Important Notice

UNIVERSITY OF CALGARY

Optimizing Diagnostic Fracture Injection Test (DFIT) interpretation using Machine Learning (ML) methods

by

Lukas Sadownyk

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN GEOSCIENCE

CALGARY, ALBERTA
SEPTEMBER, 2022

Abstract

Diagnostic Fracture Injection Tests (DFIT), are commonly used to derive key parameters for hydraulic fracture design and modeling. Although this process can identify properties needed for well optimization, it is also time intensive, affected by interpretation bias, and incomplete data. In this thesis, I address these adversities by applying unsupervised clustering methods: *K-Means, DB-Scan, Hierarchical modeling, and Gaussian mixture models* to identify point density variation that correlates to key parameters on a DFIT pressure decline. Deep Neural networks (DNN) trained using labeled DFITs are further tested for event prediction. To test these methods a variety of platforms are tested such as *R-Studio Shiny Web App®* to create user-friendly testing platforms and *Python®* for its computational ability when faced with supervised learning methods. Collectively unsupervised and supervised learning methods show significant promise in the DFIT interpretation realm.

Acknowledgements

I would first like to thank the directors and sponsors of CREWES project, who have funded me during my master's research. Among those, I sincerely thank my supervisor Dr. Kristopher Innanen, who has given me guidance not only on my research but also on my career. Among all CREWES fellows, I would also like to express my special gratitude to Marcelo Guarido who has helped be build and develop skills required to be a true data scientist. The TOC (Tight Oil Consortium) group has been a essential contributor to the dataset for this work and I also thank them for their critiques and suggested improvements. Among the TOC fellows I would like to extend a special gratitude to Danial Zeinabadybejestani for creating testing DFIT synthetics as well as meeting weekly to discuss ideas. Lastly, I would like to thank my friends and family for supporting me on this journey, it has been a pleasure.

To the vastness of exploration on this planet.

Table of Contents

List of Figures and Illustrations

List of Tables

List of Symbols, Abbreviations and Nomenclature

Symbol or abbreviation	Definition
DFIT	Diagnostic Fracture Injection Test
Sh_{min}	Minimum horizontal stress
Sh_{max}	Maximum horizontal stress
$ISIP$	Instantaneous Shut In Pressure
Effective ISIP	Effective Instantaneous Shut In Pressure
Literal ISIP	Literal Instantaneous Shut In Pressure
$P_{reservoir}$	Reservoir pressure
P_{Linear}	Linear flow pressure
$P_{Compliance}$	Compliance pressure as part of method to estimate Sh_{min}
$P_{Tangent}$	Tangent pressure as part of method to estimate Sh_{min}
Analytical value	Values estimated from DFIT curves using current methods from literature
True value	Actual geologic key parameter
ΔP_1	pressure difference one
ΔP_2	pressure difference two
ΔX_1	time function difference one
ΔX_2	time function difference two
t_e	injection time
Δt	shut in time
Pp	pore pressure
nD	Nano Darcy
SRV	Stimulated Reservoir Volume
DAS	Distributed Acoustic Sensing
NFP	Net Fracture Pressure
NHS	Net Horizontal Stress
ML	Machine Learning
DNN	Deep Neural Network
PCA	Principle Component Analysis
G-time	Dimensionless time function

CNN	Convolutional Neural Network
LSTM	Long Short Term Memory
K-means clustering	Centroid based clustering, where 'K' is the number of clusters
DB-scan clustering	Density based clustering method
Hierarchical clustering	Tree based clustering method
Gaussian Mixture model clustering	Bayesian method for cluster designation

Epigraph

It's easy to lie with statistics. It's hard to tell the truth without statistics.

Andrejs Dunkels

Chapter 1

INTRODUCTION

1.1 Motivation

To ensure successful and economic development of low permeability, hydrocarbon bearing organic-rich shales, a fracture stimulation design must be implemented. The process of designing and modelling a fracture program is a computationally intensive and iterative process that requires the estimation of multiple geologic and mechanical properties. These include fracture half-length, Instantaneous Shut-In Pressure ($ISIP$), minimum horizontal stress (Sh_{min}), reservoir pressure ($P_{reservoir}$), fracture extension pressure, reservoir permeability, and fluid content (Clarkson et al., 2012). Although some of these parameters can be directly measured from core samples, many studies have identified the challenges of replicating in-situ conditions to produce accurate results (Clarkson et al., 2012; Venieri et al., 2020). To address this challenge, innovative technologies such as DFITs have been designed to measure key parameters such as $ISIP$, Sh_{min}, and $P_{reservoir}$ in the borehole of a well (Jung et al., 2016). Although the DFIT method is powerful, it is important to discuss its adversities for constant improvement. These include unintended human error and time consumption when analytically interpreting multiple DFIT curves. Further adversities arise due to missing or obscured data trends. Collectively, these examples provide the opportunity to test and evaluate the ability

of Machine Learning (ML) methods to solve these issues. This will be the focus of this thesis.

In the first chapter, I will examine what hydraulic fracturing does for a reservoir and why DFITs are an important measurement tool for optimizing hydraulic fracture networks. This will be followed by current methods of DFIT interpretation, and proposed ML methods for optimizing DFIT interpretation.

1.1.1 Mechanics of hydraulic fracturing

The inherently low average permeability in unconventional reservoirs such as the Duvernay Formation (nanoDarcies (nD)) requires the implementation of processes such as multistage hydraulic fracturing to effectively flow hydrocarbons (Venieri et al., 2020). The primary mechanisms of associated rock failure can fall into the tensile and shear domains depending on the orientation of the stress regime (Tarrahi et al., 2015). To understand *why* failure occurs in the subsurface, stress regimes can be decomposed into principal stresses.

Rocks in the subsurface are affected by three principal stresses, these include the maximum principle stress (σ_1), intermediate principle stress (σ_2), and minimum principle stress (σ_3) (Fossen, 2016). For example, in the Duvernay Formation, σ_1 coincides with maximum horizontal stress (Sh_{max}), σ_2 coincides with the lithostatic stress (σ_v), and σ_3 coincides with the minimum horizontal stress (Sh_{min}) (Shen et al., 2018). Equation 1.1 can be applied to understand the process of hydraulic fracturing. To induce hydraulic fracture, fluid can be pumped into a formation of interest to increase reservoir pore-pressure (Pp). An increase in pore-pressure (Pp) decreases the effective stress (σ_{eff}) applied to a medium while σ_3 (Sh_{min}) remains constant (Equation 1.1) (Tarrahi et al., 2015). Once the Pp overcomes Sh_{min} and surpasses the material specific failure envelope (Mohr-Coulomb, Figure 1.1), the rock fails, creating permeability to enhance production (Siddhamshetty et al., 2019).

$$\sigma_{eff} = \sigma_3 - Pp \tag{1.1}$$

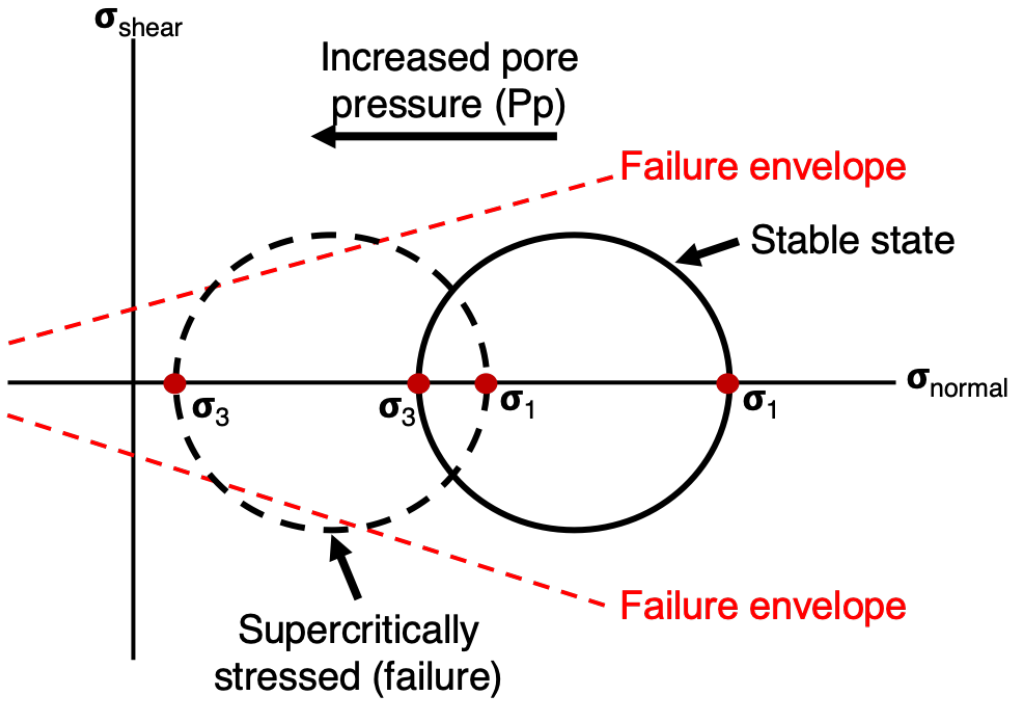


Figure 1.1: Diagram showing the Mohr-Coulomb failure mechanism for a medium that is in stable state within the failure envelope (dashed red line) and the same medium with an increased Pp that is super-critically stressed (passed the failure envelope). At this point the medium would fail to reach a stable stress state.

To effectively develop these fractures within a borehole, the 'plug-and-perforation' technique is an example of one method used to create multiple isolated fracture clusters from toe to heel (Siddhamshetty et al., 2019). This process isolates sections of a wellbore, perforates the casing to create fluid paths into the formation, and injects high pressure proppant (water, silica, chemical additives) to propagate and prevent the closure of a fracture network (Siddhamshetty et al., 2019). The propagation of fracture networks is complex and substantially effected by in-situ stress regimes and natural fracture networks. This creates difficulties when trying to predict production forecasts based on interpreted stimulated reservoir volumes (SRV). In efforts to understand SRV, various studies attempt to predict and model stimulated fracture propagation as well as interaction with natural fractures to optimize economics. Pre-hydraulic stimulation methods include simulations via finite element modeling (Wang and

Chen, 2019) using parameters estimated from DFITs and core. Post hydraulic stimulation fracture analysis can also be carried out using seismic attribute mapping such as ant-tracking or image logs (Farghal and Zoback, 2014), as well as microseismic methods using DAS and geophones (Molenaar et al., 2012; McKean et al., 2019; Liu et al., 2020).

This thesis will focus on pre-stimulation fracture design techniques which are heavily dependent on the estimation of key parameters such as $ISIP$, Sh_{min}, and $P_{reservoir}$ in tandem with having sufficiently long enough recorded pressure decline observations. The following sections will describe how a DFIT can be used to estimate these parameters as well as the meaning of each parameter. Following this, current methods of extracting these parameters from DFITs is discussed as well as the new proposed methods of key parameter extraction using ML methods.

1.2 What is a DFIT?

Diagnostic Fracture Injection Tests (DFITs) can be used as a cost effective experiment to derive key parameters for pre-stimulation fracture design and production forecasting. This method is beneficial because of its relative cost savings and ability to accurately measure parameters in an in-situ environment (Clarkson et al., 2012). Information retrieved from a DFIT can also be used to infer fracture complexity and reservoir behaviour by interpreting and calculating Net Fracture Pressure (NFP) and Net Horizontal Stress (NHS) (Potocki, 2016). These values are obtained by subtracting Sh_{min} from ISIP and Pp from Sh_{min}, respectively. A study by Potocki (2016) found that as NFP increases, the tectonic regime becomes more complex with the introduction of faults and brittle minerals. This results in complex fracture networks and decreased performance of a hydraulic fracture job. It has also been suggested that NHS can be used to determine if fracture propagation is controlled by Pp or tectonic stresses (Potocki, 2016). Sullivan et al. (2019) relates high Post Fracture Pressure Decay (PFPD) rates to high permeability and Young's modulus and efficient cluster activity in

DAS data, demonstrating an effective stimulation result. This correlates to small magnitude values of NFP and NHS from DFIT methods. These studies demonstrate the vast amount of reservoir quality data held within pressure decay curves.

To perform a DFIT, a small volume of fluid is pumped into the target formation to create a hydraulic fracture. By measuring the downhole or surface pressure change over time (pressure-time series) and observing the pressure decline after the hydraulic fracture is created key parameters can be derived. These parameters include: ISIP, Sh_{min}, $P_{reservoir}$, permeability, and decline rates (Clarkson et al., 2012; Mohamed et al., 2020).

1.3 Events observed in a DFIT

Common events of interest that can be extracted from a DFIT pressure decay include: ISIP, Sh_{min}, and $P_{reservoir}$. To understand these events further, Figure 1.2 displays an idealized example of a DFIT with key events overlain.

In Figure 1.2, pressure is increased in a well as fluids are injected into the well. Eventually, the pressure will be great enough to induce a fracture at the break down pressure point. After this point, pressures will decrease as fractures propagate. Upon seeing this event, operators will stop injection, causing an increase in pressure drop rate, this is what is known as the 'literal ISIP' coinciding with 'stop-injection' on Figure 1.2. A second and perhaps more meaningful definition of ISIP is known as the effective ISIP. This is indicated at the location labeled 'ISIP' in Figure 1.2. Mechanically, these alternate ISIP pressures coincide with a difference in what are known as near field and far field pressure effects (McClure et al., 2019). At initial stop injection (literal ISIP), the pressure decline in Figure 1.2 is rapid, this because of the torturous fracture path geometries that exist between the wellbore and reservoir creating a pressure difference. Only sometime after the initial shut-in will you get an accurate pressure for the fracture itself that is opening and propagating. Methods in differentiating these two ISIP pressures will be discussed in the next section.

Following the effective ISIP, a slight deviation in pressure decline after this event corresponds to the fracture closure pressure (Sh_{min}). This point can also be associated with a switch in pressure response from being fracture dominated to reservoir dominated.

Interpretations following Sh_{min} are often heavily reliant on the time length pressure decline recorded. This late time section of decline curve can be split into pseudo-linear and pseudo-radial flow. Initial reservoir conditions will be dominated by pseudo-linear flow from the created fractures. After prolonged time and complete fracture closure flow regime will switch to pseudo-radial flow giving information about reservoir properties such as $P_{reservoir}$ and derived permeability.

With these definitions in mind, we can now analyse a DFIT from the Duvernay Formation (Figure 1.3). Collectively, Figures 1.2 and 1.3 reveal that the interpretation of these key parameters directly from a pressure decline curve is non-trivial; special analytical techniques are required for us to extract these key parameters. These techniques involve manual interpretation of derivative curves such as G-function, Bourdet derivatives, first-order derivatives, and Agarwal time (Zanganeh et al., 2018; Liu and Ehlig-Economides, 2018).

In the following section current analytical methods for interpreting these curves and associated adversities will be dicussed. This will be followed by the proposal of ML methods in efforts to address these adversities.

1.4 Interpreting a DFIT

Multiple analytical methods have been developed based on derivative curves, line fitting methods, and approximations to DFIT induced fracture behaviour to achieve accepted results. This thesis will discuss some of these methods in reference to ISIP, Sh_{min}, and $P_{reservoir}$. In doing so, the associated strengths and weaknesses of each method will be outlined.

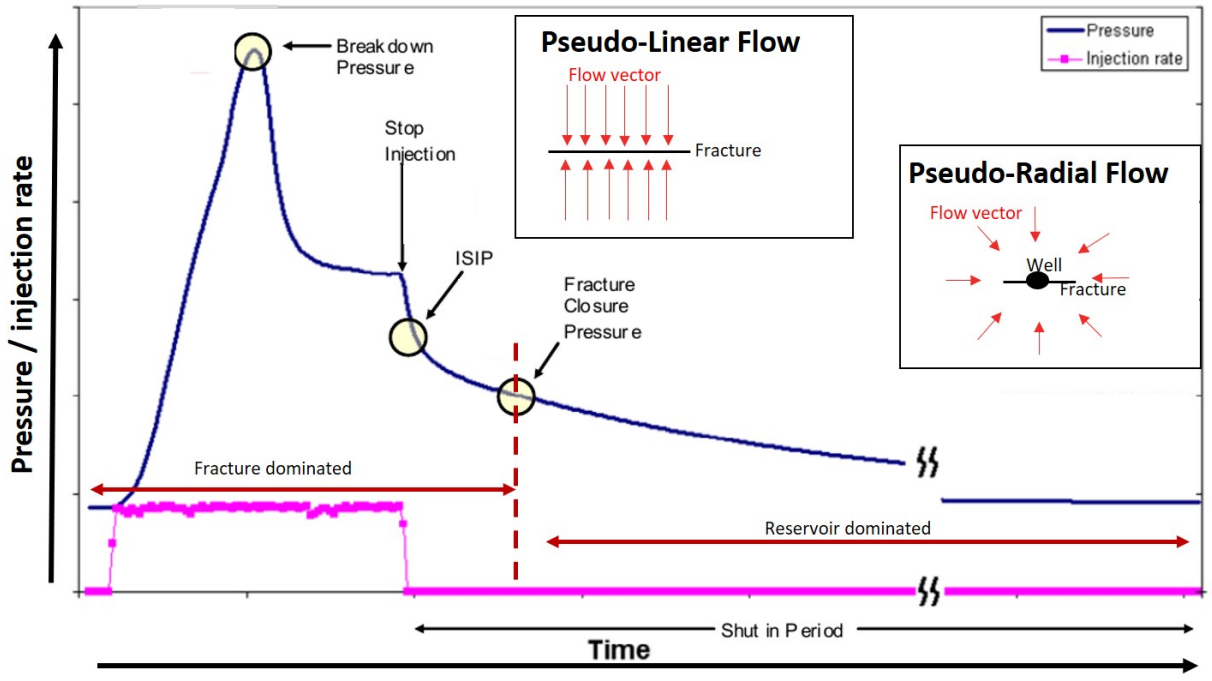


Figure 1.2: An example of a theoretical DFIT adapted from (Cramer and Nguyen, 2013).

1.4.1 Analytical methods

The identification of each key parameter on a pressure decline requires analysis of different derivative curves that can be noisy depending on the field data acquired. To address this, the Bourdet-derivative (Bourdet et al., 1989) can produce smooth derivative values of pressure difference (ΔP) with respect to the defined time function (ΔX). This derivative will be applied later in this thesis to smooth noisy field data for clustering analysis. An example of this calculation is displayed in Equation 1.2. This equation integrates pressure differentials (ΔP_1 and ΔP_2) and their respective time step (ΔX_1 and ΔX_2) to create a smoothing average of the derivative curve (Der_i).

$$Der_i = \frac{(\frac{\Delta P_1}{\Delta X_1}\Delta X_2 + \frac{\Delta P_2}{\Delta X_2}\Delta X_1)}{\Delta X_2 + \Delta X_1} \tag{1.2}$$

Using the Bourdet-derivative method and defined time functions, interpretation procedures can be followed to identify events on a DFIT pressure decline. The respective interpretation

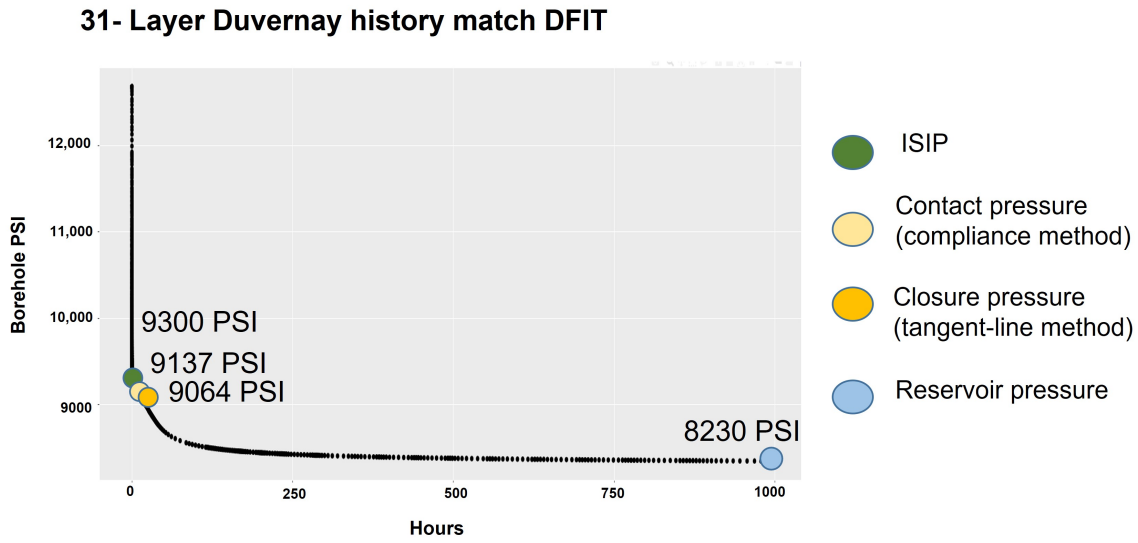


Figure 1.3: ResFrac® history matched pressure falloff data obtained from simulation of a DFIT performed in the 31-layer Duvernay Formation model. Interpreted events along the curve are superimposed and color coded. Approximations of the Sh_{min} are made using both the compliance (McClure et al., 2016) and tangent line method (Barree et al., 2009).

procedures used for DFIT analysis are listed in the following subsections for each event as follows:

ISIP

The identification of ISIP involves plotting pressure decline and derivative pressure decline data with respect to unitless G-time (to be defined below). Using the derived G-function time, fracture fluid leak off after well shut-in is assumed to be linearly proportional to G-time (Nolte, 1988). This can be helpful for identifying processes that cause deviation from the straight line, such as fracture closure. The most common derived form of G-time for fracture leak off is displayed in Equation 1.3 where t_e is the injection time for the DFIT and Δt is the shut-in time (time elapsed since injection ceased). It is important to note that the G-time function is derived assuming Carter leak-off. This means that pressure in a fracture will remain constant over time, which is mechanically not true during a true DFIT experiment. McClure et al. (2019) shows with modelling that these variations have little effect when

achieving accurate key parameter interpretations.

$$Gtime(\Delta t) = \frac{4}{3}[(1 + \frac{\Delta t}{t_e})^{1.5} - (\frac{\Delta t}{t_e})^{1.5}] \tag{1.3}$$

Using McClure et al. (2019) principles of DFIT interpretation, ISIP values can be analytically extracted by plotting pressure and the first order derivative of pressure with respect to G-time (Figure 1.4). This is achieved by finding the first local minimum of the first order derivative of pressure with respect to G-time, interpolating it to the pressure decline line, fitting a tangent line, and extracting the pressure value at G-time equal to zero. This value is known as the Effective ISIP. The Literal ISIP can more simply be extracted at the time injection is ceased, however, this value has little meaning with respect to reservoir properties.

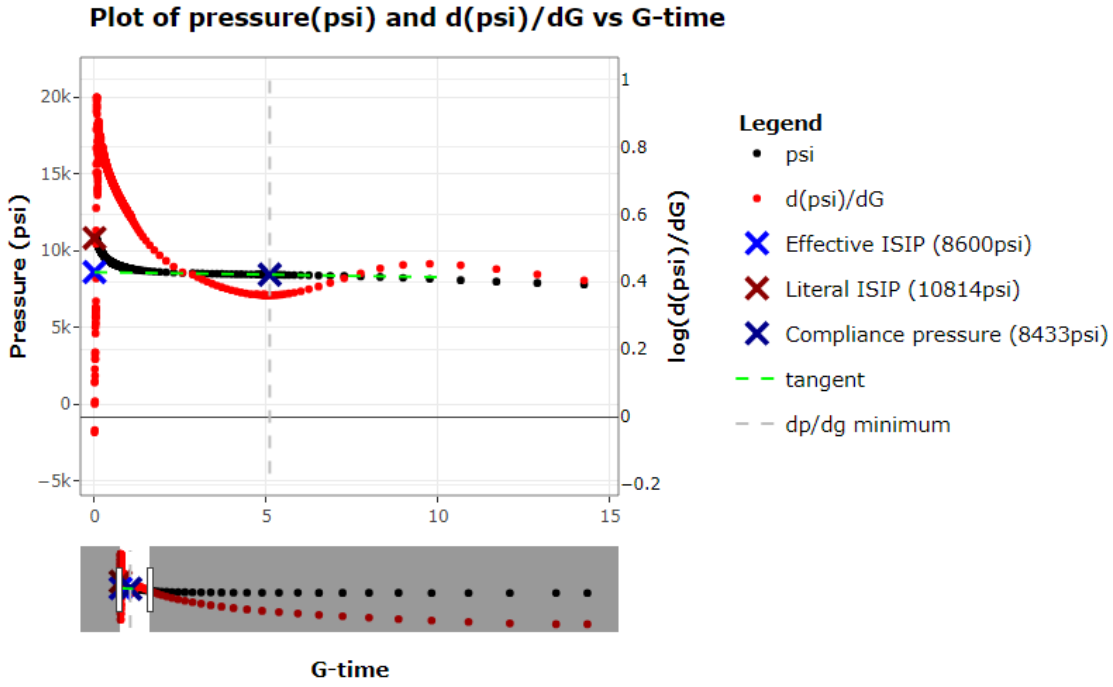


Figure 1.4: Example of ISIP analytical interpretation using pressure and derivative pressure curves with respect to G-time on a Duvernay Formation DFIT synthetic. Contact pressure using the compliance method is also indicated using the Compliance method (McClure et al., 2016). The bottom sub-plot shows the section of curve that has been magnified for this analysis.

