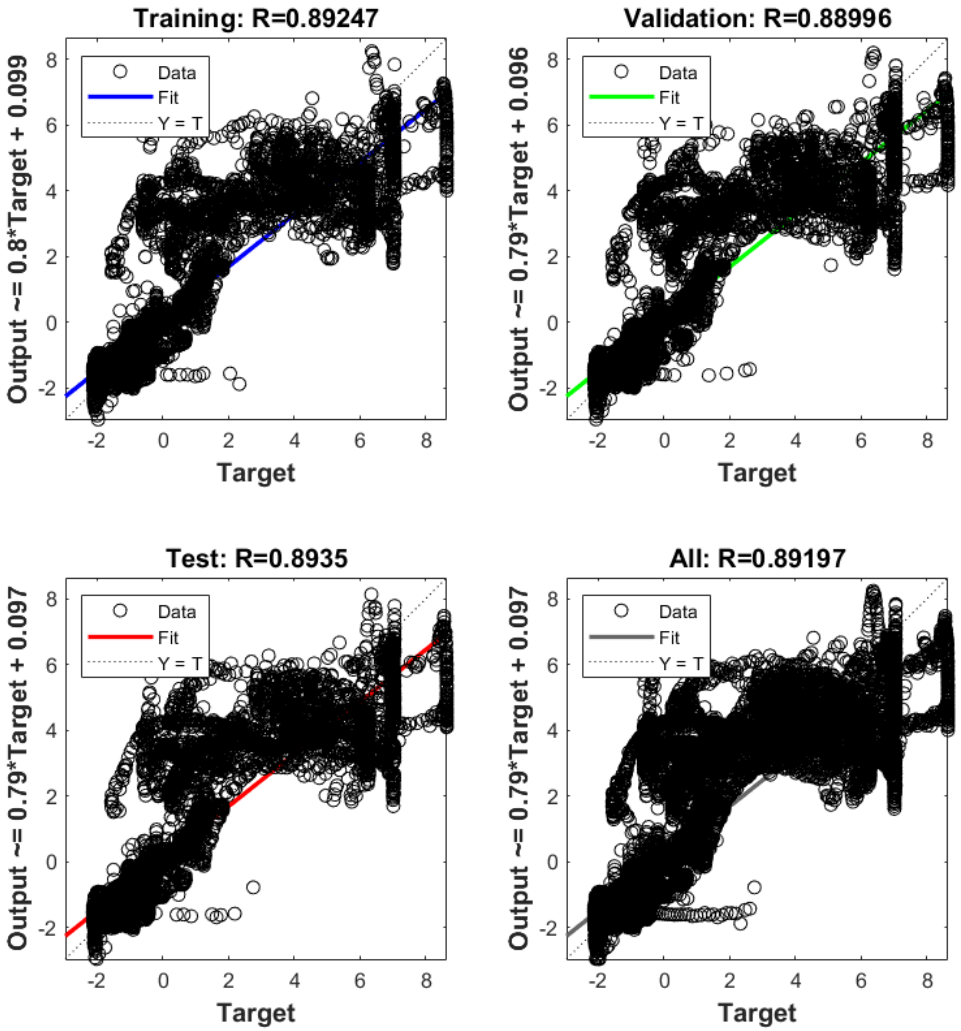


Figure 4.9: NN training results, first iteration.

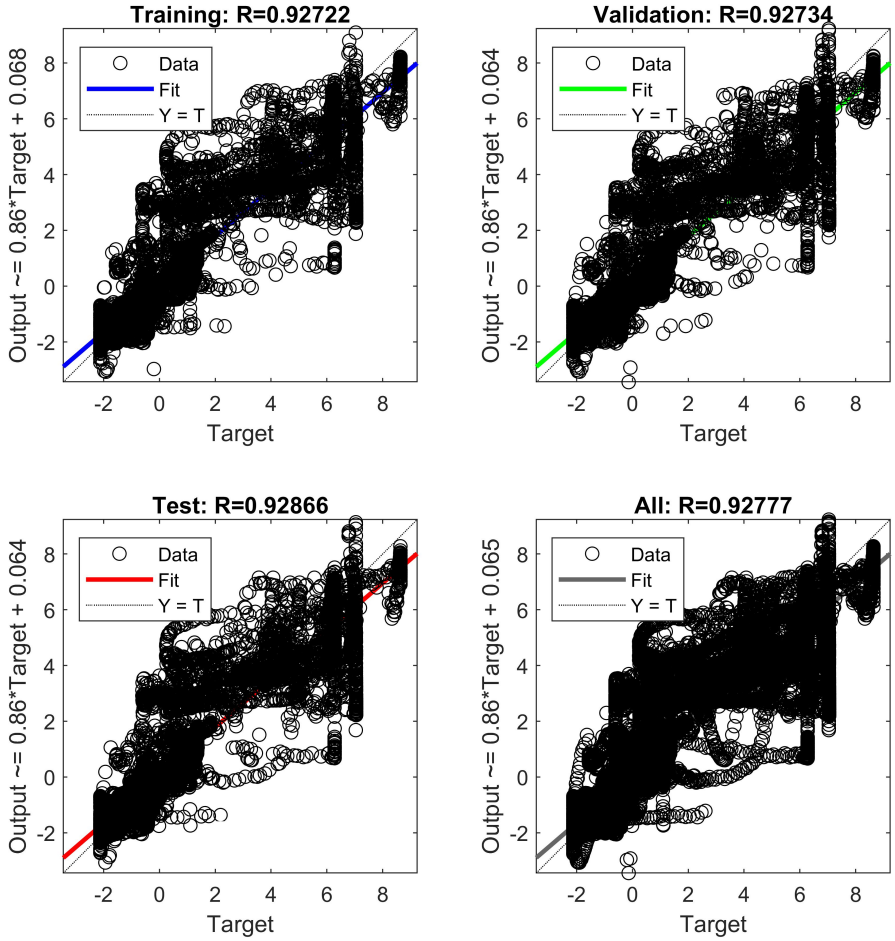


Figure 4.10: NN training results, second iteration.

4.4.3 Control test, first iteration

As can be seen in Figure 4.11, the servo test began at 2.5 hour to check if the system can inhibit slugging and achieve a stable initial state. The process was kept around the setpoints beforehand. The MPC achieves that with punctual injections of natural gas, both setpoints can not coexist. When the gas inlet flow setpoint increased the punctual injections decreased in intensity and the MPC reduced the opening of the oil production valve, as can be seen in Figure 4.12. When the oil setpoint increased, the NMPC could not follow most of the gas inlet setpoint and both the punctual injections and the gas injection mean were increased.

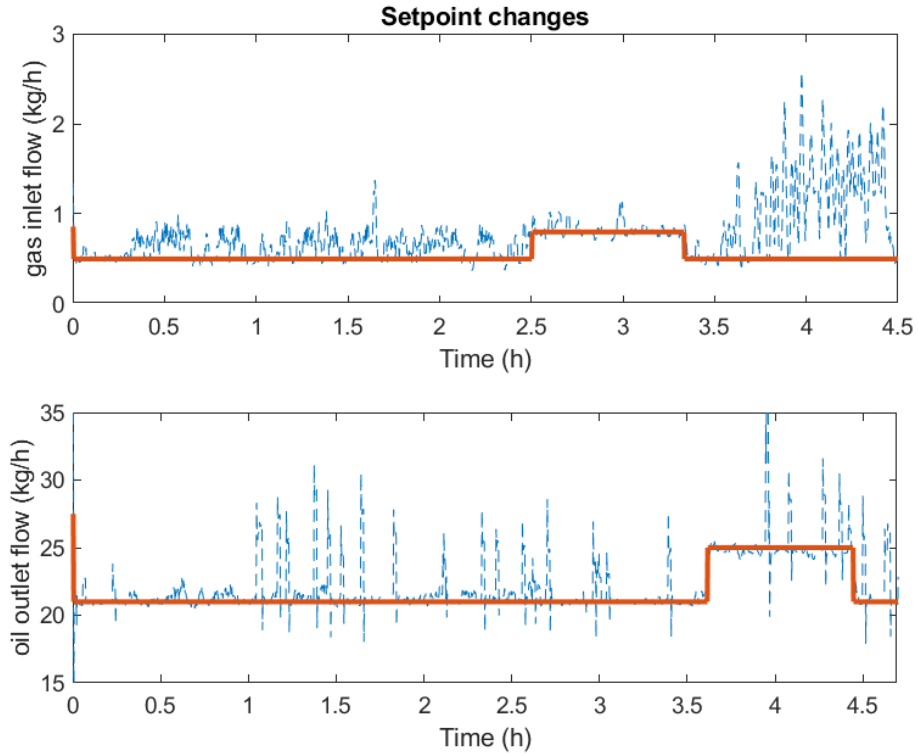


Figure 4.11: Process response to setpoint changes.

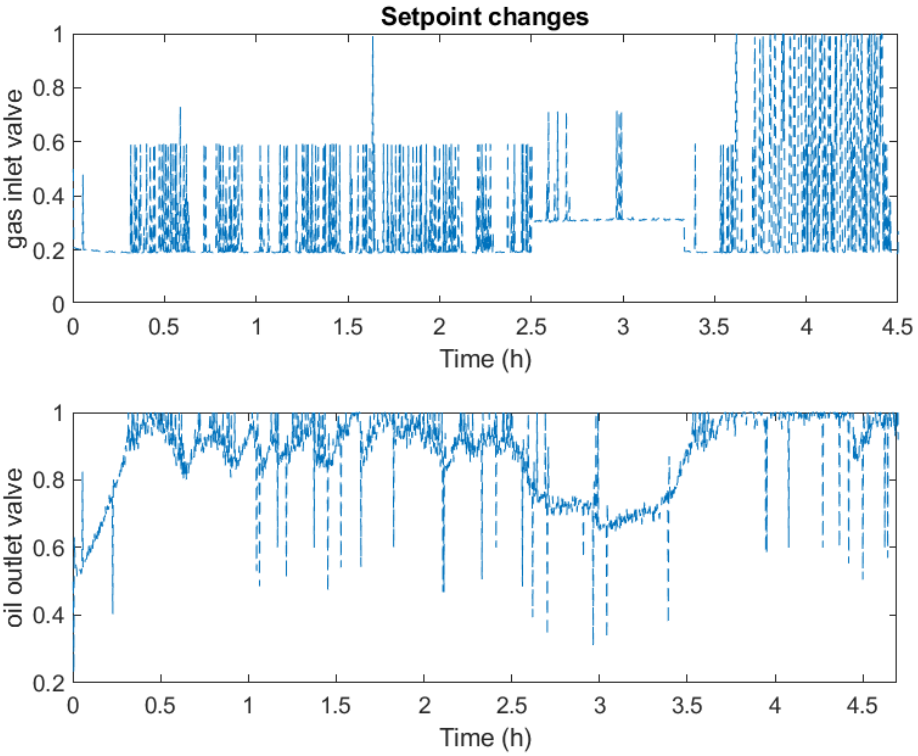


Figure 4.12: NMPC response to setpoint changes.

As can be seen in Figure 4.13, the process responded well to unmeasured disturbances. When P_{res} decreased from 16 to 15 MPa the NMPC increased the gas injections. when PI increased from 2.47 to 3.00 mg/(s·Pa) the gas injections decreased and the output flow valve opening was reduced to compensate for the increased productivity, as can be seen in Figure 4.14.

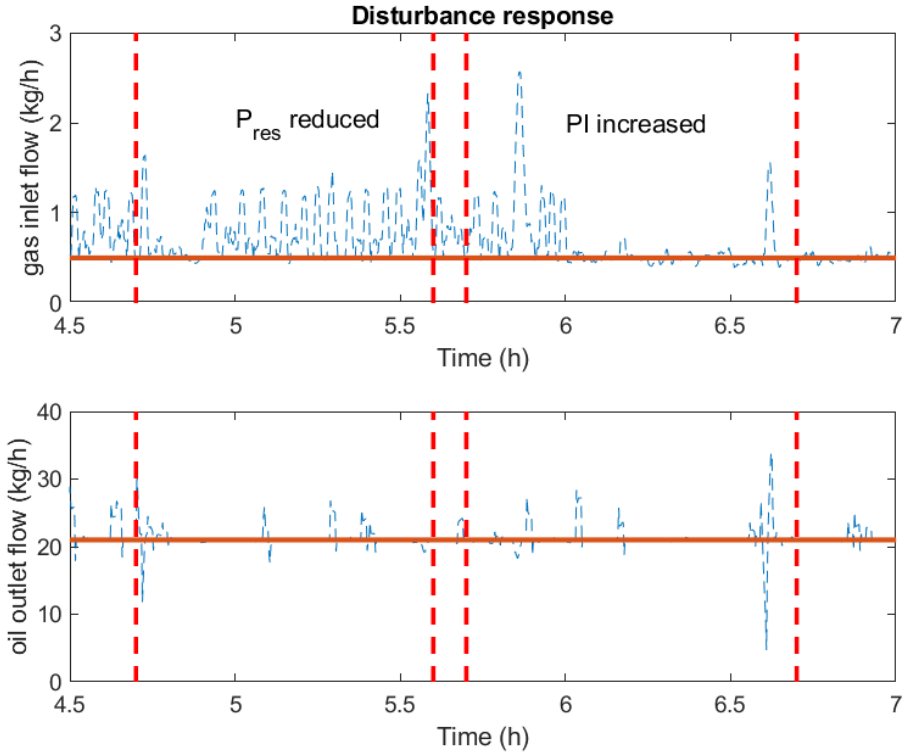


Figure 4.13: Process response to unmeasurable disturbances.

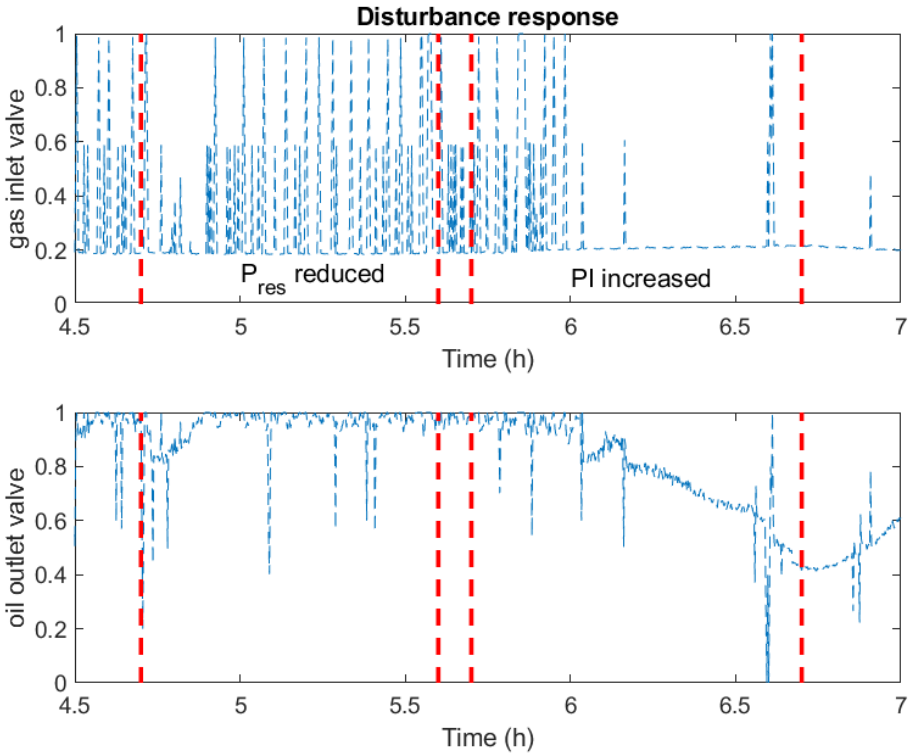


Figure 4.14: MPC response to unmeasurable disturbances.

As can be seen in Figure 4.15, the MPC was solved inside the sampling time, indicating that the model can be used for actual control. The objective function value varied in several orders of magnitude, so it was plotted in log scale.

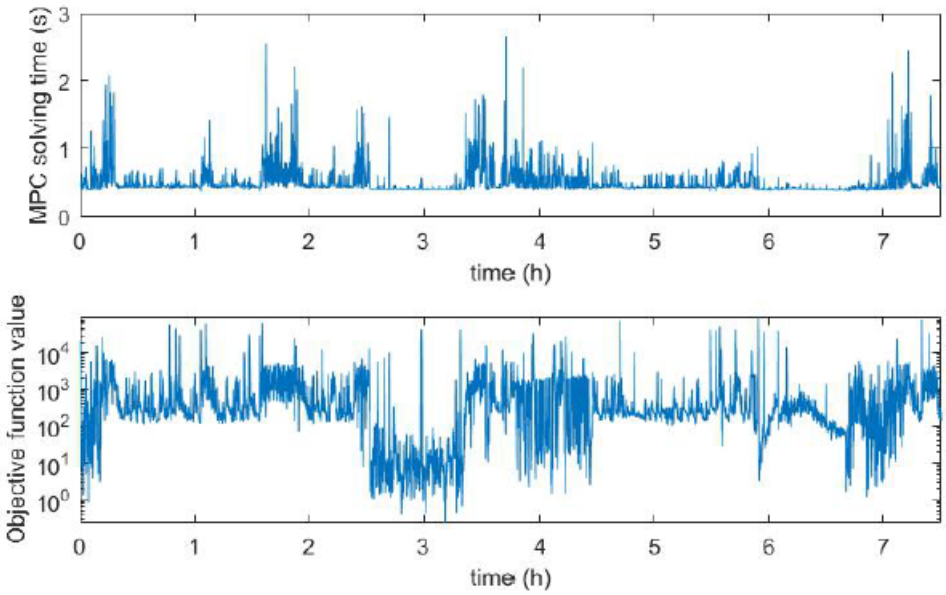


Figure 4.15: MPC solving time and objective function value.

4.4.4 Control test, second iteration

Extra control tests were added in the second control test: First a new NMPC, with perfect observability, was added to evaluate the NMPC response irrespective of the state estimator effectiveness, and a test using the EKF as the state estimator.

NMPC setup is given by Table B.3. The number of finite elements is a parameter of the CasADI's RK4 integrator. As it does not have an adaptive step size like RK45 the number of the steps must be informed to the integrator. The number of finite elements carries a trade-off between accuracy and computational cost, and the smallest number whose accuracy was on the same level as the process noise was selected.

As can be seen in Figure 4.16, the process was kept on the oil outlet flow rate setpoint in all cases of the NMPC using the perfect model. It achieved stability by doing small additional injections in the slugging rejection region. It also correctly predicted the increase of gas injection to follow the oil output flow rate setpoint increase. In Figure 4.18, it slightly increased gas injection to deal with the lower reservoir pressure, and momentously decreased oil production and increased gas injection to compensate for higher productivity index.

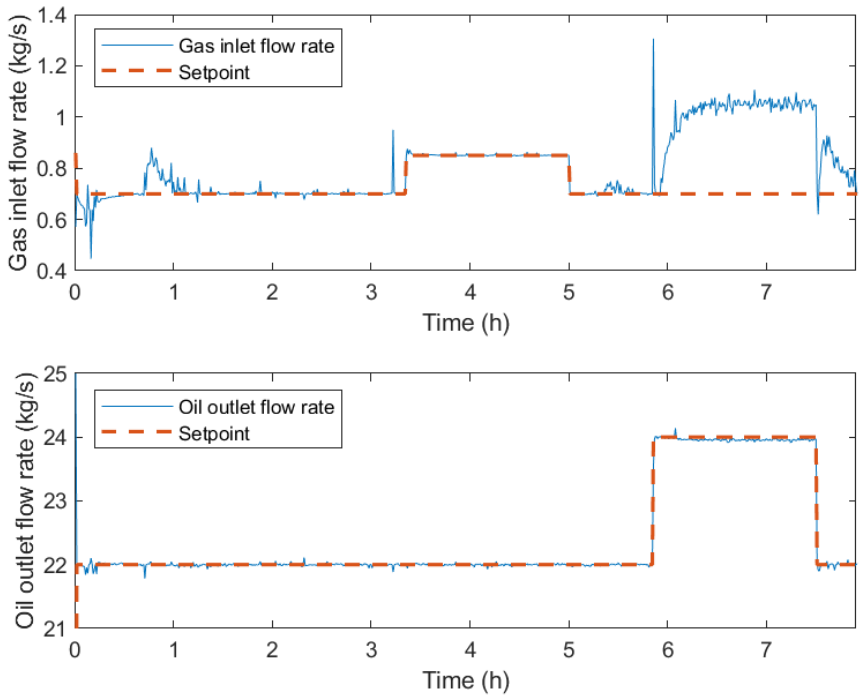


Figure 4.16: Process response to the setpoint changes using the controller a with perfect model.

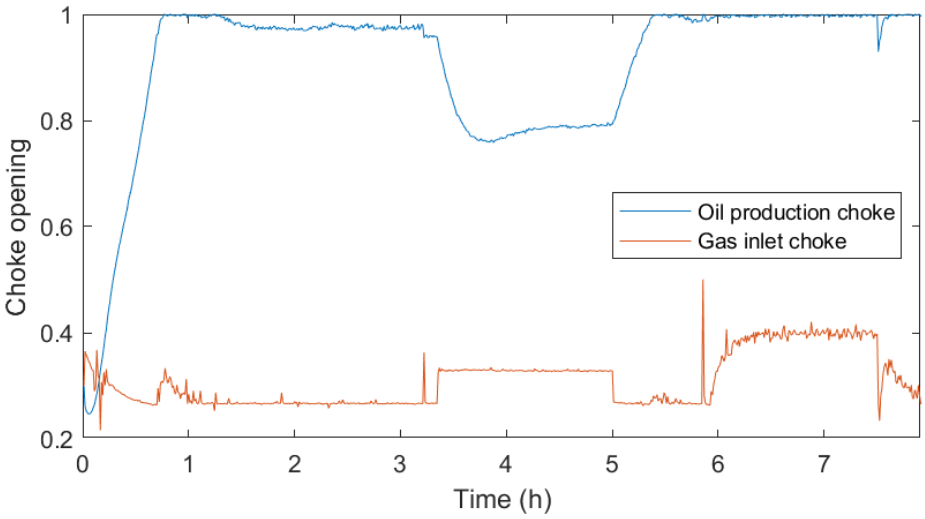


Figure 4.17: Setpoint changes using the controller a with perfect model.

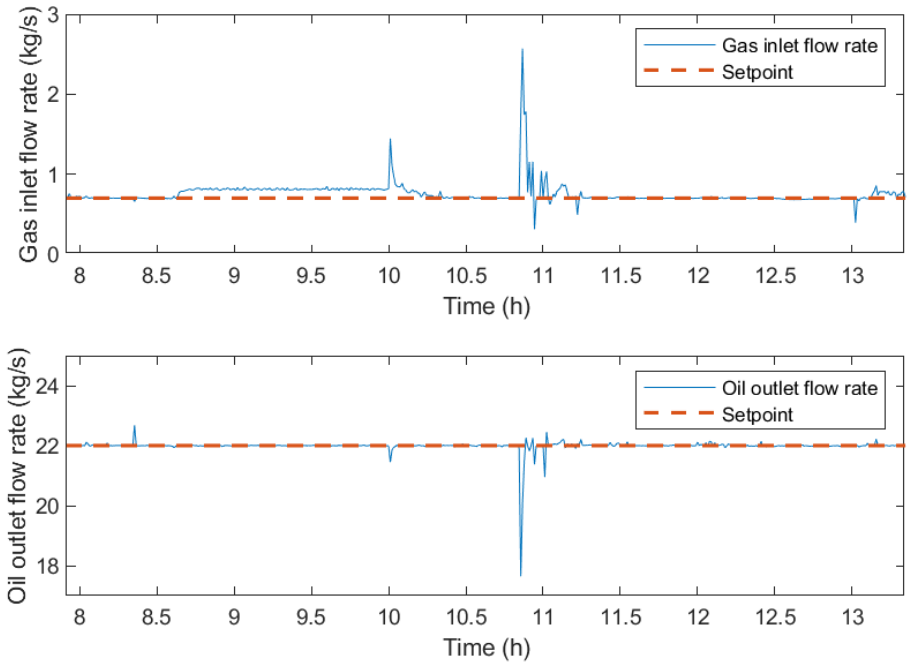


Figure 4.18: Process response to disturbances using the controller with a perfect model. P_{res} reduced between 8.3h and 10h; PI increased between 10.8h and 12.5h.

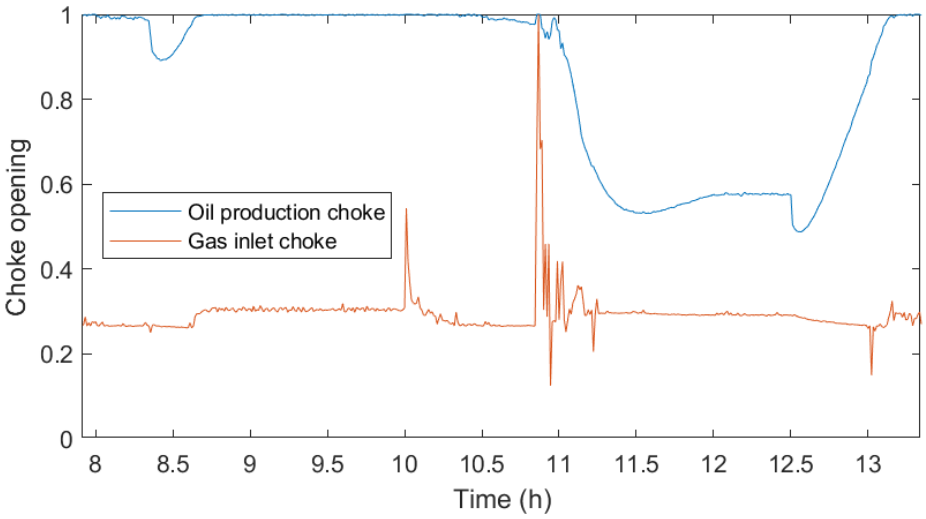


Figure 4.19: Control action to the disturbances using the controller with a perfect model. P_{res} reduced between 8.3h and 10h; PI increased between 10.8h and 12.5h.

The parameters for the NMPC using the state estimators and mismatched model are given in Table B.5. As can be seen in Figure 4.20, the EKF took 9 minutes to stabilize. Afterwards, it had a small persistent offset of around 0.53 kg/s in the oil production setpoint and 0.048 kg/s in the gas injection setpoint. However, it managed to reduce the oil production offsets during the setpoint changes. The controller using the NN as state estimator, did not present an offset in the gas injection. However, during slugging rejection it had the same offset as the EKF estimator, albeit less noisy. The solution found for the oil production setpoint change was much more noisy than the previous solutions.

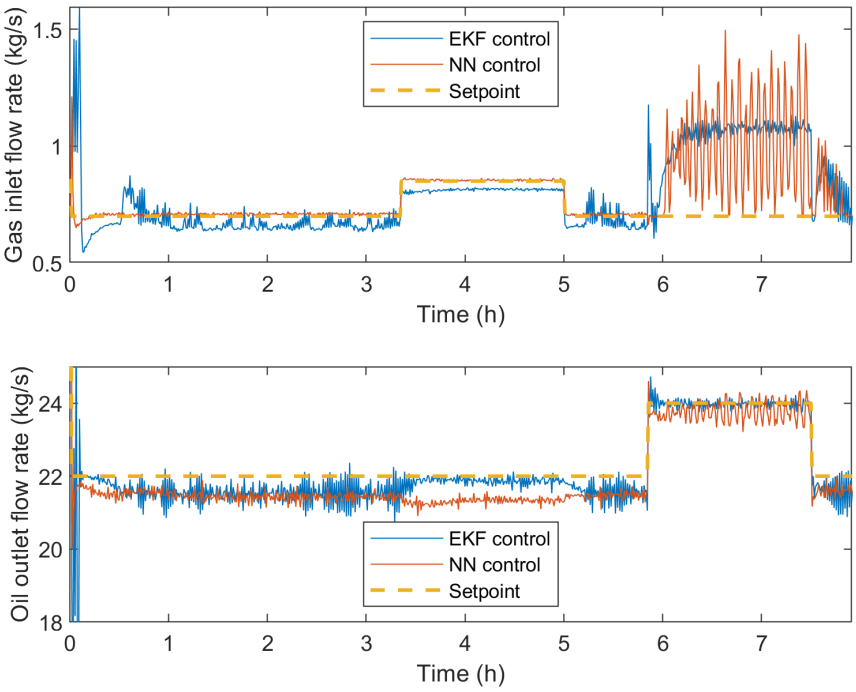


Figure 4.20: Process response to setpoint changes.

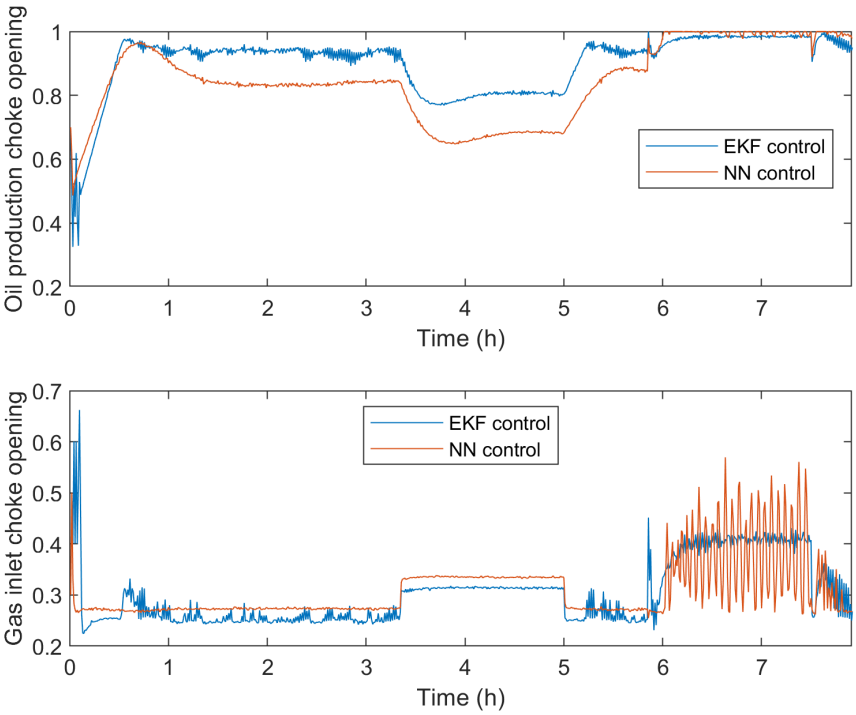


Figure 4.21: Manipulated variables response to setpoint changes.

