Pontifícia Universidade Católica
do Rio de Janeiro

Felipe da Costa Pereira

Virtual Flow Metering and Synthetic Well Logs Generation Using Multimodel Machine Learning Strategies and System Identification

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Engenharia Mecânica, do Departamento de Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica.

Advisor : Prof. Márcio da Silveira Carvalho
Co-advisor: Prof. Helon Vicente Hultmann Ayala

Rio de Janeiro
May 2025

Pontifícia Universidade Católica
do Rio de Janeiro

Felipe da Costa Pereira

Virtual Flow Metering and Synthetic Well Logs Generation Using Multimodel Machine Learning Strategies and System Identification

Dissertation presented to the Programa de Pós–graduação em Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica. Approved by the Examination Committee:

Prof. Márcio da Silveira Carvalho
Advisor
Departamento de Engenharia Mecânica – PUC-Rio

Prof. Helon Vicente Hultmann Ayala
PUC-PR

Prof. Florian Alain Yannick Pradelle
Departamento de Engenharia Mecânica – PUC-Rio

Dr. Fabio Cesar Diehl
Petrobras

Rio de Janeiro, May the 14th, 2025

Felipe da Costa Pereira

Graduated in electrical engineering with an emphasis on electronics and computing by the Federal University of Rio de Janeiro, UFRJ (Rio de Janeiro - RJ, 2007). Graduated in engineering by Ecole Centrale de Lyon (Lyon - France, 2007).

Bibliographic data

Pereira,Felipe da Costa

Virtual Flow Metering and Synthetic Well Logs Generation Using Multimodel Machine Learning Strategies and System Identification / Felipe da Costa Pereira; advisor: Márcio da Silveira Carvalho; co-advisor: Helon Vicente Hultmann Ayala. – 2025.

106 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Mecânica, 2025.

Inclui bibliografia

1. Engenharia Mecânica – Teses. 2. Aprendizagem de máquina. 3. Medidores virtuais de vazão. 4. Perfilagem de poços. 5. Generalização empilhada. 6. Identificação de sistemas. I. Carvalho, Márcio da Silveira. II. Ayala, Helon Vicente Hultmann. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Mecânica. IV. Título.

CDD: 621

To the cherished memory of my dear parents, Isidro and Balbina, whose influence on my life is beyond measure. I am forever thankful for their sacrifices and love, which have supported me every single day.

Acknowledgments

I would like to express my profound gratitude to all those who supported me throughout my journey in completing this dissertation.

First and foremost, I am deeply thankful to my co-advisor, who was actually the advisor in most part of the journey, Professor D.Sc. Helon Ayala, for taking me out of the cave and showing me a completely new world of possibilities through the world of academic research and scientific writing. His guidance, support, encouragement and mastery have been invaluable. The valuable insights and feedback have significantly enhanced my research and influenced my academic path. I also thank him for the opportunities provided and for all the patience and availability during the last couple of years.

I would also like to express my profound gratitude to Professor D.Sc. Márcio Carvalho, who became my official advisor by the end of the process and whose contribution, knowledge, and support were of extreme importance in making this accomplishment possible.

Many thanks to my wife Carla and my daughter Helena, for daily recharging my supply of love and energy and for being my constant source of joy.

I cannot forget to thank Pedro Henrique Cardoso Paulo, my colleague at Petrobras and at PUC-RJ, my personal python guru and co-author of the publications presented in this dissertation. I hope we can keep on thinking of new ideas and projects to work on together.

I also thank Luiz Fofano, my fellow reservoir engineer at Petrobras, for encouraging me to pursue this goal, always listening, debating, giving feedback, proposing ideas and providing insights.

Finally, I thank Petrobras for giving me this opportunity to enhance my career with such an important accomplishment.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001

Abstract

Pereira,Felipe da Costa; Carvalho, Márcio da Silveira (Advisor); Ayala, Helon Vicente Hultmann (Co-Advisor). **Virtual Flow Metering and Synthetic Well Logs Generation Using Multimodel Machine Learning Strategies and System Identification**. Rio de Janeiro, 2025. 106p. Dissertação de Mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Multiphasic flow metering and well logging are key information that helps oil and gas companies understand reservoir characteristics and make strategic decisions. Recently, machine learning models have become an alternative to traditional data acquisition procedures based on physical hardware or first principle models, since the latter are, in general, expensive, time-consuming, and often unreliable. In this work, new machine learning approaches are proposed to enhance the performance of virtual flow metering models and synthetic well log generation. For that purpose, this dissertation develops three application cases based on actual well data from the Volve Field in the Nothern Sea. To address the problem of virtual flowmetering, the approach of system identification optimal lag detection was used to enhance performance. By using stacking generalization, which involves combining the predictions of base learners into a second regression problem, along with training a second model specialized in predicting residuals, reduced the RMSE by 29% and 13%, respectively, when compared to other benchmark regressors. For well log generation purposes, seasonal decomposition was used whether as a feature extraction method or as an ensemble strategy approach. Models in which the input space was expanded based on seasonal decomposition reduced the RMSE in 27% for compressional slowness log. Ensemble-based models specialized in predicting trend and seasonal parts of the data outperformed the baseline models for 6 of the 8 examined estimators and reduced the RMSE in 5.2% compared to the baseline strategy. These findings provide new guidelines for research in the field of virtual flow metering and well logs generation as well as in other domains.

Keywords

Machine learning; Virtual flow meters; Well logging; Stacking generalization; System identification.

Resumo

Pereira,Felipe da Costa; Carvalho, Márcio da Silveira; Ayala, Helon Vicente Hultmann. **Medição Virtual de Vazão e Geração de Perfis Sintéticos de Poços de Petróleo Utilizando Estratégias de Aprendizagem de Máquina Multimodelos e Identificação de Sistemas Dinâmicos**. Rio de Janeiro, 2025. 106p. Dissertação de Mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Medição de fluxo multifásico e perfilagem de poços são informações chave que ajudam empresas de petróleo e gás a caracterizar reservatórios e tomar decisões estratégicas. Nas últimas décadas, modelos de aprendizado de máquina vêm se tornando uma alternativa aos métodos tradicionais baseados em hardware ou modelos de princípios físicos, pois estes últimos geralmente são onerosos em tempo e custo e podem apresentar falhas. Neste trabalho, propomos novas abordagens a fim de melhorar o desempenho dos modelos de medição de fluxo multifásica e a geração de perfis sintéticos de poços. Assim, esta dissertação desenvolve três casos de estudo baseados em dados reais de poços do campo de Volve, no Mar do Norte. Primeiro, a aplicação de modelos empilhados e identificação de sistemas melhorou as métricas de RMSE em 4,5% a 29% em comparação com modelos benchmark. Identificação de sistemas mostrou-se eficaz para avaliar os lags ótimos, enquanto os ensembles empilhados potencializaram os diversos modelos base pré-treinados. Em segundo lugar, uma abordagem de treinamento em duas etapas, aplicando um modelo de resíduos, melhorou a qualidade das previsões das vazões em 1% a 13%, dependendo da complexidade do estimador base. No terceiro caso, a decomposição sazonal foi usada como extração de atributos bem como estratégia de ensemble para geração de perfis. Enquanto a primeira promoveu um ganho de 27% em RMSE para o perfil sônico, a segunda superou os modelos benchmark para 6 dos 8 estimadores testados para todos os perfis e obteve redução do RMSE em 5,2% para o perfil de densidade. Em todos casos testados, ambos os métodos obtiveram resultados melhores do que a estratégia convencional. Esses resultados fornecem novas perspectivas para a pesquisa no campo da medição virtual de vazão e geração de perfis sintéticos de poços, bem como em outros domínios.

Palavras-chave

Aprendizagem de máquina; Medidores virtuais de vazão; Perfilagem de poços; Generalização empilhada; Identificação de sistemas.

Table of contents

List of figures

List of tables

List of Abreviations

ML – Machine Learning

MPFM – Multi-Phase Flow Meter

VFM – Virtual Flow Meter

PDG – Pressure Downhole Gauge

TPT – Temperature and Pressure Transducer

ANN – Artificial Neural Network

RNN – Recurrent Neural Network

CNN – Convolutional Neural Network

MLP – Multi-layer Perceptron

LSTM – Long-Short Term Memory

GRU – Gated Recurrent Unit

ARX – Autoregressive Model with Exogenous inputs

NARX – Non-linear Autoregressive Model with Exogenous inputs

ODE – Ordinary differential equation

TPDAT – Time Patch Dynamic Attention Transformer

DR – Data Reconciliation

DT – Sonic Transit Time log

DTS – Sonic Transit Time log of the Shear Wave

NPHI – Neutron Porosity log

GR – Gamma Ray log

PSO – Particle Swarm Optimization

PINN – Physics-informed Neural Network

NMR – Nuclear Magnetic Resonance log

XGBoost – Extreme Gradient Boosting

AdaBoost – Adaptive Boosting

OLS – Ordinary Least Squares

SVM – Support Vector Machines

SVR – Support Vector Regression

KNN – K-nearest Neighbors

DT – Decision Trees

CART – Classification and Regression Tree

GB – Gradient Boosting

LightGBM – Light Gradient Boosting Machine

t-SNE – t-Distributed Stochastic Neighbor Embedding

SISO – Single Input and Single Output

MISO – Multiple Input and Single Output

FS – Free Simulation

OSA – One Step Ahead

API – Application Programming Interface

IFAC – International Federation of Automatic Control

CBA – Congresso Brasileiro de Automática

SBA – Sociedade Brasileira de Automática

RACEHM – Resistivity log

RHOB – Bulk Density log

PHIF – Final Porosity log

T – Trend Component of a Seasonal Decomposition

S – Seasonal Component of a Seasonal Decomposition

R – Residual Component of a Seasonal Decomposition

All models are wrong, but some are useful

George E. P. Box, .

1
Introduction

Artificial intelligence (AI) is a comprehensive scientific field focused on developing systems capable of interacting with their surroundings and optimizing their actions to successfully reach objectives [1]. This involves developing computational models of cognitive processes such as learning, reasoning, and problem-solving [2].

A notable subfield of the wide concept of AI is Machine Learning (ML), which consists of algorithms that allow computer systems to learn from data without being explicitly programmed [3]. These algorithms identify patterns and regularities in data, enabling them to make predictions or decisions on unseen data and being, thus, able to solve complex problems and identify complicated patterns as long as sufficient data is available [4].

In recent years, there has been a growing interest in AI across numerous sectors of the global economy, with the oil and gas industry being particularly significant [5]. Exploring and producing hydrocarbon reservoirs is a vital activity in the global economy as the oil and gas industry is responsible for the primary fuel sources used around the world [6]. A variety of branches of this industry are increasingly adopting AI and ML to enhance various aspects of its upstream and downstream disciplines such as predictive maintenance and equipment monitoring [7, 8, 9], drilling and completion [10, 11, 12], pipeline monitoring [13, 14, 15] and supply chain and logistics [16].

More specifically into the subsurface domain, a set of recent works notably improved the expertise of the operators in the field of seismic interpretation [17, 18], petrophysics [19, 20], geological characterization [21, 22], reservoir engineering [23, 24, 25], history matching [26, 27], production optimization [28, 29], compositional simulation [30, 31], geomechanics [32] and carbon capture and storage (CCS) [33].

1.1
Measurements in Offshore Production Systems

Typical offshore petroleum production systems consist of several wells connected to a processing unit. Each well produces a fluid that is a multiphase mixture of oil, gas, and water phases [34]. The streams from such wells are

usually collected either on the seabed (subsea manifold) or at the processing unit (production separator), as schematically represented in figure 1.1. The production separator then divides the inlet stream into oil, gas and water so that the volumes and rates of each of the three phases in the outlet can be accurately obtained using conventional single-phase flow sensors [35, 36].

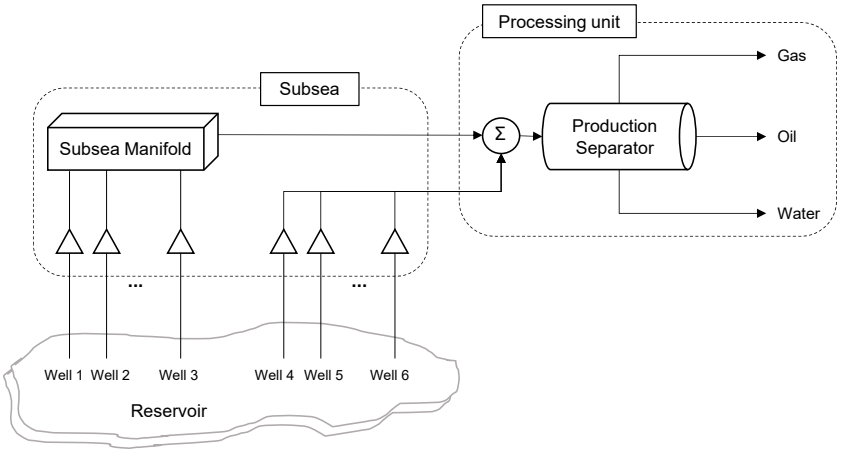


Figure 1.1: Simplified hypothetical 6-well offshore production system diagram. Wells 1, 2 and 3 are tied to a subsea manifold while wells 4, 5 and 6 are directly connected to the production unit (satellite wells).

Although the total production rate of the unit is valuable information, accurately measuring the oil, water, and gas flow rates for each individual well is critical for short-term reservoir management decisions [37], such as optimization of production and allocation of rates [35, 38]. Operators also rely on the production information from each well to make long-term strategic decisions to evaluate the economic value of their assets, increasing the value extracted from the operated field [35, 39].

Therefore, a second separation system is usually available to separate fluids from a single well to accurately measure its three-phase flow rates [40]. This vessel is called a test separator, and this operation of measuring flow rates is known as a well test. As test-separator measurements are obtained for a single well at a time and a well test requires hours of continuous stabilized flow [36, 41], most of the time the well flow rates are not continuously acquired. The full historical production flow rates for each of the connected wells are then uncertain, as they are usually reported as an apportionment of the total production of the unit. Another disadvantage of this method is that the operational conditions on the flow line can change when the well is tied to the

test separator, so that the actual rate is different from the measured one [41]. Furthermore, the well test separator system requires a dedicated pipe and a separator vessel, which increases the cost of the project [35]. A schematic view of this approach is illustrated in figure 1.2.

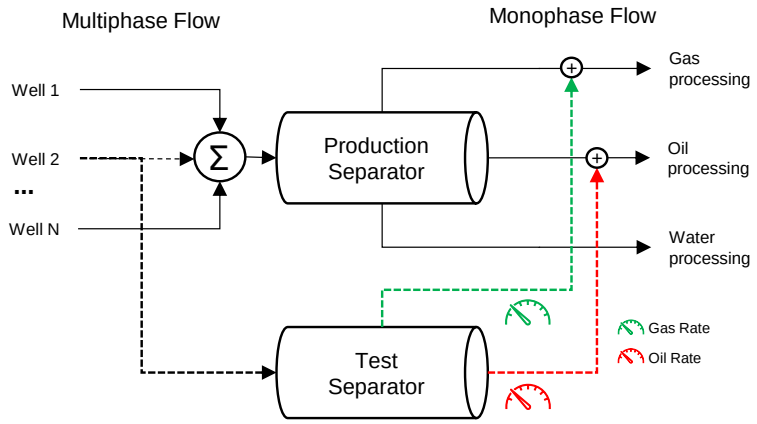


Figure 1.2: Well test scheme, production of well 2 is hypothetically alligned to the test separator. After separation, in monophsic conditions, the oil, water and gas rates can be properly measured.

An alternative to the traditional well test procedure and the test-based apportionment to determine individual wells' historic production is the use of physical multi-phase flow meters (MPFM). These devices are commonly installed well by well and have the ability to measure the three phases without separating them [35]. The main advantage over well testing is that MPFM can track well rates in real time [35]. However, since multiphase flow is complex, turbulent, and chaotic, the acquisition of accurate, reliable, and repeatable rate measurements can be a challenge [42]. Furthermore, MPFM is expensive equipment that can have a low accuracy outside its operational range [35]. A full study on the principles of multiphase flow metering is provided by Falcone [34] and an overview of the technology is presented by Falcone et al. [43].

To address the drawbacks and challenges posed by the above-mentioned methods, oil and gas operators have widely been using Virtual Flow Meters (VFM) as another alternative for well rate estimation. VFM systems have the advantage of not requiring the installation of additional measuring equipment and are divided into two main categories, depending on their modeling paradigm: First Principle VFM and Data-driven VFM [35].

First Principle VFMs are based on numerical flow models composed of all the production system components: fluid and reservoir models, well, valves, subsea equipment and pipes geometry, as well as facilities process variables. Such models are based on conservation equations [35] and apply

an optimization algorithm to tune their parameters to minimize the mismatch between the measured and simulated data. These data are usually acquired by well sensors, which are schematically represented in Figure 1.3.

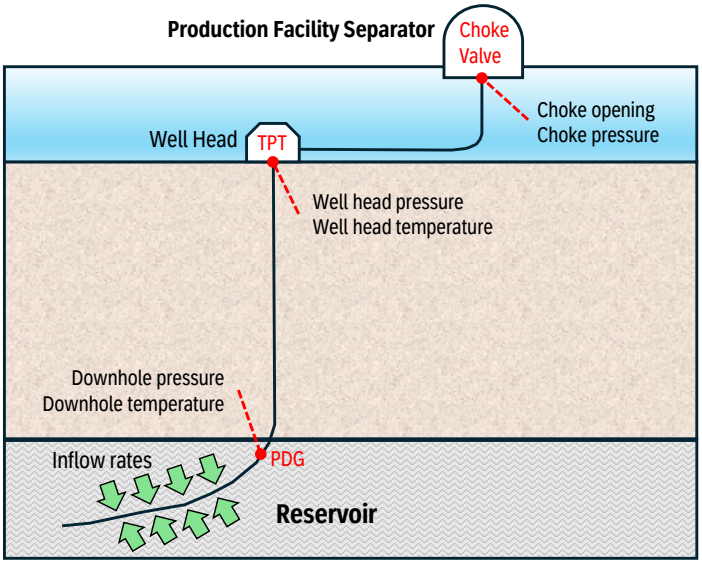


Figure 1.3: Typical offshore well and its pressure-temperature sensors: PDG (Pressure Downhole Gauge) and TPT (Temperature and Pressure Transducer, located in the well head).

The dependence on well sensors can also be challenging as there is always a risk of measurement loss. The downhole pressure gauge (PDG), one of the most valuable sensors that provide useful information for reservoir management [44], has a high failure rate [45]. Historically, pressure and temperature measuring systems exhibited low reliability [45] and, in offshore wells, replacing this equipment once damaged is not a common operation due to the high costs associated with the workover operation and the high environmental risks [46].

According to Bikmukhametov and Jäschke [35], given the intrinsic formulation of the optimization procedure and the computational cost, such models usually give a steady-state response, one point at a time. Moreover, they present limitations such as the high cost of building and maintenance, the need for frequent tuning, as well as the high sensitivity to fluid properties and measurement errors. A full review of first-principles VFMs, which is not the main subject of this article, is provided in detail in [35]. First-principle VFMs are the most commonly used today in commercial VFM systems.

1.2 Well Logs

Well logs are records of physical rock properties obtained by a measurement device along a wellbore [47]. Geological properties deriving from well logs are key data for understanding reservoirs, quantifying in-place volumes, production potential and hydrocarbon reserves. These subsurface data are used by geoscientists, petrophysicists, and engineers to assess reservoir volumes, production potential, and geological features. This subsurface information helps estimate the petrophysical properties to build consistent and accurate 3D reservoir models, an essential tool to predict the behavior of the reservoir and develop profitable projects [48].

However, log data acquisition is expensive and time-consuming, often resulting in missing or incomplete information as these data acquisitions require time and expensive resources, making costs a frequent limitation in obtaining well log data [49]. In addition, for cost-saving policies or operational issues such as borehole damage and collapse, it is common to have incomplete or missing well log data [50].

1.3 Objectives

According to the research of Bikmukhametov and Jäschke [35], the potential of ensemble learning models for VFM application is a promising technique to be explored. Using this statement as motivation, the main goal of this work is to explore the capabilities of multi-model strategies to fill the gap of ensemble learning model applications in the VFM literature. We hope to achieve good results by combining the predictions of heterogeneous models and using their predictions as a new input set to another learning problem, which is the idea behind stacked generalization proposed by Wolpert [51].

For the purpose of comparing the accuracy of stacking ensembles with other benchmark algorithms, we build, train, and evaluate machine learning-based NARX black-box models (nonlinear autoregressive model with exogenous inputs) to predict well flow rates using available field measures such as pressure gauges, surface valve positions, and historical production data, comparing their results with benchmark models. We aim to show also that such an approach is robust for different wells and operating conditions. Therefore, we train and evaluate models over different predicted variables on an actual field dataset with severe transient behavior.

Moreover, we apply the system identification technique of optimizing model order to evaluate this effect on the regressor's performance.

Additionally, we propose to evaluate the use of residual model training as a way of improving the VFM predictions, in what we refer to as a two-step training procedure.

Regarding the synthetic well log models, research in the literature has mainly focused on neural network architectures. Our study means to address two gaps in the field. First, few or no studies have evaluated the decomposition of input and output signals as a feature engineering technique. In addition, ensembles composed of estimators specialized in specific characteristics of the data, such as models specialized in learning low and high-frequency patterns, were also neglected. Thus, in this work, we aim to test the effectiveness of seasonal decomposition acting either as a feature engineering procedure or as an ensemble strategy to improve synthetic well log generation performance.

1.4 Original Contributions

The contributions of this dissertation were developed using the datasets described in chapter 4 as detailed in this chapter. The specific contributions for each work are described below:

1. Virtual Flow Metering Using Stacking Ensemble Techniques and Non-linear System Identification;

   The original contribution of this work is to demonstrate that stacking ensemble models are an innovative paradigm with high potential for VFM applications, as they perform significantly better than other machine learning regressors for the studied cases. The use of stacking ensembles and non-linear system identification techniques led to a remarkable increase in the performance of such models when compared to benchmark algorithms. The proposed architecture improved RMSE metrics by 18.5% to 29% compared to linear models, 5% to 13% compared to non-linear regressors, and 1.3% to 4.5% regarding other ensemble-based regressors.

   Although this work focuses on rate estimation, the method herein proposed can be applied to any measurement device in which system identification can be useful to recover missing data, replace faulty measurement equipment, or handle uncertain measurements. As an example of application, a damaged PDG could be replaced by a virtual model using other measurements as input variables [52].

   As the nature of these methods does not provide an approach to time dependency, we introduced some concepts of system identification, in particular, the model order optimization. The observed results show that

the distinct regressors can achieve their high performance on different lags, which turns the model order into a parameter to be optimized in addition to the hyperparameters of each algorithm.

The objective of this work is shared with most of the previously cited papers, with the main contribution of adding a system identification approach to the problem by enhancing model capacity by exploring the best lag in each algorithm. The development of a Python-implemented class that easily performs time series regression by using any scikit-learn regressors allowed us to easily train and compare models, adding time dependency information for any desired state-of-the-art regression method.

Besides the objectives shared with previously mentioned works, this study also contributes to the literature by providing a free simulation run capable of predicting long-term historic production based on measurements of sensor data and given only some initial conditions. The free simulation run approach is capable of estimating long-term historic production, which can be used to recover or fix incorrect production allocation between wells.

2. Application of Residual Training to Improve the Performance of Virtual Flowmetering Models

   In this study, the use of residual models proved to be a robust technique to imporve the performance of VFM predictors, as a second estimator is trained to predict the errors of the base estimator. This two-step training approach improved the quality of the model's predictions by 1% to 13% depending on the complexity of the base estimator. This contribution is part of a more comprehensive work on feaure engineering techniques for applications in VFM field.

3. Well Logs Estimation Using Multimodel Strategies and Feature Expansion Based on Seasonal Decomposition;

   This work makes a significant contribution to well log estimation by employing two key strategies: Firstly, a seasonal decomposition is proposed as a simple technique to expand the input feature space by breaking the data into trend, seasonal, and residual components. This technique has led to an 8.5% increase in R^2 and a 27% decrease in RMSE for the predictions of DT relative to baseline.

   Subsequently, it details an ensemble-based technique in which two estimators are independently trained to capture the trend and the combined

seasonal and residual terms of the input and output. Of the eight estimators evaluated for the NPHI, DT, and resistivity logs, only two did not show improvement with this method. Moreover, the ensemble models surpassed the base case in all four logs examined. Collectively, these methods outperformed the baseline in all scenarios tested.

Each of the aforementioned contributions was published either as a journal or a conference paper. The references to these publications are the following:

1. **PEREIRA, F. D. C.**; PAULO, P. H. C.; CARVALHO, M. D. S. ; AYALA, H. V. H. **Virtual flow metering using stacking ensemble techniques and nonlinear system identification**. Fuel, 394:134893, 2025.

2. PAULO, P. H. C.; **PEREIRA, F. D. C.**; AYALA, H. V. H. **System Identification Techniques for Soft Sensors and Multiphase Flow Metering**. IFAC-PapersOnLine, v. 58, n. 15, p. 538-543, 2024.

3. **PEREIRA, F. D. C.**; PAULO, P. H. C.; CARVALHO, M. D. S. ; AYALA, H. V. H. **Well Logs Estimation Using Multimodel Strategies and Feature Expansion Based on Seasonal Decomposition**. Proceedings of the Brazilian Congress of Automation (CBA). Rio de Janeiro: SBA. 2024

1.5 Organization

This work is divided into six main chapters.

Chapter 1 is focused on the introduction of this dissertation. It contextualizes machine learning paradigms and presents the main topics of the oil and gas industry used in this dissertation, as well as objectives and a summary of original contributions achieved.

Chapter 2 presents a literature review showing the advances in the fields of virtual flow metering (VFM) and well log estimation.

Chapter 3 is devoted to the methods used in this dissertation. In Section 3.1 theoretical basis of machine learning techniques adopted is presented. Sections 3.2 and 3.3 describe, respectively, the system identification approach for representing a time-series problem and the validation methodology, jointly with the metrics considered.

Chapter 4 is dedicated to presenting the Volve oil field datasets used in this dissertation.

Afterwards, in Chapter 5 the fundamental contributions of this thesis are given, where each section represents a case study with its proposed methodology, results and discussion.

Finally, final remarks and conclusions, as well as suggestions for upcoming future works, are presented in Chapter 6.

2
Literature Review

This chapter is dedicated to the most recent advances in the fields of virtual flow metering (VFM) and well logs estimation with a particular focus on machine learning techniques. It reviews recent research in these areas, highlighting various model architectures, training strategies, and case studies, as well as current limitations, literature gaps, and promising directions for future research.

2.1
Virtual Flowmetering

Since oil and gas companies have increased their amount of investments in digitalization and data acquisition in recent decades [53], the reservoir management area has benefited from machine learning methods that have emerged as powerful tools to address these key challenges [39], serving as a complementary approach to the conventional first-principles method. According to Alakeely et al. [54], these models are powerful tools for tasks where the actual model between the input and output variables is not well defined or does not exist. The difficulty in modeling the dynamic behavior of production well systems is that they present a wide range of operating conditions, making the prediction task a challenge [46, 55]. This happens because the physical phenomena that occur in a certain time window on a well production history are not expected to be repeated over time as the water and gas fractions tend to rise and the pressure levels tend to vary as a consequence of production. Once the models are used under the same operational conditions in which they were trained, data-driven VFM can perform accurate real-time predictions [35].

