UNIVERSIDADE FEDERAL DO CEARΞ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUPMICA
PROGRAMA DE GRADUA¤ űO EM ENGENHARIA QUPMICA

KLAUS FEIO SOARES RICHARD

USE AND IMPROVEMENT OF MACHINE LEARNING IN ADSORPTION RESEARCH AND PROCESSES

FORTALEZA
2024

KLAUS FEIO SOARES RICHARD

USE AND IMPROVEMENT OF MACHINE LEARNING IN ADSORPTION RESEARCH AND PROCESSES

Tese apresentada ao Curso de Engenharia Quºmica da Universidade Federal Do Cear®, como requisito parcial para obten´²o do tºtulo de Doutor em Engenharia Quºmica. Ξrea de concentra´²o: Processos Quºmicos e Bioquºmicos.

Orientador: Prof. Dr. Mois¶ıs Bastos Neto
Coorientadora: Profa. Dra. Diana Cristina Silva de Azevedo

FORTALEZA

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

R385u Richard, Klaus Feio Soares.
Use and improvement of machine learning in adsorption research and processes / Klaus Feio Soares Richard. – 2024.
103 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Química, Fortaleza, 2024.
Orientação: Prof. Dr. Moisés Bastos Neto.
Coorientação: Profa. Dra. Diana Cristina Silva de Azevedo.

1. Adsorção. 2. Machine Learning. 3. Modelagem. 4. Otimização. I. Título.

CDD 660

KLAUS FEIO SOARES RICHARD

USE AND IMPROVEMENT OF MACHINE LEARNING IN ADSORPTION RESEARCH AND PROCESSES

Thesis presented to the department of Chemical engineering of Universidade Federal do Cear®, as a partial requisite to obtain the Doctorate degree in Chemical Engineering. Concentration Area: Chemical and Biochemical Processes.

Approved at: DD/MM /AAAA

THESIS COMMITTEE

_____

Prof. Dr. Mois¶s Bastos Neto (Advisor)

Universidade Federal do Cear®(UFC)

_____

Prof. Dr. Diana Cristina Silva de Azevedo (Co-advisor)

Universidade Federal do Cear®(UFC)

_____

Prof. Dr. Andreia Pereira da Silva

Universidade Federal do Cear®(UFC)

_____

Prof. Dr. Sebast² o Mardonio Pereira de Lucena

Universidade Federal do Cear®(UFC)

_____

Prof. Dr. Argimiro Resende Secchi

Universidade Federal do Rio de Janeiro (UFRJ)

_____

Prof. Dr. M®cio Andr¶ Fernandes Martins

Universidade Federal da Bahia (UFBA)

For you:

My Bride and Love, Nayade

Mother, Suzi Cristina.

Sister, Raíssa.

Nephews, Laura e Johann.

Goddaughter, Maria Esther.

ACKNOWLEDGEMENTS

To N®yade Santos, my bride and great love, for all the time together with me, making me smile and happy.

To my dear mother, Suzi Cristina, for the protection, great amounts of love and wisdom given to me to be prepared for life.

To my sister, Raºssa, for reminding of not taking myself too seriously all the time.

To my family, for the ever present support.

To CAPES, for the funding of my Graduation study.

To my advisors, Prof. Dr. Mois¶s Bastos Neto and Prof. Dr. Diana Cristina Silva de Azevedo, for guidance and for pushing me into new challenges.

To Prof. MSc. Antonio Eurico Belo Torres and Rafael Morales-Ospino for laying down the foundations to build my work upon.

To Prof. Dr. Arvind Rajendran of University of Alberta, Canada, for helping me solve the multiple problems faced during this work and providing crucial infrastructure to perform all the computational simulations.

To my colleagues of LPACO_2 for all the cooperation.

`For every complex problem there is an answer
that is clear, simple, and wrong._
- H. L. Mencken

RESUMO

Este trabalho investiga mßltiplos m¶todos de aprendizado de m®quina para uma aplica´²o de modelagem substituta e otimiza´²o de um modelo de adsor´²o por oscila´²o de press²o validado experimentalmente, avaliando sua exatid²o e precis²o em compara´²o com o m¶todo mais amplamente utilizado de Redes Neurais Artificiais. Al¶m disso, foram explorados alguns meios de melhorar a precis²o dos modelos de aprendizado de m®quina com conhecimento e par°metros dispon°veis do processo, seguido pela otimiza´²o dos par°metros de Pureza e Recupera´²o do sistema, finalizando com uma quantifica´²o do tempo computacional total empregado. Todas as etapas descritas foram desenvolvidas e finalizadas com sucesso utilizando a linguagem de programa´²o Python de cÆdigo aberto e os resultados esperados e inesperados foram discutidos e a otimiza´²o foi finalizada e ampliada.

Palavras-Chave: Adsor´²o; Machine Learning; Otimiza´²o; Modelagem;

ABSTRACT

This work investigates multiple methods of machine learning for an application of surrogate modelling and optimization of an experimentally validated Pressure Swing Adsorption model, evaluation their accuracy and precision compared to the more widely use method of Artificial Neural Networks. In addition, some means of improving the machine learning models accuracy with at-hand process knowledge and parameters were explored, which was followed by the optimization of the Purity and Recovery parameters of the system, finishing with a quantification of the total computational time employed. All steps described were developed and finished successfully using the open source Python programming language and the expected and unexpected results were discussed and the optimization was finalized and expanded.

Keywords: Adsorption; Machine Learning; Optimization; Modelling;

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

侮	Affinity parameter, isotherm models (1/Pa)
侮麻	Temperature-independent specific affinity parameter (Pa^{-1})
嫣	Denotes array element, vector or matrix.
価	BET Concentration parameter (mol/m^3)
価	CO_2 concentration ($mol\ m^{-3}$)
価	N_2 concentration ($mol\ m^{-3}$)
侵	Heat capacity of adsorbent ($J\ kg^{-1}\ K^{-1}$)
侵	Heat capacity of the column wall ($J\ kg^{-1}\ K^{-1}$)
価	Equilibrium concentration (mol/m^3)
価	BET model saturation concetration (mol/m^3)
価	Initial fluid concentration, ZLC model (mol/m^3)
侵	Mass heat capacity (J/kg.K)
侵	Molar heat capacity (J/mol.K)
価	Fluid concentration of component `x_ (mol/m^3)
価	Fluid concentration (mol/m^3)
侖	Convective derivative.
侖	Inner diameter of the column (m)
侖	Outer diameter of the column (m)
侖	Overall diffusion coefficient (m^2/s)
侖	Micropore diffusion coefficient at infinite temperature. (m^2/s)
侖	Molecular diffusion coefficient (m^2/s)
侖	Knudsen diffusion coefficient (m^2/s)
侖	Axial dispersion (m^2/s)
侖	Sorbate diffusivity (m^2/s)
侖	Thermodynamic Sorbate diffusivity (m^2/s)
侖	Intracrystaline diffusivity (m^2/s)
侖	Intracrystaline Themodynamic diffusivity (m^2/s)
侖	Effective diffusivity (m^2/s)
侖	Macropore diffusion coefficient (m^2/s)
侖	Diffusion coefficient due to surface diffusion (m^2/s)
侖	Diffusion coefficient due to plug flow (m^2/s)

侗	Activaction Energy (J)
偏攻	Fixed bed porosity
偏屹	Particle porosity
侃	Diffusivity temperature function
侳	Gravitational acceleration constant (m s^{-2})
兮洵	Enthalpy of Adsorption (J mol^{-1})
俱洵	Isosteric Enthalpy of Adsorption (kJ mol^{-1})
÷	Hidden layer cumulative sum.
俗	Sorbate flux (mol/m²s)
供	Intercept constant of linear models.
妯	Denotes a property from component i, j etc.
侷	Thermal conductivity (W m^{-1} K^{-1})
偏	Film Mass transfer coefficient (s^{-1})
偏	Gas thermal conductivity (W m^{-1} K^{-1})
偏屹	Flowrate calculation parameter
偏攻	Column wall thermal conductivity (W m^{-1} K^{-1})
依	Freundlich equilibrium constant
依	Henry equilibrum constant
偏	Linear Driving force constant (s^{-1})
伽	Bed length (m)
伽	Column dead volume (m)
伽	Spatial parameter at ZLC model
佢	ZLC column length (m)
侏	Molar mass of fluid (g/mol)
侹	Equilibrium adsorbed mass (kg)
侹	Adsorbent mass (kg)
侹	Adsorbed mass at a given time (kg)
侈	Number of mols of nitrogen gas entering the column (s^{-1})
侈	Number of mols of nitrogen gas leaving the column (s^{-1})
侠	Molar flux of component `x_ (mol/m²s)
t	MLP output example
侸	Fluid Viscosity (Pa.s)
侸	Chemical potential (J)

佣eq	Equilibrium Pressure at experimente (Pa)
佣out	Outlet Column pressure (Pa)
佟i	Partial pressure of sorbate (Pa)
佣	Thermodynamic Pressure (Pa)
佣prod	N_2 Productivity (mol h^{-1} kg^{-1})
佣pur	N_2 Purity
偌	Energy dissipation function (W/m3)
偶	Specific adsorbed concentration (mol/kg)
偶max	BET model maximum adsorbed concentration parameter (mol m^{-3})
偶eq	Specific equilibrium adsorbed concentration (mol kg^{-1})
偶i	Equilibrium adsorbed concentration (mol m^{-3})
偶ave	Averaged bed adsorbed concentration (mol m^{-3})
偶m	Maximum adsorbed concentration, isotherm models (mol m^{-3})
偏	Volumetric rate of generation of heat in the control volume (W m^{-3})
偏ads	Heat transferred due to adsorption (J)
偏	Enthalpy parameter of affinity coefficients (kJ mol^{-1})
偶	Adsorbed concentration (mol m^{-3})
偺	Gas constant (J mol^{-1} K^{-1})
偺p	Particle radius (m)
偺	Dimensionless radial coordinate, micropore diffusional model
偺c	Crystal radius of the sorbent (m)
偺w	Radial coordinate for the conduction at calorimeter wall (m)
偺	Volumetric rate of generation of component `x_
偺rec	Recovery of N_2
偺	Radial coordinate, diffusion models (m)
偠b	Bed density (kg m^{-3})
偠w	Wall density (kg m^{-3})
偠p	Adsorbent particle density (kg m^{-3})
偠	Generic fluid density (kg m^{-3})
做	Cauchy stress tensor (Pa)
侦in	Column inner initial temperature (K)
侦ext	External temperature (K)
侵ads	Time of the adsorption step (s)

俑区	Time of the pressurization step (s)
俑贩	Stoichiometric Time (s)
俑现贩	Total Skarstrom cycle time (s)
侦	Generic Temperature (K)
俄	Time (s)
侧	Gas internal energy (J)
俣	Velocity vector (m/s)
俣吝	Sorbate velocity (m/s)
侏娟	Volumetric flowrate (mE/s)
係	Fluid velocity ($m\ s^{-1}$)
係	MLP output constant example
促	Angular coefficients (weights) of linear machine learning models
促喃贩	Intermediate weight between hidden layer and output of MLP models
俄	Machine learning model variable (feature)
俅香	Gas inlet composition
俅杨	Fraction of fluid molecules on the solid surface
俅屹	Fraction of fluid molecules in the pore surface
徐	Axial Coordinate, Chromatographic column model (m)
Subscript	
□□□嫩 娲	number under letter, denotes multiple instances of a parameter
俩	Carbon dioxide
侳	Gas concentration, PSA model
侵	Inner
俸娄	inlet
侹俐我	maximum
侹俸娄	minimum
侠	Nitrogen
悦	Outer
俩娱妥	Outlet

$\tilde{p}$ Scaled

$\tilde{y}$ Total gas concentration, PSA model

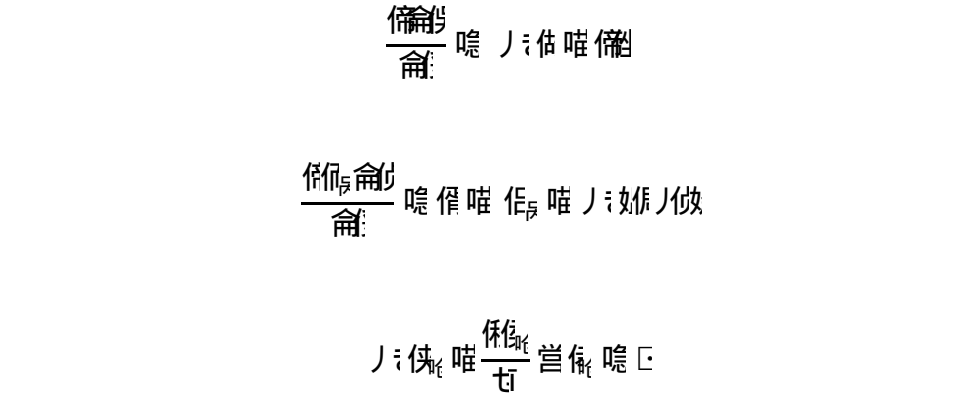
$\tilde{z}$ Wall

SUMMARY

1 INTRODUCTION

1.1 Adsorption and Surrogate Modelling.

Adsorption is a phenomenon where one or more components of a phase concentrate spontaneously in the surface of a solid. This process, by its nature is composed of two transport phenomena mechanisms, mass, through the concentration of the component, and heat by the change of energy through the phase transition. Thus, the modelling of adsorption has to take into account at least two equations to describe the change of properties, mass balance and energy balance (Bird et al., 2007), and if in the system of interest has also movement of fluid, an additional momentum balance should be added to the model. The general form for the Momentum, Mass and Energy balance are described, respectively, by Equations 1.1, 1.2 and 1.3.



As most of the practical uses of adsorption involves movement of fluid (Ruthven, 1984), these three equations most likely will be present in an accurate model. Sometimes it is possible to apply simplifications to the system, such as considering an instantaneous transfer between the free phase and the adsorbed phase, thus removing the dynamic mass balance equation, only requiring models for thermodynamic equilibrium (Do, 1998). However, in common industrial processes of adsorption such as Pressure Swing Adsorption, PSA, such simplification are not so easily applicable and due to the large scale fluid motion, the modelling also has to employ Computational Fluid Dynamics, CFD, models, which are often computationally costly (Anderson, 1995).

With these complex equations, often the application and optimization of adsorption processes run into the `Curse of dimensionality_ (Bellman, 2010) which states that, for each additional parameter to be optimized, the number of simulations increase exponentially, so a

model that runs once a time of the order of 10 h, the optimization of n parameters would require 10^{n+1} hours, which quickly reaches a time of years for the optimization to be complete, thus being impracticable.

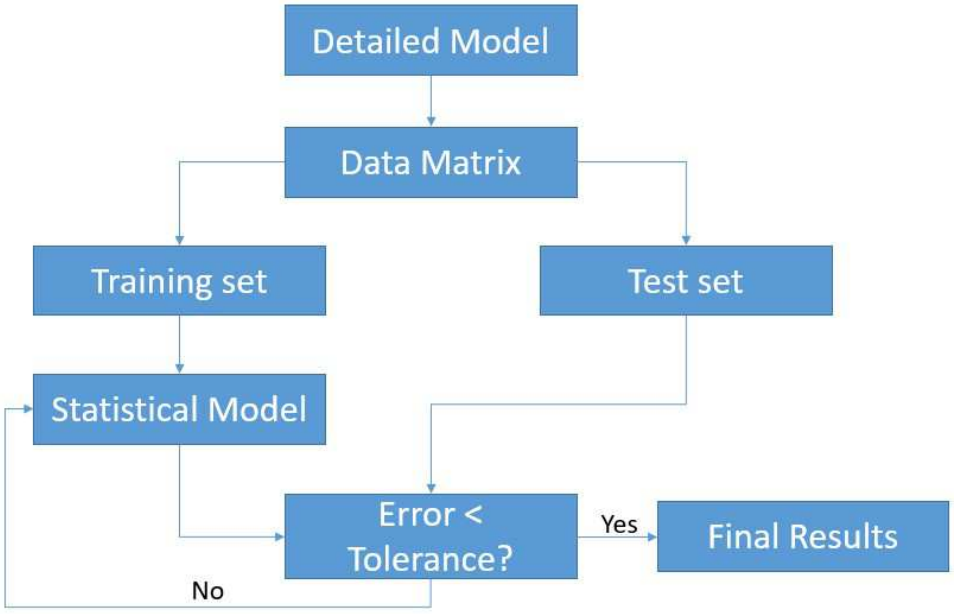
Due to this problem, in engineering it˘s often employed what is called Surrogate modelling (Forrester et al., 2008), which adds a simpler, Surrogate model, to model the system of interest. The general model is only employed to offer a representative sample of data for the process, and the Surrogate is tuned to generate results that are within an acceptable margin of error in comparison to the rigorous model, thus reducing the computational time required.

1.2 Machine Learning.

Machine learning is a field that employs statistical models computationally with the goal of general prediction and optimization. To apply the models, it is necessary to gather data organized into `targets_ the dependent variables and `features_ the independent values. The full dataset is then split into the training set, that is fed to the model which fits appropriate parameters for regression or classification, and the test set, which is used to validate the model. (Hastle et al., 2009). The field of statistical learning has seen an increased growth in recent years, exhibiting considerable versatility, being able to predict medical conditions to market prices (Mæller and Guido, 2017).

These models can be used as surrogates, as most of them employ simple statistical techniques for regression and classification, not being too much time consuming, but being devoid of any physical theory, they can only be used purely for quantitative analysis and can˘t give theoretical information about the predicted system. The use of machine learning as a surrogate is simply that the training and test datasets would be generated by the detailed model, Figure 1 shows a brief description of the use of statistical learning as a surrogate model for modelling and optimization.

Figure 1.1 ¯ Machine Learning as surrogate model.



Source : Author

There is a wide variety of models that use different approaches for prediction, the simplest is K-Nearest neighbors and most complex is Artificial Neural Networks, also know more recently as deep learning. Each model has one or more parameters that are tuned through a grid search, until a best value is achieved, for the K-Nearest neighbors, the parameter `K _ is the number of neighbors, a non-zero natural number.

1.3 Objectives

The general objective of this work is to use and evaluate Machine Learning as a surrogate model for the optimization of the Purity and Recovery of Nitrogen gas (N_2) that is produced by the separation process of a mixture of CO_2 and N_2 via Pressure Swing Adsorption, while looking for ways to introduce process knowledge in the statistical models.

As specific objectives, these are:

- ¿ Evaluate model accuracy
- ¿ Evaluate the effects of scaling in model accuracy
- ¿ Evaluate model sensitivity
- ¿ Analyze if there are ways of improving the models with process knowledge
- ¿ Choose most accurate and convenient model
- ¿ Obtain optimal values for purity and recovery
- ¿ Filter the results based on practical importance

2 LITERATURE REVIEW

2.1 Recent research and historical context

The interest in the optimization of PSA processes was always present in the literature, going back to works from Nilchan (1997), which uses a phenomenological model to construct a non-linear programing optimization problem solved with a sequential quadratic programming algorithm, applied to a 2-bed model in a 4-stage cycle. Barg (2000) did the modelling and optimization of an industrial PSA unit with 6 beds and 3 layers of different adsorbents for the purification of hydrogen gas, dealing with a problem with significantly more mathematical complexity.

More works, such as Jiang et al. (2005) using successive square programming for the optimization over some case studies of VSA processes, Agarwal et al. (2008) which uses a surrogate reduced order model applied to a two-bed four step PSA process for the separation of hydrogen gas and methane and Boukouvala et al. (2017), which expand the optimization techniques over a class of grey box models, show that there is a consistent relevance of the subject in the literature of the last three decades. A common ground over all of the cited works is the emphasis in using optimization methods that provide viable computational times required.

When surrogate modelling is included, in general, the surrogates are used for optimization of performance variables, such as gas purity, recovery and energy cost. Hasan et al. (2011) employed a surrogate-based optimization using a synergistic combination of a traditional adsorption model with Design and Analysis of Computer experiments, DACE. The relevant variables for optimization were Energy, Economic, Environmental and Capture costs of Carbon Dioxide.

Beck et al. (2015), also uses surrogate modelling for optimization of VPSA systems, the used surrogate is known as Kriging, a statistical interpolation technique modeled as a Gaussian process, the optimization parameters were Purity, recovery and molar work. The previous articles both discuss the necessity of fast converging models for tractable optimizations, but not many of them specifically deals with Machine learning or data science. These findings show that even with the rapid evolution of computer software and hardware, the computational times for simulations and the curse of dimensionality are still ever-present problems and also shows how new the field of data science and machine learning is and how it expanded.

To further elucidate this, Beck et al. (2016) discusses the importance of the field of data science for chemical engineering, it discusses the many benefits of the field, and explains the different types of learning, such as the supervised and unsupervised learning. The possible applications range from chemical reaction modelling, nanoscale modelling, synthetic biology and many more. The highlights for the potential of the models is explained through the use of artificial neural networks, the most powerful method of statistical learning.

With the context of the novelty of machine learning, the application in adsorption is reflected by a very recent rise in published research. Ye et al. (2019) uses neural networks to optimize the process of purifications of hydrogen gas through PSA, the machine learning model is applied in MATLAB and the predicted parameters were purity and recovery. Similar works were published by Ma et al. (2019) which applies ANN˘s for hydrogen purification and Leperi et al. (2019) for CO_2 capture. Specifically dealing with optimization using machine learning models, Sant Anna et al. (2017) uses ANN˘s to model and optimize and single-bed VPSA process for the separation of CH_4 and N_2.

More recent works also couple the machine learning modelling and optimization with molecular simulation, providing a computational treatment of the multiple parts of the adsorption applications, works from Bobbitt and Snurr (2019) and Burns et al. (2020) integrate molecular simulation and artificial neural networks for screening of Metal organic frameworks, MOFs and process optimization.

The most data science intensive work comes from Pai et al. (2020) which applies machine learning methods for experimentally validated models of pressure swing adsorption processes. The paper investigates the impact of the training set sizes in the accuracy of the predictions for performance variables for multiple statistical models and uses artificial neural networks to evaluate the impact of the training set sizes for the concentration profiles of the PSA columns, as this variable is a vector, only ANN˘s are possible for the predictions.

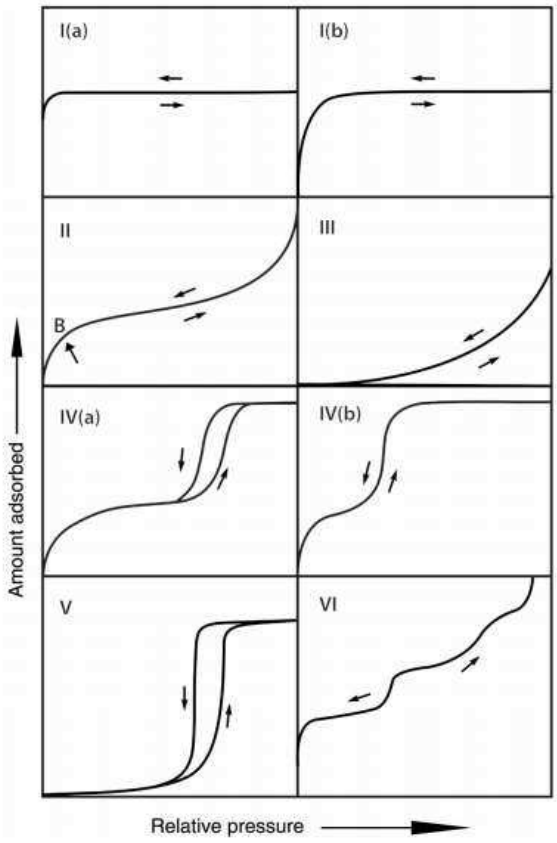
2.2 General Adsorption Fundamentals

Adsorption is a physical or chemical phenomenon where a chemical species in a fluid concentrates itself on the surface of a solid, spontaneously, due to a chemical potential gradient. This process reduces the concentration of the compound in the fluid until a thermodynamic equilibrium is reached. The adsorbed species is the adsorbate and the solid is the adsorbent. Adsorbents have different affinities for different compounds, some are attracted to the surface more than others are, and this difference is the principle of the use of adsorption to separate

different chemicals. Detailed theoretical description of the phenomenon can be found in Ruthven (1984) and Do (1998).

The amount of a compound, which is adsorbed by a solid, in equilibrium, can be correlated with bulk fluid concentration, for liquid adsorbates, and partial pressure, for gaseous adsorbates. These correlations are constructed in graphs known as adsorption or equilibrium isotherms. Different adsorbent-adsorbate systems will exhibit different forms of isotherms; the most recent report of IUPAC (Thommes et al., 2015) discusses these types of curves, which are shown in Figure 2.1.

Figure 2.1 – Types of adsorption isotherms



Source : Thommes et al. (2015)

2.2.1 Equilibrium Models

Equilibrium models for adsorption, also known as adsorption isotherms, are equations that are meant to appropriately correlate the adsorbed quantity of a species with a measurable property, such as concentration, in a fixed temperature. Some models are derived from theory, such as the Henry Law limit for low concentrations, and the Langmuir Isotherm, which assumes

that the adsorption happens in a monolayer and that the solid is energetically homogeneous, most models, however, are semi-empirical.

Table 2.1 lists some models that were reviewed by Foo and Hameed (2010). In principle, it is possible to choose a model based on the adsorbate-adsorbent system properties, but generally, as discussed in Limousin et al. (2007), there is some trial and error. The choice escalates from trying to fit simpler models to more complex one by addition of parameters and modification of the equations to improve the model fit to the experimental data.

Table 2.1 – Some Adsorption Isotherm Models.

Isotherm	Equation	Application Example	Reference
Henry Law	$q_e = K_H C_e$	Low concentration sorption	Classical Thermodynamics
Langmuir	$q_e = \frac{q_m K_L C_e}{1 + K_L C_e}$	Methylene blue by Activated carbon	Langmuir (1916)
Freundlich	$q_e = K_F C_e^{1/n}$	Ammonium by Zeolites	Freundlich (1906)
Sips	$q_e = \frac{q_m K_S C_e^{1/n}}{1 + K_S C_e^{1/n}}$	CO_2 by activated carbon	Sips (1948)
Toth	$q_e = \frac{q_m K_T C_e}{\left(1 + K_T C_e^{1/n}\right)^{n}}$	H_2 by activated carbon	Toth (1971)
BET	$q_e = \frac{q_{max} K_{BET} C_e}{(C_s - C_e)\left[1 + (K_{BET} - 1)\frac{C_e}{C_s}\right]}$	Nitrogen at 77K in various sorbents	Brunauer et al (1938)

Source: Modified from Foo and Hameed (2010)

2.2.2 Kinetic models and diffusion mechanisms

The transport of chemical species through the adsorbent pores can by described by a diffusional process which has the driving force from the chemical potential as first recognized by Einstein (1906). The form of the energy balance of a differential element of diffusion is given by Equation 2.1

$$J_i = -D_i \frac{\partial C_i}{\partial x} \qquad (2.1)$$

Applying the thermodynamic definition of chemical potential and applying the definition of flux, Equation 2.2 is derived.

$$J_i = -D_i \frac{C_i}{RT}\frac{\partial \mu_i}{\partial x} \tag{2.2}$$

Diffusivity can be defined by the following terms of Equation 2.3

$$D_{ie} = D_i \frac{C_i}{RT}\frac{\partial \mu_i}{\partial \ln C_i} \tag{2.3}$$

Which can be rearranged to express the thermodynamically corrected diffusivity as described by Darken (1948), Equation 2.4.

$$D_{ie} = D_{self} \frac{\partial \ln C_i}{\partial \ln C_i} \tag{2.4}$$

The accurate description of the diffusion of species in porous adsorbents is dependent on the pore size relative to particle size of the system. IUPAC (Sing et al., 1985) defines three main types of pores based on size; Macropores have widths exceeding about 50 nm; Mesopores comprise between 2 nm and 50 nm and Micropores have widths below 2nm.