As can be seen in Figure 4.22, the NMPC using both state estimators found a similar solution for the reservoir pressure reduction. However, the NMPC using EKF as a state estimator could not deal with the increased PI disturbance and suffered from slugging. The NMPC using NN as a state estimator did manage to avoid slugging during PI increase, albeit the oil production was not kept around the setpoint.

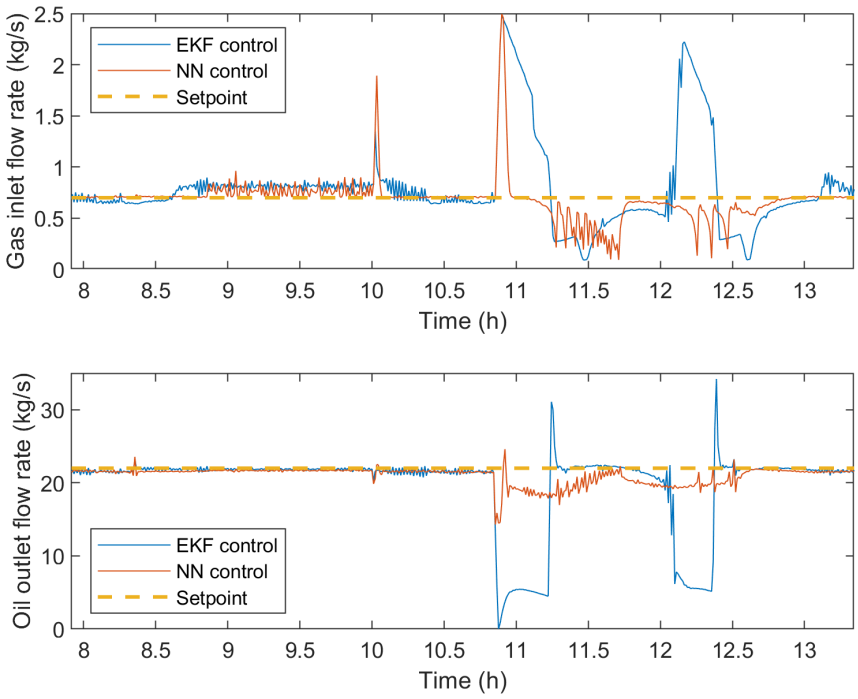


Figure 4.22: Process response to disturbances. P_{res} reduced between 8.3h and 10h, PI increased between 10.8h and 12.5h.

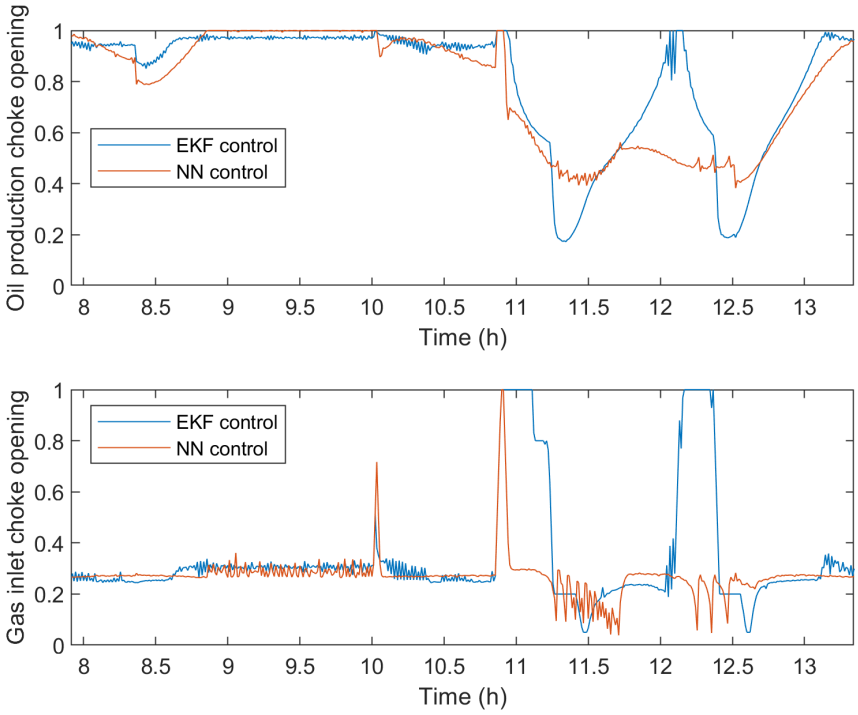


Figure 4.23: Process response to disturbances. P_{res} reduced between 8.3h and 10h, PI increased between 10.8h and 12.5h.

As there are other control strategies to compare the controller to, visual inspection is not enough anymore. So it was decided to use a numerical metric, RMSE. A summary of the results can be seen in Table 4.2. In general the neural network NMPC was better than the EKF NMPC in keeping the gas injection around the setpoint, and successfully avoided slugging during the increased PI disturbance. However, it performed worst in keeping the oil production at the appropriate rate during the setpoint changes, where the EKF NMPC managed to significantly reduce the offset. Neither of them managed to remove the offset between the setpoint during slugging rejection, both predicting the system would suffer from slugging if the production was increased.

Table 4.2: Summary of the control experiments results, RMSE.

	Gas injection error			Oil production error		
	Perfect	EKF	NN	Perfect	EKF	NN
Slugging rejection	0.0643	0.129	0.0784	0.617	1.78	1.57
Gas inlet sp change	0.00401	0.0379	0.00612	0.0154	0.188	0.693
Oil outlet sp change	0.326	0.347	0.349	0.0570	0.162	0.354
P_{res} reduction	0.102	0.107	0.0748	0.0569	0.246	0.401
PI increase	0.230	0.765	0.366	0.413	10.7	2.34

The results of this work were published in the paper "Development of a Nonlinear Model Predictive Control for Stabilization of a Gas-Lift Oil Well" (ROJAS SOARES *et al.*, 2022).

4.5 Summary on how the proposed methodology was used

The methodology proposed in Figure 3.1 was used to align process knowledge with the data analysis, allowing for an effective feature selection that was used in both the internal state estimation and the process control. It also helped to pursue a substantial control improvement through the "Get more data" loop, in which new data was generated using a different and faster state estimation technique that used more sensor information, therefore being more information rich data.

It also inspired two new control strategies, which provided more clarity about the effectiveness of the control strategy using NN. This new controls strategies also led to a review of the employed metrics as visual inspection was not enough and numerical metrics were required.

Chapter 5

Oil well fault detection

5.1 Problem description

There is an increasing demand for better safety and productivity in offshore oil wells. The open seas are harsh environments and faults, even if they are rare, can cause significant damage. Operational failure in oil wells may lead to productivity loss, environmental disasters and damage to the equipment. It is important to have an abnormal event management system in place to warn the operators that a fault is happening or about to happen, and which fault is happening.

To allow for more studies on the area, Petrobras released the 3W dataset, described in VARGAS *et al.* (2019). The main objective in this dataset is to detect the faults and their transients in the process in a timely manner, although it allows for other types of studies, like rare abnormal event detection, data treatment and selection, etc. The dataset includes 8 types of faults and is still being augmented, with regular updates.

In the present case study, machine learning models will be used to detect the faults in the 3W dataset. These models are expected to warn operators about issues in the process and help them to mitigate and solve those issues.

Figure 5.1 illustrates the process.

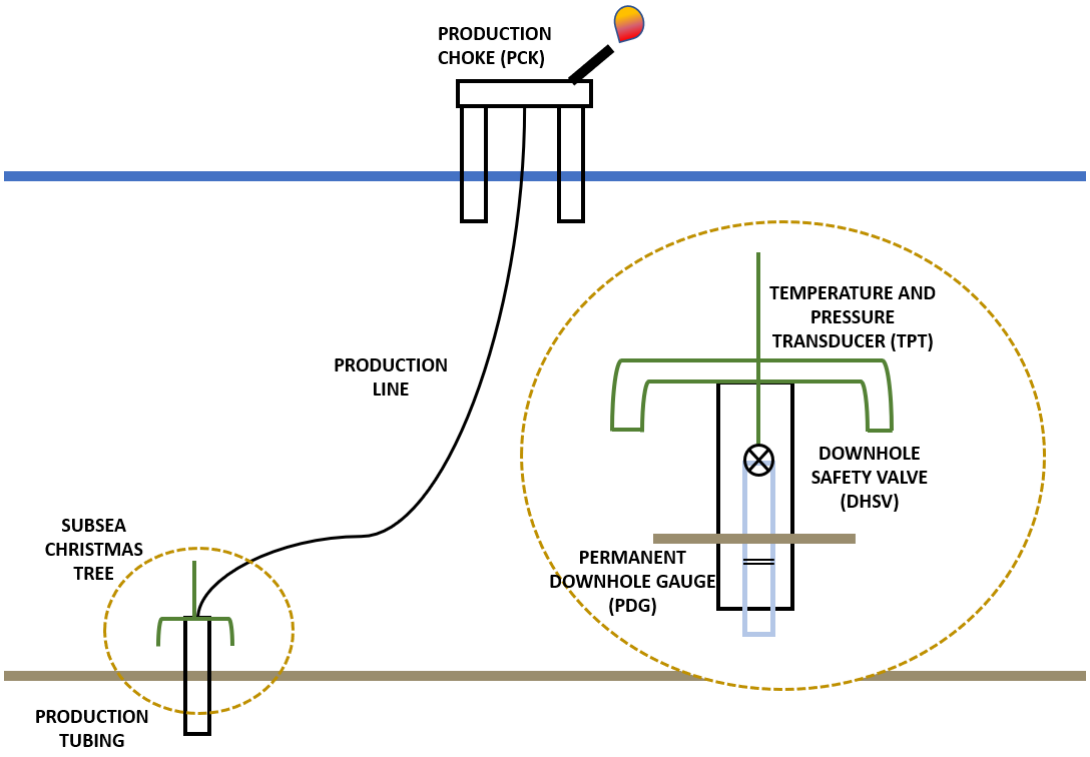


Figure 5.1: Offshore oil well illustration.

5.2 Literature review

There are some works using this dataset which will be discussed below.

MELO *et al.* (2022) did an overview of this and other benchmark datasets and models relevant to chemical engineering. They made an exploratory data analysis of the dataset, and confirmed some results found regarding, albeit with a different methodology, the overly interpolated periods and process drift of each well over time.

MARINS *et al.* (2021) used the first four standardized moments of each sensor, plus the maximum, minimum and the three quartiles, extracted every 50 seconds. They did not perform data cleaning or missing value imputation, but used a PCA to reduce the number of features. For data separation they used a train/test separation of 70/30%, and for validation used 5-fold cross-validation with the data separated by instance. They used random forest for classifying the status of the process. The work focused on accuracy and proposed three setups for fault detection: the first classifying between normal and faulty operation, the second classifying all classes in a one-vs-rest style, and the third using the native multiclass classification procedure from random forest classifier. They achieved an overall 94% accuracy, despite the accuracy of the real cases being significantly lower, ranging from 0% for fault 8 to 95.4% for fault 4, with a median of 86.0%.

CARVALHO *et al.* (2021) removed the features referent to the gas lift, which is not always used since the wells are naturally flowing. They extracted the maximum, minimum, median, mean and variance, with a sliding window of 900 seconds, and they removed the missing values, but kept the outliers. For validation they used 2 strategies: one with random cross-validation of all concatenated data, and another separating the data per well. They focused in flow instability, class 4 of the dataset. For the more realist second strategy, they achieved 67.16% and 68.06% macro F1-score and for accuracy: 87.42% and 81.62%.

DO NASCIMENTO *et al.* (2021) used time delayed stacked autoencoders to reduce the dimensionality of the dataset before feeding it to two anomaly detection models: isolation forests and one-class support vector machines. They interpolated the missing values and standardized the data. They used only normal condition samples for training and tested against faults and other normal conditions. They achieved F1-scores ranging between 73.30% for fault 7 and 99.39% for fault 2. They did not report any results for faults 3 and 4.

TURAN and JÄSCHKE (2021) also used several statistics for feature generation along with their changes and the coefficients of a polynomial model, with the time window being treated as an hyperparameter that is later optimized. They did not find feature selection to improve the results, but this may be due to overfitting and data leakage as they did not separated the data in a time conscious manner. They achieved an overall F1-score in the test data ranging from 49% for fault 2 and 95% for fault 4, with a macro average of 85%. Accuracy ranged from 60% for fault 2 to 98% for fault 1.

5.3 Materials and methods

In this thesis, the 3W oil well dataset, described in VARGAS *et al.* (2019), was used. The dataset is comprised of 9 folders containing csv files regarding normal operation and 8 types of fault. Those files contain information from 21 offshore oil wells. The data can be either real data, simulated data generated by OLGA simulator or hand drawn by experienced operators. The data is sampled every second.

There are 5 sensors in every well. One in the Permanent Downhole Gauge (PDG), two at the Temperature and Pressure Transducer (TPT) and two at the Production Choke (PCK). There are no consistent flowrate, phase or composition measurements.

- Pressure at the PDG (P-PDG).
- Pressure at the TPT (P-TPT).

- Temperature at the TPT (T-TPT).
- Pressure upstream of the PCK (P-MON-CKP).
- Temperature downstream of the PCK (T-JUS-CKP).

If gas lift is being used, there are 3 additional sensors, that will not be used here since they spend most of their time with Not a Number value.

- Pressure downstream of the gas lift (P-JUS-CKGL).
- Temperature downstream of the gas lift (T-JUS-CKGL).
- Flowrate of the gas lift (QGL).

For metrics, F1-score, precision and recall were used according to the guidelines proposed by VARGAS *et al.* (2019). Accuracy will also be reported as it is an easy to interpret metric in classification problems and is used in other papers about the 3W dataset. Precision is defined as the amount of positive samples classified as positive divided by the amount of samples classified as positive, Equation 5.1. In this case positive stands for the occurrence of a fault. Recall is defined as the amount of positive samples classified as positive divided by all positive samples, Equation 5.2. F1-score is the harmonic mean between precision and recall, Equation 5.3. Accuracy is the percentage of samples the model classifies right, Equation 5.4. Accuracy and F1-score will be reported in the main text while precision and recall will be reported in Appendix C.2. In the following equations, true refers to correct classification, false refers to incorrectly classification. e.g., a false negative means a sample incorrectly classified as negative.

$$Precision = \frac{true\ positive}{true\ positive + false\ negative} \tag{5.1}$$

$$Recall = \frac{true\ positive}{true\ positive + false\ positive} \tag{5.2}$$

$$F1\text{-}score = \frac{2}{recall^{-1} + precision^{-1}} \tag{5.3}$$

$$Accuracy = \frac{true\ positive + true\ negative}{all\ samples} \tag{5.4}$$

5.4 Results

5.4.1 Data formatting

The data comes in 9 folders depending on the type of fault it studies. Each folder contains CSV files with 2 to 5 hours of sensor data sampled every second.

It also contains a class label.

To facilitate the analysis the data was downsampled to 30 seconds. 30 seconds was chosen because the original paper informed that the fastest fault, spurious closure of DHSV, may need only a 5 minutes window to be observed, so the sampling time was chosen as one tenth of that. The data was concatenated into a single CSV file. It was added information about from which folder, file and well the data belongs, as well as the data origin, whether it was hand drawn, simulated or from a real well. This reduced the dataset size from 4.78 GB to 166 MB.

The hand drawn pressure data was given in bar, so they were converted to Pa in accordance with the other pressure measurements in the consolidated dataset. They were not used in the end, but future studies wanting to use this part of the dataset should be aware of this unit conversion, as most open source data manipulation tools do not have automatic unit conversion.

5.4.2 Data quality evaluation

The sensors in harsher conditions are expected to be less reliable, while the ones at the surface are probably the most reliable since they can receive regular maintenance. This poses the order from less to more reliable: P-PDG; P-TPT and T-TPT; P-MON-CKP and T-JUS-CKP.

Looking for sensors with valid information showed that P-TPT and T-TPT are the most reliable sensors. However, for a more in-depth analysis a second derivative test was applied. All data in this dataset is interpolated, due to the behavior of the system that extracted the dataset and the historian. However, long-term interpolation may imply communication issues or gaps in the historian. Real-life data has noise, which will be magnified if the numerical second derivative is taken, while linearly interpolated or constant data will have a zero second derivative. A numerical second-order approximation of the second derivative over 60 seconds is given by Equation 5.5, using in this case $h = 30s$. This analysis showed that many P-TPT or T-TPT values were interpolated for more than 1 minute, as can be seen in Table 5.1. The samples where invalid and interpolated values appear can be seen in Figures 5.2 and 5.3, respectively. An example of overly interpolated data is shown in Figure C.1.

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} \tag{5.5}$$

Table 5.1: Percentage of invalid (offline) or interpolated values per sensor.

Sensor	% offline	% interpolated
P-PDG	63.85%	14.38%
P-TPT	0.45%	32.32%
T-TPT	0.45%	25.5%
P-MON-CKP	8.04%	8.90%
T-JUS-CKP	12.11%	13.72%

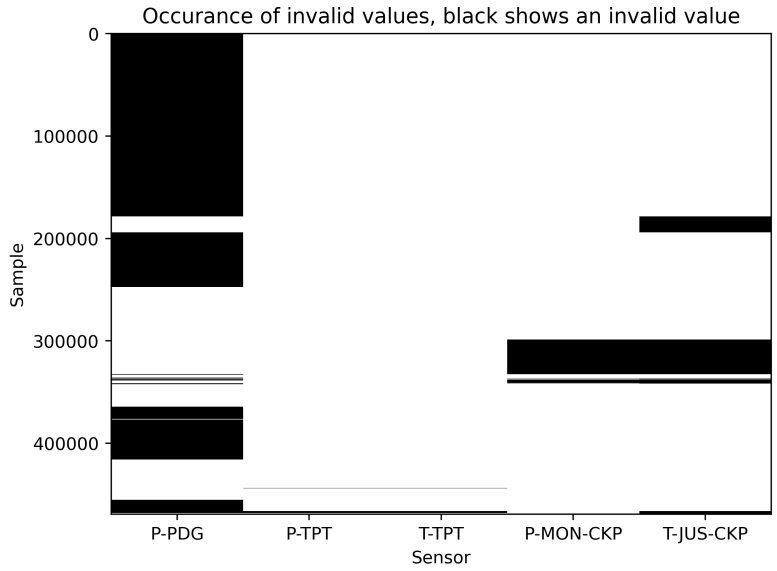


Figure 5.2: Occurance of invalid values.

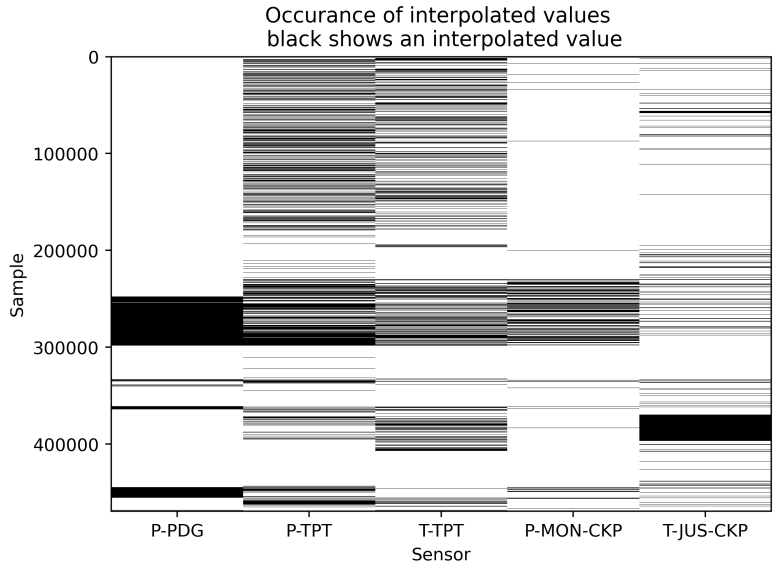


Figure 5.3: Occurance of interpolated values.

Another aspect that should be brought to attention is a low amount of regular condition data in wells 9 to 20. Wells 14 and 18 have no normal condition data,

well 19 has only 3 minutes of normal condition. This diminishes how the models can be tested as they can not be tested properly for false positives. Also any well-dependent normalization will carry higher uncertainty. Table 5.2 shows which wells have no valid values for each sensors.

Table 5.2: Wells in which a certain sensor has no normal values.

Sensor	Wells with no normal sample
P-PDG	1, 2, 4, 5, 9, 13, 14, 18 and 19
P-TPT	14, 18, 19
T-TPT	14, 18, 19
P-MON-CKP	8, 11, 13, 14 and 18
T-JUS-CKP	3, 8, 11, 12, 13, 14, 18, 19, 20 and 21

Looking per folder in Table 5.3, it can be seen that most of them do not have enough samples of P-PDG, which is expected given the previous analysis. Folder 2 has many missing values for P-MON-CKP and T-JUS-CKP. Folder 8 has plenty of missing values from all sensors but P-MON-CKP, therefore it will not be analyzed in this work.

Table 5.3: Percentage of invalid sensors per folder.

Folder	0	1	2	3	4	5	6	7	8
P-PDG	69.83%	34.56%	29.76%	3.25%	62.18%	0.00%	100.00%	100.00%	56.22%
P-TPT	0.03%	0.00%	0.38%	0.09%	0.02%	2.15%	0.00%	0.12%	56.18%
T-TPT	0.03%	0.00%	0.38%	0.09%	0.03%	2.15%	0.00%	0.12%	56.18%
P-MON-CKP	10.13%	0.00%	75.80%	0.09%	0.02%	0.00%	0.00%	0.12%	0.00%
T-JUS-CKP	14.78%	0.00%	86.67%	0.12%	0.03%	0.00%	0.00%	0.12%	100.00%

5.4.3 Feature engineering

Three forms of first principle flow estimation were designed, two based on pressure difference and one based on temperature difference. Assuming negligible height difference between the TPT and the PDG, and considering the pressure drop comes from the Downhole Safety Valve (DHSV), as can be seen in Figure 5.1, and that it can be modeled as a regular valve.

$$\tilde{Q}_{p1} \propto \sqrt{P\text{-}PDG - P\text{-}TPT} \tag{5.6}$$

where $\tilde{Q}_{p1}$ is the estimated flowrate. It should be noticed that DHSV opening is assumed constant, which is not true, as there is a closure fault for this valve. If the DHSV is closed, $P\text{-}PDG$ is expected to increase, while $P\text{-}TPT$ would decrease, making the flow estimation increase, while the true flow is negligible.

The second pressure equation considers the Bernoulli equation, based on the pressure difference between $P\text{-}TPT$ and $P\text{-}MON\text{-}CKP$. The height difference cannot be considered negligible, so a term considering the pressure loss due to gravitational effect is added. This term should be different from well to well and may change over time, but here it is assumed constant, given the overall lack of information about the wells.

$$\tilde{Q}_{p2} \propto \sqrt{P\text{-}TPT - P\text{-}MON\text{-}CKP - \rho \cdot g \cdot z} \tag{5.7}$$

where $\tilde{Q}_{p2}$ is the estimated flowrate, ρ is the fluid density, g is the gravity constant and z is the height. Simulated data gives an estimation of $\rho \cdot g \cdot z$. As there are some simulated faults that stop oil production like fault 2, taking $\tilde{Q}_{p2} = 0$ and find the smallest value of $P\text{-}TPT - P\text{-}MON\text{-}CKP$, this gives an estimation for $\rho \cdot g \cdot z$. The value found was 1.1 MPa in the simulated data, which will be used here. For the real data, the smallest value is 2.3 MPa, but it is unclear if the flow had really stopped.

The temperature difference equation is given assuming the pipeline to the PCK exchanges heat (W) only with the sea and thermodynamic effects, as gas expansion are negligible, Equation 5.9. Modeling this pipeline as a heat exchanger using the log mean temperature difference, results in Equation 5.8.

$$LMTD = \frac{(T\text{-}TPT - T_{subsea}) - (T\text{-}JUS\text{-}CKP - T_{seasurface})}{log(T\text{-}TPT - T_{subsea}) - log(T\text{-}JUS\text{-}CKP - T_{seasurface})} \tag{5.8}$$

$$W = Cp \cdot Q_t \cdot (T\text{-}TPT - T\text{-}JUS\text{-}CKP) = UA \cdot LMTD \tag{5.9}$$

rearranging it becomes Equation 5.10:

$$\tilde{Q}_t \propto \frac{LMTD}{(T\text{-}TPT - T\text{-}JUS\text{-}CKP)} \tag{5.10}$$

where $\tilde{Q}_t$ is the estimated flowrate, W is the exchanged heat, Cp is the fluid heat capacity, T_{subsea} is the temperature under the sea and $T_{seasurface}$ is the temperature at the sea surface. UA is the overall heat transfer coefficient and $LMTD$ is the logarithmic mean of the temperature difference. Again, T_{subsea} and $T_{seasurface}$ can be inferred from the simulated data. Assuming that when the flow is stopped T-TPT converges to T_{subsea}, the value found is 2.97 °C, similarly for $T_{seasurface}$ and T-JUS-CKP, the value found is 24.1 °C. This equation does not account for fluid chilling due to gas expansion or liquid evaporation at lower pressures. One issue with this estimation is that in case of strong flow decrease, T-TPT cools faster than T-JUS-CKP, making the denominator become to negative and

making $\tilde{Q}_t$ looks like it went to infinity then inverted signal.

These equations are oversimplifications of the process, have no appropriate unit due to the lack of proportionality constants, are dependent on well geometry and oil composition. If flowrate could be inferred with these simple equations, oil well flow estimation would not be such an active field of research, but those transformations may carry some information about the system's state, so they were added and it was later determined whether they carry relevant information for the models.

One issue of these equations is that they need more than one sensor, and a broken sensor may invalidate these new features. As was shown in the data quality analysis, sensor failure is a common issue. Another strategy was to use the derivative of the sensors, to see the direction where the system is moving. They are expected to be relevant for flow instability detection. The derivatives were taken using a Savitsky-Golay (SAVITZKY and GOLAY, 1964) filter of order 1 to smooth the measurement noise.

5.4.4 Unsupervised analysis

The first analysis was a descriptive analysis of the normal condition. It was clear that each well had a different operational condition, as can be seen in Figure 5.4, and the operational condition changed overtime. In the wells with more data samples it was clear the normal operational condition varied overtime. While operational drift is an expected result, the magnitude of the variance indicates that normalization must be flexible for each well and each point in time. So any system developed for well monitoring must be able to be updated over time. In this work it was decided to only update the normalization. It should be noticed that in an eventual deployment of this system the operator must keep a record of what is a normal condition for the well.

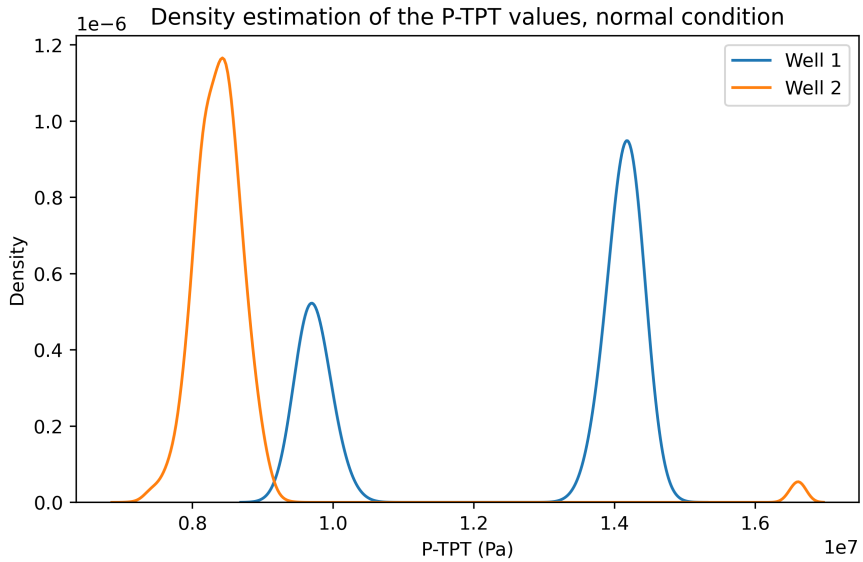
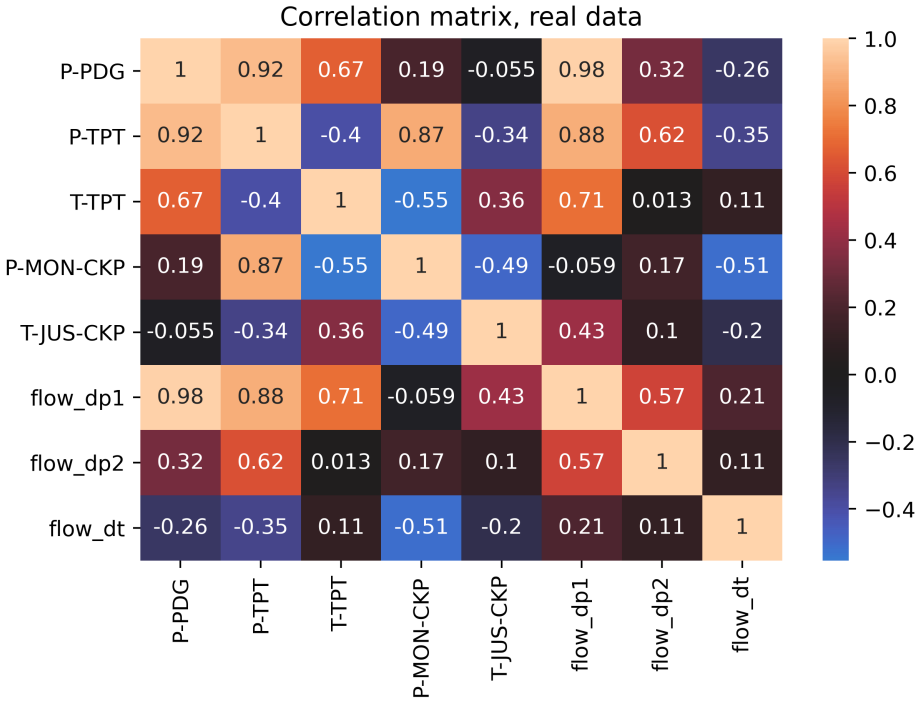
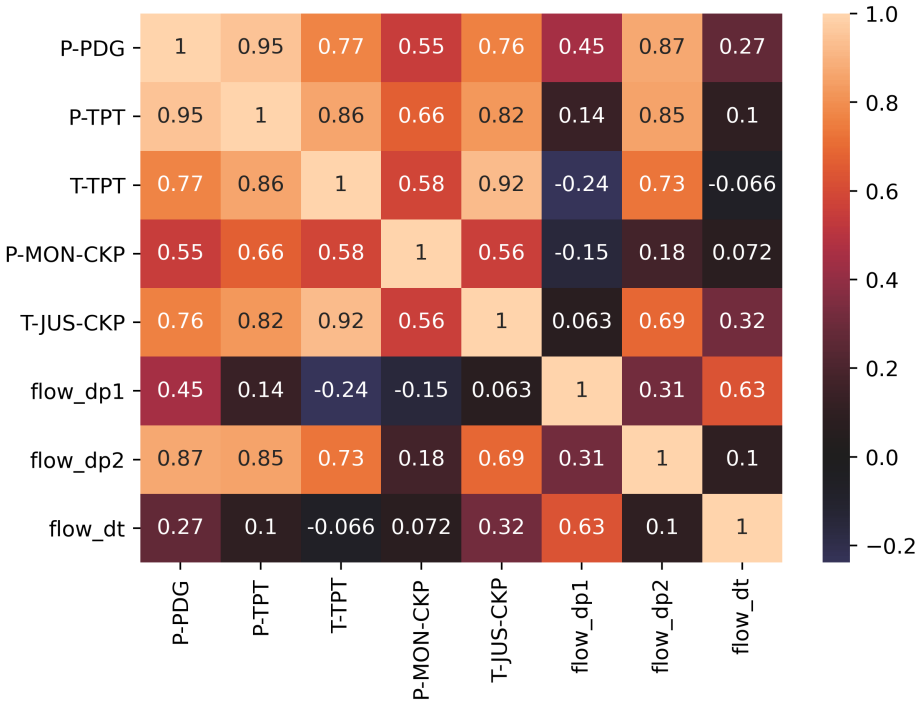


Figure 5.4: Density estimation of P-TPT values.