Data-driven VFMs consist of mathematical models capable of learning relationships between field measurements and production rate information. The process of tuning the model parameters to minimize the error between the predicted and actual data is called training [56]. Bikmukhametov and Jäschke [35] stated that such data-driven models can be deployed at a lower cost when compared to first-principle VFMs, since the modeling of the production system and knowledge about their physical parameters are not necessary. Such characteristics of data-driven VFMs can also be considered as a drawback,

as they present a black-box essence in which the results and parameters are usually not physically explainable [57]. Although there are data-driven models that give a dynamic approach to the problem of rate prediction, most algorithms are formulated in a steady-state way since they respond at the same time that pressure and temperature data are read [35]. Potential prospects of application for Virtual Flowmeters are digital twins for well separators or physical multiphase flowmeters and oilfield real-time monitoring. As potential challenges, we can cite the quality and availability of input data, lack of explainability of black-box models and the constant necessity of retraining the models as the operating conditions vary significantly.

Recently, several works have been published in the field of data-driven virtual flowmeters, evaluating their potential as measurement tools [58]. Mercante and Netto [40] applied deep learning techniques for actual well data, showing that performance increases when using multilayer neural networks that combine layers of MLP (Multi-layer Perceptron), LSTM (Long-Short Term Memory) and GRU (Gate Recurrent Units), although training time also increases for those models. Mercante and Netto [59] also trained common regression techniques and several neural network architectures as three-phase rate predictors using data provided by flow simulator software. The convolutional neural network models were then tested in real data cases, showing good results. Alakeely and Horne [60] proposed a Deep Learning method using surface measurements to predict well flow rates using autoencoders to generate additional input and compare the results with classic Gilbert correlation methods. Sabaa et al. [56] employed downhole electrical submersible pump (ESP), as well as fluid properties and data from the well measurement sensor to predict well flow rates using an MLP model. Manami et al. [58] performed a comparison of static and dynamic ANN architectures to predict two-phase flow rates. MLP-based NARX (nonlinear autoregressive model with exogenous inputs) model outperformed the static MLP.

Liu et al. [61] showed that auto-regressive Echo State Networks parameterized with Genetic Algorithms were predictive in short-term forecasts of three field cases and outperformed other classes of machine learning models. Gryzlov et al. [62] tested three of the most popular architectures of ANNs (Feedforward, Recurrent and Convolutional) to predict production rates on simulated well data and well tests. Bikmukhametov and Jäschke [57] proposed some methods to combine first principle models with data-driven techniques, improving their performance and resulting in more explainable and transparent data-driven models by the addition of physical features. Sandness et al. [36] developed a multitask learning procedure in which the training data was shared between

different wells. Such an approach substantially reduced the prediction error in all tested assets. Gryzlov et al. [63] compared benchmark machine learning models with latent ordinary differential equation (ODE) models in the case of simulated horizontal well production data. Song et al. [53] demonstrated that LSTM required less training data and performed more accurately than MLP and Random Forest in an actual Chinese field case. This work also reported the benefits of transfer learning in data volume requirements and processing time.

Andrade et al. [64] used Data Reconciliation to improve the phenomenological model's accuracy on gas reservoir VFMs. Ma et al. [65] presented a hybrid model combining a mechanical model and a neural network for identifying flow characteristics. The model leverages a distributed virtual flow meter framework which reduces the computational expense of simulation procedures. Regarding the Volve dataset, which we are using as our case study, Paulo et al. [66] provided a study of the influence of different feature engineering approaches, and Huang et al. [67] proposed an innovative Time Patch Dynamic Attention Transformer (TPDAT) achieving remarkable results compared to other algorithms for short-term oil predictions.

Two recent papers have used ensemble models in VFM applications. Al-Qutami et al. [68] achieved higher precision by combining ANN and regression trees (RT) models when comparing their performance with homogeneous ensemble models based on ANN or RT. The number of base learners was estimated using a Simulated Annealing (SA) algorithm, and the combining method was a simple average. AL-Qutami et al. [69] also proposed an ensemble that combines various neural network learners by changing their training sets, architecture, and training parameters. For the learners' optimal selection and best-combining strategy, an adaptive simulated annealing optimization was performed. This innovative method showed better performance compared to bagging and stacking ensembles.

Table 2.1 summarizes recent data-driven VFM publications, showing a wide set of application cases and the use of diverse techniques to increase model performance, but only a few [40, 68, 69] discussed the strategy of combining different models.

As observed in this brief literature analysis, ensemble learning model applications for data-driven virtual flow meters applications are rare, although this technique has been extensively used in other applications [35]. Furthermore, according to the research of Bikmukhametov and Jäschke [35], the potential of ensemble learning models for VFM application is undisclosed and a promising technique to be explored in future research. Moreover, the literature

Table 2.1: Summary of recent VFM data-driven publications.

Model	**Application Scenario**	**Conclusion Remarks**	**Ref., Year**
Data Reconciliation	Gas Reservoir	DR + Physical model increases performance	[64], 2022
Physical + ANN	Simulated Data + Actual Field	computational time improve	[65], 2024
FFT, Residual Model	Volve Dataset	FFT and residual increases model accuracy	[66], 2024
TPDAT	Volve Dataset	TPDAT outperforms LSTM and Transformers	[67], 2024
LSTM, GRU, MLP	UK OGA data - 181 actual wells.	hybrid MLP, LSTM and GRU better results	[40], 2022
Common Regressors, ANN	3-phase rates, simulator/actual well data	Better performance of CNNs	[59], 2023
MLP, Autoencoders	Actual norwegian field data	Autoencoders outperforms Gilbert models	[60], 2021
MLP	Electrical submersible pump (ESP) wells.	MLP is a good predictor in all pumping conditions	[56], 2023
MLP NARX	Gas well many flow regimes	MLP-based NARX better than MLP static approach	[58], 2023
Echo State, Genetic Algortihms	Three field, short term forecast. AR model	ESN+GA outperformed other ML model classes	[61], 2022
ANNs (RNN, FFNN, CNN)	Simulated well data and well tests data	Data-driven models usefull dynamic systems	[62], 2021
LSTM, XGBoost, 1st Principle	First Principle + Data-driven VFM	Explainability by the addition of physical features	[57], 2020
MLP, XGBoost, Multi Task	Actual assets, 55 wells	Shared training improves performance	[36], 2021
Latent ODE, Benchmark	Simulated data from an horizontal well	Latent ODE better but requires more training time	[63], 2022
LSTM, MLP, Random Forests	Two offshore wells in South China	Transfer learning: better performance and less data	[53], 2022
Ensembles, Simulated Annealing	5 years well test data from 8 producer wells	Hybrid models better than homogeneos MLP or RF	[68], 2017
Ensembles, Simulated Annealing	Surface and downhole well test data	Simulated Annealig best aggregation strategy	[69], 2018

lacks the use of long-term free-run simulation models, which can predict the full history of a well's behavior given only an initial condition. A detailed review of VFM research containing the state of the art in this field and its main shortcomings is provided in [35].

The works of Bikmukhametov and Jäschke [35] and Wolpert [51] were the ones that most contributed to this dissertation. The first presented a complete review of the literature on virtual flow metering, pointing to some important gaps in research which we decided to address in this study, while the second provided the approach of stacking ensembles whose effectiveness for virtual flow metering applications deserved to be assessed given the wide use in other fields of application.

Additionally, Al-Qutami et al. [68] and AL-Qutami et al. [69] provided inspiration in some way by making ensemble predictions but neglecting the power of using heterogeneity among base learners. The methods used in these works helped us think about combining the base models of an ensemble in a wiser way.

Finally, Huang et al. [67] also provided some perception on the originality of free simulation runs which we propose in this study. This paper used the same dataset, and the comparison of its results to our findings helped us conclude that the differences in methodologies were quite relevant, as in our case, a free simulation run on the total test set is considered to evaluate performance.

2.2 Synthetic Well Log Generation

Recently, with the increasing capabilities of artificial intelligence technologies, oil and gas companies have been extensively applying digital solutions to achieve greater value. As an industry with an intensive investment profile, upstream companies have increased their efficiency and reduced their costs by applying data-driven solutions [5].

Li and Gao [50] compared the performance of LSTM and RNN models predicting missing parts of sonic transit time (DT), resistivity and porosity (NPHI) logs. The same family of logs was reconstructed by Zhang et al. [70], where cascaded LSTM cells proved to be better predictors than fully connected neural networks. Shi et al. [71] used hybrid models that combine CNN, LSTM and the attention mechanism to predict missing parts of porosity and DT logs.

The attention mechanism was also used with the same CNN-LSTM architecture as a feature extractor in Shan et al. [48] and as a period-skip mechanism in Yang et al. [72]. Lin et al. [73] proposed a missing

well-log transformer (MWLT) based on self-attention instead of a recursive RNN structure. Wu et al. [74] showed that particle swarm optimization (PSO) combined with CNN-LSTM can improve model accuracy and save time searching for hyperparameters. Zheng et al. [75] presented a fusion of LSTM-CNN with seismic data and a physics-informed neural network (PINN) trained to predict actual P-wave measurements, overcoming the limitations of data availability for training and the absence of physical principles in traditional deep learning models. Mehrad et al. [76] used CNN-based hybrid models with optimization algorithms to predict shear wave velocity logs and a combination of multilayer perceptron and genetic algorithms as a high-performance feature extractor. Wang et al. [77] showed that a combination of CNN, bidirectional and gated recurrent unit (BGRU), and the attention mechanism outperforms other deep learning methods when predicting missing well log data from two actual datasets, including a Volve field well. Tatsipie and Sheng [78] used RNN to generate a set of pseudo-well log curves departing only from gamma-ray (GR) log measurements with reasonable accuracy.

Zeng et al. [79] presented an approach of Bayesian uncertainty analysis showing the importance of quantifying uncertainties in well log predictions. Tamoto et al. [80] trained four supervised algorithms on an actual data set from the Santos Basin, Brazil: multilayer perceptron (MLP), Adaptive Boosting, XGBoost, and CatBoost to create synthetic nuclear magnetic resonance (NMR) well logs. A Shapley additive explanations (SHAP) analysis is used to assess the impacts of each input feature. Similar ensemble techniques were applied in Oliveira and Carneiro [81] to predict geochemical logs of a Brazilian pre-salt field. In this study, AdaBoost outperformed all other algorithms. In Dabi et al. [49], the author studies tree-based ensemble methods such as gradient boosting and random forests as density log predictors of an actual North Sea oil field.

3
Methods

This chapter provides an overview of the methods, tools, and procedures employed in this work. It details the machine learning pipeline, the algorithms used, and the application of the system identification approach to the supervised learning problem. Additionally, the processes for model construction and validation are described.

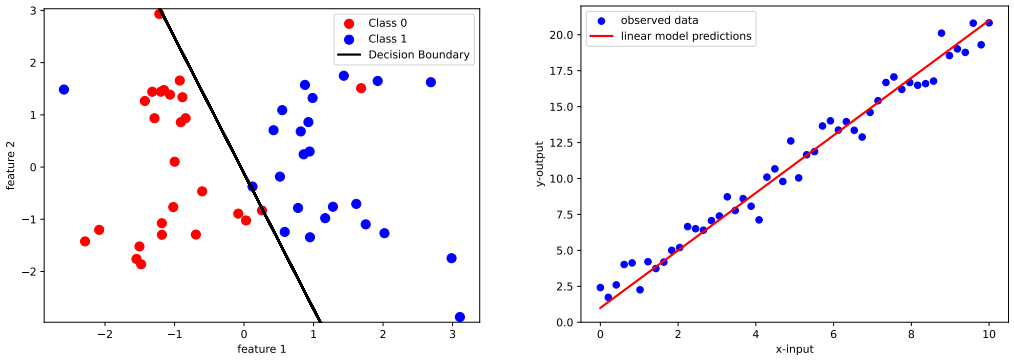
3.1
Machine Learning

Machine Learning (ML) is a subset of Artificial Intelligence (AI) that employs algorithms capable of learning iteratively from data without explicit programming, thereby enhancing their ability to infer from new data. ML algorithms are categorized based on the nature of the data used for training. When using labeled data, with input/output pairs, the approach is known as supervised learning. On the other hand, when algorithms use unlabeled data, consisting solely of inputs, this is called unsupervised learning. Additionally, there is a third category where algorithms learn via trial and error, known as reinforcement learning.

3.1.1
Regression and Classification Problems

Supervised machine learning can be categorized into classification and regression. The nature of the algorithm's output determines whether a task is classified as one or another. A classification problem happens when the desired output is a categorical variable that represents a class. A simplified example of a classification problem is exhibited in Figure 3.1(a). In such a problem, the model tries to map a hyperplane (black line) that separates the samples according to their respective classes. The case depicted refers to a binary classification problem with the input data having only two features.

If the output is continuous, as in the case of a measured quantity, it is called a regression problem. Figure 3.1(b) illustrates a simplified version of a regression problem, where the blue dots indicate the measured value of the output y while the red line shows the predicted values of a linear model.



3.1(a): Classification

3.1(b): Regression

Figure 3.1: Simplified classification (a) and regression (b) supervised machine learning problems. In (a), the black line corresponds to the hyperplane that separates the two classes. In (b), the blue dots correspond to the actual measured output y and the red line represents the linear regressor outcome.

3.1.2 Supervised Machine Learning Algorithms for Regression

Since there is no optimal general-purpose approach to a supervised machine learning problem [82], this work considers several supervised learning algorithms from different families, which are described in the following.

3.1.2.1 Linear Regression

Linear Regression is the regression method where the target value y is expected to be a linear combination of the inputs $x_i, \quad i \in \{1, \cdots, p\}$. Considering $\hat{y}$ the predicted values of the regression:

$$\hat{y} = \omega_0 + \omega_1 x_1 + \cdots + \omega_p x_p \tag{3-1}$$

The vector $\omega = (\omega_1, \cdots, \omega_p)$ represents the coefficients for each one of the inputs and ω_0 the intercept term.

Modifying the input vector to $X = \{1, x_1, \cdots, x_p\}$ to fit the intercept, the algorithm fits a linear model with coefficients $\omega = \{\omega_0, \cdots, \omega_p\}$, including the intercept, to minimize the residual sum of squares between the observed targets y and the targets predicted by the linear approximation $\hat{y}$, in a process also called Ordinary Least Squares (OLS) [4]:

$$\min_{\omega} \|\hat{y} - y\|^2 = \min_{\omega} \|X\omega - y\|^2 \tag{3-2}$$

3.1.2.2 Ridge Regression

Ridge regression addresses some of the problems of OLS regression, such as multicollinearity, overfitting, and high-dimensional datasets. When independent variables are highly correlated, OLS estimates can become unstable and have high variance. Small changes in the data can lead to large changes in the estimated coefficients, making them unreliable [83].

To deal with that issue, Ridge Regression adds a penalty term of the form $\alpha\|\omega\|^2$ to the loss function, which shrinks the coefficients of correlated variables. This shrinkage can stabilize the estimates, leading to more reliable and interpretable results.

The ridge coefficients minimize a penalized residual sum of squares of the form:

$$\min_{\omega} \|X\omega - y\|^2 + \alpha\|\omega\|^2 \tag{3-3}$$

The complexity parameter $\alpha \geq 0$ controls the amount of shrinkage: the larger the value of α, the greater the amount of shrinkage, and thus the coefficients become more robust to collinearity.

3.1.2.3 Polynomial Regression

Polynomial Regression represents a type of regression where the relation between the output y and the inputs is expressed as an nth degree polynomial in X.

Although polynomial regression fits a nonlinear model to the data, it is in fact linear, in the sense that the regression function is linear in the unknown parameters that are estimated from the data. Thus, polynomial regression is a special case of linear regression once it can be formulated as a pipeline in which we first combine the inputs into polynomial terms with degrees less than or equal to the specified degree. For example, if an input sample is two-dimensional $X = \{x_1, x_2\}$, the degree-2 polynomial features used as input terms are $\bar{X} = \{1, x_1, x_2, x_1^2, x_1x_2, x_2^2\}$, and the expression of the estimated values of the regression is given by:

$$\hat{y} = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_1^2 + \omega_4 x_1 x_2 + \omega_5 x_2^2 \tag{3-4}$$

The OLS algorithm is then used to determine the coefficients $\omega = \{\omega_0, \omega_1, \cdots, \omega_5\}$ coefficients by solving:

$$\min_{\omega} \|\bar{X}\omega - y\|^2 \tag{3-5}$$

3.1.2.4 Support Vector Machines

A support vector machine (SVM) designs one or several hyperplanes in a high-dimensional or infinite-dimensional space, which are used for tasks such as classification and regression [4]. The goal of the algorithm is to classify d-dimensional data by identifying a hyperplane that divides the instances into groups belonging to the same class. For any dataset, there are numerous potential hyperplanes. The selected hyperplane could be the one that maximizes the separation from the nearest data points, known as hard margin classification, or it could be the one that minimizes this distance, referred to as soft margin classification. Hard margin classification prioritizes correctly classifying all the training data, which may lead to overfitting, whereas soft margin classification tolerates some level of misclassification of training data. The choice between soft and hard margin is determined by the regularization hyperparameter C [84].

The algorithm solves the optimization problem given by:

$$\min_{\omega,b,\zeta}(\frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n}\zeta_i), \tag{3-6}$$

subject to

$$y_i(\omega^T\phi(x_i) + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, i = 1, \cdots, n$$

where $y_i \in \{-1, 1\}$, w and b parametrize the hyperplane, $C > 0$ is the regularization hyperparameter, ζ_i is the margin distance, and $\phi(x_i)$ is a kernel function that maps x_i into a higher-dimensional space, where the data is separable.

The method can be extended to solve regression problems. Support Vector Regression (SVR) is a type of SVM designed for regression tasks. It extends the principles of SVM, which are primarily used for classification, to predict continuous outcomes by finding a function that approximates the relationship between input features and continuous target values. It does this by defining a margin of tolerance around a predicted value within which errors are considered acceptable. The goal of SVR is to fit the best hyperplane that captures the majority of the data points while minimizing the prediction error outside this margin [84].

3.1.2.5 K-Nearest Neighbors

K-Nearest Neighbors (KNN) regression is a non-parametric, instance-based algorithm used for predicting a continuous target variable based on the

similarity of new data points to their nearest neighbors in the training data. The value assigned to a query point is computed based on the mean of the values of its nearest neighbors [85].

A distance metric is used to determine the proximity of data points in the feature space, usually the Euclidean distance, given by:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{p}(x_{ik} - x_{jk})^2}, x_i \neq x_j \tag{3-7}$$

where the index k denotes the value of the kth feature of the dataset and $\{x_i, x_j\}$ two instances of the training dataset.

The method does not assume any specific functional form or relationship between variables, making it suitable for complex, non-linear relationships [85]. It stores all training data and uses it directly to make predictions, rather than building a general model. The choice of the number of neighbors K is a critical parameter that can be tuned using techniques like cross-validation to optimize performance. For a new data point, the algorithm identifies its K nearest neighbors in the training data based on the chosen distance metric, and the predicted value is typically calculated as the average of the target values of the K nearest neighbors. Some variations use weighted averages, where closer neighbors contribute more to the prediction.

3.1.2.6 Multi-Layer Perceptron

The Multilayer Perceptron (MLP) represents a kind of artificial neural network, serving as a mathematical framework to simulate the workings of the human brain [4]. Generally, an MLP is composed of multiple neuron layers, which include the input layer, one or more hidden layers, and the output layer [86]. Each neuron is described mathematically by:

$$s_k = \phi(b_k + \sum_{j=1}^{m} \omega_{kj} x_j) \tag{3-8}$$

where $\phi(.)$ is the activation function and w_{kj} , b_k, x_j, and s_k are, respectively, the weights, bias, inputs, and output of the k-th neuron. The activation function is the source of the nonlinearity of the model and it can be of many types such as threshold function, the logistic function, the hyperbolic tangent function, and the Rectified Linear Unit function [86].

In an MLP, every layer, with the exception of the output layer, includes a bias neuron and is fully connected to the next layer. Basically, the computation of the output is done by the propagation of the values from the input to the output layer. A node collects the outputs of the nodes in the previous layers,

computes the value defined in Eq. 3-8 and passes this value to every node in the subsequent layer.

The training process consists of finding the weights that minimize the loss function by using gradient descent in a method called back-propagation [86].

3.1.2.7 Decision Trees

Decision Trees (DTs) are a type of non-parametric supervised learning approach utilized for both classification and regression tasks. The primary objective is to develop a model capable of forecasting the value of a target variable by deriving straightforward decision rules based on the data characteristics [87]. A decision tree is structured hierarchically, consisting of the root node, internal nodes, and leaf nodes. Classification happens by tracing a route that begins at the root node, moves through the internal nodes, and concludes at a leaf node which corresponds to a particular class [4].

While DTs are straightforward to interpret and require minimal data pre-processing, they have the potential to form complex trees that may not generalize well. To mitigate this, techniques such as pruning, defining a minimum number of samples required at leaf nodes, or limiting the tree's depth are employed [88]. These parameters should be fine-tuned using methods like cross-validation. Furthermore, DTs can be sensitive to slight data alterations, which might lead to entirely different tree structures. This instability is reduced by integrating decision trees in an ensemble.

A common architecture of DTs is the CART (classification and regression tree), which produces binary trees, meaning each internal node has exactly two child nodes. The algorithm selects the best feature j and the respective threshold t_j to split the dataset based on Gini impurity (for classification) or mean squared error (for regression) [88]. Then, it splits the data into subsets based on the best splits until reaching a stopping criterion. The algorithm also prunes the tree to remove unnecessary nodes that do not contribute significantly to predictive accuracy and avoid overfitting. To make predictions for new data, it uses the majority class (classification) or the average value (regression) of the instances in the leaf nodes.

3.1.2.8 Trees-based Models

Random Forest is an ensemble learning algorithm that is used for both classification and regression tasks. It combines the predictions of multiple trees

to improve accuracy and control overfitting. It uses a technique called bagging (bootstrap aggregating) to create the training datasets for the individual trees. In bagging, multiple subsets of the training data are created by randomly sampling with replacement [89]. Each tree in the forest is trained on a different bootstrap sample, which introduces diversity among the trees.

The main difference between Bagging and Random Forest Regressors is that while the first does not introduce randomness in feature selection, the latter does so by considering a random subset of features at each split in the decision tree [90, 91].

Each decision tree is constructed independently, using the bootstrap samples and the randomly selected subsets of features.

For classification tasks, the final prediction of the Random Forest is made by majority voting among the individual trees. For regression tasks, the average of the predictions from all the trees is taken as the final prediction [90].

Random Forests are, then, less sensitive to noise and overfitting compared to individual decision trees. It often accomplishes high accuracy in both classification and regression tasks due to the ensemble nature and provides explainability on feature importance. On the other hand, the model can be more complex and less interpretable than a single decision tree and more computationally expensive.

Extra Trees, or Extremely Randomized Trees, is another ensemble learning method similar to the Random Forest that adds more randomness in the way the trees are built by using random thresholds for splits [92].

3.1.2.9 Gradient Based Ensembles

Gradient Boosting (GB) is an ensemble learning technique used for regression (and classification) that builds a predictive model in a step-by-step way. It combines the predictions of several weak learners (models that perform slightly better than random guessing, typically limited-depth decision trees) to create a strong predictive model [93].

The idea is to build models sequentially, with each new model attempting to improve the errors made by the previous models, the differences between the predicted values and the actual target values of the previous model.

The term "gradient" in gradient boosting refers to the use of gradient descent to minimize the loss function. After training the initial model, the algorithm computes the gradient of the loss function with respect to the predictions, which indicates the direction in which to adjust the predictions to reduce the error [4].

For each iteration m, a new weak learner $h_m(x)$ is fitted to the negative gradient of the loss function, which represents the errors made by the previous ensemble of models. The predictions are updated as follows:

$$f_m(x) = f_{m-1}(x) + \gamma_m h_m(x) \tag{3-9}$$

where γ_m is a learning rate that controls how much the new tree contributes to the overall prediction. It is a hyperparameter that scales the contribution of each tree. Smaller learning rates often result in better performance but require more iterations (trees) to converge.

Two additional methods, which represent evolutions of the Gradient Boosting (GB), were also used in this work: LightGBM (LGBM) [94] and XGBoost, or Extreme Gradient Boosting (XGB) [95]. The main differences comparing these two algorithms to the first-mentioned method are that while GB grows trees and adds nodes level by level, LGBM grows trees leaf-by-leaf, focusing on the leaf with the maximum gain, and XGB uses a more sophisticated approach to find the best split, whether using leaf- or level-wise approach. Another important difference is regarding the better performance of LGBM and XGB compared to GB, especially for large amounts of data. Whereas LGBM uses a histogram-based approach to speed up training, XGB uses features such as parallel processing to improve training speed.

3.1.3 Data-driven Model Pipeline

A view of the entire data-driven model creation is presented in this subsection. The overall process of ML model creation is given in Fig. 3.2.

The usual pipeline of data-driven models normally incorporates the steps as follows:

i. Data Loading: This step consists of obtaining data, which can be of various types; for example, numerical simulations, experimental design, or real-world data.

ii. Data Pre-processing: the second step of the pipeline is the pre-processing of data. Procedures like filtering, normalizing, scaling, selection, outlier removal, merging, and encoding prepare data for the training to come.

iii. Exploratory Analysis: A very important step to better understand data and find relationships and insights in the dataset. It can be achieved through visualization methods such as histograms, crossplots, and correlation matrices, or more sophisticated methods such as self-organizing maps [96] or t-SNE [97].

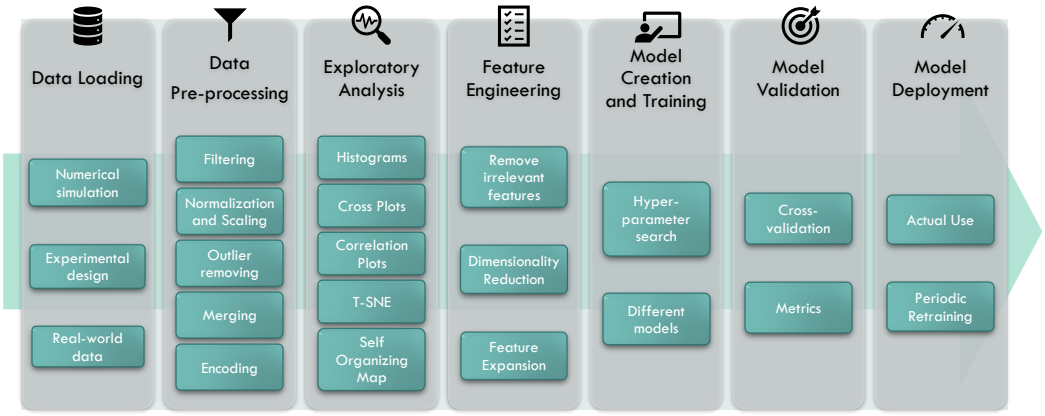


Figure 3.2: General approach for constructing a machine-learning analysis. Any described step can be revisited at any time depending on the on the results of the evaluation of the subsequent steps. The nature of this procedure is iterative.

iv. Feature Engineering: This step refers to the transformation of the data into a new set of variables, ideally reducing the dimensionality and preserving the information present in the original data. An effective feature extraction removes irrelevant and redundant data, improving the learning process and reducing the tendency to overfit by mitigating the curse of dimensionality [4]. Expanding the dimensionality of data to higher-dimensional spaces is also possible.

v. Model Creation and Training: Model creation corresponds to the construction of models capable of performing the desired task. A series of families of specific models should be considered because different data behaves differently according to the models they are fitted with. Implementing hyperparameter optimization methods also must be implemented to find an optimal combination model for the problem. This approach not only enhances the performance of the models but also helps in preventing overfitting and underfitting.

vi. Model Validation: In this step, the models created are tested with unseen data in some manner or adopting any cross-validation strategy. Performance metrics are then evaluated. In case of success, at the end of the current step, an ideal model is found, and the ML model maker can head to the next step.

vii. Deployment of the model: The conclusive phase in developing a data-driven model involves deploying it for everyday use in actual environ-

ments. For successful implementation, the model must be validated using new real-world data originating from the same source as the data it routinely processes. Ensuring that each new data sample can later be used for retraining the model is crucial, as this facilitates enhanced model generalization. This ongoing process is vital for the continual improvement of the machine learning framework.