Minimum horizontal stress (Sh_{min})

Interpretation of Sh_{min} tends to have two approaches in the industry. The first of which is known as the *tangent line method* (Barree et al., 2009). In this method, a plot of G*dp/dG vs G-time will be constructed and a tangent line will be fit to the linear up track trend on the curve (Figure 1.5). The first point of deviation away from this tangent line indicates an estimate of Sh_{min}. This method has come under scrutiny from articles such as McClure et al. (2019), showing the method does not have any mathematical support and often will underestimate Sh_{min}.

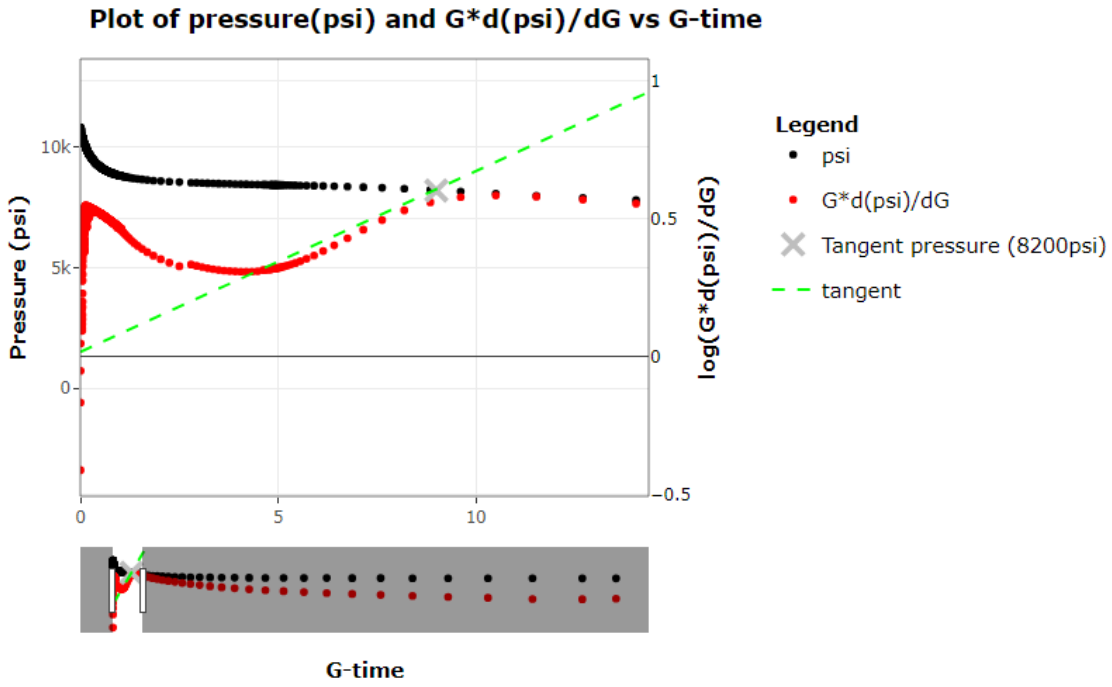


Figure 1.5: Example of Tangent line closure pressure analytical interpretation using pressure and G*derivative pressure curves with respect to G-time on a Duvernay Formation DFIT synthetic. This method is outlined in Barree et al. (2009). The bottom sub-plot shows the section of curve that has been magnified for this analysis.

An alternative method for the determination of Sh_{min} is known as the *compliance method* McClure et al. (2016). This method is based on the principle that as fracture asperities come into contact, the rate of decline on dp/dg vs G-time plots will become more rapid. Based on this observation, the compliance method selects a pressure at the minimum of the dp/dg

or G*dp/dg plots. To get a true Sh_{min} value, 10% will be added to the value followed by a subtraction of 75psi (McClure et al., 2019). The identified contact pressure is displayed in Figure 1.4. Comparing the tangent line and compliance methods from Figures 1.5 and 1.4, respectively, pressures of 8433psi and 8200psi are estimated for closure. Applying the McClure et al. (2019) method of subtracting 75psi from the compliance pressure this gives the accurate Sh_{min} (8358psi) for this synthetic example showing the shortcomings of the tangent method.

Reservoir Pressure ($P_{reservoir}$)

The estimation of $P_{reservoir}$ to derive permeability also proves to be an interpretive effort. In Figure 1.2, it is observed that flow behaviour in post fracture closure can be split into two domains: pseudo-linear flow P_{Linear} and pseudo-radial flow ($P_{reservoir}$). An estimation of $P_{reservoir}$ is the desired result, however, often unattainable due to the recording time of a DFIT. To address this adversity, the implementation of log time versus log(time*dp/t) plots are created to examine late time DFIT behaviour (Figure 1.6). McClure et al. (2019) demonstrates that a -1/2 slope on the log time versus log(time*dp/t) indicates the initiation of linear flow behaviour. This is represented by the dashed green line in Figure 1.6. If this value is extrapolated to the pressure decline curve it shows that linear flow starts at 146 hours and 7099psi. The line fitting process on this curve can be difficult.

Using the interpreted trend of linear flow, a plot of pressure vs the inverse of the square root of time is used to interpolate this trend to zero time. This is demonstrated in Figure 1.7 where the green line interpolates the trend to a pressure of 6982psi.

Other methods of $P_{reservoir}$ interpretation involve plotting the derivative of pressure with respect to Agarwal time, multiplied by Agarwal time vs pressure. Agarwal (1980) defined this method based on observing recovery drawdown after pumps are turned off in a DFIT and can be modelled with Equation 1.4. In this equation, t_p represents the injection time

and Δt is the shut-in time after injection.

$$t_{Agarwal} = \frac{t_p \Delta t}{t_p + \Delta t} \tag{1.4}$$

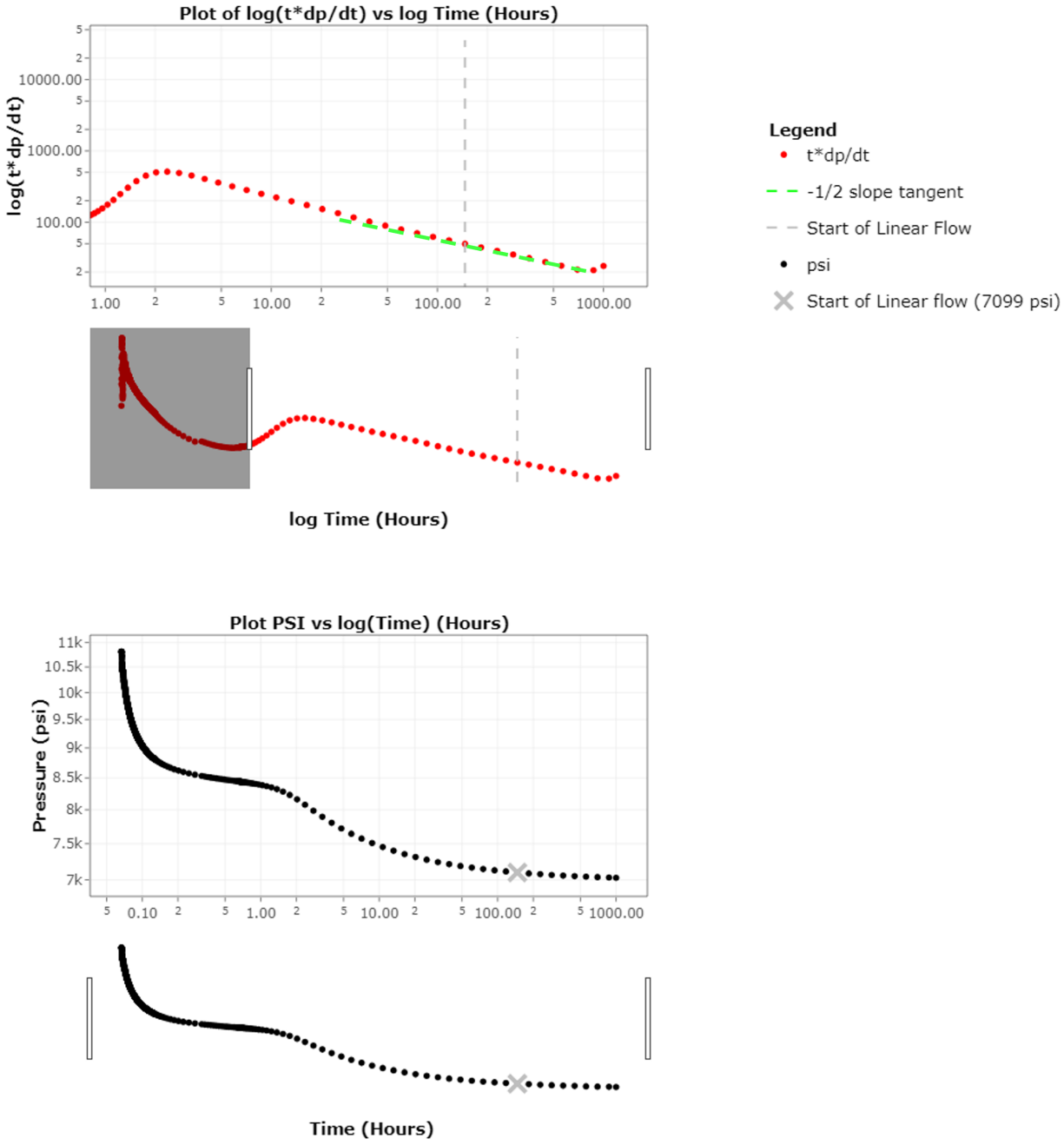


Figure 1.6: Plots of log(t*dp/dt) and psi vs log time (hours) used to derive the start of linear flow.

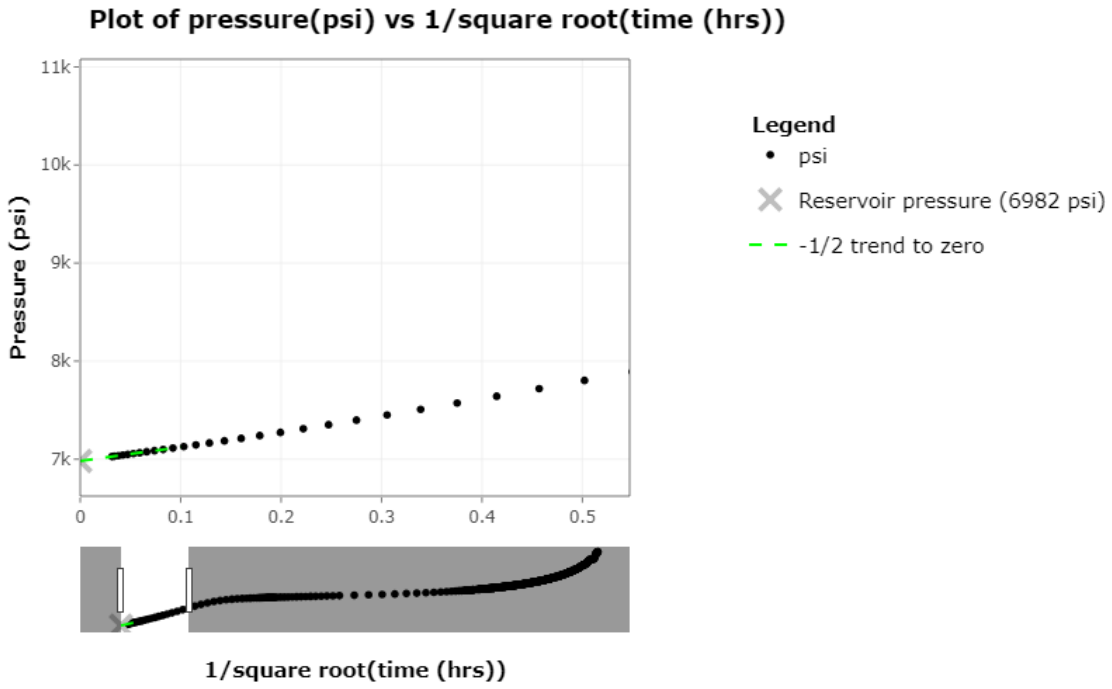


Figure 1.7: Plot of pressure (psi) vs the inverse of the square root of time (hours).

1.4.2 Associated analytical method interpretation bias

As discussed, there are multiple methods for estimating key DFIT parameters. With this in mind, it is important to note that bias can arise from unintentional human error and assumptions made to create derivative curves. Human error may arise when fitting the tangent lines to plots such as those demonstrated in Figures 1.4, 1.5, and 1.6. Moreover, most analytical interpretation is based on simplified mechanical behaviour assumptions for fracture closure in the subsurface (Nolte, 1988; McClure et al., 2019). One of the goals of this thesis is to apply ML methods to avoid these biases and present interpretations that are solely based on the physics of the input parameters from a DFIT. The next section will discuss methods that need to be satisfied to effectively apply ML.

1.5 Intersection of analytical methods, data visualization, ML, and programmable web apps

For large industry datasets, the field of data science has become an increasingly popular area of study via an integration of mathematical, computational, and area expertise (Williams, 2018; Prevos, 2019). Conway's Venn diagram depicted in Prevos (2019) is one way to visualize this multidisciplinary approach (Figure 1.8). From this perspective, a data scientist exploring DFIT data would have background knowledge of the physics affecting fracture closure (*domain knowledge*), while also the understanding of application of regressive, or filtering methods to extract information from the DFIT (*mathematics*) in an efficient matter (*computer science*). Multiple computing languages can be applied in the *computer science* bubble, in this thesis, *R-Studio*® and *Python*® are implemented.

To get a holistic interpretation of the data science problem at hand, it is important to implement each of the qualities displayed in Figure 1.8. Otherwise, only segments to the solution may be attainable. For example, an individual with only domain knowledge and mathematical knowledge may not be able to optimize the problem to commercialize

the result in a computer science setting; this is *research*. Individuals with only computer science and mathematical experience will lack the knowledge to produce results that are truly meaningful to the area of study; this is *machine learning*. Lastly, individuals with only domain and computer science background can create *dangerous* interpretations of the data without knowledge of the assumptions in the mathematics, this is the *danger zone*. It is perhaps the most *dangerous* area as it is easy to do. For example, performing an inversion without knowledge of the type of regularization or prior model weighting in the calculation can produce vastly biased results. A data scientist must work to balance the forces of each of these disciplines.

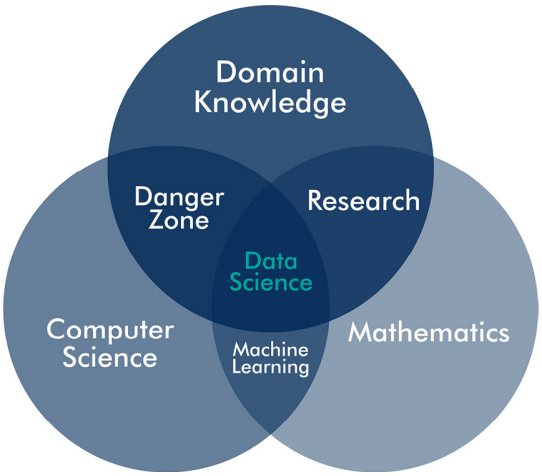


Figure 1.8: Conway's Venn diagram depicting the attributes that make up a data scientist (Prevos, 2019).

Each of these categories in Figure 1.8 are very generalized, and can be expanded further or reorganized. For example, Chollet (2018) focuses on the machine learning aspect, creating subgroups of ML and Deep learning under the artificial intelligence bubble. For this thesis, I have integrated ideas from Williams (2018); Prevos (2019) into a new Venn diagram used to describe the chapters of this thesis (Figure 1.9). This venn diagram contains three categories: *Analytical Interaction*, *Visualization*, and *ML*. These categories will be deconstructed in the next subsections.

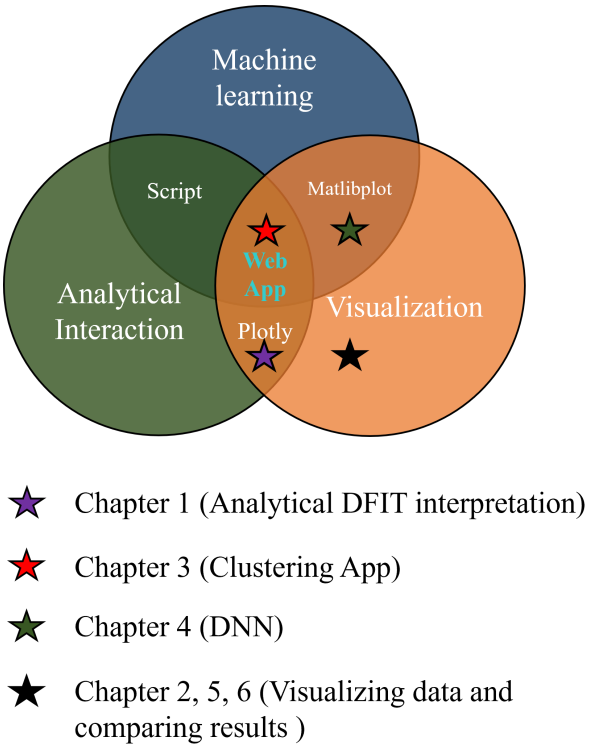


Figure 1.9: Venn diagram of ideas used to analyse DFITs in this thesis. Starred locations indicate techniques brought together for analysis in each chapter.

1.5.1 Analytical Interaction

Analytical interaction is thought of as the process of DFIT interpretation performed in the prior *Analytical methods* section. This part of analysis helps to develop a deeper understanding of the problem and also fits within the domain knowledge bubble in Figure 1.8 from Prevos (2019). In the case of DFIT analytical interpretation, the interactive process allows the user to fit different lines to the DFIT curve and evaluate if the result is geologically meaningful. Furthermore, the analytical DFIT analysis also used the data visualization spectrum to make aesthetically interpretable figures. This was achieved using the *Plotly R graphing library*. Without visualization, an interpreter would simply be looking at the resulting numbers or *script* (Figure 1.9).

While this technique is useful for learning *why* certain interpretations fit the data better, it is also subject to human bias. As demonstrated in the prior analysis of the DFIT curve, the

tangent method can be easily fit to many solutions, creating a degree of uncertainty in our solution. Additionally, depending on the prior background of the interpreter, locations of the drawn tangent line may vary significantly. This effect of human bias can be evaluated further in texts by Dror (2020) and Budowle et al. (2009) where a taxonomy of three categories for human bias is dissected for forensic DNA analysis. Applying this to the geoscience world, Bond et al. (2007) discusses that when dealing with spatially limited and low resolution data, geoscientists must rely on prior experience, this leads to 'conceptual uncertainty'. In this study, 'conceptual uncertainty' lead to only 21% of interpreters identifying the correct structural features in a seismic dataset. Further examples of bias in geoscience is dissected in a publication by Baddeley et al. (2004).

Recent studies have applied unsupervised ML methods in attempt to escape this bias (Ippolito et al., 2021). This reduction of bias using unsupervised methods for DFIT analysis is one of the goals of this thesis.

1.5.2 Visualization

The process of effective data visualization allows the interpreter to discover the trends and their inherit meaning. This was already demonstrated in the *Analytical methods* section where *Plotly R graphing library* was used to interactively plot DFIT derivative curves and color code line trends/ magnify features to identify anomalous features that correlate to key parameters. In this example the combination interactivity and data visualization allows for both the *Analytical Interaction* and *Visualization* spectrum's of Figure 1.9 to be covered. Visualizations can also be applied to the *ML* domain without interactivity using stationary plots such as those produced by *Matlibplot*.

Effectively displaying large datasets is a difficult task. A study by Gorodov and Gubarev (2013) looks further into effective methods of large dataset representation by comparing different features of the data side by side to improve interpretability. Other studies by Froner et al. (2013) show how different data visualizations via color bar manipulation can lead to

varying bias in reservoir area interpretations. Creating multiple visualizations of the data is one way of addressing this bias. Geoffrey (2018) further discusses the effect of cognitive bias affecting user decision making when looking at visualizations and making 'gut feeling decisions'. This thesis relies heavily on visualization techniques to interpret results, therefore, bias must be considered when designing experiments.

1.5.3 Machine Learning (ML)

ML provides a method of determining complex non-linear relationships between variables in large data sets. Although methods and ideas of ML have been developed many years ago in seminal works such as: MacQueen (1967), Ester et al. (1996), Ward (1963), Redner and Walker (1984), only recently, have these ideas gained popularity in the geoscience community. Examples of this new uprise can be found in publications: Wang and Chen (2019); Shen et al. (2020); Saikia et al. (2020); Chen et al. (2021); Ippolito et al. (2021) to list a few. ML can be split up into general categories of supervised and unsupervised, and reinforced learning methods. In the example of supervised learning, the input and target result is known (Bishop and Nasrabadi, 2006). In unsupervised methods an input exists, but, there is no specified target output (Bishop and Nasrabadi, 2006). Reinforced learning will learn preferred weights from various data sets and their desired outcome (Goodfellow et al., 2017). In this thesis, both supervised and unsupervised method will be explored. Combination of this domain with the other domains of Figure 1.9 is discussed in the previous sub-sections.

Unsupervised ML methods can be used in the attempt to escape this bias, however, it can be more time consuming to get to a solution. Ippolito et al. (2021) gives an example of this where well log facies identification is performed using unsupervised learning in conjunction with supervised learning to reduce bias. Supervised learning methods can provide a quick analysis once the weights in the network have been trained (Yang and Ma, 2019) but also suffers from bias due to the selected training set. In a study from Hu et al. (2021) it is demonstrated that CNN trained to predict salt diapir velocity models will incorrectly

categorize all other geologic models as salt. It is important that a wide spread of geologic instances is included are in your training set to avoid this bias.

1.5.4 Merging all domains: Interactive web applications

Interactive web applications are used as a method of merging *Analytical Interaction*, *Visualization*, and *ML* domains in this thesis (Figure 1.9). The *R-Studio Shiny Web App®* platform provides user-friendly coding language to design these apps for interactive ML event identification on DFIT pressure declines.

Little literature exists for the development of *R-Studio Shiny Web App's®* in the geoscience field, Chapter 3 of this thesis will serve to evaluate the benefits of using this method.

1.6 Thesis Objectives

Despite the recent rise in ML applications to large datasets, little literature exists applying these methods for DFIT curve interpretation. Instead, current studies address interpolating missing DFIT pseudoradial flow data using Gradient Boosting (GB) and Random Forest (RF) regression methods (Mohamed et al., 2020) and the integration of real-time well stimulation datasets (injected proppant volumes, downhole pressures, and microseismic events) to identify stimulation program trends using CNN, Autoencoders (AE) and Support Vector Regression (SVR) (Shen et al., 2020; Alatrach et al., 2020; Wang and Chen, 2019; Pandey et al., 2020). This study aims to fill this gap and develop a workflow to identify reservoir parameters $ISIP, Sh_{min}$, and $P_{reservoir}$ from DFITs with the aid of unsupervised clustering algorithms: *K-Means, DB-Scan, Hierarchical modeling, and Gaussian mixture models*, and supervised tree based learning and DNN. The application of these methods intends to speed up interpretation times for datasets consisting of many DFIT curves and to eliminate human bias. Implementation and visualization of these clustering methods are complemented by the development of the *CREWES DFIT Clustering App* using *Shiny Web*

Apps from Rstudio. More computationally heavy DNN based methods are handled with the use of *Google Collab Pro* to exploit the GPUs for computational efficiency. Figure 1.10 displays the development of ideas in this thesis color coded to the method of data science applied.