Correlation analysis was done to the variables, both within each well and in the overall normal condition data. Correlation analysis is sensitive to outliers so it should be performed with care for abnormal condition data. The correlation analysis was done to both the real data and simulated data. Interesting divergences occur: in the real data P-TPT and T-TPT are weakly negatively correlated, while in the simulated data they are strongly positively correlated. In the simulated data T-TPT and T-JUS-CKP are strongly correlated as expected but not in the real data. The correlation analysis also show that since P-PDG and P-TPT are strongly correlated, they likely carry similar information; demonstrating that P-PDG, the less reliable measurement, can be removed without much loss of information. Another thing that should be noted is that $\tilde{Q}_{p2}$ has little correlation with either $\tilde{Q}_{p1}$ and $\tilde{Q}_t$ in the simulation data. Since they are trying to estimate the same thing they were expected to be correlated. This indicates that $\tilde{Q}_{p2}$ may be an ill-engineered feature.



(a) Correlation matrix, real data.



(b) Correlation matrix, simulated data.

Figure 5.5: Correlation matrices for real (a) and simulated data (b).

Reading the concatenated csv files, and reviewing the raw files to make sure an error was not done during concatenation, faults 3 and 4 have no real transition data, while fault 7 have only one real example of consistent fault state, the other

real examples being transient state.

5.4.5 Data selection

Finally, the data select is chosen as following: It was used simulated and real data for training and validation and real data for testing. Although there is a version 1.1.1 of the dataset available as of June 2023, the 1.0.0 version of the dataset was used in this work. The features P-PDG and $\tilde{Q}_{p1}$ were not used since P-PDG is too unreliable. Even in folders where P-PDG is present it is likely that it will stop working in the future, invalidating the model. In the first iteration all variables but P-PDG and $\tilde{Q}_{p1}$ were included. In the second iteration, the knowledge created in the first iteration was applied to select some features. In the third iteration, the first derivative was added and passed through feature selection. For further feature selection, it was used both the feature importance given by the random forest, and the permutation importance. The permutation importance is defined as: if a feature is scrambled during prediction, how much does the accuracy decreases in response. If a feature is more important, the bigger will be the decrease. This is better explained in Appendix A.3.

For the test dataset, the data of some wells were hold out, to simulate the model being used against a never-seen-before well. The models were validated using a grouped K-fold scheme, where the data is divided in 4 hours blocks, to reduce data leakage from future data, and then tested with the held out data. The wells whose data were held out are summarized in Table 5.4. The main metric to be reported is F1-score, per suggestion of the original 3W dataset paper, but accuracy is also reported. Precision and recall are reported in Appendix C.2. The mathematical formulation of F1-score, precision and recall is given in Appendix A.2.

Table 5.4: Held out well data.

Fault	Held out well
1	1
2	10
3	1
4	7
5	15
6	2
7	1

5.4.6 The development iterations

In the first iteration the dataset was constructed with process knowledge, data cleaning, and feature engineering, trained with data analysis based feature selection. Then the second cycle was done using machine learning based feature selection, both with feature importance given by the Random Forest and permutation based. It also adds a per well normalization scheme. The third step included the addition of sensor derivatives and tests of their relevance. The final step was hyperparameter optimization and model selection; as it is the most time consuming, it was done last. The results in Table 5.5 are the results for the first and last iteration, with the intermediate results being displayed in Appendix C.2.

Table 5.5: Results of the methodology application, validation data.

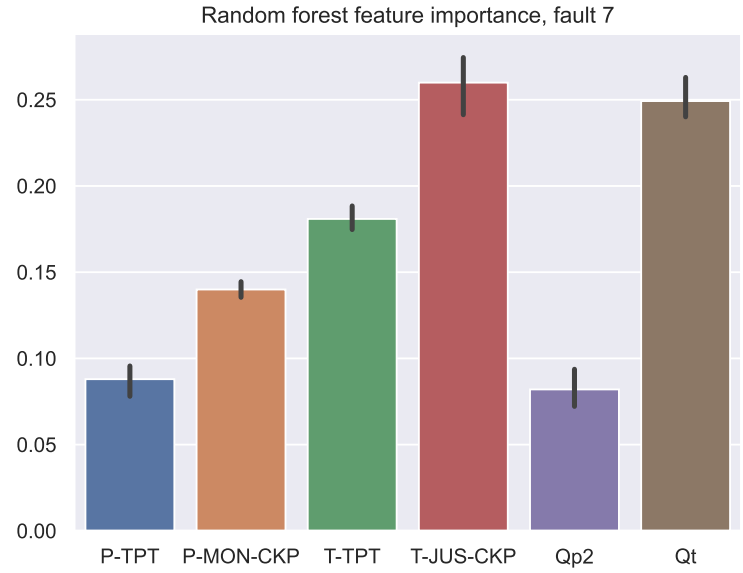
	First iteration		Last Iteration	
Fault	Accuracy	F1-score	Accuracy	F1-score
1	85.65%	95.34%	95.89%	99.69%
2	99.44%	95.32%	99.83%	99.63%
3	99.85%	99.81%	99.87%	99.83%
4	94.91%	89.56%	97.61%	94.29%
5	72.02%	98.43%	98.20%	99.62%
6	74.86%	96.03%	92.48%	99.69%
7	99.85%	97.55%	99.76%	95.47%

The relevant features are described in Table 5.6 along with the best type of model found. In 6 models the inferred flowrates were deemed important, indicating the process knowledge based on feature engineering was relevant. 6 models used the derivatives, referred here by the suffix "_der" after the sensor name. The relevant features were selected by analyzing which sensors displayed low feature importances according to both Random Forest and Permutation feature importances. For example: As can be seen in Figures 5.6a and 5.6b, P-TPT and Qp2 were deemed not relevant for the fault 7 classifier, so they were excluded. The best type of model was chosen by cross-validation with random search.

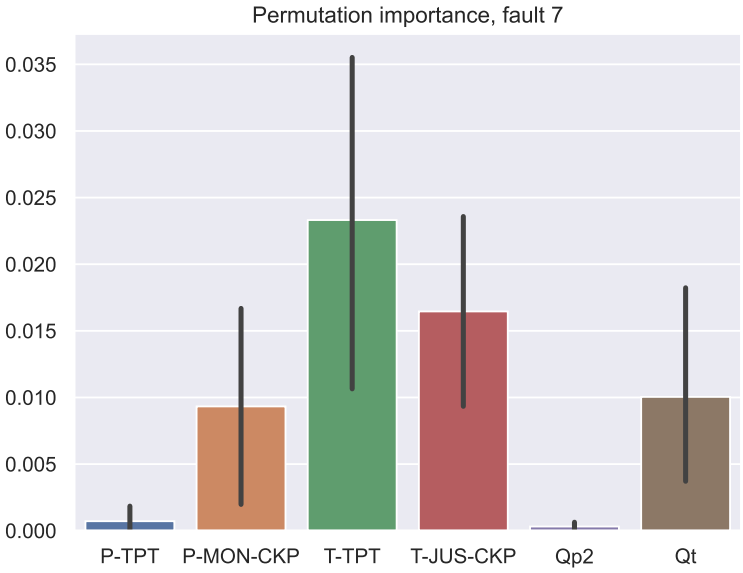
Table 5.6: Relevant features and best model for each fault.

Fault	Relevant features	Best model
1	P-TPT, T-TPT, Qp2, P-TPT_der, T-TPT_der, P-MON-CKP_der, T-JUS-CKP_der	GB
2	P-TPT, T-TPT,P-TPT_der, T-TPT_der	RF
3	P-MON-CKP,T-TPT, Qt, P-TPT_der, T-JUS-CKP_der	GB
4	P-TPT, P-MON-CKP, T-TPT, Qp2, P-TPT_der	GB
5	P-MON-CKP,T-JUS-CKP, Qp2,P-TPT_der, P-MON-CKP_der	GB
6	P-MON-CKP, T-JUS-CKP, Qp2, P-MON-CKP_der, T-JUS-CKP_der	GB
7	P-MON-CKP, T-TPT, T-JUS-CKP, Qt	MLP

The features and permutation importances, given for each fold, were collect and their mean plotted, along with a small error bar equal to their standard deviation. Examples of such plots are Figures 5.6a and 5.6b, for fault 7. It can be seen that the Random forest feature importance always attribute some importance to a feature due to bootstrapping, while for the Permutation importance the decrease can be zero or even negative. They generally agree, with some details of difference. For example, in the permutation importance T-TPT was the most important feature, while in the Random Forest feature importance T-JUS-CKP and Qt were the most important. One aspect that should be noticed is that permutation importance has more variance than the Random Forest feature importance, due to being a differential measure and being calculated on the validation dataset, in which the model was not trained on.



(a) Feature importances.



(b) Permutation importances.

Figure 5.6: Feature importance for the random forest (a) and permutation importances (b).

The test dataset was left for last on purpose to avoid data leakage, and not analyzed until the final iteration. In a real industrial project, the real time process could be used as the test dataset. As can be seen in Table 5.7, there are unusual results, like in the results of the last iteration model of fault 1 and first iteration model for fault 7, both which have high accuracy but low F1-score. This is due to the unbalanced nature of the dataset. There are much more normal condition

data than fault data, so a model that predicts only normal condition has high accuracy and low F1-score, which happened in the aforementioned cases.

In summary, for faults 2 to 6 the methodology provided a significant improvement, for fault 7 there were small improvements as this fault were hard to predict on the first iteration, and there were a decrease on performance in fault 1. In this case the suggestion is to roll back the first model. Overall, the results were better than CARVALHO *et al.* (2021) for fault 4, achieving a 22.19% improvement. Compared to MARINS *et al.* (2021) better results for faults 2 (2.95% improvement), 5 (11.5% improvement) and 6 (28.5% improvement) were achieved. The methodology created a median improvement of 13.25% in accuracy and 44.23% in F1-score.

Table 5.7: Results of the methodology application, test data.

	First iteration		Last Iteration		Improvement	
Fault	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
1	98.81%	83.03%	97.08%	32.84%	-1.73%	-50.19%
2	24.52%	38.13%	91.75%	86.34%	67.23%	48.21%
3	3.98%	3.83%	17.23%	54.40%	13.25%	50.57%
4	0.00%	0.00%	93.04%	90.25%	93.04%	90.25%
5	68.34%	48.12%	94.80%	92.35%	26.46%	44.23%
6	94.25%	46.16%	99.49%	80.58%	5.24%	34.42%
7	97.75%	32.95%	97.76%	32.96%	0.01%	0.01%

5.4.7 Model analysis

Here model analysis is performed in a more explicit manner, as it is essential for the methodology, and should be included in any presentation and reports whenever the product is delivered or demonstrated for stakeholders.

It was expected that Faults 6 and 7, quick restriction and scaling at the PCK respectively, to have relevant sensors at the PCK. In fact the model for fault 6 uses only sensors at the PCK and a flow estimation. However, feature selection for Fault 7 also found T-TPT to be relevant. This is probably because scaling is a factor of the fluid temperature over the whole pipeline, therefore information on its temperature on the beginning of the pipeline is important. A similar analysis could be done about fault 2 if the sensors on the surface worked, but as seen in Table 5.3, they spend most of their time at fault, so P-TPT and T-TPT were chosen based solely on their availability.

Some derivatives and a flow inference were used for faults 3 and 4, severe slugging and flow instability, respectively. This is also expected as both are

sudden phenomena related to the flowrate, causing fast shifts in the readings therefore generating strong derivatives.

Analyzing the behavior of the classifier over time on the test dataset (illustrated in Fig. 5.6 for fault 6), which would be similar to a real-time scenario, it can be seen that in the last iteration the model notices the shift to transient state quickly, and starts to shift to predict fault state around 26 minutes before the transition occurs, but only steadily predicted the fault 20 minutes after it occurred. These time frames match the evaluation window Petrobras operators use to detect fault 6, of around 15 minutes.

Taking a look at the probabilities, it can be seen that the probabilities for normal condition shift fast, while the probabilities of transition and fault states are smoother, which makes sense since the confusion are mostly between them. The last model has the confidence in its prediction steadily increased with time, while the first iteration model shows a steady probability estimation.

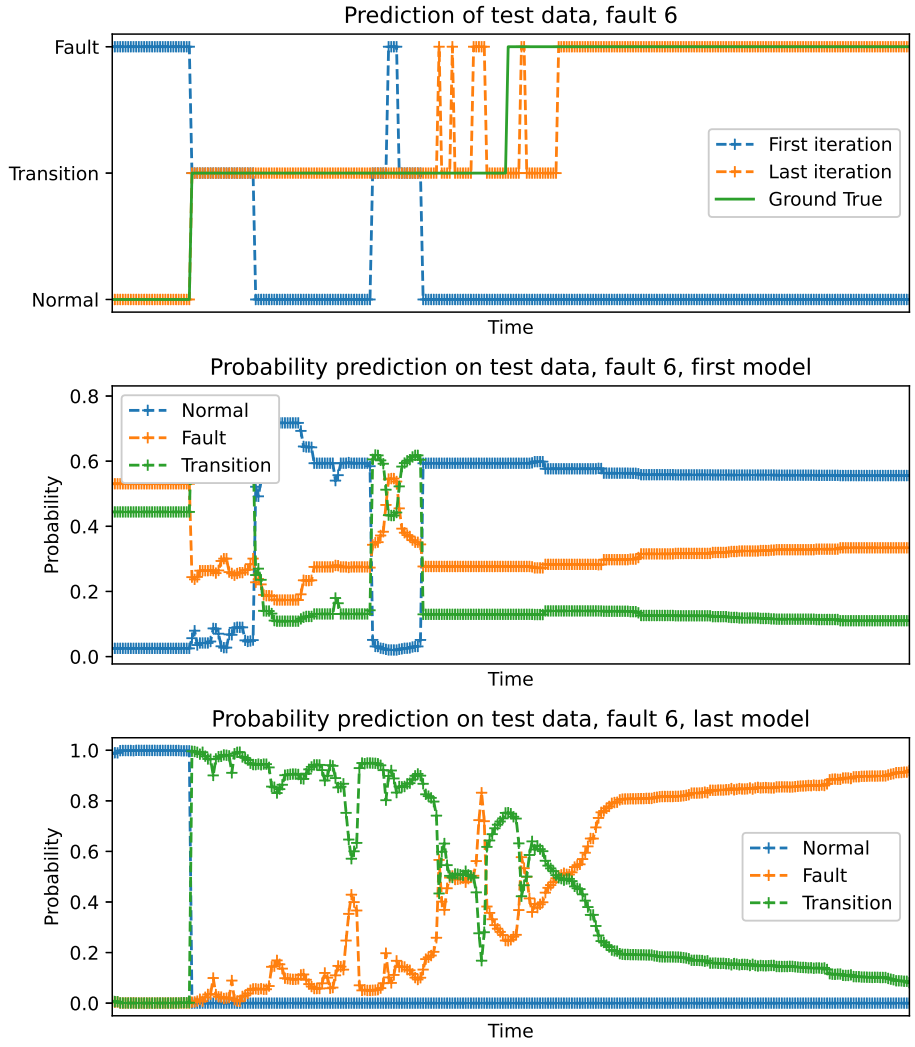


Figure 5.7: Example on how prediction works for fault 6.

Taking a look at the partial dependence plot for the derivative of P-TPT, Figure 5.8, more extreme values of the derivative leads to a higher chance of the process being classified as fault 4. This matches engineering knowledge, as flow instability would lead to data with higher fluctuations, therefore bigger derivatives.

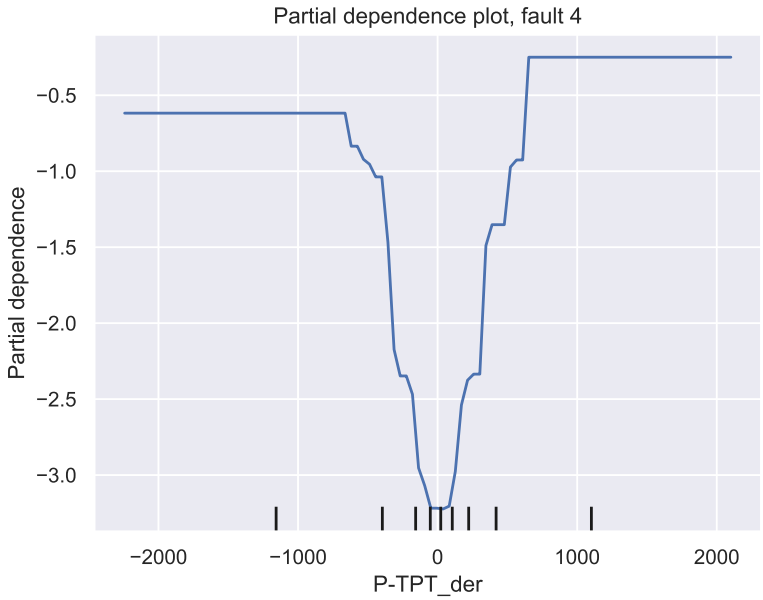


Figure 5.8: Example on how P-TPT_der affects the model for fault 4.

5.5 Summary on how the proposed methodology was used

The methodology was used to continuously create new features, and validating their utility both by modeling analysis and rationalizing their choice in terms of expected fault behavior. Features focused on dynamic behavior were used to encode time-dependent information on the dataset in a way not previously seen in the literature. Rationalizing the feature choice would also facilitate model acceptance in an industrial project using the models. Better results were achieved in the test dataset than MARINS *et al.* (2021) for faults 2 (2.95% improvement), 5 (11.5% improvement) and 6 (28.5% improvement). Improved outcomes were also obtained in comparison to the results of CARVALHO *et al.* (2021) for the F1-score of fault 4, while the difference may be due to different dataset treatment choices.

The methodology application was limited due to lack of access to a specialist or operator in the process to help guide and validate it, since the process

knowledge available was limited to general engineering knowledge and information about the unique features of the wells was unavailable. This matter will be addressed in the industrial case studies.

Chapter 6

Low density polyethylene production

6.1 Problem description

Low Density Polyethylene (LDPE) is a thermoplastic polymer produced through the polymerization of ethylene. It has a density between 0.917 and 0.930 g/cm^3 and is more branched than high density polyethylene (HDPE) . It was the first polymer industrially manufactured in 1933. Its lower density is due to branching, on about 2% of the carbon atoms, which increases the volume occupied by the molecule. It is used in applications that require flexibility but not strength like films and some containers (SAUTER *et al.*, 2017).

The most important rheological characteristics of the polymer are melt flow index (MFI) and density. MFI measures how easy the polymer flows and is important in extrusion processes. It is correlated with viscosity and molecular weight, higher MFI implies lower viscosity and branching, and lower molecular weight.

In order to measure MFI, a sample of the polymer is heated beyond its melting point and forced to flow through a capillary tube using a piston under a specified weight. The method is described in the standards ASTM D1238 and ISO 1133 (ISO 1133-1:2011). The measurement precision depends on the polymer and MFI being measured, but stays around 3-8 %. The purpose of this work is to develop a soft sensor for MFI from reactor variables, particularly online measurements. This soft sensor will help operators to keep the product under the desired specifications and reduce product loss.

6.2 Literature review

Polyethylene polymerization in this and other high pressure processes are described by the following reactions (ZHOU *et al.*, 2001).

Initiator decomposition
$R-O-O-R \longrightarrow 2\,R-O\cdot$ (Reaction 4.1)

Chain initiation
$R-O\cdot + CH_2=CH_2 \longrightarrow R-O-CH_2-CH_2\cdot$ (Reaction 4.2)

Propagation
$R-O-(CH_2)_n-CH_2\cdot + H_2C=CH_2 \longrightarrow R-O-(CH_2)_{n+2}-CH_2\cdot$ (Reaction 4.3)

Chain transfer to monomer
$R-O-(CH_2)_n-CH_2\cdot + H_2C=CH_2 \longrightarrow CH_2=CH-(CH_2)_{n+1}-CH=CH_2 + R-O-CH_2-CH_2\cdot$ (Reaction 4.4)

Combination termination
$R-O-(CH_2)_n-CH_2\cdot + R-O-(CH_2)_k-CH_2\cdot \longrightarrow R-O-(CH_2)_{n+k+2}-O-R$ (Reaction 4.5)

Disproportionation termination
$R-O-(CH_2)_n-CH_2\cdot + R-O-(CH_2)_k-CH_2\cdot \longrightarrow R-O-(CH_2)_{n-1}-CH=CH_2 + R-O-(CH_2)_k-CH_3$ (Reaction 4.6)

The reaction starts with the initiator decomposition, Reaction 4.1, in this case an organic peroxide. The decomposition creates two free radicals, which then attack the ethylene double bond, attaching themselves to the ethylene and transferring the radical to a carbon atom, Reaction 4.2. This is the beginning of the chain, that then attacks another ethylene molecule to grow, Reaction 4.3.

Sometimes the chain transfer to a monomer, terminating a chain and beginning other, Reaction 4.4. The polymerization terminates when two chains attack each other, forming either one long chain; a combination termination, Reaction 4.5; or forming two chains, one with a saturated carbons and another with an unsaturated bond, Reaction 4.6.

There are many works trying to estimate MFI in an industrial setting. However, there is no uniform numeric metric in these works, although visual inspection is popular for obvious reasons. In fact, during this case study custom

metrics were developed as will be seen in subsection 6.4.2.

HO LEE *et al.* (2008) estimate MFI of a HDPE plant from a dynamic empirical model to reduce transition time. Model accuracy was evaluated by visual inspection and by integrated squared error. The main features used are reactants concentration and temperature.

FARSANG *et al.* (2015) used dynamic PCA to estimate MFI from a polypropylene plant. They started with 22 variables but reduced them to 8 principal components. They also evaluated model accuracy by visual inspection. LIU *et al.* (2017) employed local gaussian process regression, and obtained relative error between 10-12 %. They did not specify exactly which variables were used but they included reactor pressure, temperature, liquid level, and catalyst flow rate.

FEIL *et al.* (2004) present a semi-mechanistic modeling approach where neural networks describe the polymerization along with DAEs. Again the paper used visual inspection to evaluate the model.

The subject of this case study is an LDPE autoclave process, owned and operated by Braskem. Fresh ethylene is supplied to the plant and compressed along with ethylene from the extruder feeding vase by the primary compressor. After the first compressor, recycled ethylene from the high pressure separator is added to the stream along with the modifiers if necessary. The secondary compressor then compresses the ethylene stream until the desired reactor pressure. The stream is injected through 4 feeding valves in reactor 1 and 1 feeding valve in reactor 2 (HAM and RHEE, 1996).

Organic peroxides are the catalysts of this reaction. They are injected in 3 levels in the first reactor and 2 levels in the second reactor. Peroxide injection controls the reaction temperature, as polymerization is strongly exothermic and there is no cooling in the reactor, only in the streams between the reactors. The reactors also have thick walls to stand the high pressures of the process, so they are considered adiabatic.

Liquid polyethylene and non-reacted ethylene go through a pressure-reducing valve, a cooling heat exchanger and into the high-pressure separator. Most of the non-reacted ethylene is separated there. Conversion is around 20% in each pass. The ethylene goes to the recycle stream and reenters the process between the first and second compressors (MOEN *et al.*, 1999).

The bottom stream of the separator is cooled and sent to the extrusion feeding vase, were further ethylene is removed, recompressed and returned to the process before the first compressor, as mentioned before. The extruder granulates the polymer, that is then cooled, bagged and stored. The LDPE used for MFI measurement is sampled just after the extruder. The process is shown in Figure 6.1.

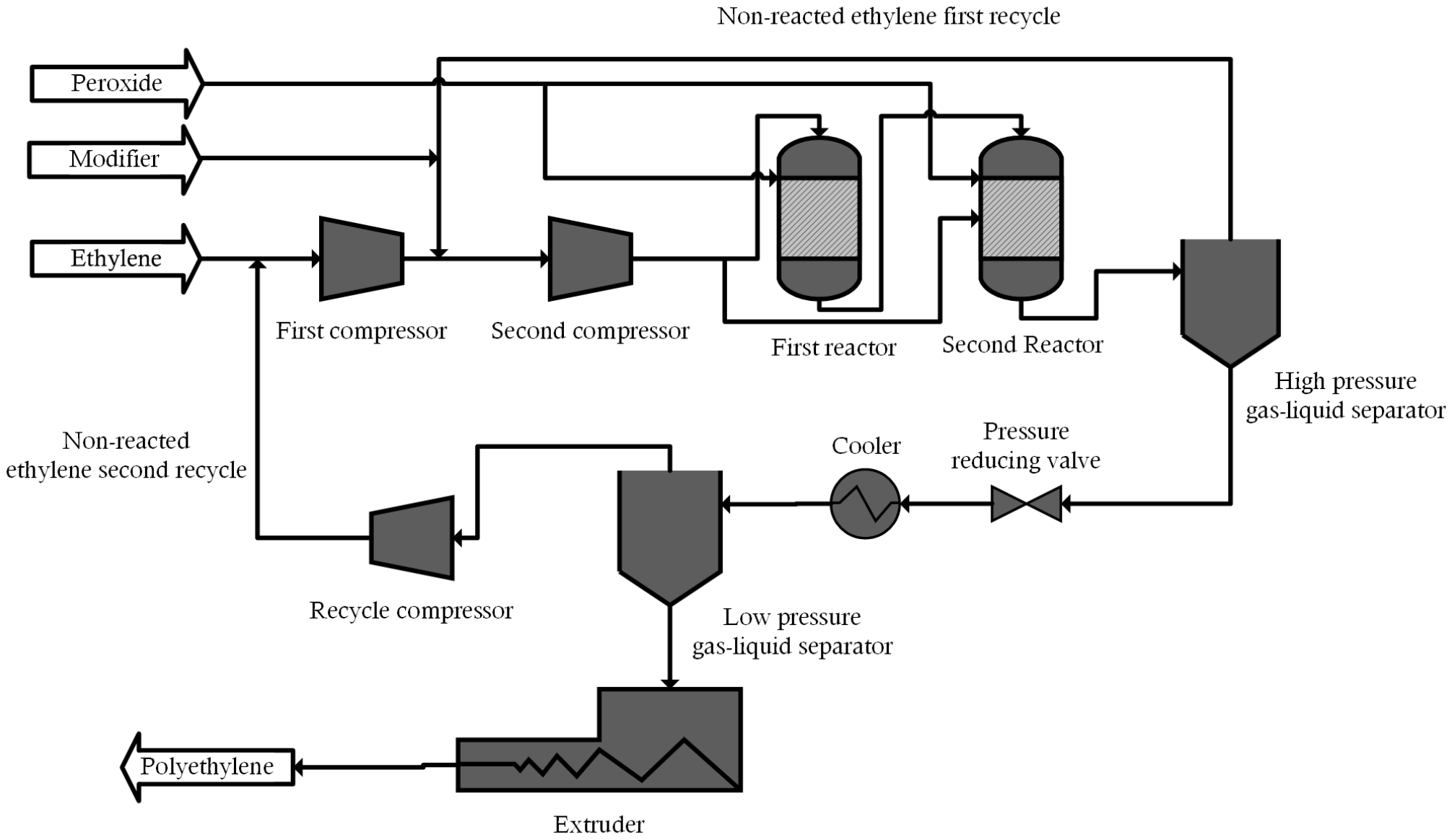


Figure 6.1: LDPE process flowchart.

The process has a residence time in the order of minutes, however the varying recycle streams and multiphase nature of the reaction make exact estimation difficult. Overall, the process dynamic takes around 40 minutes according to the plant documentation.

There are 2 major sources of uncertainty on the MFI data: the MFI sampling time and the extruder. The dataset was inconsistent, in some samples the polymer was received in the laboratory before it was sampled or at the same time. The difference between timestamps is usually not much, around 5-10 minutes, but as the process residence time was in the order of minutes this uncertainty affected the model. Other source of uncertainty is the extrusion. Extrusion changes the MFI, sometimes strongly (DING *et al.*, 2022). It was assumed that whatever change in MFI caused by the extruder was consistent.