3.2 System Identification

Black-box system identification is a technique used to find a mathematical relationship between input and output using only available measurements, without prior knowledge regarding the structure, parameters, or physical principles of the model [98, 99].

These models are usually defined in terms of transfer functions and differential equations. Discrete-time systems are characterized by difference equations. The output at a given instant depends on the previous values of the inputs and the past behavior of the system. The linear structure of a discrete-time system is represented as in Equation 3-10 [99]. This model is called ARX (AutoRegressive with eXogenous input).

$$\begin{aligned} y(k) = &-a_1 y(k-1) - \cdots - a_{na} y(k-na) + \\ &+ b_1 u(k-1) + \cdots + b_{nb} y(k-nb) + \xi(k) \end{aligned} \tag{3-10}$$

$y(k)$, $u(k)$ and $\xi(k)$, $(k = 1, 2, ...)$ are the output, input signals and noise term, respectively, and na and nb are integers that correspond to the maximum delay of the output and input signals, respectively.

As a simple example, let us consider $n_a = 3$ and $n_b = 2$ the model lags for the output and the input, respectively, as depicted in Figure 3.3. In this case, the regression task consists of estimating model parameters using filled circles as target values and blank circles as regression terms, where y has a lag of $n_a = 3$ and u has a lag of $n_b = 2$.

In this example, the linear problem of system identification consists of finding the parameters vector $\boldsymbol{\theta}$ that minimizes the error ξ in the equation 3-11. These parameters can be estimated directly using the least-squares algorithm [99].

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \xi \tag{3-11}$$

where $\mathbf{y}$ is the target vector for regression, $\boldsymbol{\theta}$ is the vector of parameters and $\mathbf{X}$ is the regression matrix:

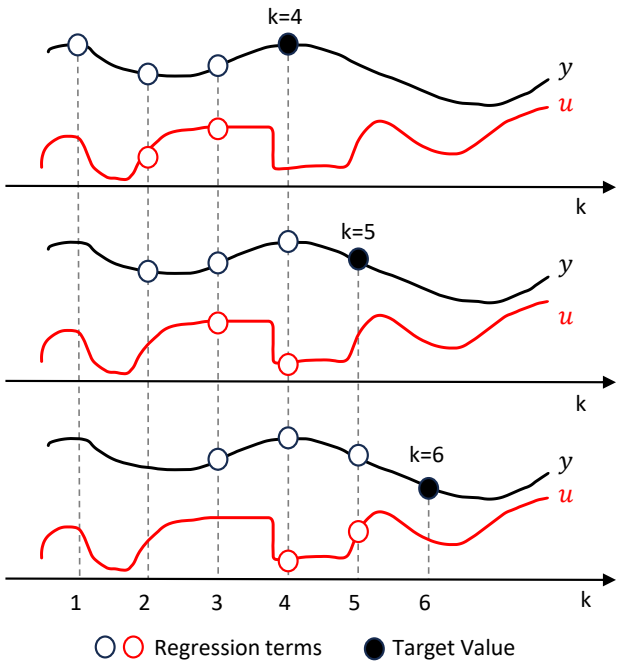


Figure 3.3: Example of discrete-time representation of input(u) and output(y) signals or measurements of a given system.

$$\mathbf{y} = \begin{bmatrix} y(4) \\ y(5) \\ y(6) \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} -y(3) & -y(2) & -y(1) & u(3) & u(2) \\ -y(4) & -y(3) & -y(2) & u(4) & u(3) \\ -y(5) & -y(4) & -y(3) & u(5) & u(4) \end{bmatrix} \quad \boldsymbol{\theta} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \end{bmatrix}$$

With N the sample size and introducing $p = 1 + max(n_a, n_b)$ as the index of the first target value, the general expressions for $\mathbf{y}$ and $\mathbf{X}$ become:

$$\mathbf{y} = \begin{bmatrix} y(p) \\ y(p+1) \\ ... \\ y(N) \end{bmatrix} \tag{3-12}$$

$$\mathbf{X} = \begin{bmatrix} -y(p-1) & ... & -y(p-n_a) & u(p-1) & ... & ...u(p-n_b) \\ -y(p) & ... & -y(p+1-n_a) & u(p) & ... & ...u(p+1-n_b) \\ -y(p+1) & ... & -y(p+2-n_a) & u(p+1) & ... & ...u(p+2-n_b) \\ ... & ... & ... & ... & ... & ... \\ -y(N-2) & ... & -y(N-1-n_a) & u(N-2) & ... & ...u(N-1-n_b) \\ -y(N-1) & ... & -y(N-n_a) & u(N-1) & ... & ...u(N-n_b) \end{bmatrix} \tag{3-13}$$

Non-linear models were introduced by Leontaritis and Billings [100] to deal with a class of systems that linear models could not represent. Non-linear

Auto-Regressive models with eXogenous inputs (NARX) are models in which the relationship between inputs and outputs is modeled by a nonlinear function $f(\cdot)$ [99]:

$$y(k) = f(y(k-1), y(k-2), \ldots, y(k-na), u(k-1), u(k-2), \ldots, u(k-nb)) + \xi(k) \tag{3-14}$$

Given a subset of available data as a training set and considering $f(\cdot)$ as any regression method capable of mapping the regression matrix $\mathbf{X}$ to the corresponding labeled target vector $\mathbf{y}$, we face the machine learning-supervised regression problem: $y = f(X)$.

Once the order of the model (n_a,n_b) is selected, $\mathbf{y}$ and $\mathbf{X}$ are built according to equations 3-12 and 3-13 and then any regression method can be used to create a model. After defining the structure of the system, which corresponds to choosing a machine learning algorithm corresponding to $f(\cdot)$, the tuning of its hyperparameters is obtained by training.

In equations 3-10 and 3-14, the architecture of the system is a single input and a single output (SISO), although the systems can have one or more inputs and outputs. In our case, five inputs were used to provide estimations for the output as described in Table 4.1. The system we describe in this case is then a multiple input and single output (MISO). In this case, the regression matrix $\mathbf{X}$ has more terms, and a new lag has to be introduced for each new input.

Several functions can be trained to obtain the behavior of a producer well: classical approaches such as linear regressor (ARX model) and polynomial regressor (Polynomial NARX model) or any other sophisticated machine learning algorithm. In the latter case, the models are called machine-learning-based NARX models.

3.3 Model Construction and Validation

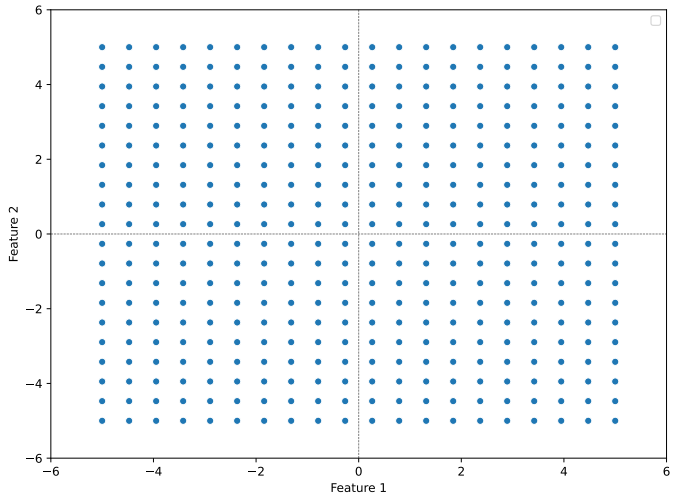
3.3.1 Hyperparameter Tuning with k-fold Cross Validation

Hyperparameter tuning is a crucial step in the machine learning model development process. It involves finding the optimal set of hyperparameters that govern the behavior of a machine learning algorithm [4]. Hyperparameters are different from model parameters as they are set before the training process and can significantly affect the model's performance.

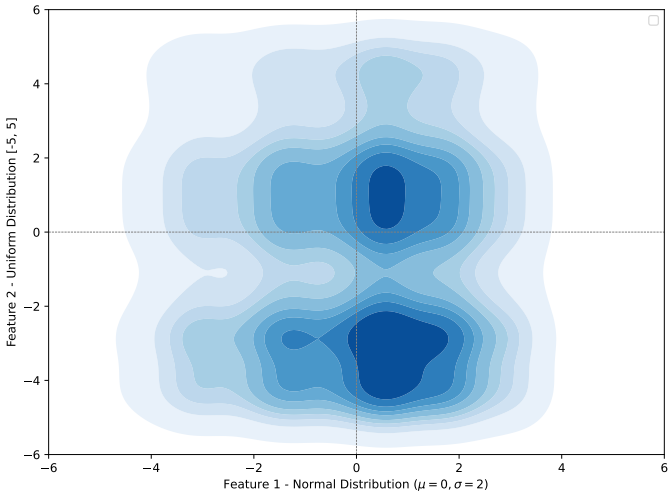
In hyperparameter search, the first step is to identify which hyperparameters to optimize. Then, a range or a set of values for each hyperparameter is

specified. This can be done using techniques such as grid search or randomized search.

While grid search involves specifying a grid of hyperparameter values and evaluating every combination, randomized search randomly samples from a range of hyperparameter values and according to some distribution of probabilities, which can be more efficient than grid search. Both processes are illustrated in Figure 3.4. In Figure 3.4(a), a grid search is performed by combining all the possible combined values of features 1 and 2, and in Figure 3.4(b) the combinations are sampled considering, as an exemple, feature 1 as hypothetically uniform in the interval $[-5, 5]$ and feature 2 as respecting a Normal distribution of $\mu = 0$ and $\sigma = 2$.



3.4(a): Grid seach with all cobinations of the two features.



3.4(b): Region in the input space of features 1 and 2 where points sampled according to their conjoint distributions.

Figure 3.4: Hyperparameter search in the input space of features, where (a) represents a grid search and (b) a randomized search.

k-Fold Cross-Validation is a technique used to assess the performance of a model and to help in hyperparameter tuning. It involves partitioning the training dataset into k subsets (folds). The model is trained on $k-1$ of these folds and validated on the remaining fold. This process is repeated k times, with each fold serving as the validation set once. The results from all iterations are averaged to produce a single performance metric.

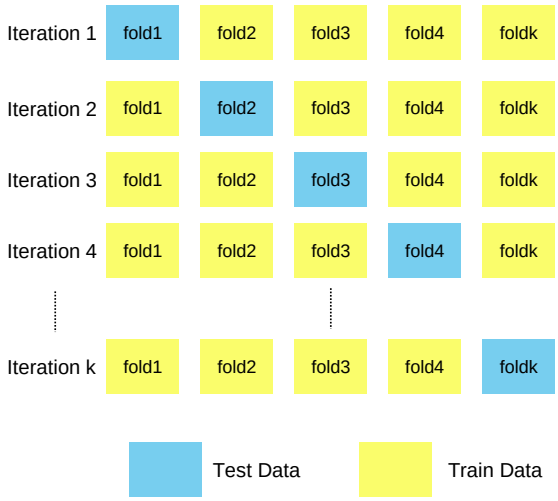


Figure 3.5: Schematic representation of the k-fold cross-validation procedure. The dataset is divided into k parts, and at each iteration $k-1$ folds are used for training while 1 fold is kept for checking the model performance.

Combining hyperparameter search with k-fold cross-validation means to evaluate all the possible combinations of the hyperparameters previously generated. For each realization of hyperparameters, we compute the metrics considering their average performance in all folds and finally select the best combination of hyperparameters that yields the best average performance.

By validating the model on multiple subsets of the training data, k-fold cross-validation provides a more reliable estimate of the model's performance and helps prevent overfitting to the training data as it avoids bias associated with insufficient data sampling. The averaging of results helps to smooth out variability in performance estimates that could arise from a single train-test split.

Hyperparameter tuning with k-fold cross-validation is a powerful technique for optimizing machine learning models. It provides a systematic way to evaluate different hyperparameter configurations while ensuring that the model's performance is robust and generalizes well to unseen data. By carefully selecting hyperparameters, models can significantly improve their performance.

3.3.2 Evaluation Metrics

The models were evaluated using two statistical quantities: the coefficient of determination (R^2) and the root mean squared error (RMSE). The mathematical formulas for the measurements mentioned are shown in equations 3-15 and 3-16, respectively, where $\hat{\mathbf{y}}_i$ and $\mathbf{y}_i$ refer to the predicted values of the output and actual observed values, respectively. The coefficient of determination (R^2) closer to 1 and RMSE closer to 0 indicate better model performance.

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2} \tag{3-15}$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{3-16}$$

4
Case Studies - Volve Field

The Volve field is a decommissioned oil field in the North Sea discovered in 1993 and operated by Equinor from 2008 to 2016. Producing from sandstones within the late middle to early upper Jurassic Hugin formation, with reservoir depths ranging from 2700 to 3100 meters, the Volve field has water depths of 80 meters on average and the western structure is heavily faulted, causing uncertain fault communication [101, 102, 103]. The production development was approved in 2005, consisting of a jack-up processing and drilling unit for the development of the field and the 'Navion Saga' vessel for the storage of stabilized oil [104].

In 2018, operator Equinor revealed a dataset [105] of the Volve field containing information on production, well design and completion, seismic data, well logs (petrophysical and drilling), geological and stratigraphical data, static and dynamic models, as well as surface and grid data [105].

This disclosure of the field data aims to provide the scientific community with actual data that can be utilized for multidisciplinary studies and the development, testing, and training of machine learning models.

4.1
Production Data

The Volve dataset contains production from 6 producers and 1 injector well, which includes dynamic data from pressure and temperature sensors, choke sizes, and production flow rates. Some of the well variables and their definitions are listed in the table 4.1.

For this study, three wells were selected because of their rich dynamic behavior: 15/9-F-1C, 15/9-F-15D, and 15/9-F-14. The data set containing the production history of well 15/9-F-1C is composed of 743 data points containing a complete set of measurement data, as well as distinct operational conditions throughout its operating life, namely: (i) high initial production followed by decline, (ii) rich transient behavior imposed by 15 shut-ins and re-openings, possible due to wellbore interventions to restore productivity, (iii) watercut ranging from 0 to 65%. These characteristics provide a robust dataset to validate the generalizability of the models. It is worth mentioning that this

Table 4.1: Volve dataset available production variables

Variable	**Definition**	**Unit**
BORE_OIL_VOL	Average oil flow rate	m^3/d
BORE_WAT_VOL	Average water flow rate	m^3/d
BORE_GAS_VOL	Average gas flow rate	m^3/d
BORE_LIQ_VOL	Average liquid flow rate	m^3/d
AVG_DOWNHOLE _PRES	Average downhole pressure	bar
AVG_DOWNHOLE _TEMP	Average downhole temperature	°C
AVG_WHP_P	Average wellhead pressure	bar
AVG_WHT_P	Average wellhead temperature	°C
AVG_CHOKE _SIZE_P	Average choke valve opening position	%
DP_CHOKE_SIZE	Average pressure drop caused by the choke	bar
AVG_DP_TUBING	Wellhead-Downhole Pressure difference	bar

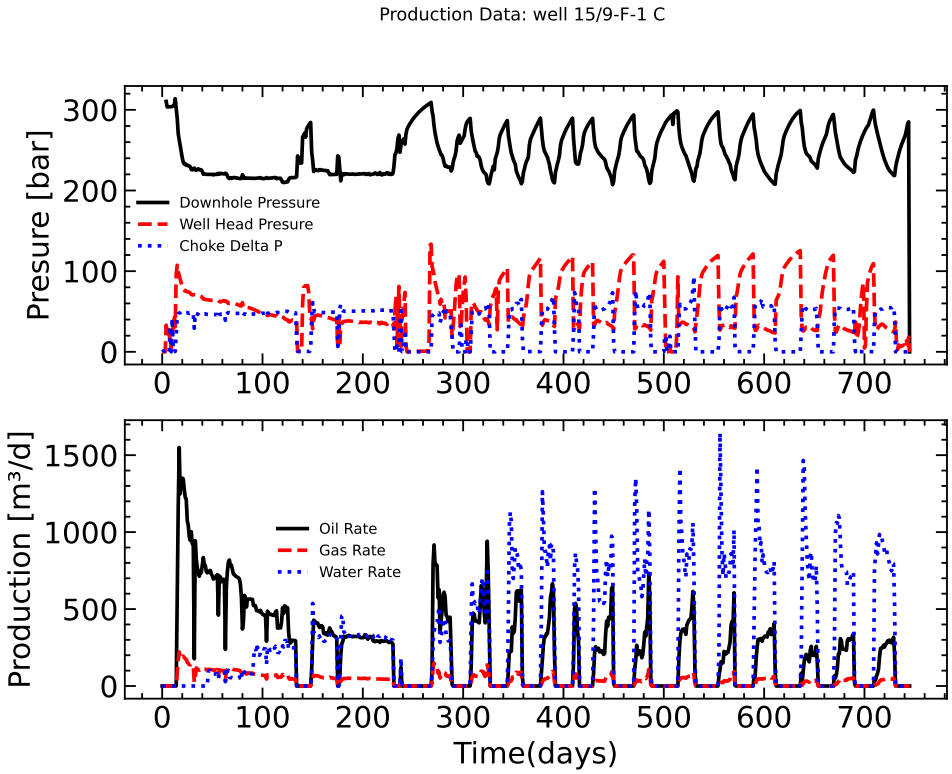
well was also selected as representative by Li et al. [106] to predict downhole pressure using deep learning techniques.

Well 15/9-F-15D dataset contains 978 datapoints. In its historic production, we observe a steady decline in oil rate and a sudden increase in production potential at about 600 days, maybe due to a rig intervention. After this period, the well is reopened with higher pressure and about 50% water cut, which also represents an important change in its operating life.

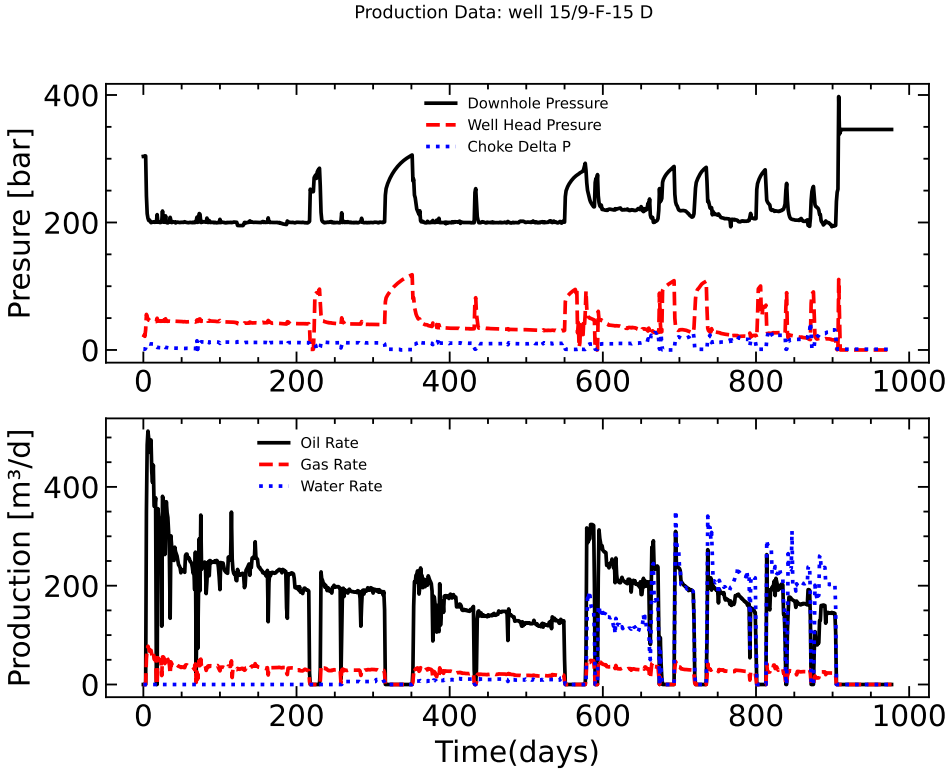
Well 15/9-F-14 has 3056 datapoints where we can observe an oil rate decline probably associated with the water breakthrough. It's also remarkable the increase in the downhole pressure, which can be associated with the water injection recovery mechanism.

In the three wells, we evaluated different production rates: namely the liquid production rate for well 15/9-F-1C, the oil production rate for well 15/9-F-15D and the gas production rate for well 15/9-F-14. The pressure measurements and production data from the three listed wells can be visualized in Figure 4.1.

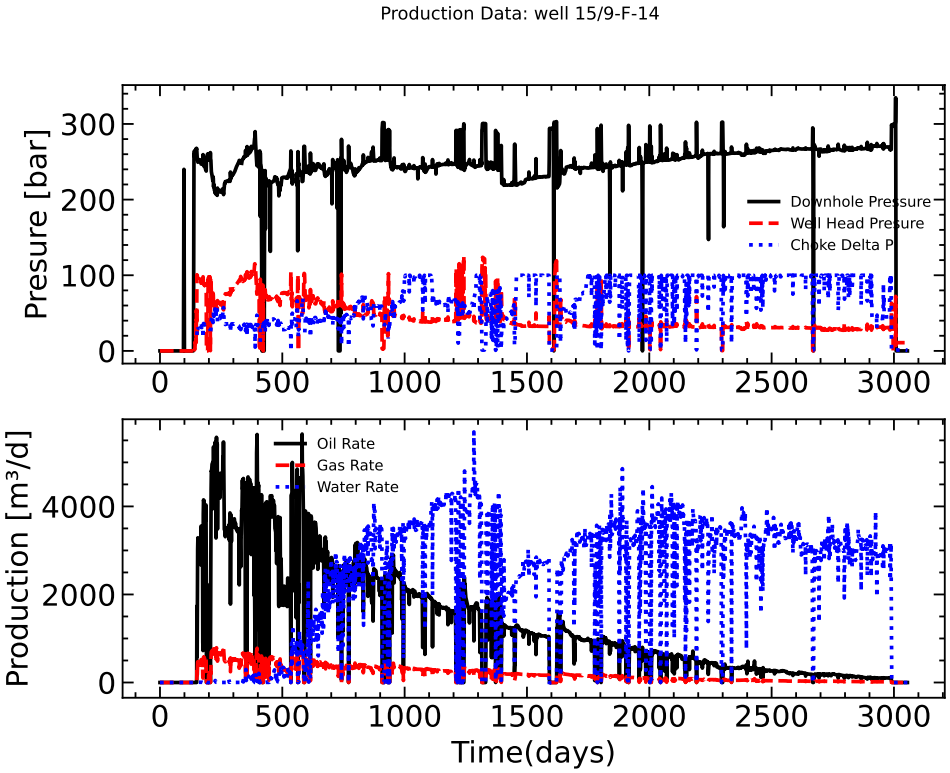
It is also interesting to note that, since the variables are all daily averages, the discrete-time approach is valid, and the time series has a constant time step of one day. Additionally, the dynamic behavior of production wells can be considered as stationary with varying boundary conditions [107].



4.1(a): Well 15/9-F-1C



4.1(b): Well 15/9-F-15D



4.1(c): Well 15/9-F-14

Figure 4.1: Production and measurement data from Volve wells

4.2 Well Logs Data

The volve field dataset provides logs of five wells, and three have the complete set described in Table 4.2. In this study, we used 15/9-F-1A and 15/9-F-11A as training wells and 15/9-F-1B as the test well.

Table 4.2: Well log curves available in the Volve logs dataset.

Acronym	**Unit**	**Log**
CALI	in	Caliper
DT,DTS	μs/ft	Compressional/Shear Slowness
GR	api	Gamma Ray
NPHI	v/v	Neutron Porosity
RACEHM	ohm.m	Resistivity (High Freq. Atten)
RACELM	ohm.m	Resistivity (Low Freq. Atten)
RPCEHM	ohm.m	Resistivity (High Freq. Phase)
RPCELM	ohm.m	Resistivity (Low Freq. Phase)
RHOB	g/cm^3	Bulk Density
PHIF	v/v	Final Porosity
SW	v/v	Water Saturation
VSH	v/v	Shale Volume

The set of selected variables used in the study is plotted in Figure 4.2 for well 15/9-F-1B.

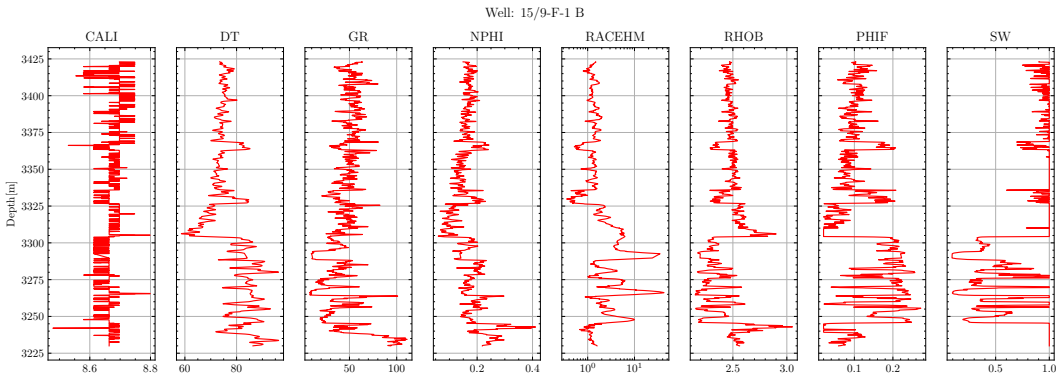


Figure 4.2: Well logs available in the Volve dataset - well 15/9-F-1B.

5
Contributions

This chapter details the main contributions of this thesis. Each section corresponds to a specific contribution, presenting its methodology, graphical abstract, key results, and related discussions.

5.1
Virtual Flow Metering Using Stacking Ensemble Techniques and Nonlinear System Identification

In petroleum production systems, data-driven virtual flow meters offer a viable option to physical hardware or first-principle models for predicting multiphase flow rates Bikmukhametov and Jäschke [35]. Although the existing literature examines various techniques, it overlooks multi-model strategies and system identification methods. This study addresses these gaps in research by employing stacked generalization models and black-box system identification to improve production rate predictions.

In this section, we developed machine learning models to predict the production rates of three actual wells of the Volve Field, based on well sensor measurements. The use of stacking ensembles and non-linear system identification techniques led to a remarkable increase in the performance of such models when compared to benchmark algorithms. The proposed architecture improved RMSE metrics by 18.5% to 29% compared to linear models, 5% to 13% compared to non-linear regressors, and 1.3% to 4.5% regarding other ensemble-based regressors.

System identification emerged as an effective method for evaluating optimal lags and temporal dependencies, while stacking ensembles improved well-rate prediction by merging different base learners, which can perform better for different instances of data. Additionally, free simulation runs predict long-term historical production based only on initial conditions. The strategy proposed in this study increases the reliability of VFM models by stacking various base learners and optimizing the model order, presenting a novel approach for the design of data-driven VFM estimators.

This work was published in the scientific journal FUEL, and the reference to this publication can be found in Section 1.4. The repository containing the

Python source files related to this work is available at [108].