For diffusion in macropores, four types of mechanisms can be identified; Molecular diffusion, for when the relative size of the molecules in comparison with the pores is small; Knudsen Diffusion, when the macropore is narrow in relation with the molecules size; Surface Diffusion, when there is significant adsorption at the pore walls and Plug flow. The diffusivities, in order of the citation of the mechanisms is shown in Equations 2.6 to 2.9. They are to be used in a Fickian form of the Diffusional flux, Equation 2.5.

$$J_i = -D_e \frac{\partial C_i}{\partial x} \tag{1.5}$$

$$D_e = D_{molecular} \tag{2.6}$$

$$D_e = D_{Knudsen} = \frac{2}{3} r_p \sqrt{\frac{8RT}{\pi M_i}} \tag{2.7}$$

$$D_e = D_{surface} = D_{surface,0} \exp\left(-\frac{E_a}{RT}\right) \tag{2.8}$$

$$\text{illegible} \quad (2.9)$$

For diffusion in micropores, diffusivity can be described as an exponential function of temperature, Equation 2.10 and the general equation for diffusion in micropores in a spherical particle is described by Equation 2.11.

$$\text{illegible} \quad (2.10)$$

$$\text{illegible} \quad (2.11)$$

More detailed description of diffusion mechanisms and models are to be found in K±rger et al (2011), Do (1998), Nicholson and Petropoulos (1985) and Bathia et al. (2004). Recent works in the general area of adsorption include CO_2 capture, (S®nchez-Zambrano et al. 2019; Morales-Ospino et al. 2020a), the promising research with Metal Organic Frameworks (MOFs) adsorbents for general use (Hossain et al. 2019; Jamshidifard et al. 2019), the growing field of Molecular Simulation (Yang et al. 2019), and in the diffusion area, implementation of new measurement methods (Siqueira et al., 2018a; Richard et al., 2020).

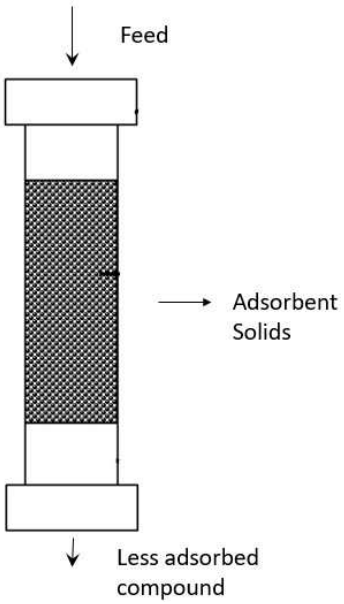
2.3 Separation adsorption processes

In general, the use of adsorption as both a separation and a purification processe revolves around the reversibility of the phenomenon, one or more variables that move the thermodynamic equilibrium are manipulated to achieve the process goal. The global steps of the process are first adsorption, where the compound with the most affinity is retained, reducing its concentration in the bulk and raising the composition of the less adsorbed one, and second the desorption or regeneration, where the most adsorbed compound is released and the solid can be used for the adsorption process again. From this point the differences in the industrial and lab-scale process depends on which variable is manipulated to move the equilibrium and what type of operation the machinery undergoes; a brief description of the most used processes is included in the following sections.

2.3.1 Fixed Bed adsorption

The fluid mixture flows into a column packed with adsorbent solids, the basic process is the same as other fixed bed industrial processes with the addition of the mass transfer caused by adsorption, as Figure 2.2 shows.

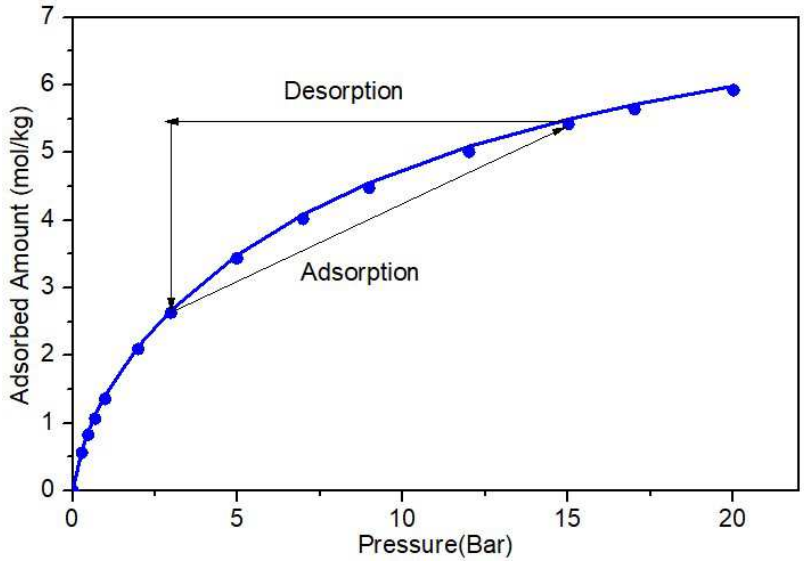
Figure 2.2 – Fixed bed adsorption column



Source: Author

The two main classes of fixed bed adsorption are Pressure Swing Adsorption, PSA and Temperature Swing Adsorption, TSA. For PSA (Ruthven et al. 1994), pressure is cyclically manipulated to promote adsorption, by increasing pressure, and desorption, by decreasing pressure in the packed bed. In a classical isotherm, the path that changes the thermodynamic equilibrium is shown in Figure 2.3, for constant temperature.

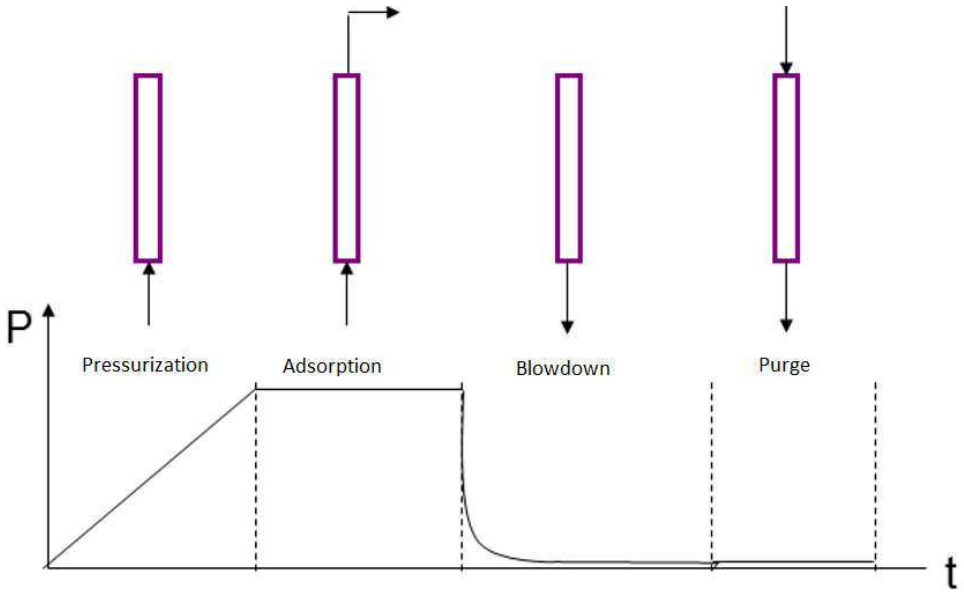
Figure 2.3 - Paths to adsorption and desorption in an equilibrium isotherm, for PSA process.



Source: Author

The simplest form of a PSA cycle is the SkarstrØm Cycle (SkarstrØm, 1960) which is divided into three steps. Pressurization, where the fluid mixture flows into the bed with its exit stream closed, raising the pressure up to the projected value, Adsorption, where in the appropriate pressure, the separation occurs. Blowdown or desorption, where the pressure is abruptly decreased to release the most adsorbed compounds, and Purge, to remove residuals of adsorbate that persists in the solids. Figure 2.4 illustrates the cycle.

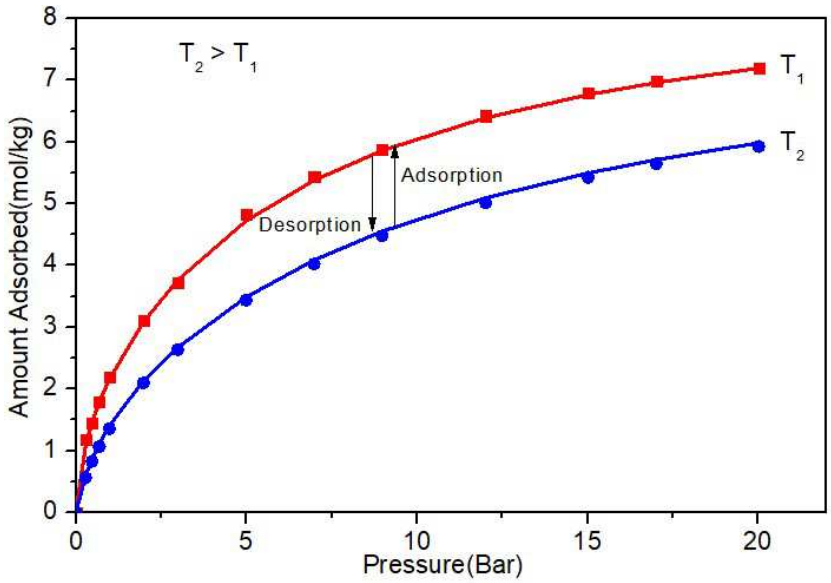
Figure 2.4 - Pressure Swing adsorption process.



Source: Author

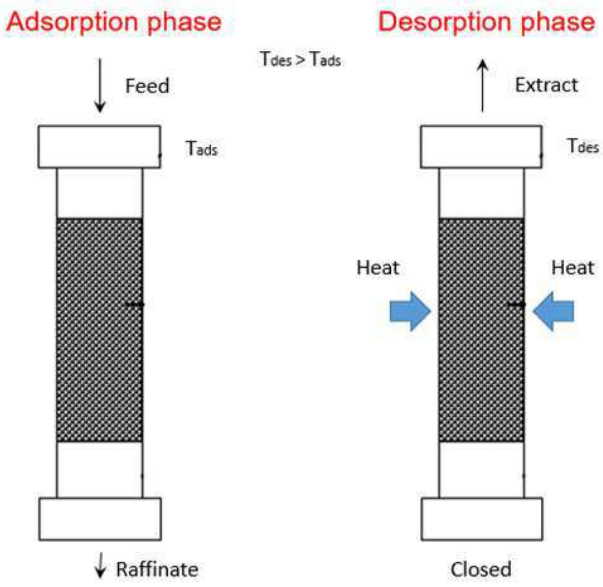
For TSA instead the temperature is changed to promote adsorption, by cooling, or promote desorption, by heating. The path in an isotherm graph is illustrated in Figure 2.5, and the basic operation of the TSA bed is shown in Figure 2.6, where initially the mixture feeds the bed and the less adsorbed compounds are collected in the outflow, when the solids saturate the process is stopped and the temperature is increased to release the more adsorbed compounds.

Figure 2.5 - Paths to adsorption and desorption in an equilibrium isotherm, for TSA process



Source: Author

Figure 2.6 - TSA process illustration



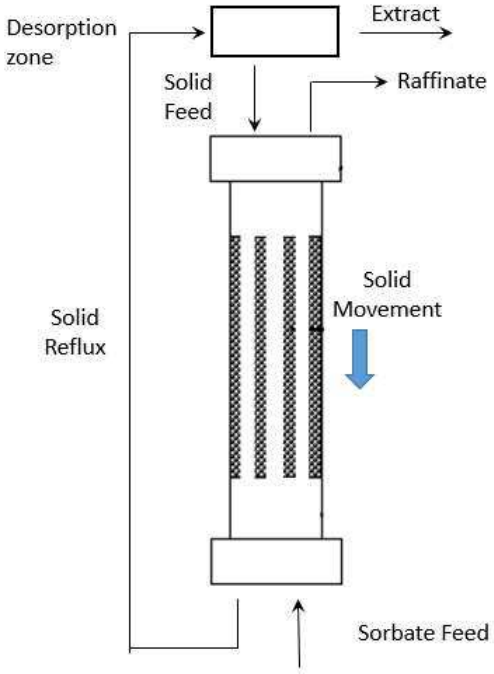
Source: Author

2.3.2 Continuous countercurrent adsorption beds

General countercurrent flow processes are more advantageous than batch or cyclic batch processes due to maximizing the driving force of the transport phenomena (Bird et al., 2007), in the case of adsorption, it maximizes the amount of adsorbed compounds per mass of adsorbent solids. Two main types of countercurrent adsorptions exist, the Moving bed, where the solid, instead of being fixed in the bed, is in continuous movement as a stream while in contact with the adsorbate fluids, and Simulated moving bed, where no actual solid movement happens, but the same advantage of the countercurrent flow is achieved.

Figure 2.7 illustrates a generic Moving bed system with two stages, the adsorption stage is where the fluid and the solid are in countercurrent contact and the desorption stage is where the saturated solids are transported to be reutilized in the adsorption stage. The solids flow downstream and it is required a solid transport system to move the used sorbent to the desorption stage.

Figure 2.7 - Moving bed illustration



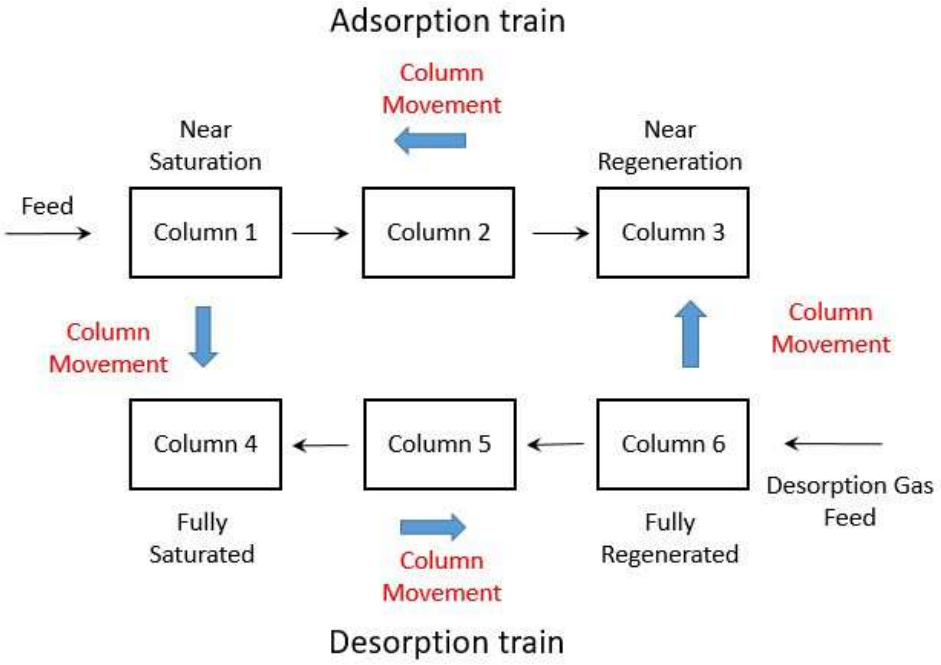
Source: Author

The greatest difficulty associated with the moving bed is the solid transport, which generates attrition and could cause obstruction problems in the tubes.

The basic concept of the simulated moving bed is shown in Figure 2.8. A series of beds are placed in two separate stages called adsorption and desorption train. In the adsorption train,

the feed flows into columns arranged in series, segmenting the mass transfer zone. The first bed becomes the most saturated while the last contains almost fully regenerated solids. In the desorption train the same series arrange is employed, a purge stream flows in series through the columns, the first one becomes the most sorbate-free while the last one maintains itself close to saturation. The simulation of the countercurrent flow then happens through switching the places of each column in and out of the two stages. Once the first column in the adsorption train becomes fully saturated, it is moved to the last position of the desorption train while the fully regenerated column in the desorption train moves to the last position of the adsorption train, this operation continues in a loop and that ensures a process similar to the countercurrent system.

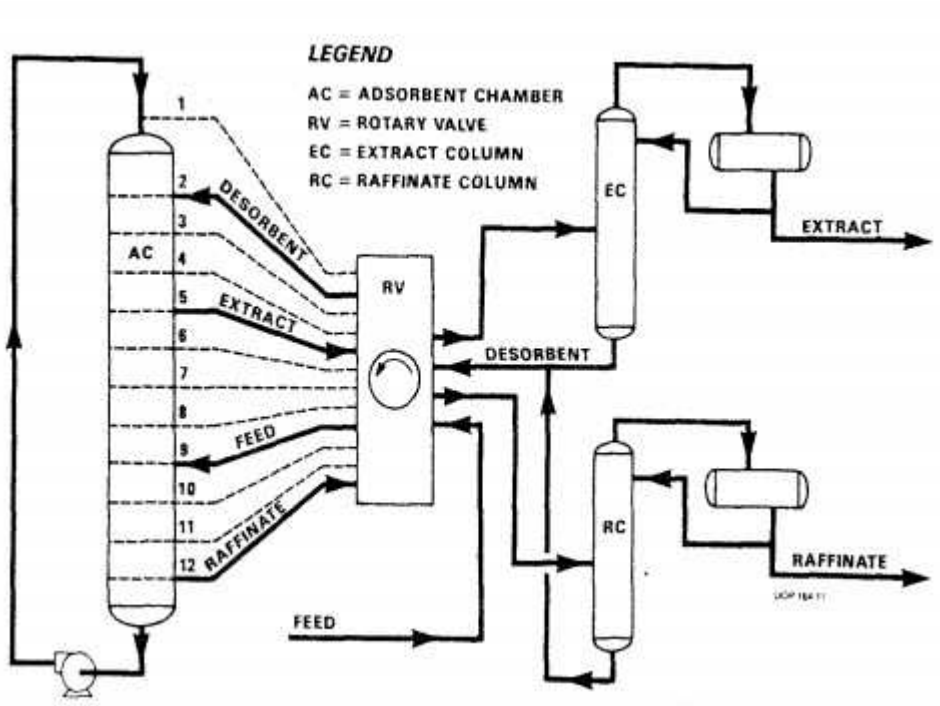
Figure 2.8 ¯ Simulated Moving bed illustration



Source: Author

The switching of the columns as described does not happen in the literal sense, most of the time the switches are made through an arrange of valves. An important industrial process that uses the simulated moving bed principle is the Sorbex (Johnson and Oroskar, 1989), which employs a single column with multiple segments, the feeds are moved continuously through a rotary valve. Figure 2.9 illustrates the operation.

Figure 2.9 – Sorbex process.



Source: Ruthven (1984)

Recent work on the field of adsorption separation processes include High Pressure PSA simulation (Siqueira et al. 2018b), Zeolite studies for CO_2 capture through TSA (Morales-Ospino et al., 2020b) and Moving bed adsorption for natural gas separation (Mondino et al. 2019).

2.4 Machine Learning and data science

Machine learning is a part of the field of Data Science that focuses on the study of algorithms that improve themselves automatically. In essence the algorithms are constructed to make predictions based on the data that is provided. The basic approach to this type of learning is first gathering a large amount of data related to a certain problem of interest, and organizing it into a matrix, where each row is labeled as a data sample and each column is labeled as a feature. One or more of those features is the parameter or variable of interest that is supposed to be predicted, which is commonly referred as the targets, and the rest of the features are the variables that are to be used to predict the target. This type of arrangement works to represent the target in function of selected features to form a function 倀 唸 倀 where 倀 is the target and 倀 a vector of features. Table 2.2 shows how a data matrix looks like for a machine learning application.

Table 2.2 - Machine learning matrix

Target	Feature 1	Feature 2	Feature 3	Feature 4
1	0.50	0.35	0.42	0.05
0	0.24	0.67	0.78	0.00
0	1.00	0.98	0.68	1.00
1	0.65	0.53	0.31	0.27
1	0	0.01	1.20	0.95

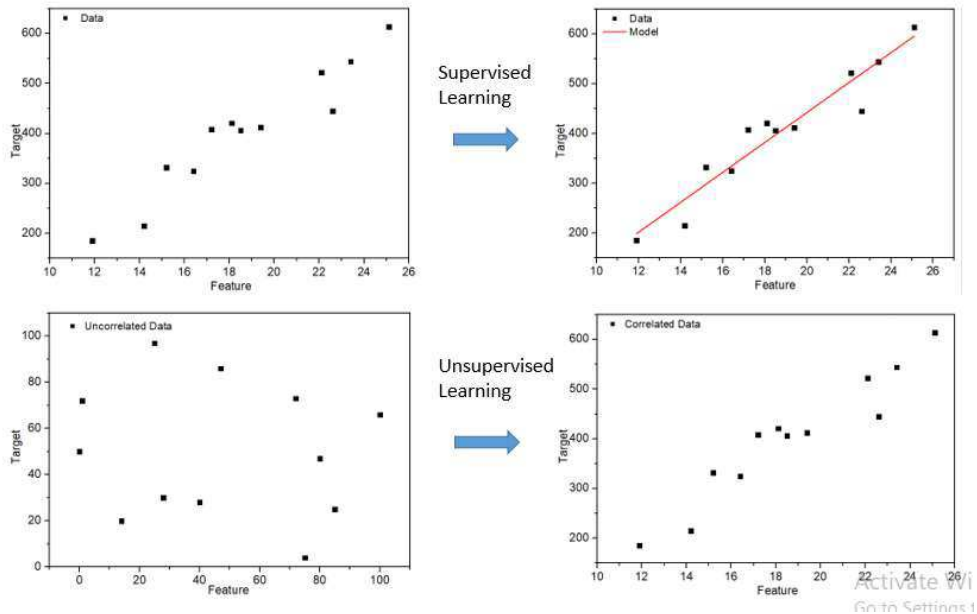
Source: Author

The two main types of problems that can be solved through machine learning are regression and classification problems. In regression problems, the algorithm predicts a numeric value of a target, as an example, the prediction of the Purity of a product of an adsorption process is done through regression. In the case of a classification problem, the algorithm predicts a class, as an example, the prediction of the malignity of a tumor is done through a classification algorithm, which indicates that the tumor is malign or not.

Considering now the types of learning employed in Machine learning, there is supervised learning and unsupervised learning. In the supervised case, the dataset fed to the model contains both inputs and outputs, that is it contains both the features required for prediction and the value of the targets, that is, the dataset already contains the expected values and the goal of the Algorithms is to find a model that generalizes the behavior. In unsupervised learning the outputs are not known and the goal of the algorithm is to find patterns in the dataset so it can use its features to predict the desired target.

Surrogate modeling using machine learning is a type of supervised learning, as the expected values for the target are already known through the detailed model, and the goal of the learning algorithm is to generalize results for faster convergence. An example of unsupervised learning is face recognition (Müller and Guido, 2017), the algorithm looks for facial patterns of a specific person to identify it and check for the person's identity through this features when necessary. Figure 2.10 illustrates both supervised and unsupervised learning.

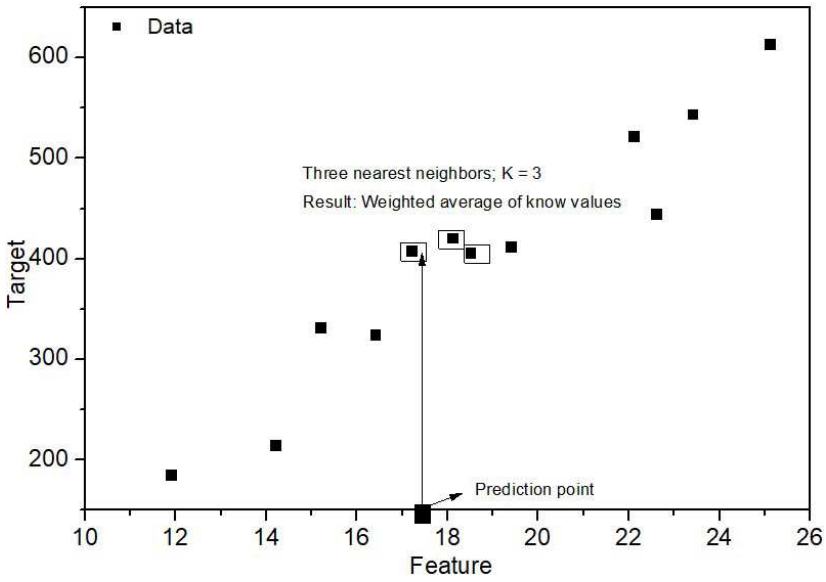
Figure 2.10 ¯ Supervised and unsupervised learning differences.



2.4.1 Main types of Machine Learning models.

The simplest type of algorithm is the K-Nearest-neighbors, the operation of this algorithm follows by first accounting for a number of known output points, then an unknown input is fed to the model, which calculates the output based on the smallest distance of known outputs. The parameter of the model is simply the number of neighbors, if the number is one, then the predicted value is simply the value of the nearest known output, any number larger than one neighbor calculated the values based on averages of the values of the nearest points. Figure 2.11 illustrates the process.

Figure 2.11 ¯ K- Nearest neighbors˘ algorithm example



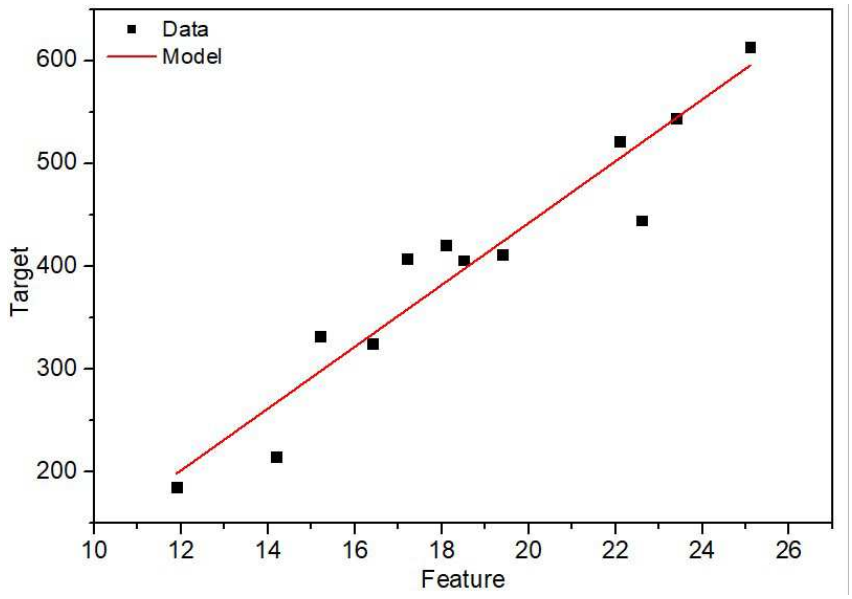
Source: Author

The second main type of model is Linear regression and classification. The general for of the model is that a specific output point is modelled as a sum of linear functions for all the features provided by the dataset, as Equation 2.12 shows.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \quad (2.12)$$

Variations of the linear model commonly used are Ridge and Lasso models, these models add a constraint to all the β constants so their value approaches zero, such process is known as Regularization and is used to avoid overfitting of the model on the data. Figure 2.12 shows a well-known Linear Regression to illustrate these models. As the Linear models are a more robust form of algorithm, in general they perform better than the kNN models.

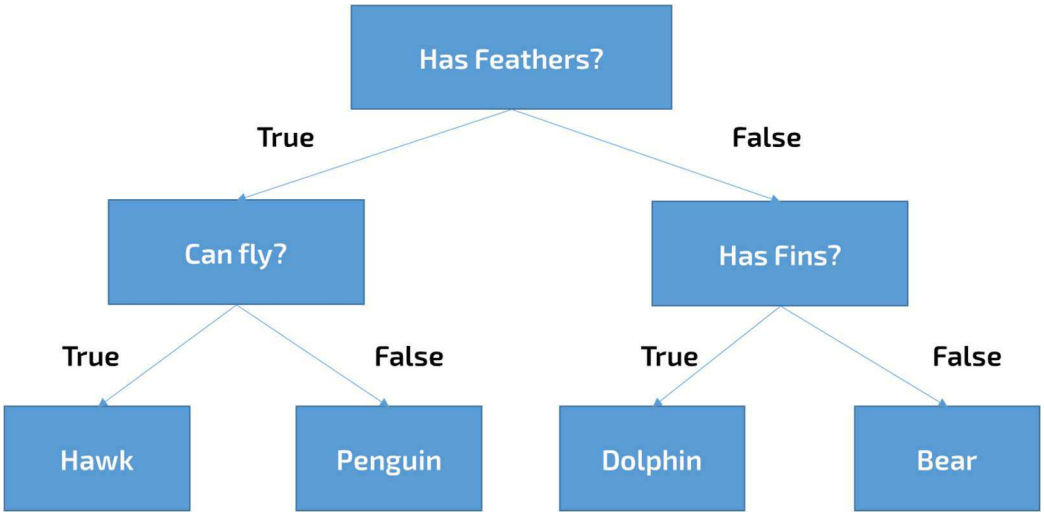
Figure 2.12 ¯ Linear algorithm example



Source: Author

A more widely used type of algorithm are Decision Tree and its variations. The method of learning of the model is simply through a series of If/else questions based on known data, narrowing until an answer is found, Figure 2.13. The controlled parameter is the depth of the tree, if the relation between targets and features is more complex, it will require a deeper tree to make its decision.