6.3 Materials and methods

Data processing, models, visualizations and analysis used in this work were done in Python. All implemented models were developed using Scikit-learn (PEDREGOSA *et al.*, 2011).

MFI is measured every 2 hours, however due to the dynamics and residence times previously mentioned there are probably fluctuations in the MFI that are not sampled. The process already has a rough MFI estimate using the extruder electrical current. For LDPE with higher MFI, less power is required by the extruder to pass the polymer and the extruder current decreases. However, this estimate is qualitative and does not allow for automatic control during production. MFI control is done by modifier addition, propene or butene depending on the grade. A chromatographer measures butene and propene concentration along with ethane, propane and CO_2. The chromatographer has a 10 minutes delay and its measurements may not be reliable, as it needs regular recalibration.

Sensor data was sampled from the database every minute from 2012 to 2017, with gaps due to plant shutdown. During most of 2018 the plant operated under undesirable conditions so no data from this period is available. The initial dataset has 1.8 million samples. The dataset consists of sensors the operators believe to be related to MFI, even if only slightly. The measurements selected were: 5 concentrations; 5 peroxide flows, recycle flow, modifier flow, two pressures from the first reactor, 1 from the second reactor; 18 reactor temperatures corresponding to the zones of each reactor, 9 for each reactor; 4 temperatures regarding interreactor cooling; temperature at the cooler before the high pressure separator, and 5 miscellaneous measurements regarding compressor operation and plant productivity. There were also 9 sensors related to the extruder, includ-

ing extruder current and several temperatures.

The total number of reactor measurements is 43, some highly correlated with each other, especially the temperatures. Measurements outside sensor limits were removed. There were no direct measurements of total ethylene inflow but it can be inferred from the compressor measurements.

MFI data was extracted from another database. MFI dataset was created by manual annotation by the lab technicians. Besides MFI, the following data was also included: the grade being produced, the time in which the polymer was sampled from the extruder outlet, the time in which the lab received the sample, and upper and lower bound for the grade MFI.

6.4 Results

6.4.1 Data treatment and analysis

Data was received in several text files each containing one sensor. Data reformat was merging the sensor datasets and treating them. Correlating sensor data to the lab analysis was a later task. First step of data treatment was removing points with a bad status reading. Second step was removing periods were the process was shut down. These periods can be identified by a productivity tag, a reliable tag that measures the polyethylene output. Third step was removing outliers and above/below 3 standard deviations from the mean. This latter treatment was reversed, as the multimodal nature of the process put many points from less frequent grades outside of the range, as can be seen in Figure 6.2. Given that the association between the grade being produced and sensor data was done in a later step, it was not possible to do this outlier removal per grade. Note that the figures have no scale, as the data is anonymized per the company request.

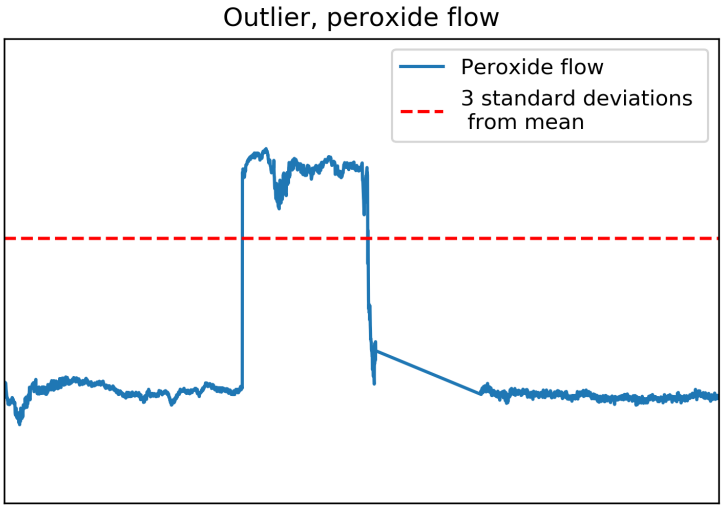


Figure 6.2: Good points that would be removed.

For data cleaning of the lab analysis, first data points with duplicated timestamps were removed. Per operator suggestion, sampling timestamp was chosen as the most reliable timestamp. Later the data points whose MFI was significantly higher or lower than the specified grade were removed. According to operators those points were likely measurement errors.

Another treatment regarded the evaluation of grade differences, to check if the green polymer rheological properties are different from regular polymer. The operators expected that the green polymer would be different, but process knowledge indicated that they should not be, as the only difference was the source of ethylene, and the ethylene used for polymerization is a high-purity one. Therefore they are expected to be similar.

Statistical analysis, specifically t-tests and effect size, showed that they are not, which was expected from a chemical engineering perspective as the ethylene used in polymerization is high purity with practically only some occasional ethane as impurity. These tests are better explained in Appendix A.4. The only grade pairs with a statistically relevant t-test, grades 1 and 2; 3 and 4, had low effect size. The basic statistical analysis can be seen in Table 6.1. Note that those values are not the actual MFI but an anonymised transformation.

A mean and delay analysis was done to merge both datasets with diferent sampling times and with a delay with each other. The idea was: fit a simple overfitting resistant non-linear regression model, where the inputs are the data sampled with a delay of L minutes and smoothed by taking a mean of the previous M minutes. This procedure is inspired by linear delay finding algorithms, for example the one implemented in MatLab (MATLAB) system identification

Table 6.1: Grades characteristics.

Grade	Number of samples	MFI mean	MFI standard deviation	Modifier	Green	T-test p-value	effect size
1	9190	0.165	0.012	Propene	No	0.00%	0.22
2	2123	0.169	0.011	Propene	Yes	—	—
3	1240	0.167	0.027	Propene	No	2.04%	0.2
4	71	0.171	0.013	Propene	Yes	—	—
5	83	0.174	0.039	Propene	Yes	—	—
6	932	0.0593	0.011	—	No	—-	—
7	215	0.0326	0.0098	Butene	No	13.01%	0.16
8	74	0.031	0.007	Butene	Yes	—	—-
9	492	0.830	0.077	—	No	80.68%	0.06
10	15	0.825	0.078	—	Yes	—	—-
11	386	0.590	0.052	Butene	No	60.39%	0.21
12	5	0.601	0.047	Butene	Yes	—	—-

toolbox. The result of this analysis is a matrix L $\times$ M. Then the test error was plotted against L and against M. The best delay and mean is chosen as the one having the smaller root mean squared error. Several models were evaluated but a small random forest, max depth of 4 levels and 50 trees, was chosen due to overfitting resistance and non linear behaviour.

An example of this analysis can be seen in Figures 6.3 and 6.4. In those figures the RMSE of the model fitted from the data sampled at each delay and mean windows, resulting in several points for the same time. Ideally a good result is indicated by a low RMSE between higher RMSEs. This analysis can also be visualized in a heatmap, Figure 6.5. In general delay is more significant in the analysis than mean window. The main purpose of the mean window is to smooth the data and to encode information about the whole window, so minute fluctuations do not affect the model.

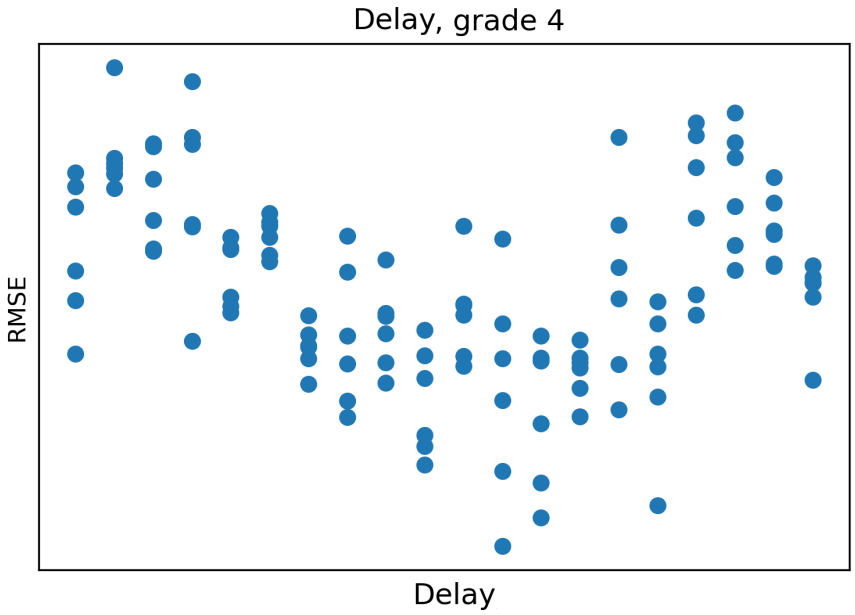


Figure 6.3: Best delay analysis, grade 4.

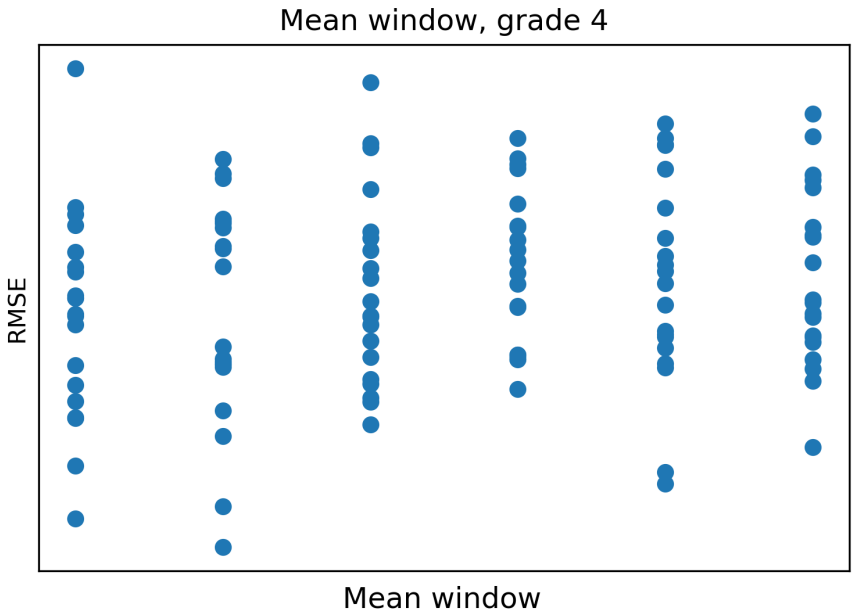


Figure 6.4: Best mean analysis, grade 4.

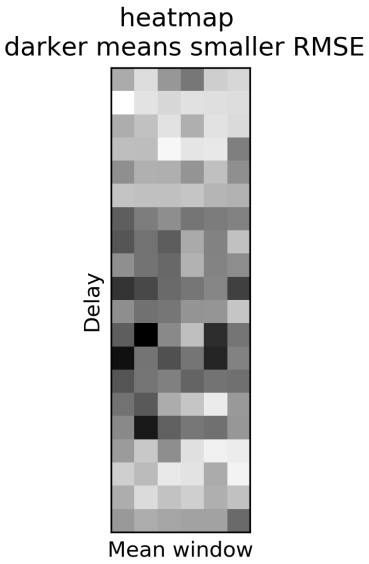


Figure 6.5: Heatmap used in mean and delay analysis.

6.4.2 Modeling

The initial proposal was to use one model for all grades. Unexpectedly, this model worked apparently really well, with R^2 of 92%. However, further examination revealed it learned only the general grade recipe, and did not learn the important smaller scale fluctuations of MFI, as can be seen in Figures 6.6 and 6.7. Therefore, it was decided to separate the grades, and make a model for each grade.

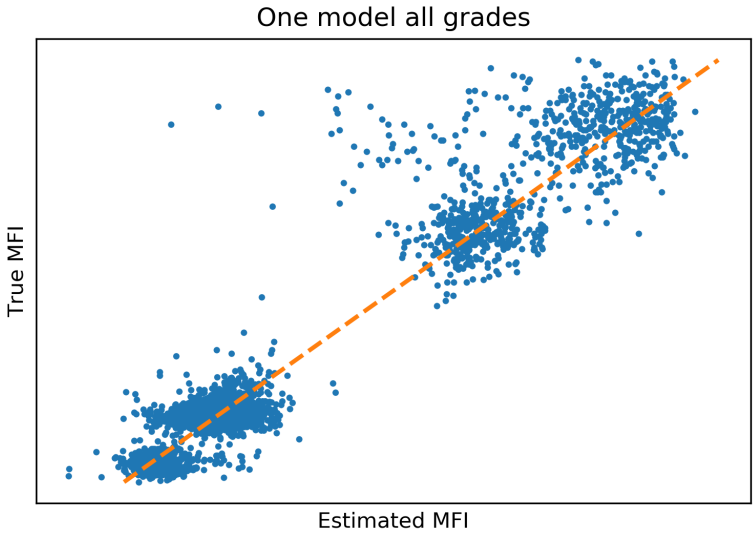


Figure 6.6: Initial model, single model for all grades, scatter plot.

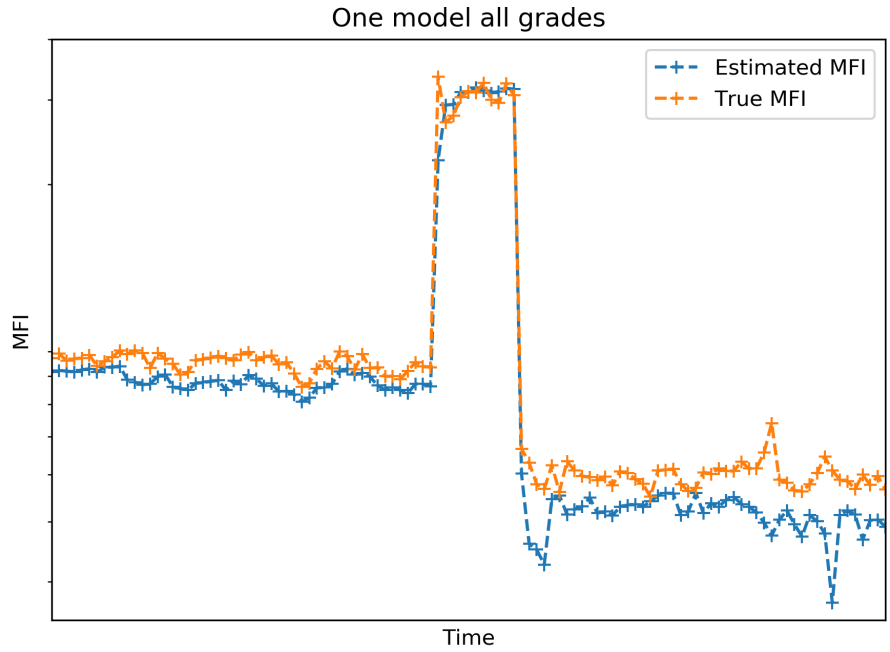


Figure 6.7: Initial model, single model for all grades, vs time.

Due to that, emerged the necessity of a grade classification model, that decides which model to use at any given moment. Most grades have unique operational conditions, which was also the reason why the initial model managed to easily learn each grade mean MFI. This result also meant the mean and delay analysis had to be redone, this time for each grade separately.

An issue created by the separation was data imbalance. Some grades had too few samples to correctly model its behavior. All green grades and their regular correspondent were joined to generate more data, and similar grades were also joined together. These new groups are called family.

The classification model was a decision tree, chosen because it was very interpretable and operational confirmation was desired. The fitted tree was dissected and its rules were similar to what operators used. However the decision tree was not able to separate families 1, 2 and 3, so these families were joined to make a new family. The final family grouping can be seen in Table 6.2.

Table 6.2: Families characteristics.

Family	Number of samples	MFI mean	MFI standard deviation	Modifier
1	12707	0.165	0.01	Propene
2	932	0.0593	0.01	——-
3	289	0.0321	0.009	Butene
4	510	0.831	0.07	——-
5	391	0.590	0.05	Butene

Unsupervised correlation on each family individually analysis showed some collinear features, especially some reactor temperatures. Collinearity can confuse some machine learning models. To remove this redundancy the means between the variables were used, if they were intensive property, or the sum of them if they were extensive properties.

Common feature transformations from chemical engineering domain knowledge were found to be not relevant, like log of concentration, log of MFI or inverse of temperatures. This was because as each variable were kept under a small controlled region, the non linear transformation were not really distinct from linear univariate transformations, so they were reversed during data normalization.

Feature selection was done by random forest feature importance. This method was chosen because the relationship between the variables and MFI is strongly non-linear. It did not revealed any distinctly relevant feature. Although it revealed some features constantly being among the most relevant, like recycle flow and modifier concentration.

A conflict between process knowledge, operators and the data is that the feature selection deemed the reactor pressure irrelevant. A high pressure gas-phase reaction should be influenced by the pressure according to both operators and theoretical knowledge. Analyzing the pressure data, Figure 6.8, it was noticed that the reactor pressure is so important that it is tightly controlled in the plant, with punctual variations only up to 0.2%. Therefore, there is not enough variation for the model to identify the effect of the pressure on the process.

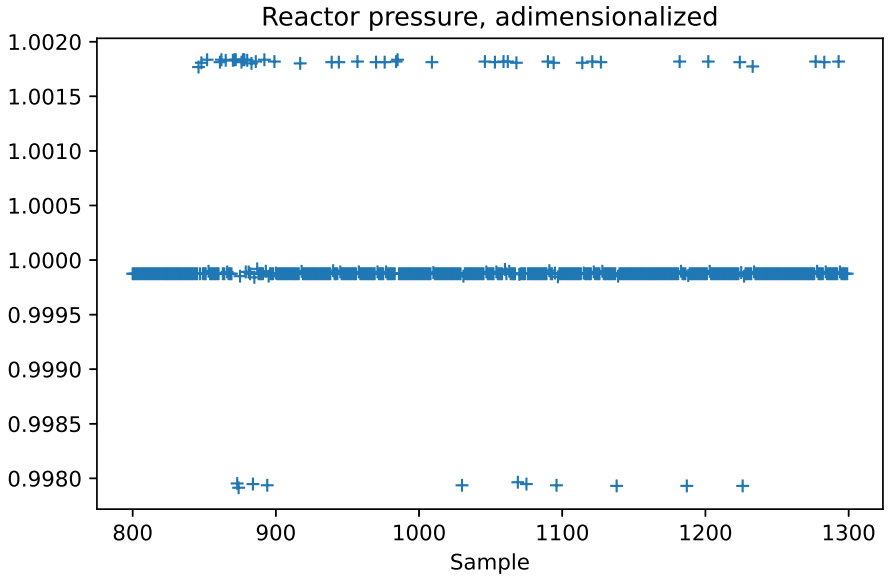


Figure 6.8: Reactor pressure.

Before going back to the regression, metrics need to be discussed. three main metrics were used: R^2, RMSE and "accuracy", all discussed in Appendix A.1. R^2 is the preferred among the statisticians, as it has nice properties, like being interpretable as explained variance and insensitive to linear transformations. Besides it compensates for shifts in variance, that is a common occurrence in the dataset.

The operators preferred RMSE, given by Equation A.3 because it had the same unit as MFI, therefore it was easier to understand. After some discussion it was also suggested by the operators to use "accuracy", Equation A.4, which expresses how often the model agrees with the analysis withing measurement error (ME) . This "accuracy" is used in quotes here because it is not really an accuracy in the statistical sense, as can be seen in Appendix A.2. This metric also was what inspired the trial of support vector regressor, as it aligns with the regression definition of the hinge loss and ignores the points that are already correctly estimated. It should be noticed that as a new metric, a baseline should be constructed to evaluate the results. It was chosen to create a baseline based on a "dumb" model. This dumb model is defined as: a model whose output is just the mean of the train data.

Trend metrics were tested, to see if the models at least agrees with the MFI change direction even if it cannot predict the value directly, like Equation A.5. These metrics were very sensitive to noise due to being a difference based metric, so they were not used. Hence, only R^2 and "accuracy" will be presented.

All implemented models were developed using Scikit-learn, also known as SKLearn (PEDREGOSA *et al.*, 2011). Generally developing their own models is not recommended by industry as it increases billable hours and delays deliveries for an effort that may not translate in a better product. Besides using libraries facilitates documentation and troubleshooting in future developments. However, the importance of studying the documentation and open issues of the chosen library is important and should be stressed for quality assurance.

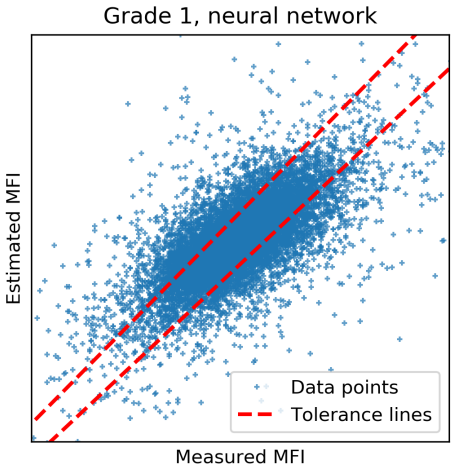
The initial models were neural networks. They were chosen because a continuous non-linear behavior. They worked well in validation, as can be seen in Figure 6.9a, but presented an unexpected behavior in production. Note that in all scatter plots the red lines are the confidence intervals used for the "accuracy" calculation.

Support Vector regressor also showed bad results initially, but it was because SKlearn implementation had a scale parameter that was not clear in documentation. After this was fixed, the SVM had a result similar to neural network. Note in Figure 6.9b a couple of points lay in a line near the middle of the estimated MFI. It is a common behaviour in SVMs with RBF kernel. When outliers appear, even if in only one feature, the kernels tend to zero and the model output tends

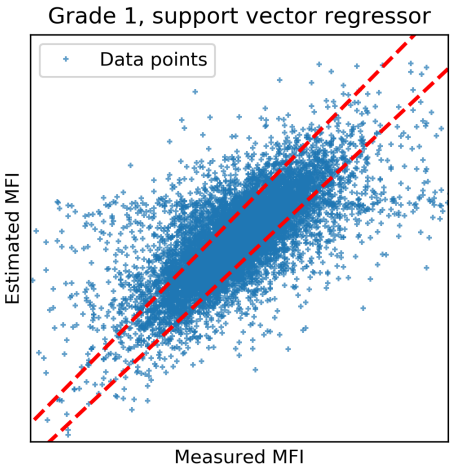
to the mean with a loss of variance.

Gradient boosting forest of decision trees was less inclined than the other models in Figure 6.9c due to a return to the mean behavior. This behavior may lead the model to miss some of the peaks and valleys that can cause product loss.

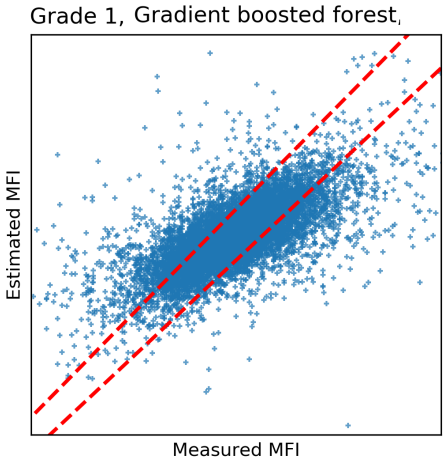
The numerical results for validation can be seen in Tables 6.3 and 6.4. No model was consistently better for every grade, and results varied considerably between grades.



(a) Initial model, grade 1, NN.



(b) Initial model, grade 1, SVM.



(c) Initial model, grade 1, gradient boosted forest.

Figure 6.9: Initial models, grade 1, scatter plots for validation.

Between receiving the data and testing some issues occurred in the plant so sensors started behaving differently from the historical dataset. That can be seen more strongly in Figure 6.10b where the RBF kernels are mostly zeroed and the

Table 6.3: Results, validation, R^2.

Grade	NN	SVM	Gradient boosted forest
All grades	92.5 %	——	———
1	39.7%	33.5%	43.7%
2	1.64%	43.2%	30.2%
3	-407%	41.8%	39.7%
4	-149%	9.15%	11.3%
5	-27.8%	17.9%	17.8%

Table 6.4: Results, validation, "accuracy".

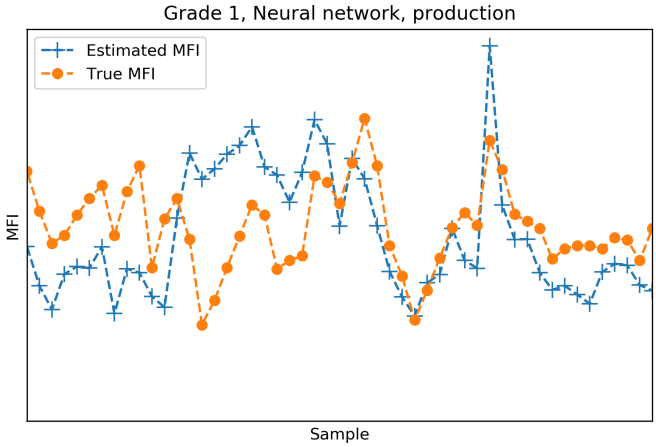
Grade	Baseline "acccuracy"	NN	SVM	Gradient boosted forest
All grades	—	56.7 %	——	———
1	76.55%	39.7%	33.5%	90.1%
2	57.99%	72.2%	77.6%	73.7%
3	62.06%	50.5%	81.7%	73.3%
4	54.60%	46.8%	66.0%	67.0%
5	51.77%	60.0%	62.5%	60.9%

model tends to the mean.

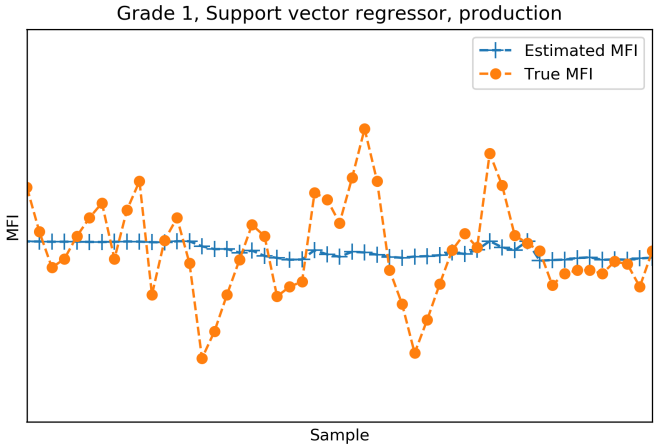
For NN, outliers do not have an expected behaviour unless strongly deviating from the training dataset, so while the model follows the general trend sometimes, it still misses many points and trends, as can be seen in Figure 6.10a.

Gradient boosted forests were the best models in testing. They are more resistant to outliers as no variables interact directly with one another. They also have a more clear behavior regarding outliers. If one variable goes beyond the training range it affects the model as if it were in the extreme limits, no matter how far it goes.

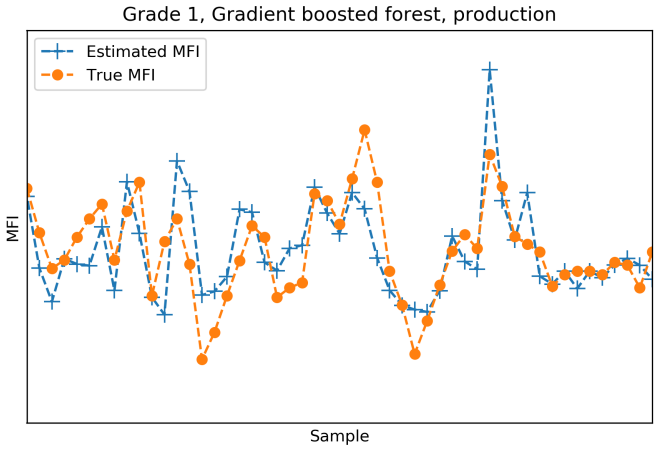
The numerical results for testing can be seen in Tables 6.5 and 6.6. Every metric for every grade was worst than validation. Gradient boosted forests showed the best metrics because it is more resistant to outliers.



(a) Initial model, grade 1, NN.

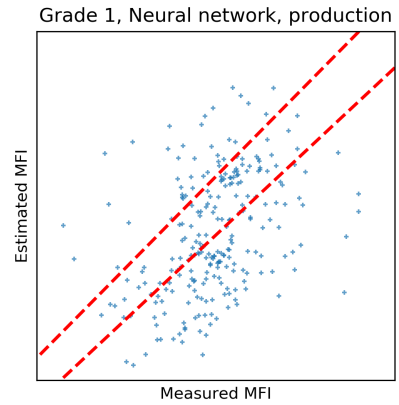


(b) Initial model, grade 1, SVR.

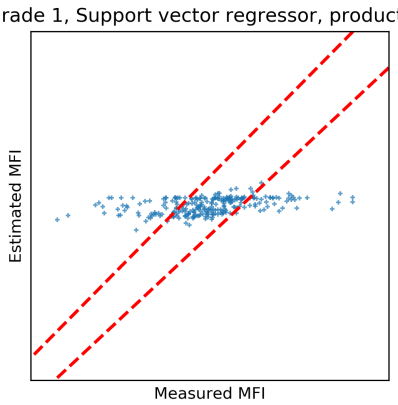


(c) Initial model, grade 1, gradient boosted forest.

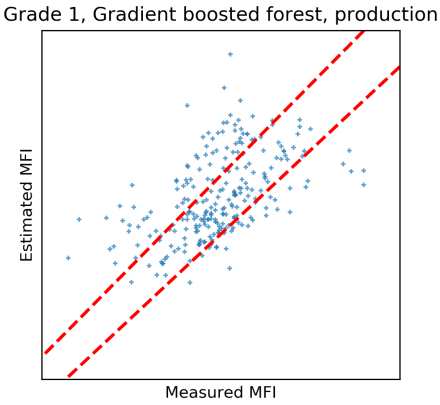
Figure 6.10: True vs estimated MFI over time for testing.



(a) Initial model, grade 1, NN.



(b) Initial model, grade 1, SVR.



(c) Initial model, grade 1, gradient boosted forest.

Figure 6.11: True vs estimated MFI scatter plot for testing.

Table 6.5: Results, test, R^2.

Grade	NN	SVM	Gradient boosted forest
1	-122%	10.5%	23.5%
2	-177%	-1.31%	1.40%
3	-8.93%	-7.12%	8.30%
4	-487%	-1.76%	-0.869%
5	-399%	-3.04%	15.8%

Table 6.6: Results, test, "accuracy".