5.1.1 Proposed Method

This study aims to demonstrate that stacking a variety of regression algorithms, from simple linear models to complex architectures, can improve the predictions of production well's dynamic behavior. This approach, known as stacking generalization, uses the predictions from different models as inputs to a new regression task.

Additionally, the system identification concepts herein applied, such as model order optimization, indicate that different models perform optimally with different lags, making the model order a key parameter alongside the hyperparameters of each algorithm. Moreover, using a free simulation run to evaluate model performance allows for long-term predictions using initial conditions and measured inputs.

The pipeline performed in this work is shown in figure 5.1. To build the regression matrices using the chosen lags and perform fit and predict methods, a Python class was written to allow the use of any regressor of the scikit-learn (sklearn) package [91]. The class also gives support for free-run simulations (FS), where simulated outputs are used as input for the prediction of the next step, and one-step ahead (OSA) simulations (where each predicted step uses only real measurement values as input). The model object is then a system identification model composed of the lags (n_a, n_{bi}, $i = \{1...5\}$) and an instance of a scikit-learn regressor corresponding to the function $f(\cdot)$, as described in Section 3.2.

As a pre-processing step, we used min-max standardization to ensure that all the variables in the regression contribute in similar weights, preventing issues related to the magnitude and the different physical values of the measurements. The min-max standardization is defined by equation 5-1:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5-1}$$

Where x' is the standardized input, x is the original input, and x_{min} and x_{max} are its minimal and maximum values, respectively.

When dealing with machine learning, a good practice to avoid overfitting and save computational time is to reduce dimensionality by selecting the most relevant variables. A simple approach is to keep only the most correlated variables with the response and to eliminate redundant variables. In our case, the correlation matrix showed different importance of variables depending on the selected well due to their unique operational characteristics. Thus, we have

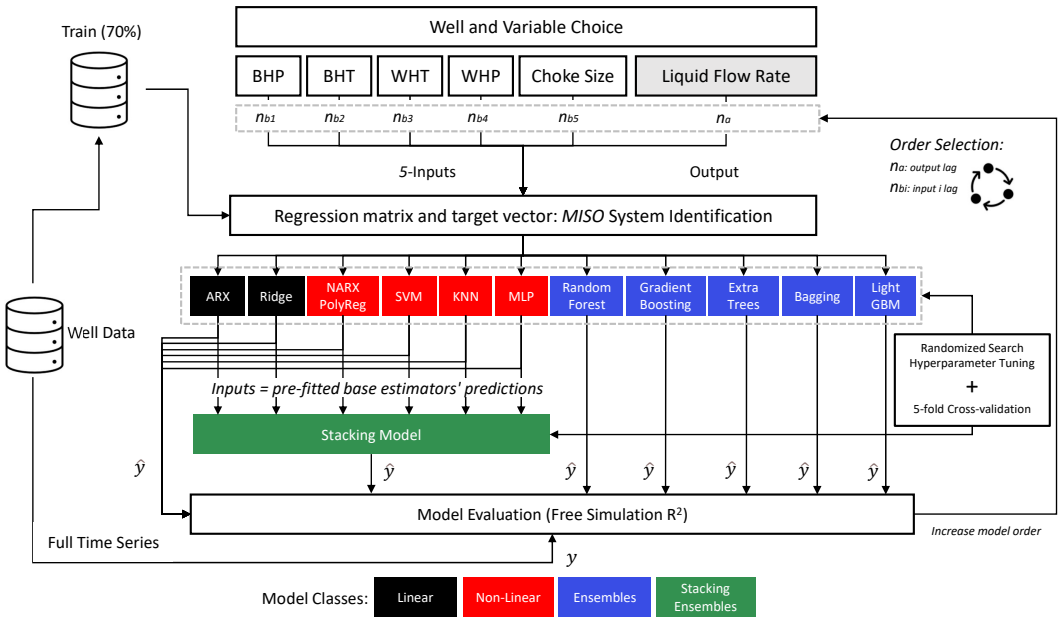


Figure 5.1: Graphical Abstract describing the main steps of the workflow.

chosen the standard set of inputs used by reservoir and production engineers in reservoir management and in first principle models, namely pressure and temperature measured in the downhole and well head, as well as the surface choke valve opening.

To perform the regression tasks, different algorithms were used. From the No-Free-Lunch theorems statement that there is no optimal general-purpose approach to learning [82] comes the importance of testing a variety of the most popular and available algorithms for regression problems [109]. The models evaluated to perform regression are summarized in Table 5.1. Linear Regressor (LinReg) and Polynomial Regressor (PolyReg) are equivalent to the classical ARX and polynomial NARX methods used in system identification. For the PolyReg model, a polynomial degree of two was assumed.

Since all considered regressors have hyperparameters that must be optimized, a randomized search approach with k-fold cross-validation was adopted. Cross-validation is an important step in machine learning pipelines. This technique consists of splitting the training data into several sets of train-validation splits. Each combination of hyperparameters is trained and evaluated in each of these subsets. This procedure avoids bias and overfitting associated with data sampling as the models are more capable of generalizing well once their performance has been validated on unseen data during the training procedure. The best hyperparameter combination is the one that performs better, on average,

Table 5.1: Regressors models used in the system identification.

Model	**Description**	**Classification**
LinReg	Linear Regression [4]	Linear
RidgeReg	Ridge Regression [83]	
PolyReg	Polynomial Regression (n=2)	Non Linear
SVM	Support Vector Machine Regressor [84]	
KNN	K-Nearest Neighbors Regressor [85]	
MLP	Multi-Layer Perceptron [86]	
RandomForest	Random Forest Regressor [90]	Ensemble
GradientBoosting	Gradient Boosting Regressor [93]	
ExtraTrees	Extra Trees Regressor [92]	
BaggingRegressor	Bagging Regressor [89]	
LightGBM	Light GBM Regressor [94]	
Stacking	Stacking Regressor [51]	Stacking Ensemble

over the 50 splits. This work used a 5-fold and 10-repetition cross-validation and a randomized search performing 100 combinations of hyperparameters.

As described in Figure 5.1 a loop was also performed through model orders ranging from 2 to 20 to investigate the lag dependency for each model and select optimal model orders.

The production history of the selected wells was divided into a train set composed of the first 70% of the production history. The test set was considered to be the whole data set.

5.1.2 Stacking Ensembles

Stacking Ensembles, originally presented by Wolpert [51] as Stacked Generalization, is a method to improve model performance by combining predictions from base learners, which are also referred to as base models or level-0 learners. The approach consists of training a second-level learner on a new learning problem using the base learner's predictions as inputs. This second-level model is also called a meta-learner, meta-model, or level-1 learner. In our case, since the base learners used had already been pre-fitted, the use of cross-validation in the process of training the meta-regressors is unnecessary [91]. Wolpert describes stacking generalization as a "black art" as there is no recipe for choosing which base learners or meta-learners can perform well for a given task [51]. The stacking process is depicted in Figure 5.2.

As depicted in Figure 5.1, in the specific case of stacking models, only linear and non-linear models were used as base learners. We discuss the reasons in the results section of this chapter. All the classes of regressors were tested as

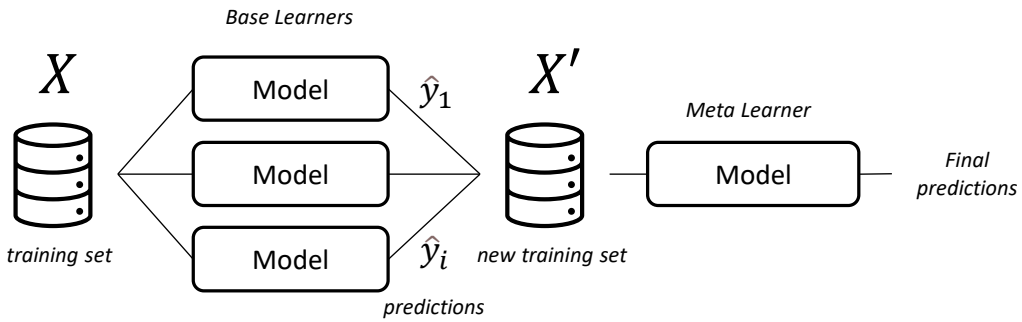


Figure 5.2: Stacked Genaralization

base learners or meta-regressors to evaluate which of them performed better in our case. Table 5.2 lists the algorithms that performed better as meta-regressors.

Table 5.2: Meta Regressors for the Stacking Ensemble models.

Meta Model	**Description**
RidgeReg	Ridge Regression
PolyReg	Polynomial Regression (n=2)
MLP	Multi-Layer Perceptron
RandomForest	Random Forest Regressor
GradientBoosting	Gradient Boosting Regressor
ExtraTrees	Extra Trees Regressor
BaggingRegressor	Bagging Regressor
LightGBM	Light GBM Regressor
XGBoost	Extreme Gradient Boosting Regressor [95]

5.1.3 Results

In this section, we present the results obtained for the selected models using the approach proposed in Section 5.1.1. The results are reported as free simulations (FS) for each regressor. For each estimator, the FS results are compared with the actual data to qualitatively display the adherence of the models to the problem they aim to represent. This is one of the reasons why we chose to set the test data to be the whole data series, as it would be more robust to have a long-term forecast to evaluate model accuracy.

The selected exogenous inputs were pressure and temperature measured in the downhole and well head, as well as the surface choke valve opening. We present in detail the results for well 15/9F-1C, as the other wells obtained similar results.

All models were trained with lags ranging from 2 to 20 days to assess the dependence of model quality on its order. These delays seem reasonable considering that production and sensor data are reported daily. For each of these lags, we performed a randomized search to select the best hyperparameters for each model. On the randomized search performed for the LinReg and PolyReg models, the only parameter adjusted was the 'positive' parameter from the sklearn LinearRegressor API [110]. For the set of tested regressors (Table 5.1), the optimized parameters, their range of values, and their distribution are listed in Table 5.3. The best hyperparameters found for each estimator are also presented in this section.

Table 5.3: Parameters used in the randomized search cross validation and their ranges, values and distributions

Model	**Parameter: Values**	**Distribution**
LinReg, RidgeReg	'positive': [True, False]	-
PolyReg	'Reg__positive': [True, False]	-
SVM	'C': (1e-3, 1e3), 'kernel': ['poly', 'rbf', 'sigmoid','linear']	log uniform
	'degree': (2,6), 'gamma': (1e-4, 1e0)	random int, log uniform
KNN	'n_neighbors': (1,100), 'weights': ['uniform', 'distance']	random int
MLP	'activation': ['relu', 'tanh'], 'early_stopping': [True]	-
	'learning_rate_init': (1e-4, 0.01)	log uniform
	'alpha': (1e-5, 1e-3), 'tol':(1e-7,1e-1)	log uniform
	'hidden_layers': [1,2,3,4], 'hidden_units':[20,60,100]	-
RandomForest	'n_estimators': (5,100), 'max_depth': (1,20)	random int
	'min_samples_split': (2,20), 'min_samples_leaf': (1,20)	random int
	'max_features': (0,1), 'max_leaf_nodes': (10,150)	uniform, random int
GradientBoosting	'n_estimators': (5,100), 'max_leaf_nodes': (10,150)	random int
	'loss': ['squared_error', 'absolute_error', 'huber', 'quantile']	-
	'learning_rate': (1e-5, 1e1)	log uniform
	'criterion': ['friedman_mse',' squared_error']	-
	'subsample': (0,1), 'max_features': (0,1)	uniform
	'max_depth': (1,20), 'min_samples_split': (2,20)	random int
	'min_samples_leaf': (1,20)	random int
ExtraTrees	'n_estimators': (5,100), 'max_depth': (1,20)	random int
	'criterion': ['squared_error', 'absolute_error'],	-
	'min_samples_split': (2,20), 'min_samples_leaf': (1,20)	random int
	'max_features': (0,1)	uniform
BaggingRegressor	'n_estimators': (5,100)	random int
	'max_samples': (0,1), 'max_features': (0,1)	uniform
	'bootstrap': [True, False]	-
LightGBM	'num_leaves': (5,100), 'n_estimators': (10,200)	random int
	'learning_rate': (1e-3, 1)	log uniform
Stacking	cv: 'prefit'	-

The results of the model performance as a function of the selected lag, in terms of R^2 and $RMSE$, are illustrated in figures 5.3 and 5.4, respectively.

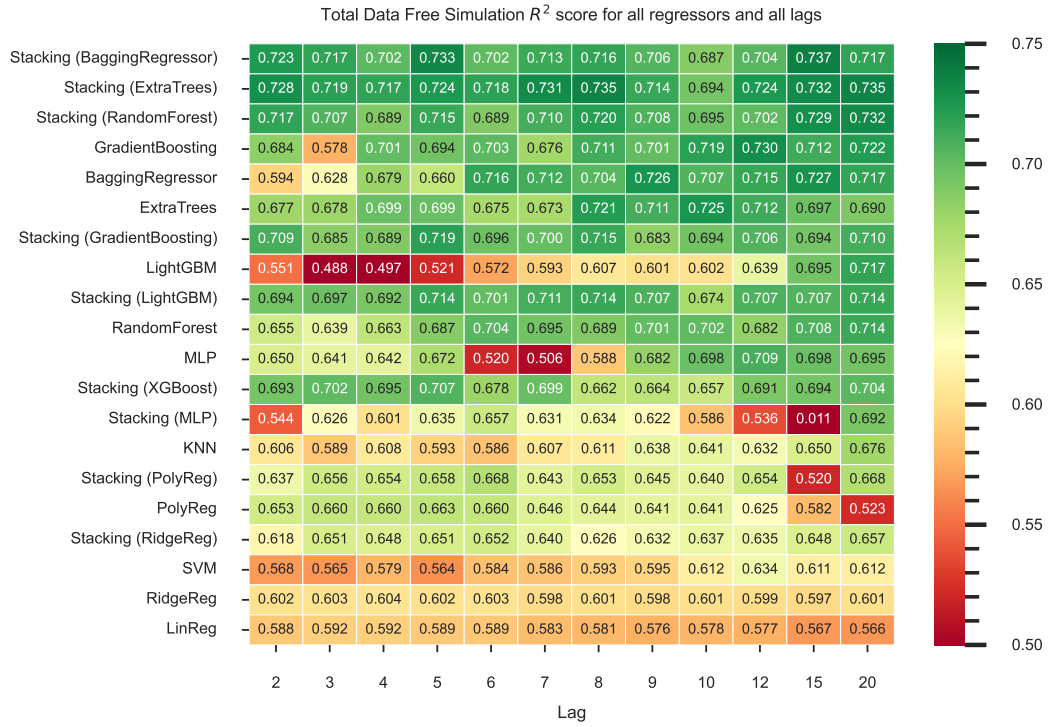


Figure 5.3: Heatmap of R^2 by model and lag (15/F9-1C)

Figure 5.4: Heatmap of $RMSE$ by model and lag (15/F9-1C)

Although one-step-ahead predictions are useful for some sets of applications, we focus on FS values of R^2 and RMSE once this kind of prediction is more reasonable for long-term applications, as the models need only an initial condition to perform a long-term simulation. The results are then presented for both the train set (first 70% data points) and the total time series (test data). The final models are ranked according to their FS R^2 scores for the total data.

Figures 5.3 and 5.4 show a considerable difference in the behavior of the models with respect to their order. In general, models benefit from higher lags once they add time-dependent information, except for linear regressors and PolyReg regressors, in which increasing order worsens the performance of the models. One possible explanation for this behavior is that linear models are too simple to deal with a larger range of delayed series values. Thus, for these simple approaches, increasing the order of the model does not have a positive impact. In contrast, more sophisticated algorithms tend to respond well to an increase in model lag. Still, concerning the more simple model architectures, Figure 5.5 shows that a simple non-linear approach, such as a polynomial expansion of degree 2 (combining the entries by multiplying them two by two and then performing a linear regression), can increase model quality significantly. In this case, the decrease in RMSE is about 7.7% compared to the RidgeReg linear regression.

Figure 5.5 also shows that nonlinear regressors present a wider range of R^2, depending on the order. The model order is, then, an additional hyperparameter to be tuned. In fact, the analysis of Figure 5.4 shows that the variation in the RMSE metric considering the best and worst lag differs considerably for each model, from 3% (linear models) to 79% (Stacking+MLP).

As described in section 5.1.2, stacking ensemble modeling consists of training a meta regressor (also referred to as 'final regressor') to fit observed data using base estimators' predictions as input. As the stacking regressor base models had previously been trained using 5-fold cross-validation, the meta-regressors of the stacking ensemble were trained using only the pre-fitted base estimators' predictions as input, eliminating the use of cross-validation at this step [51],[91]. Thus, the original input data was used only to train the base learners (see Figure 5.1).

The results reported in this section consider stacking ensembles that used only linear and nonlinear models as base estimators, since the use of other types of ensembles as base learners led to a decrease in model quality.

Approximately 12.4% decrease in $RMSE$ and 12.2% increase in R^2 is reported in Table 5.4 between the best stacking ensemble architectures (#1)

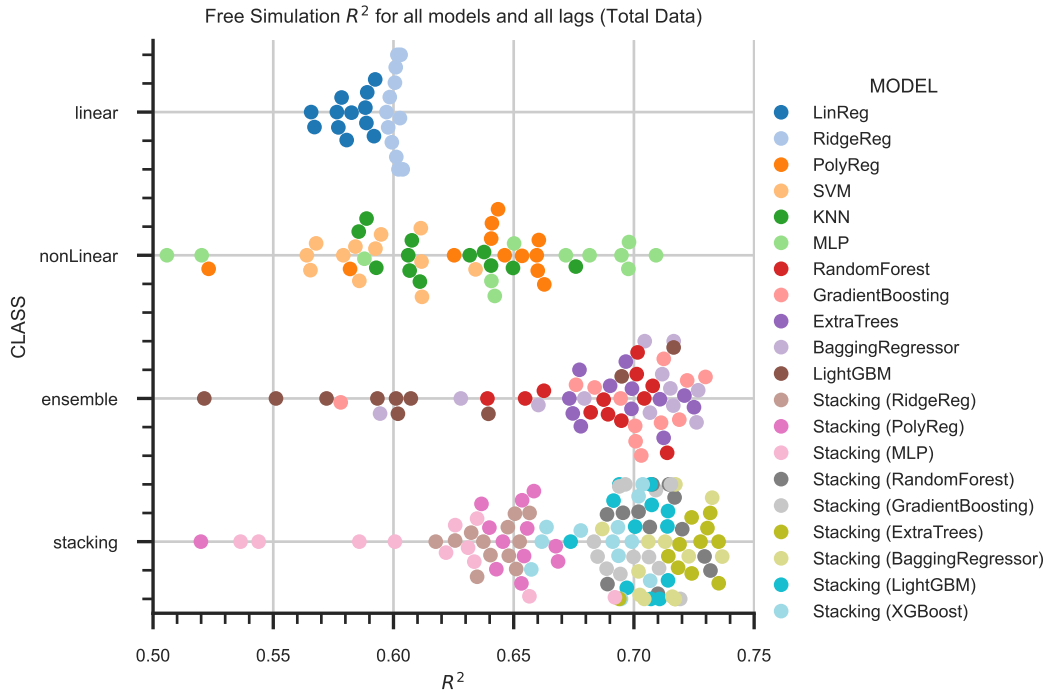


Figure 5.5: Performance of all models by class (all lags - 15/F9-1C)

and the worst (#17), showing that the performance of these models is very dependent on the meta-regressor algorithm to combine predictions from base learners.

The stacking ensemble strategy of combining linear and non-linear base learners to boost performance appears to be effective according to Figure 5.5. An interesting point is that MLPs, RidgeReg, and PolyReg present poor performance when used as stacking ensemble metamodels when compared to ensembles that use other ensembles such as Bagging and Boosting techniques as metaregressors. In this case study, Bagging and Boosting models (here called 'ensemble' class) disturbed the performance of stacking models when used as base learners, but improved model quality when playing the role of meta-regressors, due to their capabilities of better combining the predictions from base learners. In this study, our stacking models achieved better results with more complex meta-regressors.

The parameters of the metaregressors of the stacking ensembles (Table 5.2) were also fine-tuned using a randomized search with the same ranges and distributions specified in Table 5.3. For this specific case, the models were labeled as 'Stacking($regressor$)'. For example, 'Stacking($BaggingRegressor$)' means a stacking ensemble using linear and nonlinear base models and a BaggingRegressor object as a metaregressor.

For each model evaluated, Table 5.4 shows the main scores, the best lag,

the mean fit time of the random search, and the relative improvement $RMSE$. Models are ranked in table 5.4 according to their FS R^2 score for the prediction of the total data.

Table 5.4: Metric results for all the selected models, ranking based on R^2 Free Run Simulation over the total data and best lag for each model (well 15/F9-1C).

#	Model Class	Model (Regressor)	Best Lag	Train Data R^2 FS	Train Data RMSE FS	Total Data R^2 FS	Total Data RMSE FS	Mean fit time (ms)	% RMSE Improve Best/Current
1	stacking	Stacking (BaggingRegressor)	15	0.7853	0.1223	0.7367	0.1403	126	-
2	stacking	Stacking (ExtraTrees)	20	0.7759	0.1249	0.7353	0.1407	83	0.3
3	stacking	Stacking (RandomForest)	20	0.7696	0.1267	0.7318	0.1416	94	0.9
4	ensemble	GradientBoosting	12	0.7553	0.1305	0.7297	0.1422	426	1.3
5	ensemble	BaggingRegressor	15	0.7618	0.1288	0.7267	0.1429	528	1.8
6	ensemble	ExtraTrees	10	0.7667	0.1275	0.7250	0.1434	222	2.2
7	stacking	Stacking (GradientBoosting)	5	0.7569	0.1301	0.7193	0.1449	228	3.2
8	ensemble	LightGBM	20	0.7387	0.1349	0.7165	0.1456	215	3.6
9	stacking	Stacking (LightGBM)	5	0.7647	0.1280	0.7142	0.1462	36	4.0
10	ensemble	RandomForest	20	0.7376	0.1352	0.7138	0.1463	420	4.1
11	nonLinear	MLP	12	0.7062	0.1430	0.7092	0.1475	424	4.9
12	stacking	Stacking (XGBoost)	3	0.7315	0.1368	0.7054	0.1484	50	5.5
13	stacking	Stacking (MLP)	20	0.6959	0.1455	0.6921	0.1517	159	7.5
14	nonLinear	KNN	20	0.7218	0.1392	0.6758	0.1557	1	9.9
15	stacking	Stacking (PolyReg)	20	0.6790	0.1495	0.6685	0.1574	3	10.9
16	nonLinear	PolyReg	5	0.6551	0.1550	0.6627	0.1588	60	11.6
17	stacking	Stacking (RidgeReg)	20	0.6763	0.1501	0.6566	0.1602	1	12.4
18	nonLinear	SVM	12	0.6231	0.1620	0.6341	0.1654	173	15.2
19	linear	RidgeReg	4	0.5964	0.1676	0.6038	0.1721	4	18.5
20	linear	LinReg	4	0.5846	0.1701	0.5924	0.1746	2	19.6

Although the model order is related to the training time, as higher lags represent higher input space dimension, we can infer that, in general, the computational cost seems to increase according to: $t_{KNN} < t_{linear} < t_{nonlinear} < t_{stacking} < t_{ensembles,MLP}$. Stacking ensembles are expected to be computationally expensive as they use a set of complex models as input, but this table considers the time consumed by training only the metaregressor, since the base regressors had been previously trained. The order of magnitude for creating such a stacking ensemble model was of about 43 minutes on average, using a regular PC (Core i5 1.60GHz processor and 16GB RAM memory) considering the training of all base estimators. We believe that this time is reasonable and does not hinder the use of these models in practical applications.

Figure 5.6 shows the overall best performance of the stacking ensembles considering all the model lags tested. The same scatterplots suggest that this class of models, as well as the ensemble class, shows a tendency to overfit, as the test score (total data) is slightly worse than the train score. That behavior is expected since the test data was not seen in the training procedure. We believe that the models were capable of generalizing well when predicting unseen data, as the overfitting is minimized by the use of a k-fold cross-validation procedure, as described in Section 3.3.1.

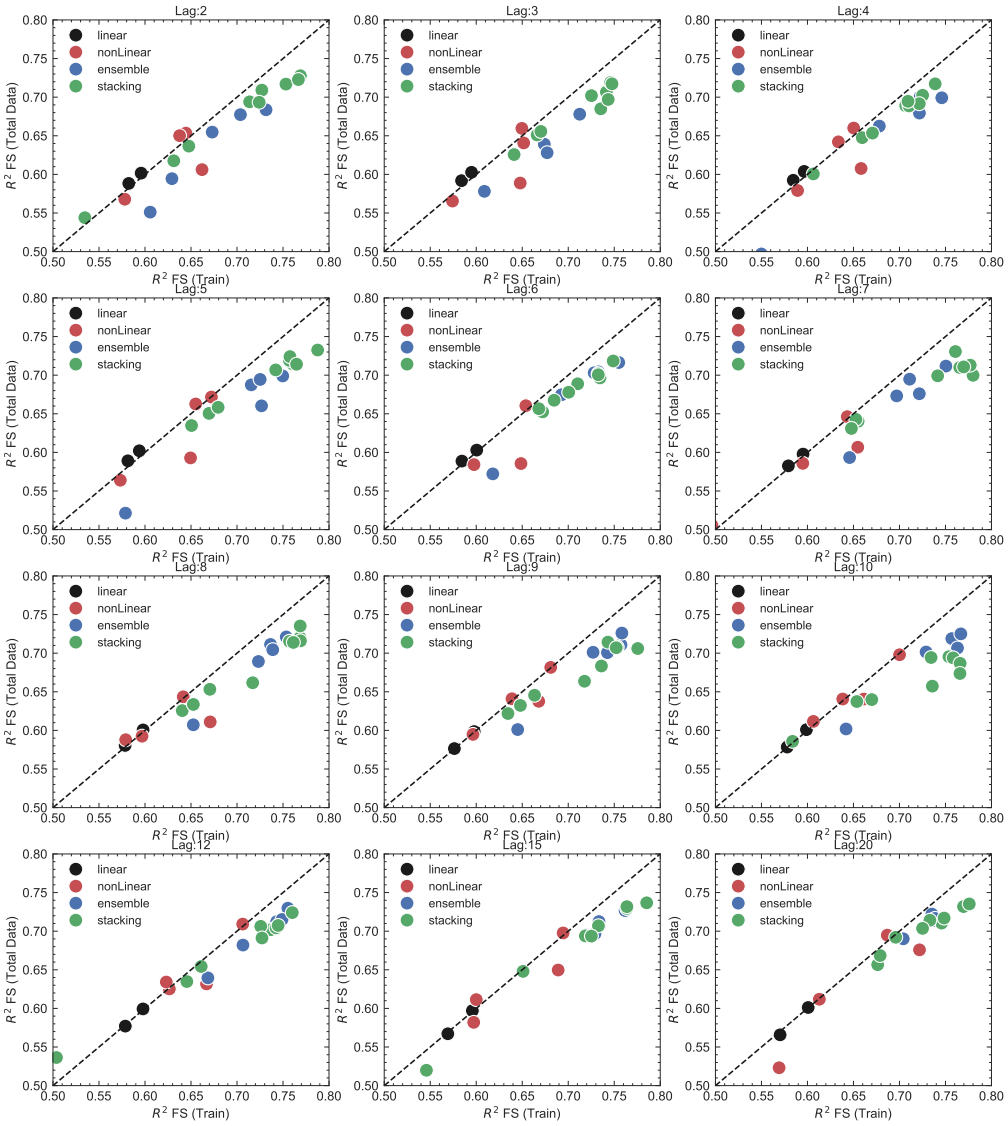


Figure 5.6: Well 15/F9-1C train and test (total data set) Free Simulation R^2 score for all lags tested.