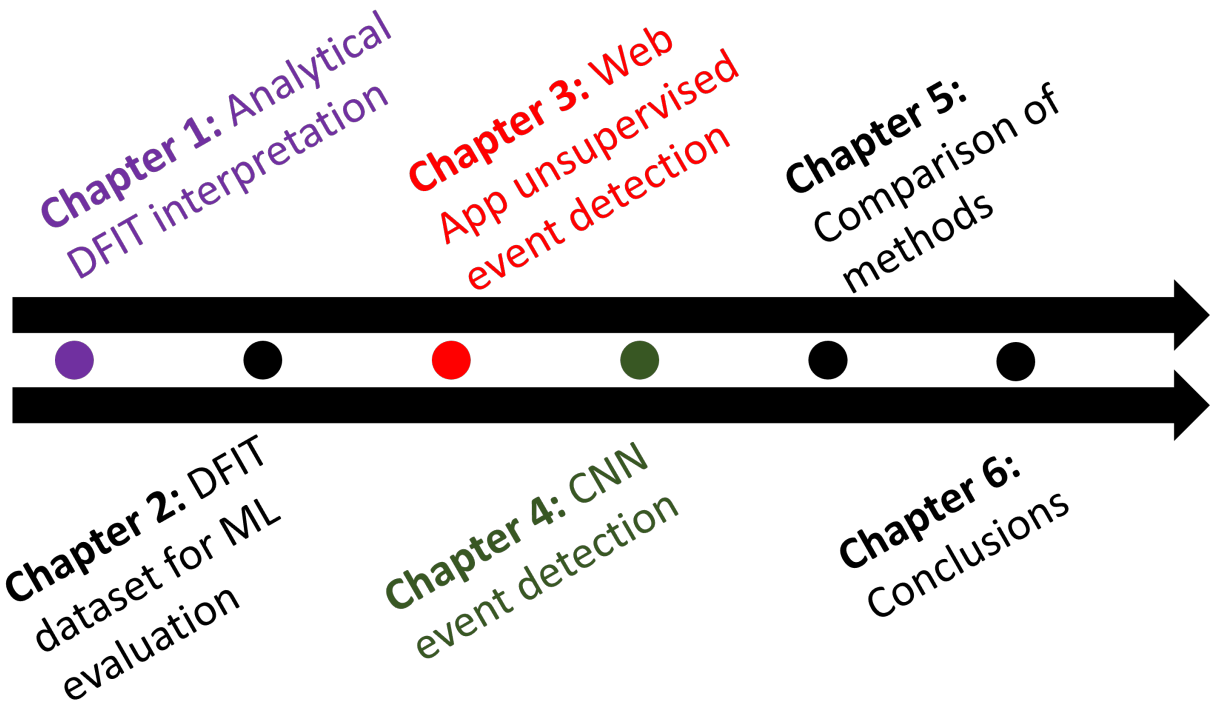


Figure 1.10: Directory of the development of ideas in this thesis, color coded to the location it falls on the data science Venn diagram developed in Figure 1.9.

Chapter 2

DFIT dataset for ML evaluation

2.1 Background

To test the effectiveness of ML methods for DFIT interpretation a combination of synthetic and field DFIT data is evaluated. The use of synthetic data allows for full calibration of the ML methods with known key parameters *ISIP*, *Shmin*, P_{Linear} and $P_{reservoir}$ and later implemented on field data where these parameters are unknown. Generation of synthetic data was achieved using the ResFrac® simulator. ResFrac® is a fully coupled hydraulic fracturing, reservoir, and well bore simulator that models rigorously the key physical processes involved in DFITs. The detail of ResFrac® conceptual model and numerical approach is described in McClure et al. (2021). Ultimately, ResFrac® is capable of generating a synthetic pump-in/shut-in response needed to train ML methods. In total, 27 synthetic examples are used to calibrate and test ML methods in this study.

Parameters that can be manipulated within the ResFrac® simulator are listed in Table 2.1. These parameters feed into the construction of a finite difference model from which pressure decline vs time can be observed. A complete table of all synthetic cases and corresponding ResFrac® input parameters is located in appendix (Figure A.1). When compiling a training dataset, it is important to consider any biases that may be introduced, there must be a

uniform distribution of training possibilities to prevent non-generality (Kim et al., 2019). An example of addressing these biases can be found in Geirhos et al. (2018) where data augmentation is applied as a bias mitigation. In this thesis, each of the parameters in Table 2.1 is varied as a method of addressing this bias and simulating a range of geologic conditions. Figure 2.1 demonstrates a plot of 24 of the synthetics created in this study with varying geologic conditions. The additional curves generated are designed to simulate conditions of the Duvernay Formation, Alberta, Canada.

Geologically, the Duvernay Formation was deposited during a transgressive period of relatively high global sea level, allowing for the creation of restricted low energy off-shore anoxic environments containing black shales that are up to 250m thick (Knapp et al., 2017; Switzer et al., 1994). The low permeability of this formation (nanoDarcies) makes it an optimal target for hydraulic fracture induced production optimization (Dunn et al., 2012). To simulate the performance of tested ML methods on field data, an additional DFIT from the Duvernay Formation is included in the dataset bringing the total number of curves to 28.

Each training and testing dataset is formatted as a .csv file with two columns: Time (hours) and borehole pressure (PSI). Borehole pressure is defined as the pressures measured at the depth of investigation in the well bore. Depending on the desired event classification for the dataset, additional boolean columns of 'True' or 'False' can be added to indicate the location of events on the pressure decline curve. Figure 2.2 demonstrates a plot of all DFIT datasets used in this study. Notably, the highest pressure Duvernay DFIT example appears to be time delayed and contains additional early time information when compared to the other DFITs. This happens to be the field DFIT test, containing information about the pressure ramp up of the formation. This information in the curve does not contain geologic information, instead it is fully dependent on the pumping rate as show in Figure 1.2. For this reason, this data can be eliminated as it can create ML bias and computational burden (Figure 2.3). Collectively, these formatted datasets are used for further augmentation and event classification in this Thesis.

Table 2.1: ResFrac parameters

Resfrac Variable	Definition
WBS	Wellbore storage factor
NWT	Near-wellbore tortuosity
S90	Closure process duration
E0max	Start of closure process
YM	Young's modulus for aperture of fracture
Geometry	Fracture flow style
Resfluid	Reservoir fluid
Sh_{min}	Minimum principle stress
P_{res}	Reservoir Pressure
Layers	Number of geologic layers

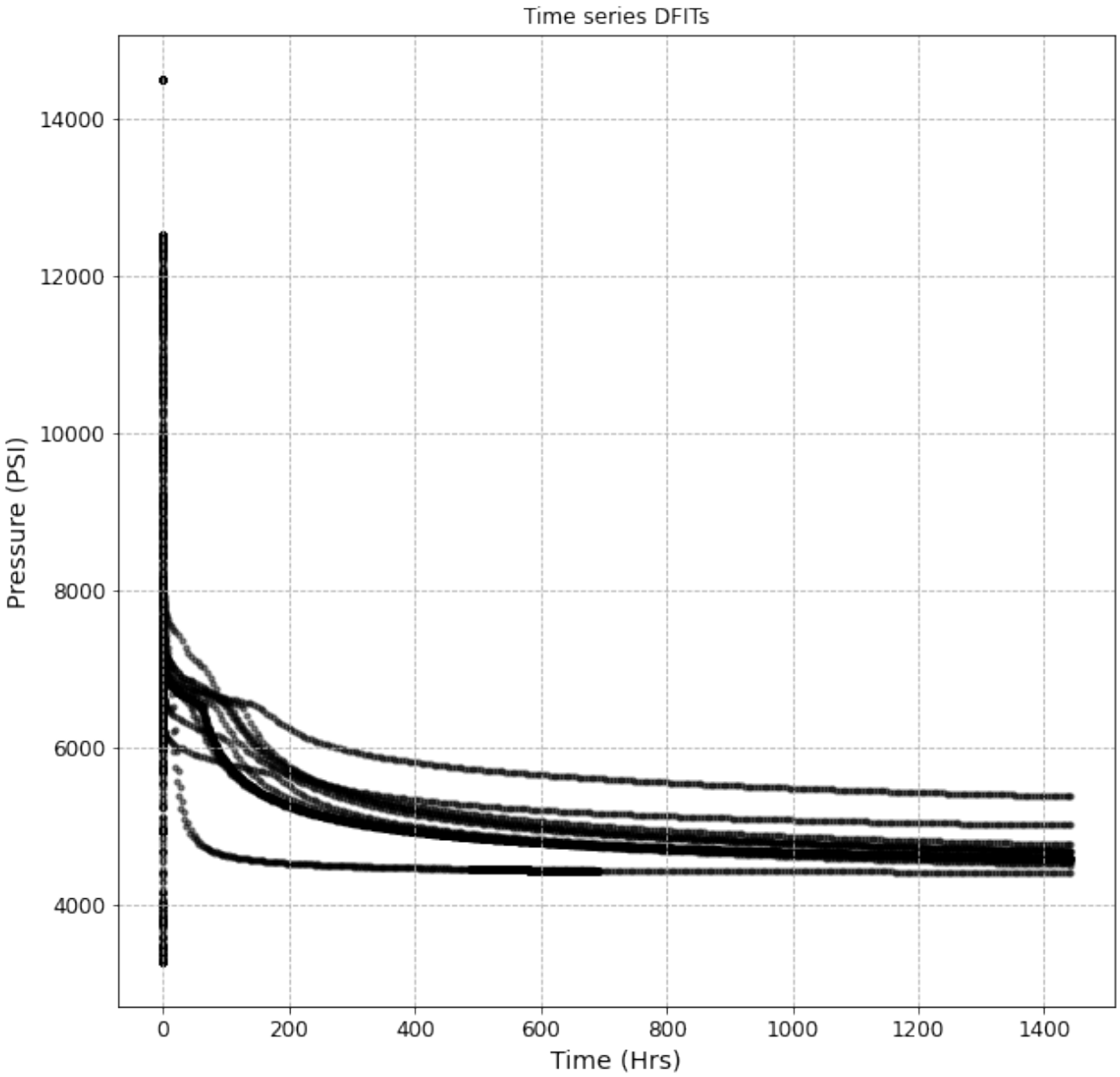


Figure 2.1: Summary plot of all synthetics generated using ResFrac®.

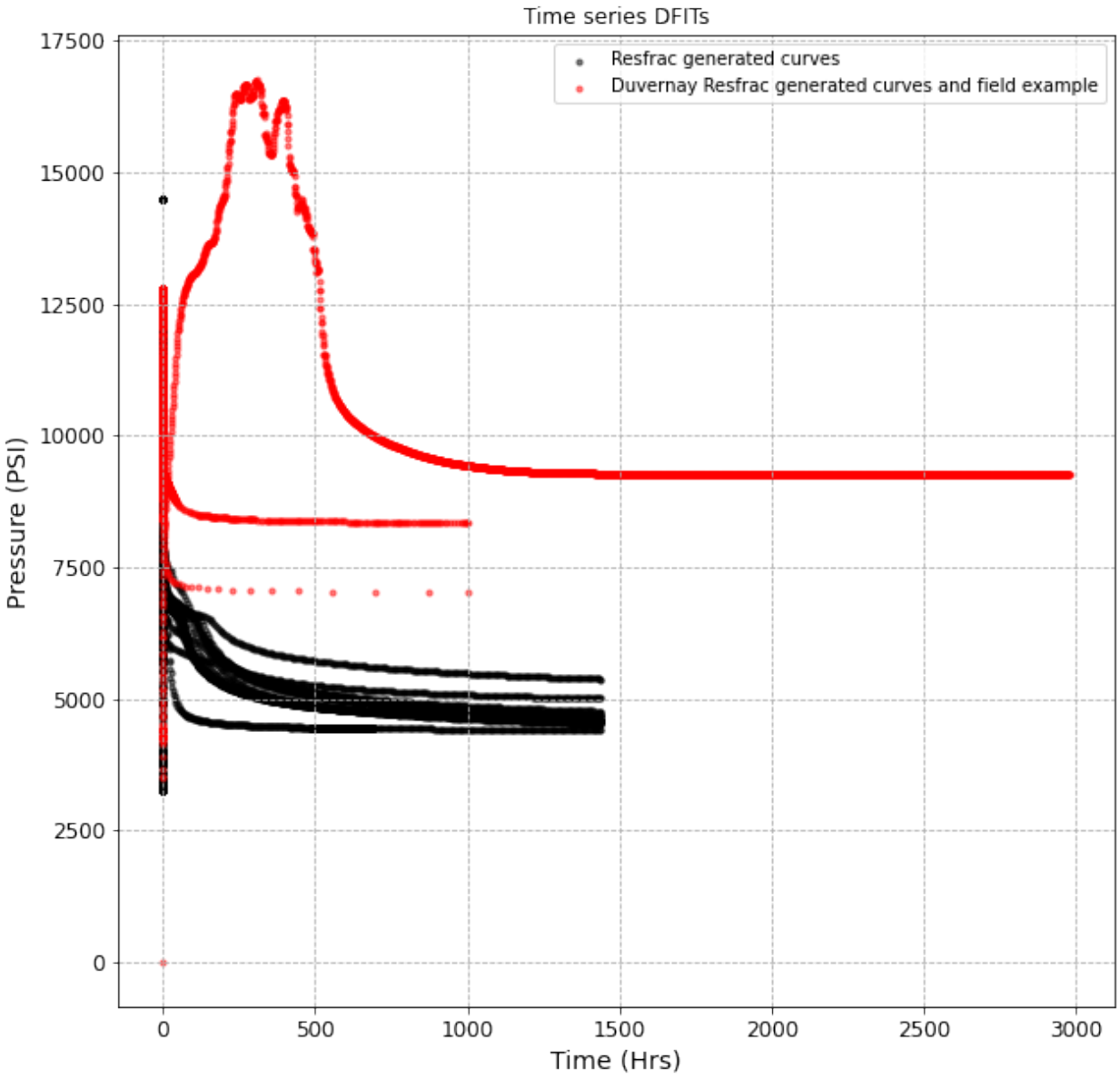


Figure 2.2: Summary plot of all synthetics generated using ResFrac® with Duvernay generated synthetics and field cases included (red).

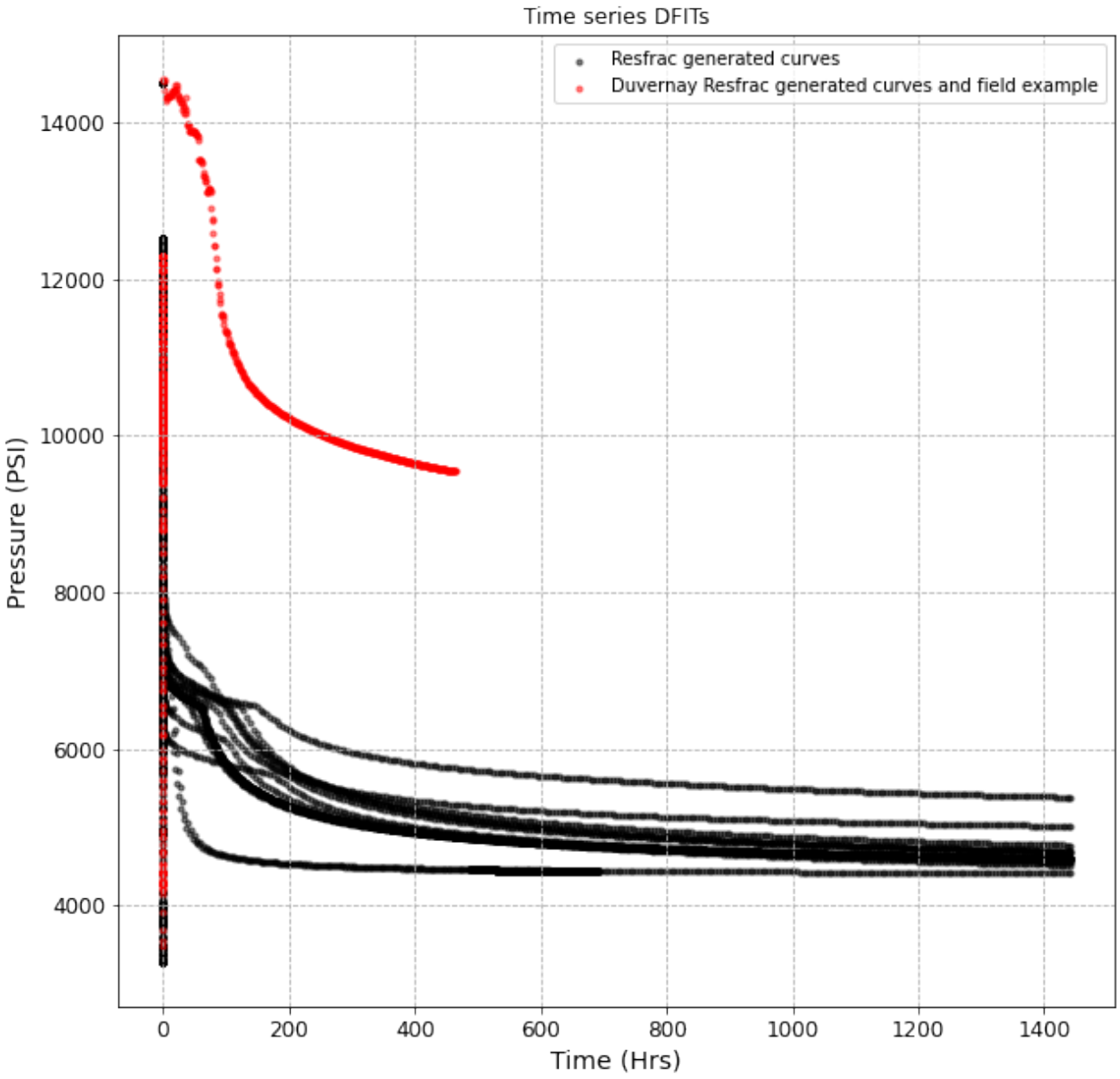


Figure 2.3: Summary plot of all synthetics generated using ResFrac®. With augmented Duvernay generated synthetics and field cases included (red).

Chapter 3

Shiny web applications for unsupervised DFIT event detection

3.1 Introduction

The first method of DFIT interpretation optimization falls in the center of Figure 1.9. This is achieved by creating the the *CREWES DFIT Clustering App* to test the feasibility of unsupervised ML methods in *R-Studio Shiny Web App* coding platform. In doing so, all domains of the data scientist Venn diagram are merged to create a holistic interpretation. Examples of web apps being applied to data analytics vary from risk assessments (McGuinness and Higgins, 2021) to Covid-19 tracking platforms (Valls et al., 2020). Here I will apply web apps in a new method for DFIT interpretation.

With the use of *R-Studio Shiny Web Apps*, this study evaluates the ability of unsupervised clustering methods: *K-Means* (MacQueen, 1967), *DB-Scan* (Ester et al., 1996), *Hierarchical modeling* (Ward, 1963), and *Gaussian mixture models* (Redner and Walker, 1984) to identify key parameters: *ISIP*, Sh_{min}, and $P_{reservoir}$ in multivariate data. *K-Means* was selected as the baseline method for defining hyper-parameters; *DB-scan* is tested for handling noisy data; *Hierarchical modeling* offers cluster visualizations for hyper-parameter selection, and

Gaussian mixture models offers the ability to fit model shape distributions in the form of probabilities.

The choice of applying unsupervised learning methods was influenced by its ability to eliminate bias that might otherwise exist in training datasets for supervised learning methods. Mathematically, it is hypothesized that the clustering algorithms will be segmenting the pressure decline curves based on the density of point distributions along original and derivative of pressure decline curves. The observations of these clusters will later allow for mathematical inferences to be made about *why* the clusters appear where they do. This approach follows a similar approach to the study by Ippolito et al. (2021) where well log facies identification is performed using unsupervised learning in conjunction with supervised learning to reduce bias. Although supervised learning will not be applied in this study, the developed method could be used to create training datasets for supervised learning applications. The application of unsupervised clustering methods also recreates a real life scenario where catalogs of events may be incomplete, or inaccurate making supervised learning unfeasible. Other studies by Li et al. (2021) use clustering more generally to identify anomalies in multivariate datasets. This idea closely parallels this study's method of identifying key-parameters ('anomalies') in a multivariate set of DFIT and pressure derivative curves.

3.2 Methods

To evaluate the DFIT clustering method, three DFIT curves were clustered and compared to results from manual interpretation. These curves include history matched pressure decline models generated from a 'simple' 3-layer Duvernay system, and a 'complex' 31-layer Duvernay system. ResFrac® simulator was used to generate synthetic pump-in/shut-in response. The detail of ResFrac® conceptual model and numerical approach is described in McClure et al. (2021). The use of modeled curves allows for key parameters (*ISIP*, Sh_{min}, and $P_{reservoir}$) to be known as simulation inputs eliminating any interpretation bias. Field data from a

DFIT acquired in the Duvernay near Fox Creek, Alberta, Canada is lastly tested using the optimized hyperparameters from the model cases. In this example, key parameters are manually interpreted from the field DFIT in time (t), G-time, Agarwal time domains as well as their corresponding derivatives. Downsampling of this data was required to speed up computation times in the clustering app.

Using these three pressure decline curves, the compliance method (McClure et al., 2016) and tangent methods (Barree et al., 2009) are both used for the estimation of Sh_{min} outputting pressures of $P_{compliance}$ and $P_{tangent}$, respectively for the field DFIT. The manual interpretation workflows for interpreting these values is also discussed in Chapter 1.

Full evaluation of the clustering app is achieved by comparing estimated values to interpreted *ISIP*, $P_{compliance}$, $P_{tangent}$, and P_{Linear} in the *Analytical value comparison* section (3.4.4), and the corresponding Sh_{min}, and $P_{reservoir}$ in the *True value comparison* section (3.4.5). In this study, the cluster boundaries are defined as the event locations along the curve.

It should be noted that to extract the full value of clustering methods, multiple input parameters must first be calibrated. These include: (1) input variables, (2) hyperparameters. and (3) preferred clustering methods. The following sub-sub sections will discuss these in more depth.