Figure 2.13 ¯ Classic example of a Decision tree



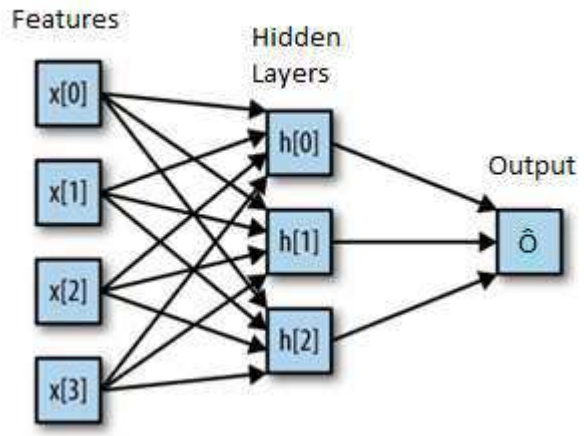
Source: Author

Variations of pure decision trees include Random Forests and Gradient Boosting trees, which are more sophisticated form of Trees that are built on to avoid overfitting as it˘s a common occurrence when very deep trees are required for prediction. In the case of Random

forests, the algorithms averages on an ensemble of regular decision trees, for Gradient trees, the gradient boosting technique is applied, where the residual error (the prediction values minus the actual value) are fitted to be proportional to the gradients of the mean squared error of the model, thus multiple regular decision trees are trained with the boosting technique to avoid overfitting.

Lastly the most robust type of algorithms is the Neural Networks, also known more recent as deep learning (G¶ron, 2019). There are multiple different types of statistical methods inside the Neural network algorithms, the simplest one that explains the overall process is the Multilayer Perceptrons (MLPs). For this, the MLPs are constructed through the computation of multiple weighted sums of Linear Models, starting from a regular one and progressing through multiple hidden layers. Figure 2.14 illustrates the path of the algorithm for one hidden layer and four features.

Figure 2.14 ¯ Multilayer perceptrons illustrations



Source: Modified from Mæller and Guido (2017)

Each cell of the layer is composed of a regular linear model, Equations 2.13, 2.14, and 2.15.

$$\text{[illegible]} \quad (3.13)$$

$$\text{[illegible]} \quad (2.14)$$

$$\text{[illegible]} \quad (2.15)$$

After this, a nonlinear model is applied to each one of the layer components, and those are merged in a final linear weighted sum, representing the output. Equations 2.16 to 2.18, where, in this example, the nonlinear model applied is īéā÷. In equation 2.19, a linear model is superposed onto the nonlinear model with its own intermediate weights, finishing the output.

$$\text{[illegible]} \quad (4.16)$$

$$\text{[illegible]} \quad (2.17)$$

$$\text{[illegible]} \quad (2.18)$$

$$\text{[illegible]} \quad (2.19)$$

The number of layers can be controlled by the user and as this value rises the number of weighted sums increases sharply, with this approach, usually the Neural Network can make predictions of very complex datasets.

Other than the shown examples, the versatility of Machine learning is even more evident by its use in predictions related to the COVID-19 pandemic (Alimadadi et al. 2020) and applications in public policy and general economics (Athey and Imbens, 2019). Introductory material for Machine Learning and fast application can be found in the comprehensive book from Mæller and Guido (2017), for an approach focused on Neural Networks, the work from G¶ron (2019) provides numerous approaches and applications. A more theory-heavy look into the models can be found in Hastle et al. (2009), where most of the theory of the models exposed here is discussed in detail.

2.5 Unexplored parts of Machine Learning applied to adsorption

As seen in the literature review, most works of Machine learning as surrogate modelling of adsorption focuses on the use of Artificial Neural Networks as the predictor model, being the most complex and therefore, the most expensive computationally. Since the alternative statistical models find success even on more complex systems such as market prices, it˘s reasonable to assume that they also would work for a more studied and deterministic process

such as adsorption, an analysis on how each method performs could be useful to help in decreasing further the time required for convergence.

A second point is that most research focuses on PSA processes, leaving TSA, Moving bed and Simulated moving bed unexplored. Additional focus on different processes could lead to a more generalized approach of the use of Machine Learning in adsorption. A third and final point is that mostly paid software is used to apply the algorithms. In recent years, numerous open-source alternatives are available and has seen success, constructing algorithms in this environment could help the general progress of research in this field.

3 METHODOLOGY

This section details the steps required to realize the optimization of an adsorption process via Machine Learning and the software used. Data analysis techniques to facilitate the process are discussed.

3.1 Phenomenological PSA Model

The model, developed in the software gPROMS® (Siemens, Germany) version 4.0, simulates an experimentally validated (Siqueira et al. 2017; Siqueira et al. 2018b) dynamic Pressure Swing Adsorption (PSA) fixed bed under the Skarström Cycle for the separation of CO_2 and N_2. The adsorbent used for separation was a commercial microporous activated carbon (AC) named CHARBON 500 (Carbonado Comércio de Produtos Filtrantes Ltda., Brazil), more information about the experimental aspect of the simulated PSA process can be found in the supporting information, Table 3.1 contains the properties of the AC.

Table 3.1 – Properties of the adsorbent solid

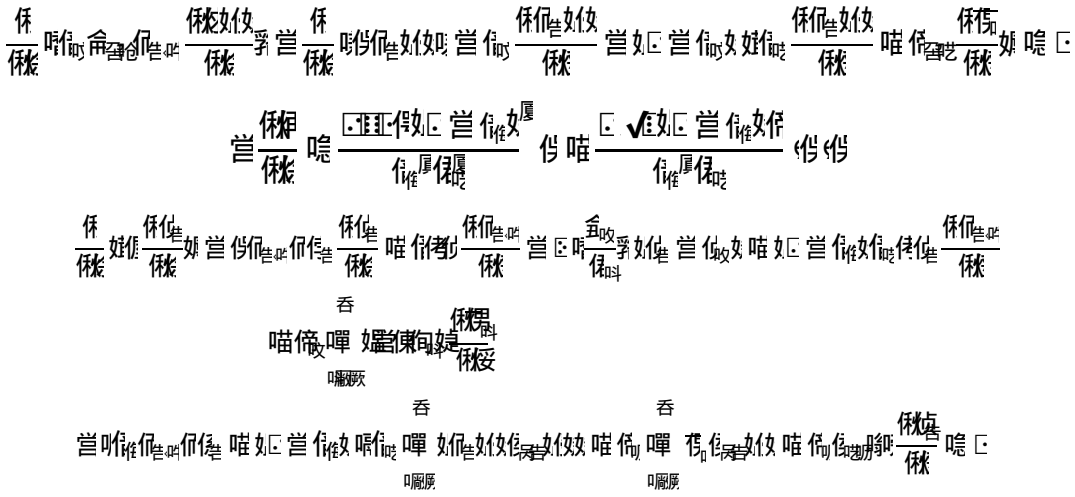
Sorbent Properties					
CHARBON 500	specific surface area	particle diameter	specific pore volume	specific solid volume	ε_p
	1025 $m^2 g^{-1}$	1 mm	4.60E-4 $m^3 kg^{-1}$	4.90E-4 $m^3 kg^{-1}$	0.48

The phenomenological model of the PSA process is based on transport phenomena equations and has been reported in multiple literature contributions: the mass balance is

described in Silva et al. (2013), energy balance can be found in the works of Ribeiro et al. (2008) and Luberti et al (2015)., momentum balance and full description of boundary conditions are included in Silva et al. (2014), Ferreira et al. (2015) and Marx et al. (2015). Once validated, the phenomenological model is used to generate the training and testing data of the machine learning algorithms. A brief description of the assumptions of the model are given as follows.

- Differential mass, heat and momentum balances are considered only in the axial direction of the column;
- The bulk-phase mass balance considers axial dispersion according to Knox et al. (2016);
- The mass transfer rate is described by the Linear Driving Force model by Glueckauf (1955);
- Ideal gas law behavior is assumed over the studied pressure and temperature ranges;
- Homogeneous porosity and bed density along the column;
- Mass and heat transfer coefficients are considered to be temperature-independent under the experimental conditions.

The main equations for the balances of mass, momentum and energy are summarized, respectively, in Equations 3.1-3.3



(3.1)

(3.2)

(3.3)

Competitive adsorption equilibrium of CO_2 and N_2 was represented by the extended dual site Langmuir (DSL), as shown in Equation 3.4, and the kinetic model was Linear Driving Force (LDF), Equation 3.5.

$$\text{[illegible]} \quad (3.4)$$

$$\text{[illegible]} \quad (3.5)$$

$$\text{[illegible]} \quad (3.6)$$

Table 3.2 contains the main physical model parameters, such as mixture composition and adsorbent mass; those do not change during the sampling for the surrogate modelling or during the application of the machine learning models.

Table 3.2 – Constant simulation parameters of the PSA model.

Parameter values					
[illegible]	0.549 m	[illegible]	0.021 W m^{-1} K^{-1}	[illegible]	14.715 kJ mol^{-1}
[illegible]	0.406 m	[illegible]	1.28 mol kg^{-1}	[illegible]	14.72 kJ mol^{-1}
[illegible]	0.0286 m	[illegible]	1.92 mol kg^{-1}	[illegible]	5.76 kJ mol^{-1}
[illegible]	0.0313 m	[illegible]	7.645 mol kg^{-1}	[illegible]	-24 kJ mol^{-1}
[illegible]	0.133 kg	[illegible]	7.64 mol kg^{-1}	[illegible]	-15.07 kJ mol^{-1}
[illegible]	377 kg m^{-3}	[illegible]	5.35E-5 Pa^{-1}	[illegible]	0.136
[illegible]	0.642	[illegible]	1.61E-6 Pa^{-1}	[illegible]	0.864
[illegible]	820 J kg^{-1} K^{-1}	[illegible]	1.71E-6 Pa^{-1}	[illegible]	298.15 K
[illegible]	470 J kg^{-1} K^{-1}	[illegible]	6.85E-8 Pa^{-1}	[illegible]	298.15 K
[illegible]	7860 kg m^{-3}	[illegible]	26.54 kJ mol^{-1}	[illegible]	8.314 J mol^{-1} K^{-1}
[illegible]	52 W m^{-1} K^{-1}				

3.2 Process optimization setup and sampling

The parameters used in the modelling and optimization of the PSA process are purity (Equation 3.7) and recovery (Equation 3.8) of N_2, which are the targets of the machine learning models. Both variables are taken as the PSA process reaches cyclic steady state.

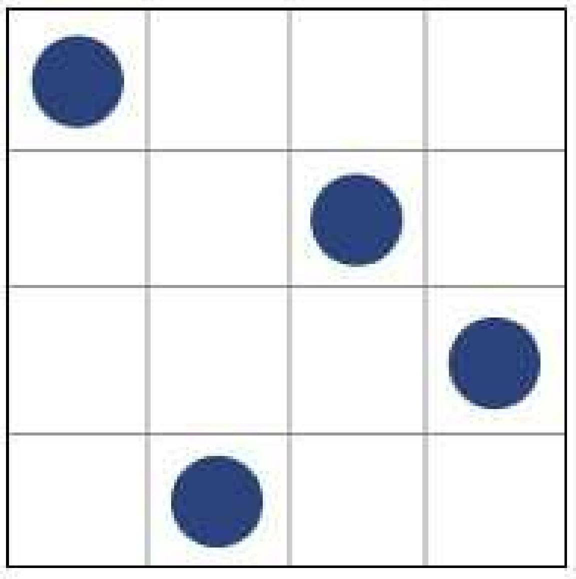
$$[illegible] \quad (3.7)$$

$$[illegible] \quad (3.8)$$

The variables, called features in the machine learning model, used to optimize the performance parameters are [illegible] with range 20s – 100s , [illegible] with range 20s – 120s and [illegible] with range 2 bar – 20 Notice that by fixing [illegible], [illegible] and [illegible] the other two times of the SkarstrØm cycle, desorption and purge, are already determined. The sampling was performed through Latin Hypercube Sampling (LHS) implemented in Python 3.7.4 (open source) using the package Design of Experiments for Python, pyDOE. The tested sample sizes ranged from 600 up to 1200 data points, with special consideration to the results with 600, 800 and 1200 data points.

The LHS method is a statistical technique used to generate representative, near-random samples of multidimensional functions, i.e. functions with more than one independent variable. In the present case, the functions are Purity and Recovery which are going to be dependent on [illegible], [illegible] and [illegible] The objective of the sampling is to get a good sample that represents the behavior of both performance parameters as the three variables change. The Latin hypercube sampling is a generalization of the Latin Square, Figure 3.1, in which a function of 2 variables is sampled in a way such that no more than one sample point is allowed for a given row and column of the sampling space, in higher dimensions, this means that no hyperplanes of the sampling space cross each other.

Figure 3.1 – Latin Hypercube sampling in 2 dimensions (latin square)

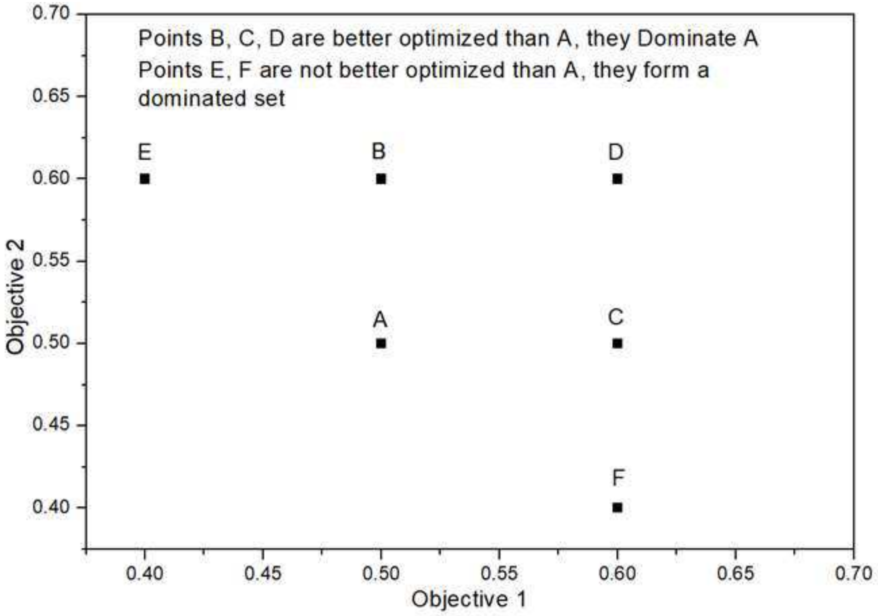


Source: pythonhosted

Once sampling and Machine learning model fitting is done, the models for Purity and Recovery are to be optimized simultaneously as to give the highest values possible for each. Since two objective functions are being optimized at the same time, the optimization problem is one of Multi-Objective-Optimization, as such there is not going to be one single optimal value, but a series of optimal values. The reason for that is because, for example, given two pairs of Purity 0.95, Recovery 0.5 and Purity 0.94 and Recovery 0.55, mathematically one pair is not better optimized than the other because in both pairs, one parameter is higher than the other, for the first pair, purity is higher, for the second pair, recovery is higher. However, a pair with Purity 0.96 and Recovery 0.6 will be better optimized than both previous pairs, as both objective functions are larger, the same would be true for a Pair Purity 0.96 and Recovery 0.55, as one objective function is bigger while the other is at least equal.

Thus, for a optimization algorithm to search for optimal values, it has to satisfy the following optimality condition: For a given pair of points [illegible] another pair [illegible] is going to be better optimized, thus, will dominate the first pair if [illegible] or [illegible] or [illegible]. Figure 3.2 illustrates this condition. If a set of pairs in a given point of the optimization does not satisfy the optimality condition and no other new points can be found, then these form a Non-Dominated Set, or a Pareto Set.

Figure 3.2 ‑ Criteria for multi objective optimization



Source: Author

The algorithm used to perform the optimization was the Non-dominated Sorting Genetic Algorithm II (NSGA-II) implemented in python language using the Python Multi-Objective Optimization (PYMOO) package to obtain the Pareto set for purity and recovery parameters. The genetic algorithm was chosen due to being the most used similar works, such as Pai et al. (2020) and Balashankar et al. (2019). A comprehensive description of the NSGA-II algorithm can be found in Kalyanmoy (2002). Once the sampling is complete, the phenomenological PSA model is ran for each combination of 傜磃区, 傜磃呂历 and 偭晃呗 for each sample size, making it 600, 800 and 1200 runs, respectively. This step of the optimization has the largest majority of computational effort required.

3.3 Python Machine Learning Setup

This subsection describes all the steps and techniques used to successfully model the PSA process with the data provided by the LHS sampling described in the previous section. The package that was used to fit the machine learning models was scikit-learn version 0.23.1, the up to date documentation can be found in scikit-learn (2023), the original paper with the methods of the package refers to Pedregosa et al. (2011). Every programming function described in this section and used in this thesis are part of the cited package.

3.3.1 Splitting of the training and test sets and data Scaling

After generating the sample, it˜s necessary to split into two subsamples of training and test sets, but there is no general rule of what percentage of the total sample is going to be. Too big of a training set will provide poor generalization of the model and tendency of overfitting and too small of a training set will provide the algorithms with too few data to be reasonably accurate. The percentage of splitting used follows the function train_test_split which uses 75% of the data to train the model and 25% to test it.

The separation of both sets is done randomly with the aid of Random number generation (RNG) functions that can be controlled inside the function, different random splitting can be achieved by changing the number at the end of the function, such numbers exist primarily to guarantee that the results are reproducible as even though the separation is done by random, the RNG function guarantees that as long as the same number is used, the data will be split in the exact same manner, with the same training and test sets for every use.

The statistical algorithms are then ready to be fitted, however not always it˜s advisable to use the original numerical representation of the data to fit the models. Given that each supervised learning algorithm can be fitted to any numerical range that the data possesses, from very small numbers to very large numbers, some algorithms have more difficulty producing good fits to certain types of data (Mæller and Guido, 2017). Scaling normalizes the data to more tractable values, the simplest type of such technique is normalization between 0 and 1.

In the present work, three types of data representation were used: Unscaled; Scaled and Polynomial Scaled. The Scaled mode uses the function min_max_scaler , which normalizes the data between 0 and 1 using Equation 3.9, while the polynomial scaling mode uses the function PolynomialFeatures of degree 2 on top of min_max_scaler to transform each feature of the data samples into polynomial products of each other, creating `new variables_. The implementation of this type of scaling is represented in a linear model in Equations 3.10 and 3.11, with 3.10 representing the model without polynomial scaling and 3.11 with polynomial scaling.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.9}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{3.10}$$

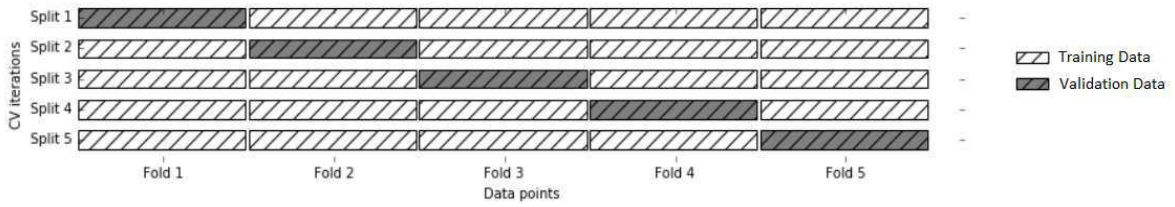
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 x_1^2 + \beta_5 x_2^2 + \beta_6 x_1^2 x_2 + \beta_7 x_1 x_2^2 + \beta_8 x_1^3 + \beta_9 x_2^3 \tag{3.11}$$

3.3.2 Machine Learning models used

Once data is properly preprocessed, the ML models can be fitted into it and their accuracy evaluated with the adjusted R^2 metric with additional support of the RMSE metric for consistency. A priori it's reasonable to just go for the most robust and complex models like ANN as they have on average better accuracy than the simpler models, however this isn't a guaranteed behavior (Mæller and Guido, 2017; Géron, 2019) thus is advisable to test a wide arrange of models and check which one is best suitable for the problem. Moreover, the more complex the statistical model, the more `blackbox_ it becomes, so even if a model like ANN has better R^2 than a Linear Regression, if this difference is negligible, like 0.995 for the formed and 0.99 for the latter, it is preferable to optimize the objectives with the later, as it's easier to interpret and extract more useful information from it.

To better evaluate model performance, the technique of k-fold cross validation is employed, this technique allows for more stable evaluation of model generalization, avoiding leakage of information between the test and training sets. This is done by making k splitting's (folds) of the training set into training and validation sets, the model is then fit into the training set and its performance is evaluated by calculating the R^2 with the validation set k times, making it k values of R^2 which are then validated to obtain the training accuracy. For this work the value of k was 3, thus each training of a single model will contain a mean of 3 R^2 values. Figure 3.3 illustrates the process of cross validation with 5 folds.

Figure 3.3 – Cross validation with 5 folds



Source: Modifier from Mæller and Guido (2017).

Each machine learning algorithm has its own hyperparameters, which are used to tune the model into fitting the data better, the best hyperparameters are found via a searching algorithm called gridsearchcv which also realizes the cross validation. The criteria for what is the best value is by evaluating the best R^2 on the training and validation sets, and then after this, the model with its optimal values is evaluated against the test set, where the R^2 on the test set

indicates the actual model accuracy, as a high value on the training set may just be overfitting. Every result in this work that shows R² refers to only the test set ones. Table 3.3 shows the evaluated models and their corresponding tuned hyperparameters and the search domain, the MLP step size varies due to the very large possible combinations of number of neurons per layer versus number of layers, but in general, each layer has the same number of neurons and the size of the neurons increase have a step of 1,2,5 or 10, depending on the differences of score, requiring a bit of experimentation. A brief discussion of each parameter can be found elsewhere (Hastle et al., 2009; Mæller and Guido, 2017; G¶ron, 2019). The type of neural networks used on this work was the Multi-Layer-Perceptrons, MLP.

Table 3.3 ¯ Machine learning models and their hyperparameters. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

model	hyperparameter	Lower bound	Upper bound	Step size
KNN	Number of neighbors	2	40	1
Ridge	Regularization weight (alpha)	0	10	0.05
Lasso	L1 regularization weight (alpha)	5E-06	0.05	5E-04
Decision Tree	Maximum tree depth	1	40	1
Random forests	Number of estimators	1	100	1
Gradient boosted trees	Maximum tree depth	1	10	1
MLP	Number of hidden layers and size of the layers	0 x 0	100 x 10	varies

The activation function of the MLP model was Rectified linear unit (Relu), the choice is justified initially as the function doesn¨t have problems with vanishing gradients and later as it was the function that provided the best score and convergence. The chosen solver was the default adam(Adaptative moment estimation) due to its adaptative learning rate and overall cost efficiency compared to the other options, more discussion on the results section. Each model has other possible tuned parameters, such as metrics and solvers, these are kept on their default value on the python package, these values can be accessed in the documentation of scikit-learn (2023). Tables 3.4-3.9 shows the constant parameters of each model.

Table 3.4 – K-Nearest Neighbors constant parameters

K-Nearest Neighbors	
Weights	Uniform
Algorithm	Automatic selection between `ball -tree_, `KD-tree_ and brute force
Metric	Minkowski
Metric power	2 (Euclidian distance)
Leaf size	30

Table 3.5 – Ridge Constant Parameters

Ridge	
Sample weight	None
Solver	Automatic selection between 7 available solvers, refer to scikit-learn (2023)
Max iterations	None
Tolerance	1E-04

Table 3.6 – Lasso constant parameters

Lasso	
Fit intercept	True
Precompute Gram Matrix	False
Max iterations	1000
Tolerance	1E-04
Warm start	False
Coefficient calculation	Cyclic

Table 3.7 – Decision Tree constant parameters

Decision Tree	
Quality Criterion	Squared error
Splitter	Best split
Minimum samples to split	2
Minimum samples at leaf	1
Maximum number of features	None
Minimum impurity decrease	0
Complexity parameter	0

Table 3.8 – Random Forests constant parameters

Random Forests	
Quality Criterion	Squared error
Maximum tree depth	None
Minimum samples to split	2
Minimum samples at leaf	1
Maximum number of features	None
Minimum impurity decrease	0
Complexity parameter	0
Maximum leaf nodes	None
Bootstrap samples	True
Out of bag sample estimates	None
Warm start	False
Maximum number of samples	None

Table 3.9 – Gradient boosted trees constant parameters

Gradient boosted Trees	
Quality Criterion	Friedman mean squared error
Number of estimators	100
Minimum samples to split	2
Minimum samples at leaf	1
Maximum number of features	None
Minimum impurity decrease	0
Complexity parameter	0
Maximum leaf nodes	None
Bootstrap samples	True
Out of bag sample estimates	None
Warm start	False
Maximum number of samples	None
Loss	Squared Error
Learning Rate	0.1
Fraction of samples for learning	1
alpha	0.9
Tolerance	1E-04

Table 3.10 - MLP constant parameters

Multilayer Perceptrons	
Solver	Adam
Alpha	1E-04
Initial learning rate	2
Maximum iterations	100000
Shuffle samples on each iteration	True
Random State	0
Tolerance	1E-4
Beta 1 (Adam)	0.9
Beta 2(Adam)	0.999
Epsilon (Adam)	None
Warm start	False
Maximum epoch faiures	10
Early stopping	False
Validation Fraction	0.1
Fraction of samples for learning	1
Activation Fucntion	Relu

3.3.3 Sensitivity tests

Sometimes the random split of the train_test_split can generate training samples that are too similar to the test set, or vice versa, which makes the test accuracy be slightly misleading. Since different numbers on the RNG parameter of the function gives different random splittings, sensitivity tests can be done to control for biased results and to test how sensitive the machine learning models themselves are relative to changes in training and test sets. On top of that, the Cross-validation folds are also chosen by random and the function gridsearchcv has its own RNG parameter that is also used to test sensitivity. Table 3.11 shows the value of the RNG parameter of both functions.

Table 3.11 ¯ RNG parameters of the splitting functions

splitting function	RNG parameters, random_state
Train_test_split	[0,1,2,11,42]
gridsearchcv	[0,1,2,11,42]

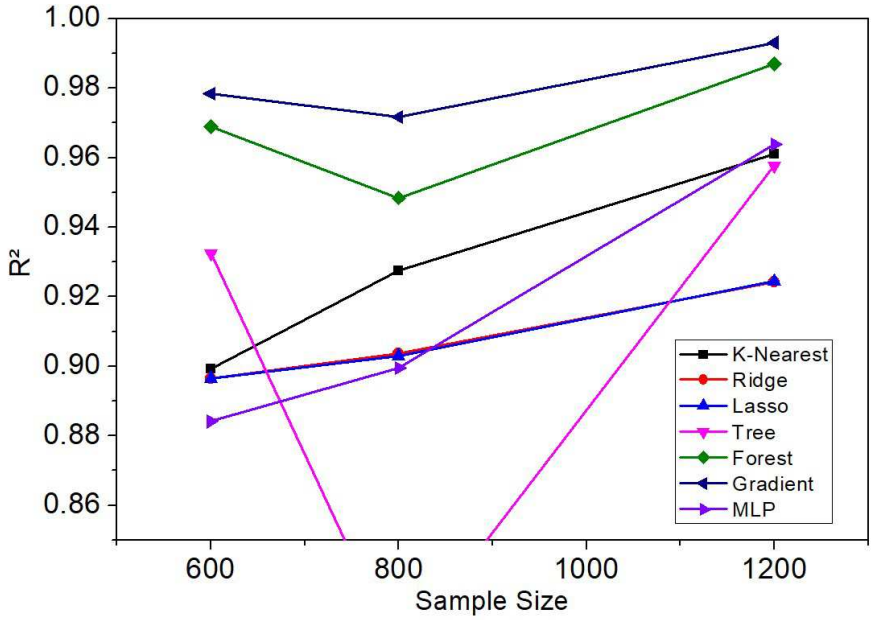
4 RESULTS AND DISCUSSION

This section shows and discusses the accuracy and sensitivity tests of the machine learning models, as well as the final optimization of the performance parameters. Methods on improving the algorithms accuracy are proposed. This section presents in more detail what was published in the article by Richard et al. (2023)

4.1 Analysis of model accuracy

Figure 4.1 shows the goodness of fit for all tested machine learning algorithms considering different total data sample sizes of 600 up to 1200 data points for both purity and recovery. Note that the RДvalues shown in the figure stand for an average of all the tested scenarios, namely Unscaled, Scaled and Polynomial scaling, therefore the results represent an `agnostic_ view about the methods of data manipulation used, they are also the average RДfor both purity and recovery.