Grade	Baseline "accuracy"	NN	SVM	Gradient boosted forest
1	69.28%	41.8%	72.9%	74.9%
2	30.85%	24.4%	49.6%	53.4%
3	58.62%	74.6%	72.9%	74.5%
4	67.50%	40.0%	64.0%	62.7%
5	86.03%	38.6%	63.5%	68.2%

Finally, The model chosen to represent the process was Gradient boosted

forests. All further tests and refinements was done with Gradient boosted forests only.

6.4.3 Second iteration

After discussing the results with the stakeholders, they were convinced to allow the addition of the extrusion measurements to the model, creating another model for better predictive capabilities. Additionally, with one model chosen, a hyperparameter optimization could be performed, since it demands more computational time. The results can be seen in Table 6.7 and Figures 6.12 and 6.13.

Table 6.7: Second iteration results.

	Reactor+extruder model		Refined reactor Model	
Grade	Accuracy	R2	Accuracy	R2
1	94.94%	57.91%	92.45%	37.66%
2	86.39%	49.37%	84.57%	36.09%
3	86.90%	50.33%	82.76%	36.86%
4	82.65%	36.48%	83.16%	15.20%
5	80.51%	33.88%	77.95%	19.36%

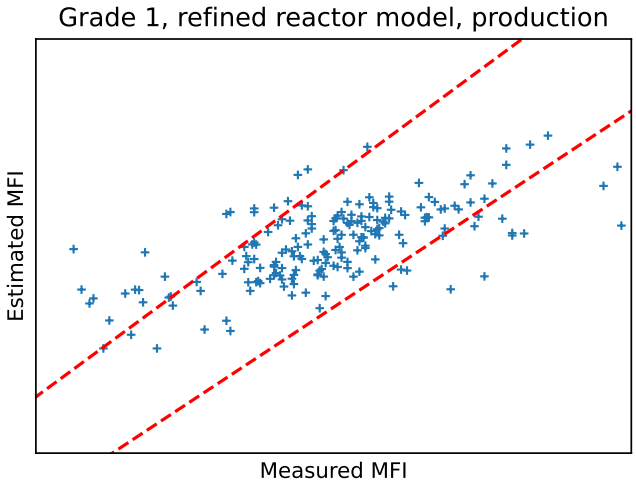


Figure 6.12: Refined reactor model.

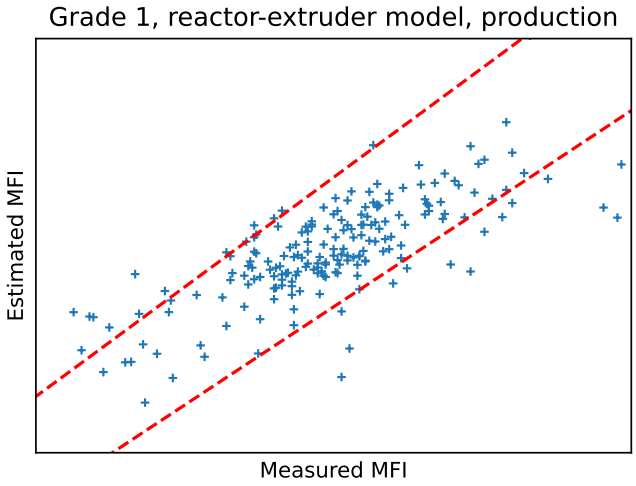


Figure 6.13: Refined reactor-extruder model.

6.4.4 Model analysis

A partial dependence plot was built to see if the model follow the expected behaviour. As can be seen in Figure 6.14, the model projects the expected behaviour for extruder current. The higher the extruder current, the lower the MFI is, since lower MFI makes it harder for the polymer to flow through the extruder, therefore requiring a higher current.

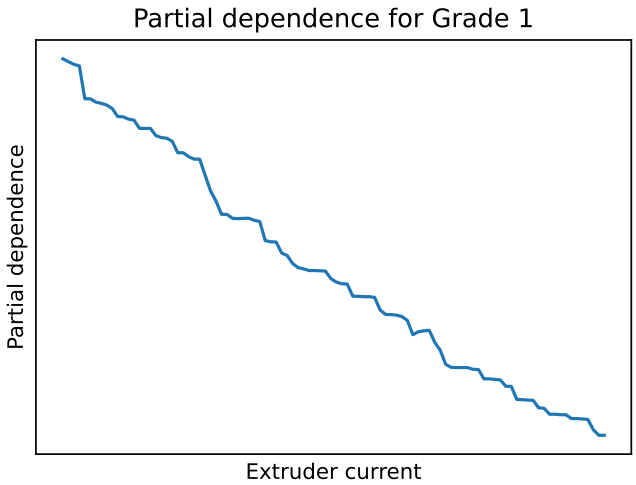


Figure 6.14: Partial dependence of extruder current.

Another thing that should be analyzed is how the operators will see the model output, since the MFI data was sampled at around every 2 hours but sensor data is sampled every minute. The model output per minute was plotted along with the True MFI, Figure 6.15. It can be seen that there are fluctuations

between MFI samples, and sometimes the model even predicts an MFI increase before it happens. This agrees with process knowledge, as the reactor dynamics have a residence time smaller than the MFI sample time, and creates a suggestion of a higher MFI testing frequency.

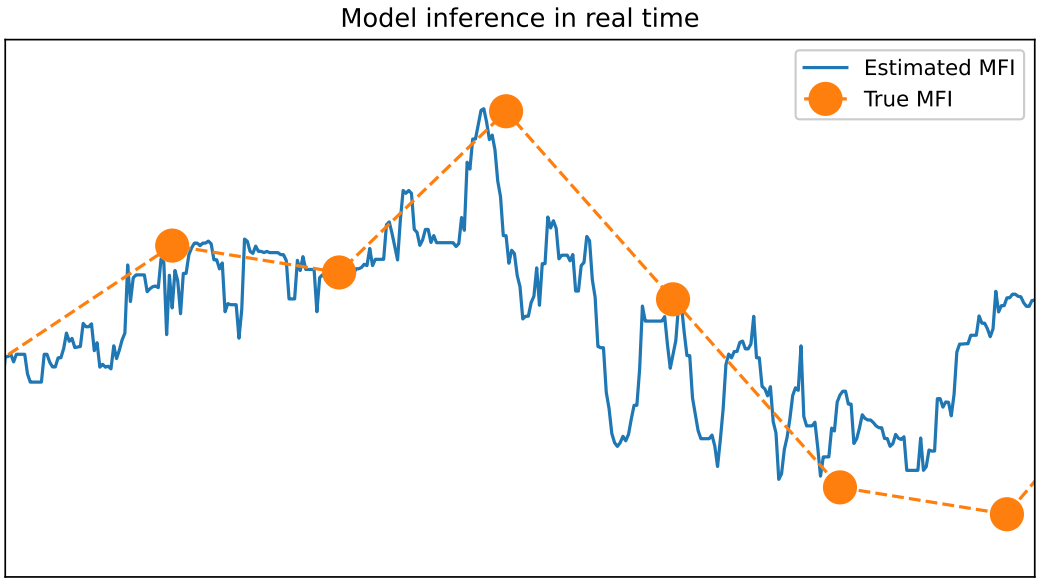


Figure 6.15: Model behavior in real time data.

6.5 Summary on how the proposed methodology was used

The methodology was useful to solve conflicts between engineering knowledge, data analysis and expert knowledge, enhancing the trust operators had on the models. It also allowed a successful persuasion of the stakeholders to change the project specifications, allowing for the development a more accurate Reactor+extruder model. The creation of custom metrics taking account the inherent uncertainty of the measurements made the improvements and capabilities of the models more pronounced.

Demonstrating how the model would behave in response to changes in important variables showed the model will behave as expected based on engineering knowledge, reducing concerns about incorrect inferences caused by extrapolation and outliers. It also helped perceive potential limitations of the model and dataset, proposing solutions for them.

Chapter 7

Dry gas seal system fault detection

7.1 Problem description

Seals are used to reduce loss of process gas on compressors and other turbomachinery. Dry gas seals were created due to the need of simpler and safer seals than wet seal systems that were commonly used (STANLEY, 2002).

There are many dry gas seal configurations. The most common are tandem, but single seals are also used. Single seals consist of a rotating ring and a stationary ring. A seal gas, typically higher pressure process gas or nitrogen, is fed to the seal. This gas is used to keep the rings from contacting each other (FORSTHOFFER, 2017).

The process in this case study is an offshore CO_2 compressor for gas reinjection. It raises the pressure of the gas produced in the platform to a high pressure and reinjects the gas in the offshore oil well, increasing production and reducing greenhouse effect gas emission. There can be several issues in the gas seal, which leads to process gas loss. In this case, resulting in a more dangerous and ecologically harmful operation. The objective of this work is to find a way to warn operators about the faults, if possible before they take place. Additionally it should identify which fault is happening and which sensors the operators should look at to notice those faults.

7.2 Literature review

Here the operation of the dry gas seal is described. During operation part of the seal gas flow into the process through a labyrinth and part flows into the seal. The geometry of the rotating ring generates a lifting force creating a gap between the two rings while seal gas is flowing. The gap is around a 3-5 µm, and is monitored through axial displacement sensors. Springs help keeping

the stationary ring in place and improve response to motor axial movement (FORSTHOFFER, 2017).

The seal gas is then vented through the primary vent, usually to a flare. If a tandem (double) seal is used, a separation gas, usually nitrogen or filtered air, is vented into the secondary seal. While in the first seal part of the gas flows into the process, part of the separation gas flows to the primary vent and part to the secondary vent. From the secondary vent the separation gas either goes to the flare or to the atmosphere. Figure 7.1 is a simplified drawing of a single seal configuration. The seal works with very tight gaps between the components, so both gases must be filtered and dried before entering the system(FORSTHOFFER, 2017).

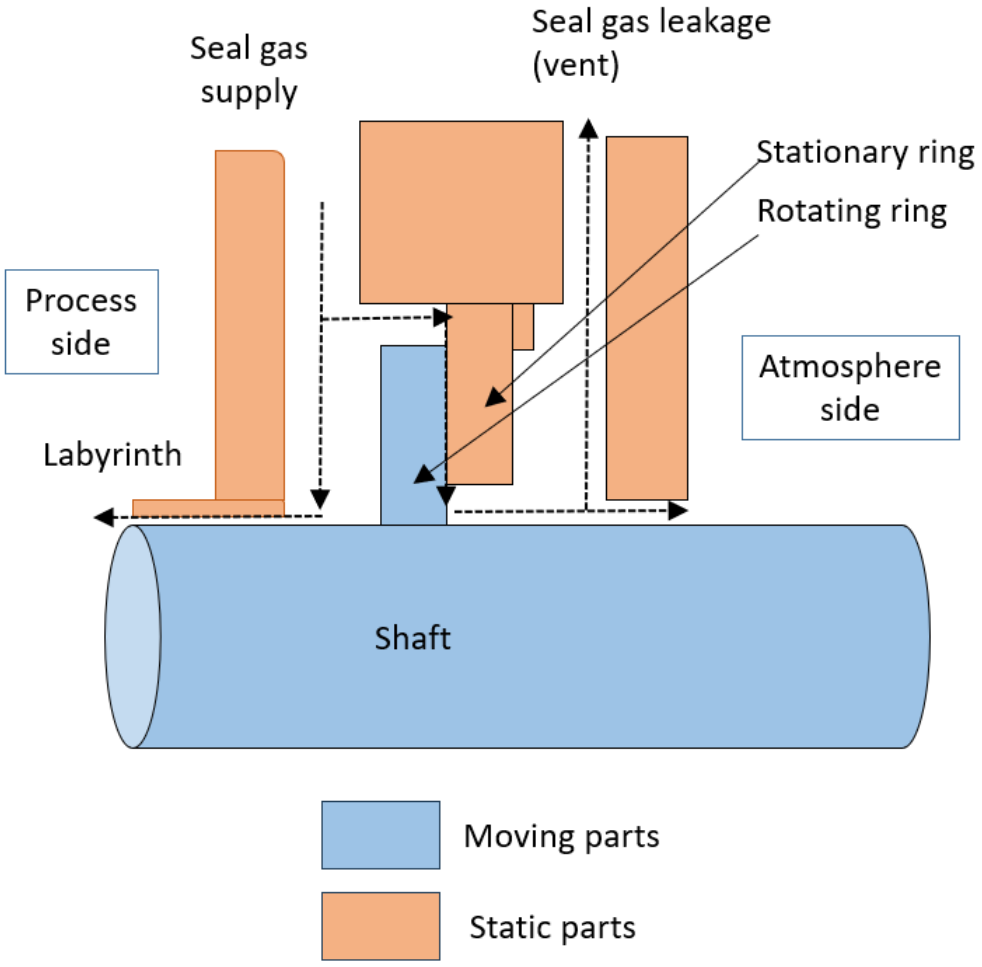


Figure 7.1: Single seal configuration.

Minimizing process gas leakage is the purpose of seals. Even though some gas leaks by design, the gap is small compared to most sealing technology so the leakage rate should be reasonably small. When the compressor is not operating the springs make the two rings contact and bring the leakage to almost

zero. Measuring leakage rate is difficult due to dependency on several hard to measure variables like gas properties and seal physical properties, so differential pressure is used as a proxy of the leakage rate.

This type of seal is preferred by the petrochemical industry as it works well at high pressures and high speeds due to being non-contacting.

There can be many problems with its operation, mainly when something other than gas enters the seal. DAY and ALLISON (2016) classified the faults in 4 types: supply contamination, process contamination, lube oil contamination and geometry/installation problems.

Supply contamination happens when something other than seal gas enters the seal. It can be caused by seal gas condensation, filter failure or build-up, among other causes. Seal gas condensation happens due to the Joule-Thompson effect and seal gas supply must be heated to avoid condensation. Filter failure may cause particles to pass to the seal, and it can be detected by the filter differential pressure (DAY and ALLISON, 2016).

Process contamination is when solids from the gas being compressed enters the seal. It usually happens when seal gas supply flow is insufficient. It is common in sour gas processes, as sulphur builds up in the seal faces (DAY and ALLISON, 2016).

Lube oil contamination occurs when oil from the bearings enter the seal. It can happen when lube oil flow is too high or separation gas pressure is too low. DAY and ALLISON (2016) recommend operating on the upper limit of the separation gas pressure to avoid lube oil contamination.

Geometry/installation problems happen during start-up and are quickly detected. They usually happen when the seal is not installed with all the necessary components like bolts or labyrinth snap rings. Another common cause is too long drive pin, which restrict axial movement.

Other non-fatal faults can happen in the process like unbalanced supply flow between the drive and non drive ends of the compressor, high separation gas flow, and compressed gas leakage into the seal system. While these faults are not critical they should be fixed to avoid further problems.

The most important sensors for dry seal gas system control and monitoring are vent differential pressure, seal and separation gas supply flow, leakage pressure to the vent, vibration and axial displacement measurements. The differential pressures are used to estimate gas flow, and are substantially noisy. Vibration and axial displacement measurements come in pairs, and each pair are expected to be correlated.

As far as the author is aware, there are no published industrial data-driven studies for dry gas seal system fault detection and diagnosis. DAY and ALLI-

SON (2016) collected and analyzed reports from seal faults given by companies, but did not analyze any sensors nor tried to do any predictive work. TOWSYFYAN *et al.* (2018) used acoustic emission to investigate faults in an experimental dry gas seal. They generated faults in the dry gas seal and evaluated if acoustic emission could differentiate between healthy and faulty regimes. They achieved satisfactory result and a clear visual separation between fault and normal states, but did not report any numerical metric. FORSTHOFFER (2017) reports some industrial faults on dry gas seals, along with their causes and corrective measures undertook by operators, but did not statistically analyze the data from the sensors.

The compression system in this case study injects CO_2 in an offshore oil well. It has two compressor trains running in parallel, while one operates the other is kept on reserve. Each compressor train has 2 stages, a high and a low pressure stage. Between each compressor there are a heat exchanger and a knock-out drum to remove any condensed water.

The seals are in the compressor drive end and the non drive end of each stage. There are a total of 4 seals. It should be noted that the compressor system never really stops, but passes through some standby periods for maintenance. Full stop may lead to permanent damage to the equipment, as the shaft bends over its own weight.

The data was collected from a historian and sampled every minute. Some of the compressor faster dynamics may be lost, but it is not expected to influence the model efficiency, as the operators would not be able to counter faster faults while they occur anyway. The sensors include compressor sensors like outlet gas pressure and temperature and knocked out drum level; and gas supply flows, seal pressure and differential pressure at the vent. There were also some sensors from the flare, like a CO_2 concentration analyzer. The datasets contained a total of 50 sensors and around 300,000 samples.

7.3 Materials and methods

A total of six datasets were received, each concerning one fault of the gas seal system. From these 6 sets, 4 datasets belonged to train B and 2 belonged to train A. Some of the datasets were overlapping, meaning that more than one type of fault happened at the same time. The occurrence of simultaneous faults is common in industrial processes, as one issue may lead to or trigger another.

It was not clear when exactly the faults started, as all datasets start at midnight, so it was acquired extra 3 days of operation before the previously assumed midnight. Even then it was not clear when the fault began, especially the time

of the day. Still, this data was labeled as prefault and detecting this prefault was tested.

The data is described in Tables 7.1 and 7.2. Events 3 and 4 overlap with each other. This dataset is balanced between normal and fault conditions. This is not usual for fault detection processes. Usually normal condition eclipse faulty conditions, like in the 3W dataset. This is expected as plants should spend more time working fine than with a fault. Normal condition operation data was chosen by operators.

Table 7.1: Fault events.

Event	Train	Fault	Fault starts at day	Fault ends at day
1	A	Seal gas unbalance	0	14
2	A	Seal gas unbalance	617	632
3	B	Seal gas unbalance	97	115
4	B	High vent diff. pressure	110	125
5	B	Seal gas unbalance	193	208
6	B	Seal gas unbalance	345	360

Table 7.2: Samples per train.

Train	Normal	Fault	Prefault
A	47159	43098	6863
B	71157	82883	12684

One limitation of the system developed for train A was lack of fault diversity. There is only one type of fault in the dataset, so there is a good chance of the model only learning one fault and ignoring other problems. This is also a risk at train B, while not that high as there are two types of fault. Also, as there is only one type of fault in train A there will be no diagnosis analysis.

A potential solution is outlier detection, where the model learns the normal condition and says whether the process is normal or not. Not normal does not indicate fault, as it may correspond to a new operation condition or a test run, but indicates a region of the data operators may take a look at. This procedure was later suggested to the stakeholders, and becomes the second iteration of the methodology.

The datasets were joined according to the compression trains, creating one consolidated dataset for each train. Redundant data points were excluded, along with points with bad values, with strings instead of numbers. Moments with constant sensor values were also removed, as they are caused by communication failure between platform and server.

The first analysis was visual inspection. It showed a huge amount of outliers. Those outliers affected most variables at the same time and were recurrent,

indicating an actual physical disturbance on the process and not on instrumentation. After discussion with operators it was discovered that these outliers corresponded to standby periods. Standby periods are when the compressor is slowed down for maintenance and check-ups. They are easy to identify as the compressor's output pressure goes down below a threshold. After this clarification the standby periods were removed from the dataset.

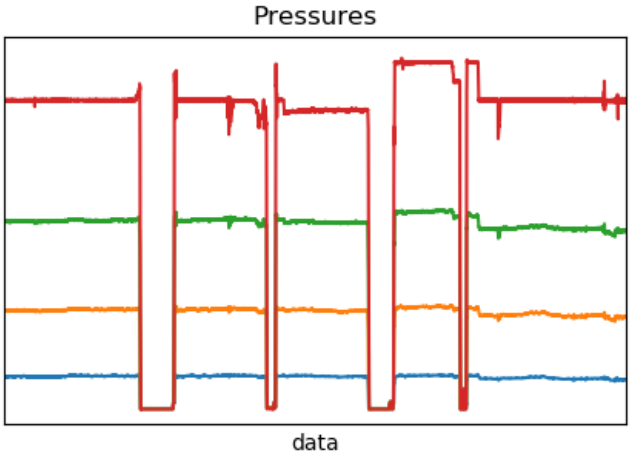


Figure 7.2: Outlier example.

Unsupervised analysis showed some expected correlations: the vibration sensors are very correlated between vertical and horizontal measurements, so are the axial displacement sensors. It is good to confirm expected results, it can reveal flaws on the data, like mislabeled tags or improperly installed equipment. Some miscalibrated sensors were found, reported and removed. The operators later issued a maintenance request for those sensors.

Disregarding domain knowledge, neither vibration nor vibration variance showed any relevant information about the faults. It was also expected that flare CO_2 concentration would be important, as it would indicate process gas leakage, but it did not appear in any feature importance method. The knockout drums level alarms are theoretically important, as liquid entering the seal is a source of problems. However, the measurements were too noisy to produce any insight and filtering them did not improve the signal, therefore they were removed.

For process monitoring a good property a model should have is continuous output, and it was explicitly requested by the stakeholders. Classification is discrete, but seeing the model progression over time allows operators to see the process status in real time instead of waiting for the result to change. The two main continuous outputs are probability or distance, depending on the model

used. Probability is restricted to [0,1] while distance has no restriction (in practice regularization keeps the values from going too far from [-1, 1]). Models that output probability can be transformed to a distance output using the logit transform, given by Equation 7.1.

$$logit(p) = log(\frac{p}{1-p}) \tag{7.1}$$

The metrics used here were accuracy and ROC-AUC, both described in Appendix A.2.

7.4 Results

For feature relevance a SVM with L1 regularization was done. It was chosen because seal operation transitions were fast and well separated, as can be seen in Figure 7.3, indicating a margin method would be a good fit. Besides, linear SVM is easily interpretable and the distance of the hyperplane may be understood as a "health measurement", helping to warn operator whether the system is close to a fault. L1 regularization was chosen for feature selection. The model is trained in a one-vs-rest scheme.

These characteristics would also suggest a tree based model, but one characteristic desired is continuous output, to evaluate if the model is approaching a fault, and if so, which sensors are showing it.

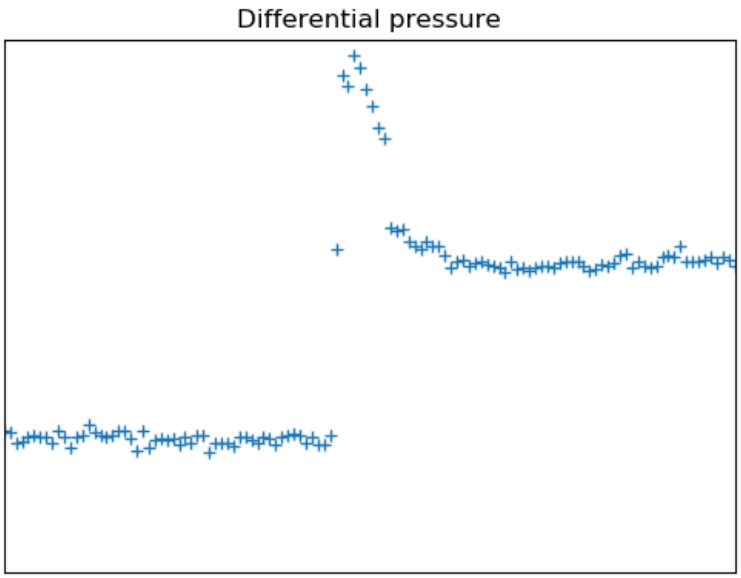


Figure 7.3: Differential pressure fast transition.

Ideally an event would have been separated to use as test and validation, but

there are few events on the dataset, so validation was done using blocks with 2 days of data. One issue on fault detection and diagnosis testing is that a fault might take months to happen, and it is hard to say if the model works on real time.

Validation produced good results in train A for fault/normal detection, as showed by Figure 7.4. However it was not able to detect the difference between fault and pre-fault. This indicates that fault prevention may not be an useful application of this model.

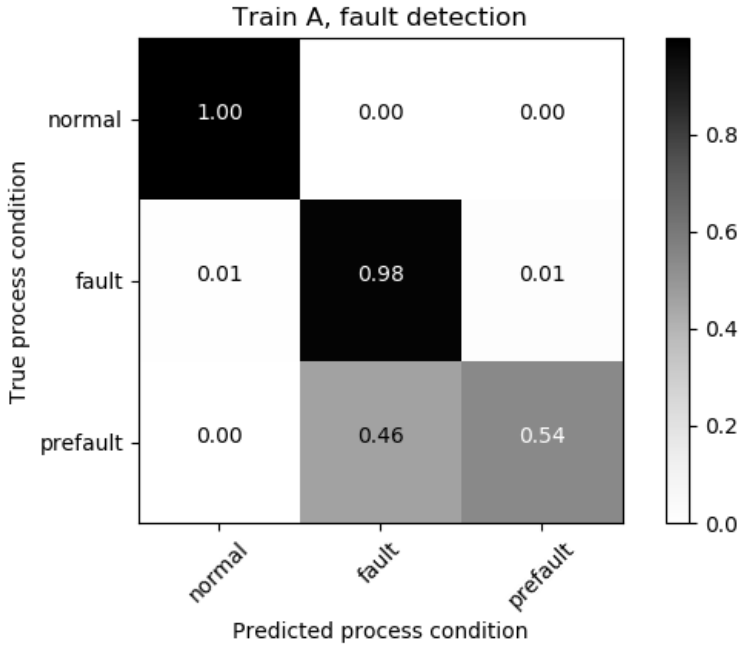


Figure 7.4: Fault detection results, train A.

The SVM with L1 regularization managed to remove most variables, keeping 8 relevant variables. Mostly one seal gas supply flowrate and leakage pressure. This conflicts with some domain knowledge, the other supply flowrate should also be important. This account on how L1 regularization deals with collinearity, by removing one of the collinear features, in this case the other supply flowrate.

For train B the results were similar as showed by Figure 7.5. However the system experienced more difficulty detecting prefault. In this case L1 regularization did not remove most variables, keeping around 30 features. This can be due to more fault variety leading to more information necessary to the model properly classify the faults.

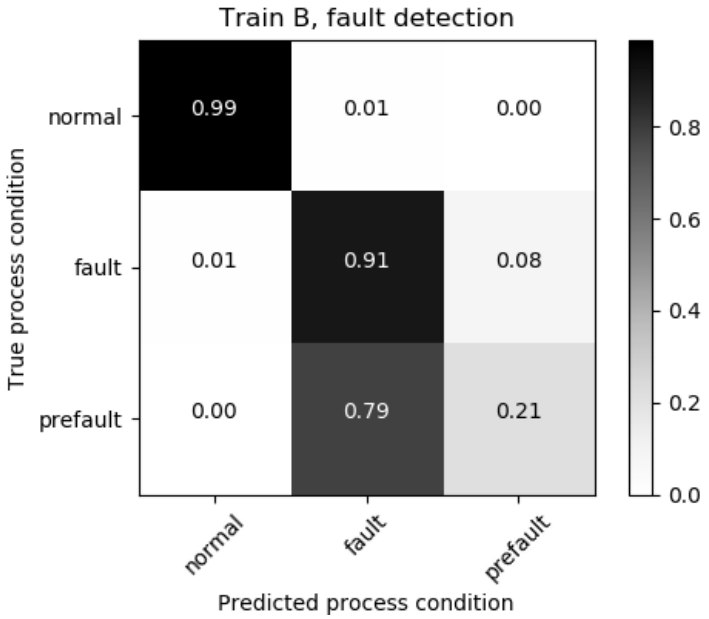


Figure 7.5: Fault detection results, train B.

For train B fault diagnosis, to compensate for the moments when the fault overlaps a new fault class were created, called "both". Again the model was not able to differentiate prefault from the faults. However, the model was able to differentiate between faults with a good degree of accuracy, even in the case of the both class, as can be seen in Figure 7.6. Overall accuracy was 86.4%

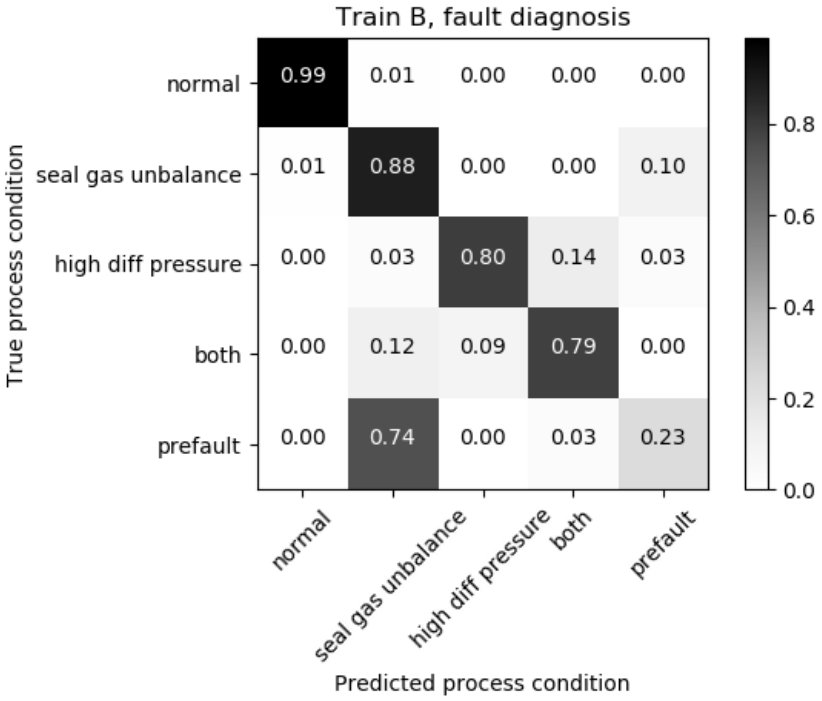


Figure 7.6: Fault diagnosis results, train B.

Visualizing the real time model output was one of the desired properties of

Table 7.3: Fault classification results.