Similar results were also found for wells 15/9-F15D and 15/9-F14, even with a higher improvement of stacking models compared to the other classes. A summary of the results for the three evaluated wells is shown in Table 5.5, where the metrics for the best model of each class of estimators are reported. The last column represents the relative gain of the stacking approach compared to each model.

Table 5.5: Summary of results for the three evaluated wells.

Well	**Output**	**Model**	**Lag**	**Rank**	**Class**	R^2	**RMSE**	**ΔRMSE %**
15/9-F1C	Liquid Rate	Stacking (Bagging)	15	1	Stacking	0.7367	0.1403	-
		GradientBoosting	12	2	Ensemble	0.7297	0.1421	1.3
		MLP	12	3	Non Linear	0.7092	0.1475	4.9
		RidgeReg	4	4	Linear	0.6038	0.1721	18.5
15/9-F15D	Oil Rate	Stacking (Bagging)	20	1	Stacking	0.7149	0.0949	-
		GradientBoosting	15	2	Ensemble	0.6871	0.0994	4.5
		KNN	12	3	Non Linear	0.6546	0.1044	9.1
		RidgeReg	6	4	Linear	0.4969	0.1273	25.5
15/9-F14	Gas Rate	Stacking (Bagging)	2	1	Stacking	0.8928	0.0701	-
		BaggingRegressor	9	2	Ensemble	0.8853	0.0725	3.3
		RidgeReg	9	3	Linear	0.7830	0.0998	29.7
		MLP	2	4	Non Linear	0.7799	0.1110	36.8

For the same estimators, Table 5.6 illustrates the best combination of hyperparameters tuned by the randomized search cross-validation procedure for the liquid production rate of well 15/F9-1C.

Table 5.6: Best hyperparameters achieved by the randomized search cross-validation procedure (omitted hyperparameters set to default values) - well 15/F9-1C.

#	Model	Lag	Params
1	Stacking (Bagging Regressor)	15	max_features=0.6765 , max_samples=0.9609 n_estimators=43
2	Stacking (Extra Trees)	20	max_depth=14, max_features=0.3907 min_samples_split=8, n_estimators=81
3	Stacking (Random Forest)	20	max_depth=15, max_features=0.4813 max_leaf_nodes=37, min_samples_split=9 n_estimators=99
4	Gradient Boosting	12	criterion='squared_error', learning_rate=0.07498 loss='huber', max_depth=9 max_features=0.2030, max_leaf_nodes=105 min_samples_leaf=18, min_samples_split=18 n_estimators=74, subsample=0.8991
5	Bagging Regressor	15	max_features=0.2703, max_samples=0.9006 n_estimators=85
6	Extra Trees	10	max_depth=14, max_features=0.3907 min_samples_split=8, n_estimators=81
7	Stacking (Gradient Boosting)	5	criterion='squared_error', learning_rate=0.04243 loss='absolute_error', max_depth=10 max_features=0.2472, min_samples_leaf=2 min_samples_split=14, n_estimators=88 subsample=0.4407
8	Light GBM	20	learning_rate=0.0373, n_estimators=119 num_leaves=35
9	Stacking (Light GBM)	5	learning_rate=0.19038, n_estimators=50 num_leaves=5
10	Random Forest	20	max_depth=11, max_features=0.2391 max_leaf_nodes=102, min_samples_leaf=4 min_samples_split=9, n_estimators=52
11	MLP	12	alpha=9.6073-05, early_stopping=True hidden_layer_sizes=(60, 60, 60, 60) learning_rate_init=0.003936 tol=2.3591-06
12	Stacking (XGBoost)	3	learning_rate=0.05566, max_depth=3 n_estimators=98
13	Stacking (MLP)	20	activation='tanh', alpha=896-6 early_stopping=True hidden_layer_sizes=(100, 100, 100) learning_rate_init=137.4-6, tol=1.2833-05
14	KNN	20	n_neighbors=4, weights='distance'
15	SVM	12	C=11.2726, degree=5 gamma=5074-6

5.1.4 Discussion

The main conclusion about these results is that multi-model strategies outperformed linear, nonlinear, and bagging and boosting ensemble models. Furthermore, the strategy of stacking predictions from diverse models such as KNN, SVM, MLP, linear, and PolyReg as base learners seems to wisely combine their strengths and heterogeneities to boost the quality of the final predictions. The stacking ensembles outperformed all other classes, reducing the RMSE error by 4.9% to 36.8% compared to the best nonlinear model and by 1.3% to 4.5% compared to the best ensemble technique.

Although these results are somehow in line with those obtained by Al-Qutami et al. in [68],[69], there is a significant difference between them: While Al-Qutami et al. achieved the diversity of base learners by changing their architecture based on neural network parameters, we obtained in this study heterogeneity by stacking completely different approaches, from simple linear ARX to NARX-based SVM, KNN, and MLP machine learning models. Therefore, the performance improvement obtained by stacking ensembles in this work can be explained by the complexity and diversity of the stacking base learners, since we expect that different algorithms can perform better for different instances of data.

Hence, we could infer that continually adding more complex models like ensembles as base learners would improve model quality even more, but, in contrast to this idea, tests showed the exact opposite behavior as the model accuracy got worse when we added such models as base learners. In that case, it seemed that models started to unlearn, which confirms the "black art" characteristic of stacked generalization mentioned in Wolpert's original paper [51].

According to Wolpert [51], when it comes to combining predictions from base models, stacking ensembles are a more sophisticated form of cross-validation. Stacking uses all base learners together to minimize their errors with respect to a given training set, while cross-validation techniques measure which base model performs better in predicting a partition of the training set when trained with the other partitions of it, and the winner takes it all.

In this case of study, stacking ensembles also performed better than other ensemble approaches such as boost algorithms, which reduce bias by sequentially improving the performance of base learners by learning with predecessor errors [93] and bagging ensembles such as ExtraTrees [92] and Random Forests [90], which reduce variance by randomly splitting the training data set into partitions to achieve better generalization.

In summary, given the complexity of the physical phenomena of multiphase flow, a reasonable strategy to achieve a good model would be to wisely combine heterogeneous learners and carefully understand which model order applies best. According to this study, stacking ensembles and system identification approaches are powerful tools to address these issues, respectively. In future work, more strategies to deploy stacking ensembles, in the sense of better choosing base learners and meta-regressors, can be investigated, as well as feature engineering methods can also be applied.

An important limitation on data-driven methods is that each well has different characteristics, such as productivity, flow regime, water and gas fractions. Even in the same well, there is still difficulty in modeling the dynamic behavior of production since the physical phenomena that take place within a specific time frame of a well's production history are unlikely to be replicated over time, as the measurement quantities tend to vary.

Even though this intrinsic characteristic of data-driven models can be observed, it is also remarkable that our models presented a good performance even for unseen data, as depicted in the Figures 5.7, 5.8 and 5.9, which show the time series comparison of the actual data and the best model FS predictions for wells 15/F9-1C, 15/F9-15D and 15/F9-14, respectively. The figures show the best model achieved by each class. The unseen test data is hatched by the gray rectangles. The well 15/9F-1C has its shut-ins and openings represented even if in the test period the operating condition presents different water cut and gas oil ratios. In well 15/9F-15D, some oil sharp declines are not caught by the model, but in general, the model behaves well for unseen data, considering that the time window has a different operating condition of producing higher water rates. Finally, for well 15/9F-14, the model is also capable of predicting unseen data, especially considering the level of production and pressure different from the training period.

Other points to remark on these timeseries are the overall better accuracy of the stacking ensemble models compared to the other models, as well as the benefit from a free-run simulation run, in which a long-term simulation can be achieved given only the initial condition points corresponding to the order of the model, resulting in the full-time series predicted using previous predictions as new inputs.

The models achieved by this study show a reasonably low forecast error and a high value of the determination coefficient R^2, especially considering that the measured data, whether production rates or input sensor measurements, contain noise and are reported on a daily average basis. Therefore, it is reasonable to suppose that the same models trained using a higher sampling

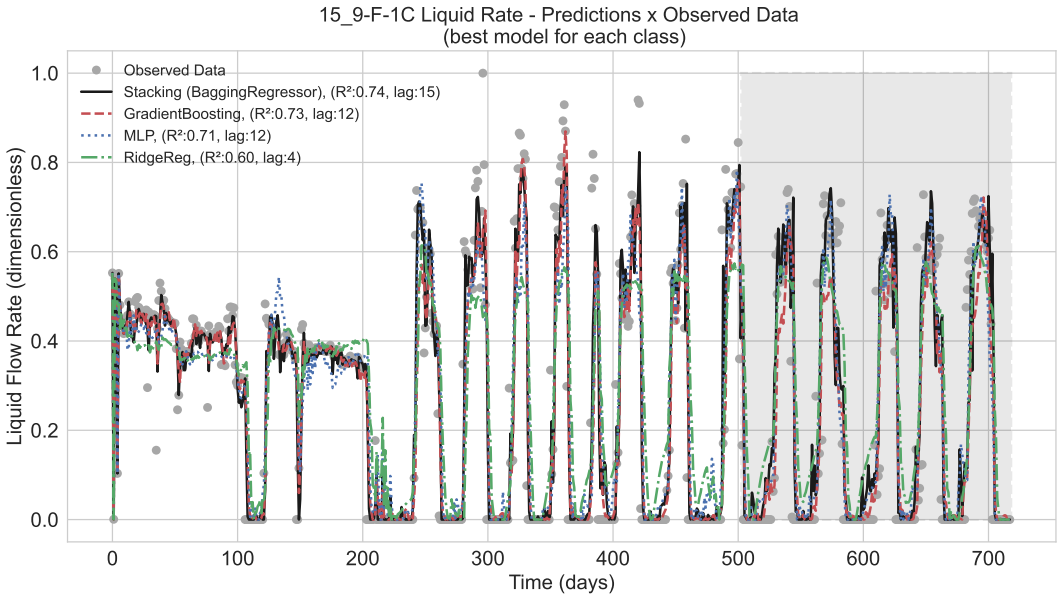


Figure 5.7: Predictions for well 15/F9-1C - Best model from each class

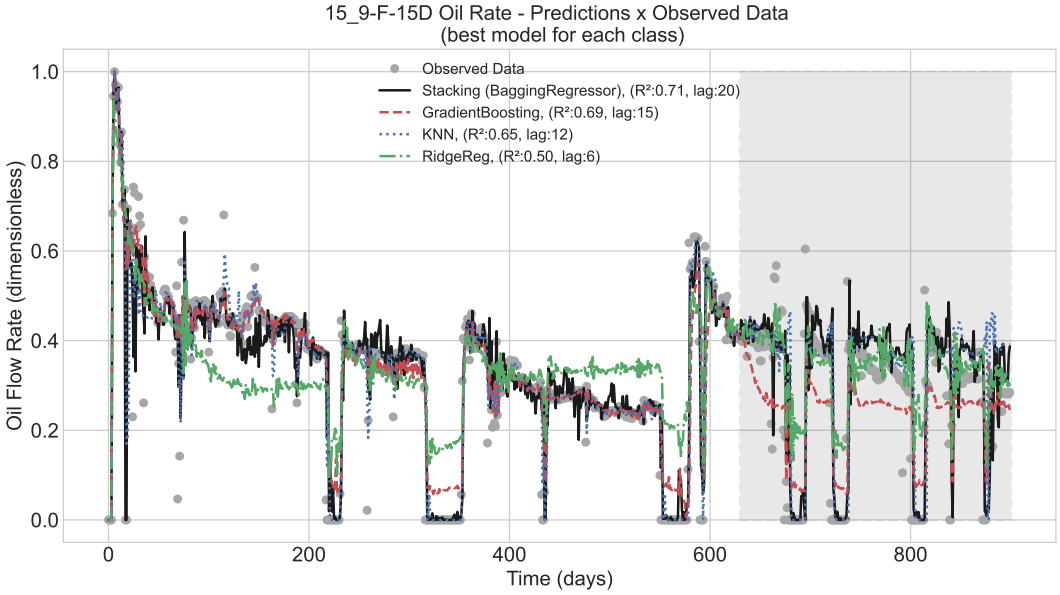


Figure 5.8: Predictions for well 15/F9-15D - Best model from each class

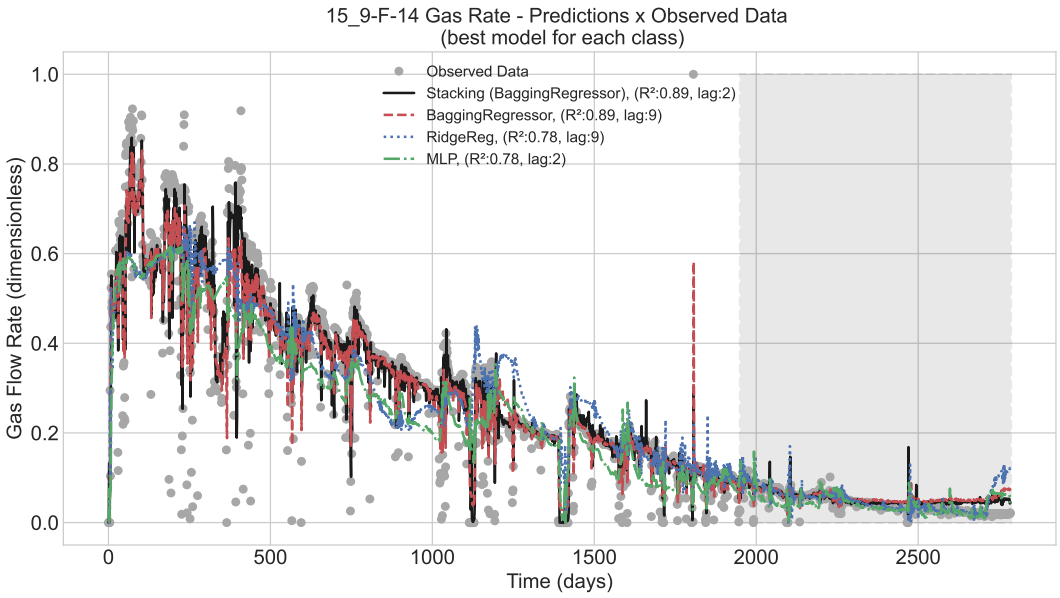


Figure 5.9: Predictions for well 15/F9-14 - Best model from each class

frequency of the data would lead to more accurate predictions. For well 15/F9-1C, a cross-plot of predicted vs. observed data is shown in figure 5.10.

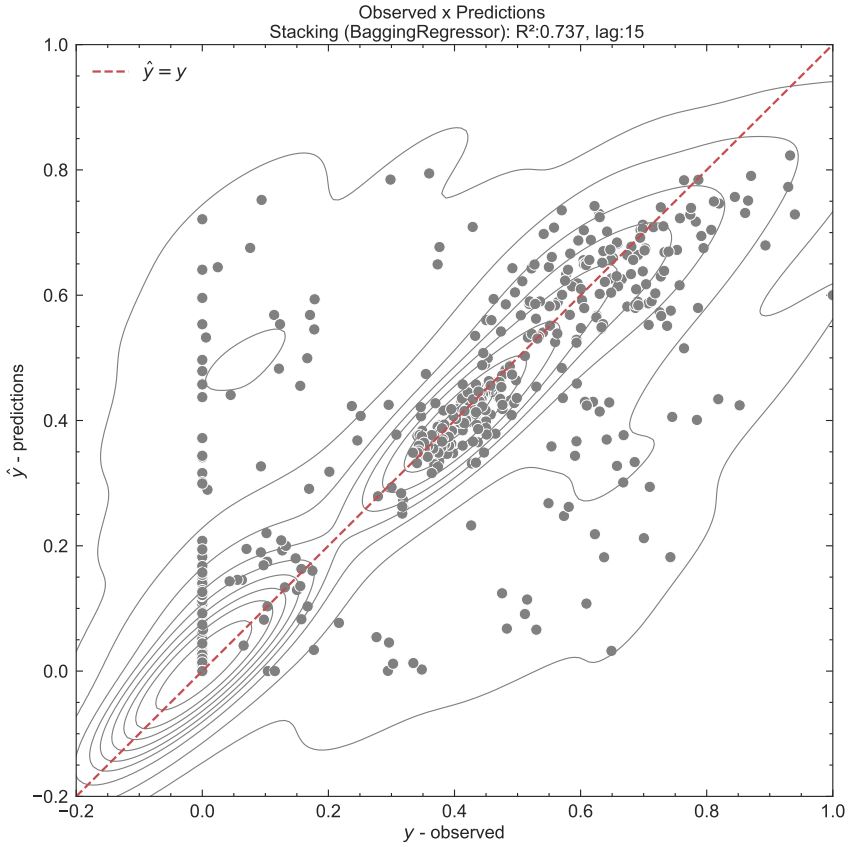


Figure 5.10: Comparison between the best model predictions and observed values (15/F9-1C)

A drawback of black-box models is their interpretability, especially in cases where understanding the underlying relationships is crucial. The interpretability regarding the lags, base estimators, or the input features can be relevant in our case. Besides that, black-box models combined with models designed with a priori phenomenological information can also become more accurate. Gray-box approach models were recently presented in de Sousa et al. [111], Pires et al. [112] and do Lago et al. [113].

5.2 Residual Training to Improve Virtual Flowmetering Models

This section investigates a second approach to improve the accuracy of machine learning-based virtual flowmeters. In this case, a second estimator was trained to predict the errors of the base estimator. The results showed that this two-step training approach, applying a residual predictor, improved the quality of the model's predictions by 1% to 13% depending on the complexity of the base estimator.

This contribution, as part of a more comprehensive work on feature engineering techniques for applications in the rates prediction problem, was presented at the 20th Symposium on System Identification of the International Federation of Automatic Control (IFAC), which took place in July 2024 at Northeastern University in Boston, MA. The reference to this publication is provided in Section 1.4.

5.2.1 Proposed Method

The method proposed in this study involves creating a residual model to improve the overall performance of the predictions. The values of errors estimated by the residual model were then added to the base model to compose a final prediction.

Figure 5.11 depicts the graphical abstract of the proposed approach. After the definition of the regression matrix (X) and the target vector (y) from the training set, two models were trained. In the training and validation step, both base and residual models use X as the input set, but the second is trained to predict the error as the desired output. This step involves evaluating the residual error y_{res} based on the values predicted by the base model $\hat{y}$ as:

$$y_{res} = y - \hat{y} \tag{5-2}$$

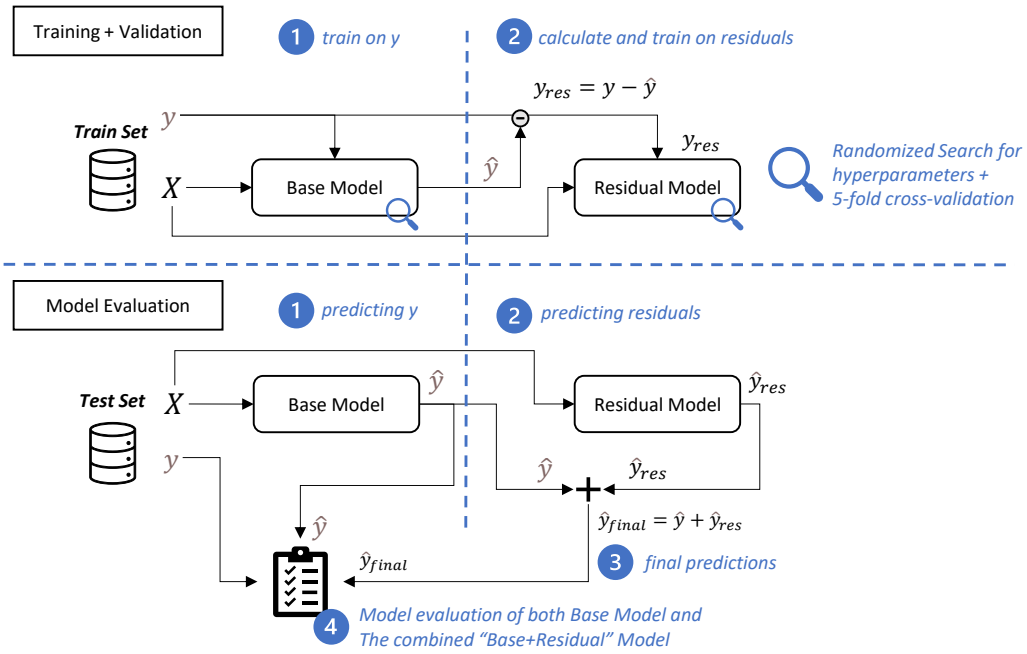


Figure 5.11: Graphical Abstract: Two step training and residual model approach.

In the model evaluation procedure, which consists of obtaining predictions for the test set, the base and residual models predict the values $\hat{y}$ and the residuals y_{res}, respectively. These predictions are then summed to compose a final prediction. To evaluate the improvement of such an approach, R^2 and $RMSE$ metrics are calculated for both the base model, which consists of a baseline for comparison, as well as the combination of the two-step training procedure. The final predictions for the two-step training model are:

$$\hat{y}_{final} = \hat{y} + \hat{y}_{res} \tag{5-3}$$

The regression algorithms listed in table 5.7 were both trained as base and residual models.

Table 5.7: Regressors models used both as base and residual models

Model	Description
LinReg	Linear Regression [4]
PolyReg	Polynomial Regression (n=2)
SVM	Support Vector Machine Regressor [84]
KNN	K-Nearest Neighbors Regressor [85]
MLP	Multi-Layer Perceptron [86]
Decision Tree	Decision Tree Regressor [87, 88]
RandomForest	Random Forest Regressor [90]
GradientBoosting	Gradient Boosting Regressor [93]
ExtraTrees	Extra Trees Regressor [92]

5.2.2 Results

In this section, we present the results obtained by the two-step training approach. The liquid production rate from well 15/F9-1C were used to evaluate the effectiveness of the models, using the same set of exogenous inputs as in the previous case (chapter 5.1): downhole and wellhead pressure and temperature, as well as the surface choke valve opening.

The pre-processing steps, namely the input/output feature selection, data standardization, and the assembly of the regression matrix, were the same as described in chapter 5.1, as well as the evaluation of different model orders and the hyperparameter tuning with cross-validation.

The results, in terms of free simulation R^2 are presented in Figure 5.12 for all the evaluated base models and for both one- and two-step training cases. The variation in the boxplots is due to the combination of multiple residual regressors and model orders evaluated. The boxplots represent min, max, quartiles, and median while the black dots represent the mean score of each model in each case. It is remarkable that all the base model estimators benefit from a residual model prediction, but this increase in performance by the use of the 2-step training model is more evident for simpler models such as LinReg, PolyReg, SVM, and MLP regressors, while more sophisticated base models such as RandomForest, GradientBoosting, and ExtraTrees see no relevant improvement.

To evaluate the performance of the regressors as residual models, the Figure 5.13 presents the statistics of the R^2 increase obtained by each of the algorithms. The variation in the boxplots represents the combination of multiple base regressors and the different model lags. RandomForest, GradientBoosting, and ExtraTrees regressors presented very similar results acting as residual predictors. For some cases, these models achieved an increase

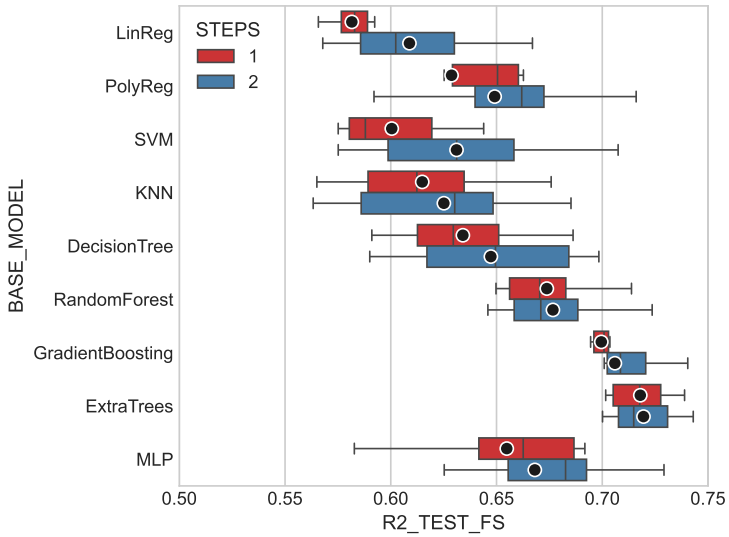


Figure 5.12: Analysis of the impact of the residual modeling approach regarding each one of the base estimators.

of more than 15%, and on average, considering all orders and base models, they improved the R^2 approximately by 5%. On the other hand, those models also present some cases in which their performance presented the poorer results, decreasing the R^2 by 5% to 7%. This shows the importance of combining several models and investigating optimal lags, as previously presented in chapter 5.1.

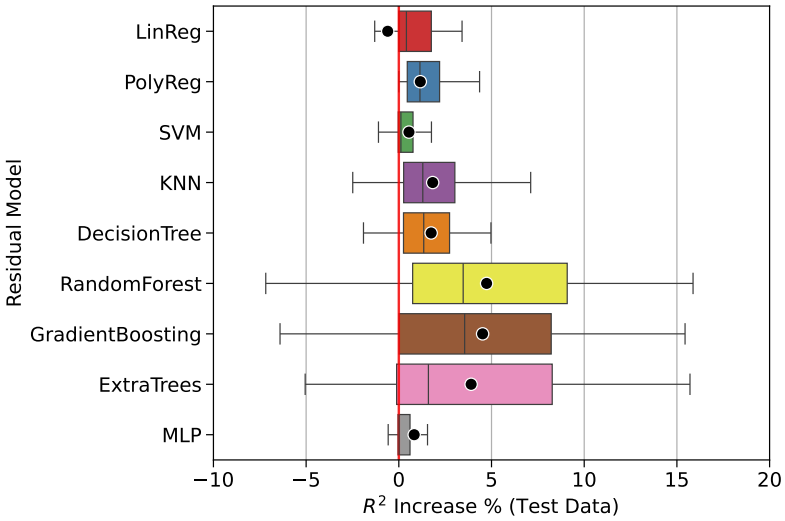


Figure 5.13: Overall performance of estimators acting as residual regressors.

In Figure 5.14, we analyze the impact of the model order on the results. The graph illustrates the relationship between the model order and the R^2 for both one-step and two-step training cases, for each of the base case models evaluated.