(1) Input variables

The input parameter encompasses the selection of multiple derivative curves that are input into the clustering algorithm. Variables used for interpretation and multivariate analysis in the clustering application are displayed in a correlation matrix in Figure 3.1. In this figure, there are fourteen time-dependent variables that can be interpreted to derive key parameters. To decrease noise effects, the Bourdet derivative (Duong, 1989) was applied to curves as a way of smoothing the data, this is indicated by a 'B' in Figure 3.1. To account for the different scales of measurement (Figure 3.1) scaling was applied before input into clustering algorithms

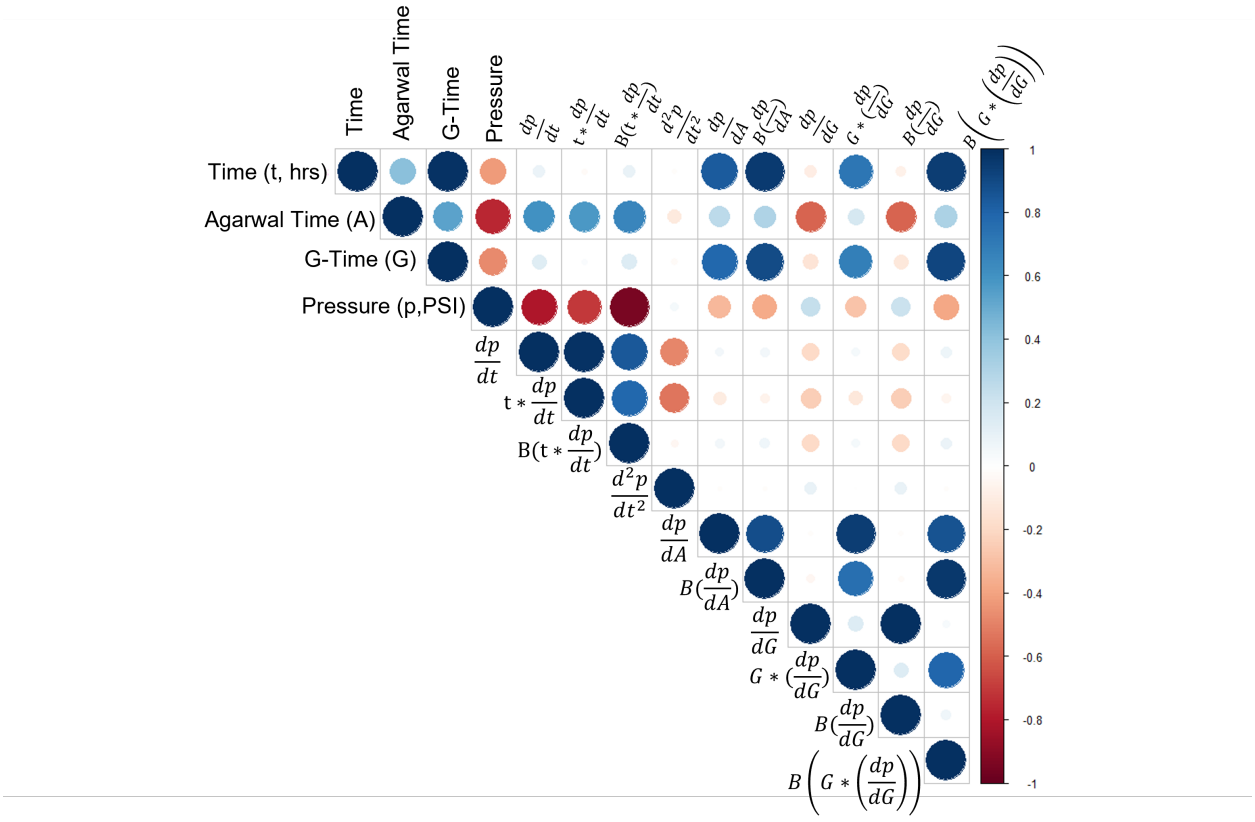


Figure 3.1: Correlation matrix of all variables that can be derived from a pressure vs time DFIT measurement. The main diagonal corresponds to a variable correlated to itself, therefore, correlation is 1 (positive correlation large blue circle). Outside of the diagonal, correlations are displayed between different variables. Red circles indicate negative correlation. The bottom left corner of the matrix has been eliminated due to its symmetry.

to avoid any data bias. To evaluate optimal variables for clustering, various combinations are iterated and culminated spread sheets are visualized on Principle Component Analysis (PCA) correlation circles (section 3.3.1). PCA is a way of identifying and visualizing dimensions with most variation in high dimensional data (Pearson, 1901).

(2) Hyperparameters

Hyperparameters control the structure of the clustering method calculations (Yu and Zhu, 2020). This may include number of clusters formed, search radius, and minimum points required to form a cluster. A full description of the hyperparameters for each clustering

method can be found in the appendix (Table B.1). To evaluate optimal hyperparameters in this study, various combinations are iterated and summarized spread sheets to find the optimal variable inputs.

(3) Clustering method

Four clustering methods listed in section 3.1 are tested in this experiment, each with their own mathematical searching criteria (Appendix Table B.2). To evaluate these methods the three metrics of performance are compared. These include:

(A) Average number of events identified (varied hyperparameters).

(B) Average number of unclassified points along curve (varied hyperparameters).

(C) Average repeatability of classified points for varied hyperparameters (varied hyperparameters).

Following the calibration and evaluation of the clustering app's performance, PCA is then used in the app to explore *why* cluster boundaries appear where they do (Figure 3.2). The following subsection 3.2.1 will discuss the construction of the *CREWES DFIT clustering App.*

3.2.1 CREWES DFIT Clustering App

The developed *CREWES DFIT Clustering App* addresses the challenge of visualizing fourteen-dimensional data (Figure 3.1) with the application of unsupervised clustering and PCA. To develop the *CREWES DFIT Clustering App R-Studio®* programming software was used. *R-Studio®* offers the ability to design and create interactive web apps (*Shiny web app*) for data manipulation and visualization. The benefits of creating and using the *Shiny web app* include:

(1) **Time saved**: Eliminate the need for a user to run multiple sections of code to generate several plots for various clustering analyses types.

(2) **Reactive variables**: User can quickly manipulate hyperparameters for data fit.

(3) **Intuitive display**: Back-end code runs without the user requiring extensive knowledge of programming.

Figure 3.2 displays a screen capture of this developed app. Additional features such as elbow plots and tree diagrams are also included as visualizations within the app to aid in hyperparameter definition (Figure 3.2). The app allows for the user to magnify curves, observe different clustering methods, and visualize where clusters fall on DFIT pressure declines.

For a more detailed workflow of the DFIT clustering app, a dataflow diagram is presented in Figure 3.3. This figure demonstrates the difference between user interactive (green outline) and hidden features (red outline) of the App. In this workflow, a user will upload their pressure versus time DFIT curve and default parameters will cluster the data. The user can then visualize how the clusters fall on the multiple curves created in the web app process (Figure 3.2). This information can be used to update hyperparameters, and re-perform clustering calculations. If the user is satisfied with the end interpretation, parameter adjustment is no longer required. Figure 3.3 is also color coded based on the data science domain applied to the data from Figure 1.8. Notably, the app allows for the integration of all three data science domains.

3.3 Results

3.3.1 Variable optimization

Using the 'simple' and 'complex' Duvernay models, optimal hyperparameters and variable combinations were tested to fit DFIT events representing key pressure data. The iterative

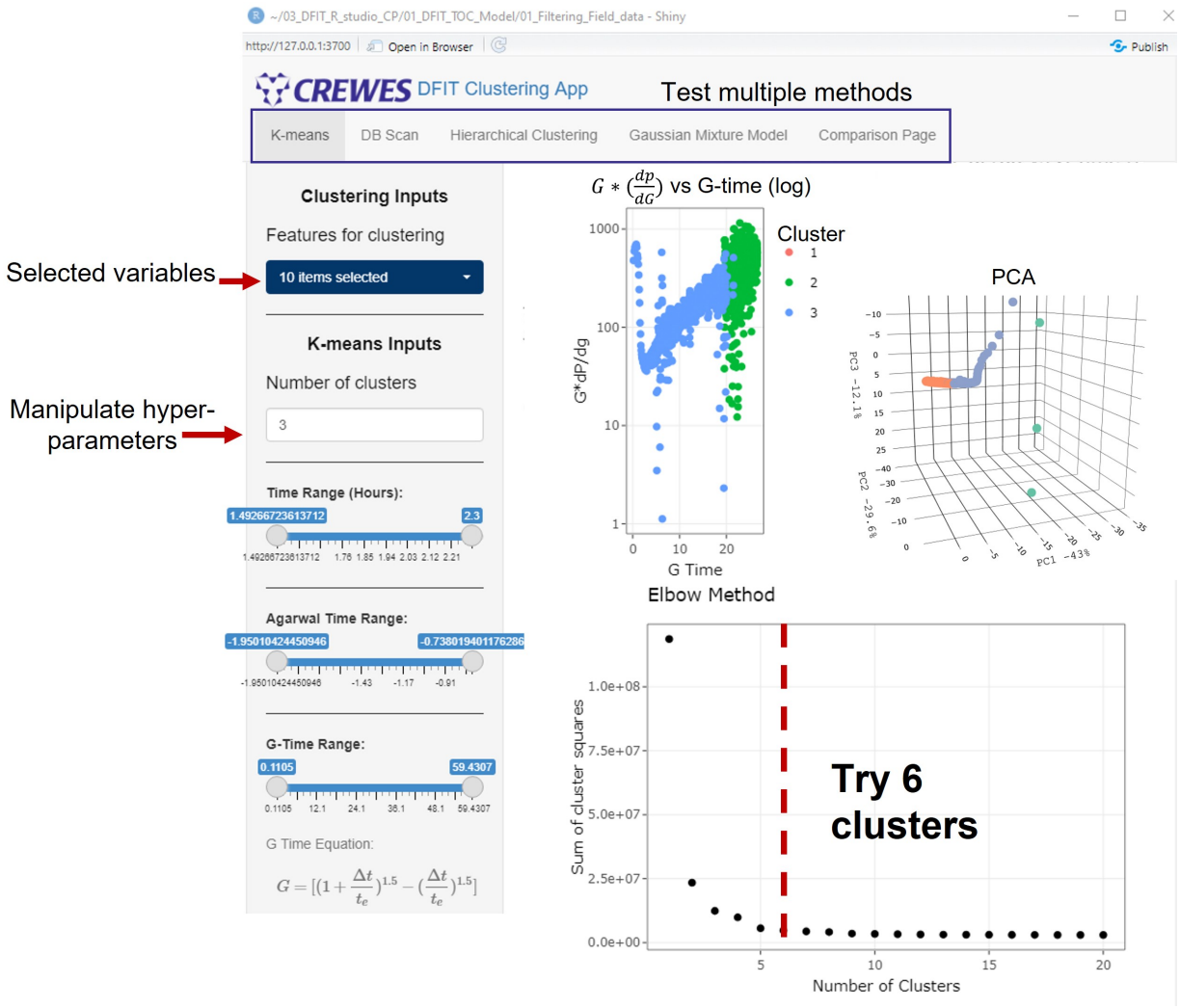


Figure 3.2: Screen capture of the interactive *CREWES DFIT Clustering App* with its various analytical features highlighted.

process also allowed for inferences to be made about the relative importance of variables and their overall contribution to clustered events. This is displayed as a PCA circle in Figure 3.4 for K-means clustering. In this figure, the DFIT pressure vs log-time curve for the 'complex' model is clustered using variables groups identified by the PCA correlation circle. Plot A shows an example where only G-time derivatives are used as inputs of the clustering algorithm: the outputs of the clustering algorithm are displayed on a pressure (PSI) vs log time plot with key events identified and their corresponding error relative to the model inputs. Plot B shows the a similar plot using Time, G-time, and Agarwal time

derivatives. Plot C illustrates an example where only the time derivate is clustered. Plot D shows the result of using all variables in the clustering algorithm.

3.3.2 Hyperparameter and clustering method optimization

Following variable analysis, elbow plots and visual inspection are used to determine optimal hyperparameters for each clustering method. A summary of all iterations for each clustering method applied to each of the three DFITs is also displayed in histograms in Figures: 3.5, 3.6, 3.7. Applying the three measures of performance allows for this optimization to be quantified. Optimization results for each clustering method are displayed in Table 3.1. Using these clustering parameters, the three measures of performance can be further used to determine the best clustering method out of the four methods tested (Figure 3.8). These measures of performance are compared for the three DFIT model datasets in this study (Figure 3.8).

3.3.3 Final cluster interpretation (analytical values)

The combination of the optimized variables, hyperparameters, and clustering method that produced cluster boundaries correlated to key parameter events displayed in Figure 3.9 for the 'simple,' 'complex,' and raw field Duvernay DFIT curves, respectively. On these plots, the percent difference from the true/interpreted value is also displayed. It is noted that the metrics of evaluation generally give results that do not include P_{Linear} for this reason, Figure 3.10 was created to show clustering for this key parameter. Summary plots in Figures: 3.5, 3.6, 3.7, are also used to make inferences about the relationship of clustering boundaries to analytical values.

3.3.4 Final cluster interpretation (true values)

The same plots described in section 3.3.3 are used to investigate the effectiveness of clustering boundaries to identify Sh_{min} and $P_{reservoir}$. Percent difference for the respective values is also

displayed in Figure 3.9.

3.3.5 Mathematical analysis of clustered features

Following the identification of optimal clustering variables, hyperparameters, and clustering methods, the mathematics of *why* clusters appear where they do is explored by examining PCA plots of fourteen-dimensions reduced to three (Figure 3.11).

Table 3.1: Optimized Clustering Hyperparameters.

# Clustering method	Parameters
K-means	6-clusters
DB-scan	Minimum points = 10, Search radius = 0.1
Hierarchical Clustering	Ward-D, 6-clusters
Gaussian mixture model	VEE (Ellipsoidal, equal shape and orientation) 6-clusters

3.4 Discussion

3.4.1 Variable optimization

This new workflow using *Rstudio Shiny Web Apps*® has helped us understand and evaluate the feasibility of using unsupervised clustering methods to interpret events on a DFIT pressure decay curve that can be used to derive key parameters. To achieve this, variable contributions from derivative curves were first explored using a PCA correlation circle. In Figure 3.4, it appears that reducing the original and derivative data into two dimensions (Dim 1 and Dim 2) has created natural clusters of correlation. To further explore these clusters of correlation, the clustering algorithm, K-means, was run using only the variables within each cluster. Plot A shows that clustering of only G-time derivatives has resulted in events *ISIP* (highest relative error), $P_{tangent}$, and $P_{compliance}$ event identification. Plot B illustrates that an accurate *ISIP*, and lowered accuracy $P_{tangent}$ and P_{Linear} estimates, can be obtained when clustering is

performed using Time, G-time, and Agarwal time derivatives. Using time derivatives results in P_{Linear} being the only event identified in plot C. Collectively, these clusters of correlation appear to extract pressure information from different segments of the curve. The G-time events appear to give accurate $P_{tangent}$ and $P_{compliance}$ approximations (early time variables). This differs from the Time, G-time, and Agarwal time cluster which results in accurate values for *ISIP* and P_{Linear}. These clusters of variables will be referred to as the late/early time variables based on this distribution. Lastly, the time derivative cluster appears to correlate with only the late-time variable P_{Linear} (late-time variables).

Collectively, each variable cluster in the PCA plot appears to tell us different information for early and late time segments along a DFIT pressure decline. Most notably, the early time variables (G-time derivatives) exist on the opposite side of the PCA correlation circle from the late time variables (Time derivatives) suggesting that these clusters are negatively correlated. This interpretation is supported by no event overlap existing in plots A and C. Plot B contains information from both the early and late time stages of the DFIT. Therefore, its cluster of variables has some correlation to the pure late and early time variables displayed in the correlation circle of Figure 3.4 lying orthogonal to these clusters. This may imply that depending on the events required from the clustering algorithm, different selections of variables can optimize output results for early/late time DFIT events. For the purpose of this study, a generalized approach was taken whereby all clusters of variables are merged to produce a holistic interpretation of the DFIT curve. This is displayed in plot D of Figure 3.4. In this plot, it appears that an averaging of the components of each variable cluster has created a holistic interpretation. This plot identifies *ISIP* with improved accuracy compared to plots A and C, improved $P_{tangent}$, compared to plots B and C, and a P_{Linear} that was non-existent in plot A.

3.4.2 Hyperparameter optimization

The collective 14-variable approach then allows for cluster hyperparameters to be optimized. Manually testing parameters leads to the optimizations displayed in Table 3.1. Generally, 6-clusters for *K-means, Hierarchical*, and *Gaussian* clustering methods appear to fit the data best.

A histogram summary of cluster boundary value vs iterations of hyperparameters for each clustering method and DFIT is also displayed in Figures: 3.5, 3.6, 3.7. Figure 3.5 shows the results of varying hyperparameters for the 'simple' Duvernay model. In this Figure, the summary plot of all clustering method iterations summed appears to have convergence around the actual values of *ISIP*, $P_{tangent}$, $P_{compliance}$, Sh_{min}, and P_{Linear}. Notably, $P_{reservoir}$ does not have any occurrence on this plot, this is because $P_{reservoir}$ (7000psi) occurs at a pressure that is lower than the last measurement of the DFIT pressure decline (7027psi). This observation of missing $P_{reservoir}$ applies to the other two curves in this study as well. Further decomposition of the histogram into clustering methods reveals *DB scan* to have the best convergence followed by *Gaussian mixture models*, *Hierarchical clustering*, and *K-means*.

The summary histogram in Figure 3.6 also displays general convergence to key parameters *ISIP*, $P_{tangent}$, $P_{compliance}$, Sh_{min}, and P_{Linear}. In this case the frequency of values between $P_{tangent}$ and $P_{compliance}$ appears to be skewed towards $P_{tangent}$. Perhaps this is due to Sh_{min} having a close overlap to $P_{tangent}$ and the clustering algorithms detecting this. Further decomposition of the histogram into clustering methods reveals *DB scan* to have the best convergence followed by *Gaussian mixture models*, *Hierarchical clustering*, and *K-means*.

The summary histogram from the Field Duvernay DFIT in Figure 3.7 displays weaker convergence to key parameters *ISIP*, $P_{tangent}$, $P_{compliance}$, and Sh_{min}. P_{Linear} cannot be identified due to missing data on this DFIT. The weaker convergence of the summary plot is likely due to the poor performance of individual clustering methods *K-means* and *Hierarchical clustering*. Closer inspection of the *DB scan* histogram yields higher correlation to actual values and the *Gaussian Mixture* histogram appears to have the closest correlation to the

actual key parameters. Comparing back to Figure 3.6, Sh_{min} appears to have poor correlation to histogram frequency in Figure 3.7. Perhaps this is because Sh_{min} does not overlap with $P_{tangent}$ or $P_{compliance}$ values and roughly bisects them. As a result, cluster values bimodally distribute to areas of physical change on the curve, such as $P_{tangent}$ and $P_{compliance}$. This may be the result of biases in fracture flow behaviour when generating these curves (see Carter flow discussed in Chapter 1).

3.4.3 Clustering method optimization

Following the optimization of variable inputs and hyperparameters, clustering methods are quantitatively compared using the three parameters of evaluation in the form of a cluster performance matrix (Figure 3.8). Using the 'simple' model, *DB scan* appears to have identified the highest percentage of events, while having the lowest average percentage of noise points and highest repeatability with hyperparameter variation. Observation of the *DB scan* histogram (Figure 3.5) also appears to show the strongest convergence to the key events while having limited outlier points.

This trend appears to shift as the model becomes more complex. For the complex Duvernay model, *DB scan* appears to retain high event identification and repeatability, however, the quality of value uniqueness is significantly reduced by the number of unidentified points that appear in the result (12). Compared to other methods, *Hierarchical clustering* has improved metrics for all three measures of performance, and the *Gaussian-based* method appears to have optimal performance with the highest average percentage of events identified and repeatability. Inspection of Figure 3.6 also reveals *DB scan* and *Gaussian mixture models* as the top performers, with *DB scan* having slightly better convergence.

Analysis of the field data revealed the *Gaussian mixture model* method as the top performer. In the field case, the quality of the *DB scan* method appears to have degraded in quality while the K-means method has improved. Further inspection of histograms in Figure 3.7 also reveals stronger convergence in the *Gaussian mixture model* compared to *DB scan*

with more outlier results. Perhaps the noise in the field case has degraded the performance on DB scan, creating more outlier clusters that give trivial interpretation.

It is important to note that Figure 3.8 shows the averages of multiple iterations of the clustering algorithms with varied hyperparameters to quantify the algorithm's ability to handle deviations from optimal hyperparameters (repeatability). Individual tests with optimized variable input, hyperparameters, and clustering method were found to output 75 -100% identification. The differentiator between these percentages is merely dependent on if the length of recorded data is long enough to extract $P_{reservoir}$. This applies to the field test case. In the following subsection we will explore how optimized clustering methods perform when interpreting DFIT data.

3.4.4 Analytical value comparison

When comparing interpreted analytical values to clustered results, *DB scan* methods appear to produce optimal results when the DFIT is 'simple' and degrade as complexities are introduced. This is illustrated in Figure 3.9 where the application of *DB scan* to the complex model (plot B) has resulted in more noise clusters being identified, degrading the uniqueness of key parameter event detection on the curve. DB-scan applied to the simple model appears to have identified accurate *ISIP* and $P_{compliance}$ events in Figure 3.9A, with little noise, while other events appear to be missed. This is explained by the lower apparent sampling of the simple model data creating sparse point density displayed in Figure 3.9A. Perhaps the sampling interval has affected the accuracy of the clustering methods and this may be a subject of future study.

For the field test, (Figure 3.9C) the *Gaussian mixture method* appears to have identified *ISIP*, $P_{compliance}$, and $P_{tangent}$ pressures with only one unidentified cluster boundary that exists between $P_{compliance}$ and $P_{tangent}$. In this case, the field data were not collected for a long enough time to accurately identify a P_{Linear}, therefore, no interpretation exists in these plots. Overall, the *Gaussian mixture method* appears to handle noisy data better than *DB*

scan. This may be explained by the ability to fit probability distributions to the data that better handle noise. Future experiments may consist of filtering the DFIT data to see if this will improve clustering results.

Further analysis of Figure 3.9 reveals that none of these optimized parameter plots has picked up on the P_{Linear} event. Moreover, Figures 3.5 and 3.6 show a relatively high convergence to P_{Linear} values on their summary plots, this brings the question: why are the optimized results not capturing this event? The answer to this question demonstrates the adversity of using an average ranking method to determine optimal hyperparamters for all events. The reality is different combinations of hyperparameters will yield different optimizations of events. This is similar to observations made about variables in section 3.4.1. An example of a differently optimized hyperparameter *Gaussian mixture model* is displayed with P_{Linear} accurately picked in Figure 3.10. This demonstrates that although we have optimized our input variables to pick all boundaries, hyperparameters can still be optimized. An example of this was demonstrated in section1.4 where the late vs early fracture physics boundary requires slightly different optimizations of parameters.

3.4.5 True value comparison

The identification of a relatively accurate Sh_{min} appears to have occurred on plots B and C of Figure 3.9. Further inspection of histograms such as *Gaussian mixture model* in Figure 3.7 also reveal contrasting results where Sh_{min} has poor frequency of occurrence. This may be interpreted as the inherent biases introduced by fracture physics assumptions made in the derivation of input clustering curves in section 1.4.

DFIT recording times cut short lead to inconclusive results being formed about the identification of $P_{reservoir}$ using clustering methods. However, a recorded dataset that encompass a true $P_{reservoir}$ most often is not economical for an operator, therefore, this simulates real life results.

3.4.6 Cluster mathematics

There still remains the question as to *why* these clusters appear where they do. Inspection of Figure 3.9 does note conclusively show why these clusters occur. This question can be addressed when observing three-dimensional PCA plots for the complex model with K-means applied (Figure 3.11). In this figure, events appear to correlate with inflections in data trend (see *ISIP* and $P_{tangent}$ in original and first magnification of PCA) and frequency variation in data (see the largest magnification). This implies that a combination of dimensionality and frequency changes in the data affect point density and cluster distribution. It is hypothesized that the input of these manual interpretation derivative curves into the clustering algorithms has allowed for these changes to be identified that line up with DFIT parameter events.

3.5 Conclusions

Successful development of unconventional hydrocarbon reservoirs is dependent on designing and modeling an effective stimulation program. This typically requires estimation of critical parameters *ISIP*, Sh_{min}, and $P_{reservoir}$ via manual interpretation methods using DFIT pressure falloff data. Although this process may produce values for stimulation modeling, it is time-consuming and can be affected by human interpretational bias. To address this adversity, a new method of applying unsupervised clustering methods in the *CREWES DFIT Clustering App* was developed. This app allowed for quick visualization and manipulation of clustering variables and hyperparameters to find the best fit interpretation for three sets of DFIT pressure falloff data. Clustering calibration results found that different variable inputs into the clustering algorithm result in different events being identified on a DFIT pressure decline. These were classified and early and late time variables. For a generalized interpretation of the DFIT curve, the variable clusters are merged to identify late and early events that occur on the pressure decline.

Optimized results suggested that the *DB scan* method can accurately define event

boundaries on the 'simple' DFIT model, however, the introduction of geologic complexity and noise degrades the result as more unclassified clusters appear in the interpretation. This is where the Gaussian mixture method appears to handle noise variations with improved accuracy for the 'complex' and field DFIT tests. It is hypothesized that the ability to change the shape of the probability distribution fitting the data in this method has addressed any noise contamination and effects of geologic complexity. Future studies may focus on eliminating this noise with filters.

While analytically interpreted values: *ISIP*, $P_{tangent}$, $P_{complaince}$, and P_{Linear} appear to have strong correlation to cluster boundaries, true Sh_{min} appears to be less consistent. This is interpreted to be an effect of biases introduced in the derivative curve generation from fracture mechanics assumptions. $P_{reservoir}$ was not evaluated due to missing data in all DFIT curves.