Figure 4.1 – Overall R^2 for evaluated scaling scenarios.

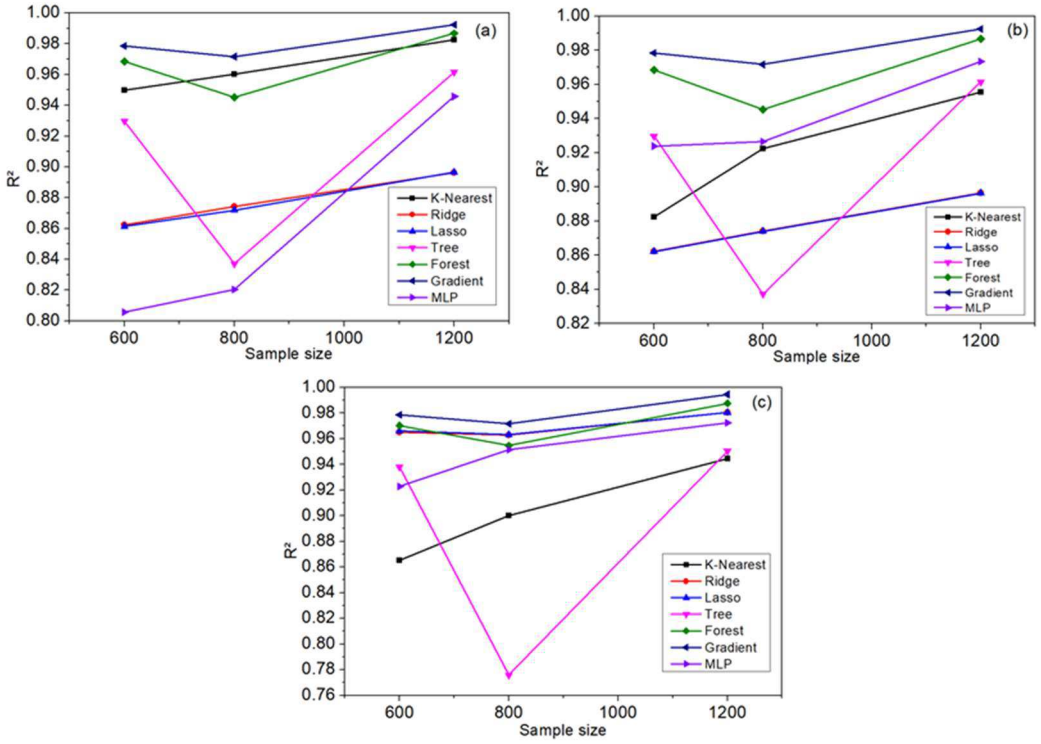


Source: Author

K-Nearest, Ridge and Lasso models exhibit a pattern of steady increase of R^2 with data sample size, however the decision tree-based models (Decision trees, Random forests and Gradient boosted trees) show a more unstable pattern with no significant variations for Random forests and Gradient boosted trees from sample size 600 to 800 and a more pronounced decrease in R^2 for Decision trees in the same range. MLP model also exhibits a pattern of increasing R^2 with sample size. Remarks from these results are that the simplest model, K-Nearest, was the overall third best model (based on R^2 for the maximum amount of sample sizes. Interestingly, the MLP model, the one which is expected to have the better overall fitting as in Beck et al. (2016) is the worst at the lowest data sample size.

In order to remove biases from the data manipulation methods for better evaluation of the algorithms, Figure 4.2 was plotted to show the R^2 for each machine-learning model in the Unscaled, Scaled and Polynomial Scaling scenarios.

Figure 4.2 – Overall R² for the (a) Unscaled, (b) Scaled and (c) Polynomial Scaling scenarios.



Source: Author

Splitting up the scaling scenarios revealed no significant change in the patterns and the Decision tree model still shows a high drop in performance for 600 to 800 data points. That represents an anomaly with no clear explanation, but since it only happens with a single model, it doesn't affect the general results and, for practical applications, the decision tree model would be discarded due to this behavior in this specific situation. With respect to the relative accuracy of the models, in the Unscaled scenario, remarkably the k-nearest neighbors was tied with Random Forests and Gradient methods as the best ones in the N = 1200 case. The MLP model, while unscaled, was the worst performing method at the lowest amount of data, N = 600. The underperformance of the MLP without data scaling is to be expected.

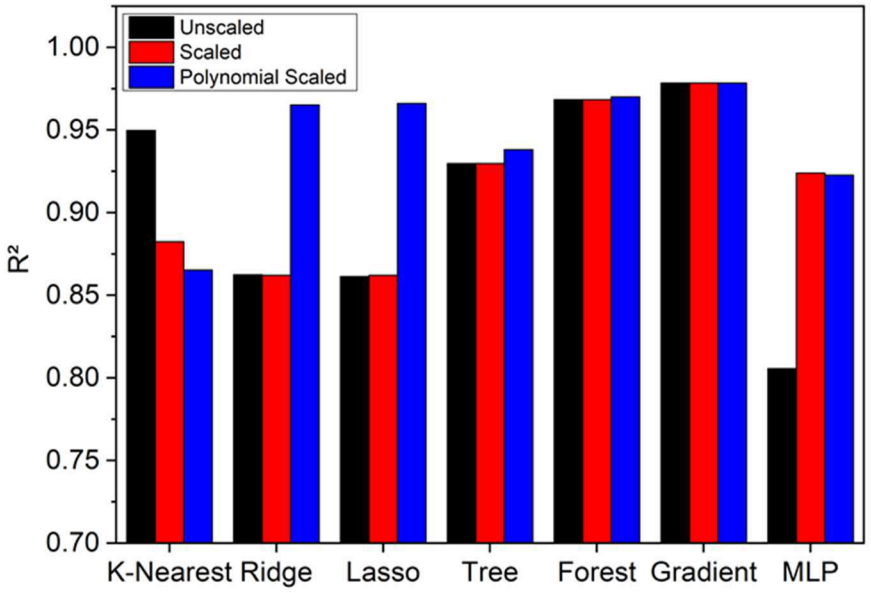
In the Scaled scenario, the relative performance of each method changed. For N = 600, scaling improves the MLP model performance by a significant margin, while worsening the K-Nearest model to a great extent. The other models preserve their relative performance. Lastly, in the Polynomial Scaling scenario, the K-Nearest model is aggravated even further, being the worst performing model in N = 600. However, for the linear-type models, Ridge and Lasso had their performance significantly improved being tied with Random Forests and Gradient methods as the best methods at all sample sizes. This result is in line with what the literature establishes about tree-based algorithms, since they are reported to be widely successful in a

multiple range of problems (Fernandez-Delgado et al., 2015; Mæller and Guido, 2017; G¶ron, 2019). The MLP model is consistently outperformed by the modified decision tree methods in all situations and performs worse than the linear turned into Polynomial of 2nd degree models, which is an unexpected result, because the MLP model was initially considered as the most robust for the regressions.

Figure 4.3 helps visualizing the multiple effects of scaling in the models for a sample size of 600. Further analyses of such effects for samples size of 800 and 1200 are presented in Figures 4.4 and 4.5, respectively. Table 4.1 summarizes such effects qualitatively. Tables 4.2 to 4.7 shows the hyper parameter for each model of both purity and recovery variables. Figure 4.6 shows the same evaluation of the accuracy of the models by using the inverse RMSE. The alternate metric yields similar results to that of Figure 4.3, using R². The choice for the inverse RMSE was done in order to allow a more straightforward comparison between both metrics in the same score scale (i.e., higher score = better fit), as the conventional RMSE metric would relate lower values to better fits. The same metric is used to evaluate the sample sizes of 800 and 1200, which are presented in Figures 4.7 and 4.8.

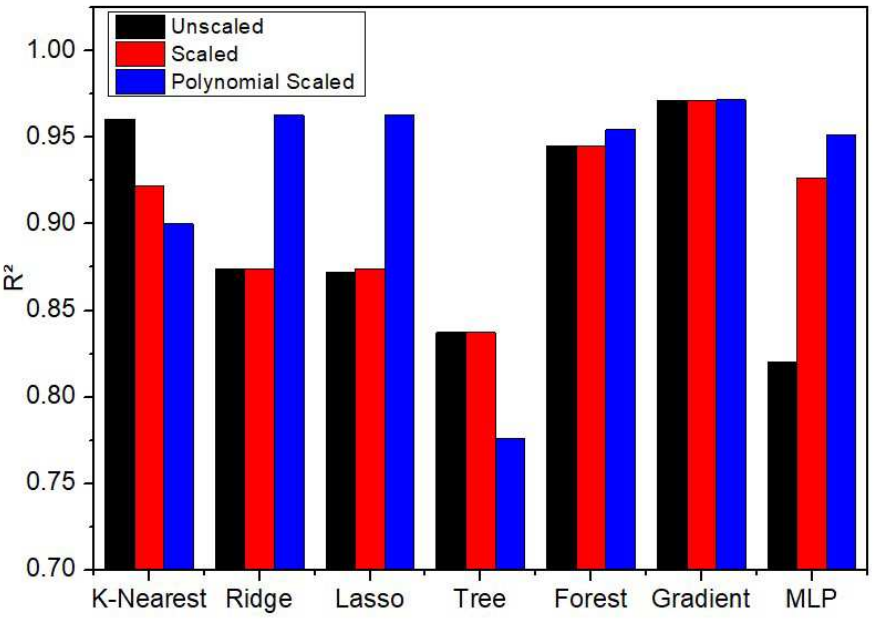
These findings provide an important insight on the decision making regarding the choice of a machine learning model. Contrary to what is to be expected, simpler linear models can 6perform significantly better than non-linear models such as the MLP neural network even with a smaller sample size. Since the MLP model only becomes consistent in higher sample sizes, choosing it for the problem at hand would require an unnecessary increase of computational resources employed to generate more data for the algorithm, while similar results could be achieved with simpler models with lower sample sizes. This implies that a wide screening of machine learning models is advised for a more efficient problem-specific modelling.

Figure 4.3 ¯ Effects of scaling for sample size = 600.



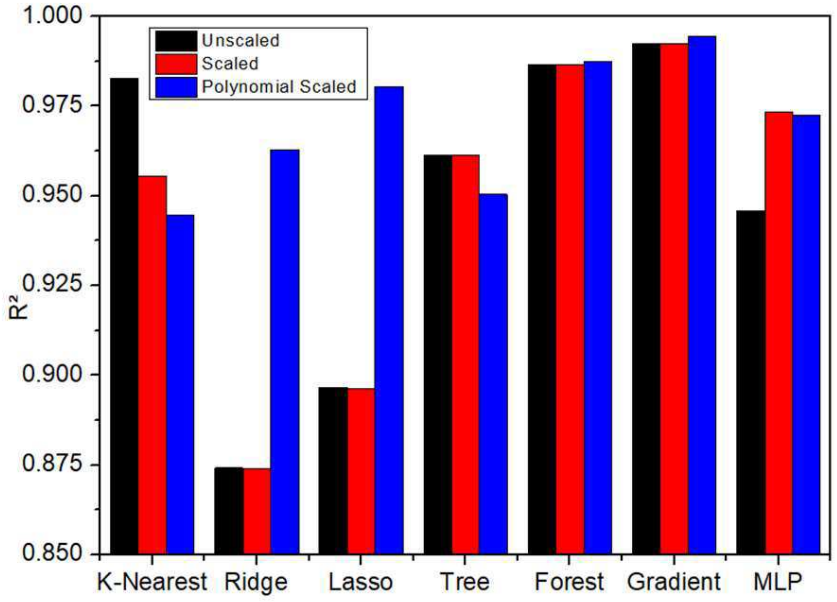
Source: Author

Figure 4.4 ¯ Effects of scaling for sample size = 800.



Source: Author

Figure 4.5 – Effects of scaling for sample size = 1200.



Source: Author

Table 4.1 – Effects of the scaling mechanisms on RД

Method	Scaled	Polynomial Scaled
KNN	Decreases	Decreases further
Ridge	Negligible effect	Large improvement
Lasso	Negligible effect	Large improvement
Decision Tree	Unclear effect	Unclear effect
Random Forests	Negligible effect	Negligible effect
Gradient boosted trees	Negligible effect	Negligible effect
MLP	Large improvement	Possible improvement

Table 4.2 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Unscaled mode – Figure 4.2 a). Variable: Purity. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

Method - Unscaled	N600(R²)	N800(R²)	N1200(R²)
KNN	4 (0.889)	4(0.929)	6(0.951)
Ridge	0 (0.887)	0(0.888)	0(0.873)
Lasso	5E-06 (0.886)	5E-06(0.881)	5E-06(0.875)
Decision Tree	13 (0.904)	8(0.82)	13(0.923)
Random Forests	100 (0.945)	56 (0.936)	100(0.972)
Gradient boosted	4 (0.965)	4 (0.958)	5(0.981)
MLP	80x10(0.753)	70x5(0.686)	100x9(0.882)

Table 4.3 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Unscaled mode – Figure 4.2 a). Variable: Recovery. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

Method - Unscaled	N600(R²)	N800(R²)	N1200(R²)
KNN	4(0.985)	12(0.979)	8(0.997)
Ridge	10(0.957)	0(0.953)	10(0.966)
Lasso	5E-06(0.957)	5E-06(0.953)	5E-06(0.966)
Decision Tree	14 (0.971)	4(0.810)	15(0.988)
Random Forests	78 (0.990)	67(0.941)	67(0.996)
Gradient boosted	4 (0.991)	1(0.981)	5(0.997)
MLP	90 x 9 (0.981)	100x9(0.992)	90 x 9(0.998)

Table 4.4 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Scaled mode – Figure 4.2 b): Purity. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

Method - Scaled	N600(R²)	N800(R²)	N1200(R²)
KNN	6 (0.878)	6(0.933)	6(0.959)
Ridge	0.30 (0.886)	0.3(0.888)	0.05(0.874)
Lasso	5E-06(0.887)	5E-06(0.888)	5E-06(0.874)
Decision Tree	13 (0.904)	8(0.82)	13 (0.923)
Random Forests	100 (0.945)	56(0.936)	100(0.972)
Gradient boosted	4 (0.965)	4 (0.958)	5(0.981)
MLP	80x4 (0.885)	100x10(0.85)	100x9(0.91)

Table 4.5 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Scaled mode – Figure 4.2 b). Variable: Recovery. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

Method - Scaled	N600(R²)	N800(R²)	N1200(R²)
KNN	8(0.945)	8(0.941)	6(0.973)
Ridge	0.15(0.957)	0(0.953)	0.1(0.966)
Lasso	5E-06(0.957)	5E-06(0.953)	5E-06(0.966)
Decision Tree	14 (0.971)	4(0.810)	15(0.988)
Random Forests	78 (0.990)	67(0.941)	67(0.996)
Gradient boosted	4 (0.991)	1(0.981)	5(0.997)
MLP	90 x 9 (0.992)	50x7(0.985)	90 x 9(0.998)

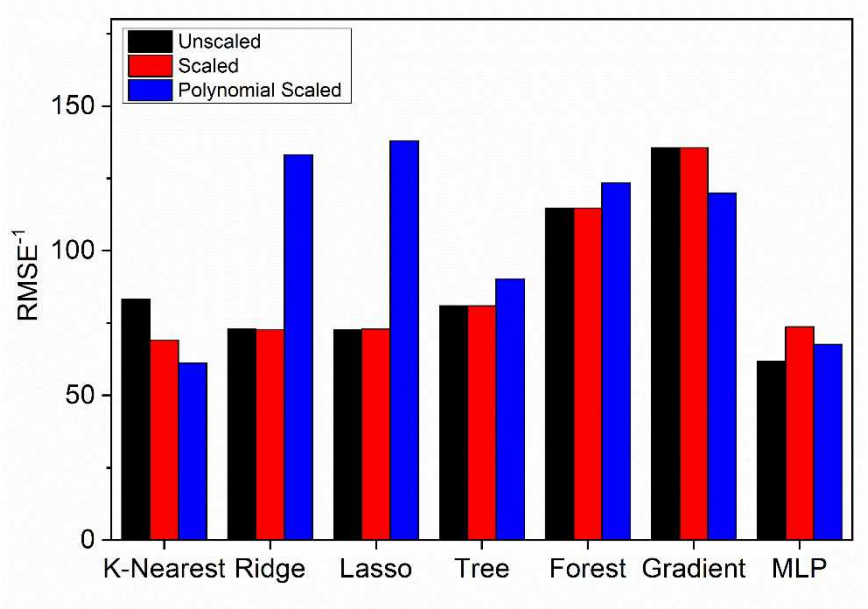
Table 4.6 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Polynomial scaling mode – Figure 4.2 c): Purity. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

Method – Poly	N600(R²)	N800(R²)	N1200(R²)
KNN	6(0.847)	4(0.913)	4(0.948)
Ridge	0.10 (0.957)	0(0.954)	0(0.969)
Lasso	5E-06 (0.957)	5E-06(0.954)	5E-06(0.970)
Decision Tree	9 (0.919)	17(0.861)	12(0.928)
Random Forests	100 (0.957)	100(0.941)	45(0.978)
Gradient boosted	3 (0.956)	4 (0.950)	5(0.989)
MLP	100x7 (0.847)	90x4 (0.85)	100x9(0.92)

Table 4.7 – Tuned hyperparameters of each machine learning algorithm for each tested sample size. Name of the parameters as given in Table 2. Unscaled mode – Figure 4.2 c). Variable: Recovery. (MLP parameter reading, A x B; A = Layer size; B = Number of hidden Layers).

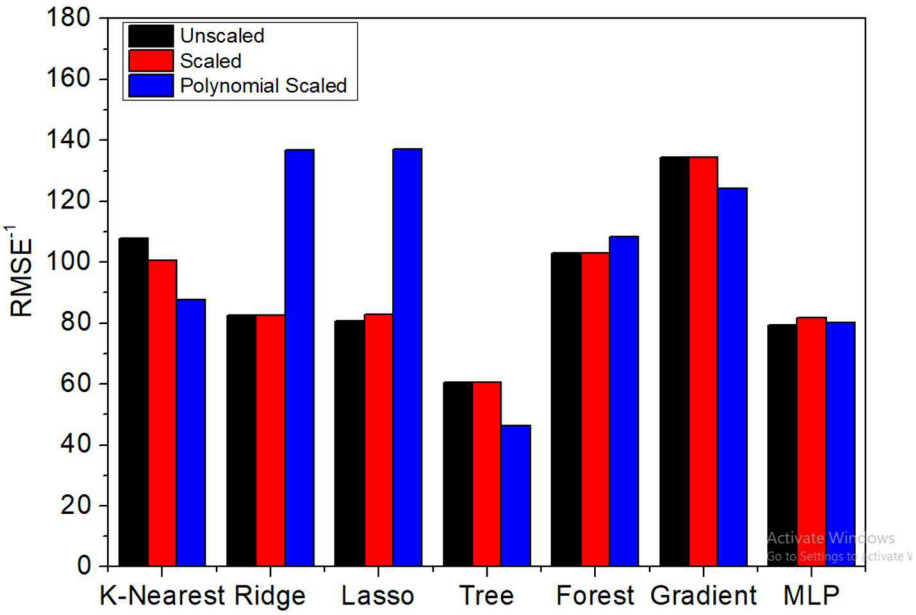
Method - Poly	N600(R²)	N800(R²)	N1200(R²)
KNN	6(0.924)	8(0.915)	4(0.963)
Ridge	0.05(0.994)	0(0.989)	0(0.997)
Lasso	5E-06(0.995)	5E-06(0.989)	5E-06(0.997)
Decision Tree	12(0.980)	7(0.57)	13(0.985)
Random Forests	78 (0.990)	67(0.955)	100(0.995)
Gradient boosted	4 (0.987)	1(0.979)	4(0.997)
MLP	100x9 (0.991)	100x 9(0.992)	90 x 9(0.998)

Figure 4.6 ¯ Effects of scaling for sample size = 600 using RMSE-1 metric.



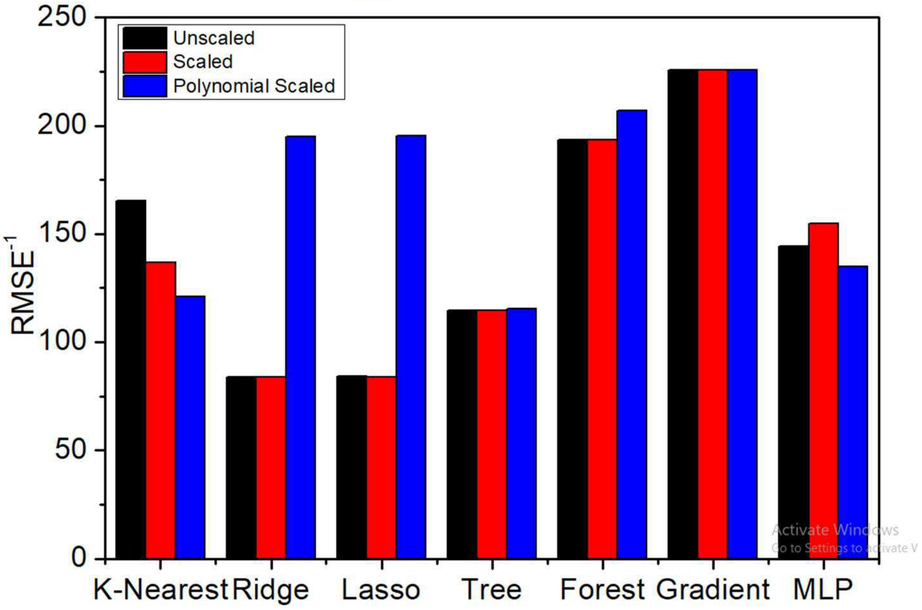
Source: Author

Figure 4.7 ¯ Effects of scaling for sample size = 800 using RMSE-1 metric.



Source: Author

Figure 4.8 ¯ Effects of scaling for sample size = 1200 using RMSE-1 metric.



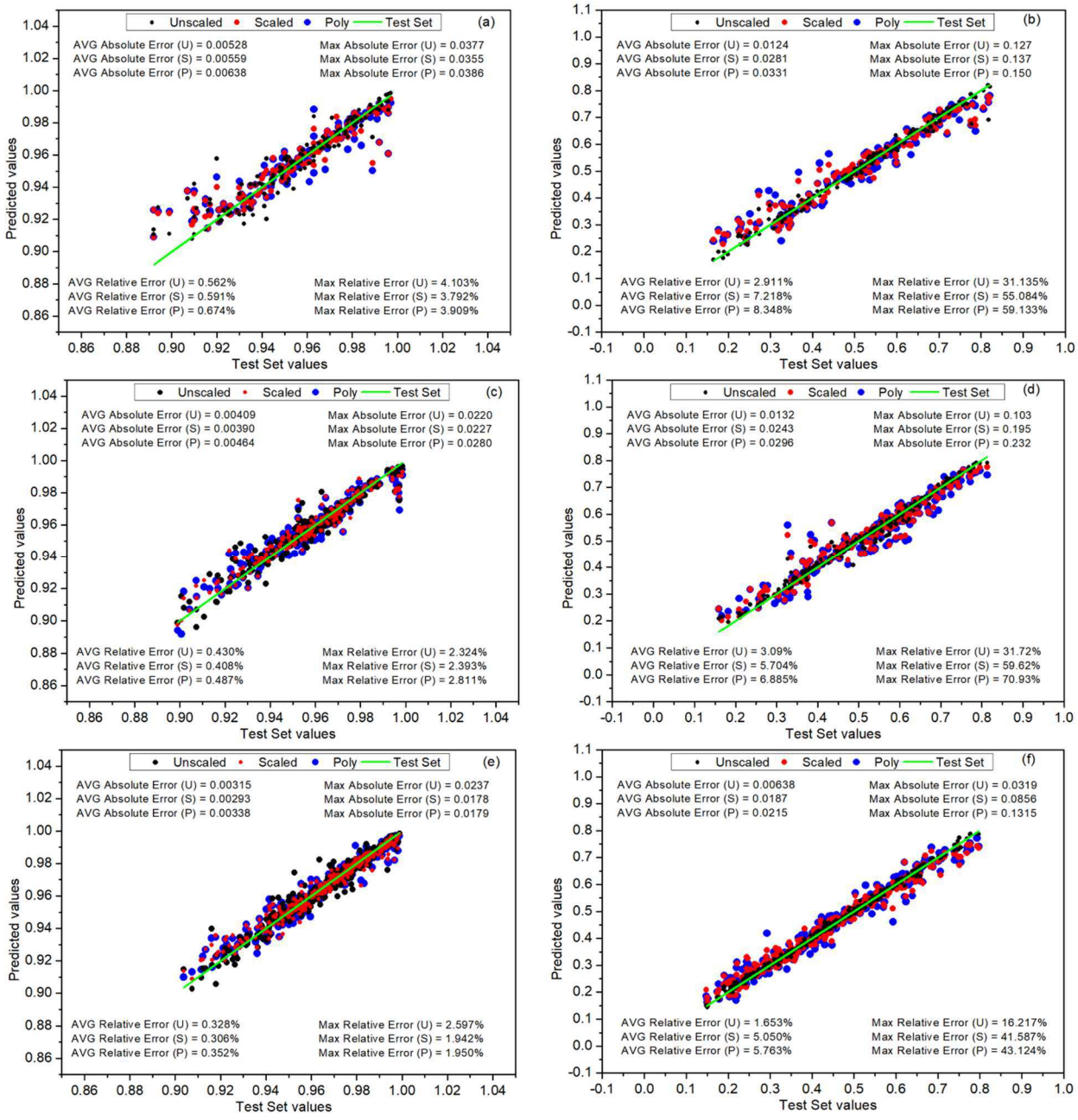
Source: Author

Additionally, Figures 4.9 to 4.14 shows the comparison between the predicted values by the machine learning models with the detailed model values, with additional measures of Average and Maximum absolute error (AAE) and average and maximum relative error (ARE), Equations 4.1 and 4.2 with 俅being the actual value and 俄the predicted value and 俆the number of observations. The graphs present the predicted values for all 3 scaling scenarios on all sample sizes, with the abbreviations of `U, S, P_ meaning, respectively, Unscaled, Scaled, and polynomial scaling. The Ridge and Lasso models are collectively represented by Figure 10 due to them predicting almost the same values.