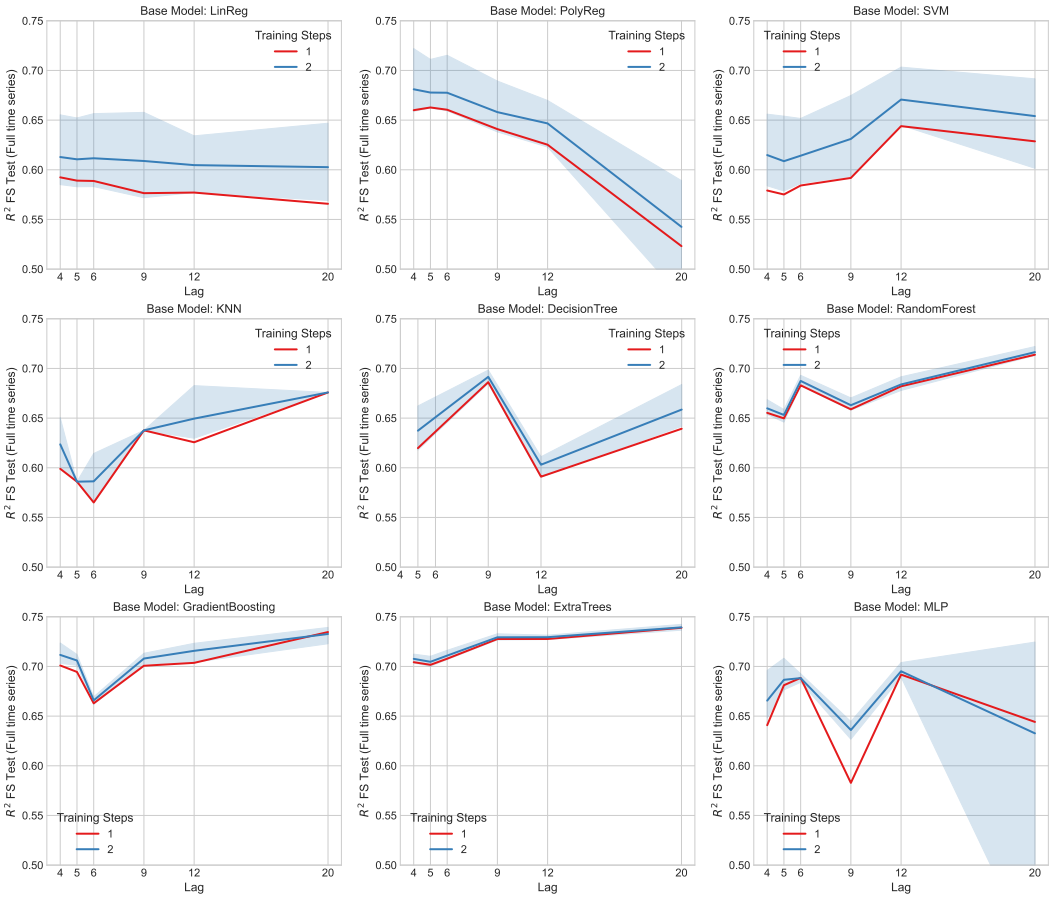


Figure 5.14: Impact of the 2-step training procedure and the model lag on each of the 9 base estimators evaluated.

By examining the trends, we can draw meaningful conclusions about the effectiveness of the model order optimization and the use of the residual model approach, such as:

- The trends show that each model suffers different impact depending on their complexity;
- Linear, Polynomial, SVM and MLP models improved their performance using the residual models, although they did not clearly benefit from model order increase;
- In the particular case of Polinomial Regression, the increase in the model lag significantly worsens the model prediction performance, the algorithm clearly did not deal well with more input features;
- KNN algorithm has a slighlty improvement with order increase and by the use of the residual models;
- Trees-based ensenble models, namely DecisionTree, RandomForest, GradientBoosting and Extratrees present no significant improve by using of

residual models, but perform better with the increasing of the model order, due to their capacity of dealing with more features.

– For the MLP regressor, R^2 was highly impacted depending on the residual estimator.

Finally, the heatmap presented in Figure 5.15 illustrates the R^2 score of all possible combinations of base and base/residual models, considering the optimal lag of each combination. The first column represents the base model strategy with no additional residual model. It can be deduced that more complex regressors, like tree-based models and MLP, obtain minimal advantage from the residual model approach due to their already better performance. However, simpler regressors, such as linear and polynomial types, exhibit greater improvement using this strategy, particularly when paired with more sophisticated regressors to handle errors.

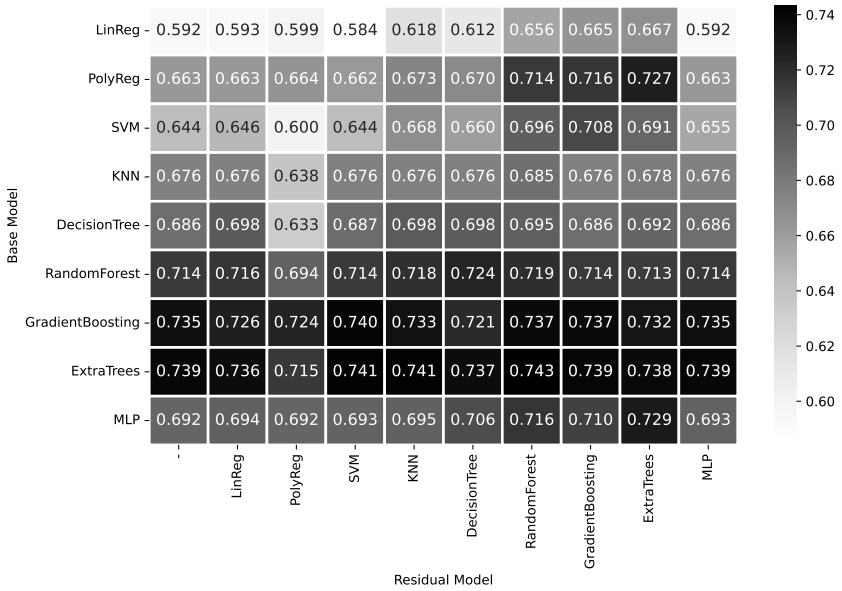


Figure 5.15: Heatmap of possible combinations of base and residual models, evaluated on each combination's optimal lag.

A summary of the best model results, as well as their respective increase in R^2 and $RMSE$ is provided in Table 5.8. This table shows the metrics for both base and 2-step training models. For comparison purposes, the corresponding base case model was selected with the same model order as the respective base/residual model. The most relevant increase in performance was observed with MLP, LinReg, PolyReg and Knn estimators. As discussed in the previous paragraphs, more sophisticated approaches such as Trees-based ensembles presented a poor increase in performance due to their intrinsic high accuracy for rate prediction purposes.

Table 5.8: Comparison of the best base+residual models combinations evaluated with their respective base case as baseline models for comparison.

Base Model	**Residual Model**	**n**	R^2 **2 step**	R^2 **base**	R^2 Δ%	**RMSE 2 step**	**RMSE base**	**RMSE** Δ%
ExtraTrees	RandomForest	20	0.743	0.739	0.6	0.139	0.140	0.8
Grad. Boost.	SVM	20	0.740	0.735	0.8	0.139	0.141	1.1
MLP	ExtraTrees	20	0.729	0.644	13.2	0.142	0.163	12.8
PolyReg	ExtraTrees	4	0.727	0.660	10.1	0.143	0.159	10.4
RandomForest	DecisionTree	20	0.724	0.714	1.4	0.144	0.146	1.7
SVM	Grad. Boost.	12	0.708	0.644	9.9	0.141	0.163	9.4
DecisionTree	LinReg	9	0.698	0.686	1.8	0.150	0.153	2.0
KNN	RandomForest	12	0.685	0.626	9.5	0.153	0.167	8.3
LinReg	ExtraTrees	9	0.667	0.576	15.7	0.158	0.178	11.3

The final predictions for MLP and KNN models are shown respectively in Figures 5.16 and 5.17. The quality of prediction of the MLP model is significantly increased by the use of a residual model, especially for predicting the successive shut-ins.

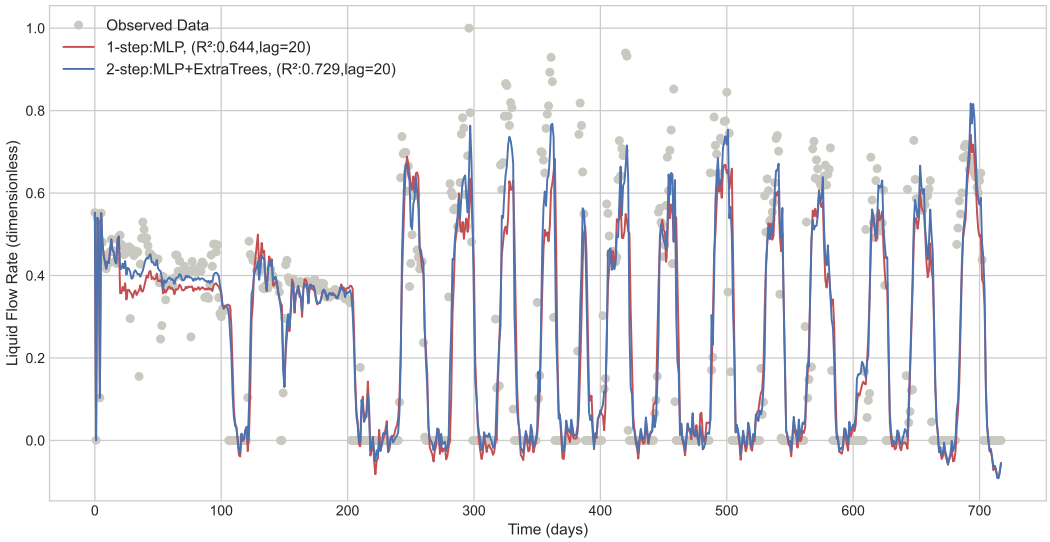


Figure 5.16: MLP Predictions for one and two step training cases

For the KNN model, the predictions of the base case and the 2-step model are slightly different, being noticed only in the reopenings from well shut-ins.

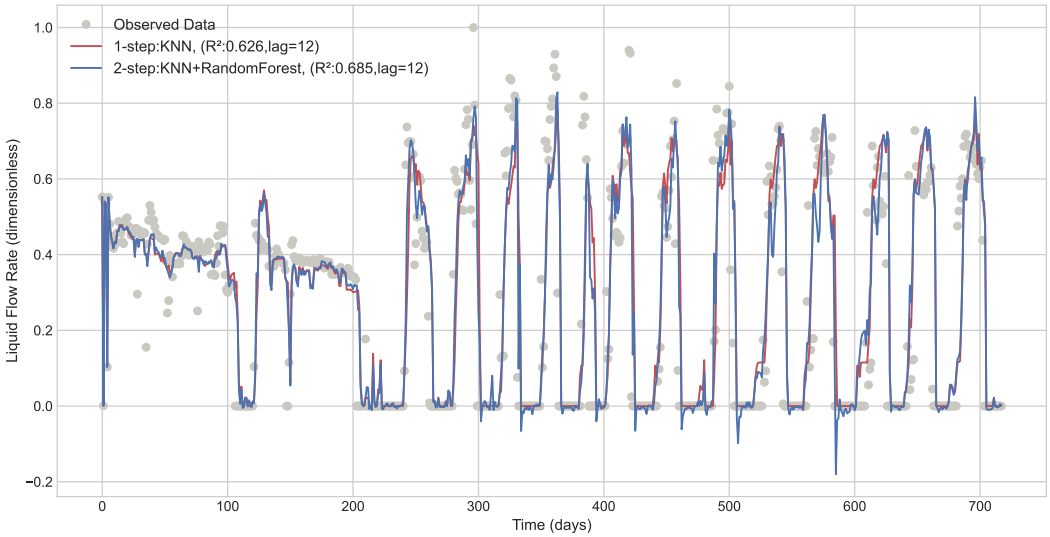


Figure 5.17: KNN Predictions for one and two step training cases

5.2.3 Discussion

An important conclusion about these results is that the use of residual models to improve the performance of well rate estimator models can increase model performance and reliability, but strongly depends on the capabilities of the base model. Significant improvement was observed for linear and some non-linear models such as MLP and Polynomial regression, while more sophisticated algorithms based on gradient and trees presented a poor gain in performance due to their prior ability as regular regressors.

In complement to the analysis and conclusions presented in chapter 5.1 regarding the importance of the time dependency (optimal lag) of each algorithm, once again this aspect should be highlighted as an important conclusion, as each of the evaluated regressors benefits differently with respect to a model order increase. Therefore, the optimal lag becomes an important parameter to be optimized in addition to those belonging to the mathematical formulation of each algorithm.

An important insight of this study, which deserves a discussion, is that more simple models can achieve the level of performance of more complex algorithms, although the results also show that the tree-based ensemble models performed better whether as base or residual models, indicating that at least one of these parts of the combined model must count on more complex algorithms.

In Huang et al. [67], an innovative Time Patch Dynamic Attention Transformer (TPDAT) is proposed to predict oil production rates using the

same dataset (Volve field) that we used in our study. Although the authors achieved remarkable results, we cannot directly compare our findings with those of Huang et al. for two main reasons. First, the use of water production as an input limits the applicability of those models, as it is not an actual sensor measurement. Second, and more importantly, our formulation is different from Huang et al. and applies for different purposes: while in the reference paper, the prediction is a short-term (20% of the data) one-step-ahead (OSA), in our work we aim to provide a free simulation run prediction capable of predicting long-term historic production based only on measurements of sensor data and given only some initial condition. This means that we only use the initial conditions and obtain predictions by iterating previous predictions, which is in fact much harder than OSA. Our formulation differs from Huang et al. and previous papers by proposing a different approach, which in itself constitutes an original contribution. Additionally, since the free-simulation run is an original formulation, we chose to compare the performance of our models to usual benchmark literature algorithms.

5.3 Well Logs Estimation Using Multimodel Strategies and Feature Expansion Based on Seasonal Decomposition

In this section, we developed machine learning models to create synthetic porosity, compressional slowness, resistivity and bulk density logs, focusing on two new approaches based on seasonal decomposition to enhance model quality compared to usual supervised learning. Feature decomposition, expanding the input space in trend and combined seasonal-residual terms, was the winning strategy for three of the four evaluated logs and promoted a gain of 8.5% in R^2 and reduced RMSE by 27% in the case of the compressional slowness log. Furthermore, the presented ensemble-based method, in which estimators are specialized in trend and combined seasonal-residual predictors, outperformed the baseline models for 6 of the 8 examined estimators for both porosity, compressional slowness and resistivity predictions. In addition, this strategy proved to be the best for the bulk density log, improving R^2 by 1.7% and reducing RMSE by 5.2%. In all four scenarios examined, both methods achieved better results than the baseline strategy and proved to be robust techniques for well-log estimation purposes.

This work was presented at the 25th Brazilian Congress of Automation (CBA) of the Brazilian Society of Automation (SBA), which took place in October 2024 in Rio de Janeiro, Brazil. The contribution is referenced in Section 1.4.

5.3.1 Proposed Method

Supervised learning is the category of machine learning problems in which models are trained in a labeled data set (train set) to predict new data (test set). Predicting a continuous value, in our case the well log, is known as a regression problem. To perform these predictions, several regressors were evaluated, from simple models such as linear models to more complex algorithms such as gradient-boosting regressors. A list of the estimators used in this study is shown in Table 5.9.

Table 5.9: Evaluated Estimators

Estimator (Model)	Algorithm
LinReg	Linear Regression
PolyReg	Polynomial Regression of degree 2
SVM	Support Vector Machine
MLP	Multilayer Perceptron
RandomForest	Random Forest
AdaBoostRegressor	Adaptative Boosting
GradientBoosting	Gradient Boosting
XGBoost	Extreme Gradient Boosting

In this study, we used 15/9-F-1A and 15/9-F-11A as training wells and 15/9-F-1B as the test well, all belonging to the Volve dataset described in Section 4.2. The main reason for selecting those wells is that they have the complete set of well logs including DT and DTS logs. Such models could be used to predict DT and DTS logs for the wells where this information was not acquired.

A 100-iteration randomized hyperparameter search with cross-validation was used for fine-tuning. For LinReg and PolyReg, only the coefficient signal was optimized. The optimized SVM parameters were C, kernel, degree, and gamma. MLP models optimized the activation function, early stopping, learning rate, tolerance, and hidden layers. Other estimators optimized the number of estimators, the maximum features, and the maximum depth. The code and data files are available in the repository [114].

As an initial exploratory analysis, a cross-correlation matrix was calculated to evaluate the similarity between the variables is shown in Figure 5.18.

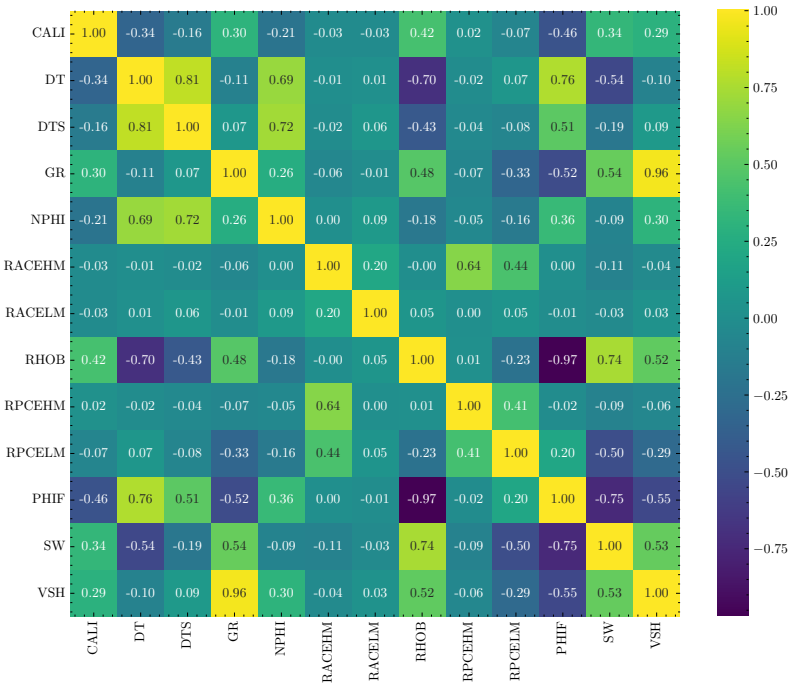


Figure 5.18: Correlation between variables.

To avoid using strongly correlated logs to predict one another and also to eliminate redundant features, some of the log curves were eliminated from the analysis. Upon the available logs, only one of the resistivities, RACEHM, was kept since all resistivities presented a similar correlation profile with other variables. The same is true for DT and DTS, where only the first was used. The shale volume curve, VSH, was also eliminated as a result of its high correlation with GR. Although RHOB and PHIF present a high correlation, both were kept as main features as the tests showed that these increased results in the tests performed, and PHIF was not used when predicting RHOB data.

Table 5.10 describes the selected inputs/outputs used in this study. Attempts to predict other valuable variables, such as the GR log, were made, but without satisfactory results. .

Table 5.10: Variables used as inputs and outputs.

Inputs	Output
GR, NPHI, RHOB, PHIF, RACEHM	DT
DT, GR, NPHI, SW, RACEHM	RHOB
CALI, DT, GR, RHOB, PHIF, RACEHM	NPHI
DT, GR, NPHI, RHOB, SW	RACEHM

To normalize the logarithmic behavior of the resistivity, we applied a *log* transform to this variable. All inputs and outputs were scaled to the interval

$[0, 1]$. The data were then resampled every three points to save computational time.

Table 5.11 provides a summary of the available data points that show that 35% of the data was used as a holdout set for model evaluation.

Table 5.11: Number of datapoints.

Well	Train/Test	Datapoints
15/9-F-1A	Train	2099
15/9-F-11A	Train	1402
15/9-F-1B	Test	1903 (35%)

5.3.1.1 Seasonal Decomposition

Seasonal decomposition is a technique used in time series analysis that consists of breaking down a time series into trends, seasonality, and residuals [115, 116]. In classical additive decomposition, a timeseries can be decomposed as:

$$y = T + S + R \tag{5-4}$$

In equation 5-4, the trend component T denotes the long-term progression or patterns observed in the data. The seasonal component S captures the recurring patterns that occur within a specific period. Lastly, the residual component R accounts for the random variations or noise in the data not explained by the trend or seasonal components.

In our study, we decomposed the variables within the depth domain, choosing a seasonality period of 20 points (roughly 8 meters). This parameter was established by testing values between 10 and 100, which helped to separate low- and high-frequency terms for inputs and outputs, although it has no physical significance. Figure 5.19 illustrates the decomposition of the GR log for well 15/9-F-1B.

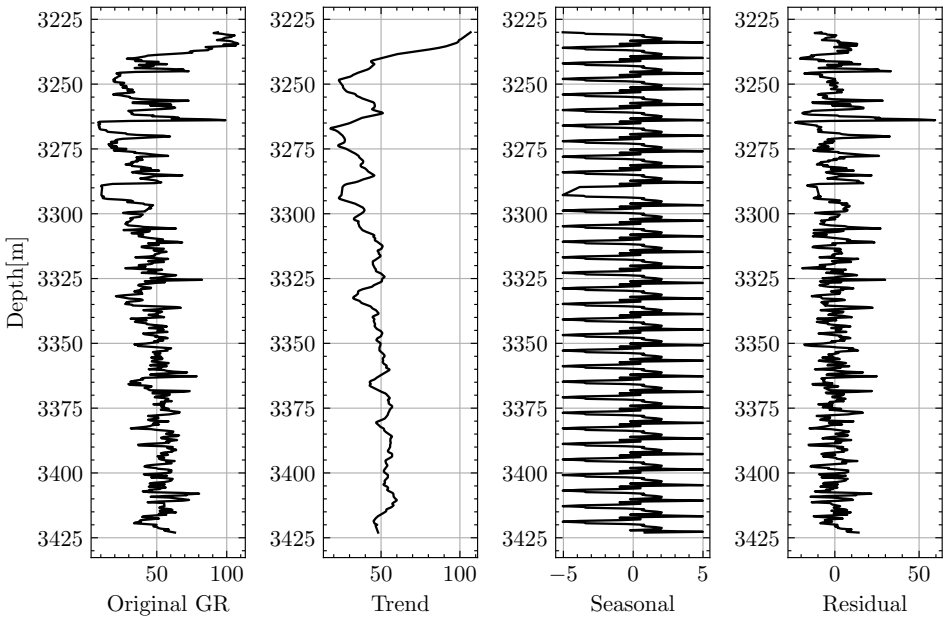


Figure 5.19: Decomposed GR for well 15/9-F-1B

The graphical abstract depicting the experimental design used in this study is presented in Figure 5.20.

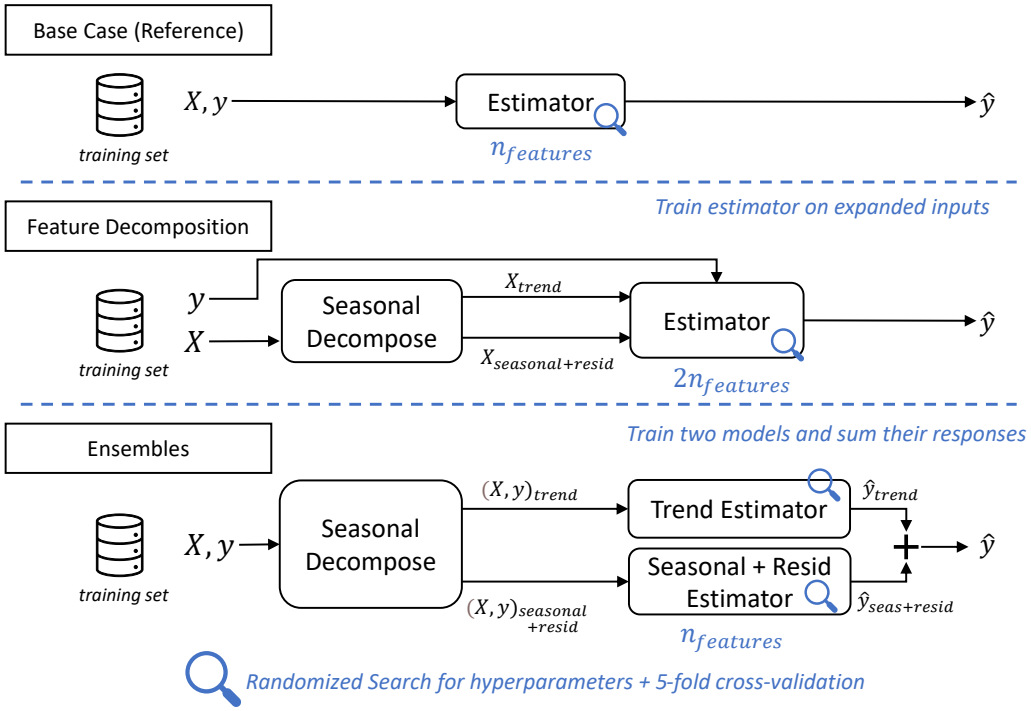


Figure 5.20: Graphical Abstract: well logs estimation using multimodel strategies and feature expansion based on seasonal decomposition.

The baseline (reference) case for comparison was the Base Case in which the models were trained conventionally on the original input and output data.

The Feature Decomposition case was designed to evaluate the use of seasonal decomposition as a feature expansion method. The input matrix was decomposed into trend, seasonal, and residual terms. For simplicity and to save training time, the seasonal and residual terms were grouped into one. The estimators are then trained in an expanded input space with the original output.

The third case proposed in this study (Ensembles) aims to answer the question if it is possible to improve model performances by creating an ensemble in which each of them is specialized in predicting different parts of the data. For that purpose, we trained a model to predict the trend and another to predict the combined seasonal-residual terms, summing their responses to obtain the final prediction. In this case, the output is also decomposed. The eight estimators listed in Table 5.9 were trained both as trend estimators and combined seasonal-residual estimators, resulting in a total of 64 possible ensembles for each log. For both Feature Decomposition and Ensembles, it is important to remark that the same procedure of decomposing the input must be applied to the test set when evaluating the models.

Throughout this work, [X].[Y] denotes an ensemble estimator composed of the "X" model as a trend estimator and the "Y" model as a seasonal-residual estimator. For example, Linreg.Adaboost. Similarly, to represent a model trained with expanded feature input (feature decomposition strategy), we use [X](DEC). For example, Linreg (DEC).

5.3.2 Results

5.3.2.1 Cross-validation and Model Selection

To perform the choice of the best hyperparameter set, a 5-fold cross-validation procedure with 10 repetitions was implemented, performing a total of 50 train/validation splits. For each estimator, the best combination of hyperparameters was chosen based on the highest mean R^2 score in the 50 cross-validation splits, and their results are summarized in Figure 5.21 for the base and feature decomposition cases, showing a clear improvement in the performance of the latter.

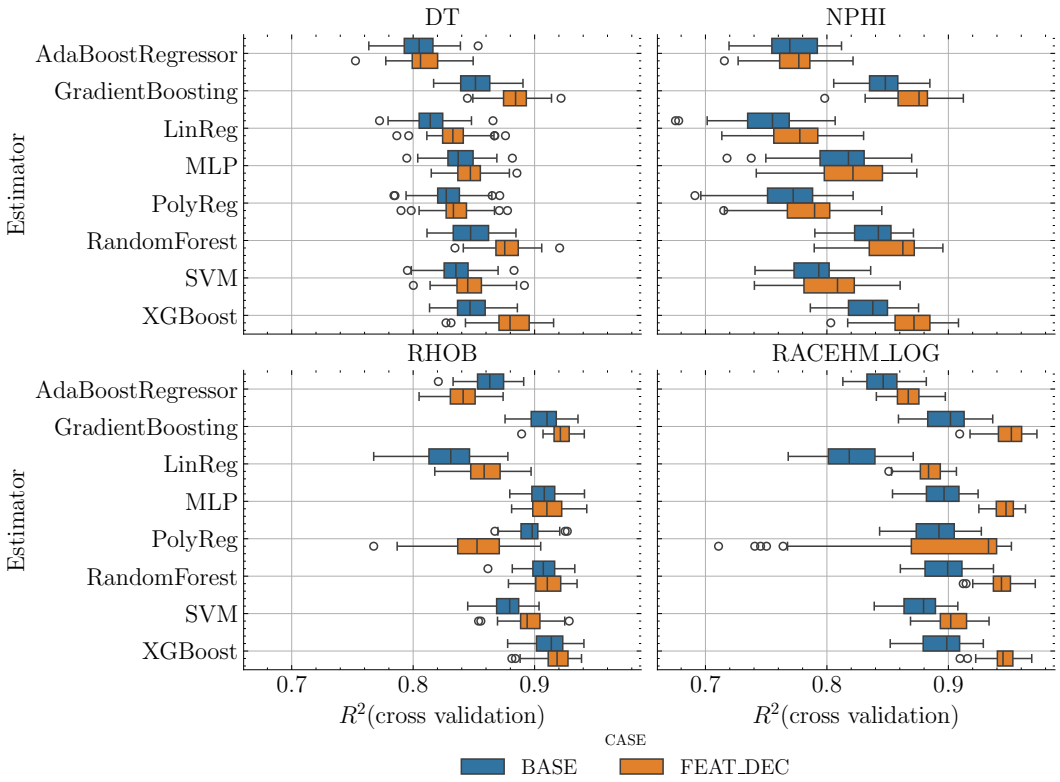


Figure 5.21: Cross-validation results for the best hyperparameter set for each estimator: base case vs feature decomposition.