Understanding *why* clusters occur where they do is achieved by using PCA to reduce the fourteen-dimensional data down to three dimensions. This process revealed that cluster boundaries occur at inflection points (changes in dimensionality) and frequency variation in the data correlates to a variation in point density.

The *CREWES DFIT Clustering App* offers the ability to quickly interpret and reduce bias in DFIT-derived parameter estimates. Future chapters will evaluate the effectiveness of applying supervised learning methods as an alternate means for DFIT interpretation.

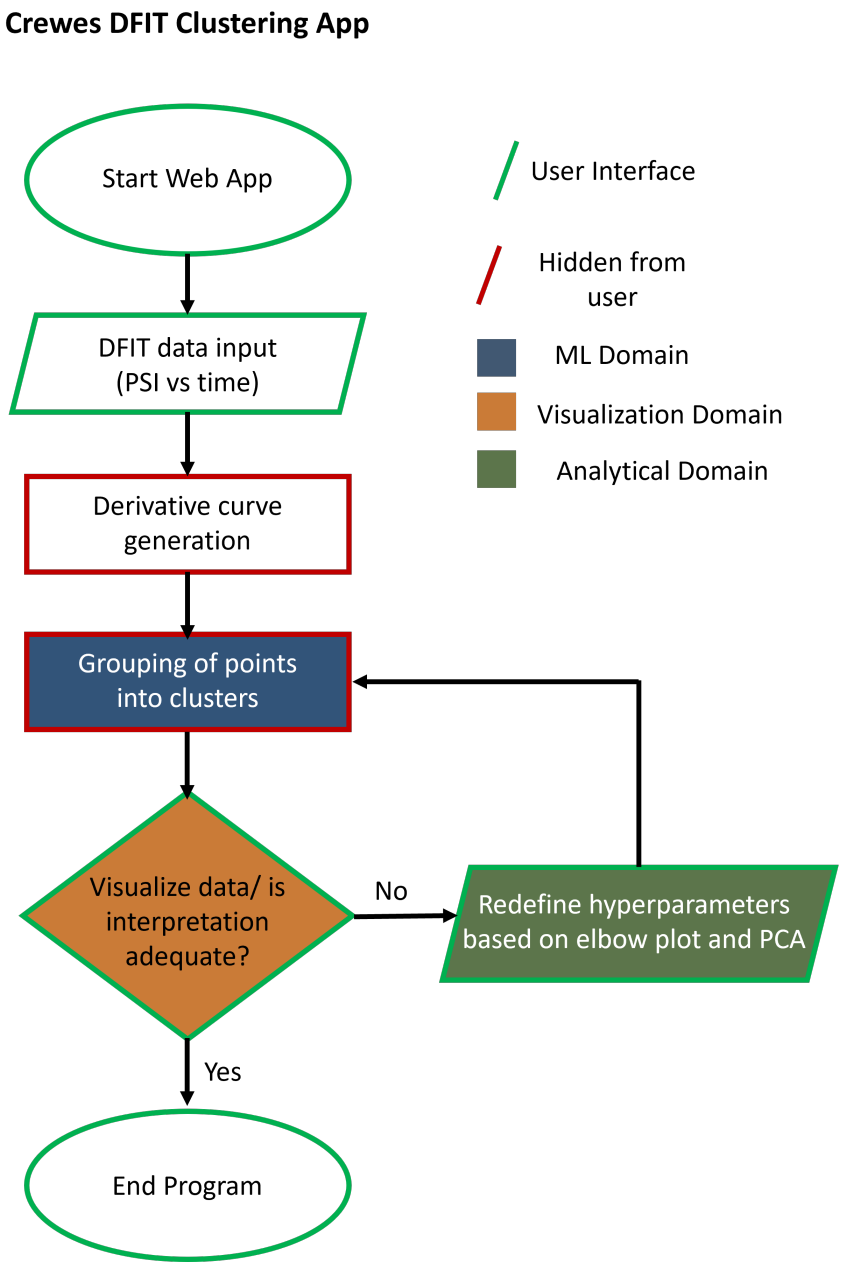


Figure 3.3: Dataflow diagram for the CREWES DFIT Clustering App. Green outlines indicate user interface while red outlines indicate features hidden from the user. Operations are also color coded based on their relationship to the data scientist domains in Figure 1.9.

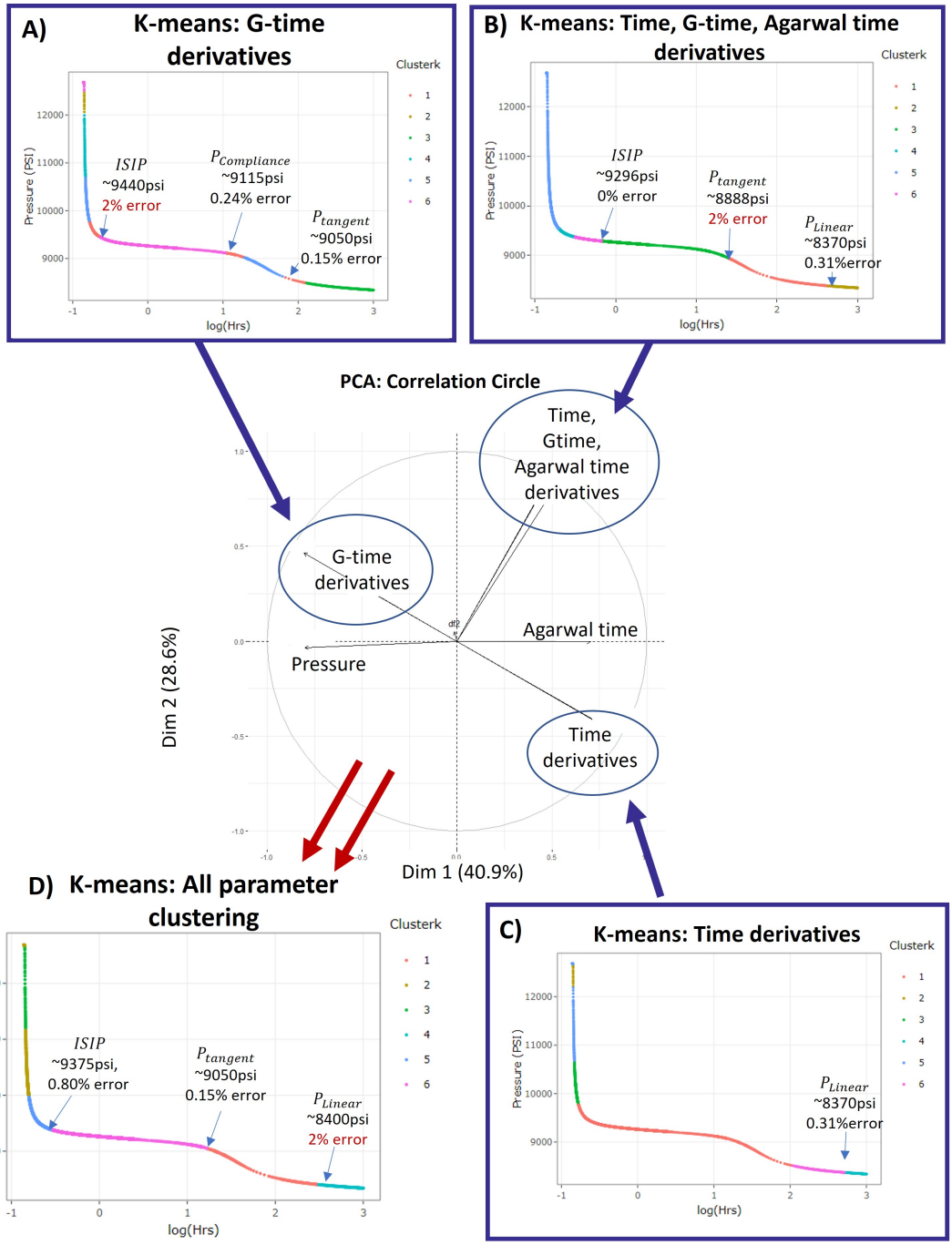


Figure 3.4: Results of testing different variable clustering for K-means using the 31-layer DFIT Duvernay model. Subsets for variable testing were determined using the PCA correlation circle (center) where clusters of variables correspond to high correlation. The axes on this plot are labeled Dim 1 and Dim 2 representing dimensions that capture 41% and 28% of the variation in the data respectively. Variables outside of clusters indicate an increasing negative correlation.

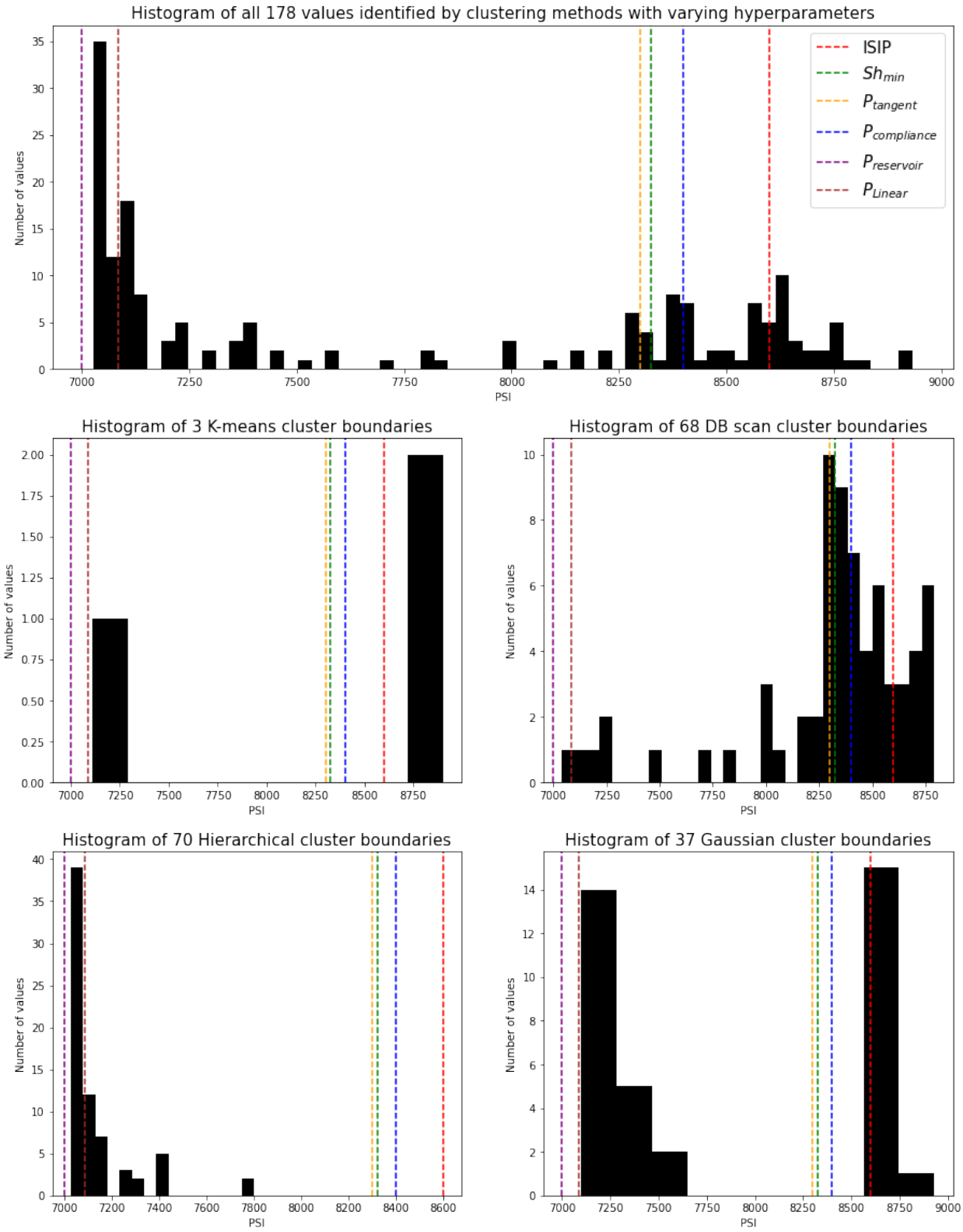


Figure 3.5: Summary plots of hyperparameter variation for each clustering method in the simple Duvernay DFIT model. The top sub plot is the sum of the bottom four sub plots. Dashed lines indicate the location of the true values $ISIP$, Sh_{min}, $P_{tangent}$, $P_{compliance}$, $P_{reservoir}$, and P_{Linear}

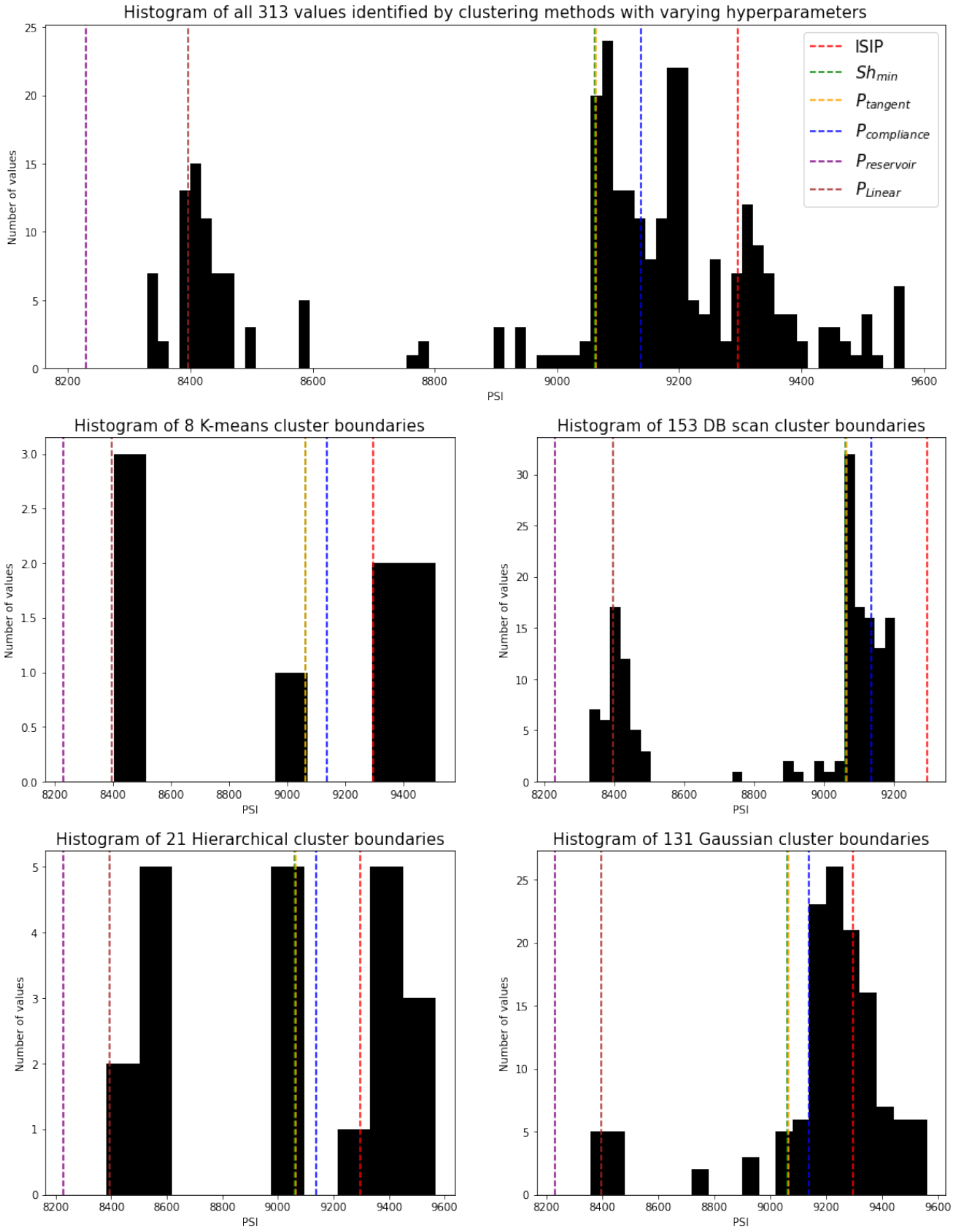


Figure 3.6: Summary plots of hyperparameter variation for each clustering method in the complex 31 layer Duvernay DFIT model. The top sub plot is the sum of the bottom four sub plots. Dashed lines indicate the location of the true values *ISIP*, Sh_{min}, $P_{tangent}$, $P_{compliance}$, $P_{reservoir}$, and P_{Linear}.

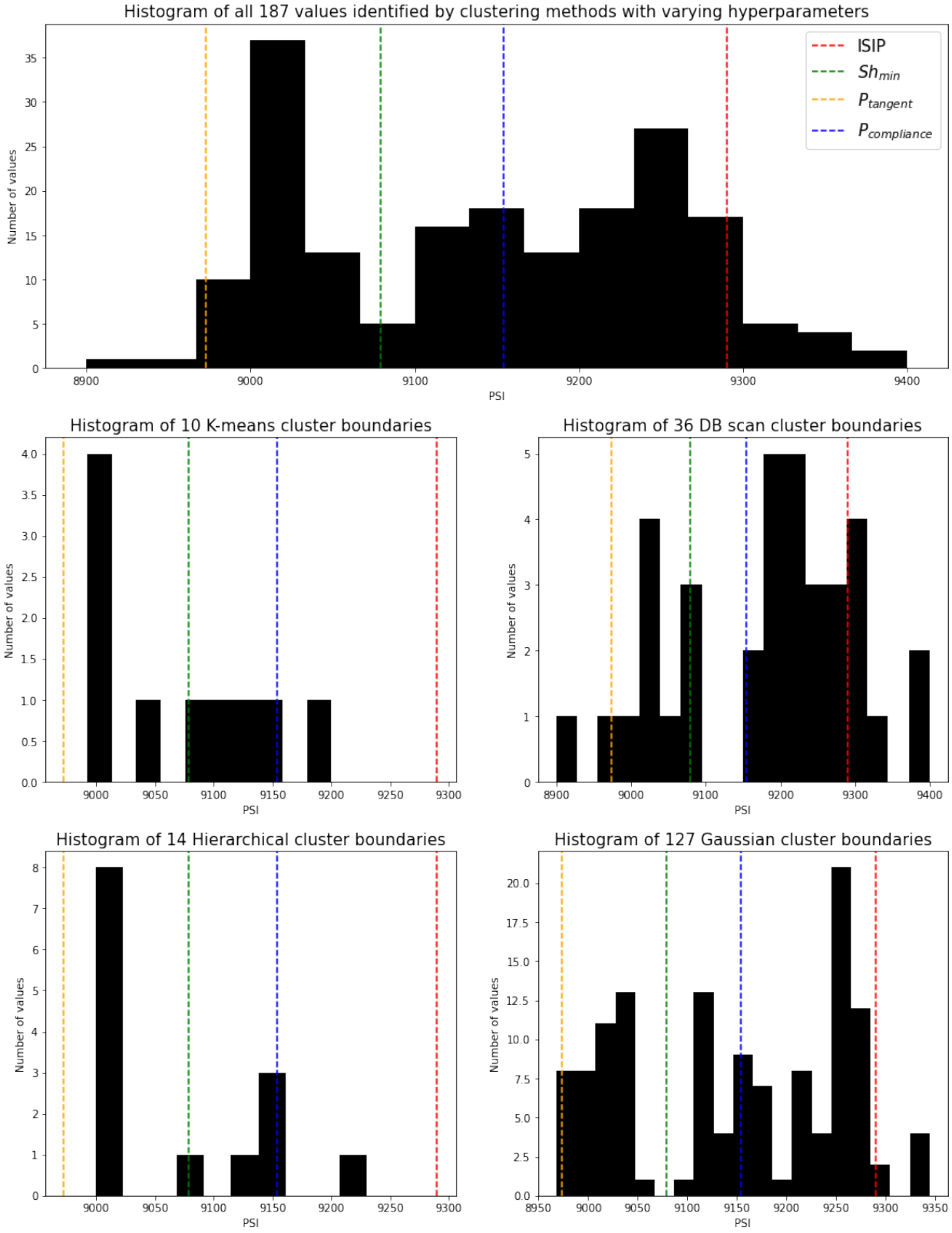


Figure 3.7: Summary plots of hyperparameter variation for each clustering method for the Field Duvernay DFIT. The top sub plot is the sum of the bottom four sub plots. Dashed lines indicate the location of the true values $ISIP$, Sh_{min}, $P_{tangent}$, $P_{compliance}$, $P_{reservoir}$, and P_{Linear}.

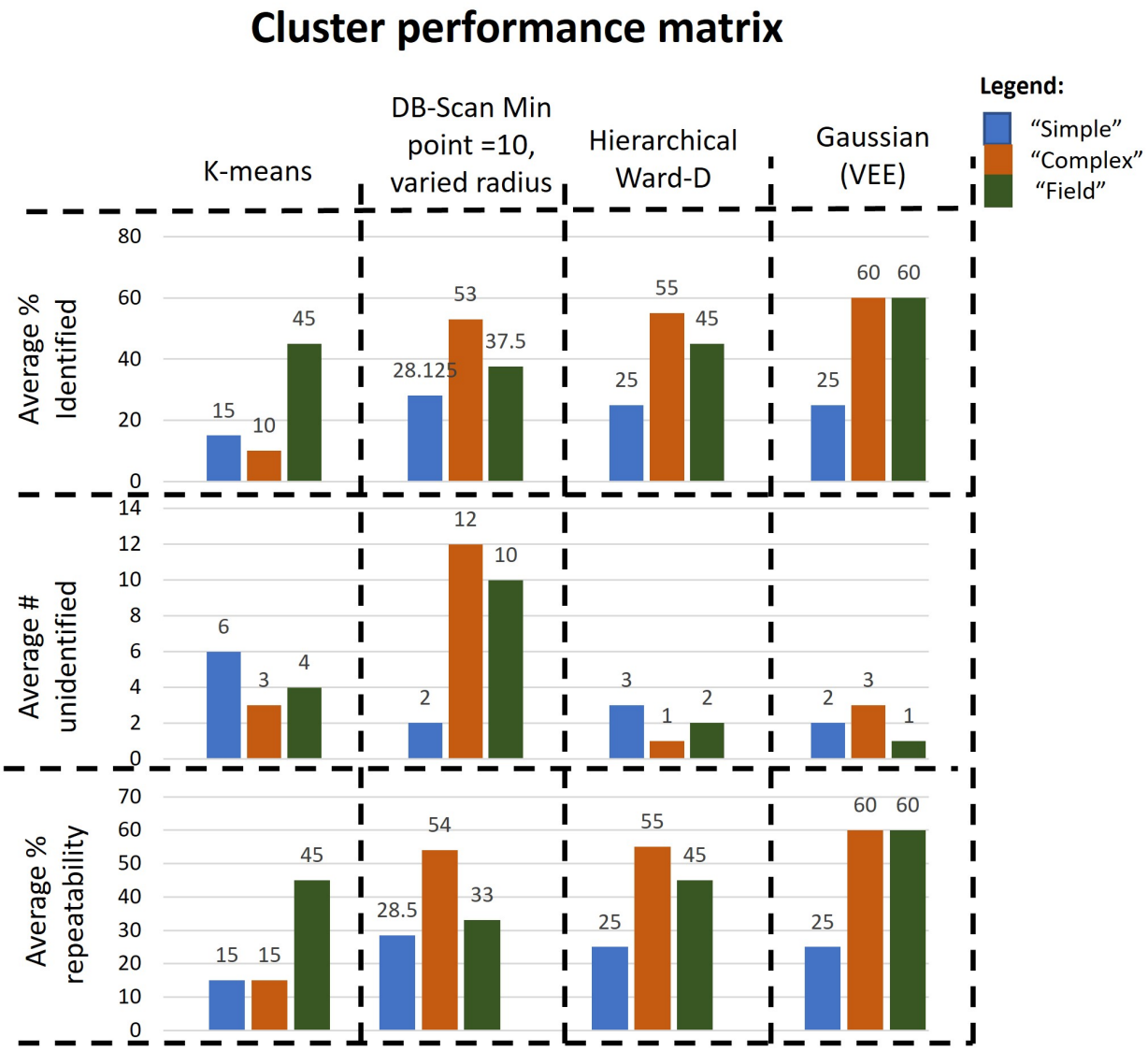


Figure 3.8: The cluster performance matrix compares the three designed metrics of evaluation to selected clustering type. Bar plots for each dataset tested are colour coded, blue = 'simple' model DFIT, orange = 'complex' model DFIT, green = Field DFIT.