		ROC-AUC		F1 score	
Train	Accuracy	Normal	Fault	Normal	Fault
A	95.9%	99.9%	97.1%	99.3%	95.7%
B	88.8%	99.9%	94.7%	99.0%	89.1%

the model. To do so the hyperplane distance from the normal-vs-rest model is plotted against time. Distance from the hyperplane is adimensional, but it can be interpreted as a "health" measurement. In Figure 7.7, between both periods there is a standby moment and later the compressor returns, after a couple of hours the model output crosses the zero line and the model starts classifying it as normal operation

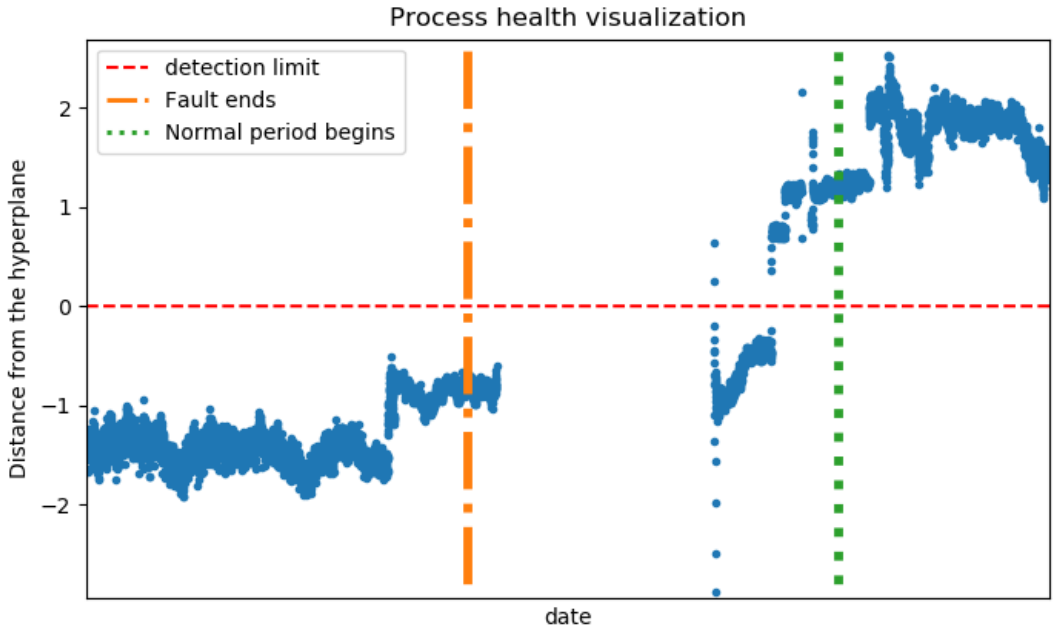


Figure 7.7: SVM output for fault detection, train B.

7.5 Second iteration

Over the development of the first iteration, as a better insight of the process and dataset was achieved, an unexpected demand became clear: the current dataset alone was insufficient to train a suitable monitoring system, as the existing faults in the dataset and normal periods were insufficient to gather a proper understanding of the process, as they likely did not explore a good part of the possible feature space. A form of detecting new faults became necessary. Therefore, the second iteration focus on dataset construction, normal/fault labeling

and interpretable abnormal event detection and diagnosis.

7.5.1 Interpretable abnormal event detection

A feature of the desired abnormal event detection is to inform which sensors should be investigated exactly. The compression system is complex, it is difficult for an operator to look at all 50 possible variables at the same time. Therefore, the algorithm used is detection by Mahalanobis distance. Mahalanobis distance is a multivariable extension of the z-score normalization. It allows to evaluate not only if the process leaves a current known "normal behavior region", but which sensors lead it to leave this region. It is explained more in-depth in Appendix A.5.

A central point in the Mahalanobis distance calculation is the covariance matrix estimation. Covariance estimation through the usual sample estimation approach is susceptible to outliers, which is undesirable in an abnormal event detection task. There are several approaches for robust covariance matrix estimation, based in either sample selection approaches, that is, removing samples there are likely to be an outlier following some metric; or shrinkage, which is a form of regularization that makes the smallest and biggest eigenvalues of the covariance matrix closer to each other, making it better conditioned.

The algorithm used for covariance estimation was Minimum Covariance Determinant (MCD) (ROUSSEEUW and DRIESSEN, 1999), better explained in Appendix A.6. Visual inspection was used for outlier detection, with a percentile based approach for automation of the process and to facilitate identification. Given the nature of MCD it will always describe some points as highly above a certain percentile, so the detection process can not be fully automatized, so visual inspection is the better approach. An example of detection of abnormal event is Figure 7.8. Note that the y-axis is in log scale. This method of abnormal event detection is not robust to changes in operational points, which are common in this process.

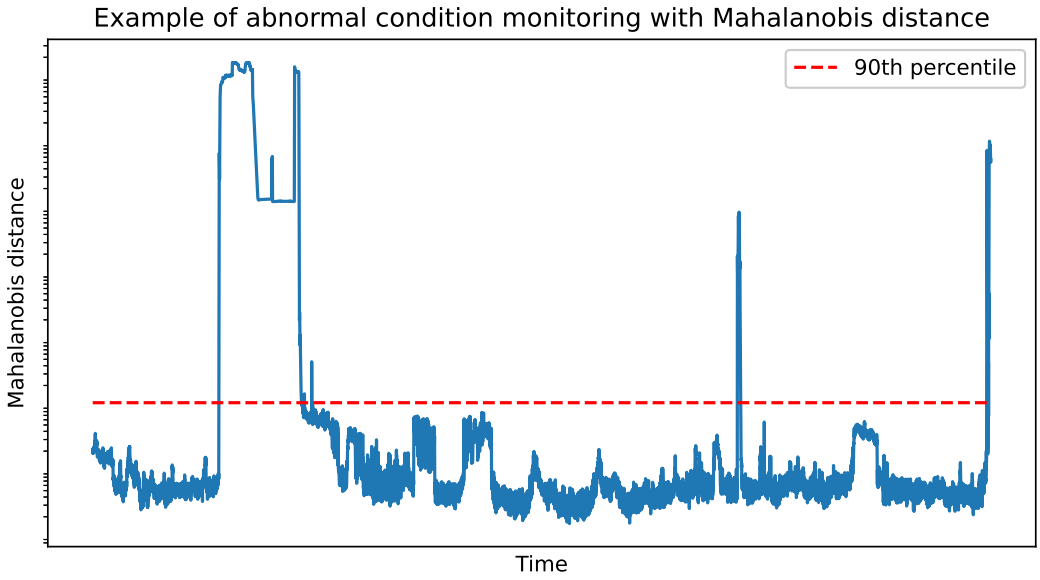


Figure 7.8: Outlier detection with Mahalanobis distance.

For outlier diagnosis an example is given in Figure 7.9. it can be seen that sensor 2 presents little deviation, sensor 1 presents some deviation and sensor 3 presents a strong deviation. So it would be reported for the operator to look for sensor 3.

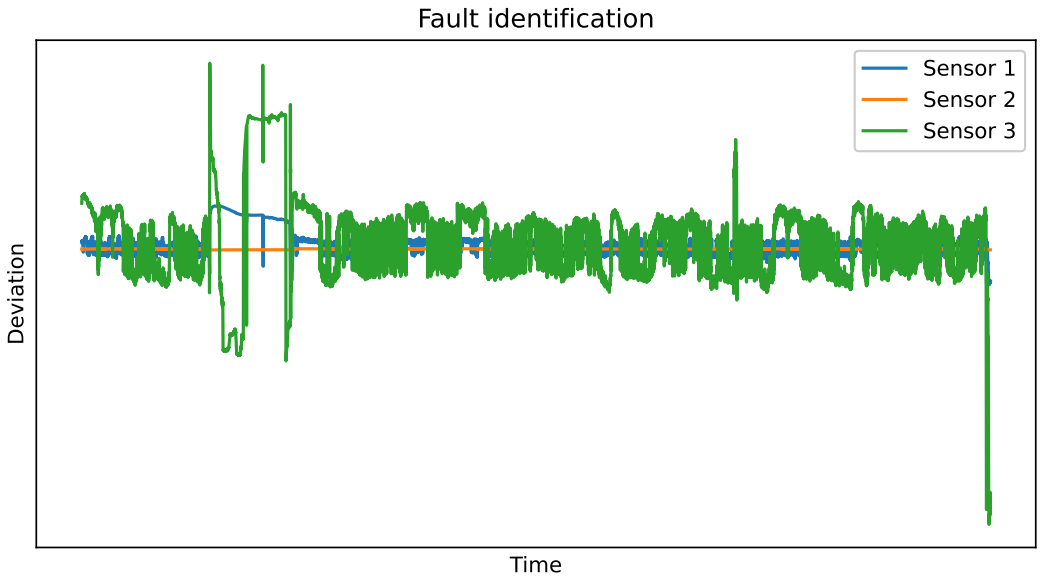


Figure 7.9: Outlier diagnosis with Mahalanobis distance.

7.5.2 Dataset labeling

To make a compilation of the operational points it was used a clustering method: Gaussian mixture model. This model fits the data to several clusters,

modeled as Gaussian distributions with individual mean and variance. There is a similarity to the Mahalanobis distance, and it was chosen to keep both techniques' results consistent. In theory it could also be used for abnormal condition detection, but underwhelming experimental results coupled with a lack of interpretability made the Mahalanobis distance preferable.

The clustering was also used to reduce the workload of the operators looking for a fault. If the outlier was classified as a previously known normal cluster with high likelihood they would be disregarded for inspection, since they are likely just the same operational point. If they were a previously classified fault cluster with high likelihood, the points would be classified as that fault, with the operators informing if the decision was correct. Figure 7.10 is an example of process monitoring using the clustering as an extra visual layer. If cluster 2 was a previously known example, they would be classified accordingly to the previous designation.

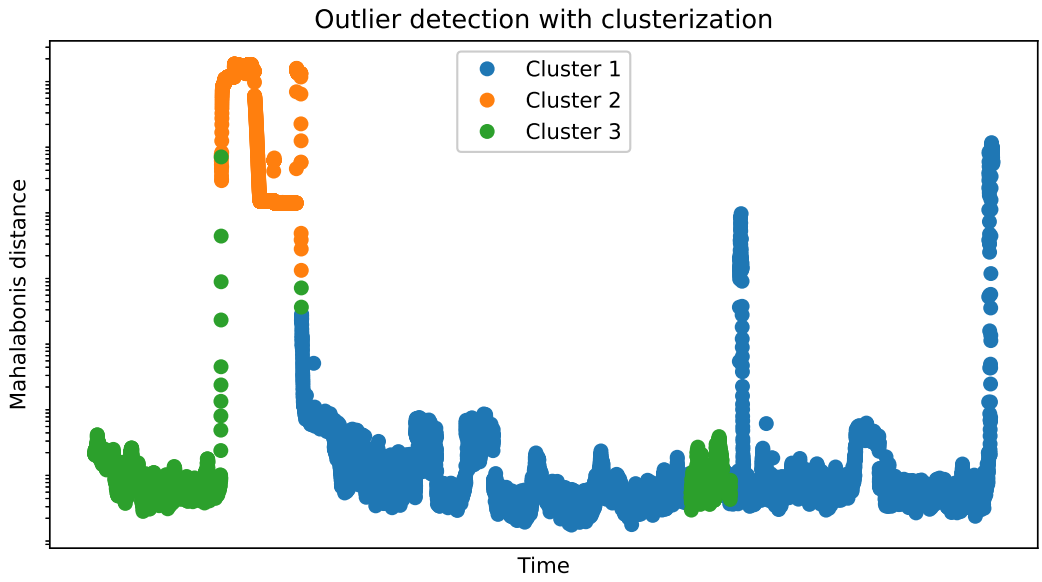


Figure 7.10: Outlier detection with clustering.

All the described monitoring received operator feedback. The process results were summarized in a weekly report and operators looked at the process to see if it was at fault or not. Later those points were classified as normal or a fault. A new fault was discovered during this process, along with new instances of high vent differential pressure fault.

7.6 Model analysis

In theory, the log likelihood given by the Gaussian Mixture could be used for outlier detection. A low log likelihood mean that the sample is far from any cluster, as seen in Figure 7.11. However the issue is the identified outliers usually only had a punctual decrease and soon returned to regular log-likelihood values.

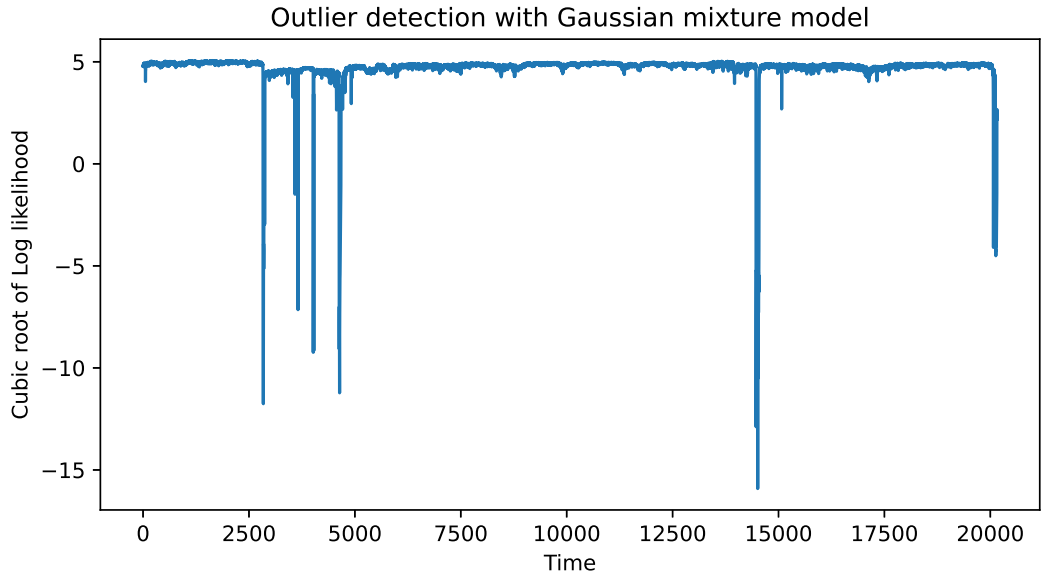


Figure 7.11: Outlier detection with Gaussian mixture models.

Another analysis performed was to see the clusters transitions over time. If the Gaussian mixture has too many clusters it may attribute 2 or more clusters to the same operational condition. Visually, it can be seen as the model constantly switching between 2 or more, as can be seen in Figure 7.12. When this happen the model is retrained with less clusters, using the previous model as an initial guess, with one of the clusters being alternated removed.

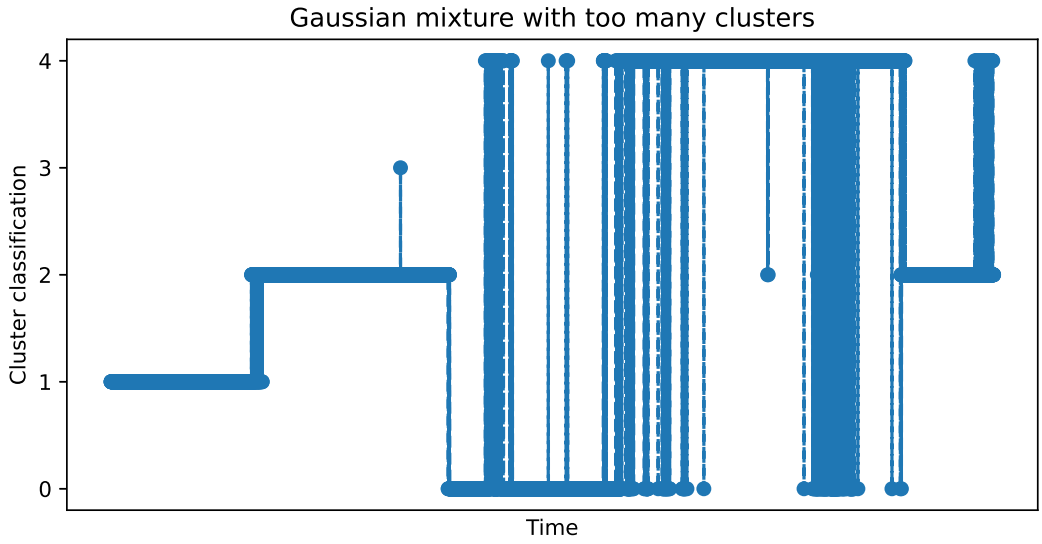


Figure 7.12: Example of improperly tuned clustering.

7.7 Summary on how the proposed methodology was used

The methodology summarized in Figure 3.1 was used to improve the dataset initially given through the "Get more data" loop. Prefault samples were extracted and concatenated to the dataset as the need for fault detection before the actual fault happening is necessary for a good fault detection system. The methodology was also used to better visualize how the model behaves during the fault transitions.

Finally, the proposed methodology was used to critically assess the dataset and reevaluate the capabilities of a tool that could be build with it. The whole task was reconsidered given the dataset characteristics and the demands passed from the operators through consistent meetings and feedback, resulting in a new data collection, labeling and classification process. The current framework is installed in the industry partner servers. Human interaction and process knowledge were essential for building a solution that meets the needs of the stakeholders involved in this project.

Chapter 8

Conclusions

The purpose of this work is to propose an iterative methodology for applying machine learning in chemical process, with constant refining and reevaluation of the development processes, guided by data analysis and process knowledge for improved results and acceptability. In order to develop and demonstrate this methodology, four cases studies were used: one control problem with simulated data, one using the public 3W dataset and two using private datasets given by industry partners. In this thesis it was demonstrated how to apply this methodology and how it improves the results and acceptability of the solutions developed.

In the first case study, control of a gas lift oil well, issues of observability, data generation and process control were explored. The data generation was weaved into an already existing NMPC methodology, smoothly integrating machine learning with traditional modeling and control. Control with NN as the state estimator was better than control with an EKF as state estimator for all gas injection setpoint tasks. It had more difficulty with oil production setpoint tasks. However, it managed to stave off slugging during PI increase while the EKF control did not.

The methodology was used to align process knowledge with the data analysis, allowing for an effective feature selection that was used in both the internal state estimation and the process control. It also helped to pursue a substantial control improvement through the "Get more data" loop. The new data, generated using a different and faster state estimation technique, used more sensor information, therefore being a more information rich data.

In the second case study, oil well fault detection with the 3W dataset, a comprehensive exploratory data analysis was done to gather a better understanding of the dataset, help setup the train, validation and test split, and to develop a cursory data selection. Novelty feature engineering techniques were developed to increase the dataset dimensionality. These extra features were evalu-

ated both with feature selection techniques and in terms of how logically they could be used to explain the faults. Further model analysis showed how they would work on a real-time application. The methodology generated substantial improvements in 5 of the 7 models, with median improvements of 44.23% for F1-score and 13.25% for the accuracy. The results were an improvement over previously found results on the literature.

The methodology encompassed the generation of new features while assessing their value through both modeling analysis and process knowledge analysis. The selected features enabled the incorporation of temporal information into the dataset. Furthermore, explicating the rationale behind feature selection as the methodology recommends would serve to enhance the model adoption within industrial projects that leverage these models.

In the third case study, MFI estimation from online sensor data, insightful data analysis was used to clear conflicts between operators, process knowledge and statistical analysis, increasing the number of samples in each grade and making a more explainable feature selection. Meetings with the operators helped develop new metrics better suited for their needs. Overall, the operators were satisfied with the results, that achieved agreement with the lab analysis within the measurement error up to 92.45% of the time. The models are currently running on their plants and helping guiding the operator's decisions.

The methodology was instrumental in resolving discrepancies among engineering, data analysis, and expert perspectives, thereby improving operators' confidence in the model. Additionally, it facilitated the persuasion of stakeholders to modify project specifications, leading to the creation of the Extruder+reactor model. By designing tailored metrics, the capabilities of the models became more evident. Model analysis presenting the model's responsiveness to alterations in relevant variables demonstrated its alignment with established engineering knowledge, easing doubts stemming from inputing new never-seen-before data. The methodology also aided in identifying potential model and dataset constraints, offering potential solutions to them.

Finally, in the fault detection problem in the gas seal system, a successful fault detection application was developed, including a health measurement for the process. This initial solution achieved up to 95.9% accuracy. However an in-depth analysis showed that the current dataset was insufficient to ensure a proper machine learning model, so a complete review of the task was performed. A framework for dataset building and annotation was developed, centered in explainable abnormal event detection and using clusterization, to help facilitate the assignment. The explainable abnormal event detection helped operators to find regions where there may have been a fault, and which sensors to look at

to evaluate if the period was at fault or not. Clusterization was used to catalog different normal and sometimes faulty conditions.

The methodology was applied to enhance the original dataset by incorporating prefault samples. It was further utilized to offer insights into the model's behavior during fault transitions. Moreover, the methodology played an important role in subjecting the dataset to a more rigorous evaluation, leading to a reassessment of the feasible applications of models generated using the dataset. The entire task was reevaluated in light of dataset attributes and operator requirements, culminating in the development of a novel framework for data collection, labeling, and classification. Human interaction and process expertise played an essential role in devising a solution that properly addressed the needs of both operators and engineers.

Overall, the methodology successfully guided the application of machine learning in the case studies and managed to improve the results and the products generated, especially increasing the acceptability of the processes' operators.

The author expects this work will help future chemical engineers going into the data science field to improve their projects. There are still some aspects of machine learning development that this methodology could explore, like model update methodology to deal with process drift. Model update is a tough subject to approach, even in traditional modeling frameworks. Processes, feedstock and product specifications change all the time, and a method to adapt the model without having to redo the whole identification project is desired by the industry.

References

AL-MALAH, K., 2016, *Aspen Plus: Chemical Engineering Applications*. ISBN: 978-1-119-13123-6.

ANDERSSON, J. A. E., GILLIS, J., HORN, G., et al., 2019, "CasADi – A software framework for nonlinear optimization and optimal control", *Mathematical Programming Computation*, v. 11, n. 1, pp. 1–36. doi: <10.1007/s12532-018-0139-4>.

BABALOLA, A., OBUBU, M., 2019, "Modelling Colinearity in the Presence of Non-Normal Error: A Robust Regression Approach", *Annals of Biostatistics and Biometric Applications*, v. 2 (07), pp. 1–9. doi: <10.33552/ABBA.2019.02.000549>.

BATISTA, G. E., MONARD, M. C., OTHERS, 2002, "A Study of K-Nearest Neighbour as an Imputation Method." *HIS*, v. 87, n. 251-260, pp. 48.

BAUGHMAN, D. R., LIU, Y. A., 1995, *Neural Networks in Bioprocessing and Chemical Engineering: With Disk*. 1st ed. Orlando, FL, USA, Academic Press, Inc. ISBN: 0120830302.

BELYADI, H., HAGHIGHAT, A., 2021, *Machine Learning Guide for Oil and Gas Using Python*. Gulf Professional Publishing. ISBN: 978-0-12-821929-4. Availability: <https://www.sciencedirect.com/science/article/pii/B9780128219294000019>.

BERGSTRA, J., BREULEUX, O., BASTIEN, F., et al., 2010, "Theano: a CPU and GPU math expression compiler". In: *Proceedings of the Python for scientific computing conference (SciPy)*, v. 4. Austin, TX.

BIKMUKHAMETOV, T., JÄSCHKE, J., 2020, "Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models", *Computers and Chemical Engineering*, v. 138, pp. 106834. doi: <https://doi.org/10.1016/j.compchemeng.2020.106834>. Availability: <https://www.sciencedirect.com/science/article/pii/S0098135419313675>.

BISHOP, C. M., 1995, "Training with Noise is Equivalent to Tikhonov Regularization", *Neural Comput.*, v. 7, n. 1 (jan.), pp. 108–116. doi: <10.1162/neco.1995.7.1.108>. Availability: <http://dx.doi.org/10.1162/neco.1995.7.1.108>.

BOSER, B. E., GUYON, I. M., VAPNIK, V. N., 1992, "A Training Algorithm for Optimal Margin Classifiers". In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pp. 144–152, New York, NY, USA. ACM. ISBN: 0-89791-497-X. Availability: <http://doi.acm.org/10.1145/130385.130401>.

BREIMAN, L., 2001, "Random Forests", *Machine Learning*, v. 45, n. 1 (Oct), pp. 5–32. doi: <10.1023/A:1010933404324>. Availability: <https://doi.org/10.1023/A:1010933404324>.

CARVALHO, B. G., VAZ VARGAS, R. E., SALGADO, R. M., et al., 2021, "Flow Instability Detection in Offshore Oil Wells with Multivariate Time Series Machine Learning Classifiers". In: *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pp. 1–6.

CHAPMAN, P., CLINTON, J., KERBER, R., et al., 2000, "CRISP-DM 1.0 Step-by-step data mining guide", (August). Availability: <https://maestria-datamining-2010.googlecode.com/svn-history/r282/trunk/dmct-teorica/tp1/CRISPWP-0800.pdf>.

CHIANG, L., RUSSELL, E., BRAATZ, R., 2001, *Fault Detection and Diagnosis in Industrial Systems*, v. 12.

DAY, M., ALLISON, T., 2016, "Analysis of Historical Dry Gas Seal Failure Data". p. V009T24A015, 06.

DIAS, A. C. S. R., ROJAS SOARES, F., JÄSCHKE, J., et al., 2019, "Extracting Valuable Information from Big Data for Machine Learning Control: An Application for a Gas Lift Process", *Processes*, v. 7, n. 5. doi: <10.3390/pr7050252>. Availability: <https://www.mdpi.com/2227-9717/7/5/252>.

DIEHL, F. C., ALMEIDA, C. S., ANZAI, T. K., et al., 2018, "Oil production increase in unstable gas lift systems through nonlinear model predictive control", *Journal of Process Control*, v. 69, pp. 58–69. doi: <https://doi.org/10.1016/j.jprocont.2018.07.009>. Availability: <https://www.sciencedirect.com/science/article/pii/S0959152418301380>.

DIEHL, F. C., MACHADO, T. O., ANZAI, T. K., et al., 2019, "10 increase in oil production through a field applied APC in a Petrobras ultra-deepwater well", *Control Engineering Practice*, v. 91, pp. 104108. doi: <https://doi.org/10.1016/j.conengprac.2019.104108>. Availability: <https://www.sciencedirect.com/science/article/pii/S0967066119301273>.

DING, Y., ABEYKOON, C., PERERA, Y. S., 2022, "The effects of extrusion parameters and blend composition on the mechanical, rheological and thermal properties of LDPE/PS/PMMA ternary polymer blends", *Advances in Industrial and Manufacturing Engineering*, v. 4, pp. 100067. doi: <https://doi.org/10.1016/j.aime.2021.100067>. Availability: <https://www.sciencedirect.com/science/article/pii/S2666912921000374>.

DO NASCIMENTO, R. S. F., BARBOSA, B. H. G., VARGAS, R. E. V., et al., 2021, "Detecção de anomalias em poços de petróleo surgentes com stacked autoencoders". In: *Simpósio Brasileiro de Automação Inteligente-SBAI*, v. 1.

EIKREM, G. O., IMSLAND, L., FOSS, B., 2004, "Stabilization of Gas Lifted Wells Based on State Estimation", *IFAC Proceedings Volumes*, v. 37 (01). doi: <10.1016/S1474-6670(17)38752-9>.

FARSANG, B., BALOGH, I., NÉMETH, S. Z., et al., 2015, "PCA Based Data Reconciliation in Soft Sensor Development - Application for Melt Flow Index Estimation". .

FAYYAD, U., PIATETSKY-SHAPIRO, G., SMYTH, P., 1996, "From data mining to knowledge discovery in databases", *AI magazine*, v. 17, n. 3, pp. 37.

FEIL, B., ABONYI, J., PACH, P., et al., 2004, "Semi-mechanistic Models for State-Estimation – Soft Sensor for Polymer Melt Index Prediction". In: Rutkowski, L., Siekmann, J. H., Tadeusiewicz, R., et al. (Eds.), *Artificial Intelligence and Soft Computing - ICAISC 2004*, pp. 1111–1117, Berlin, Heidelberg. Springer Berlin Heidelberg. ISBN: 978-3-540-24844-6.

FLETCHER, R., 1987, *Practical Methods of Optimization*. Second ed. New York, NY, USA, John Wiley & Sons.

FORSTHOFFER, M. S., 2017, "Chapter 9 - Dry Gas Seals". In: Forsthoffer, M. S. (Ed.), *Forsthoffer's More Best Practices for Rotating Equipment*, Butterworth-Heinemann, pp. 433 – 458. ISBN: 978-0-12-809277-4.

Availability: <http://www.sciencedirect.com/science/article/pii/B9780128092774000097>.

FORTUNA, L., GRAZIANI, S., RIZZO, A., et al., 2006, *Soft Sensors for Monitoring and Control of Industrial Processes (Advances in Industrial Control)*. Berlin, Heidelberg, Springer-Verlag. ISBN: 1846284791.

GEREVINI, G., FARENZENA, M., TRIERWEILER, J., 2018, "Slugging attenuation using Nonlinear Model Predictive Control in offshore oil production", *Journal of Petroleum Science and Engineering*, v. 165 (02). doi: <10.1016/j.petrol.2018.01.054>.

HAALAND, S. E., 1983, "Simple and Explicit Formulas for the Friction Factor in Turbulent Pipe Flow", *Journal of Fluids Engineering*, v. 105, n. 1, pp. 89–90. doi: <10.1115/1.3240948>. Availability: <http://dx.doi.org/10.1115/1.3240948>.

HAM, J. Y., RHEE, H.-K., 1996, "Modeling and control of an LDPE autoclave reactor", *Journal of Process Control*, v. 6, n. 4, pp. 241–246. doi: <https://doi.org/10.1016/0959-1524(95)00052-6>. Availability: <https://www.sciencedirect.com/science/article/pii/0959152495000526>.

HAN, H., GUO, X., YU, H., 2016, "Variable selection using Mean Decrease Accuracy and Mean Decrease Gini based on Random Forest". pp. 219–224, 08.

HARROU, F., SUN, Y., HERING, A. S., et al., 2021, *Statistical Process Monitoring Using Advanced Data-Driven and Deep Learning Approaches*. Elsevier. ISBN: 978-0-12-819365-5. Availability: <https://www.sciencedirect.com/science/article/pii/B9780128193655000073>.

HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J., 2001, *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA, Springer New York Inc.

HO LEE, E., KIM, T., KOO YEO, Y., 2008, "Prediction and quality control of the melt index during production of high-density polyethylene", *Korean Journal of Chemical Engineering*, v. 25 (07), pp. 613–622. doi: <10.1007/s11814-008-0103-5>.