俥俳俯 唬 ⼆俵ₙ 岜 俶ₙ俿 / 俧 (4.1)

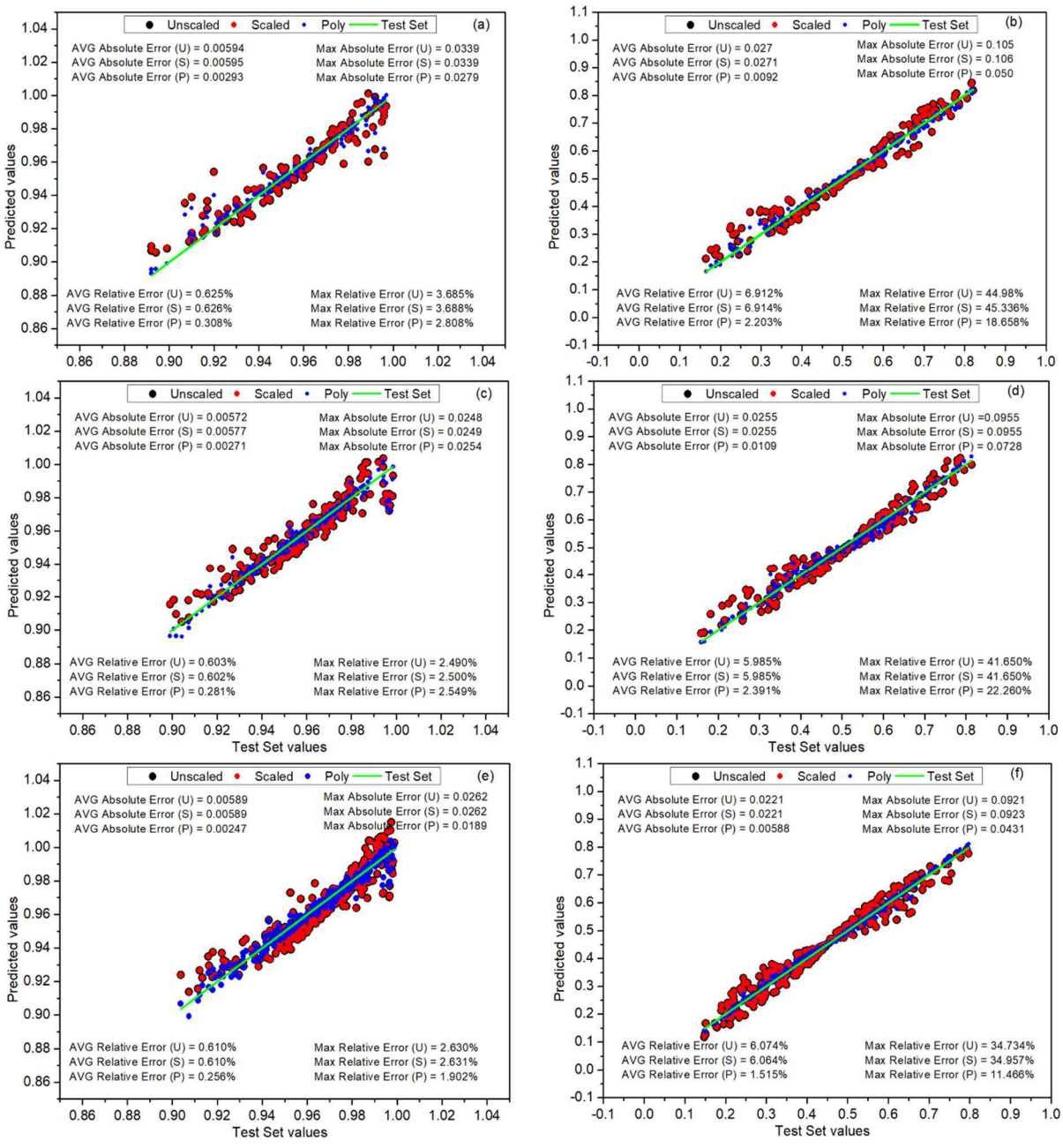
俥俳俯 唬 ⼆ 俵ₙ 岜 俶ₙ俿 / 俵ₙ / 俧 (4.2)

Figure 4.9 ¯ Comparison of predicted values by the K-Nearest Neighbors model with the detailed model test set values on (a) Purity 600 size , (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



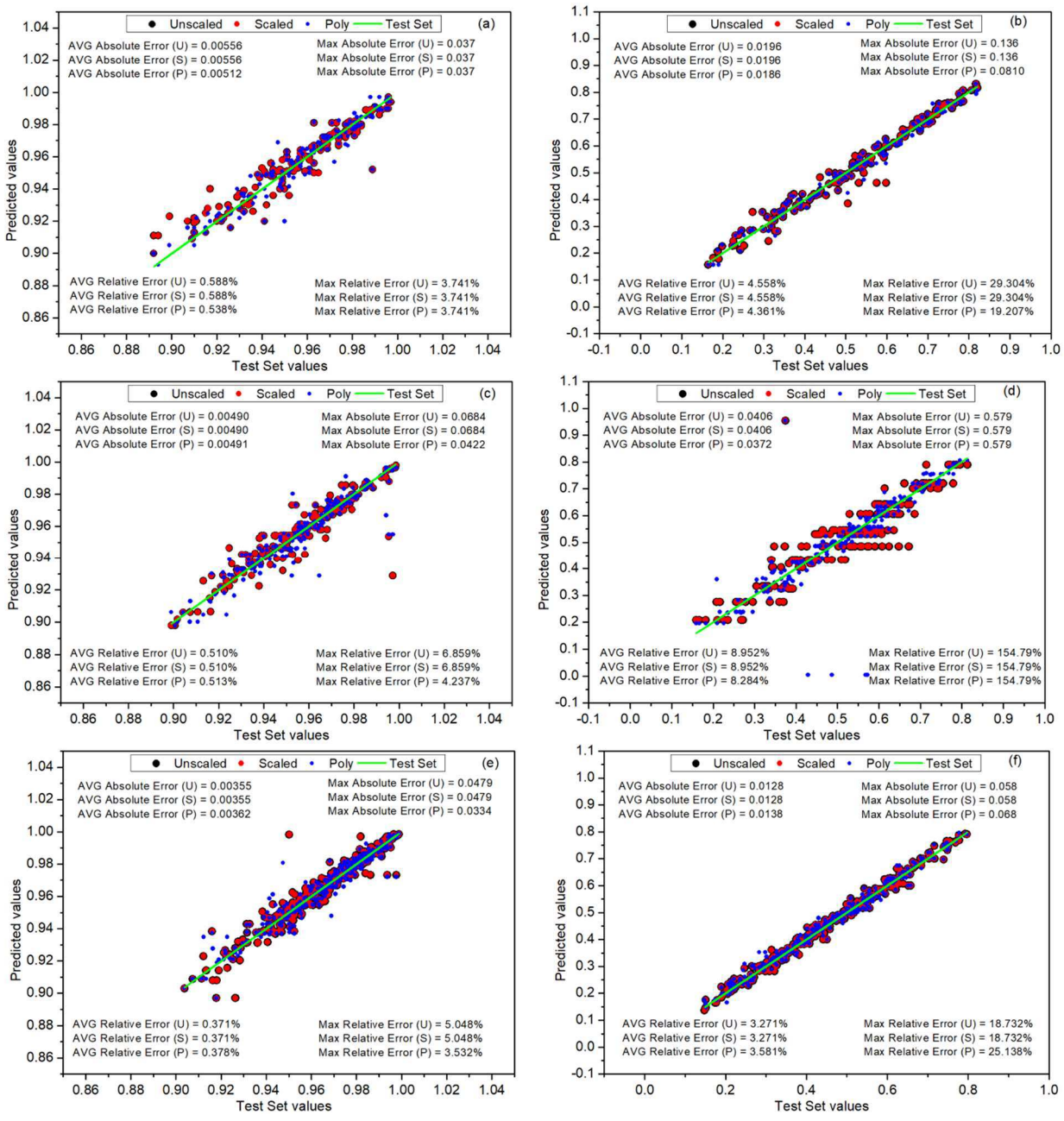
Source: Author

Figure 4.10 ¯ Comparison of predicted values by the Ridge and Lasso models with the detailed model test set values on (a) Purity 600 size, (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



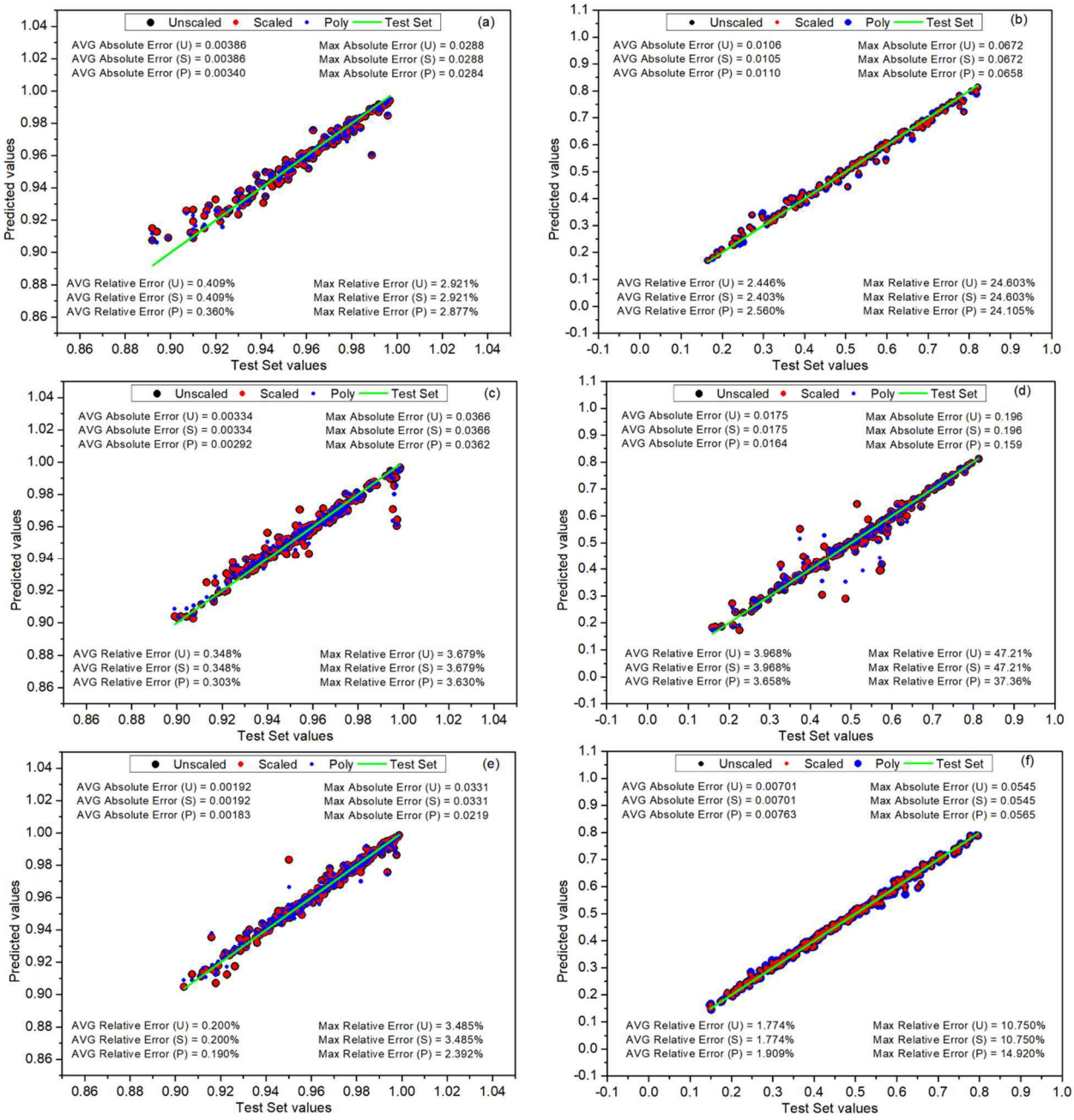
Source: Author

Figure 4.11 – Comparison of predicted values by the Decision Tree model with the detailed model test set values on (a) Purity 600 size, (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



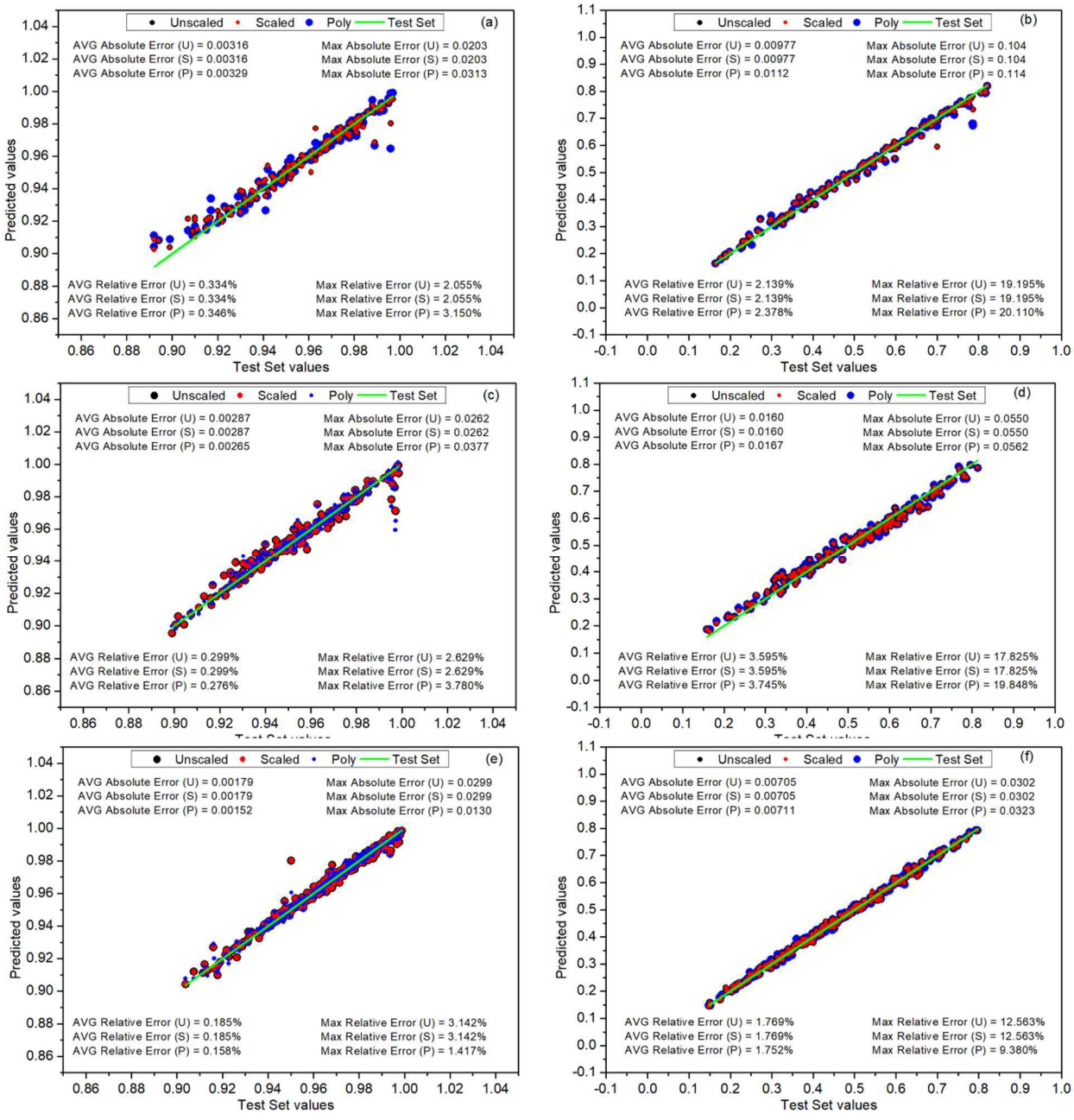
Source: Author

Figure 4.12 ¯ Comparison of predicted values by the Random Forests model with the detailed model test set values on (a) Purity 600 size, (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



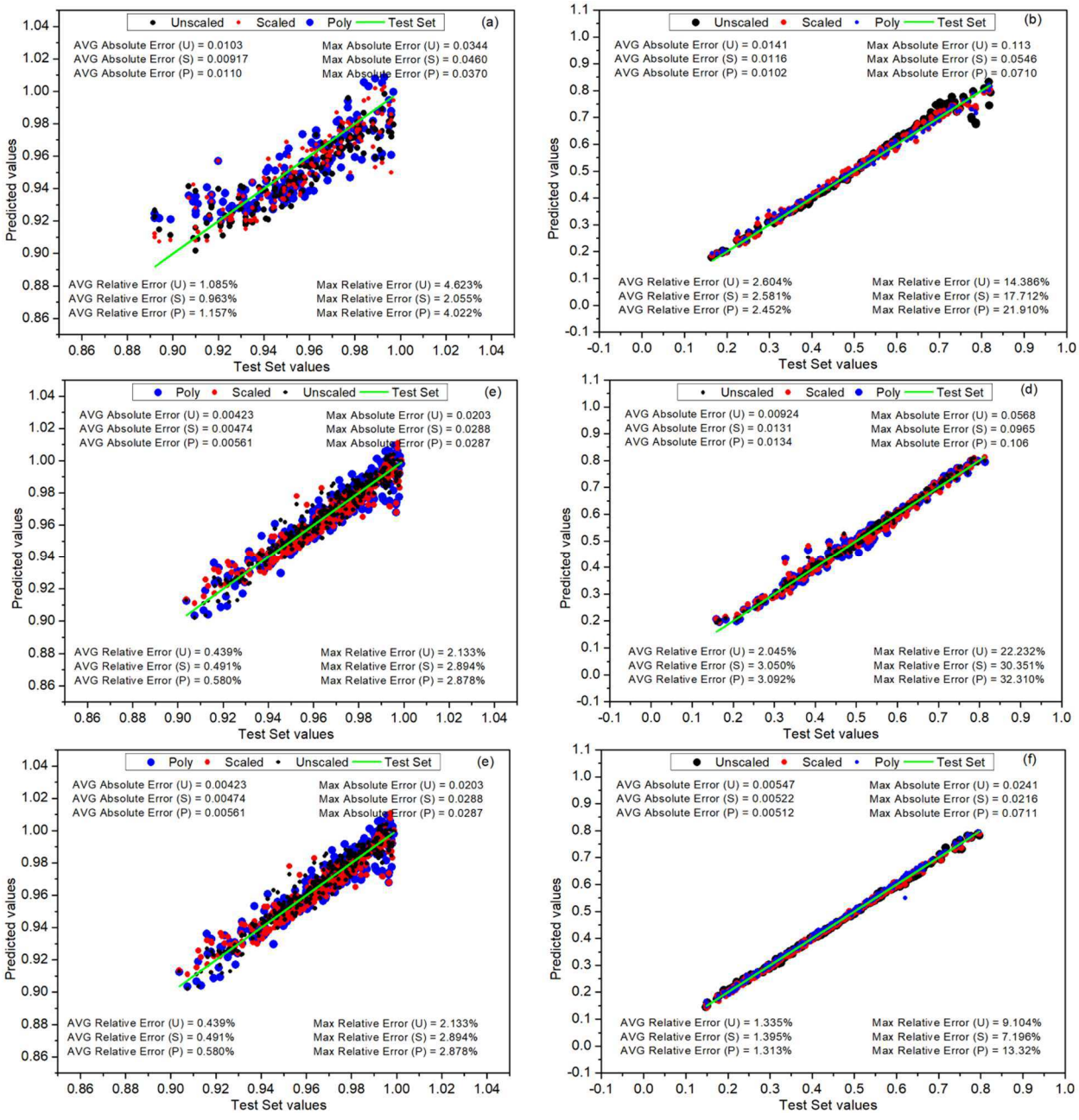
Source: Author

Figure 4.13 – Comparison of predicted values by the Gradient Boosted trees model with the detailed model test set values on (a) Purity 600 size, (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



Source: Author

Figure 4.14 ¯ Comparison of predicted values by the Neural networks (MLP) model with the detailed model test set values on (a) Purity 600 size, (b) Recovery 600 size, (c) Purity 800 size, (d) Recovery 800 size, (e) Purity 1200 size and (f) Recovery 1200 size.



Source: Author

4.2 Hyperparameter and training critical Analysis

This section describes in more detail the accuracy results of section 4.1 as well as the anomalous results that were shown, while showing the training part of each models with its hyperparameters in more detail.

4.2.1 Neural Networks performance

It˘s possible for Neural Networks to perform worse than algorithms like Random Forests and Gradient Boosted regression trees, however performing worse than linear models is not to be expected, even while unscaled, Tables 4.2 to 4.7 show that the worse behavior is due to the purity variable, as the RДvalues for the recovery are largely superior to the linear models, thus the discussion focuses on this variable. Given that the simplest neural network possible is one with zero hidden layers, it becomes a linear regression model, when no activation function is provided, thus at the very least, the MLP model should have been performing similar to both linear models tested. Thus, its possible that poor hyperparameter training and overfitting are the cause of the worse accuracy, especially considering that the hyper parameters of table 4.2 shows a complex model of 10 hidden layers each with 80 neurons for the 600 sample size, which would possibly indicate that the model is too complex for the data size it fits.

As a sanity check, Figure 4.15 shows a snippet of a code training the MLP model of the purity variable to the 600 size sample, without any hidden layers at all, no regularization parameters, and no activation functions. The resulting negative RДshows that the model is not behaving as the linear model its supposed to, and hints that the table 4.2 results do not indicate overfitting.

Figure 4.15 ˘ Linear fit of the MLP model.

```
from sklearn.neural_network import MLPRegressor
M_N = MLPRegressor(hidden_layer_sizes=[] , random_state = 0, max_iter=100000,
alpha = 0, activation = 'identity')
M_N.fit(X_train, np.ravel(y_train))
M_N.score(X_test,np.ravel(y_test))
res7 = M_N.predict(X_test)

print('R² score : {}' .format(M_N.score(X_test,y_test)))
```

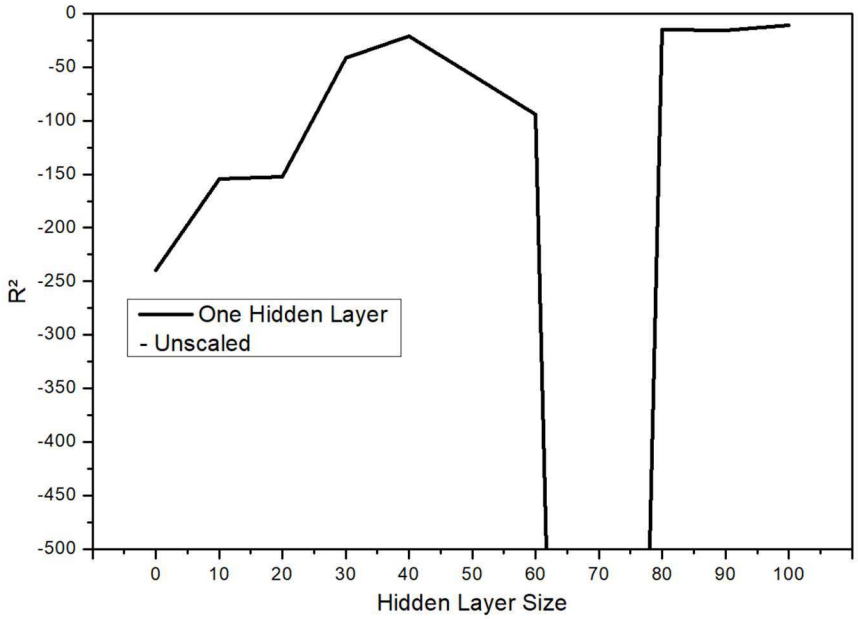
```
R² score : -176.81515216245134
```

Source: Author

Indeed, as Figure 4.16 shows, when doing a parametric sweep of the model using one hidden layer from sizes 0 to 100, it shows that, though almost every model has an abysmal score, the ones with higher complexity perform on average better than the linear one. Figure 4.16 shows both the training score and the cross-validation score are low, which in fact would indicate underfitting at a first glance. This is further shown by Figure 4.17 showing the entire

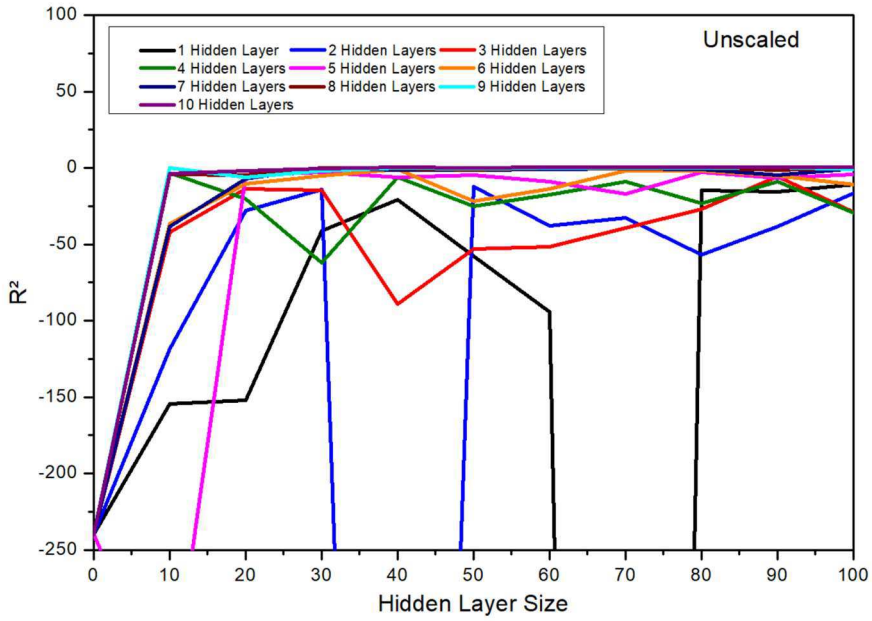
parametric sweep from 0x0 to 100x10 hidden layer sizes, though some oscillations happen, on average the models with more layers and more neurons on each layer perform better, which culminates with the 80x10 value of Table 4.2, as the more complex models are one of the few ones that actually provide a non-negative R².

Figure 4.16 – Unscaled hyperparameter training of the MLP model with one hidden layer



Source: Author

Figure 4.17 – Unscaled hyperparameter training of the MLP model with multiple layers



Source: Author

Further, also from Figure 4.17, it shows that once the model reaches enough layers, about 7 and above, increasing the number of neurons produces diminishing results, thus a model of 40x7 layer sizes performs similarly to the chosen 80x10, only with a small difference of decimals in the R², even though one model is much more complex than the other.

These number of anomalous results are, instead, a not uncommon behavior of neural networks when no data scaling is used, in addition to smaller sample sizes, and its likely due to the behavior of the solver of the model. The used solver, adam, is a modified gradient-descent solver with adaptative learning rates, thus, it only consider first order gradients in the optimization algorithm to obtain the weights of the function, this is in contrast with traditional newton-method optimizations, which considers second order gradients. Given a function 𝑓(𝑥) and a step size parameter 𝛼 the minimization by the gradient descent and Newton's method are shown respectively by Equations 4.3 and 4.4

$$x_{n+1} = x_n - \alpha \nabla f(x_n) \tag{4.3}$$

$$x_{n+1} = x_n - \alpha \frac{\nabla f(x_n)}{\nabla^2 f(x_n)} \tag{4.4}$$

With the first order method, in an unscaled scenario, gradients may be dominated by the larger features, ending up in slow or convergence to sub-optimal values. To test this hypothesis, scikit learn has a solver option called Limited-memory Broyden–Fletcher–Goldfarb–Shanno Algorithm (LBFGS) which is a quasi-newton method that accounts for second order gradients, which can be used to train the model in its simplest form. This solver does produce reasonable results with an R² of 0.882 being very close to the Ridge and Lasso unscaled values of 0.887, Figure 4.18.

Figure 4.18 – Linear fit of the MLP model with LBFGS solver.