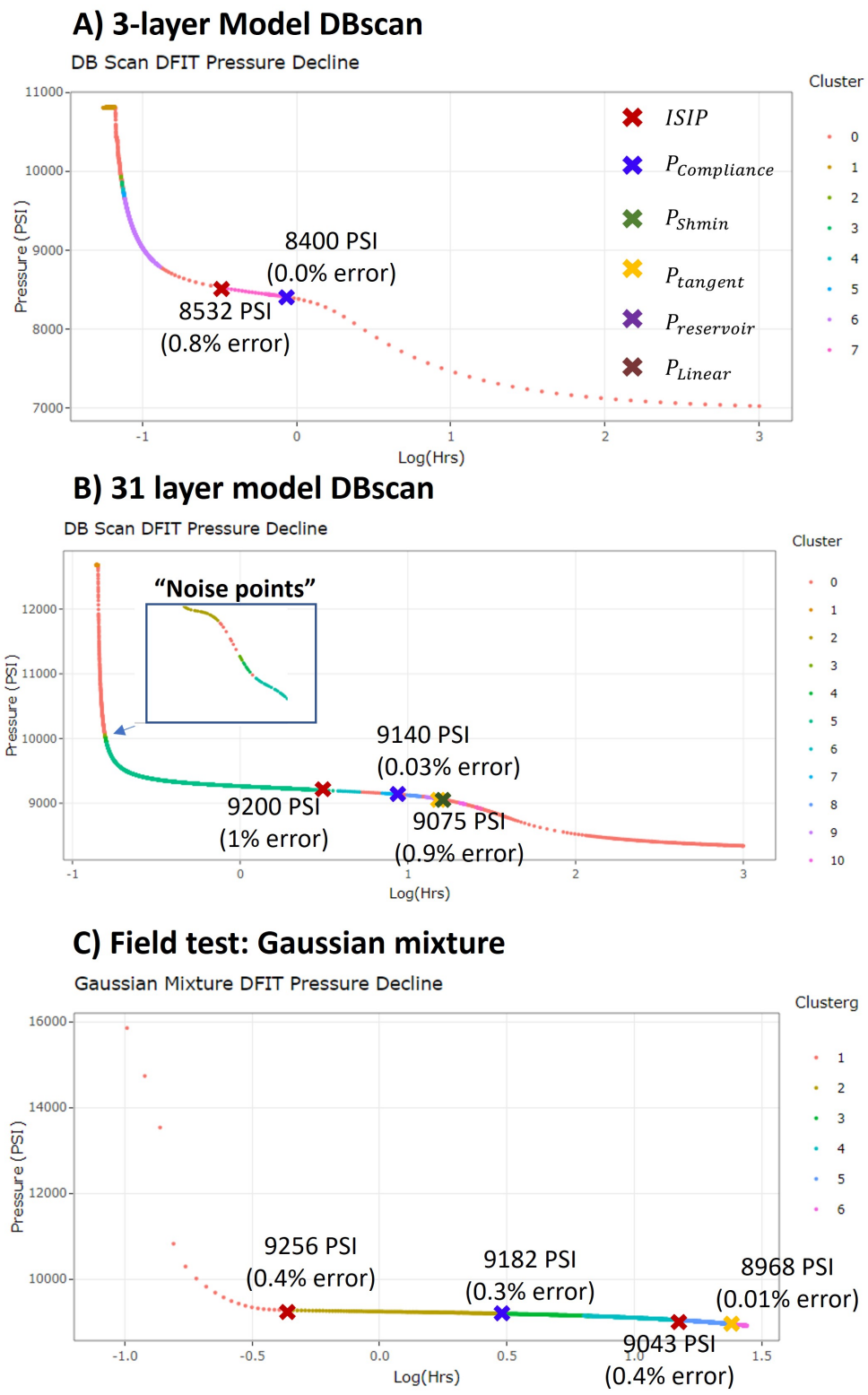


Figure 3.9: Results of applying optimal variable combination, hyperparamters, and clustering method to each of the three datasets in this study. Percent error is indicated on each plot to quantify difference from true/interpreted values.

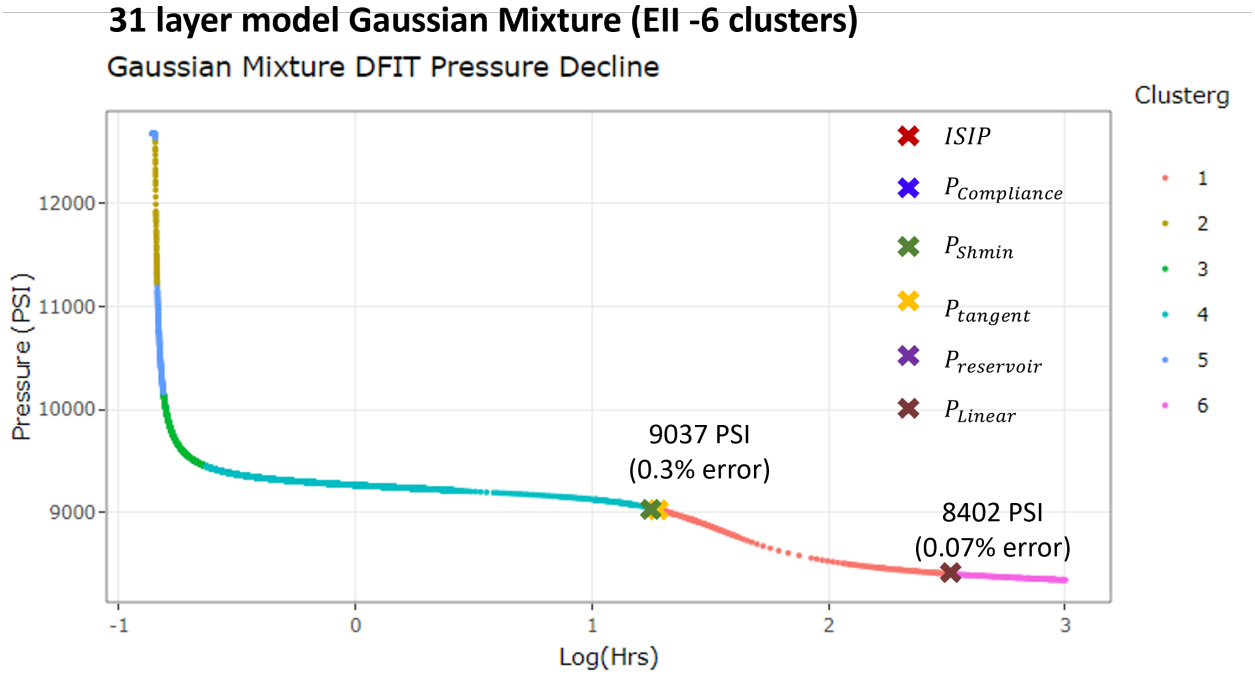


Figure 3.10: Results of applying P_{Linear} optimization of hyperparamters for the 31 layer Duvernay DFIT. Percent error is indicated on each plot to quantify difference from true/interpreted values.

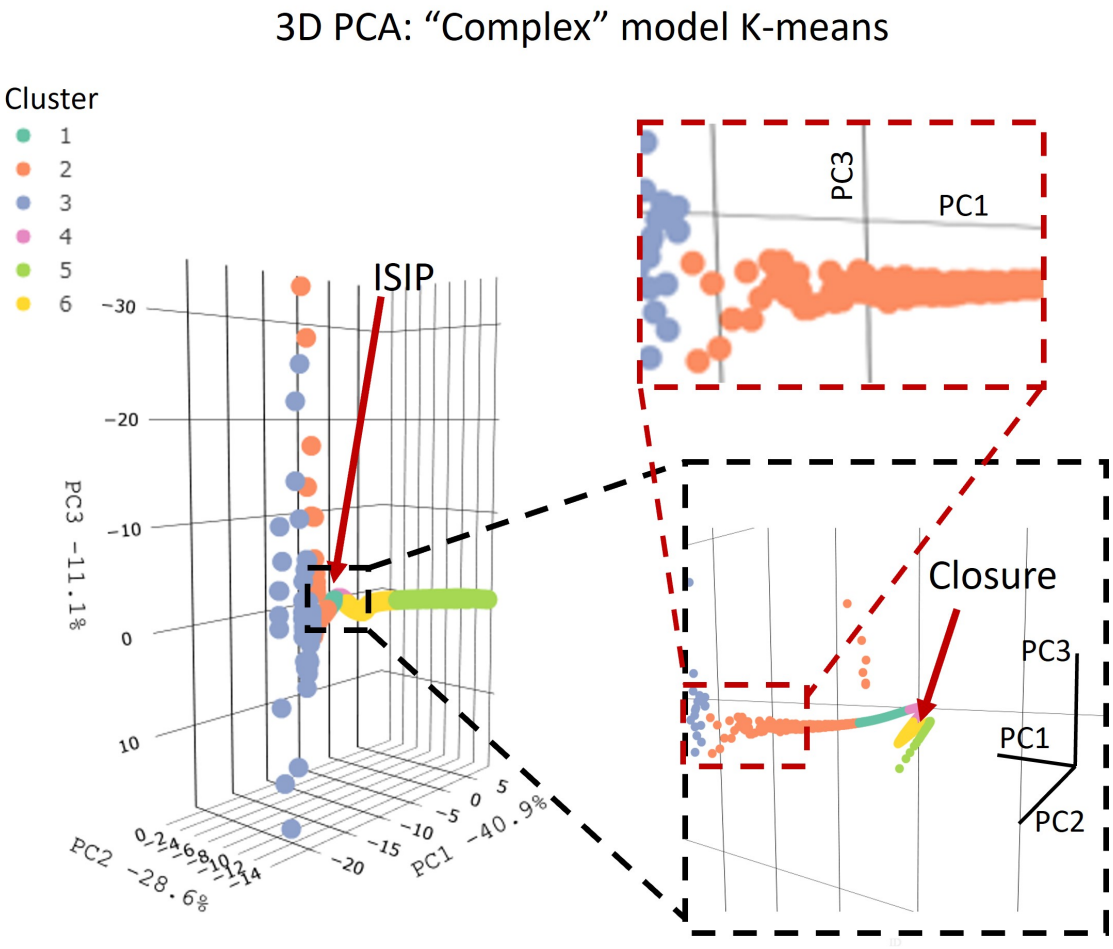


Figure 3.11: Three-dimensional principle component plot used to understand distribution of clusters created from DFIT data. This case shows the ‘complex’ DFIT model with its associated identified events using the K-means algorithm. This figure includes different perspectives of the data to understand variation.

Chapter 4

Supervised methods for DFIT event detection

4.1 Introduction

In efforts of scrutinizing the *CREWES DFIT Clustering app*, this chapter develops a method of supervised DFIT event detection. Referring back to the DFIT ML domains Venn diagram defined in Figure 1.9, this method for event detection falls in the location of the green star where only *ML* and *Visualization* domains overlap. This is because the end user will no longer have the ability to interactively adjust the method. Instead, a constant pre-trained model will be used. One of the key reasons for abandoning this interactivity is the inherent benefit of saving the end user time with an adequately trained model. For example, in the study by Yang and Ma (2019), after training a salt diapir velocity prediction model over 18 hours it only takes 2 seconds to predict the resulting velocity models for test data inputs. This aspect of time saving is enticing compared to the iterative process required at times with the *CREWES DFIT Clustering app*. In this chapter I will evaluate this method.

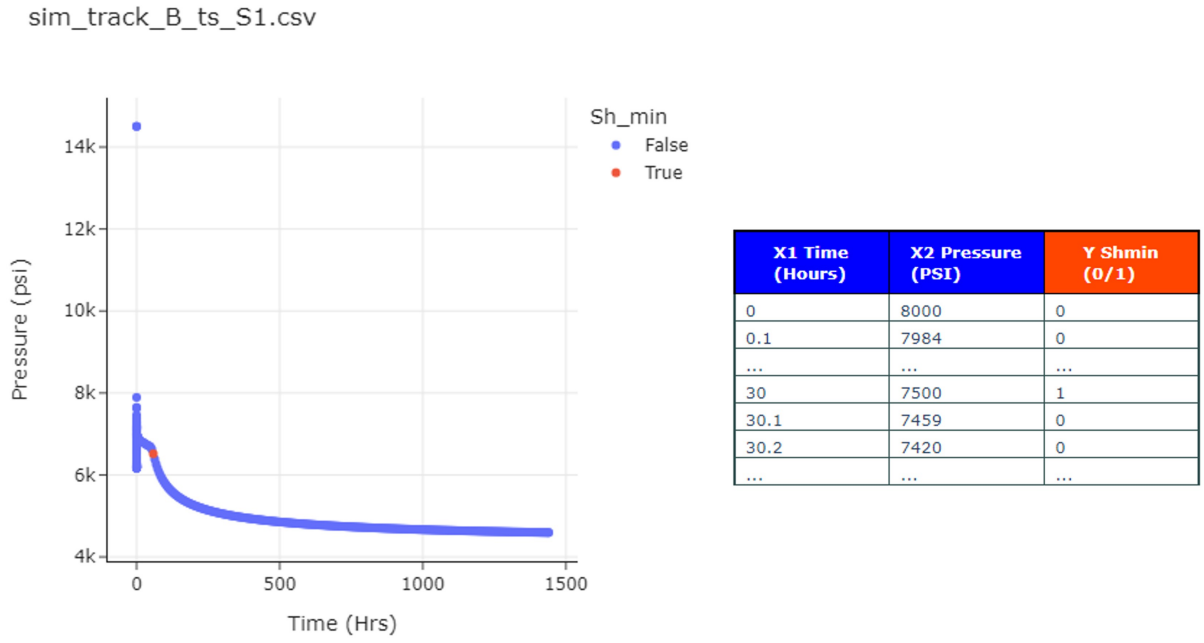
Supervised time series classification is a thoroughly studied problem applied in areas such as microseismic event detection (Othman et al., 2021), meat quality classification

(Ismail Fawaz et al., 2019), and cardiac arrhythmia (Savalia and Emamian, 2018). Within the supervised learning category multiple methods of data classification exist. In this chapter, three supervised algorithms are tested: Decision trees (Myles et al., 2004), Random decision forests (RF) (Ho, 1995), and Deep Neural Networks (DNN) (Cios, 2018). These techniques attempt to provide desired outputs (Y) for a given set of inputs (X). This is achieved by splitting a dataset into training and testing sets. The training set will allow the learning method to develop a model for classification and the testing set will act as a quality control measure to ensure the model has predictability for other datasets. Comparatively, by providing the answers to the learning process, this juxtaposes the unsupervised method tested in the previous chapter where the solutions were not provided.

The goal of designing these methods is to save the end user's time by eliminating the interactivity portion of the DFIT data science Venn diagram (Figure 1.9). Furthermore, this learning method aims to eliminate the human bias discussed in Chapter 1. Human bias will be eliminated through the development of complex no-linear relationships between the desired output and input as opposed to interpretive analytical methods introduced in Chapter 1. Fracture behaviour physics assumptions will also be eliminated as this method looks at only the pressure vs time data and not derivative plots. Collectively, this will allow for quick and accurate interpretation of Sh_{min} and therefore, efficient optimization of the hydraulic stimulation process to improve economics of de-risk play types.

4.2 Methods

As displayed in Chapter 1, identifying the location of events such as Sh_{min} is non-trivial when looking at a simple pressure vs time DFIT decline curve alone. Analytical methods addressed this adversity by looking at trends in the derivative curves, creating a more interpretable plot for the human eye. One of strengths of supervised methods is its ability to identify this hidden relationship without the need for the human interpreter to visualize multiple



X1 Time (Hours)	X2 Pressure (PSI)	Y Shmin (0/1)
0	8000	0
0.1	7984	0
...	...	...
30	7500	1
30.1	7459	0
30.2	7420	0
...	...	...

Figure 4.1: Example of a single input file for training supervised learning methods. The learning method will find a relationship between input x variables (X1 and X2) and the desired output classification (Y).

derivative plots.

To evaluate the supervised DFIT event detection method, the full DFIT catalogue of 28 curves described in Chapter 2 are implemented. Figure 4.1 displays a illustration of a single DFIT belonging to the 28 curve dataset. In this Figure, 'X' consists of the time series of pressure and time samples from the pressure decline (Figure 4.1). The 'Y' is then a boolean output of True or False indicating the desired event label at the point in the time series, this is Sh_{min} in Figure 4.1. The goal of the applied supervised learning technique will be to derive the non-linear relationship between the input 'X' and desired output 'Y' (Müller and Guido, 2016). For the purpose of this study and to simplify the predicting problem to a single dimension, the predicted output will remain as the single key parameter Sh_{min}.

To avoid overfitting and therefore, generalize the learning model for this system, 24 curves of similar format to Figure 4.1 are applied to the training of the classification models. As

described in Chapter 2, each of the 24 curves input into the training process contains varying geologic parameters and reservoir conditions. An example of the adversities of overfitting can be found in the publication from Hu et al. (2021) where a convolutional neural network (CNN) is trained using only seismic data from regions of extensive salt diapirism. This resulted in the model classifying areas of varying geologic characteristics, such as structure, incorrectly as salt diapirs. The split of data into 24 labelled training curves and four testing curves closely follows the supervised rule of data splitting 80% to 20% (Géron, 2019). The four testing curves belong to the Duvernay Formation subset. For consistency in file naming of the larger dataset, .csv file names are displayed on the titles of resulting classification plots. Based on these file names 01_ResFrac_Demo_format_simple_Duvernay.csv coincides with the 'simple' 3-layer geologic simulation, 02_Duvernay_format_31_layers.csv is the 'complex' 31 geologic layer simulation, 03_Duvernay_DFIT_Formatted_LS_Oct2021_Field.csv is a field DFIT dataset and 04_Synthetic_4_Jan_2022.csv is an additional 'simple' 3-layer geologic simulation with higher Sh_{min} conditions. This will be referred to as the second 'simple' model. A full description of the parameters used to generate testing and training datasets is described in Appendix Figure A.1.

The following sub sections will describe the two methods of data formatting for (1) applying Decision trees (Myles et al., 2004) and RF classification (Ho, 1995) and (2) DNN (Cios, 2018). Implementation of these methods was achieved using *Python®* on *Google Colab* and *Jupyter Notebooks*. The *Keras®* machine learning library was applied for supervised methods. Application of platforms such as *Google Colab* gave access to high powered Graphics processing units (GPU).

4.2.1 (1) Decision trees and Random decision forest trees (RF)

Decision trees and RF both fall into the classification regime of ML. The historical method of using Decision trees can be thought of as an algorithm that finds optimal splits in the categorical data to minimize the systems entropy (Myles et al., 2004; Géron, 2019). This will

create a single tree of conditions that categorize the elements of the system. This method was selected as a baseline classification method for the DFIT classification problem.

RF can be thought of as an updated version of the Decision tree, where multiple decision trees are merged and used to classify 'N' data points from the system of interest. The average of classification given by this process is the overall solution (Ho, 1995; Géron, 2019). One of the benefits of using this method over Decision trees is it ability to generalize the solution and avoid overfitting to one solution. This is achieved by averaging the multiple solutions produced by the Decision trees also known as ensemble learning (Ho, 1995; Géron, 2019; Chollet, 2018). This method was selected as a comparison to the basic Decision tree algorithm to assess potential improvements. In this study, the *n_estimators* hyperparameter (number of trees) was adjusted accordingly to optimize results.

The benefit of using Decision and RF tree classification is that the input data length for any curve can be of varying size. This allows for the 'time' (X1) and 'pressure' (X2) columns from the 24 training sets to be simply stacked into a list size of (13495, 2) with their corresponding Sh_{min} labels being stacked into a list of size (13495, 1). The result of plotting this data is displayed in Figure 4.2. Using this formatted data a classifier model is created. Performance of the classification model is then evaluated by comparing the solution to the predicted output.

4.2.2 (2) Deep Neural Networks (DNN)

The application of DNN to this classification problem is hypothesized to create a more flexible and computationally quick Sh_{min} prediction model. Derived from Artificial Neural Networks (ANN), DNN offers the ability to solve non-linear problems (Géron, 2019; Chollet, 2018). DNNs are composed of multiple layers of neurons allowing for classification of these problems (Figure 4.3). The neurons within each layer of (Figure 4.3) can also be referred to as threshold logic units (TLU). These TLUs essentially take in a weighted sum of the inputs and pass information forward if the activation function is satisfied (Géron, 2019). The final layer of

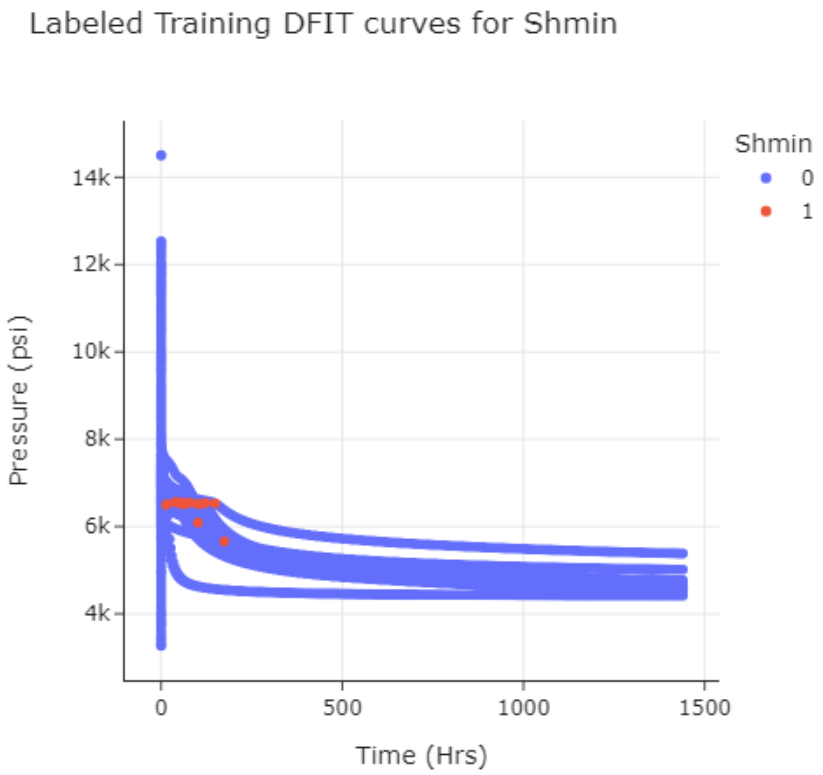


Figure 4.2: Plot of the collective labelled training sets to be input into the Decision tree and RF classifiers.

the DNN will contain the output of nodes equal to the number of classifications required in the system. The final classification will then be compared to the 'ground truth' label of the training set, weights will be updated based on the classification error in a process known as back propagation (Géron, 2019). Different activation functions can be selected to optimize a problem. Some examples include the sigmoid function and rectified linear unit function (ReLU) (Chollet, 2018). In this study, the ReLU activation was applied due to its faster computation time (Géron, 2019).

Figure 4.3 demonstrated how the DNN will be applied to the training set of 24 DFIT curves with varying parameters. In this figure, the input layer will accept the time (X1) and associated pressure values (X2) in the input layer for each curve as separate 'mini batches'. This is achieved by formatting the dataset into a 3D dataframe of columns X1, X2 and the associated Sh_{min} labels (Y). In this formatting, the third dimension is defined as the different DFIT curves generated for this experiment. Notably, this data formatting is more complex compared to the tree classification methods where only 2-dimensions are required and DFIT curve separation is not.

An additional complexity of this method when compared to tree classification is that the input size of each of the curves must be the same length. As displayed in Figure 4.4 the number of data points contained in each training set varies. This also applies for the testing dataset in Figure 4.7. Based on this observation, all datasets need to be truncated to the shortest dataset, containing 466 points. Notably, the field dataset could not be cut to the first 466 points, instead a windowed area had to be selected. This is a result of a varied sampling interval on the field dataset to save memory space.