Hutter, F., Kotthoff, L., Vanschoren, J. (Eds.), 2018, *Automatic Machine Learning: Methods, Systems, Challenges*. Springer. In press, available at http://automl.org/book.

ISO 1133-1:2011, 2012, *Plastics — Determination of the melt mass-flow rate (MFR) and melt volume-flow rate (MVR) of thermoplastics — Part 1: Standard method*. Standard, International Organization for Standardization, Geneva, CH, Dex.

JAHANSHAHI, E., SKOGESTAD, S., HANSEN, H., 2012, "Control structure design for stabilizing unstable gas-lift oil wells", *IFAC Proceedings Volumes*, v. 45, n. 15, pp. 93 – 100. doi: <https://doi.org/10.3182/20120710-4-SG-2026.00110>. Availability: <http://www.sciencedirect.com/science/article/pii/S1474667016304256>. 8th IFAC Symposium on Advanced Control of Chemical Processes.

JAMES, G., WITTEN, D., HASTIE, T., et al., 2014, *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated. ISBN: 1461471370, 9781461471370.

JOE QIN, S., GUO, S., LI, Z., et al., 2021, "Integration of process knowledge and statistical learning for the Dow data challenge problem", *Computers and Chemical Engineering*, v. 153, pp. 107451. doi: <https://doi.org/10.1016/j.compchemeng.2021.107451>. Availability: <https://www.sciencedirect.com/science/article/pii/S0098135421002295>.

JOHNSSON, T., 1992, "A procedure for stepwise regression analysis", *Statistical Papers*, v. 33, n. 1 (Dec), pp. 21–29. doi: <10.1007/BF02925308>. Availability: <https://doi.org/10.1007/BF02925308>.

JOLLIFFE, I., CADIMA, J., 2016, "Principal component analysis: A review and recent developments", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, v. 374 (04), pp. 20150202. doi: <10.1098/rsta.2015.0202>.

JORDANOU, J., ANTONELO, E., CAMPONOGARA, E., 2019, "Online learning control with Echo State Networks of an oil production platform", *Engineering Applications of Artificial Intelligence*, v. 85 (10), pp. 214–228. doi: <10.1016/j.engappai.2019.06.011>.

JORDANOU, J. P., CAMPONOGARA, E., ANTONELO, E. A., et al., 2018, "Nonlinear Model Predictive Control of an Oil Well with Echo State Networks", *IFAC-PapersOnLine*, v. 51, n. 8, pp. 13 – 18. doi: <https://doi.org/10.1016/j.ifacol.2018.06.348>. Availability: <http://www.sciencedirect.com/science/article/pii/S2405896318306785>. 3rd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production OOGP 2018.

JUI HSIEH, C., WEI CHANG, K., JEN LIN, C., et al., 2008, "A dual coordinate descent method for large-scale linear svm". In: *ICML '08: Proceedings of the 25th international conference on Machine learning, 408–415*. ACM.

JULIER, S., UHLMANN, J., 2004, "Unscented filtering and nonlinear estimation", *Proceedings of the IEEE*, v. 92, n. 3, pp. 401–422. doi: <10.1109/JPROC.2003.823141>.

KINGMA, D. P., BA, J., 2014, "Adam: A Method for Stochastic Optimization", Availability: <http://arxiv.org/abs/1412.6980>. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

KRISHNAMOORTHY, D., FOSS, B., SKOGESTAD, S., 2016, "Real-Time Optimization under Uncertainty Applied to a Gas Lifted Well Network", *Processes*, v. 4, n. 4. doi: <10.3390/pr4040052>. Availability: <https://www.mdpi.com/2227-9717/4/4/52>.

KURAMOTO, A. S. R., DAVYS CARVALHO MELO DE OLIVEIRA, A., LOPES TORRES, P. H., et al., 2021. "METHOD OF ASSESSMENT OF THE QUALITY OF THE BURN OF THE GASES IN THE FLARE AND ADJUSTMENT TO THE VAPOR FLOW RATE IN A CONTINUOUS AND CONSTANT WAY". Availability: <https://ppubs.uspto.gov/pubwebapp/>.

LIU, D. C., NOCEDAL, J., 1989, "On the limited memory BFGS method for large scale optimization", *Mathematical Programming*, v. 45, pp. 503–528.

LIU, Y., LIANG, Y., GAO, Z., 2017, "Industrial polyethylene melt index prediction using ensemble manifold learning-based local model", *Journal of Applied Polymer Science*, v. 134 (03). doi: <10.1002/app.45094>.

LJUNG, L., 1986, *System Identification: Theory for the User*. USA, Prentice-Hall, Inc. ISBN: 0138816409.

MARINS, M. A., BARROS, B. D., SANTOS, I. H., et al., 2021, "Fault detection and classification in oil wells and production/service lines using random forest", *Journal of Petroleum Science and Engineering*, v. 197, pp. 107879. doi: <https://doi.org/10.1016/j.petrol.2020.107879>. Availability: <https://www.sciencedirect.com/science/article/pii/S0920410520309372>.

MATLAB, 2012. "MATLAB". The MathWorks, Natick, MA, USA.

MCKINNEY, W., 2010, "Data Structures for Statistical Computing in Python". In: van der Walt, S., Millman, J. (Eds.), *Proceedings of the 9th Python in Science Conference*, pp. 51 – 56.

MELO, A., CÂMARA, M. M., CLAVIJO, N., et al., 2022, "Open benchmarks for assessment of process monitoring and fault diagnosis techniques: A review and critical analysis", *Computers and Chemical Engineering*, v. 165, pp. 107964. doi: <https://doi.org/10.1016/j.compchemeng.2022.107964>. Availability: <https://www.sciencedirect.com/science/article/pii/S0098135422003003>.

MITCHELL, T. M., 1997, *Machine learning*, v. 1. McGraw-hill New York.

MOEN, O., LIEN, M., LYNGMO, L. B., et al., 1999. "Process for the production of polyolefins in an autoclave reactor". out. 12. US Patent 5,965,674.

MOHRI, M., ROSTAMIZADEH, A., TALWALKAR, A., 2012, *Foundations of Machine Learning*. The MIT Press. ISBN: 026201825X, 9780262018258.

Montavon, G., Orr, G. B., Müller, K. (Eds.), 2012, *Neural Networks: Tricks of the Trade - Second Edition*, v. 7700, *Lecture Notes in Computer Science*. Springer. ISBN: 978-3-642-35288-1. Availability: <https://doi.org/10.1007/978-3-642-35289-8>.

MUIR, B. M., MORAY, N., 1996, "Trust in automation. Part II. Experimental studies of trust and human intervention in a process control simulation", *Ergonomics*, v. 39, n. 3, pp. 429–460. doi: <10.1080/00140139608964474>. Availability: <https://doi.org/10.1080/00140139608964474>. PMID: 8849495.

MURPHY, K. P., 2013, *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.], MIT Press. ISBN: 9780262018029 0262018020.

NELDER, J. A., MEAD, R., 1965, "A Simplex Method for Function Minimization", *Computer Journal*, v. 7, pp. 308–313.

NOR, N. M., HASSAN, C. R. C., HUSSAIN, M. A., 2020, "A review of data-driven fault detection and diagnosis methods: applications in chemical process systems", *Reviews in Chemical Engineering*, v. 36, n. 4, pp. 513–553. doi: <doi:10.1515/revce-2017-0069>. Availability: <https://doi.org/10.1515/revce-2017-0069>.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., et al., 2011, "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, v. 12, pp. 2825–2830.

PEIXOTO, A. J., PEREIRA-DIAS, D., XAUD, A. F., et al., 2015, "Modelling and Extremum Seeking Control of Gas Lifted Oil Wells", *IFAC-PapersOnLine*, v. 48, n. 6, pp. 21 – 26. doi: <https://doi.org/10.1016/j.ifacol.2015.08.004>. Availability: <http://www.sciencedirect.com/science/article/pii/S240589631500871X>. 2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production OOGP 2015.

PENG, D.-Y., ROBINSON, D. B., 1976, "A New Two-Constant Equation of State", *Industrial and Engineering Chemistry Fundamentals*, v. 15, n. 1, pp. 59–64. doi: <10.1021/i160057a011>.

PERRY, R. H., GREEN, D. W., MALONEY, J., 1997, *Perry's chemical engineers' handbook (ed.)*, v. 1. Seventh, international edition ed. New York, NY, USA.

PLATT, J. C., 1998, *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. Relatório técnico, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING.

QIN, S., CHIANG, L., 2019, "Advances and Opportunities in Machine Learning for Process Data Analytics", *Computers Chemical Engineering*, v. 126 (04). doi: <10.1016/j.compchemeng.2019.04.003>.

RAILEANU, L. E., STOFFEL, K., 2004, "Theoretical Comparison Between the Gini Index and Information Gain Criteria", *Annals of Mathematics and Artificial Intelligence*, v. 41, n. 1 (may), pp. 77–93. doi: <10.1023/B:AMAI.0000018580.96245.c6>. Availability: <https://doi.org/10.1023/B:AMAI.0000018580.96245.c6>.

RAWLINGS, J., MAYNE, D., DIEHL, M., 2017, *Model Predictive Control: Theory, Computation, and Design*, v. 1. Madison, Wisconsin, USA, Nob Hill Publishing, LLC.

RIBEIRO, C., MIYOSHI, S., SECCHI, A., et al., 2016, "Model Predictive Control with quality requirements on petroleum production platforms", *Journal of Petroleum Science and Engineering*, v. 137, pp. 10 – 21.

doi: <https://doi.org/10.1016/j.petrol.2015.11.004>. Availability: <http://www.sciencedirect.com/science/article/pii/S0920410515301704>.

ROBBINS, H., MONRO, S., 1951, "A stochastic approximation method", *Annals of Mathematical Statistics*, v. 22, pp. 400–407.

ROJAS SOARES, F. D., SECCHI, A. R., BEZERRA DE SOUZA, M. J., 2022, "Development of a Nonlinear Model Predictive Control for Stabilization of a Gas-Lift Oil Well", *Industrial and Engineering Chemistry Research*, v. 61, n. 24, pp. 8411–8421. doi: <10.1021/acs.iecr.1c04728>. Availability: <https://doi.org/10.1021/acs.iecr.1c04728>.

ROKACH, L., MAIMON, O., 2014, *Data Mining With Decision Trees: Theory and Applications*. 2nd ed. River Edge, NJ, USA, World Scientific Publishing Co., Inc. ISBN: 9789814590075, 981459007X.

ROSSUM, G., 1995, *Python Reference Manual*. Relatório técnico, Amsterdam, The Netherlands, The Netherlands.

ROUSSEEUW, P., DRIESSEN, K., 1999, "A Fast Algorithm for the Minimum Covariance Determinant Estimator", *Technometrics*, v. 41 (08), pp. 212–223. doi: <10.1080/00401706.1999.10485670>.

SALAU, N. P., TRIERWEILER, J. O., SECCHI, A. R., et al., 2009, "A New Process Noise Covariance Matrix Tuning Algorithm for Kalman Based State Estimators", *IFAC Proceedings Volumes*, v. 42, n. 11, pp. 572–577. doi: <https://doi.org/10.3182/20090712-4-TR-2008.00092>. Availability: <https://www.sciencedirect.com/science/article/pii/S1474667015303359>. 7th IFAC Symposium on Advanced Control of Chemical Processes.

SANSANA, J., JOSWIAK, M. N., CASTILLO, I., et al., 2021, "Recent trends on hybrid modeling for Industry 4.0", *Computers and Chemical Engineering*, v. 151, pp. 107365. doi: <https://doi.org/10.1016/j.compchemeng.2021.107365>. Availability: <https://www.sciencedirect.com/science/article/pii/S0098135421001435>.

SAUTER, D., TAOUFIK, M., BOISSON, C., 2017, "Polyolefins, a Success Story", *Polymers*, v. 9 (06). doi: <10.3390/polym9060185>.

SAVITZKY, A., GOLAY, M. J. E., 1964, "Smoothing and Differentiation of Data by Simplified Least Squares Procedures." *Analytical Chemistry*, v. 36,

n. 8, pp. 1627–1639. doi: <10.1021/ac60214a047>. Availability: <https://doi.org/10.1021/ac60214a047>.

SAWILOWSKY, S., 2009, "New Effect Size Rules of Thumb", *Journal of Modern Applied Statistical Methods*, v. 8 (11), pp. 597–599. doi: <10.22237/jmasm/1257035100>.

SCHWEIDTMANN, A. M., ESCHE, E., FISCHER, A., et al., 2021, "Machine Learning in Chemical Engineering: A Perspective", *Chemie Ingenieur Technik*, v. 93, n. 12, pp. 2029–2039. doi: <https://doi.org/10.1002/cite.202100083>. Availability: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cite.202100083>.

SEVERTSON, R. B., FRANK, L., ERICSON, G., 2017, "What is the team data science process? - azure architecture center", *Azure Architecture Center | Microsoft Learn*, (Mar). Availability: <https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview?view=azureml-api-2>.

SHALEV-SHWARTZ, S., SINGER, Y., SREBRO, N., et al., 2011. "Pegasos: Primal Estimated sub-gradient solver for SVM". .

SIMON, D., 2006, *Additional topics in Kalman filtering*, v. 1. Hoboken, New Jersey, USA, John Wiley Sons, Ltd. ISBN: 9780470045343. Availability: <https://onlinelibrary.wiley.com/doi/abs/10.1002/0470045345.ch10>.

STANLEY, J. S., 2002, "Dry Gas Seal System Design Standards for Centrifugal Compressor Applications", *Prcodeedings of the Thirty first Turbomachinnery Symposium*, pp. 145– 152.

SUN, Y., BROCKHAUSER, S., 2022, "Machine Learning Applied for Spectra Classification in X-ray Free Electorn Laser Sciences", *Data Science Journal*, (Aug). doi: <10.5334/dsj-2022-015>.

SUTTON, R. S., BARTO, A. G., 2018, *Reinforcement Learning: An Introduction*. The MIT Press. Availability: <http://incompleteideas.net/book/the-book-2nd.html>.

THIL, S., GILSON, M., 2005, "A BAYESIAN APPROACH TO CLOSED-LOOP SYSTEM IDENTIFICATION", *IFAC Proceedings Volumes*, v. 38, n. 1, pp. 644–649. doi: <https://doi.org/10.3182/20050703-6-CZ-1902.00108>. Availability: <https://www.sciencedirect.com/science/article/pii/S1474667016361201>. 16th IFAC World Congress.

TOWSYFYAN, H., GU, F., BALL, A. D., et al., 2018, "Tribological behaviour diagnostic and fault detection of mechanical seals based on acoustic emission measurements", *Friction*, (Nov). doi: <10.1007/s40544-018-0244-4>. Availability: <https://doi.org/10.1007/s40544-018-0244-4>.

TURAN, E. M., JÄSCHKE, J., 2021, "Classification of undesirable events in oil well operation". In: *2021 23rd International Conference on Process Control (PC)*, pp. 157–162.

VARGAS, R., MUNARO, C., CIARELLI, P., et al., 2019, "A realistic and public dataset with rare undesirable real events in oil wells", *Journal of Petroleum Science and Engineering*, v. 181 (07), pp. 106223. doi: <10.1016/j.petrol.2019.106223>.

VENKATASUBRAMANIAN, V., 2019, "The promise of artificial intelligence in chemical engineering: Is it here, finally?" *AIChE Journal*, v. 65, n. 2, pp. 466–478. doi: <10.1002/aic.16489>. Availability: <https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.16489>.

VON CHAMIER, L., LAINE, R. F., JUKKALA, J., et al., 2021, "Democratising deep learning for microscopy with ZeroCostDL4Mic", *Nature Communications*, v. 12, n. 1 (Apr), pp. 2276. doi: <10.1038/s41467-021-22518-0>. Availability: <https://doi.org/10.1038/s41467-021-22518-0>.

WAGNER, W., 1973, "New vapour pressure measurements for argon and nitrogen and a new method for establishing rational vapour pressure equations", *Cryogenics*, v. 13, n. 8, pp. 470 – 482. doi: <10.1016/0011-2275(73)90003-9>. Availability: <http://dx.doi.org/10.1016/0011-2275(73)90003-9>.

WELCH, G., BISHOP, G., 1995, *An Introduction to the Kalman Filter*, v. 1. Chapel Hill, North Carolina, USA, University of North Carolina at Chapel Hill.

ZHOU, W., MARSHALL, E., OSHINOWO, L., 2001, "Modeling LDPE Tubular and Autoclave Reactors", *Industrial & Engineering Chemistry Research*, v. 40, n. 23, pp. 5533–5542. doi: <10.1021/ie0010823>. Availability: <https://doi.org/10.1021/ie0010823>.

Appendices

Appendix A

Statistical techniques

A.1 Regression metrics

All regression metrics are given in terms of error, the difference between the model output $\hat{y}_i$ and the data point y_i, Equation A.1.

$$error_i = y_i - \hat{y}_i \tag{A.1}$$

$$R^2 = 1 - \frac{\sum_{i=0}^{n} error_i^2}{\sum_{i=0}^{n} (y_i - E(t))^2} \tag{A.2}$$

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n} error_i^2}{n}} \tag{A.3}$$

$$Accuracy = \frac{\sum_{i=0}^{n} |error_i| < ME}{n} \tag{A.4}$$

$$Trend\ accuracy = \frac{\sum_{i=1}^{n} ((\hat{y}_i - \hat{y}_{i-1}) \cdot (y_i - y_{i-1}) > 0)}{n} \tag{A.5}$$

R^2 can be negative if the sum of the error squared is bigger than the target's variance. The result that R^2 is always positive is true only for when it stands for the correlation coefficient squared. For linear regression, when analyzing the output of the training dataset, the metric R^2 and the correlation coefficient squared are the same.

A.2 Classification metrics

Visualization of results is important for communication. The most common form of visualization in classification tasks is confusion matrix. It is a matrix in which the rows are the true data class and the columns the model predicted class. A confusion matrix displaying good results has most elements in the di-

agonal, as the model agrees mostly with the data. In unbalanced datasets the confusion matrix is better presented normalized by the number of class members. A sample confusion matrix is given in Figure A.1.

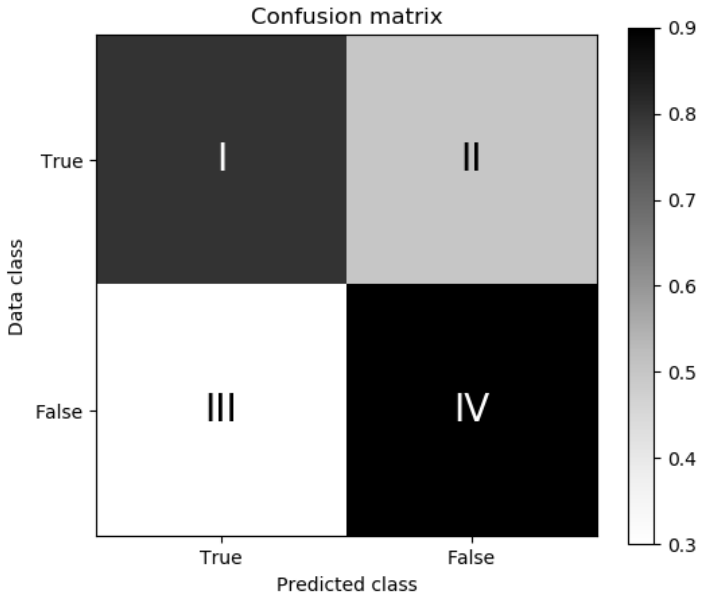


Figure A.1: Confusion matrix example.

Another important aspect is numerical metrics. To facilitate understanding the equations will be given in terms of the squares used in confusion matrix example. The most common and straightforward metric is accuracy: how many samples the model gets right, Equation A.6. But for fault detection and diagnosis other information about the model are also of interest. Precision and recall are important too. In this context, precision is how many detected true faults are relative to all detected faults, Equation A.7; recall is how many detected true faults are relative to all true faults, Equation A.8. To combine both metrics usually a metric called F1-score is used. F1-score is the harmonic mean between precision and recall, Equation A.9.

$$Accuracy = \frac{I + IV}{I + II + III + IV} \tag{A.6}$$

$$Precision = \frac{I}{I + III} \tag{A.7}$$

$$Recall = \frac{I}{I + II} \tag{A.8}$$

$$F1 = \frac{2}{precision^{-1} + recall^{-1}} \tag{A.9}$$

Another common metric is receiver operating characteristic, area under curve

(ROC-AUC). ROC is a plot where the horizontal axis is the true positive rate and the vertical axis the false positive rate, and the data points are sampled at different detection thresholds. This plot shows how the true and false positive rates are dependent on the threshold limits and if the detection thresholds can be moved to prioritize false alarms or false negatives. To reduce the number of plots, the area of the ROC curve is usually reported. The best area possible is 1, a random classifier would give an area of 0.5.

These metrics are designed for binary classification, except accuracy. As the classification is run in a one-vs-rest scheme, they will be evaluated for normal and fault conditions.

A.3 Permutation importance

Permutation importance was first proposed by BREIMAN (2001) and developed in terms of Random Forests. First the baseline accuracy or whatever metric is desired is calculated using the test/validation dataset. Afterwards one variable is shuffled randomly and the desired metric is recalculated. This procedure is repeated for every feature on the dataset. The permutation importance is given by the reduction of the metric evaluated. More important features create a more prominent decrease. Since the random shuffling adds a stochastic factor, the procedure has a relatively high variance and can be repeated n times to evaluate the range of importance values. Compared to Random Forest feature importance, it has the advantages of having less bias to cardinality and less overfitting since it uses data not seen by the model. It also may return a negative or zero importance, while Random Forest feature importance always return some importance to a variable.

A.4 Statistical tests

T-test is a statistical test used to examine if the mean of a population is statistically different from another. In this test the null hypothesis is that the mean of the populations is the same, and the alternative hypothesis is that those means are different. The test is calculated with Equations A.10 and A.11.

$$t = \frac{(\mu_1 - \mu_2)}{s \cdot \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \tag{A.10}$$

$$s = \sqrt{\frac{(n_1 - 1) \cdot var_1 + (n_2 - 1) \cdot var_2}{n_1 + n_2 - 2}} \tag{A.11}$$

Where t is the t statistic, μ are the means of the populations, var is the variance and n are the amount of samples in each population. The t-statistic is assumed to come from Student's t-distribution of degree of freedom = $n_1 + n_2 - 2$, and if its value is below the threshold for statistical significance the null hypothesis is rejected.

It should be noticed that not rejecting the null hypothesis is different than accepting the null hypothesis. But since the purpose of the test is to make an argument towards the stakeholders of the project, a well-known and established statistical test is preferable than a more complex approach. To give a better statistical significance to the T-test an effect size measurement was used among the group, specifically Cohen's d. Effect size is explained in SAWILOWSKY (2009), but generally 0.2 is considered low effect size, 0.5 is considered a medium effect size and 0.8 is considered a high effect size. The Cohen's d is calculated with Equation A.12.

$$d = \frac{(\mu_1 - \mu_2)}{s} \tag{A.12}$$

Where d is Cohen's d. Subscripts 1 and 2 denote the population with the biggest and the smallest means respectively. Cohen's d is similar to t-test, with the only difference being the $\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$ term. When there are too much samples this term makes the test reject the null hypothesis even when the difference between means are negligible.

A.5 Mahalanobis distance

Mahalanobis distance, Equation A.14, is a multivariable extension of the z-score normalization, Equation A.13. Instead of the standard deviation σ, it uses the covariance matrix Σ; and instead of using scalar algebra, it uses matrix operations.

$$z = \frac{x - \mu}{\sigma} \tag{A.13}$$

$$z = \sqrt{(\bar{x} - \bar{\mu})^T \cdot \bar{\bar{\Sigma}}^{-1} \cdot (\bar{x} - \bar{\mu})} \tag{A.14}$$

Just like z-score is related to Student's t distribution, Mahalanobis distance is related to Hotelling's t-squared distribution. It is possible to determine a p-value of a data point belonging to a given gaussian multivariate distribution as the null hypothesis, but since it depends of a normality assumption it was not used. Visual inspection was preferred instead. Which sensors contribute the most for the Mahalanobis distance can be given through Equation A.15.

$$\bar{D} = \bar{\bar{\Sigma}}^{-1} \cdot (\bar{x} - \bar{\mu}) \tag{A.15}$$

where D is a deviation vector.

A.6 Robust covariance estimation

Covariance matrix is a statistic sensitive to outliers. In a outlier detection task this is an important issue, so robust covariance estimators were required for the test. The algorithm used, minimum determinant covariance estimator, is based on the idea that it should be found n "good" samples in the data, where n is a hyperparameter. It is called minimum determinant covariance because the method consistently produces a covariance with a smaller determinant. The determinant can be interpreted as the "volume" of a matrix, and a covariance estimated without outliers would tend to be smaller.

The algorithm works as following: First n samples are selected randomly from the dataset, their mean and covariance are estimated and with those the Mahalanobis distances of all samples is calculated. The samples are ordered from the smallest to highest mahalanobis distance, and the first n samples are selected. With those n samples new Mahalanobis distances are calculated. This process iterates until convergence.

n should be chosen with care. Too few samples and the covariance matrix may be singular, with determinant equal zero and the algorithm may not converge.

A.7 Gaussian mixture models

Gaussian mixture model is a probabilistic model supposing that the data comes from a combination of Gaussian distributions. It was not designed in principle for clusterization, but the means it calculates can be interpreted as the centers of the clusters. In fact GMMs can be interpreted as K-means clusterization with a covariance estimate.

The log-likelihood of each sample in the GMM is given by

$$l(\bar{x}|\theta) = \sum_{i=1}^{K} log(w_i \cdot N(\bar{x}|\bar{\mu}_i, \bar{\bar{\Sigma}}_i)) \tag{A.16}$$

where $l(\bar{x}|\theta)$ is the log likelihood given sample $\bar{x}$ and parameters θ. K is the number of clusters, w_i is the weight of each cluster in the mixture and $N(x|\mu_i, \Sigma_i)$

is the multivariate normal probability function of sample $\bar{x}$ given mean μ_i and covariance matrix $\bar{\bar{\Sigma}}_i$. The further $\bar{x}$ is from the mean the smaller is the log likelihood. This log-likelihood calculation comes with an assumption of normality, so it should not be used to make probabilistic assertions over the dataset, as no real process is Gaussian.

Appendix B

GLOW control

B.1 Additional images

Regarding open-loop simulation, using the parameters found and the same step changes as the data generated for parameter estimation, we can better see the differences between the NMPC and the well model, as shown in Figure B.1. In the region without slugging, there is a near constant offset in the gas mass in the annulus. This is due to the Peng-Robinson equation of state, that has lower compressibility factor at the pressure ranges in the annulus and, therefore, can hold more gas.

The high variations in the annulus after 20 hours appears to be slugging, however it is only a stronger step response as after the step change the system converges to a stable solution.

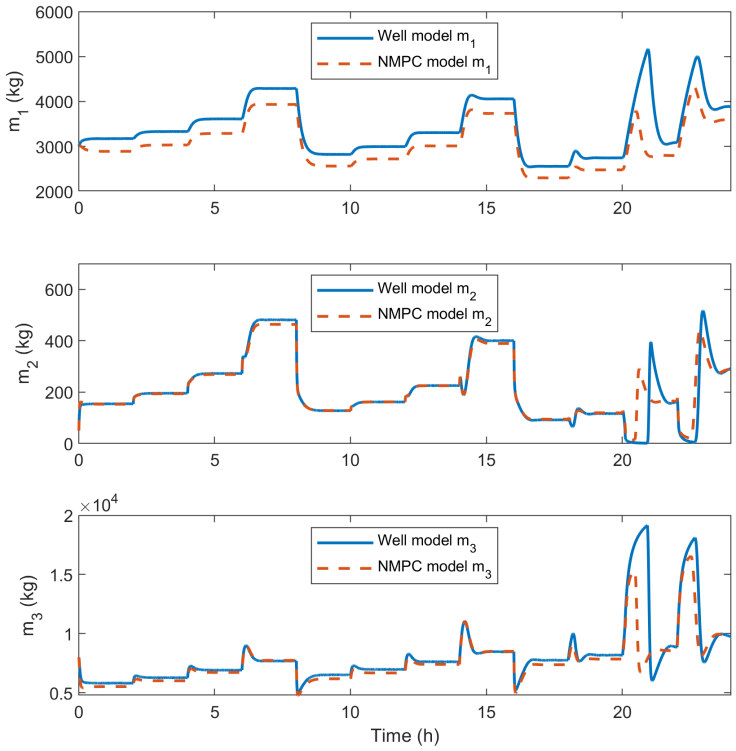


Figure B.1: Comparison between NMPC and GLOW model in an open loop simulation without slugging.

In the slugging region, depicted in Figure B.2, we start to see more differences. During slugging, we see the increase in m_1 due to the interruption of the gas flow into the tubing and its rapid reduction after the pressure increases enough to pass to the tubing, since the gas inlet flow is low. On the tubing side, m_2 also increases momentarily when the gas flow restarts but rapidly decreases when the gas flow is interrupted, while m_3 quickly decreases when gas flow is resumed since it lowers the density of the mixture, increasing oil production, but accumulates when gas flow is interrupted.

The slugging period is much lower in the NMPC model, this is due to the lower pressure in the tubing bottom than in the GLOW model. This happens both due to the lack of pressure drop equations in the NMPC model and due to the higher estimated GOR.

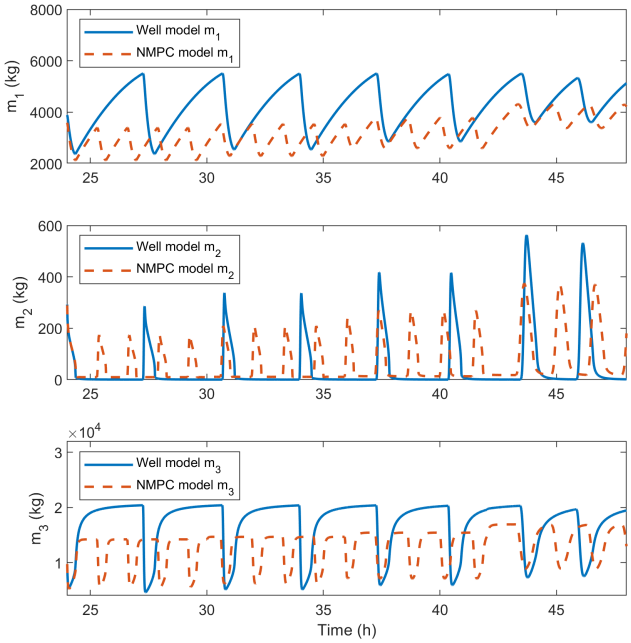


Figure B.2: Comparison between NMPC and GLOW model in an open loop simulation. during slugging.