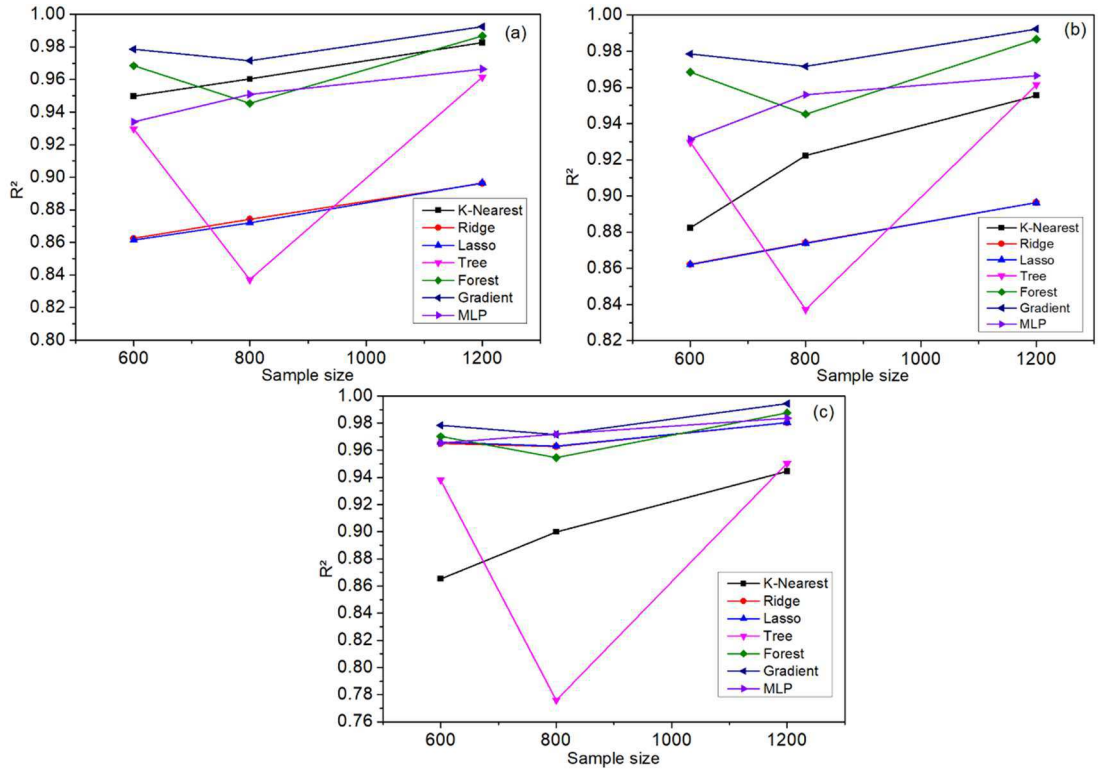
```
from sklearn.neural_network import MLPRegressor
M_N = MLPRegressor(hidden_layer_sizes=[] , random_state = 0, max_iter=100000,
alpha = 0, activation = 'identity', solver = 'lbfgs')
M_N.fit(X_train, np.ravel(y_train))
M_N.score(X_test,np.ravel(y_test))
res7 = M_N.predict(X_test)

print('R² score : {}' .format(M_N.score(X_test,y_test)))
```

```
R² score : 0.8822056218670024
```

Source: Author

The cost of superior convergence of the quasi-newton method comes with significantly more computational resources needed to finish the training, which made it unviable to train the whole range of 0x0 to 100x10 hidden layer sizes that was trained with the adam solver. However, using a narrower search domain, limiting the number of hidden layers to 4 and neuron size from 10 to 100 with 10 step size, a general improvement is found and the new standing of the ANN compared with the other models can be visualized in Figure 4.19 for all scaling scenarios. Table 4.8 shows the individual values of RДfor both purity and recovery and their hyperparameters.

Figure 4.19 - Performance of MLP model with the LBFGS solver



Source: Author

Table 4.8 - Tuned hyperparameters of MLP with their individual RД purity and recovery.

Variable	N600(RД	N800(RД	N1200(RД
Purity Unscaled	60x4(0.887)	70x4 (0.910)	80 x 4 (0.935)
Recovery Unscaled	90 x 4 (0.981)	100x4(0.992)	90 x 4(0.998)
Purity Scaled	80x3 (0.872)	100x4(0.927)	80x3(0.935)
Recovery Scaled	90 x 4 (0.992)	50x4(0.985)	90 x 4(0.998)
Purity Poly	0x0 (0.94)	0x0(0.952)	0x0(0.969)
Recovery Poly	100x4 (0.991)	100x 4(0.992)	90 x 4(0.998)

This shows a significant improvement of the MLP model and the results are more of a match with what was expected, being overall the 3th-4th best performer among all scaling scenarios. It also highlights the impressive performance of the K-nearest neighbors˘ model in the unscaled mode, able to outperform all but the gradient model. Thus, it has been shown that the underperformance of the MLP model mainly in the lowest sample sizes and while unscaled, was due to the solver choice.

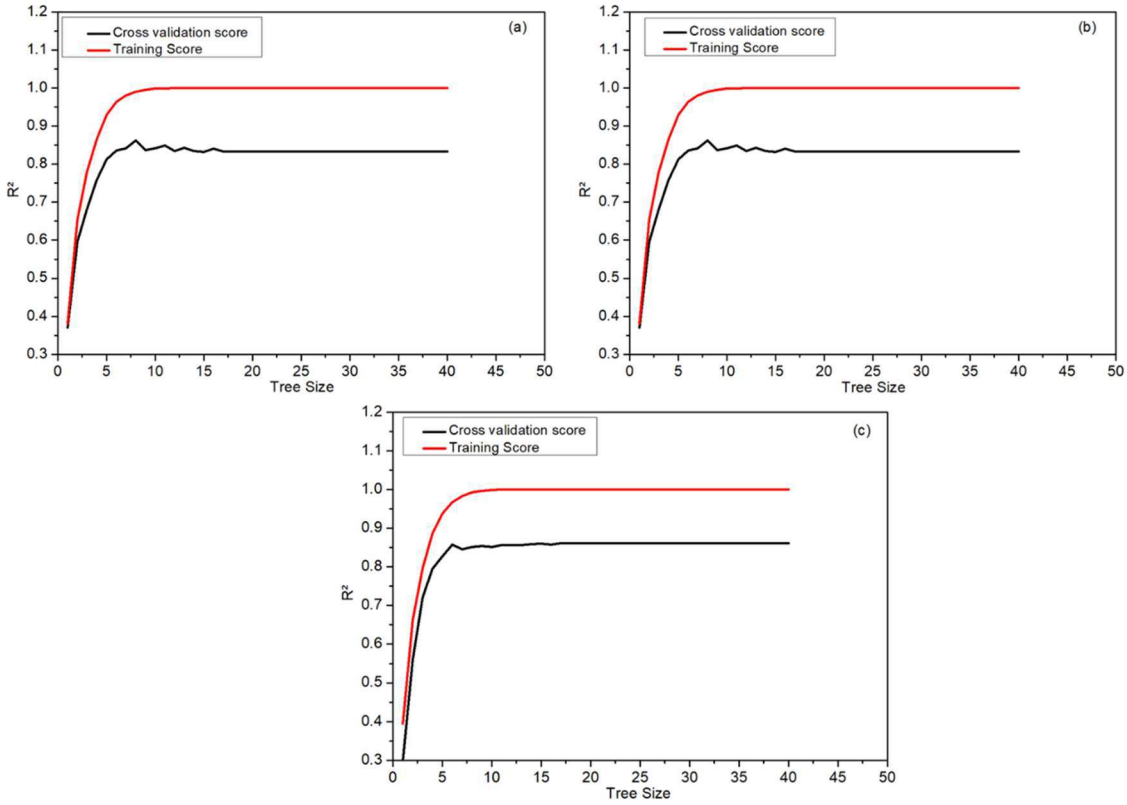
Another possibility of the low performance was that the layers tested consistent on layers of the same size, which might not be the best way for the algorithm to approach. Due to the size of the neural networks, it not possible to confirm with certainty that some possible neuron structure in the domain of 0x0 to 100x10 hidden layers is better than those so far tested, since there are more than 10^{20} possible combinations in the domain. However, a testing with LHS over 5000 possible neuron configurations on the default adam solver with 600 data samples, on the purity variable, provided a best R^2 of 0.78 on the Unscaled scenario, 0.88 on the Scaled scenario and 0.87 on the polynomial scaling scenario, which don˘t significantly differ from the obtained original results, but may hint on a marginal possible improvement of randomizing the hyperparameter selection.

However, the evaluation above was very computationally costly, requiring an approximated runtime of 8 hours to fully train all the models, which hints that more modest sample sizes for the hyperparameter be utilized as a reasonable tradeoff for a small improvement in accuracy.

4.2.2 Decision Tree behavior

The large drop of accuracy of the decision tree model while raising the data size is completely unexpected, it cannot be attributed by small statistical variance since the drop goes up to 0.4 units of R^2. Some values of the hyperparameters in tables 4.2 to 4.7 may indicate under or overfitting. Figure 4.20 shows the cross-validation score for each scaling scenario for the 800 sample size for different hyperparameters, focusing on the purity variable. The behavior of the training set scores is as expected, the model tends to memorize the training pattern to reach R^2 of 1, which does not correspond to the validation score, which plateaus at about 0.8. However, there is no visible overfitting as the validation score remains stable as the tree depth increases, albeit with variations in the range of 5 to 20, where the best values, that were selected in Tables 4.2 to 4.7, stand. It remains unclear as to why the performance declined.

Figure 4.20 ¯ Hyperparameter training of the decision tree model at 800 sample size, purity variable for (a) Unscaled, (b) Scaled, (c) Polynomial scaling scenarios.



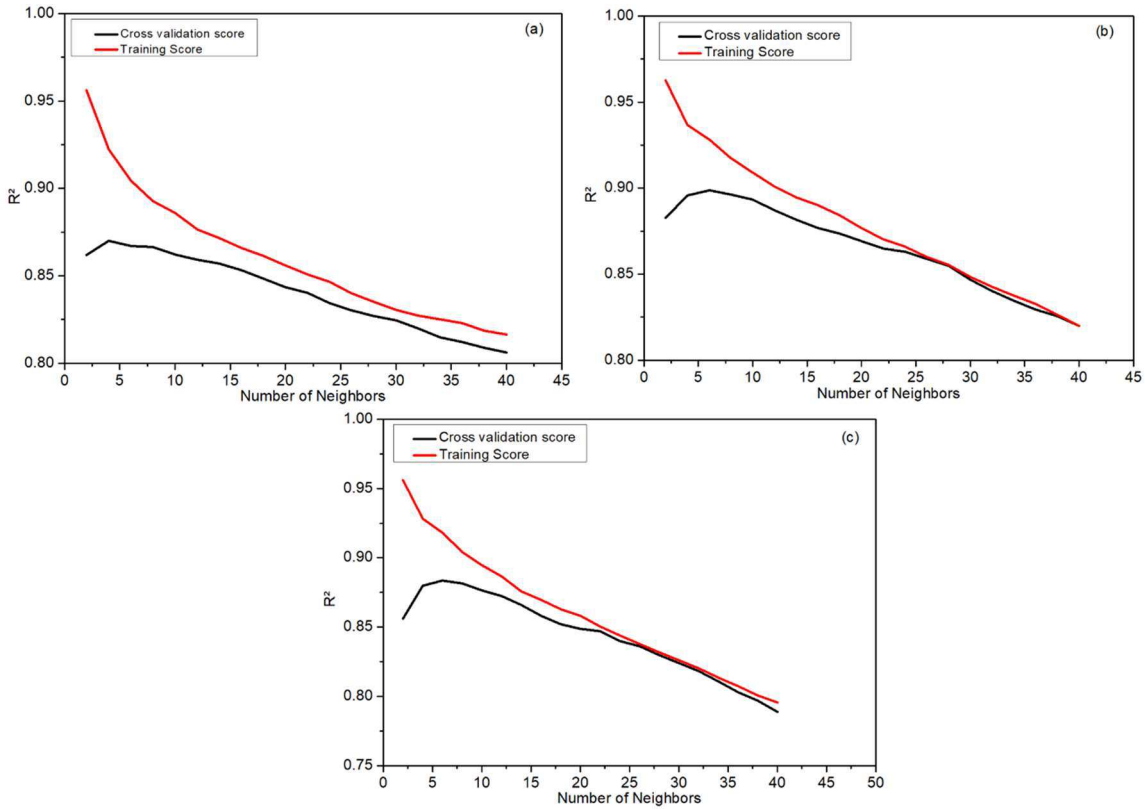
Source: Author

4.2.3 Remaining models

Given that the remaining models didn˜t exhibit any unusual behavior, the following analysis will be brief compared to the previous ones, the analysis here will focus on the purity variable, since it˜s the one with more variability in the R Дand at 600 sample size, as the other sample size values exhibit similar qualitative behavior, differing mostly on the value of R Дand specific best hyperparameters.

For the K-Nearest Neighbors, Figure 4.21 shows an expected pattern for the training score, lower values of K tend to overfit to the training data and yield high R Дwhich consistently falls as K increases. On the validation score, however, the optimal value of K seems to be at the range of 4 to 8, so from 0 to 4 the model overfits and from 8 and beyond, underfits.

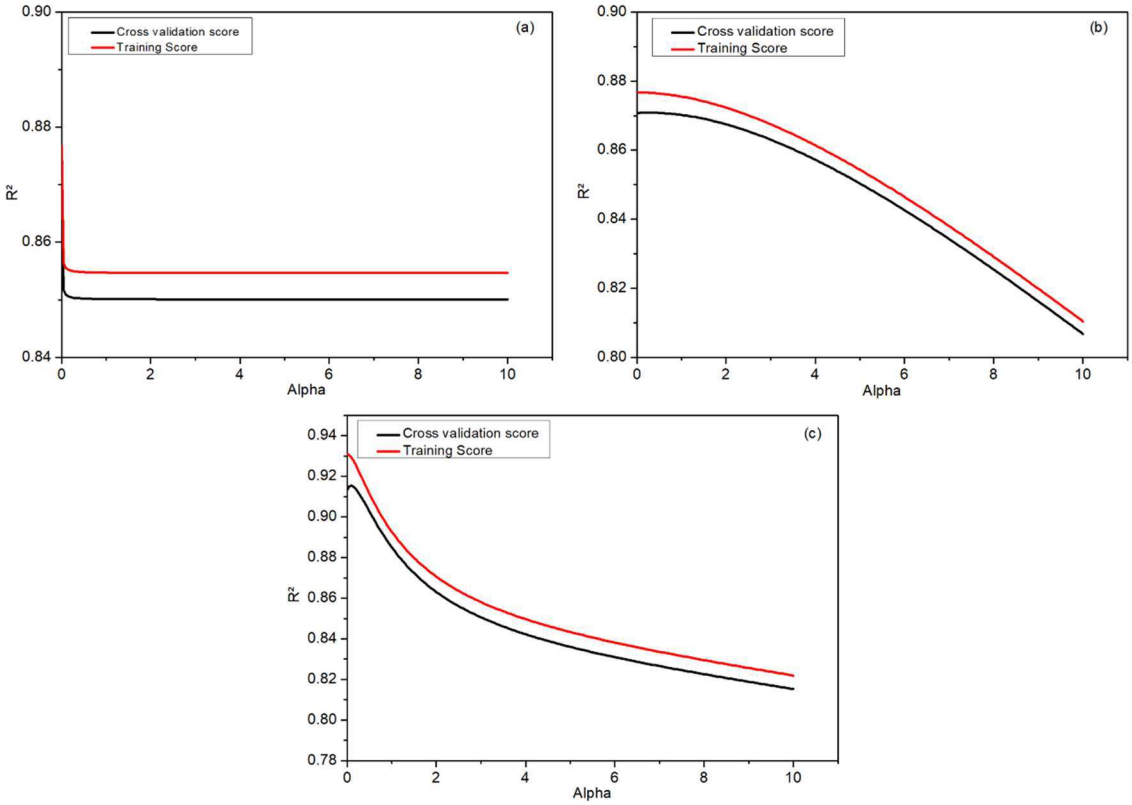
Figure 4.21 – Hyperparameter training of the K-Nearest Neighbors model at 600 sample size, purity variable for (a) Unscaled, (b) Scaled, (c) Polynomial scaling scenarios.



Source: Author

For the Ridge model, Figure 4.22, it shows a consistent pattern on the training set that signals that the higher the regularization parameter, the lower the R² score, being a sharp decline in the unscaled scenario and smoother drops on the remaining scenarios. However, for the validation score, the trend has a maximum score between 0 and 0.3 for alpha, which leads to an optimal value different from 0 in the unscaled and polynomial scenarios. The lasso model exhibits an almost identical pattern, but the optimal values for R² are all found with the smallest regularization parameters, in the present case, 5E-06, as can be seen in tables 4.2-4.7. This indicates that all three features, Adsorption pressure, Pressurization time and Adsorption time, have a large impact on the value of purity, which is to be expected, thus, regularization, except in some fortunate cases, is ineffective on the present application.

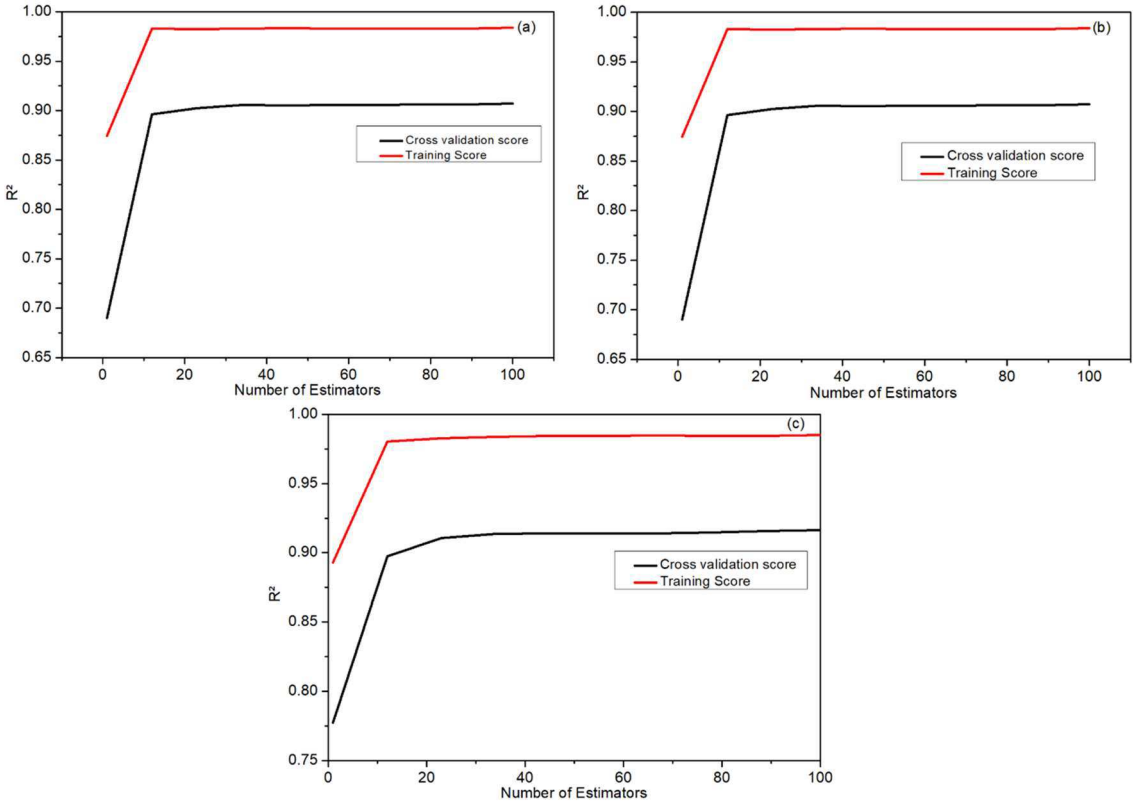
Figure 4.22 ¯ Hyperparameter training of the Ridge model at 600 sample size, purity variable for (a) Unscaled, (b) Scaled, (c) Polynomial scaling scenarios.



Source: Author

For the Random forests˘ algorithm, Figure 4.23, the training pattern is expected, as the number of estimators increase, the algorithm memorizes the training data yielding an RДof 1. This pattern is the same for the validation score, showing that increasing the number of estimators does not cause overfitting tending to a very slowly increasing asymptote at about 40 estimators with the highest RДfound at 100 estimators. However, it˘s important to note that, although the highest RДpossible for this case is 100, this is due to very small decimals of difference, in practice, a model with 40 estimators would perform identically to a model with 100, but with the added benefit of being significantly less complex and less computationally expensive.

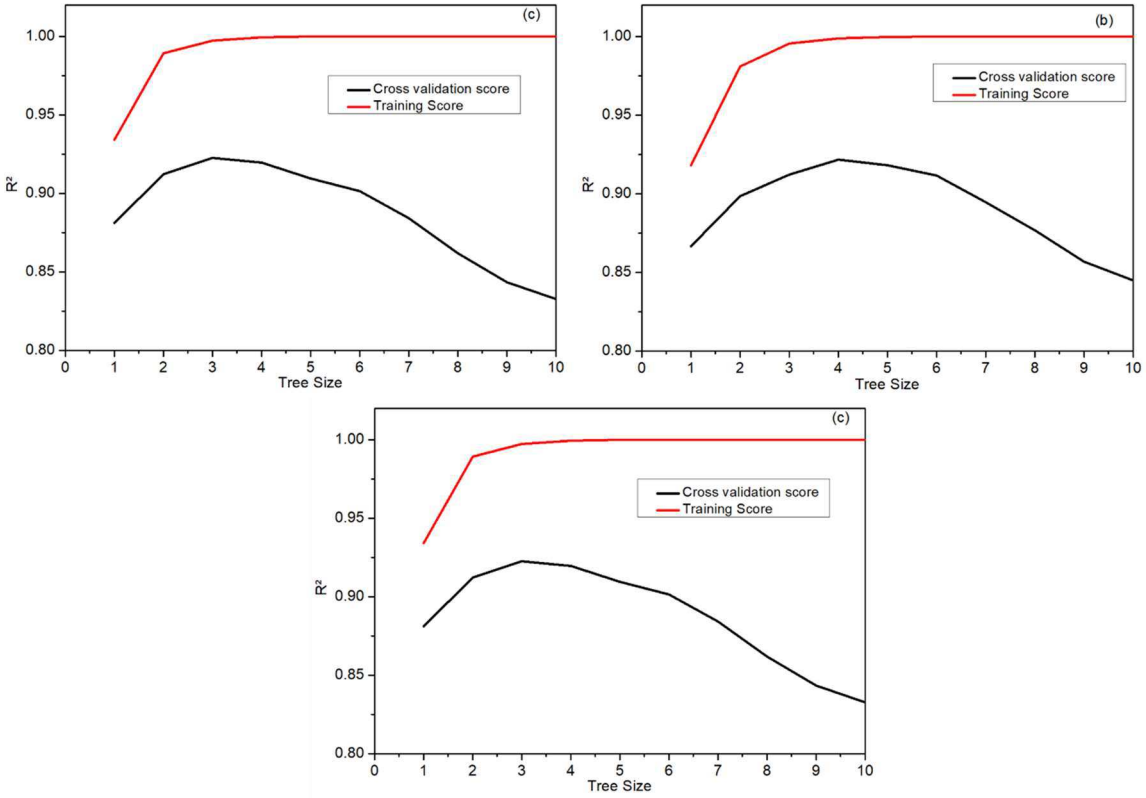
Figure 4.23 ¯ Hyperparameter training of the Random Forests model at 600 sample size, purity variable for (a) Unscaled, (b) Scaled, (c) Polynomial scaling scenarios.



Source: Author

Lastly, the Gradient Boosted Regression trees algorithm shows, Figure 4.24, the expected memorization training patterns of decision tree-based models, but differs from both Random forests and regular decision trees in that, on the validation score, it shows overfitting past a certain depth. The optimal values are generally located between 3 and 5.

Figure 4.24 – Hyperparameter training of the Gradient model at 600 sample size, purity variable for (a) Unscaled, (b) Scaled, (c) Polynomial scaling scenarios.



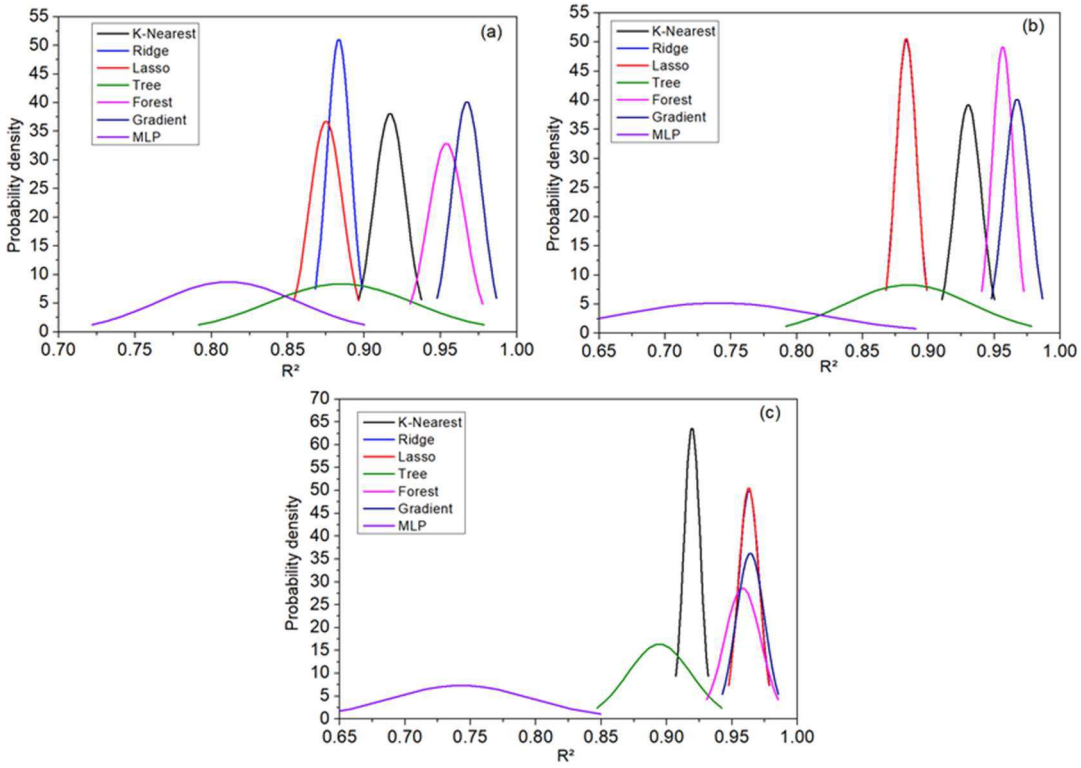
Source: Author

4.3 Model sensitivity to RNG parameter choice

As previously explained, each algorithm was tested to check how sensitive they are to the pseudo-random functions used to split data into the training, validation and test sets and to perform the cross validation. Since the choice of the parameter is up to the user, several options can be taken into account and the desired outcome is that the chosen machine-learning model is not affected significantly by this choice. Reproducing the same procedure in the previous section, each test was done separately for the unscaled, scaled and polynomial scaling scenarios.

Figure 4.25 shows the sensitivity of the algorithms for a fixed sample size of 800. One should note, however, that the R^2 of those figures are representative of only the purity parameter, while the results presented in section 3.1 are an average for the fits of both purity and recovery, the reason for choosing purity is due to this variable being the one that presents the most variability in R^2 model by model.

Figure 4.25 ¯ RNG sensitivity for a sample size of 800 for (a) Unscaled, (b) Scaled and (c) Polynomial scaling scenarios.