The resulting training dataframe will contain 24 curves, 466 points per variable, with the two variables being time and pressure. This results in a dataframe of size 24,466,2. A matching dataframe of Sh_{min} labels is also created of size (24,466,1). The resulting plot of labelled training datasets is displayed in Figure 4.6. Two of these 24 training curves were used as a validation set to test for generality during the learning process. The testing dataset

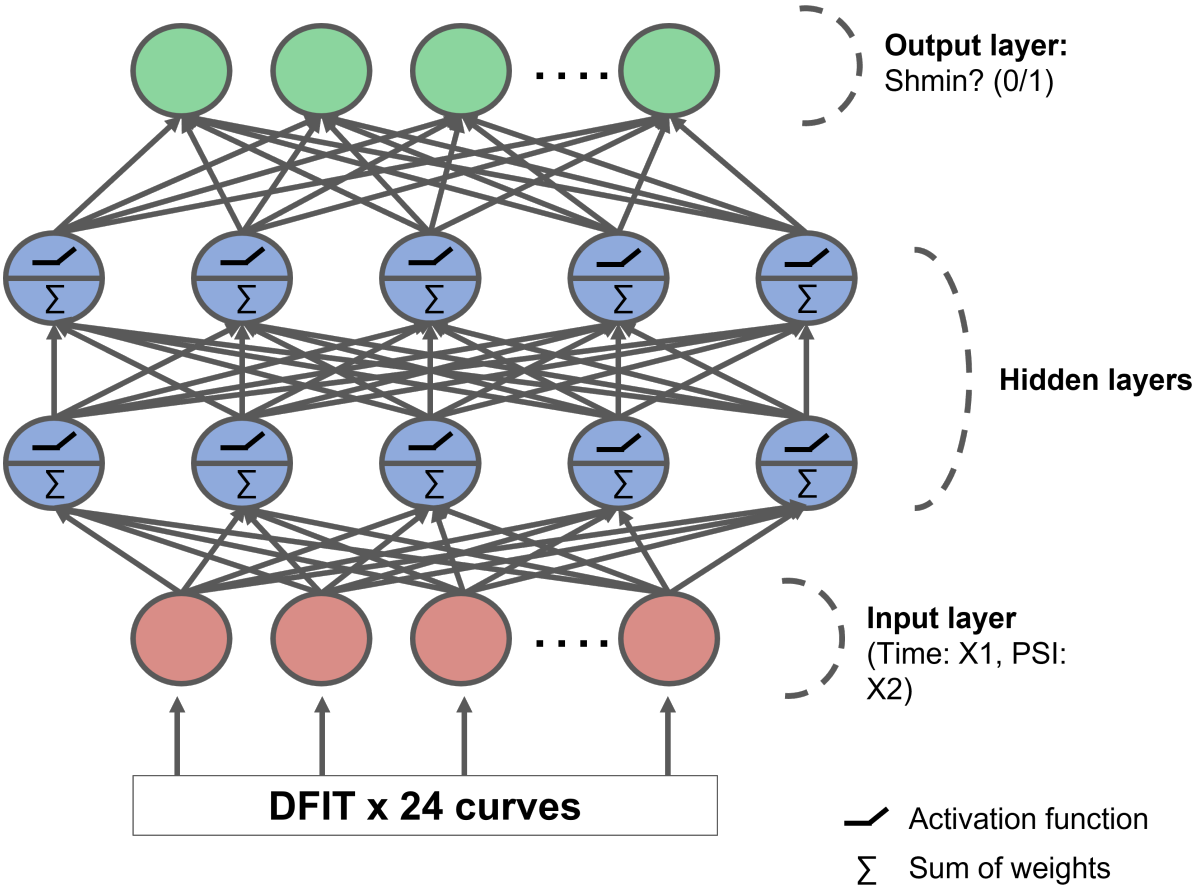


Figure 4.3: Diagram of the DNN structure applied for DFIT classification.

will be of size (4,466,2) with a matching labels data frame of (4,466,1). A resulting plot of labelled testing datasets from this process is found in Figure 4.7. It can be observed that the field case (03_Duvernay_DFIT) and 31 layer Duvernay model (02_Duvernay) appear to have the least spatial variation in data trend after limiting the number of samples to 466. This will be used as a test to determine the influence of missing curve segments on the train supervised model classifier. Both the testing and training data frames were scaled before input into the learning methods to account for different scales of measure.

When constructing the DNN model in Figure 4.3, multiple parameters must be optimized. Starting with the input and output, 466 points must be presented in both cases (Figure 4.3). These consist of pressure and time variables for input and output probability of a Sh_{min}

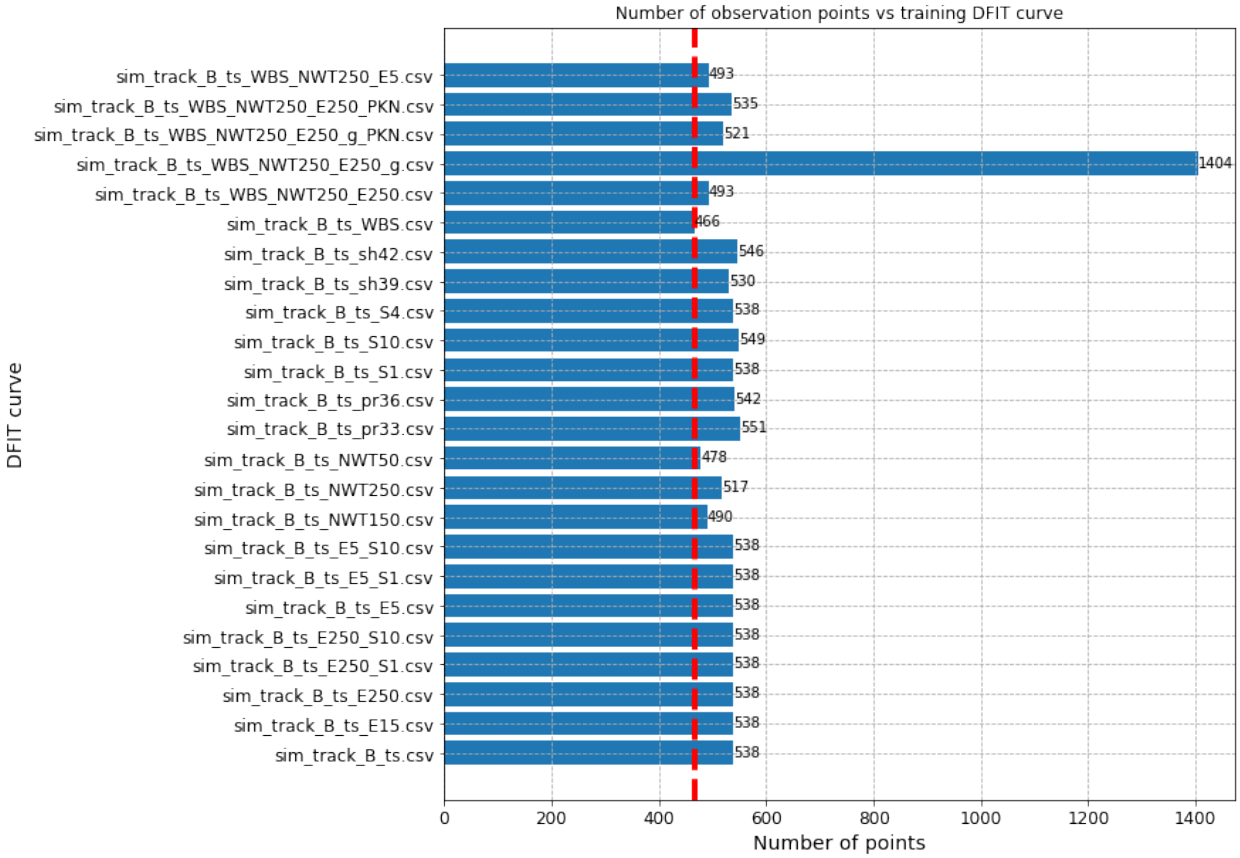


Figure 4.4: Number of points per training dataset, red dashed line shows the cut off applied to data to make input numbers uniform for the DNN.

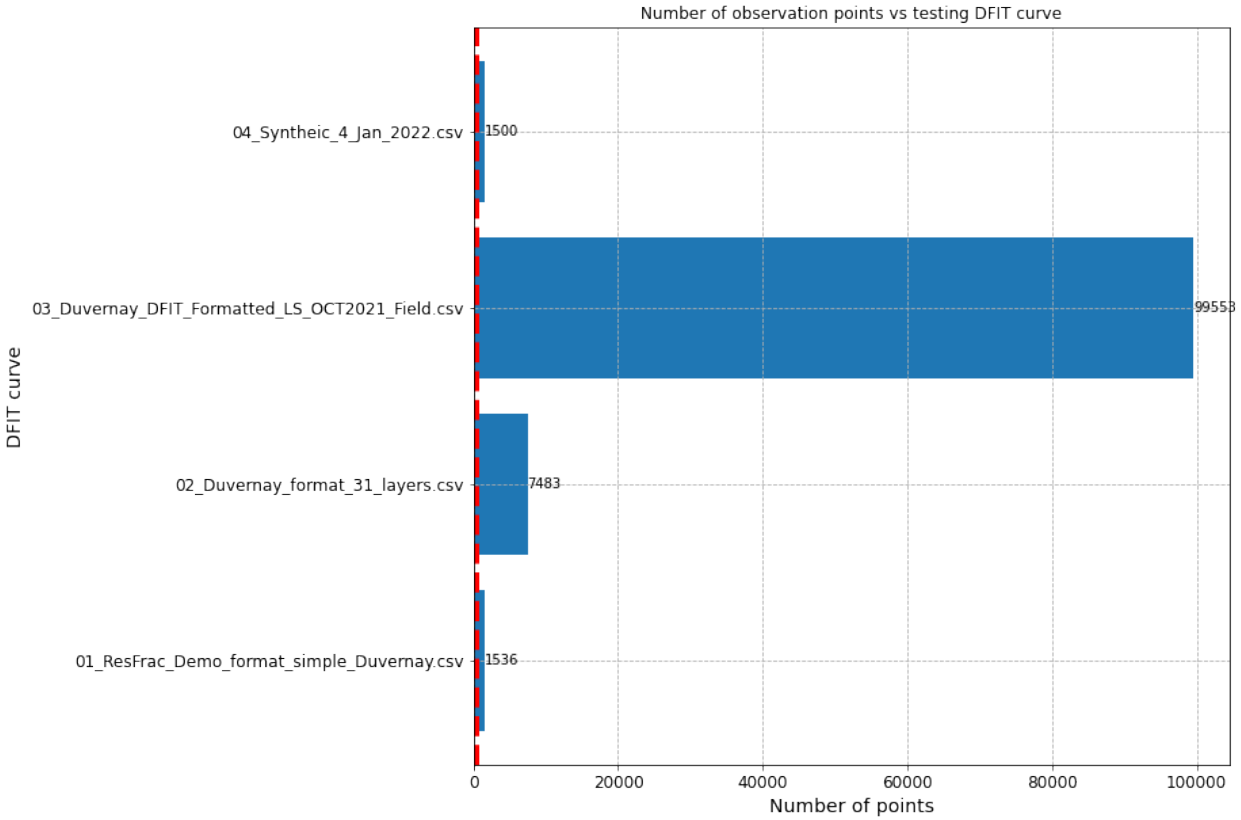


Figure 4.5: Number of points per testing dataset, red dashed line shows the cut off applied to data to make input numbers uniform for the DNN.

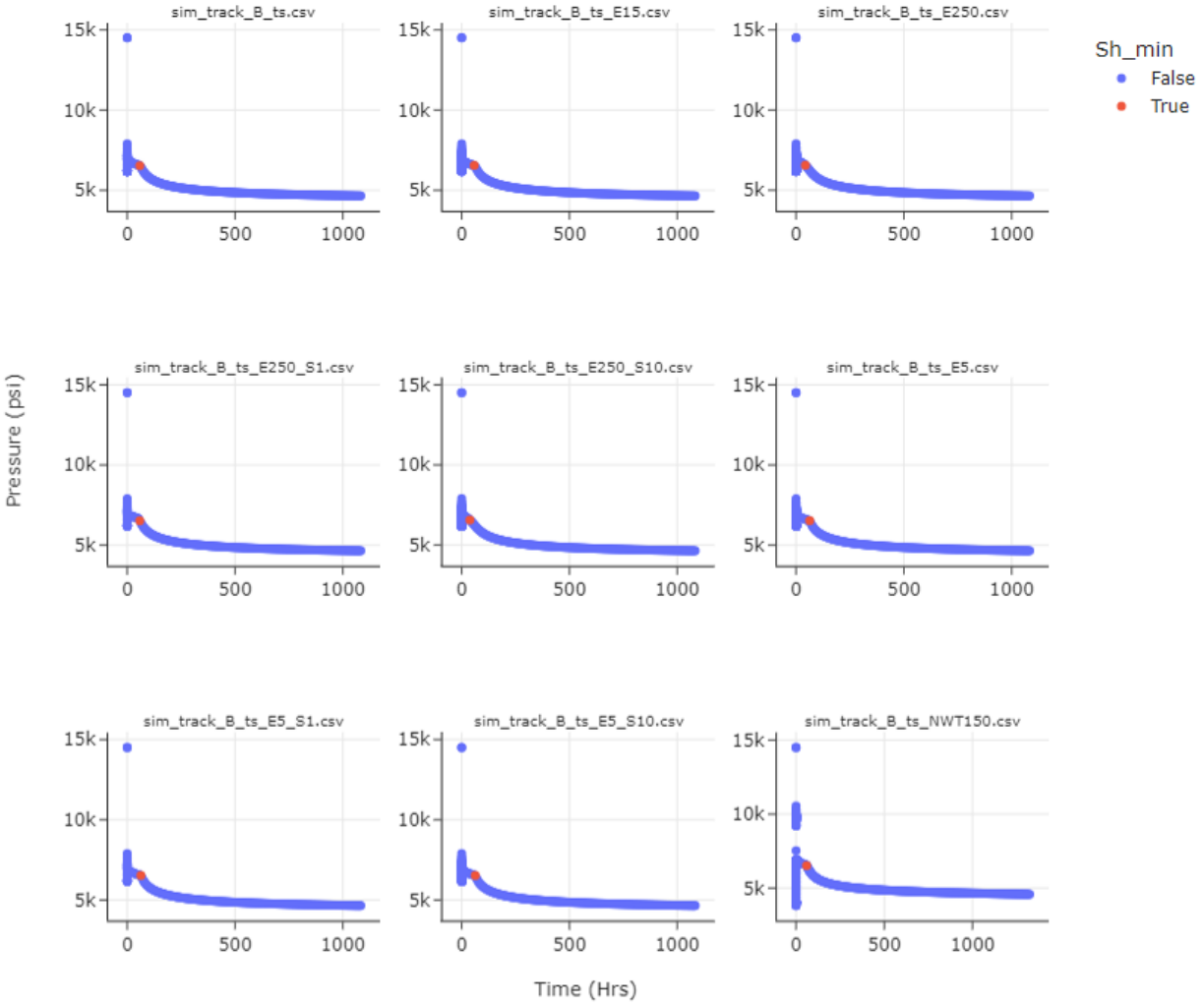


Figure 4.6: Subset of the 24 training curves formatted to 466 points with Sh_{min} labels (red).

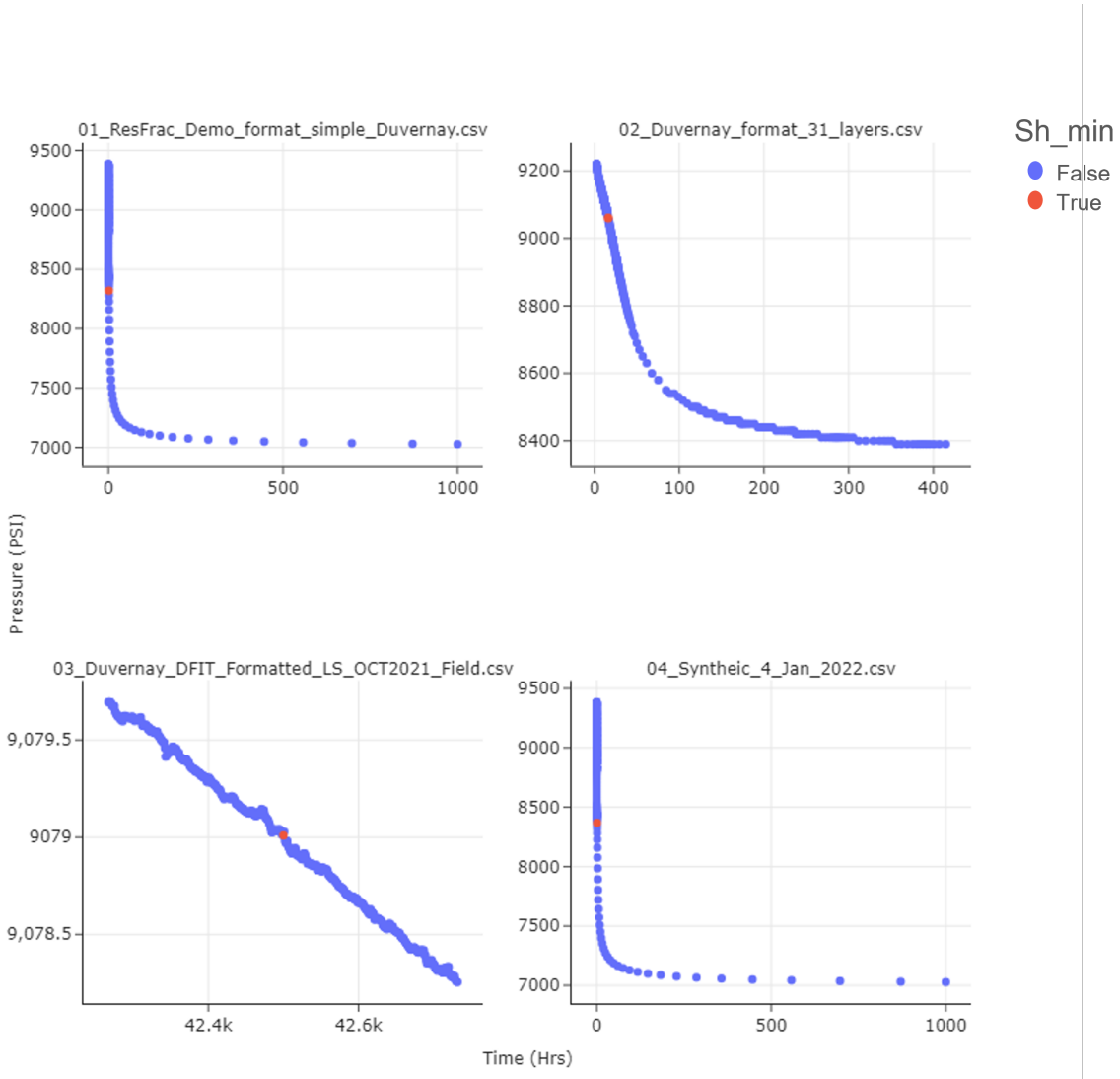


Figure 4.7: Four testing curves formatted to 466 points with Sh_{min} labels (red).

classification. These input and output neurons must remain a constant size.

Within the 'black box' hidden layer framework of the DNN, multiple hyperparameters also need to be optimized (Figure 4.3). These include: hidden layer activation functions, output layer activation functions, number of hidden layers, number of hidden neurons per hidden layer, loss functions, optimizer for back propagation, number of epochs. As discussed earlier, the ReLU activation function is applied in this experiment for computational ease. A sigmoid function was used for the output layer for ease of interpretability, placing probability values between 0 and 1.

To understand the effects of varying hidden layers, each classification experiment is repeated using one hidden layer, followed by five, and ten. Deeper neural networks (more hidden layers) are hypothesized to handle complex problems with fewer neurons per layer (Géron, 2019). This will be tested by decreasing the neuron count per layer for each increase in the number of hidden layers. More neurons per hidden layer can aid in the classification of complex problems, however, a trade off in the system must be made to avoid overfitting. Two rules of thumb are applied for determining the number of neurons in each of these cases:

(1) The number of neurons are 2/3 of the input layer or 2/3 of the number of input and output neurons added (Boger and Guterman, 1997).

(2) The number of hidden layer neurons should be less than twice the number of neurons in the input layer (Berry and Linoff, 2004).

Using these guidelines, the three DNN learning models are created in Figures 4.8, 4.9, 4.10. An additional model is created explicitly not following these guidelines to test their grounds in Figure 4.11. In these figures, the shape of each layer is described (Output shape) and the number of trainable parameters for each layer is also included (# Param). The summed trainable parameters is included at the bottom of each plot. These models use

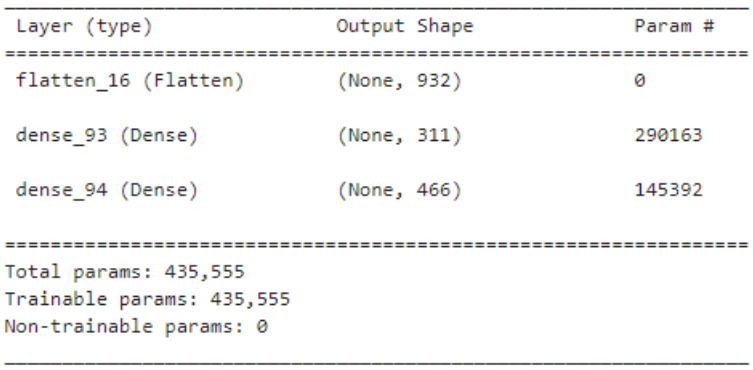


Figure 4.8: DFIT classification DNN with 1 hidden layer.

categorical cross entropy (Good, 1992) for calculating loss and stochastic gradient descent for back propagation (SGD) (Kiefer and Wolfowitz, 1952).

Learning curves are used for the determination of the optimal number of epochs to train the system. These curves display the loss for the validation and training datasets. An accuracy score is also included for each model validation/training epochs. Collectively these metrics aided in the interpretations of the varied learning models.

To conclude this methods section, Figure 4.12 shows a summary dataflow diagram for method (1) and (2). In Figure 4.12 the end user would not have access to the 'supervised method calibration' portion of the pseudo algorithm, instead the user would only be using the output model for event prediction in the 'supervised user interface' pseudo algorithm. This shifts the paradigm away from analytical interaction and instead, emphasis is put on proper data formatting for input into the predictor model. Arguably, this is a 'black box' solution and may have its adversities when the model is not generalized. At the same time, the aspect of abandoning 'Analytical interaction' in the DFIT Venn diagram (Figure 1.9) has the hypothesized advantage of saving the end users time. These adversities and benefits will be discussed in the following sections.

Layer (type)	Output Shape	Param #
flatten_17 (Flatten)	(None, 932)	0
dense_95 (Dense)	(None, 400)	373200
dense_96 (Dense)	(None, 250)	100250
dense_97 (Dense)	(None, 100)	25100
dense_98 (Dense)	(None, 100)	10100
dense_99 (Dense)	(None, 82)	8282
dense_100 (Dense)	(None, 466)	38678

Total params: 555,610
Trainable params: 555,610
Non-trainable params: 0

Figure 4.9: DFIT classification DNN with 5 hidden layers.

Layer (type)	Output Shape	Param #
flatten_18 (Flatten)	(None, 932)	0
dense_101 (Dense)	(None, 400)	373200
dense_102 (Dense)	(None, 150)	60150
dense_103 (Dense)	(None, 125)	18875
dense_104 (Dense)	(None, 50)	6300
dense_105 (Dense)	(None, 50)	2550
dense_106 (Dense)	(None, 50)	2550
dense_107 (Dense)	(None, 40)	2040
dense_108 (Dense)	(None, 40)	1640
dense_109 (Dense)	(None, 25)	1025
dense_110 (Dense)	(None, 10)	260
dense_111 (Dense)	(None, 466)	5126

Total params: 473,716
Trainable params: 473,716
Non-trainable params: 0

Figure 4.10: DFIT classification DNN with 10 hidden layers.

Layer (type)	Output Shape	Param #
flatten_19 (Flatten)	(None, 932)	0
dense_112 (Dense)	(None, 400)	373200
dense_113 (Dense)	(None, 300)	120300
dense_114 (Dense)	(None, 40)	12040
dense_115 (Dense)	(None, 200)	8200
dense_116 (Dense)	(None, 466)	93666

Total params: 607,406
Trainable params: 607,406
Non-trainable params: 0

Figure 4.11: DFIT classification DNN with 5 hidden layers and more neurons than input and output combined. This model also has the most trainable parameters.