A similar strategy was performed for the ensemble case, where the hyperparameter search and cross-validation were performed separately for both the trend estimator and the seasonal-residual estimator. In the next sessions, we evaluate the performance of the hyperparameter-tuned estimators in terms of the test set R^2 metric.

5.3.2.2 Ensemble Results

The graph depicted in Figure 5.22 illustrates, for the DT log, the results of the ensemble strategy. Lines represent trend estimators, and columns represent estimators trained in seasonal + residual terms. The first column refers to the base case, where each line represents an estimator trained on the original input data. Each estimator was trained both as a trend predictor and as a seasonal + residual predictor, totaling 64 combinations (ensemble models).

A key point to be observed in this graph is that there are significant differences in performance when we compare the ensemble models with the base case. Only 2 of 8 estimators (PolyReg and XGBoost) did not improve

their results using the ensemble strategy, and the best improvements were observed for the MLP and SVM regressors.

Another remark is that, when comparing the results of the ensembles, they are quite similar to each other once the trend estimator is fixed. The performance of those ensembles is more related to the quality of the trend estimator, as it represents the general trend of the data. However, in most cases, the seasonal + residual estimator is responsible for an additional improvement in the quality of the predictions.

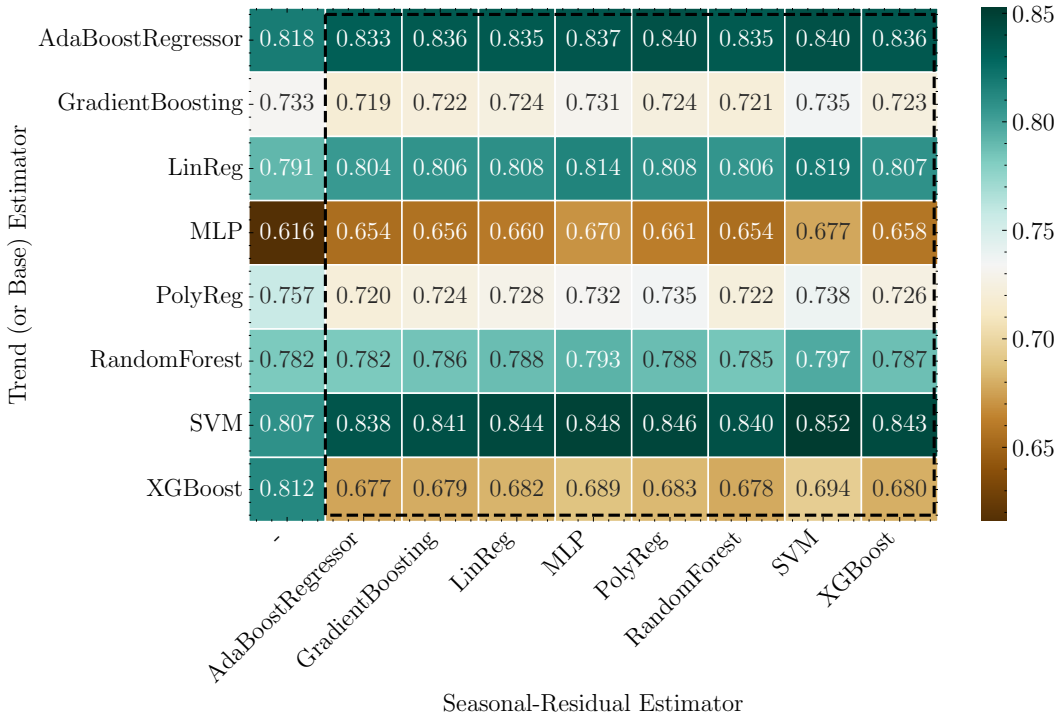


Figure 5.22: Comparison of R^2 (test), base case and the 64 ensemble combinations for DT log.

5.3.2.3 Overall Results

In Figure 5.23 we compare the three strategies for all estimators and logs. Feature decomposition clearly achieves better results for the DT log specifically with respect to MLP and PolyReg estimators. For the RACEHM log, feature decomposition also presents a better performance but is very close to the ensemble strategy for most of the estimators.

To predict the NPHI log, the performance is quite similar considering all the cases. Although the ensemble method is the winning strategy for 4 of the 8 estimators, the overall best estimator is MLP using feature decomposition.

Despite their higher complexity, the ensemble models did not significantly improve the results considering the estimators trained with decomposed inputs. The only exception is the RHOB log, where this approach overperformed both the base and the feature decomposition cases. For RHOB predictions, the feature decomposition strategy presented poorer overall performance.

It is also interesting to note that for all logs, the best overall result is found for an estimator based on the feature decomposition strategy. The best estimators are PolyReg, MLP, SVM, and SVM for the DT, NPHI, RACEHM, and RHOB logs, respectively.

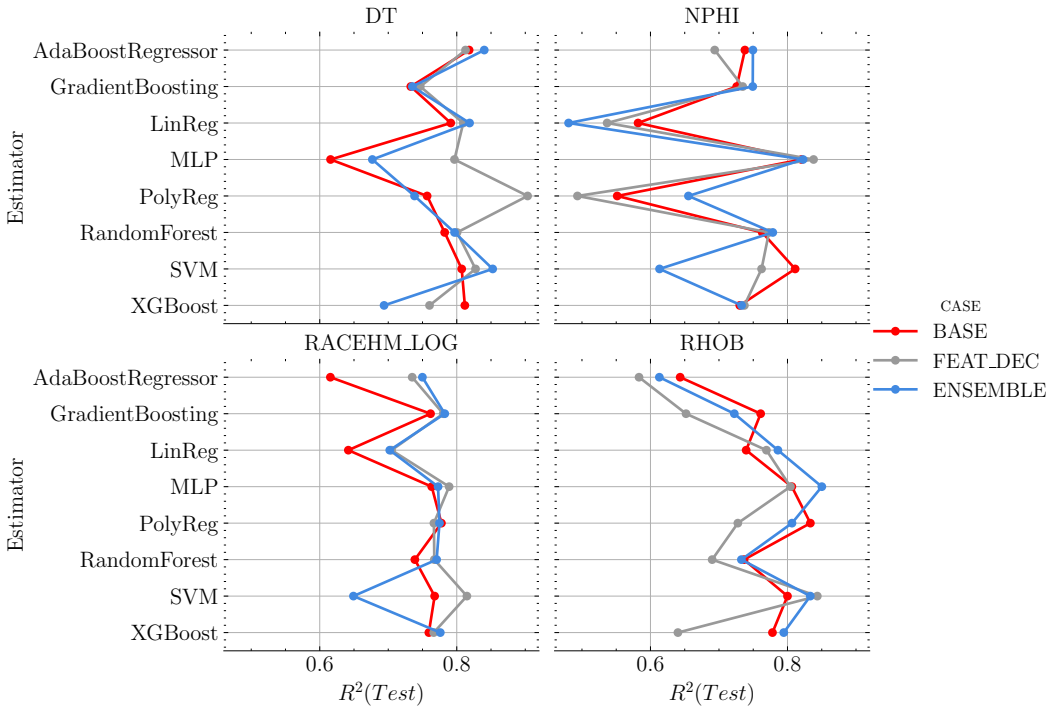


Figure 5.23: Feature decomposition and Ensemble strategy results compared to the base case.

A summary of the cases evaluated is presented in Figure 5.24. The R^2 is shown for each combination of the predicted log and the estimator. For each estimator and log, the best strategy is presented by the color, and the overall best predictor for each log is marked with the word "best". We can also remark on the overall success of the strategies proposed in this research, except for the RHOB log, but even in this case, the best performance was achieved by an ensemble model.

The proposed methods surpassed the base case in 24 of 32 combinations. These methods were notably effective for the DT and RACEHM logs, where one of these strategies overperformed the base case for 7 of the 8 estimators. In

contrast, improvements for the RHOB log were observed in only 4 estimators. Furthermore, feature decomposition and ensemble techniques increased the prediction accuracy for 6 of 8 estimators in relation to the NPHI log. Another remark is that the optimal strategy depends on the chosen estimator.

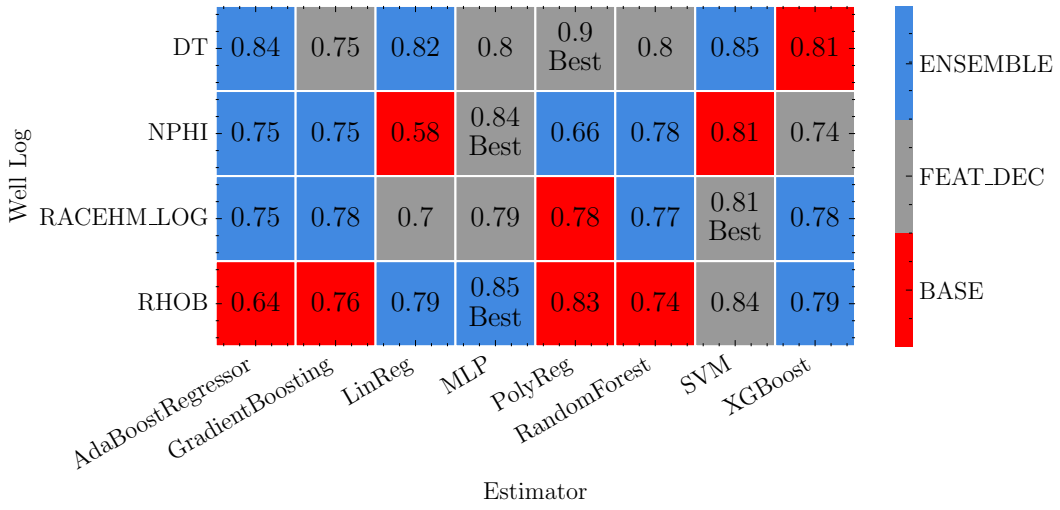


Figure 5.24: Overall evaluation of estimators and strategy.

In Figure 5.25, models are ranked in terms of R^2 and colored by strategy. As discussed in Section 5.3.2.2, similar performances can be observed for the ensemble case strategy once the trend estimator is the same; so, to simplify the comparison, only the five best models of each strategy are ranked.

None of the eight estimators was top-ranked for the four predicted logs, although SVM and MLP exhibited overall better performance. Compared to other estimators, linear regression showed the lowest accuracy. Polynomial regression showed good performance if we consider its relatively low complexity. Boosting algorithms and Random Forests were expected to perform better considering their complexity formulation and hyperparameter tuning. For all the logs evaluated, both the proposed approaches feature decomposition and the ensemble methods outperformed the usual training procedure (base case). Feature decomposition as a feature expansion method achieved the best performance in predicting all logs, except RHOB, in which ensemble models performed better.

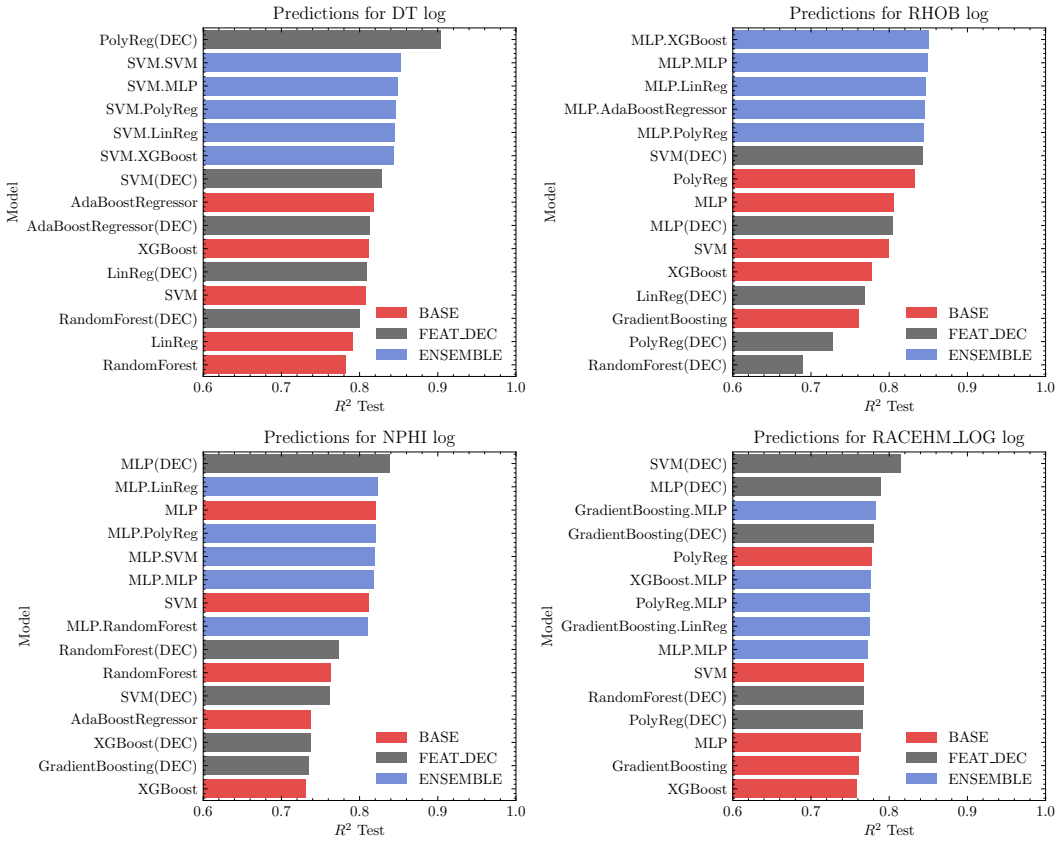


Figure 5.25: Method performance evaluation: Well logs predictions for well 15/9-F-1B (Test Set). Only the best 5 models from each case are shown.

The final predictions of the best base, feature decomposition and ensemble models, for the DT log, are shown in Figure 5.26. As the predictions are quite similar, the last track shows the errors $(y - \hat{y})$ in the predictions of the three models together. All models provided a good response to predict the DT data, although the base case Adaboost regressor has limitations in predicting the values in the range 3300-3325 m and the SVM.MLP ensemble model cumulates errors in the final section of the well (depth $>$ 3370 m), while the polynomial regression trained with the input of the feature decomposition presents better overall performance.

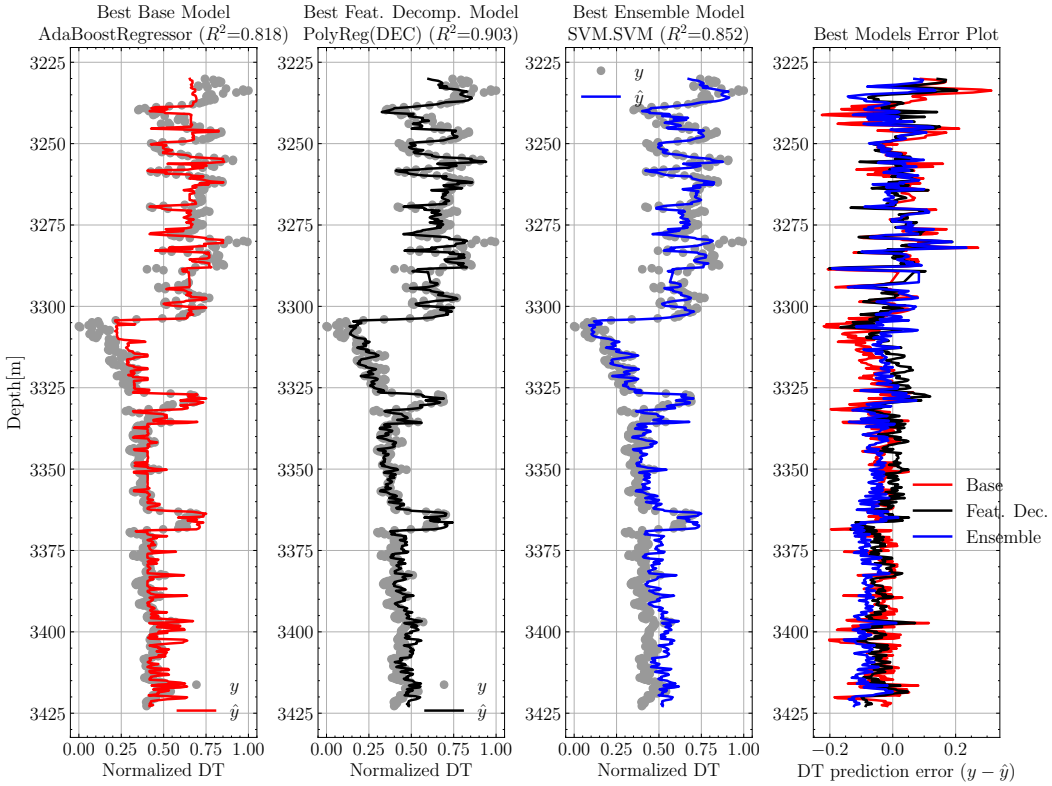


Figure 5.26: DT log predictions showing the best model of each case: base case, feature decomposition and ensemble.

The cross plot of y versus $\hat{y}$ presented in Figure 5.27 shows the best predictions for each log. Although all models adhere well to the measured data, some of them have problems predicting the extreme values of y, while the RHOB and NPHI predictors predict lower data values compared to the actual data points (y).

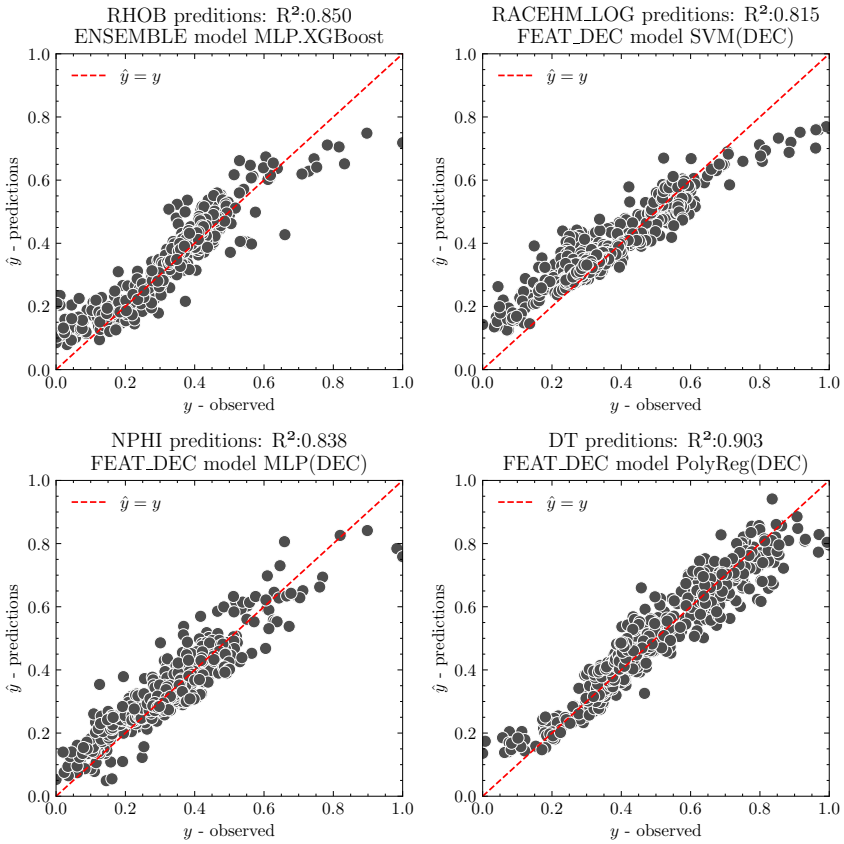


Figure 5.27: Predicted vs Actual: best models.

Finally, Table 5.12 summarizes the results of the metrics of the best model for each log and strategy. In all tasks evaluated, both strategies based on feature decomposition improved the predictions compared to the base case, ensuring a gain in correlation and a significant drop in RMSE. The most striking difference is observed for the DT log, where R^2 increases by 8.5% and a reduction of 27% in RMSE is achieved with the feature decomposition strategy.

Table 5.12: Summary of results. Best model trained for each well log and case.

Well Log	Case	Best Model	R^2 (train)	R^2 (test)	R^2 increase (test)	RMSE (train)	RMSE (test)	RMSE Drop (test)	Train Time (s)
RHOB	ENSEMBLE	MLP.XGBoost	0.95964	0.85010	1.7%	0.03610	0.05733	-5.2%	1.307
	FEAT_DEC	SVM(DEC)	0.91605	0.84355	1.0%	0.05206	0.05859	-3.0%	0.025
	BASE	PolyReg	0.90909	0.83341	-	0.05418	0.06046	-	0.003
RACEHM (resistivity)	FEAT_DEC	SVM(DEC)	0.91028	0.81479	3.7%	0.04834	0.06628	-8.7%	0.017
	ENSEMBLE	GradBoost.MLP	0.96646	0.78249	0.5%	0.02956	0.07183	-1.1%	2.821
	BASE	PolyReg	0.89717	0.77761	-	0.05175	0.07263	-	0.003
NPHI	FEAT_DEC	MLP(DEC)	0.87529	0.83806	1.8%	0.04980	0.05347	-5.0%	1.011
	ENSEMBLE	MLP.LinReg	0.85896	0.82269	0.2%	0.05296	0.05595	-0.6%	4.353
	BASE	MLP	0.84308	0.82039	-	0.05586	0.05631	-	1.633
DT	FEAT_DEC	PolyReg(DEC)	0.84608	0.90342	8.5%	0.08485	0.05728	-27.2%	0.007
	ENSEMBLE	SVM.SVM	0.84341	0.85244	3.4%	0.08558	0.07081	-9.9%	0.307
	BASE	AdaBoost	0.83346	0.81803	-	0.08826	0.07863	-	0.449

The train time of these specific object instances suggests that the ensemble models have a much higher cost in training compared to other cases. In fact, for all ensemble models, a randomized search with cross-validation is performed twice (for the trend and seasonal + residual models).

As in Table 5.12, the training time values are highly affected by the formulation and complexity of the distinct models, and it is difficult to come to such a conclusion. For that purpose, in Table 5.13, we present an overall train time analysis considering all logs and cases of mean fit time, which corresponds to the time used by the randomized search to train the 100 parameter possibilities. As there is no free lunch, the computational time increases for both proposed methods but is more significant for the ensemble strategy.

Table 5.13: Randomized search mean fit time considering 100 iterations, all models and all logs evaluated

Case	Mean Fit time (s)
Base	26.5
Ensemble	59.7
Feat. Decomp.	34.8

5.3.3 Discussion

The remarkable performance of the methods proposed herein compared to the baseline should be highlighted. Additionally, what particularly stands out is the performance of more simple algorithms such as SVM, polynomial regression, and even MLP when compared to more sophisticated methods such as boosting and tree ensembles.

Interestingly, we could also observe that choosing the better strategy depends on the output and the estimator, which shows the importance of testing various families of estimators that can perform better for different well logs. Furthermore, the combinations of input features that led to the best results were not repeated between the different logs tested, indicating that creating new uncorrelated features can lead to even better results.

An unexpected finding, as commented in Section 5.3.2.2 is that, given the study design for the ensemble case, it was inevitable that most of the seasonal + residual models obtained a similar increase in performance given a fixed trend estimator, since the latter accounts for the majority of the input information.

As a limitation, this study failed to find good models for Gamma-Ray (GR) log prediction. Additional research on this variable can be adopted, but this fact may suggest that GR is a preferable log to be acquired, once it was used for all four logs as input but difficult to predict as output. This is somehow related to Tatsipie and Sheng [78] where GR is presented as a single departure input for all log predictions. The water saturation log was not useful when used as an input. The reason for that is another question that remains unanswered.

Furthermore, these findings might not be representative of other types of reservoirs. Thus, future research is needed to test these methods in other data sets.

6
Conclusion

Accurately measuring the flow rates of each well is a crucial task in the oil and gas industry, as it allows operators to correctly allocate production, assess reservoir potential, and make investment decisions in the development phase. However, multiphase flow is a complex phenomenon, and real-time information can be difficult and expensive to acquire. This study provides new insights into the field of data-driven VFM, showing that simulating the production of a well is possible by using only standard measured commonly available sensor data and with quite good accuracy.

The system identification techniques applied in this study proved to be a valuable strategy to increase model performance, as the model order is an important parameter to be optimized in addition to the classical hyperparameters of the models. Another contribution in this research related to system identification is to provide a free simulation run prediction capable of predicting long-term historic production based only on measurements of sensor data and given only an initial condition. This means that we only use the initial conditions and obtain predictions by iterating previous ones.

Another important finding of this study is the encouraging results obtained by stacking ensemble models. The stacking of diverse base learners strengthens the performance of VFMs as each of the base learners can perform better for different instances of data. The stacking ensembles obtained considerably better performance compared to the other classes of models.

In summary, a reasonable strategy to achieve a good VFM model would be to combine heterogeneous base learners and optimize the order of the model. Stacking ensembles and system identification properly address those questions, respectively, and the findings of this article confirm the idea of Bikmukhametov and Jäschke that ensembles are a technique that deserves to be explored in research related to the rates prediction problem [35].

Additionally, this study points out that using a two-step training approach with residuals estimation can improve the performance of virtual flow metering estimators, especially in the case of simpler model architectures.

In actual oil fields, the findings of this research can help improve the rate estimates based on well sensors to serve as a digital twin for production separa-

tors or physical hardware multiphase flowmeters, reducing costs and improving the correct allocation of production to help understand reservoir characteristics to take better production management and investment decisions. The free simulation run approach is capable of estimating long-term historic production, which can be used to recover or fix incorrect production allocation between wells.

For the purpose of synthetic well logs generation, two approaches based on seasonal decomposition were proposed to improve model quality compared to standard supervised learning: feature decomposition and ensemble modeling. The case study is the prediction of actual porosity, compressional slowness, resistivity and bulk density well logs. The Volve field data set was selected to evaluate these approaches. Data from two wells (15/9-F-1A and 15/9-F-11A) were used to create and train the models, while a third well (15/9-F-1B) was used to validate them.

An exploratory analysis, based on correlation, was performed as an initial step to select variables of interest to be used as input. A variety of machine learning algorithms were evaluated in all the selected cases, among which SVM and MLP proved to be the most robust. Both proposed strategies, a feature decomposition that expands the input space into trend and seasonal-residual, and an ensemble-based model composed of estimators specialized in trend and seasonal-residual, showed a significant improvement compared to the baseline case for the four evaluated logs. Although the computational cost increases for these two approaches, the gains in performance show that these two techniques are recommended to be used, and a good selection of estimators is encouraged once the best-performing estimator changes according to the case.

6.1 Future Work

As future work in the case of the virtual flow metering models, we suggest adding a feature selection step to the current workflow. Bringing additional features, other than sensor measurements, to improve accuracy is also a goal for further research. Additionally, it is important to extend this method to other datasets, preferably to reservoirs with different geological characteristics. Furthermore, the flexibilities of stacking ensembles point us to further investigations in future works, such as stacking models of different orders or finding new combinations of base and meta-regressors. The study of the interpretability of the models, regarding lags, base estimators, or the input features' relevance, is also a subject to be addressed in future research.