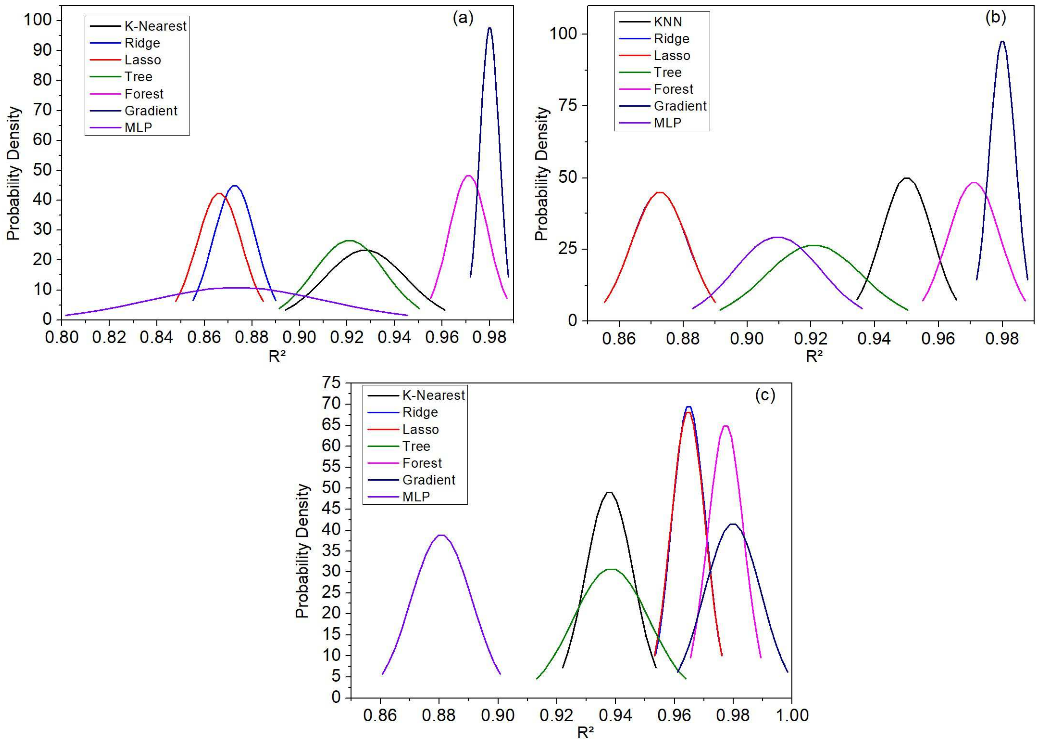
Source: Author

According to the probability densities shown in Figure 4.25, MLP model is clearly more sensitive to the RNG parameters than the other models. Besides, in opposition to what was observed in the accuracy tests, scaling does not change this behavior. The decision tree model also exhibits a large variation, but not as large as the ANN type model. That represents another unexpected result given the robustness of the neural network models and the multiple possible hyperparameters that can be tuned to improve fitting. In the present situation, despite the proper scaling, the MLP model is still capable of yielding bad fits. This unreliability would not be acceptable for an application.

Figure 4.26 shows the sensitivity for a sample size of 1200. All models have improved their precision, especially the MLP model. Considering that such improvement only happens with an increased sample size, which requires more computational effort, the models that perform the same or better for smaller sample sizes are preferred. In that case, again, the Random Forests Model and the Gradient boosted trees model were the best performing for all the types of scaling and the Ridge and Lasso models were also at the top for the polynomial scaling. The fact that these linear models have very good accuracy and precision is a great result in terms of qualitative analysis and optimization, as the equations for these models are simple

and explicit in the form of [illegible] A direct analysis of their angular coefficients can help extract the most influential parameters and show how each of them affects the prediction and physical consistency of the model, which can be easily evaluated due to the explicit correlations among the variables. For optimization purposes, simpler equations greatly improve convergence and overall technical difficulties that arise by employing more complex models.

Figure 4.26 – RNG sensitivity for a sample size of 1200 for (a) Unscaled, (b) Scaled and (c) Polynomial scaling scenarios.



Source: Author

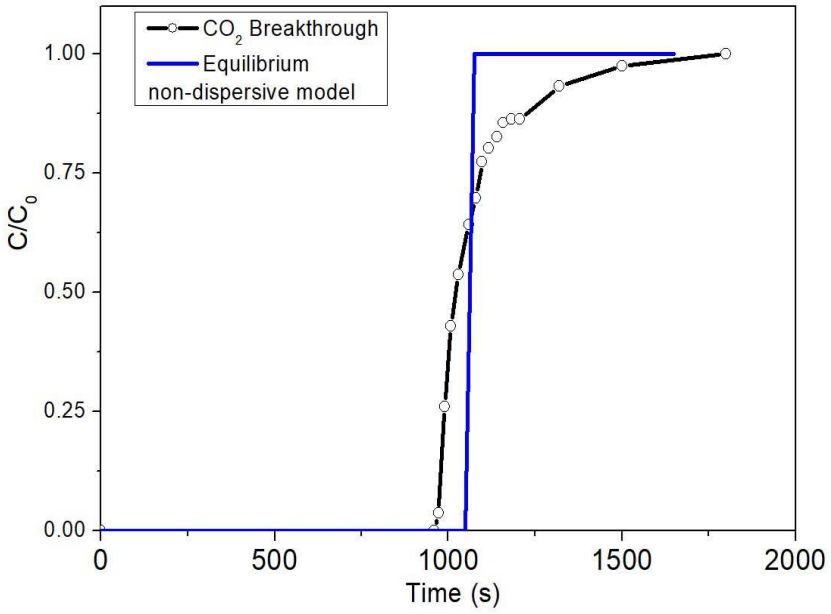
4.4 Improving model accuracy with process knowledge

Although it is possible to tweak and experiment with any machine learning and statistical modelling algorithm until it yields an adequate R^2, such manipulations are more likely to lead to physically meaningless conditions. There are no guarantees that pushing the data science work to its limits for a particular problem will work out for other problems, even within the same topic. However, it is possible to introduce additional knowledge about the process into

the machine learning without the use of any complex differential equations. The introduction of these new variables can help achieving not only a better accuracy in a given sample size, but also improve the general accuracy with less sample sizes, which is the most desirable outcome of surrogate modelling: high accuracy with the least amount of computational resources used.

To illustrate that in adsorption processes, we start by plotting Figure 4.27, which shows the experimental breakthrough curve of CO_2 and the simulated breakthrough curve according to the equilibrium non-dispersive model as in Ruthven (1984) (i.e. instantaneous adsorption and no axial dispersion) under the same operating conditions. The similarity regarded by the steepness of both curves suggests that mass transfer resistances are not significant, and the experimental tests by Siqueira et al. (2018a) show that this is the case. Therefore, we can correlate the purity variable with the mean retention time parameter (Ruthven, 1984) or stoichiometric time, as defined in Equation 4.5 alongside the other variables already in the model, 偭贶呗, 傜鸦区, 傜絮当疬. This correlation can be maintained as long as blowdown pressure is low, ensuring desorption. The relationship between raffinate purity in a PSA and the stoichiometric time of the heavy component would be that, if the adsorption phase of the PSA (傜鸦区喵傜絮当疬) is longer than the stoichiometric time, a significant drop in N_2 purity would be expected. For the machine-learning model, the new information comes from the equilibrium isotherm of CO_2 on the AC CHARBON 500, as an additional constraint, which would lead to an improvement in accuracy.

Figure 4.27 - CO_2 Breakthrough at 10% molar composition when adsorbed by CHARBON 500 versus the result from equilibrium non-dispersive model.



Source: Author

$$\text{[illegible]} \qquad (4.5)$$

A new variable is introduced, mathematically represented by [illegible] and the corresponding objective function of purity given by [illegible] instead of [illegible]. It is worth mentioning that there is effectively no new independent variable introduced, as the stoichiometric time is a function of the flowrate, which is a function of [illegible] and [illegible], and also a function of the isotherm, which is also a function of pressure, so the model would end up bearing the same independent variables. While it is true that the functions [illegible] and [illegible] carry the same independent variables, they are nonetheless different functions. This can be shown by fixing all the remaining variables and changing only [illegible] This test function is represented by Equation 4.6:

$$\text{[illegible]} \qquad (4.6)$$

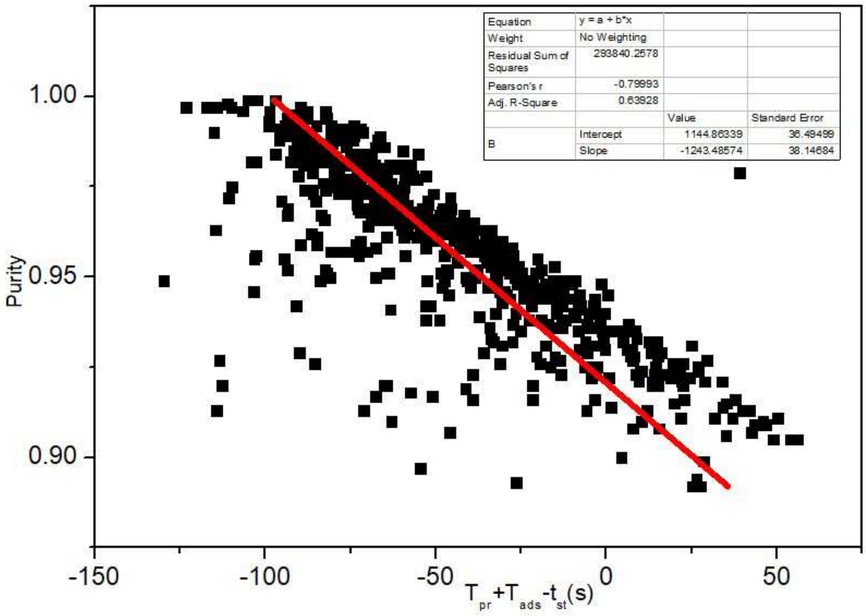
Here [illegible] can be varied at random but, analytically, if [illegible], then [illegible] That is, the function [illegible] is translationally

invariant. This result sets a clear constraint in the generic analytical function of 佤昳缅 佬缘 佲缬婧 and the proper definition of the new function is then represented by Equation 4.7 instead:

佩昕佲呲区咭佲留当佲嘅 唸 佩昕佲呲区佲留当佲嘅 础 佲呲区咭佲留当佲嘅 唦 佩然佬斑 (4.7)

Thus, the new function is the old one plus a constraint defined by a physical argument. In the present case, given the breakthrough curve, it is assumed that mass transfer resistances small enough so that the raffinate purity should have a reasonable correlation with the stoichiometric time, which carries more meaningful information, as inferred from Figure 4.28 By taking all the raw data of Purity and correlating solely with 婳驱喵佬留娃佲缬 a R^2 of 0.64 is obtained, suggesting that the new constraint indeed carries a significant portion of information.

Figure 4.28 ¯ Correlation of purity with the new variable



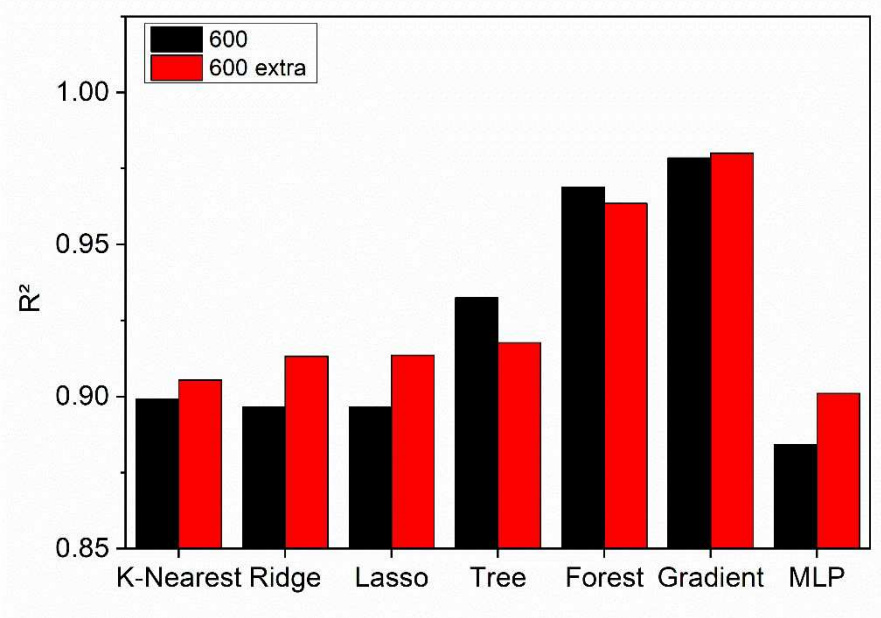
Source: Author

Figure 4.29 presents a comparison for the average R^2 in all situations (Unscaled, Scaled and Polynomial Scaling) for the sample size of 600 with (600 extra) and without (600) the stoichiometric time as a variable, for each machine-learning model. It is possible to see that the extra variable improved the performance of 5 out of the 7 models used. It is unclear why the accuracy was reduced for Random Forests and Decision Trees, although the difference for the

former is not sufficiently large to characterize a decrease in performance due to the introduction of the stoichiometric time. For the model of the Decision Trees, it might be the case of overfitting or the natural instabilities of this model as discussed in section 4.3. When applying the new variable to higher sample sizes, such as 1200, figure 4.29 shows still a large improvement for the lasso and ridge models, but no significant changes to the remaining, even with some small decrease in the case of the KNN and MLP methods.

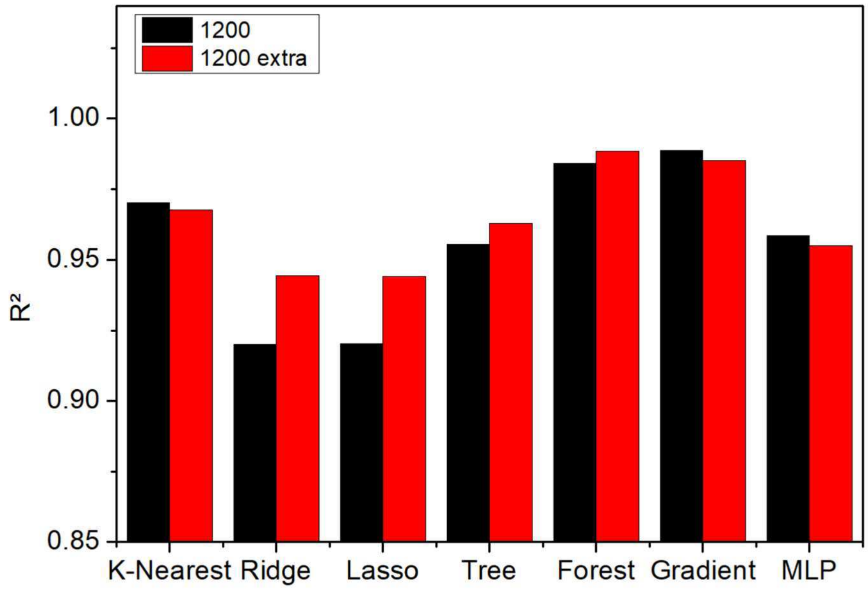
It has to be noted, though, that while intuitive that process information would cause better fitting, this improvement is not guaranteed due to the black box nature of the statistical models, though surprising, it wouldn't be impossible for the inclusion of the new information to have negative impact, as many factors outside of the physical ones influence on the accuracy of the algorithms. What is being presented is that it is worth it to try and use process knowledge in hopes of improving the model.

Figure 4.30 – Average R^2 of each method for the sample size of 600 without and with (600 extra) the addition of the stoichiometric time as a variable.



Source: Author

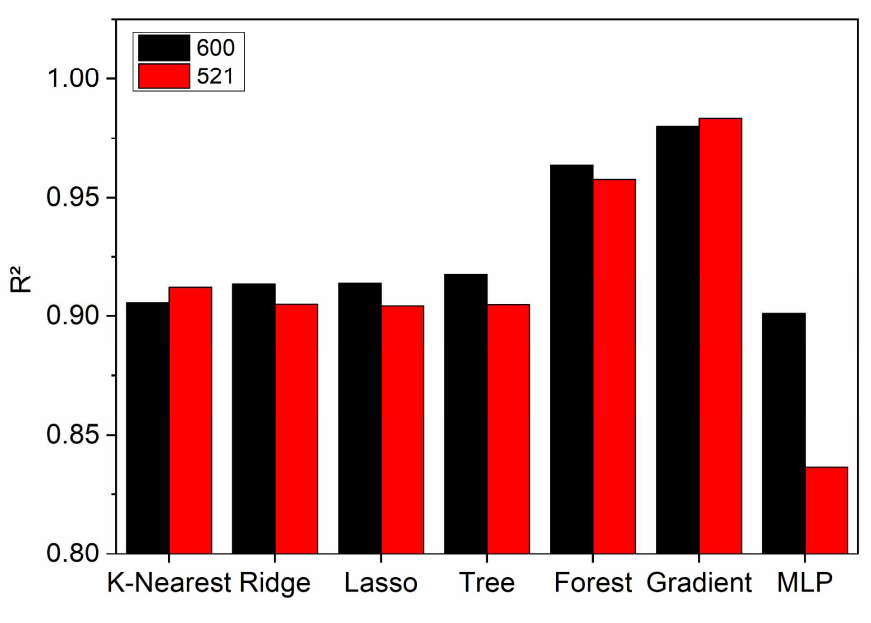
Figure 4.30 – Average R^2 of each method for the sample size of 1200 without and with (1200 extra) the addition of the stoichiometric time as a variable.



Source: Author

Provided the proper process knowledge, a step further into this approach can be taken. If the adsorption phase of the PSA exceeds the stoichiometric time, for a given adsorbent and operating conditions, the purity will be low, which is undesirable. Thus, one can beforehand use this `filter_ to exclude from the LHS sampling, the values of 媪鰯区喵偓鰯沥媫 larger than the stoichiometric time. By applying this filter onto the sample of 600 data points, the total size of the filtered sample decreases to 521. Figure 4.31 shows how the filtering affects R^2 for all the scaling scenarios.

Figure 4.31 ¯ Average R Дof each method for the sample size of 600 with the stoichiometric time compared with a `filtered_ sample of 521 data points.

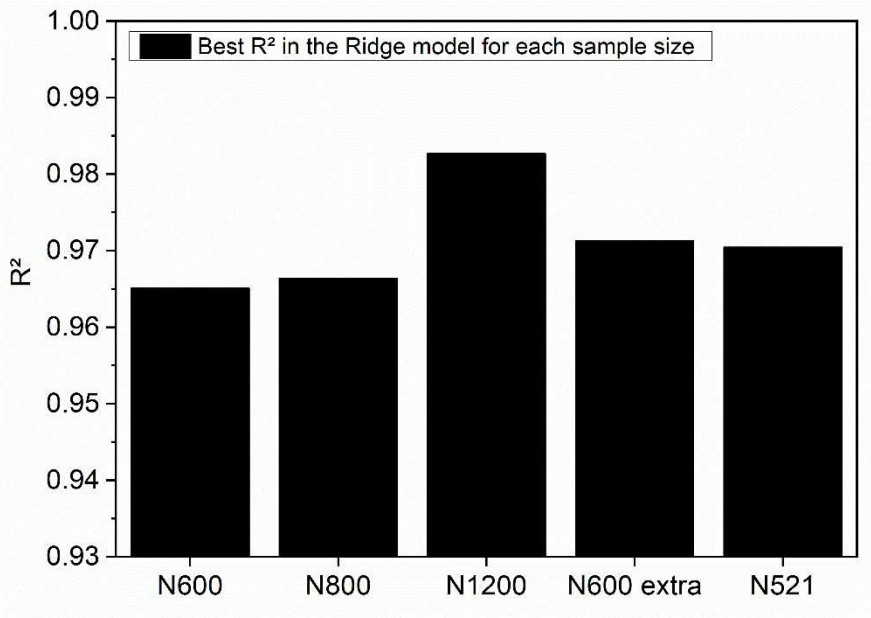


Source: Author

For every model except the MLP, the results are effectively the same, showing that there is no relevant decrease in accuracy, but the sample size required is yet again smaller, saving further computational resources. The sharp decrease for the MLP model is most likely due to its high sensitivity, as shown in section 4.3.

Finally, instead of showing how the average R Дfor all tested scenarios (Unscaled, Scaled and Polynomial scaling) compares for each model and scaling, Figure 4.33 shows the best possible fit in the Ridge model for each sample, the reason to show this specific model is due to the use of the same in the optimization and to show that even this simple model also yields very high R Дvalues. The worst performing R Дwas about 0.965, which is already high, so these are effectively all good results for an application. An even more remarkable result is that the samples of size 600 with the extra variable (600 extra) and the filtered sample of size 521 outperform both the original 600-sized sample and the 800-sized sample, demonstrating the effectiveness of the introduction of the process variable to improve the fitting.

Figure 4.33 ¯ Best R² in each sample size for the Ridge model.



Source: Author

4.5 Optimization.

To obtain the purity-recovery Pareto non-dominated front, the polynomial scaled form of the Ridge and the Lasso models were chosen due to their top performances despite the simplicity, which facilitates code implementation and convergence, despite the fact that the random forests and gradient boosted trees are the best performing methods, the simpler Ridge and lasso were chosen due to the mathematical difficulties associated with optimizing tree-based algorithms who have larger degree of discrete behavior. Also, given that the simple polynomial models were chosen, it would be possible to use alternative optimization methods that would include analytical solutions, or employment of derivative-free methods such as Bayesian optimization, however, since the code for the optimization was chosen before the models were selected as an `all purpose_ model and, as will be shown, the computational effort on the NGSA will be negligible, it was kept. The optimization was carried out with the filtered sample of 521 data points, 800 data points and 1200 data points. Table 4.9 shows the weights and the intercepts of the regressed equations of purity and recovery, according to Equation 3.11 for the variables 俄嗓嗯俑俄嗓嗯俑俄嗓嗯俑.

The NSGA-II algorithm was used to maximize the functions of purity (Equation 3.7) and recovery (Equation 3.8) subject to the following constraints: □嘵俑□嘵 and □嘵俑□嘵

with the constraint method used for scalarization. The population number of the genetic algorithm was 40 with the number of offspring of 10 and termination in the 40th generation.

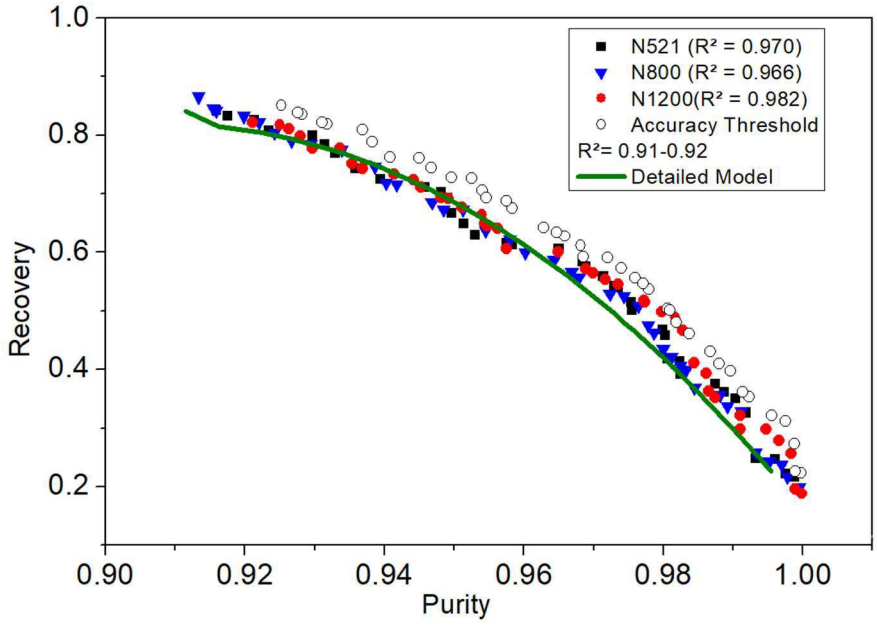
Figure 4.33 shows the results of the optimization with the Pareto non-dominated set. All evaluated sample sizes reasonably agree, which validates further the accuracy analysis of the previous sections. Since the results from the smallest sample size equals to those from the largest sample size, one could conveniently use the 521-sized sample for optimization, saving a significant amount of computational resources without loss of accuracy. Figure 4.33 also presents the accuracy threshold of RД from which these and lower values would have large errors. Table 4.10 shows the total computational time taken for the full process of data generation, regression and optimization for each size. Each optimal value of Pressure, Adsorption and pressurization times were fed into the detailed model to check if the Pareto curves are reasonably close, and Figure 4.33 shows that they are similar.

Table 4.9 ¯ Ridge coefficients for the tested data set sizes. Numbers in parenthesis `()_ indicate the sample size to which the parameter belongs.

	Purity	Recovery
[illegible]	0.936	0.439
[illegible]	0.0879	-0.464
[illegible]	-0.0892	0.658
[illegible]	0.0501	0.0898
[illegible]	-0.0357	0.149
[illegible]	0.000778	0.0478
[illegible]	-0.00394	-0.0446
[illegible]	0.0276	-0.314
[illegible]	0.0228	0.0329
[illegible]	-0.0322	-0.0592
[illegible]×[illegible]	0.934	0.439
[illegible]×[illegible]	0.106	-0.572
[illegible]×[illegible]	-0.105	0.687
[illegible]×[illegible]	0.0668	0.0863
[illegible]×[illegible]	-0.0508	0.233
[illegible]×[illegible]	0.00606	0.0145
[illegible]×[illegible]	-0.0126	-0.0313
[illegible]×[illegible]	0.0313	-0.319
[illegible]×[illegible]	0.0309	-0.0172
[illegible]×[illegible]	-0.0477	-0.00764
[illegible]	0.944	0.410

伍□□□□	0.110	-0.591
伍□□□□	-0.0993	0.640
伍□□□□	0.0496	0.0687
伍□□□□	-0.0589	0.282
伍□□□□	0.0182	-0.0569
伍□□□□	-0.0165	-0.0231
伍□□□□	0.0230	-0.247
伍□□□□	0.0357	-0.00939
伍□□□□	-0.0343	-0.0316

Figure 4.33 – Pareto optimization sets yielding a pair of purity/recovery for the selected sample sizes with their respective goodness of fit, R^2, and the accuracy threshold of R^2 from which the error would be too large for a model.



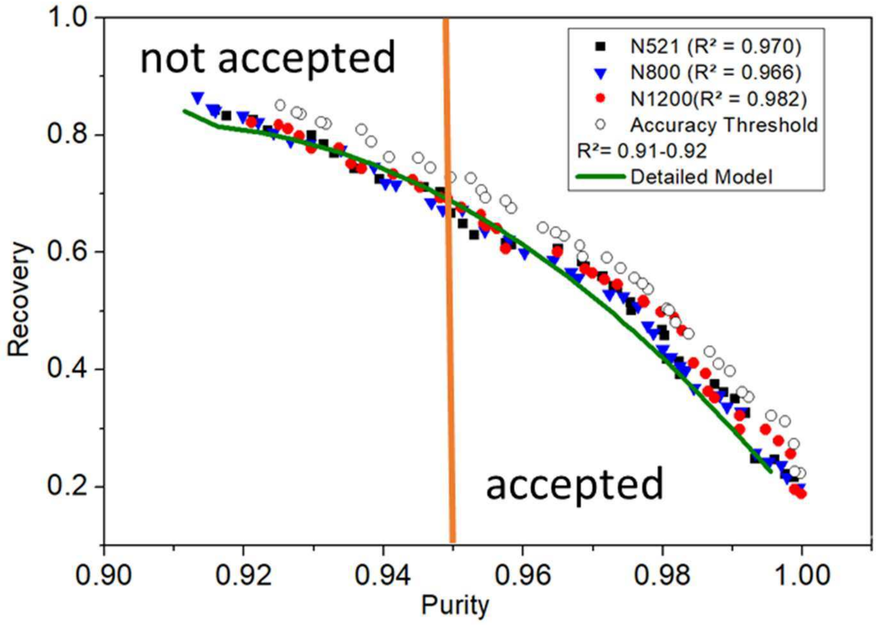
Source: Author

Table 4.10 – Breakdown of computational times.

Sample size	Data generation time	Machine learning time	Optimization time	Total time employed	Time/Longest time
521	101.30 h	0.74 h	0.20 s	102.04 h	0.44
800	155.55 h	0.74 h	0.20 s	156.29 h	0.67
1200	233.33 h	0.88 h	0.20 s	234.22 h	1.00

The pareto front of Figure 4.33, shows all optimal pairs within the tested ranges of [illegible], but not all of these values are of practical interest, literature typically reports a minimum of 95% for practical applications (Ivanova and Lewis, 2012; Karimi and Fatemi, 2021), thus Figure 4.33 can be segmented into two regions, one of accepted optimal results and one of not accepted, Figure 4.33. Looking at the accepted results, Table 4.11 shows the recovery values for each acceptable purity range from 95% to 99% and alongside it, the values of [illegible] given by the optimizer for the sample size of 1200.

Figure 4.34 – Separating the general optimized results into acceptable and not acceptable according to practical requirements.



Source: Author

Table 4.11 – Purity thresholds with associated Recovery and operational parameters for 1200 sample size.