Figures B.3 and B.4 show how the tuned EKF predicts the internal states of the well. The estimated gas in the annulus has an almost constant offset due to the difference between equations of state. In the tubing it has more difficulty in regions with both more gas and oil as it increases the pressure and makes the ideal gas assumption of the NMPC model less valid. It also overshoots considerably during the step changes at time = 8h and 16h. The NMPC model predicts well these step changes but the changes are too steep, making the linearization of the EKF to fail in giving an accurate representation. After 20h, however, the estimation is bad when the GLOW is filling up with oil. Similarly to what can be seen in Figure B.1, the NMPC model predict the well filling up to end sooner and at lower mass than well model calculates.

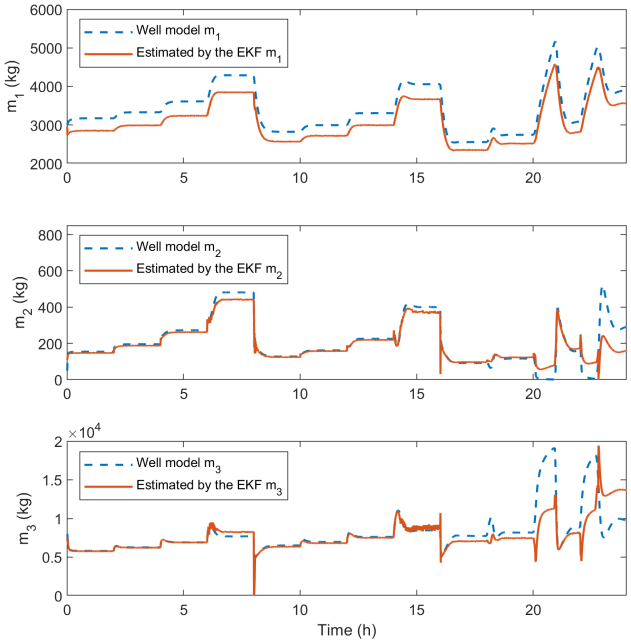


Figure B.3: Comparison between the states estimated by the EKF and the well model states, no slugging.

As we can see in Figure B.4 the EKF predicts the gas entering the annulus earlier than it actually does. It makes sense as the NMPC predicts a shorter slugging period. Overall, the states estimated by the EKF match reasonably well with the well model states given the model mismatch.

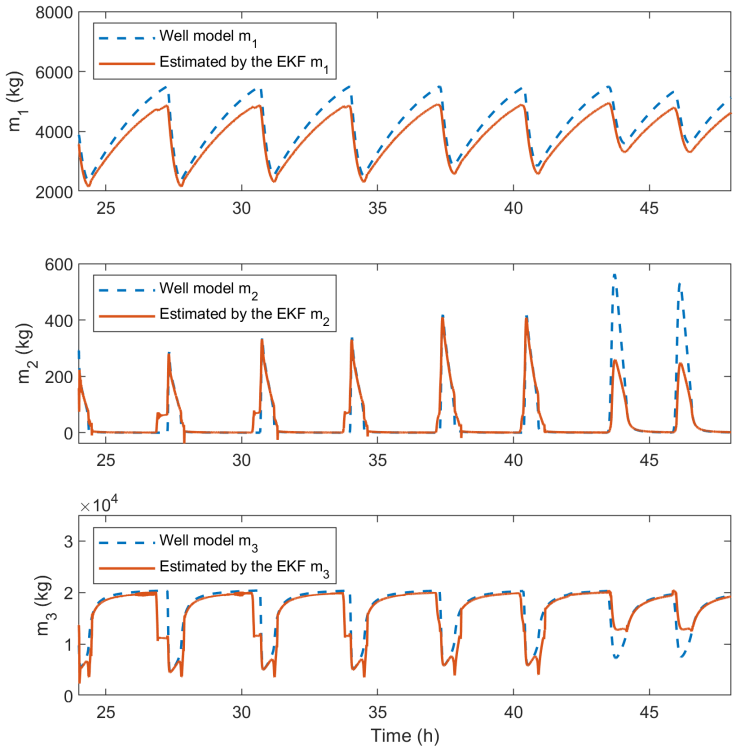


Figure B.4: Comparison between the states estimated by the EKF and the well model states, slugging.

B.2 Equations and constants

In these equations $PR_{EoS,P}$ is used to calculate pressure given temperature molar volume and $PR_{EoS,V}$ to calculate molar volume given pressure and temperature. In both cases molar volume is given in terms of density. There is only gas inside the annulus, so the pressure at top of the annulus is given by Peng-Robinson equation:

$$P_{at} = PR_{EoS,P}(T_a, \frac{M_G V_a}{m_1}) \tag{B.1}$$

and pressure at bottom is given by pressure at the top plus the gas weight:

$$P_{ab} = P_{at} + \frac{m_1 g L_a}{V_a} \tag{B.2}$$

At this point gas density is:

$$\rho_{G,ab} = M_G / PR_{EoS,V}(T_a, P_{ab}) \tag{B.3}$$

The density of the gas into the annulus is:

$$\rho_{G,in} = M_G / PR_{EoS,V}(T_a, P_{gs}) \tag{B.4}$$

Thus, gas mass flow into the annulus is:

$$w_{G,in} = K_{gs} u_2 \sqrt{\rho_{G,in} max(P_{gs} - P_{at}, 0)} \tag{B.5}$$

The gas density at the tubing top is:

$$\rho_{G,t} = \frac{m_2}{V_t + S_{bh} L_{bh} - m_3/\rho_L} \tag{B.6}$$

The pressure at the top is given by $\mathrm{PR_{EoS}}$

$$P_{tt} = PR_{EoS,P}(T_t, \frac{M_G}{\rho_{G,t}}) \tag{B.7}$$

The average density inside the tubing:

$$\bar{\rho}_{mix} = \frac{m_2 + m_3 - \rho_L S_{bh} L_{bh}}{V_t} \tag{B.8}$$

The average liquid volume fraction inside tubing:

$$\bar{\alpha}_{mix} = \frac{m_2 + m_3 - \rho_L S_{bh} L_{bh}}{V_t} \tag{B.9}$$

The gas mass fraction at bottom of the tubing is:

$$\alpha^m_{G,b} = \frac{GOR}{GOR + 1} \tag{B.10}$$

The pressure drop due to friction is needed to determine bottom-hole pressure, which is needed to determine the gas injection rate which is used to calculate the pressure drop. This part would turn the ordinary differential equation

system into a differential algebraic equation system. To keep it an ordinary differential equation system JAHANSHAHI *et al.* (2012) assumed a constant mean inlet flow rate, here inlet flow rate is replaced by an exponential filter of previous inlet flow rates, Equation 4.4. This change slightly increases the stiffness of the system.

The average superficial velocity of liquid phase in tubing:

$$\bar{U}_{sl,t} = \frac{4(1-\alpha^m_{G,b})\cdot m_4}{\rho_L \pi D_t^2} \tag{B.11}$$

The average superficial velocity of gas phase in tubing:

$$\bar{U}_{sg,t} = \frac{4(w_{G,in}-\alpha^m_{G,b})\cdot m_4}{\rho_{G,t} \pi D_t^2} \tag{B.12}$$

The average mixture velocity in tubing:

$$\bar{U}_{m,t} = \bar{U}^m_{sl,t} + \bar{U}^m_{sg,t} \tag{B.13}$$

The Reynolds number of flow in tubing:

$$Re_t = \frac{\bar{\rho}_{mix}\bar{U}_{m,t}D_t}{\mu} \tag{B.14}$$

An explicit approximation of the Colebrook-White equation proposed by HAALAND (1983) is used as the friction factor in the tubing:

$$\frac{1}{\sqrt{\lambda_t}} = -1.8 log_{10}[(\frac{\epsilon/D_t}{3.7})^{1.1} + \frac{6.9}{Re_t}] \tag{B.15}$$

The pressure loss due to friction:

$$F_t = \frac{\alpha_L \lambda_t \bar{\rho}_{mix} \bar{U}^2_{m,t} L_t}{2D_t} \tag{B.16}$$

The pressure at bottom of the tubing where gas is injected from the annulus:

$$P_{tb} = P_{tt} + \bar{\rho}_{mix} g L_t + F_t \tag{B.17}$$

The mass flow rate of gas injected into tubing:

$$w_{G,inj} = K_{inj}\sqrt{\rho_{G,ab} max(P_{ab} - P_{tb}, 0)} \tag{B.18}$$

The liquid velocity at bottom-hole:

$$\bar{U}_{l,b} = \frac{m_4}{\rho_L S_{bh}} \tag{B.19}$$

The Reynolds number of flow at bottom-hole:

$$Re_b = \frac{\rho_L \bar{U}_{l,b} D_b}{\mu} \tag{B.20}$$

The friction factor at bottom-hole:

$$\frac{1}{\sqrt{\lambda_b}} = -1.8 log_{10}[(\frac{\epsilon / D_b}{3.7})^{1.1} + \frac{6.9}{Re_b}] \tag{B.21}$$

The pressure loss due to friction from bottom-hole to injection point:

$$F_b = \frac{\lambda_b \rho_L \bar{U}_{l,b}^2 L_b h}{2 D_b} \tag{B.22}$$

The pressure at bottom-hole:

$$P_{bh} = P_{tb} + \rho_L g L_b h + F_b \tag{B.23}$$

The mass flow rate from reservoir to tubing:

$$w_{res} = PI max(P_{res} - P_{bh}, 0) \tag{B.24}$$

The mass flow rate of liquid from reservoir to tubing:

$$w_{L,res} = (1 - \alpha_{G,b}^m) w_{res} \tag{B.25}$$

The mass flow rate of gas from reservoir to the well:

$$w_{G,res} = (\alpha_{G,b}^m) w_{res} \tag{B.26}$$

The density of gas at bottom of tubing:

$$\rho_{G,tb} = \frac{M_G}{PR_{EoS,V}(T_t, P_{tb})} \tag{B.27}$$

The liquid volume fraction at bottom of tubing:

$$\alpha_{L,b} = \frac{w_{L,res}\rho_{G,tb}}{w_{L,res}\rho_{G,tb} + (w_{G_inj} + w_{G,res})\rho_L} \tag{B.28}$$

Here a liquid volume fraction assumption is used to estimate liquid volume fraction at top of the tubing:

$$\alpha_{L,t} = max(min(2\bar{\alpha}_L - \alpha_{L,b}, 0), 1) \tag{B.29}$$

From the previous assumption mixture density at top of the tubing is:

$$\rho_{mix,t} = \alpha_{L,t}\rho_L + (1 - \alpha_{L,t})\rho_{G,t} \tag{B.30}$$

The mass flow rate of mixture from production valve:

$$w_{out} = K_{pr}u_1\sqrt{\rho_{mix,t}max(P_{tt} - P_0)} \tag{B.31}$$

The volumetric flow rate of production valve:

$$Q_{out} = \frac{w_{out}}{\rho_{mix,t}} \tag{B.32}$$

The gas mass fraction at top of tubing:

$$\alpha^m_{G,t} = \frac{(1 - \alpha_{L,t})\rho_{G,t}}{\alpha_{L,t}\rho_L + (1 - \alpha_{L,t})\rho_{G,t}} \tag{B.33}$$

The mass flow rate of outlet gas from tubing:

$$w_{G,out} = \alpha^m_{G,t}w_{out} \tag{B.34}$$

Table B.2: Step changes on the process applied for data generation.

Period (h)	u_1	u_2	Period (h)	u_1	u_2
0-2	0.88	0.88	16-18	0.88	0.40
2-4	0.64	0.88	18-20	0.64	0.40
4-6	0.40	0.88	20-22	0.40	0.40
6-8	0.17	0.88	22-24	0.17	0.40
8-10	0.88	0.64	24-30	0.88	0.17
10-12	0.64	0.64	30-36	0.64	0.17
12-14	0.40	0.64	36-42	0.40	0.17
14-16	0.17	0.64	42-48	0.17	0.17

The mass flow rate of outlet liquid from tubing:

$$w_{L,out} = (1 - \alpha_{G,t}^{m}) w_{out} \tag{B.35}$$

The process constants are given in the Table B.1:

Table B.1: Model parameters.

Symbol	Description	value	unit
R	Universal gas constant	8314	$J/(kmol \cdot K)$
g	Gravity	9.81	m/s^2
μ	Viscosity	$3.64 \cdot 10^{-3}$	Pa·s
ρ_L	Oil density	760	kg/m^3
M_G	Natural gas molecular weight	16.7	gr
T_a	Annulus temperature	348	K
V_a	Annulus volume	64.34	m^3
L_a	Annulus length	2048	m
P_{gs}	Gas source pressure	140	bar
V_t	Tubing volume	25.03	m^3
S_{bh}	Cross sectional area below injection point	0.0314	m^2
L_{bh}	Length below injection point	75	m
T_t	Tubing temperature	369.4	K
GOR	Gas oil ratio	0	-
P_{res}	Reservoir pressure	160	bar
D_t	Tubing diameter	0.134	m
L_t	Tubing length	2048	m
PI	Productivity index	$2.47 \cdot 10^{-6}$	kg/(s·Pa)
K_{gs}	Gas inlet valve constant	$9.98 \cdot 10^{-5}$	-
K_{inj}	Injection valve constant	$1.40 \cdot 10^{-4}$	-
K_{pr}	Production valve constant	$2.90 \cdot 10^{-3}$	-

B.3 Relevant data

Table B.3: MPC parameters.

Parameter	value
Discretization interval	10s
Control horizon	8
Prediction horizon	8
γ	[5 for gas input flow, 6 for oil output flow]
φ	[1,1]
λ	[1,1]
Algorithm	Sequential Quadratic Programming
Max iterations	15
U tolerance	10^{-3}
Function tolerance	10^{-3}

Table B.4: NN hyperparameters.

Parameter	value
Number of neurons	6
Training algorithm	BFGS
L2 regularization term	10^{-3}
Train/test/validation split	30/35/35

Table B.5: EKF and NN state estimator NMPC parameters.

Parameter	value
Sampling time	40s
Number of finite elements	57
Control horizon	5
Prediction horizon	12
γ	[1 for gas input flow rate, 5 for oil output flow rate]
φ	[1,1]
Max iterations	35
u tolerance	10^{-3}
Objective function tolerance	10^{-3}

B.4 Extended Kalman Filter equations

Assuming there is a nonlinear dynamic system that follows the Equations B.36 and B.37.

$$\dot{x} = f(x, u, t) + q \tag{B.36}$$

$$y_k = h(x_k, u) + r \tag{B.37}$$

where $f()$ is a nonlinear function that depends on the state x and control action u. h is a function that relates the state and control action to the measurements y. q is a random Gaussian noise, with zero mean and covariance matrix Q, while r is a random Gaussian noise, with zero mean and covariance matrix R.

Additionally, the Jacobian matrices are given by Equations B.38 and B.39.

$$F = \frac{\partial f}{\partial x}\Big|_{\hat{x},u} \tag{B.38}$$

$$H_k = \frac{\partial h}{\partial x}\Big|_{\hat{x}_k,u_k} \tag{B.39}$$

The filter works in two stages. A prediction stage, Equations B.40 and B.41 and a correction stage, Equations B.42 to B.44.

Prediction:

$$\widetilde{x}_k = \hat{x}_{k-1} + \int_{k-1}^{k} f(\hat{x}, u, \tau)\, d\tau \tag{B.40}$$

$$\widetilde{P}_k = \hat{P}_{k-1} + \int_{k-1}^{k} F\hat{P} + \hat{P}F^T + Q\, d\tau \tag{B.41}$$

Correction:

$$e_k = z_k - h(\widetilde{x}_k, u_k) \tag{B.42}$$

$$K_k = \widetilde{P}_k\, H_k^T\, (H_k\, \widetilde{P}_k\, H_k^T + R)^{-1} \tag{B.43}$$

$$\hat{x}_k = \widetilde{x}_k + K_k\, e_k \tag{B.44}$$

$$\hat{P}_k = (I - K_k\, H_k)\, \widetilde{P}_k \tag{B.45}$$

where P is the states covariance matrix, z is the measurements, and e the error between the measurements and what the measurements should be given the predicted states. K_k is the Kalman gain. Variables with tilde are predicted and variables with hat are corrected.

While Q and R are given by the covariance of the Gaussian noise on the formulation, in practice they are treated as tuning parameters (SIMON, 2006). A smaller Q reflects more trust in the model while a smaller R means more trust in the measurements (SALAU *et al.*, 2009).

Appendix C

Oil well fault detection

C.1 Additional images

Figure C.1 shows an example of a sensor with overly interpolated data. In this case T-JUS-CKP.

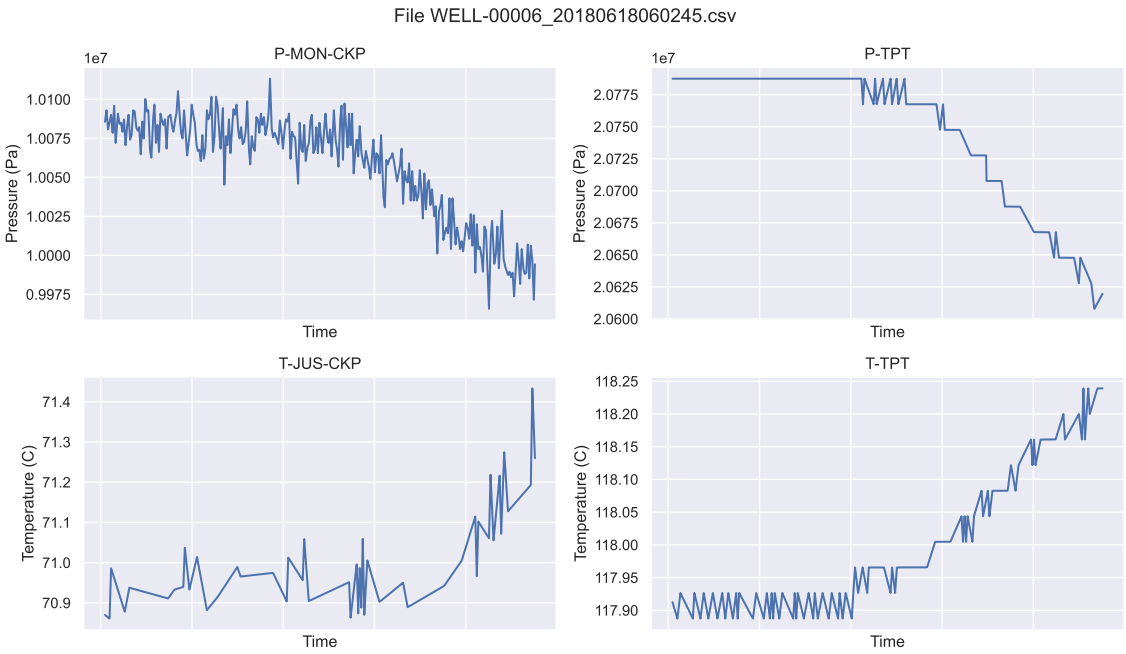


Figure C.1: Example of file with overly interpolated sensor.

Figure C.2 shows how the estimated flowrates behave on simulated data. They follow the same general direction, and $\tilde{Q}_{p1}$ agrees more with $\tilde{Q}_t$ than with $\tilde{Q}_{p2}$.

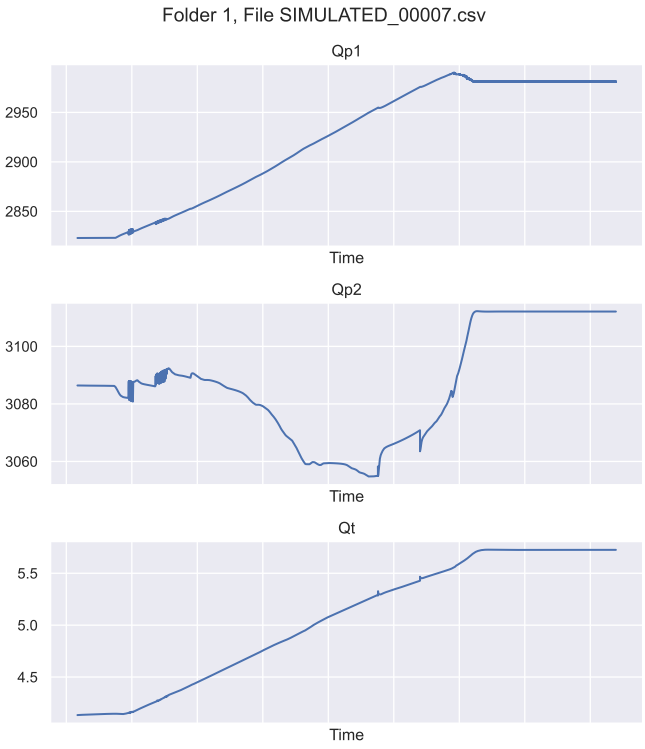


Figure C.2: Example of flowrate estimation with sensors.

C.2 Intermediate results of the iterations

The result of every step in the iterative process is summarized in Table C.1 for the validation dataset and in Table C.2 for the test dataset.

Iteration	0				1			
Fault	Accuracy	F1-score	Precision	Recall	Accuracy	F1-score	Precision	Recall
1	85.65%	95.34%	91.59%	99.41%	86.09%	98.57%	99.75%	97.41%
2	99.63%	87.03%	77.57%	99.12%	98.80%	98.20%	97.03%	99.40%
3	100.00%	100.00%	100.00%	100.00%	93.72%	94.76%	90.61%	99.32%
4	94.91%	89.59%	81.82%	98.98%	95.49%	89.51%	87.56%	91.55%
5	99.61%	95.73%	93.08%	98.53%	84.47%	99.43%	99.54%	99.33%
6	99.95%	0.00%	0.00%	0.00%	76.84%	99.21%	100.00%	98.44%
7	99.85%	97.55%	97.18%	97.92%	99.37%	94.60%	90.71%	98.85%

Iteration	2				3			
Fault	Accuracy	F1-score	Precision	Recall	Accuracy	F1-score	Precision	Recall
1	90.46%	98.74%	99.89%	97.61%	95.89%	99.69%	99.80%	99.58%
2	99.80%	98.92%	98.12%	99.74%	99.83%	99.63%	99.66%	99.60%
3	97.14%	97.45%	99.20%	95.75%	99.87%	99.83%	99.76%	99.83%
4	94.81%	87.45%	88.94%	86.01%	97.61%	94.29%	94.77%	93.82%
5	96.50%	99.35%	99.71%	98.99%	98.20%	99.62%	99.66%	99.59%
6	90.67%	99.25%	100.00%	98.52%	92.48%	99.69%	99.97%	99.42%
7	99.37%	94.61%	90.71%	98.85%	99.76%	95.47%	95.43%	95.52%

Table C.1: Results for each iteration of the methodology, validation dataset.

Iteration	0				1			
Fault	Accuracy	F1-score	Precision	Recall	Accuracy	F1-score	Precision	Recall
1	98.81%	83.03%	86.64%	80.09%	97.08%	32.84%	32.36%	33.33%
2	24.52%	38.13%	39.37%	36.98%	82.66%	76.35%	84.68%	69.50%
3	3.98%	3.83%	1.99%	50.00%	3.98%	3.83%	1.99%	50.00%
4	0.00%	0.00%	0.00%	0.00%	9.88%	8.99%	5.30%	29.79%
5	68.34%	48.12%	41.68%	58.98%	75.15%	77.88%	74.25%	81.88%
6	94.25%	46.16%	41.45%	52.07%	75.06%	78.22%	90.94%	68.63%
7	97.75%	32.95%	32.59%	33.33%	36.20%	38.99%	34.44%	44.93%

Iteration	2				3			
Fault	Accuracy	F1-score	Precision	Recall	Accuracy	F1-score	Precision	Recall
1	97.00%	32.86%	32.33%	33.33%	97.08%	32.84%	32.36%	33.33%
2	93.33%	89.25%	88.09%	90.55%	91.75%	86.34%	85.80%	87.43%
3	4.71%	51.03%	52.03%	50.06%	17.23%	54.40%	52.13%	54.63%
4	16.70%	14.31%	8.35%	50.00%	93.04%	90.25%	85.30%	95.82%
5	82.17%	75.57%	74.17%	77.02%	94.80%	92.35%	93.29%	92.60%
6	99.49%	79.74%	94.65%	70.78%	99.49%	80.58%	96.33%	69.26%
7	36.21%	38.99%	34.44%	44.93%	97.76%	32.96%	32.59%	33.33%

Table C.2: Results for each iteration of the methodology, test dataset.

Appendix D

LDPE production

D.1 Additional images

In this Appendix there are images that helped during the steps of the project analysis. They are here for future reference.

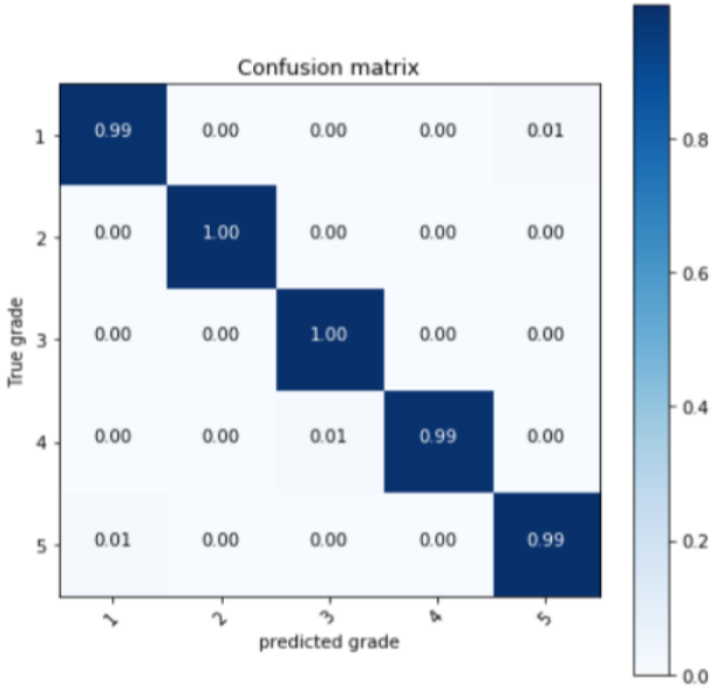


Figure D.1: Confusion matrix of the grade detector.

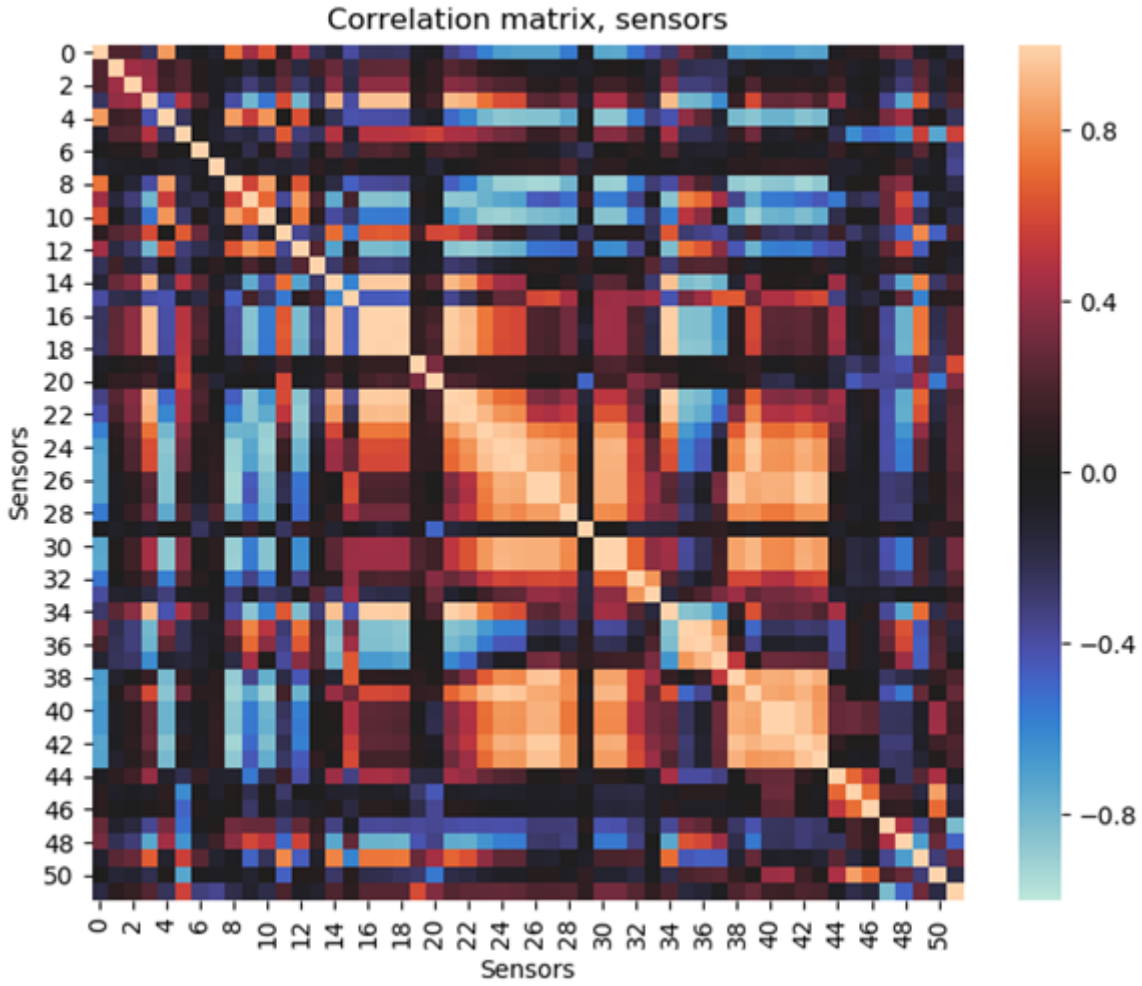


Figure D.2: Correlation matrix of the process sensors.