Regarding the well logs estimation, further investigation on the effec-

tiveness of these methods can be applied to other reservoir lithologies and in different geological contexts, as well as a deeper investigation on the GR log should be conducted.

As previously mentioned, one major limitation of data-based methods is that the results may not be representative of other types of wells and reservoirs as each well has its specific characteristics that define it as a unique system in itself. To mitigate the need for extensive retraining and adaptation of models to different operational or geological environments, building general-purpose models, also known as foundation models [117], could be an alternative. Those models are trained on huge datasets so that they can be applied across a wide range of use cases and can later have some parameters retrained for a specific application or data.

Bibliography

[1] RUSSELL, S. J.; NORVIG, P. **Artificial intelligence: a modern approach**. Pearson, Upper Saddle River, New Jersey, 2016.

[2] NILSSON, N. J. **Artificial intelligence: a new synthesis**. Elsevier, Oxford, 1998.

[3] MITCHELL, T. M.; MITCHELL, T. M. **Machine learning**. 9. McGraw-hill, New York, 1997.

[4] BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**, volume 4. Springer, Berlin, Heidelberg, 2006.

[5] KOROTEEV, D.; TEKIC, Z. **Artificial intelligence in oil and gas upstream: Trends, challenges, and scenarios for the future**. Energy and AI, 3:100041, 2021.

[6] INKPEN, A. C.; MOFFETT, M. H. **The global oil & gas industry: management, strategy & finance**. PennWell, Tulsa, Oklahoma, 2011.

[7] ARINZE, C. A.; IZIONWORU, V. O.; ISONG, D.; DAUDU, C. D. ; ADEFEMI, A. **Predictive maintenance in oil and gas facilities, leveraging ai for asset integrity management**. International Journal of Frontiers in Engineering and Technology Research, 6(1):16–26, 2024.

[8] ORRÙ, P. F.; ZOCCHEDDU, A.; SASSU, L.; MATTIA, C.; COZZA, R. ; ARENA, S. **Machine learning approach using mlp and svm algorithms for the fault prediction of a centrifugal pump in the oil and gas industry**. Sustainability, 12(11):4776, 2020.

[9] BABAYEJU, O. A.; ADEFEMI, A.; EKEMEZIE, I. O. ; SOFOLUWE, O. O. **Advancements in predictive maintenance for aging oil and gas infrastructure**. World Journal of Advanced Research and Reviews, 22(3):252–266, 2024.

[10] DMITRIEVSKY, A.; SBOEV, A.; EREMIN, N.; CHERNIKOV, A.; NAUMOV, A.; GRYAZNOV, A.; MOLOSHNIKOV, I.; BOROZDIN, S. ; SAFAROVA, E. **On increasing the productive time of drilling oil and**

gas wells using machine learning methods. Georesursy= Georesources, 22(4):79–85, 2020.

[11] TARIQ, Z.; ALJAWAD, M. S.; HASAN, A.; MURTAZA, M.; MOHAMMED, E.; EL-HUSSEINY, A.; ALARIFI, S. A.; MAHMOUD, M. ; ABDULRAHEEM, A. **A systematic review of data science and machine learning applications to the oil and gas industry**. Journal of Petroleum Exploration and Production Technology, pages 1–36, 2021.

[12] NOSHI, C. I.; SCHUBERT, J. J. **The role of machine learning in drilling operations; a review**. In SPE eastern regional meeting, page D043S005R006. SPE, Pittsburgh, Pennsylvania, USA, 2018.

[13] SOOMRO, A. A.; MOKHTAR, A. A.; KURNIA, J. C.; LASHARI, N.; LU, H. ; SAMBO, C. **Integrity assessment of corroded oil and gas pipelines using machine learning: A systematic review**. Engineering Failure Analysis, 131:105810, 2022.

[14] AL-SABAEEI, A. M.; ALHUSSIAN, H.; ABDULKADIR, S. J. ; JAGADEESH, A. **Prediction of oil and gas pipeline failures through machine learning approaches: A systematic review**. Energy Reports, 10:1313–1338, 2023.

[15] RACHMAN, A.; ZHANG, T. ; RATNAYAKE, R. C. **Applications of machine learning in pipeline integrity management: A state-of-the-art review**. International journal of pressure vessels and piping, 193:104471, 2021.

[16] ODIMARHA, A. C.; AYODEJI, S. A. ; ABAKU, E. A. **Machine learning's influence on supply chain and logistics optimization in the oil and gas sector: a comprehensive analysis**. Computer Science & IT Research Journal, 5(3):725–740, 2024.

[17] WRONA, T.; PAN, I.; GAWTHORPE, R. L. ; FOSSEN, H. **Seismic facies analysis using machine learning**. Geophysics, 83(5):O83–O95, 2018.

[18] WANG, Z.; DI, H.; SHAFIQ, M. A.; ALAUDAH, Y. ; ALREGIB, G. **Successful leveraging of image processing and machine learning in seismic structural interpretation: A review**. The Leading Edge, 37(6):451–461, 2018.

[19] XU, C.; FU, L.; LIN, T.; LI, W. ; MA, S. **Machine learning in petrophysics: Advantages and limitations**. Artificial Intelligence in Geosciences, 3:157–161, 2022.

[20] MOHAMMADPOUR, M.; ROSHAN, H.; ARASHPOUR, M. ; MASOUMI, H. **Machine learning assisted kriging to capture spatial variability in petrophysical property modelling**. Marine and Petroleum Geology, 167:106967, 2024.

[21] DA SILVA BOMFIM, L.; SOARES, M. V. T.; VIDAL, A. C. ; PEDRINI, H. **Geological reservoir characterization tasks based on computer vision techniques**. Marine and Petroleum Geology, 173:107231, 2025.

[22] KORAY, A.-M.; BUI, D.; KUBI, E. A.; AMPOMAH, W. ; AMOSU, A. **Machine learning based reservoir characterization and numerical modeling from integrated well log and core data**. Geoenergy Science and Engineering, 243:213296, 2024.

[23] NIE, Y.; XIAN, C.; LUO, J.; ZHANG, J.; WANG, Y. ; LI, C. **Bagging machine learning algorithms for rapid identification, classification, evaluation and upscaling in unconventional reservoir**. Geoenergy Science and Engineering, 246:213545, 2025.

[24] PATIDAR, A. K.; SINGH, S.; ANAND, S. ; KUMAR, P. **Enhancing pvt property predictions for black oil reservoirs through the application of supervised machine learning techniques**. Geoenergy Science and Engineering, 243:213307, 2024.

[25] MOHAGHEGH, S. D. **Subsurface analytics: Contribution of artificial intelligence and machine learning to reservoir engineering, reservoir modeling, and reservoir management**. Petroleum Exploration and Development, 47(2):225–228, 2020.

[26] CHAI, Z.; NWACHUKWU, A.; ZAGAYEVSKIY, Y.; AMINI, S. ; MADASU, S. **An integrated closed-loop solution to assisted history matching and field optimization with machine learning techniques**. Journal of Petroleum Science and Engineering, 198:108204, 2021.

[27] GUO, Z.; SANKARAN, S. **Hybrid rgnet: A physics-based model enhanced by machine learning for history matching, characterization and future prediction**. Geoenergy Science and Engineering, 246:213597, 2025.

[28] ROSTAMIAN, A.; DE MORAES, M. B.; SCHIOZER, D. J. ; COELHO, G. P. **A survey on multi-objective, model-based, oil and gas field development optimization: Current status and future directions**. Petroleum Science, 22(1):508–526, 2025.

[29] KORAY, A.-M.; BUI, D.; AMPOMAH, W.; APPIAH KUBI, E. ; KLUMPENHOWER, J. **Application of machine learning optimization workflow to improve oil recovery**. In SPE Oklahoma City Oil and Gas Symposium/Production and Operations Symposium, page D021S003R001. SPE, Oklahoma City, 2023.

[30] GAGANIS, V.; VAROTSIS, N. **Machine learning methods to speed up compositional reservoir simulation (spe 154505)**. In 74th EAGE Conference and Exhibition incorporating EUROPEC 2012, page 293. European Association of Geoscientists & Engineers, Copenhagen, Denmark, 2012.

[31] WANG, K.; LUO, J.; WEI, Y.; WU, K.; LI, J. ; CHEN, Z. **Practical application of machine learning on fast phase equilibrium calculations in compositional reservoir simulations**. Journal of Computational Physics, 401:109013, 2020.

[32] ZHAO, H. **A reduced order model based on machine learning for numerical analysis: An application to geomechanics**. Engineering Applications of Artificial Intelligence, 100:104194, 2021.

[33] YAO, P.; YU, Z.; ZHANG, Y. ; XU, T. **Application of machine learning in carbon capture and storage: An in-depth insight from the perspective of geoscience**. Fuel, 333:126296, 2023.

[34] FALCONE, G. **Chapter 3 multiphase flow metering principles**. In G. Falcone; G. Hewitt; and C. Alimonti, editors, Multiphase Flow Metering, volume 54 of Developments in Petroleum Science, pages 33–45. Elsevier, 2009.

[35] BIKMUKHAMETOV, T.; JÄSCHKE, J. **First principles and machine learning virtual flow metering: A literature review**. Journal of Petroleum Science and Engineering, 184:106487, 2020.

[36] SANDNES, A. T.; GRIMSTAD, B. ; KOLBJØRNSEN, O. **Multi-task learning for virtual flow metering**. Knowledge-Based Systems, 232:107458, 2021.

[37] KANSHIO, S. **A review of hydrocarbon allocation methods in the upstream oil and gas industry**. Journal of Petroleum Science and Engineering, 184:106590, 2020.

[38] FOSS, B.; KNUDSEN, B. R. ; GRIMSTAD, B. **Petroleum production optimization–a static or dynamic problem?** Computers & Chemical Engineering, 114:245–253, 2018.

[39] ALAKEELY, A.; HORNE, R. **Simulating oil and water production in reservoirs with generative deep learning**. SPE Reservoir Evaluation & Engineering, 25(04):751–773, 2022.

[40] MERCANTE, R.; NETTO, T. A. **Virtual flow predictor using deep neural networks**. Journal of Petroleum Science and Engineering, 213:110338, 2022.

[41] ZANGL, G.; HERMANN, R. ; SCHWEIGER, C. **Comparison of Methods for Stochastic Multiphase Flow Rate Estimation**. In SPE Annual Technical Conference and Exhibition?, SPE Annual Technical Conference and Exhibition. Amsterdam, The Netherlands, 2014.

[42] OKOTIE, V.; HOLLAENDER, F.; YOUNES, M.; ZIDAN, M.; UIJTTENHOUT, M. ; AL-KAABI, F. **Multiphase flowmeter performance: A critical piece of an offshore well management toolkit**. In Abu Dhabi International Petroleum Exhibition & Conference. OnePetro, Abu Dhabi, 2016.

[43] FALCONE, G.; HEWITT, G.; ALIMONTI, C. ; HARRISON, B. **Multiphase flow metering: Current trends and future developments**. Journal of Petroleum Technology, 54(04):77–84, 2002.

[44] CAMPONOGARA, E.; PLUCENIO, A.; TEIXEIRA, A. F. ; CAMPOS, S. R. **An automation system for gas-lifted oil wells: Model identification, control, and optimization**. Journal of petroleum science and engineering, 70(3-4):157–167, 2010.

[45] VAN GISBERGEN, S. J. C. H. M.; VANDEWEIJER, A. A. H. **Reliability Analysis of Permanent Downhole Monitoring Systems**. SPE Drilling and Completion, 16(01):60–63, 2001.

[46] FREITAS, L.; BARBOSA, B. H. ; AGUIRRE, L. A. **Including steady-state information in nonlinear models: An application to the development of soft-sensors**. Engineering Applications of Artificial Intelligence, 102:104253, 2021.

[47] ELLIS, D. V.; SINGER, J. M. **Well logging for earth scientists**, volume 692. Springer, Dordrecht, Netherlands, 2007.

[48] SHAN, L.; LIU, Y.; TANG, M.; YANG, M. ; BAI, X. **Cnn-bilstm hybrid neural networks with attention mechanism for well log prediction**. Journal of Petroleum Science and Engineering, 205:108838, 2021.

[49] DABI, S.; VISHWAKARMA, A. ; MAITI, S. **Joint implementation of ensemble and deep learning regression techniques to predict missing density logs**. In International Petroleum Technology Conference, page D012S116R001. IPTC, Riyadh, Saudi Arabia, 2022.

[50] LI, J.; GAO, G. **Digital construction of geophysical well logging curves using the lstm deep-learning network**. Frontiers in Earth Science, 10:1041807, 2023.

[51] WOLPERT, D. H. **Stacked generalization**. Neural Networks, 5(2):241–259, 1992.

[52] AGUIRRE, L. A.; TEIXEIRA, B. O.; BARBOSA, B. H.; TEIXEIRA, A. F.; CAMPOS, M. C. ; MENDES, E. M. **Development of soft sensors for permanent downhole gauges in deepwater oil wells**. Control Engineering Practice, 65:83–99, 2017.

[53] SONG, S.; WU, M.; QI, J.; WU, H.; KANG, Q.; SHI, B.; SHEN, S.; LI, Q.; YAO, H.; CHEN, H. ; GONG, J. **An intelligent data-driven model for virtual flow meters in oil and gas development**. Chemical Engineering Research and Design, 186:398–406, 2022.

[54] ALAKEELY, A.; HORNE, R. **Simulating oil and water production in reservoirs with generative deep learning**. SPE Reservoir Evaluation & Engineering, pages 1–24, 2022.

[55] APIO, A.; DAMBROS, J. W.; FARENZENA, M. ; TRIERWEILER, J. O. **Comparison of kalman filter-based approaches for permanent downhole gauge pressure estimation in offshore oil production**. Journal of Petroleum Science and Engineering, 182:106254, 2019.

[56] SABAA, A.; ABU EL ELA, M.; EL-BANBI, A. H. ; SAYYOUH, M. H. M. **Artificial Neural Network Model to Predict Production Rate of Electrical Submersible Pump Wells**. SPE Production & Operations, 38(01):63–72, 2023.

[57] BIKMUKHAMETOV, T.; JÄSCHKE, J. **Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models**. Computers & Chemical Engineering, 138:106834, 2020.

[58] MANAMI, M.; SEDDIGHI, S. ; ÖRLÜ, R. **Deep learning models for improved accuracy of a multiphase flowmeter**. Measurement, 206:112254, 2023.

[59] MERCANTE, R.; NETTO, T. A. **Virtual Multiphase Flowmeter Using Deep Convolutional Neural Networks**. SPE Journal, pages 1–14, 2023.

[60] ALAKEELY, A.; HORNE, R. **Application of deep learning methods to estimate multiphase flow rate in producing wells using surface measurements**. Journal of Petroleum Science and Engineering, 205:108936, 2021.

[61] LIU, Y.; SHAN, L.; YU, D.; ZENG, L. ; YANG, M. **An echo state network with attention mechanism for production prediction in reservoirs**. Journal of Petroleum Science and Engineering, 209:109920, 2022.

[62] GRYZLOV, A.; MIRONOVA, L.; SAFONOV, S. ; ARSALAN, M. **Artificial intelligence and data analytics for virtual flow metering**. In SPE Middle East Oil and Gas Show and Conference. SPE, Sanabis, Bahrain, 2021.

[63] GRYZLOV, A.; SAFONOV, S. ; ARSALAN, M. **Intelligent Production Monitoring with Continuous Deep Learning Models**. SPE Journal, 27(02):1304–1320, 2022.

[64] ANDRADE, G. M.; DE MENEZES, D. Q.; SOARES, R. M.; LEMOS, T. S.; TEIXEIRA, A. F.; RIBEIRO, L. D.; VIEIRA, B. F. ; PINTO, J. C. **Virtual flow metering of production flow rates of individual wells in oil and gas platforms through data reconciliation**. Journal of Petroleum Science and Engineering, 208:109772, 2022.

[65] MA, H.; HAN, G.; ZHU, Z.; WANG, B.; XIANG, X. ; LIANG, X. **Hybrid virtual flow metering on arbitrary well patterns for transient multiphase prediction driven by mechanistic and data model**. Geoenergy Science and Engineering, 243:213335, 2024.

[66] PAULO, P. H.; PEREIRA, F. C. ; AYALA, H. V. **System identification techniques for soft sensors and multiphase flow metering**. IFAC-PapersOnLine, 58(15):538–543, 2024. 20th IFAC Symposium on System Identification SYSID 2024.

[67] HUANG, T.; QIAN, H.; HUANG, Z.; XU, N.; HUANG, X.; YIN, D. ; WANG, B. **A time patch dynamic attention transformer for enhanced well production forecasting in complex oilfield operations**. Energy, 309:133186, 2024.

[68] AL-QUTAMI, T. A.; IBRAHIM, R. ; ISMAIL, I. **Hybrid neural network and regression tree ensemble pruned by simulated annealing for virtual**

flow metering application. In 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), pages 304–309. IEEE, Sarawak, Malaysia, 2017.

[69] AL-QUTAMI, T. A.; IBRAHIM, R.; ISMAIL, I. ; ISHAK, M. A. **Virtual multiphase flow metering using diverse neural network ensemble and adaptive simulated annealing**. Expert Systems with Applications, 93:72–85, 2018.

[70] ZHANG, D.; YUNTIAN, C. ; JIN, M. **Synthetic well logs generation via recurrent neural networks**. Petroleum Exploration and Development, 45(4):629–639, 2018.

[71] SHI, M.; YANG, B.; CHEN, R. ; YE, D. **Logging curve prediction method based on cnn-lstm-attention**. Earth Science Informatics, 15:2119–2131, 2022.

[72] YANG, L.; WANG, S.; CHEN, X.; CHEN, W.; SAAD, O. M. ; CHEN, Y. **Deep-learning missing well-log prediction via long short-term memory network with attention-period mechanism**. GEOPHYSICS, 88(1):D31–D48, 2023.

[73] LIN, L.; WEI, H.; WU, T.; ZHANG, P.; ZHONG, Z. ; LI, C. **Missing well-log reconstruction using a sequence self-attention deep-learning framework**. GEOPHYSICS, 88(6):D391–D410, 2023.

[74] WU, L.; DONG, Z.; LI, W.; JING, C. ; QU, B. **Well-logging prediction based on hybrid neural network model**. Energies, 14(24), 2021.

[75] ZHENG, J.; XIE, K.; WEN, C.; SHENG, G.; HE, J. ; TIAN, H. **P-wave prediction method under multi-source spatiotemporal feature fusion and physics-informed neural network**. Geoenergy Science and Engineering, 223:211515, 2023.

[76] MEHRAD, M.; RAMEZANZADEH, A.; BAJOLVAND, M. ; HAJSAEEDI, M. R. **Estimating shear wave velocity in carbonate reservoirs from petrophysical logs using intelligent algorithms**. Journal of Petroleum Science and Engineering, 212:110254, 2022.

[77] WANG, J.; CAO, J.; FU, J. ; XU, H. **Missing well logs prediction using deep learning integrated neural network with the self-attention mechanism**. Energy, 261:125270, 2022.

[78] TATSIPIE, N. R.; SHENG, J. J. **Generating pseudo well logs for a part of the upper bakken using recurrent neural networks**. Journal of Petroleum Science and Engineering, 200:108253, 2021.

[79] ZENG, L.; REN, W.; SHAN, L. ; HUO, F. **Well logging prediction and uncertainty analysis based on recurrent neural network with attention mechanism and bayesian theory**. Journal of Petroleum Science and Engineering, 208:109458, 2022.

[80] TAMOTO, H.; DOS SANTOS GIORIA, R. ; DE CARVALHO CARNEIRO, C. **Prediction of nuclear magnetic resonance porosity well-logs in a carbonate reservoir using supervised machine learning models**. Journal of Petroleum Science and Engineering, 220:111169, 2023.

[81] OLIVEIRA, L. A. B. D.; CARNEIRO, C. D. C. **Synthetic geochemical well logs generation using ensemble machine learning techniques for the brazilian pre-salt reservoirs**. Journal of Petroleum Science and Engineering, 196:108080, 2021.

[82] WOLPERT, D.; MACREADY, W. **No free lunch theorems for optimization**. IEEE Transactions on Evolutionary Computation, 1(1):67–82, 1997.

[83] MCDONALD, G. C. **Ridge regression**. Wiley Interdisciplinary Reviews: Computational Statistics, 1(1):93–100, 2009.

[84] SMOLA, A. J.; SCHÖLKOPF, B. **A tutorial on support vector regression**. Statistics and computing, 14:199–222, 2004.

[85] TAUNK, K.; DE, S.; VERMA, S. ; SWETAPADMA, A. **A brief review of nearest neighbor algorithm for learning and classification**. In 2019 International Conference on Intelligent Computing and Control Systems (ICCS), pages 1255–1260. Madurai, India, 2019.

[86] HAYKIN, S. **Neural networks: a comprehensive foundation**. Prentice Hall PTR, Upper Saddle River, NJ, United States, 1998.

[87] KINGSFORD, C.; SALZBERG, S. L. **What are decision trees?** Nature biotechnology, 26(9):1011–1013, 2008.

[88] BREIMAN, L.; FRIEDMAN, J.; OLSHEN, R. A. ; STONE, C. J. **Classification and regression trees**. Routledge, Boca Raton, FL, 2017.

[89] BREIMAN, L. **Bagging predictors**. Machine learning, 24:123–140, 1996.

[90] HO, T. K. **Random decision forests**. In Proceedings of 3rd International Conference on Document Analysis and Recognition, volume 1, pages 278–282 vol.1. Montreal, Canada, 1995.

[91] PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M. ; DUCHESNAY, E. **Scikit-learn: Machine learning in Python**. Journal of Machine Learning Research, 12:2825–2830, 2011.

[92] GEURTS, P.; ERNST, D. ; WEHENKEL, L. **Extremely randomized trees**. Machine learning, 63:3–42, 2006.

[93] FRIEDMAN, J. H. **Greedy function approximation: A gradient boosting machine**. The Annals of Statistics, 29(5):1189–1232, 2001.

[94] KE, G.; MENG, Q.; FINLEY, T.; WANG, T.; CHEN, W.; MA, W.; YE, Q. ; LIU, T.-Y. **Lightgbm: A highly efficient gradient boosting decision tree**. Advances in neural information processing systems, 30, 2017.

[95] CHEN, T.; GUESTRIN, C. **Xgboost: A scalable tree boosting system**. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pages 785–794. San Francisco, CA, USA, 2016.

[96] KOHONEN, T. **The self-organizing map**. Proceedings of the IEEE, 78(9):1464–1480, 1990.

[97] MAATEN, L. V. D.; HINTON, G. **Visualizing data using t-sne**. Journal of machine learning research, 9(Nov):2579–2605, 2008.

[98] SÖDERSTRÖM, T.; STOICA, P. **System identification**. Prentice-Hall international series in systems and control engineering. Prentice Hall, Saddle River, NJ, USA, 1989.

[99] BILLINGS, S. A. **Nonlinear system identification: Narmax methods in the time, frequency, and spatio-temporal domains**. John Wiley & Sons, West Sussex, United Kingdom, 2013.

[100] LEONTARITIS, I. J.; BILLINGS, S. A. **Input-output parametric models for non-linear systems part i: deterministic non-linear systems**. International journal of control, 41(2):303–328, 1985.

[101] MENA-NEGRETE, J.; VALDIVIEZO-MIJANGOS, O. C.; NICOLÁS-LÓPEZ, R. ; COCONI-MORALES, E. **Characterization of elastic moduli with anisotropic rock physics templates considering mineralogy, fluid, porosity, and pore-structure: A case study in volve field, north sea**. Journal of Applied Geophysics, 206:104815, 2022.

[102] VOLLSET, J.; DORÉ, A. G. **A revised triassic and jurassic lithostratigraphic nomenclature for the norwegian north sea**. Oljedirektoratet, Stavanger, Norway, 1984.

[103] KIEFT, R.; JACKSON, C.-L.; HAMPSON, G. ; LARSEN, E. **Sedimentology and sequence stratigraphy of the hugin formation, quadrant 15, norwegian sector, south viking graben**. In Geological Society, London, Petroleum Geology Conference Series, volume 7, pages 157–176. The Geological Society of London London, London, 2010.

[104] **Field: Volve**. https://www.norskpetroleum.no/en/facts/field/volve, 2023. Accessed: 2024-01-11.

[105] TRIVEDI, A. **What information is available in the volve dataset?** https://discovervolve.com/2020/04/02/__how_to_access_volve/, 2020. Accessed: 2023-01-11.

[106] LI, Y.; SUN, R. ; HORNE, R. **Deep learning for well data history analysis**. In SPE Annual Technical Conference and Exhibition. SPE, Calgary, Alberta, Canada, 2019.

[107] STORKAAS, E.; SKOGESTAD, S. **Controllability analysis of two-phase pipeline-riser systems at riser slugging conditions**. Control Engineering Practice, 15(5):567–581, 2007.

[108] DA COSTA PEREIRA, F. **Virtual flow metering using stacking ensemble techniques and nonlinear system identification - source files**. https://github.com/prj-phcp/SYSID_Trabalho/tree/master/Volve-Dataset, 2025. Accessed: 2025-06-25.

[109] KUHN, M.; JOHNSON, K. **Applied predictive modeling**, volume 26. Springer, New York, USA, 2013.

[110] **Api reference - scikit-learn 1.2.0 documentation**. https://scikit-learn.org/stable/modules/classes.html, 2023. Accessed: 2023-01-11.

[111] DE SOUSA, D. H. B.; LOPES, F. R.; do Lago, A. W.; MEGGIOLARO, M. A. ; AYALA, H. V. H. **Hybrid gray and black-box nonlinear system**

identification of an elastomer joint flexible robotic manipulator. Mechanical Systems and Signal Processing, 200:110405, 2023.

[112] PIRES, I.; AYALA, H. V. H. ; WEBER, H. I. **Nonlinear ensemble gray and black-box system identification of friction induced vibrations in slender rotating structures**. Mechanical Systems and Signal Processing, 186:109815, 2023.

[113] DO LAGO, A. W. C.; DE SOUSA, D. H. B.; DOMINGUES, P. H.; DANEKER, M.; LU, L. ; Hultmann Ayala, H. V. **Physics-informed and black-box identification of robotic actuator with a flexible joint**. IFAC-PapersOnLine, 58(15):259–264, 2024. 20th IFAC Symposium on System Identification SYSID 2024.

[114] DA COSTA PEREIRA, F. **Well logs estimation using multimodel strategies and feature expansion based on seasonal decomposition - source files**. https://github.com/felipecostapereira/SYSID_well_logs, 2025. Accessed: 2025-06-25.

[115] CLEVELAND, R. B.; CLEVELAND, W. S.; MCRAE, J. E. ; TERPENNING, I. **Stl: A seasonal-trend decomposition**. Journal of Official Statistics, 6(1):3–73, 1990.

[116] CLEVELAND, W. P.; TIAO, G. C. **Decomposition of seasonal time series: a model for the census x-11 program**. Journal of the American statistical Association, 71(355):581–587, 1976.

[117] BOMMASANI, R.; HUDSON, D. A.; ADELI, E.; ALTMAN, R.; ARORA, S.; VON ARX, S.; BERNSTEIN, M. S.; BOHG, J.; BOSSELUT, A. ; BRUNSKILL, E. **On the opportunities and risks of foundation models**. arXiv preprint arXiv:2108.07258, 2021.