Purity	Recovery	[illegible]	[illegible]	[illegible]
95 %	67.62%	112.68 s	47.48 s	11.65 bar
96%	60%	109.32 s	82.28 s	13.95 bar
97%	55.42%	43.03 s	84.06 s	13.99 bar
98%	48.85%	100.38 s	51.64 s	15.08 bar
99%	32.15%	67.67 s	89.36 s	13.76 bar

Note that a pair of purity and recovery is not necessarily tied to the same triad of [illegible] the algorithm can find two points of similar Purity and Recovery with very

different triad of parameters, to show how this is possible, Table 4.12 shows all the Pareto Front points with their respective parameters, Notice that point 97.73% of Purity has 22.50 s, 85.91s and 14.34 bar as, respectively, Pressurization time, adsorption time, and Outlet pressure, but point nearby 97.98% has 35.82 s, 95.16 s, and 16.20 bar. The algorithm can find different combinations of the operational parameters for a single pair Purity-Recovery if it is asked to generate a finer front, with many points close to one another.

Table 4.12 ¯ Pareto front pairs with associated operational parameters for 1200 sample size

Purity (%)	Recovery (%)	夾僦 (s)	夾僳聚 (s)	俱儅 (bar)
92.12%	82.23%	20.68	98.50	19.44
99.99%	18.80%	88.96	20.63	13.95
98.45%	41.14%	112.69	43.41	11.56
95.76%	60.65%	41.44	93.45	17.57
99.10%	32.15%	67.68	89.37	13.77
97.36%	54.59%	88.54	27.52	14.38
99.89%	19.59%	51.20	84.48	18.28
94.14%	73.38%	43.77	84.22	16.95
98.75%	35.22%	55.98	87.08	15.37
98.28%	46.61%	72.73	89.37	15.84
99.67%	27.87%	110.67	48.95	14.00
99.84%	25.69%	91.22	82.94	14.95
92.97%	77.80%	117.82	79.91	14.37
96.89%	57.20%	110.67	84.48	13.06
95.63%	64.03%	30.79	87.16	17.64
95.11%	67.63%	112.69	47.48	11.65
93.69%	74.26%	25.66	84.78	14.35
94.53%	71.11%	21.15	85.96	19.13
97.98%	49.79%	35.82	95.16	16.20
97.73%	51.81%	22.51	85.92	14.35
92.80%	79.83%	103.28	66.66	13.96
93.54%	75.09%	29.01	84.73	19.44
94.43%	72.42%	89.56	85.19	15.89
95.40%	66.45%	88.96	22.63	13.82
92.50%	81.81%	101.29	63.57	13.82
94.82%	69.34%	76.31	84.61	13.77
97.17%	55.42%	43.04	84.06	14.00
98.17%	48.85%	100.38	51.65	15.08
92.63%	81.14%	112.69	44.64	14.09
95.44%	64.97%	98.60	54.66	13.95
94.92%	69.22%	55.98	87.08	13.01
97.73%	51.49%	21.06	84.59	19.44
96.99%	56.41%	88.56	26.33	14.38
99.11%	29.80%	81.71	85.82	17.57
98.62%	39.38%	35.82	95.73	16.16
98.65%	36.31%	21.45	87.07	19.44
99.47%	29.78%	91.22	27.22	15.01

93.36%	77.77%	100.60	57.73	13.95
96.50%	60.09%	109.33	82.28	13.96
95.47%	64.49%	101.66	52.95	14.00

However, it is still possible to narrow down the search further by introducing another performance variable: Productivity. Its desirable not only to have substantial values of purity and recovery, but to also achieve the highest productivity possible given those values. Equation 4.8 shows how the productivity can be calculated given values of purity and recovery and Equations 4.9 and 4.10 shows how to calculate the inlet flowrate [illegible] given [illegible] and [illegible] these two equations were obtained via regression after observing the behavior of [illegible] given different values of [illegible] and [illegible]

[illegible] (4.8)

[illegible] (4.9)

[illegible] (4.10)

With these new formulas, we can explore points with similar purity and recovery and observe if there are significant differences in productivity, which there are, as Table 4.13 shows. These findings allow for a rewrite of Table 4.11 as 4.14 filtering for the highest productivity possible and table 4.12 as 4.15 for a full picture of all relevant performance variables.

Table 4.13 – Similar pairs of purity and recovery with very distinct productivity values

Purity(%)	Recovery(%)	Productivity (mol h^{-1} kg^{-1})	[illegible](s)	[illegible](s)	[illegible] (bar)
97.73%	51.49%	145.76	21.06	84.59	19.44
97.73%	51.81%	105.60	22.51	85.92	14.35
97.98%	49.79%	75.83	35.82	95.16	16.20

Table 4.14 – Purity thresholds with associated Recovery and operational parameters with highest productivity

Purity(%)	Recovery(%)	Productivity (mol h^{-1} kg^{-1})	[illegible](s)	[illegible](s)	[illegible] (bar)

95%	69.34%	58.89	55.98	87.08	13.01
96%	60.65%	87.55	41.44	93.45	17.57
97%	55.42%	63.18	43.04	84.06	14.00
98%	51.49%	145.76	21.06	84.59	19.44
99%	35.22%	34.82	55.98	87.08	15.37

Table 4.15 ¯ Pareto front pairs with associated operational parameters and productivity for 1200 sample size

Purity (%)	Recovery (%)	Productivity (mol h^{-1} kg^{-1})	亵偢熗 (s)	亵僺鼎 (s)	俱儅齾(bar)
92.12%	82.23%	236.49	20.68	98.50	19.44
99.99%	18.80%	11.43	88.96	20.63	13.95
98.45%	41.14%	17.18	112.69	43.41	11.56
95.76%	60.65%	87.56	41.44	93.45	17.57
99.10%	32.15%	24.44	67.68	89.37	13.77
97.36%	54.59%	34.24	88.54	27.52	14.38
99.89%	19.59%	24.41	51.20	84.48	18.28
94.14%	73.38%	97.92	43.77	84.22	16.95
98.75%	35.22%	34.83	55.98	87.08	15.37
98.28%	46.61%	37.79	72.73	89.37	15.84
99.67%	27.87%	14.08	110.67	48.95	14.00
99.84%	25.69%	16.27	91.22	82.94	14.95
92.97%	77.80%	38.12	117.82	79.91	14.37
96.89%	57.20%	27.13	110.67	84.48	13.06
95.63%	64.03%	119.85	30.79	87.16	17.64
95.11%	67.63%	28.45	112.69	47.48	11.65
93.69%	74.26%	135.18	25.66	84.78	14.35
94.53%	71.11%	197.70	21.15	85.96	19.13
97.98%	49.79%	75.84	35.82	95.16	16.20
97.73%	51.81%	105.61	22.51	85.92	14.35
92.80%	79.83%	42.68	103.28	66.66	13.96
93.54%	75.09%	161.31	29.01	84.73	19.44
94.43%	72.42%	49.18	89.56	85.19	15.89
95.40%	66.45%	40.04	88.96	22.63	13.82
92.50%	81.81%	44.09	101.29	63.57	13.82
94.82%	69.34%	47.54	76.31	84.61	13.77
97.17%	55.42%	63.19	43.04	84.06	14.00
98.17%	48.85%	28.70	100.38	51.65	15.08
92.63%	81.14%	40.59	112.69	44.64	14.09
95.44%	64.97%	36.14	98.60	54.66	13.95
94.92%	69.22%	58.89	55.98	87.08	13.01
97.73%	51.49%	145.76	21.06	84.59	19.44
96.99%	56.41%	35.38	88.56	26.33	14.38
99.11%	29.80%	23.96	81.71	85.82	17.57
98.62%	39.38%	59.83	35.82	95.73	16.16
98.65%	36.31%	101.18	21.45	87.07	19.44
99.47%	29.78%	18.92	91.22	27.22	15.01

93.36%	77.77%	42.51	100.60	57.73	13.95
96.50%	60.09%	30.59	109.33	82.28	13.96
95.47%	64.49%	35.04	101.66	52.95	14.00

5 CONCLUSION AND FUTURE WORKS

Several machine-learning models, even the simplest ones, are suitable for surrogate modeling and optimization of adsorption processes. Overall, the use of the ANN model of multilayer perceptrons was unstable, varying from very low to very high qualities of fit, and unexpected results which led to the choice of the alternative models, even though they are less complex. However, such instability was present mostly on the unscaled scenario in lower sample sizes, which is likely due to the behavior of the adam solver, as the hyperparameter analysis shows, thus the lower performance of the ANN˘s is not to be taken as definitive. Instead, it shows that the use of the gradient descent solvers introduce too many instabilities and Newton-based algorithms should be preferred if the computational cost is feasible. However, even with the solver correction, despite significant increase in the ANN performance, simpler models still manage to offer very close quality of fitting, which offers an advantage on its use due to lower computational cost and easier model interpretation.

The introduction of a physically meaningful constraint based on the breakthrough stoichiometric time was successful in both increasing the model accuracy for all tested sample sizes and reducing the computational time employed of the full modelling plus optimization process by more than 50%. These two findings shift the weight of the evaluated surrogate models with machine learning from finding better statistical models to introducing small bits of process information, as even the most primitive models are capable of delivering reasonable accuracies.

However, these improvements are not guaranteed since the Machine learning models are black box in nature, so their behavior is not necessarily in tune with physical consistency, it has been shown that in this case, the introduction of the process knowledge helped improve accuracies, not on all cases. It˘s then concluded that its worth it to try and use process knowledge to check if the model becomes more accurate, before trying to get more data points.

The use of a simple linear-type model with polynomial scaling was successful in obtaining the Pareto set in all sample-sizes, increasing the reliability of the previous discovered

information. The implementation of the machine learning modelling and the optimization in the open source Python language was also successful and relatively straightforward leading to easier replication and further findings in the field with a widely accessible mean. The optimized values do not generate unique sets of 倀 for very similar values of Purity and recovery, which cause ambiguity on which set to choose, this ambiguity can be overcome by expanding onto the Pareto results filtering for acceptable purity ranges and focusing on maximizing productivity to narrow down the values to obtain the values of the operational parameters Pressurization time, Adsorption time and outlet Pressure.

As possible extensions and future works on the topic is to extend the modelling and optimization for variable adsorbent solids and including the isotherm and diffusion parameters in the model, as to try and find the `best adsorbent_ for a particular set of conditions with its equilibrium and isotherm parameters would be selected to be the closest of a list of existing material creating a model capable of material screening.

REFERENCES

AGARWAL A.; BIEGLER, L. T.; ZITNEY, S. E. Simulation and Optimization of Pressure Swing Adsorption Systems Using Reduced-Order Modeling. Ind. Eng. Chem. Res. v. 48, n. 5, p. 2327¯2343, 2008.

ALIMADADI, A.; ARYAL, S.; MANANDHAR, I.; MUNROE, P. B.; JOE, B.; CHENG, X. Artificial Intelligence and Machine Learning to Fight COVID-19. Physiological Genomics. v. 52, n. 4, p. 200-202, 2020.

ANDERSON, J. D. Computational Fluid Dynamics. 1. ed. New York: McGraw-Hill Science/Engineering/Math, 1995. 574 p.

ATHEY, S.; IMBENS, G. Machine learning methods economists should know about. Annu. Rev. Econ. v. 11, p. 685-725, 2019.

BALASHANKAR, V. S.; RAJAGOPALAN, A. K.; DE PAUW, R.; AVILA, A. M.; RAJENDRAN, A. Analysis of a Batch Adsorber Analogue for Rapid Screening of Adsorbents for Postcombustion CO2 Capture. Ind. Eng. Chem. Res. v. 58, n. 8, p. 3314¯3328, 2019.

BARG, C. Simula´²o e Otimiza´²o de Unidades Industriais PSA para Purifica´²o de Hidrog¸nio. Dissertation (M.Sc in Chemical Engineering) ¯ Universidade Federal do Rio Grande do Sul, Porto Alegre, 2000.

BECK, D. A. C.; CAROTHERS, J. M.; SUBRAMANIAN, V. R.; PFAENDTNER, J. Data science: Accelerating innovation and discovery in chemical engineering. AIChE Journal. v. 62, n. 5, p. 1402¯1416, 2016.

BECK, J.; FRIEDRICH, D.; BRANDANI, S.; FRAGA, E. S. Multi-objective optimisation using surrogate models for the design of VPSA systems. Computers & Chemical Engineering. v. 82, p. 318¯329, 2015.

BELLMAN, R. E. Dynamic Programming. 3. ed. New Jersey: Princeton University Press, 2010. 392 p.

BHATIA, S. K.; JEPPS, O.; NICHOLSON, D. Tractable molecular theory of transport of Lennard-Jones fluids in nanopores. The Journal of Chemical Physics. v. 120. n. 9. p. 4472¯4485.

BIRD, R. B.; STEWART, W. A.; LIGHTFOOT, E. N. Transport Phenomena. 3. ed. New York: Wiley, 2007. 928 p.

BOBBITT, N. S.; SNURR, R. Q. Molecular modelling and machine learning for high-throughput screening of metal-organic frameworks for hydrogen storage. Molecular Simulation. v. 45, p. 1–13, 2019.

BOUKOUVALA, F.; HASAN, M. M. F.; FLOUDAS, C. A. Global optimization of general constrained grey-box models: new method and its application to constrained PDEs for pressure swing adsorption. J. Glob. Optim. v. 67, p. 3–42, 2017.

BRUANUER, S.; EMMETT, P. H.; TELLER, E. Adsorption of gases in multimolecular layers, J. Am. Chem. Soc. v. 60, p. 309–316, 1938.

BURNS, T.; PAI, K. N.; SUBRAVETI, S. G.; COLLINS, S.; KRYKUNOV, M.; RAJENDRAN, A.; WOO, T. K. Prediction of MOF performance in Vacuum-Swing Adsorption systems for post-combustion CO_2 capture based on integrated molecular simulation, process optimizations, and machine learning models. Environmental Science & Technology. v. 54, n. 7, p. 4536-4544 2020.

DARKEN L. S. Diffusion, mobility and their interrelation through free energy in binary metallic systems. Trans. AIME. v. 175, p. 184-201, 1948.

DO, D. D. Adsorption Analysis: Equilibria and Kinetics, Chemical Engineer Series. 1. ed. London: Imperial College Press, 1998. 892 p.

EINSTEIN, A. Zur Theorie der Brownschen Bewegung. Ann. Phys. v. 19, n. 371, 1906.

FERNANDEZ-DELGADO, M.; CERNADAS, E.; BARRO, S.; AMORIM, D. Do we Need Hundreds of Classifiers to Solve Real World Classification Problems? JMRL. v. 15, n. 90, p. 3133–3181, 2015.

FERREIRA, A. F. P.; RIBEIRO, A. M.; KULAC, S.; RODRIGUES, A. E. Methane purification by adsorptive processes on MIL-53(Al). Chem. Eng. Sci. v. 124, p. 79–95, 2015.

FOO, K. Y.; HAMEED, B. H. Insights into the modeling of adsorption isotherm systems. Chemical Engineering Journal. v. 156, n. 1, p. 2–10, 2010.

FORRESTER, A. I. J.; SOBESTER, A.; KEANE, A. J. Engineering Design via Surrogate Modelling. 1. ed. New York: John Wiley & Sons, 2008. 210 p.

FREUNDLICH, H. M. F. Over the adsorption in solution, J. Phys. Chem. v. 57, p. 385–471, 1906.

GÉRON, A. Hands–On Machine Learning with Scikit–Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2. ed. Boston: O'Reilly media, 2019. 600 p.

HASAN, M. M. F.; KARIMI, I. A.; FAROOQ, S.; RAJENDRAN, A.; AMANULLAH, M. Surrogate-based VSA Process Optimization for Post-Combustion CO_2 Capture. 21st European Symposium on Computer Aided Process Engineering. p. 402¯406, 2011.

HASTLE, T.; TIBSHIRANE, R.; FRIEDMAN, J. The elements of statistical learning: Data mining, inference and prediction. 2. ed. New York: Springer, 2009. 745 p.

HAYNES, H. W. The Experimental Evaluation of Catalyst Effective Diffusivity. Catalysis Reviews. v .30, n. 4, p. 563¯627, 1988.

HOSSAIN, M. I.; CUNNINGHAM, J. D.; BECKER, T. M.; GRABICKA, B. E.; WALTON, K. S.; RABIDEAU, B. D.; GRANT GLOVER, T. Impact of MOF Defects on the Binary Adsorption of CO_2 and Water in UiO-66. Chemical Engineering Science. v. 203, p. 346-357, 2019.

IVANOVA, S; LEWIS, R. Producing Nitrogen via Pressure Swing Adsorption. AICHE CEP June, p. 38¯42, 2012.

JAMSHIDIFARD, S.; HOSSEINI, S.; REZAEI, S.; KARAMIPOUR, A.; JAFARI RAD, A.; IRANI, M. Incorporation of UiO-66-NH_2 MOF into the PAN/chitosan nanofibers for adsorption and membrane filtration of Pb(II), Cd(II) and Cr(VI) ions from aqueous solutions. Journal of Hazardous Materials v. 368, p. 10-20, 2019.

JIANG, L. A.; BIEGLER, L. T.; FOX, V. G. Design and Optimization of Pressure Swing Adsorption Systems with Parallel Implementation. 8th International Symposium on Process Systems Engineering. p. 232¯237, 2003.

JOHNSON, J. A.; OROSKAR, A. R. Sorbex Technology for Industrial Scale Separation. Studies in Surface Science and Catalysis. v. 46, p. 451¯467, 1989.

K RGER, J.; RUTHVEN, D. M., THEODOROU, D. N. Diffusion in nanoporous materials. 1. ed. Germany: Wiley-VCH, 2011. 902 p.

KALYANMOY, D.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions On Evolutionary Computation. v. 6, n. 2, p. 106210, 2021.

KARIMI, K.; FATEMI, S. Methane capture and nitrogen purification from a nitrogen rich reservoir by pressure swing adsorption; experimental and simulation study. Journal of Environmental Chemical Engineering. v. 9, n. 5, p. 182-197, 2002.

KNOX, J. C.; EBNER, A. D.; LEVAN, M. D.; COKER, R. F.; RITTER, J. A Limitations of breakthrough curve analysis in fixed-bed adsorption. Ind. Eng. Chem. Res. v. 55, p. 4734¯4748, 2016.

LAERI, F.; SCHEUTH, F.; SIMON, U.; WARK, M. Host-Guest Systems Based on Nanoporous Crystals. 1. ed. Weinheim: Wiley-VCHVerlag GmbH, 2003. 662 p.

LANGMUIR, I. The constitution and fundamental properties of solids and liquids, J. Am. Chem. Soc. v. 38, n. 11, p. 2221¯2295, 1916.

LEPERI, K. T.; YANCY-CABALLERO, D.; SNURR, R. Q.; YOU, F. 110th Anniversary: Surrogate Models Based on Artificial Neural Networks to Simulate and Optimize Pressure Swing Adsorption Cycles for CO_2 capture. Industrial & Engineering Chemistry Research. v. 58, n. 39, p. 18241-18252, 2019.

LIMOUSIN, G.; GAUDET, J. P.; CHARLET, L.; SZENKNECT, S.; BARTHΠS, V.; KRIMISSA, M. Sorption isotherms: A review on physical bases, modeling and measurement. Applied Geochemistry. v. 22, n. 2, p. 249¯275. 2007.

LUBERTI, M.; KIM, Y. H-.; LEE, C. ¯H.; FERRARI, M. ¯C.; AHN, H. New momentum and energy balance equations considering kinetic energy effect for mathematical modelling of a fixed bed adsorption column. Adsorption v. 21, p. 353¯363, 2015.

MA, S.; TONG, L.; YE, F.; XIAO, J.; B§NARD, P.; CHAHINE, R. Hydrogen purification layered bed optimization based on artificial neural network prediction of breakthrough curves. International Journal of Hydrogen Energy. v. 44, n. 11, p. 5324-5333 2019.

MAIA, D. A. S.; DE OLIVEIRA, A. J. C.; NAZZARRO, M. S.; SAPAG K. M.; LΦPEZ R. H.; LUCENA, S. M. P.; DE AZEVEDO, D. C. S. CO_2 gas-adsorption calorimetry applied to the study of chemically activated carbons. Chem. Eng. Res. Des. v. 136, p. 753¯760, 2018.

MARX, D.; JOSS, L.; HEFTI, M.; GAZZANI, M.; MAZZOTTI, M. CO_2 capture from a binary CO_2/N_2 and a ternary $CO_2/N_2/H_2$ mixture by PSA: Experiments and predictions. Ind. Eng. Chem. Res. v. 54, p. 6035¯6045, 2015.

MONDINO, G.; GRANDE, C. A.; BLOM, R.; NORD, L. O. Moving bed temperature swing adsorption for CO_2 capture from a natural gas combined cycle power plant. International Journal of Greenhouse Gas Control. v. 85, p. 58¯70, 2019.

MORALES-OSPINO, R.; GOLTZMAN, Y.; TORRES, A.E.B.; VILARRASA-GARCPA, E.; BASTOS-NETO, M.; CAVALCANTE, C. L. JR.; AZEVEDO, D. C. S.; MARQUES, C. R. M.; AQUINO, T. F.; OLIVEIRA, V. R. Assessment of the potential use of zeolites synthesized from power plant fly ash to capture CO_2 under post-combustion scenario. Adsorption. v. 26, p. 1153-1164, 2020.

MORALES-OSPINO, R.; SANTIAGO, R.G.; SIQUEIRA, R.M.; AZEVEDO, D. C. S.; BASTOS-NETO, M. Assessment of CO_2 desorption from 13X zeolite for a prospective TSA process. Adsorption. v. 26, p. 813¯824, 2020.

M- LLER, A. C.; GUIDO, S. Introduction to machine learning with Python. 1. ed. Boston: O¯Reilly media, 2017. 392 p.

NICHOLSON, D.; PETROPOULOS, J. H. Calculation of the `surface flow_ of a dilute gas in model pores from first principles. Journal of Colloid and Interface Science. v. 106 n. 2, p. 538¯546, 1985.

NILCHAN, S.; PANTELIDES, C. On the Optimisation of Periodic Adsorption Processes. Adsorption. v. 4 n. 2, p. 113¯147, 1985.

PAI, K. N.; PRASAD, V.; RAJENDRAN, A. Experimentally validated machine learning frameworks for accelerated prediction of cyclic steady state and optimization of pressure swing adsorption processes. Separation and Purification Technology, v. 241, 116651, 2020.

PEDREGOSA, F. N.; VAROQUAUX, G.; GRAMFORT, A; MICHEL, V; THIRION, B; GRISEL, O et. al. Scikit-learn: Machine Learning in Python. JMRL, v. 12, n. 85, 2825-2830, 2011.

RIBEIRO, A. M.; GRANDE, C. A.; LOPES, F. V. S.; LOUREIRO, J. M.; RODRIGUES, A. E. A parametric study of layered bed PSA for hydrogen purification. Chem. Eng. Sci. v. 63, p. 5258-5273, 2008.

RICHARD, K. F. S.; TORRES, A. E. B.; MAIA, D. A. S., DE SOUSA, W. A.; CAVALCANTE, C. L.; AZEVEDO, D. C. S.; BASTOS-NETO, M. Assessing mass transfer rates in porous adsorbents using gas adsorption microcalorimetry. Chemical Engineering Science. v. 229, 115983, 2020.

RICHARD, K. F. S.; AZEVEDO, D. C. S.; BASTOS-NETO, M. Investigation and Improvement of Machine Learning Models Applied to the Optimization of Gas Adsorption Processes. Ind. Eng. Chem. Res. v. 62, n. 18, p. 7093-7102, 2023.

RUTHVEN, D. M. Principles of Adsorption and Adsorption processes. 1. ed. New York: John Wiley & Son, 1984. 453 p.

RUTHVEN, D. M.; FAROOQ, S.; KNAEBEL, K. S. Pressure Swing Adsorption. 1. ed. New York: Wiley-VCH, 1994. 376 p.

SΞNCHEZ-ZAMBRANO, K.S.; VILARRASA-GARCPA, E.; MAIA, D.A.S.; BASTOS-NETO, M.; RODRPGUEZ-CASTELLON, E.; AZEVEDO, D. C. S. Adsorption microcalorimetry as a tool in the characterization of amine-grafted mesoporous silicas for CO_2 capture. Adsorption. v. 26, p. 165-175, 2020.

SCIKIT-LEARN Machine learning in Python. Scikit-learn.org, 2023. Available in: < https://scikit-learn.org/stable/index.html>. Acessed in: 6 oct. 2023.

SING, K. S. W.; EVERETT, D. H.; HAUL, R. A. W.; MOSCOU, L.; PIEROTTI, R. A.; ROUQU§ROL, J.; SIEMIENIEWSKA, T. Reporting physisorption data for gas/soils systems with special reference to the determination of surface area and porosity. Commission on Colloid and Surface Chemistry Including Catalysis, Physical Chemistry Division, International Union of Pure and Applied Chemistry (IUPAC). Pure Applied Chemistry, v. 57, n. 4, p. 603-619, 1985.

SIPS, R. Combined form of Langmuir and Freundlich equations, J. Chem. Phys. v. 16, p. 490-495, 1948.

SIQUEIRA, R. M.; RICHARD, K. F. S.; NASCIMENTO, J. F.; MUSSE, A. P. S.; TORRES, A. E. B.; AZEVEDO, D. C. S.; BASTOS-NETO, M. Simulation of CO_2/CH_4 high pressure separation on microporous activated carbon. Chemical Engineering Communications. v. 206, n. 11, p. 1414-1425, 2018b.

SIQUEIRA, R. M.; VILARRASA-GARCIA, E.; TORRES, A. E. B; DE AZEVEDO, D. C. S.; BASTOS-NETO, M. Simple Procedure to Estimate Mass Transfer Coefficients from Uptake Curves on Activated Carbons. Chem. Eng. Technol. v. 41 n. 8, p. 1622¯1630, 2018a.

SIQUEIRA, R. M.; FREITAS, G. R.; PEIXOTO, H. R.; NASCIMENTO, J. F. D.; MUSSE, A. P. S.; TORRES, A. E. B. et al. Carbon dioxide capture by pressure swing adsorption. Energy Procedia. v. 114, p. 2182¯2192, 2017.

SILVA, B.; SOLOMON, I.; RIBEIRO, A. M.; LEE, U. H.; HWANG, Y. K.; CHANG, J. S. et al. H2 purification by pressure swing adsorption using CuBTC. Sep. Purif. Technol. v. 118, p. 744¯756, 2013.

SILVA, J. A. C.; CUNHA, A. F.; SCHUMANN, K.; RODRIGUES, A. E. Binary adsorption of CO_2/CH_4 in binderless beads of 13X zeolite. Microporous Mesoporous Mat. v. 187, p. 100¯107, 2014.

SKARSTRª M, C. W. Pressure Swing Adsorption. ExxonMobil Research and Engineering co. US2944627A, 1960.

SUBRAVETI, S. G.; LI, Z.; PRASAD, V.; RAJENDRAN, A. Machine learning-based multi-objective optimization of pressure-swing adsorption. Industrial & Engineering Chemistry Research. v. 58, n. 44, p. 20412-20422 2019.

THOMMES, M.; KANEKO, K.; NEIMARK, A. V.; OLIVIER, J. P.; REINOSO, F. R.; ROUQUEROL, J.; SING, K. S. W.; TOTH, J. Physisorption of gases, with special reference to the State equations of the solid gas interface layer. Acta Chem. Acad. Hung. v. 69, p. 311¯317, 1971.

YANG, Y.; NARAYANAN NAIR, A. K.; SUN, S. Adsorption and diffusion of methane and carbon dioxide in amorphous regions of cross-linked polyethylene: a molecular simulation study. Industrial & Engineering Chemistry Research. v. 58, n. 19, p. 8426-8436, 2019.

YE, F.; MA, S.; TONG, L.; XIAO, J.; B§NARD, P.; CHAHINE, R. Artificial neural network based optimization for hydrogen purification performance of pressure swing adsorption. International Journal of Hydrogen Energy. v. 44, n. 11, p. 5334-5344, 2019.

ZAMANIYAN, A.; JODA, F.; BEHROOZSARAND, A.; EBRAHIMI, H. Application of artificial neural networks (ANN) for modeling of industrial hydrogen plant. International Journal of Hydrogen Energy. v. 38, n. 15, p. 6289¯6297, 2013.