4.3 Results

4.3.1 (1) Decision Tree and RF Classification

Decision tree and RF classification methods were the first supervised methods applied to the DFIT Sh_{min} classification problem. The training set of 24 synthetic DFIT curves where merged into a flattened vector of 13496 points per variable (PSI and hours) and input into the respective learning method.

The Decision tree learning method was selected as the historical baseline learning method to be first applied to this classification task. Figure 4.13 displays a magnification of the training dataset with the classification model overlain. In this figure, the green shaded areas demarcate where Sh_{min} would be classified as a point on future input DFIT curves. Green points represent locations of known Sh_{min} used to train the model. This figure also includes a demagnified subplot of the total data classification on the upper right corner. The result of applying this prediction model to the four testing curves is displayed in Figure 4.15, classified Sh_{min} points are demarcated by red circles. Only one classification was made on the field dataset with a value of 9112psi, this value will be compared with the expected values of

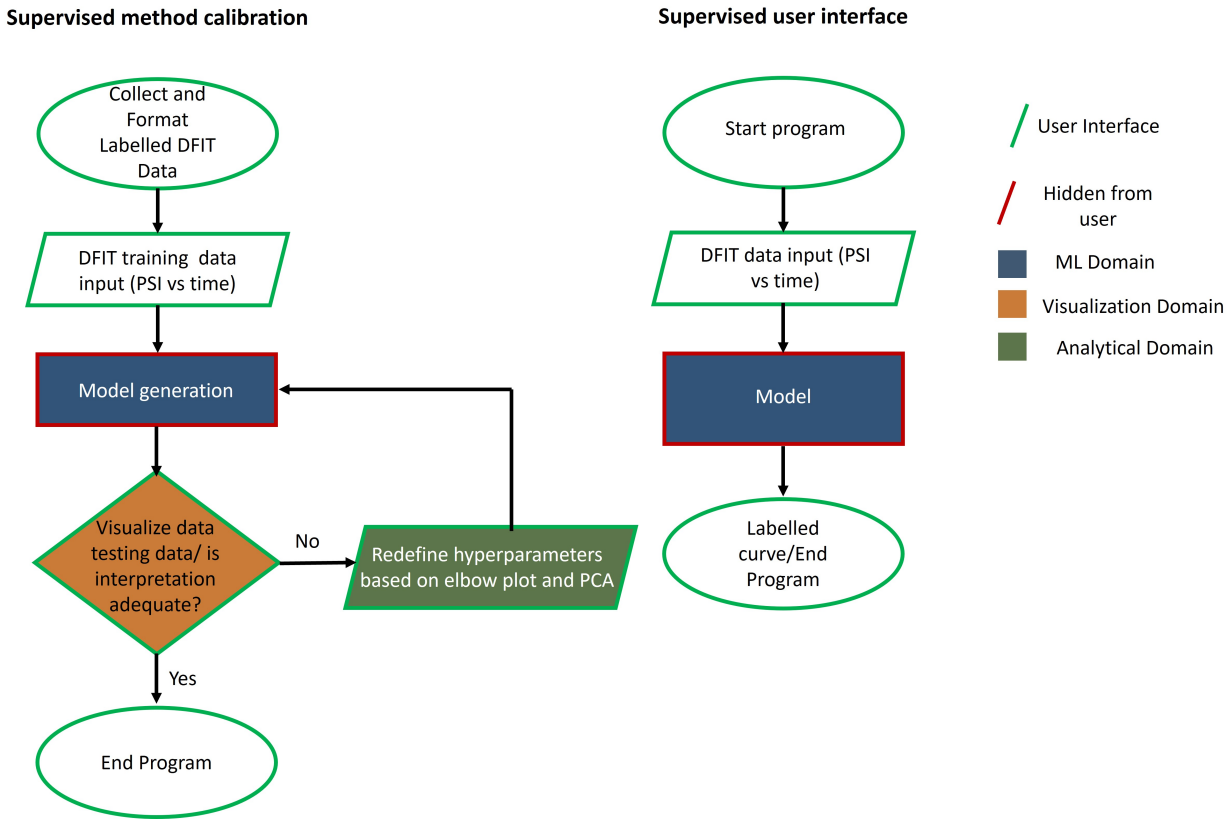


Figure 4.12: Dataflow diagram for both supervised methods (1) and (2) applied to the DFIT event identification problem. Green outlines indicate user interface while red outlines indicates features hidden from the user. Operations are also color coded based on their relationship to the data scientist domains in Figure 1.9.

Sh_{min} for the testing dataset displayed in Figure C.1 of the appendix.

The modified method of Decision tree classification; RF classification is the second supervised learning method tested in this experiment. Resulting model classification area is displayed in Figure 4.14. Classifications of the testing dataset are displayed in Figure 4.16. There are no classifications made on the testing datasets by this method.

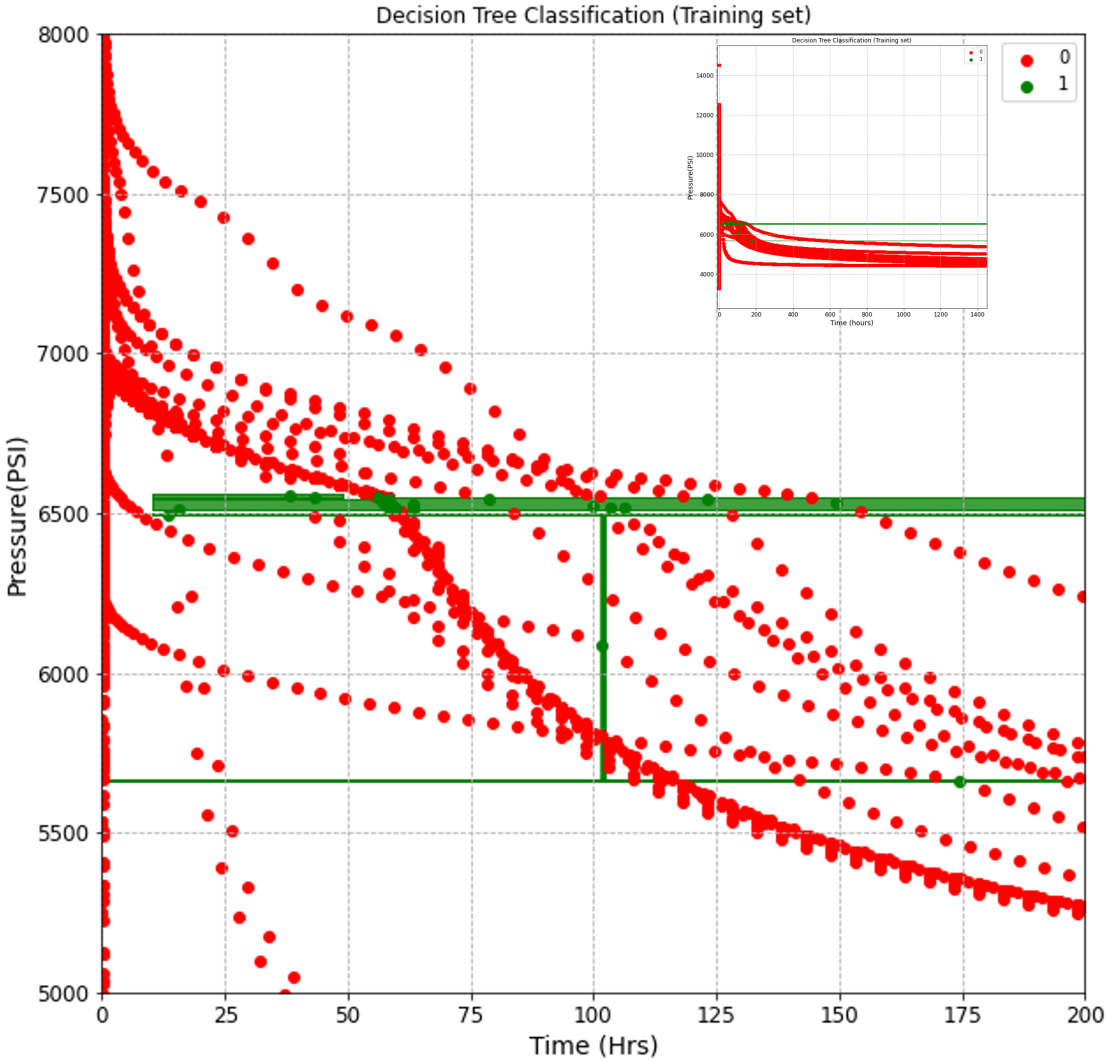


Figure 4.13: DFIT training data with the Decision tree classification model area for Sh_{min} overlain. Green points represent areas of known Sh_{min} occurrences and red points where this is False. The green shaded area represents the final classification model area where future predicted curves would attain Sh_{min} classification.

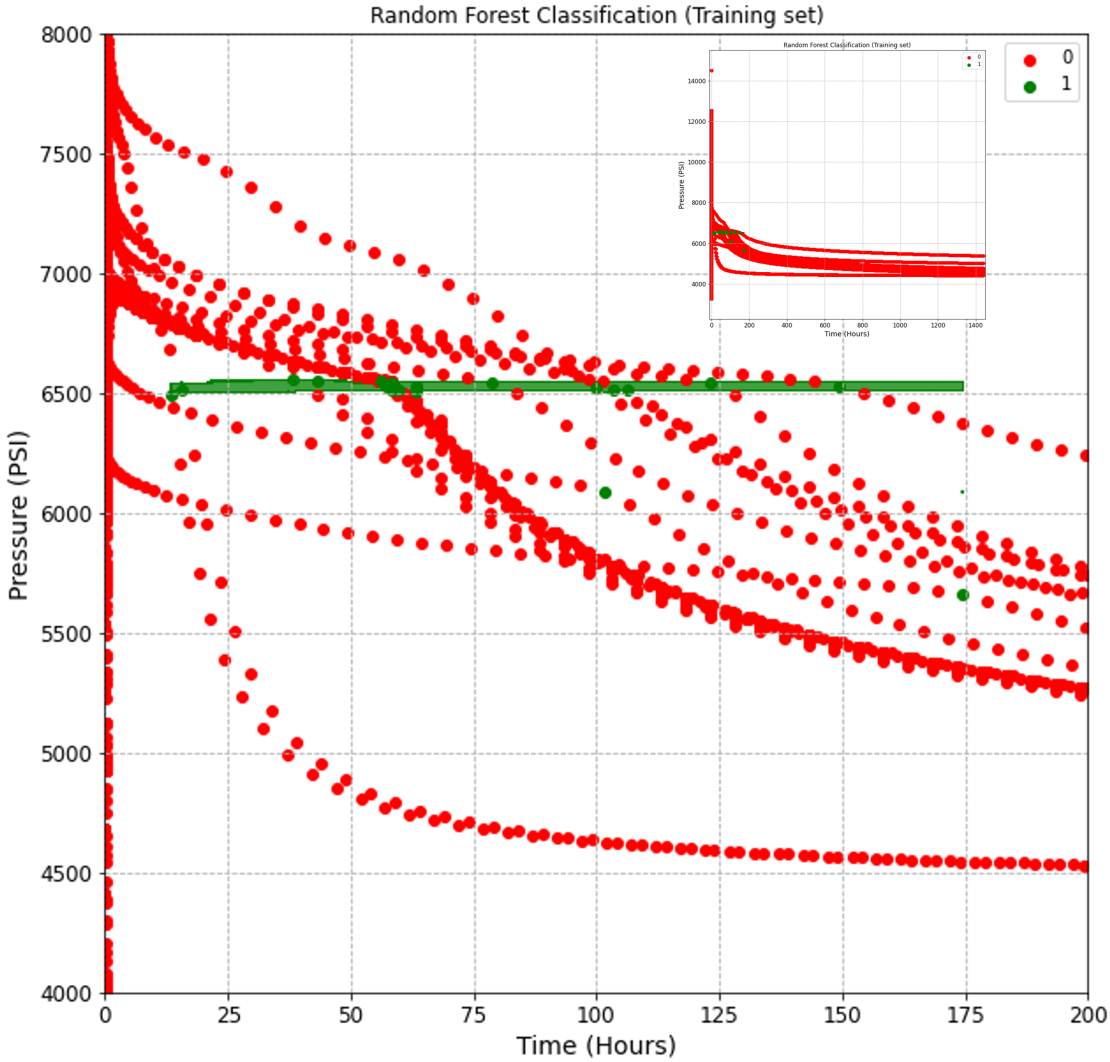


Figure 4.14: DFIT training data with the RF classification model area for Sh_{min} overlain. Green points represent areas of known Sh_{min} occurrences and red points where this is False. The green shaded area represents the final classification model area where future predicted curves would attain Sh_{min} classification.

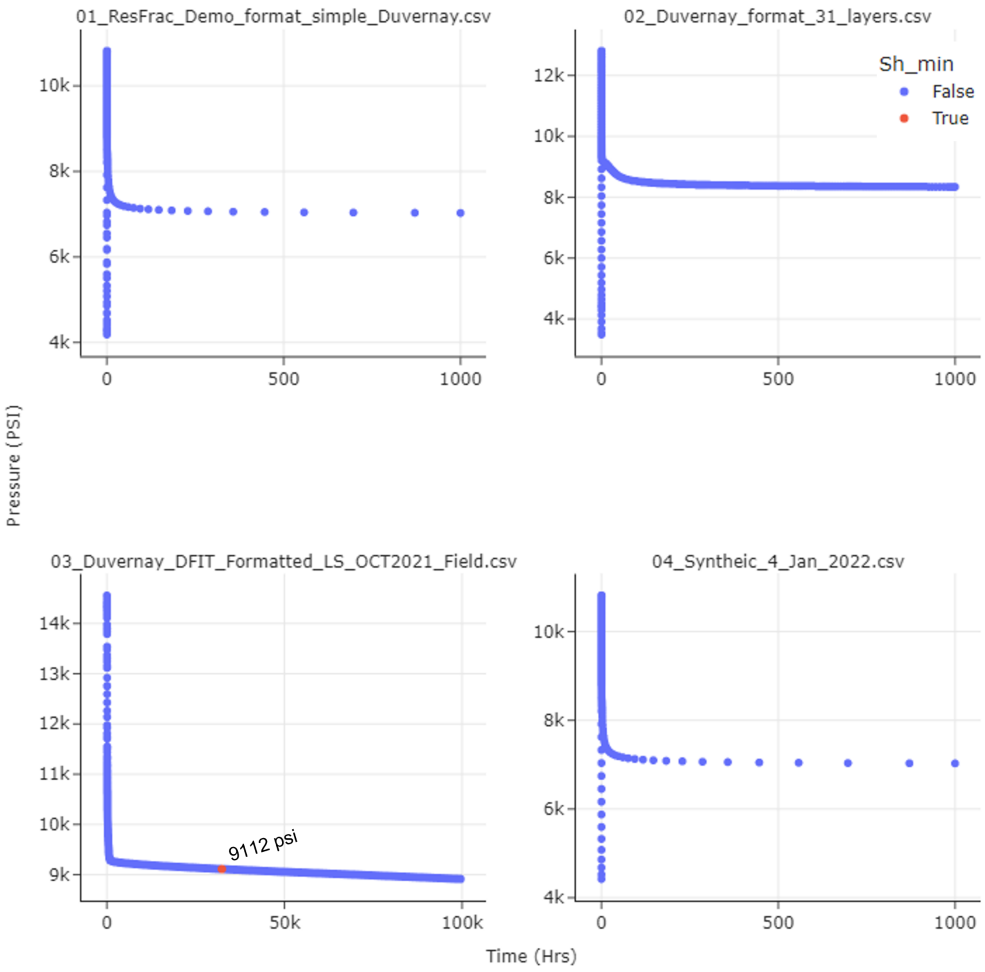


Figure 4.15: Resulting classifications (red) of Sh_{min} by the Decision tree model on the four testing datasets.

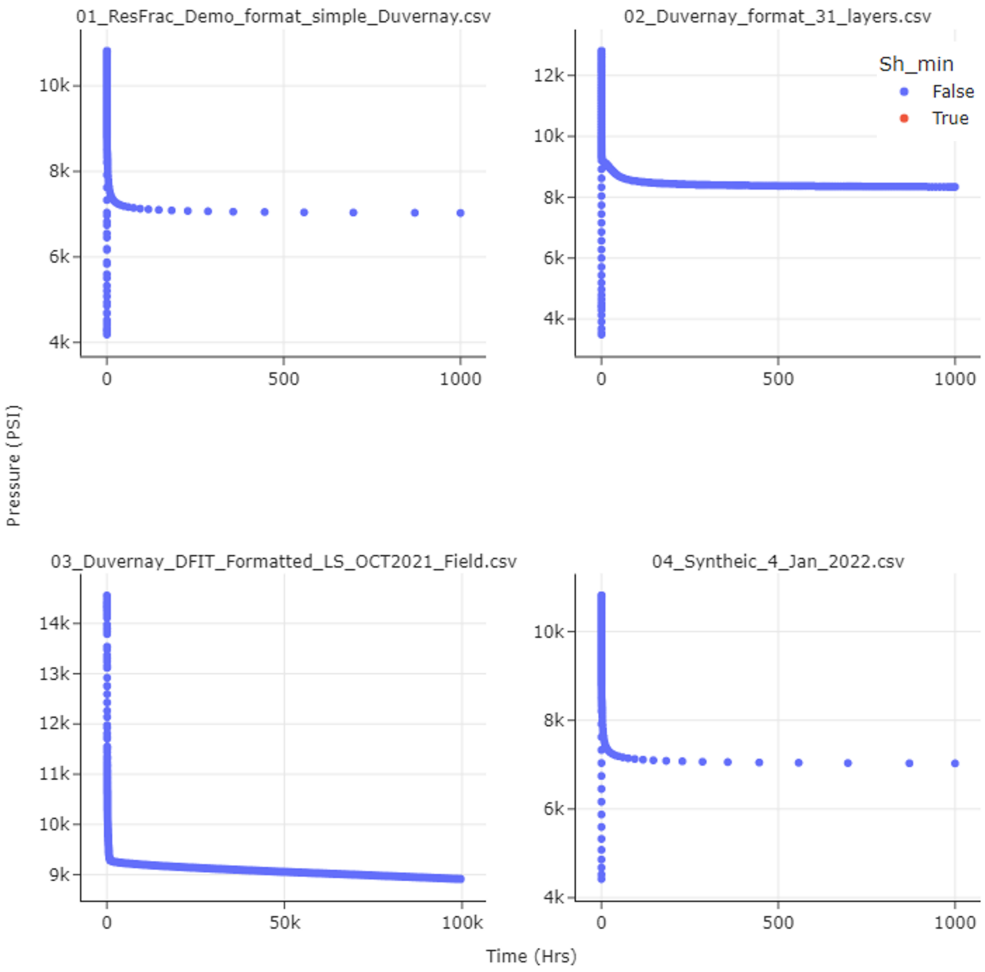


Figure 4.16: Resulting classifications (red) of Sh_{min} by the RF tree model on the four testing datasets.

4.3.2 (2) DNN Classification

The application of DNN to the Sh_{min} classification problem is the second supervised learning method applied. The training set of 24 synthetic DFIT curves were formatted into a three-dimensional data frame (24,466,2) for training and validating DNN model architectures. These included the single layer model (Model 1), five layer model (Model 2), ten layer model (Model 3) and the five layer model with additional neurons (Model 4). Models were generated using the Intel Core i7-10750H CPU. The summary of the number of training parameters and total time taken to train the model is displayed in Table 4.1. Complimentary to this, the learning curves for each model are also displayed in Figures 4.17, 4.18, 4.19, and 4.20. In these figures the loss and accuracy score for the training and validation sets is displayed converging to 0 and 1, respectively. Each model was run with 1000 epochs to ensure convergence.

Resulting classifications made on the four testing Duvernay curves is displayed in Figures 4.21, 4.22, 4.23, 4.24, and 4.25. These figures display a continuous attribute of probability of Sh_{min} occurring as a given point on the DFIT pressure decline from the applied DNN model. Figure 4.21 displays and example of model 1 results where all probabilities are included. To improve interpretability, all following plots were filtered back to only show classifications that had an above 80% (0.8) probability. These plots are also labeled with the highest probability classification output by the model (green box) and its respective error to the actual value. In cases where the highest probability value had high error (above 1%), the second highest probability classification was also included (yellow box). Lastly, Figure 4.26 displays the number of potential Sh_{min} classifications produced by each model on each curve after the percent cut off was applied.

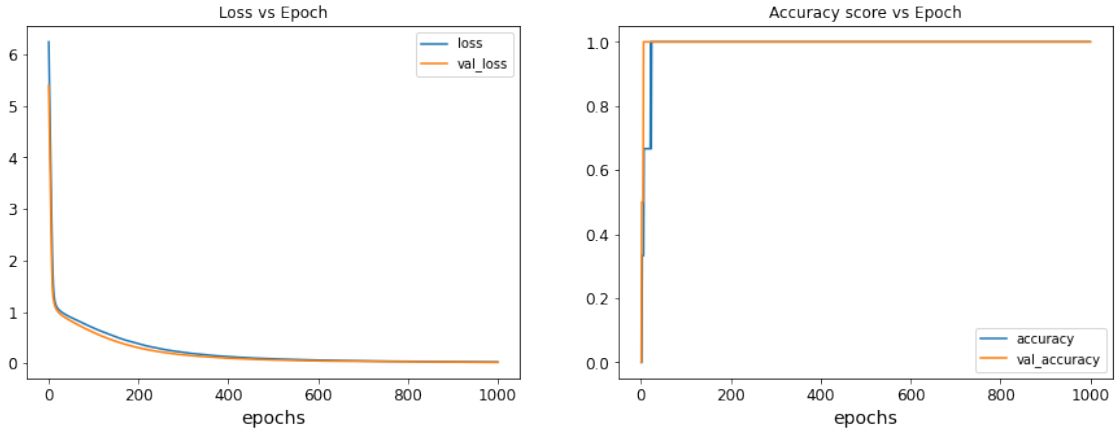


Figure 4.17: Loss vs epochs and accuracy score vs epochs for model 1. Blue and orange lines represent the training loss and validation loss, respectively.

Table 4.1: Model training times

# Model	Number of parameters	Time (seconds)
Single layer	435,555	18.9sec
Five layer	555,610	19.2sec
Ten layer	473,716	19.7sec
Five layer additional neurons	607,406	20sec

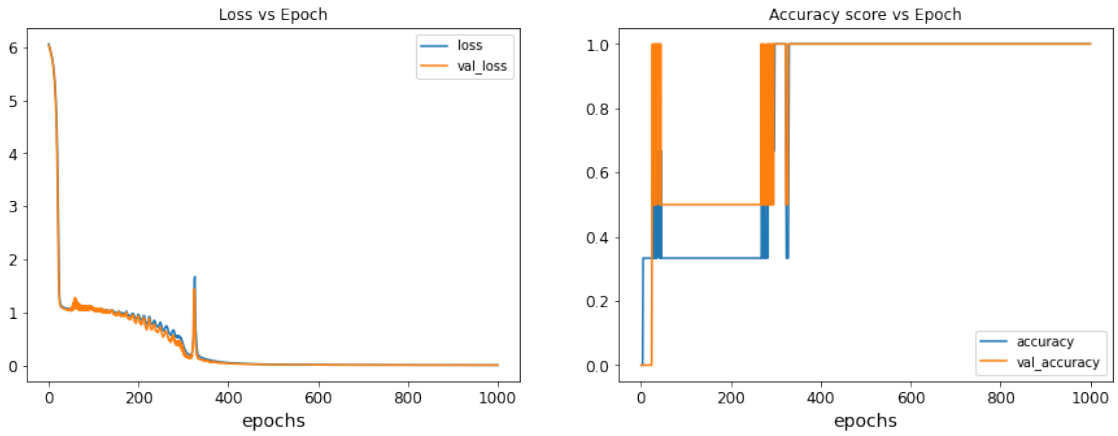


Figure 4.18: Loss vs epochs and accuracy score vs epochs for model 2. Blue and orange lines represent the training loss and validation loss, respectively.

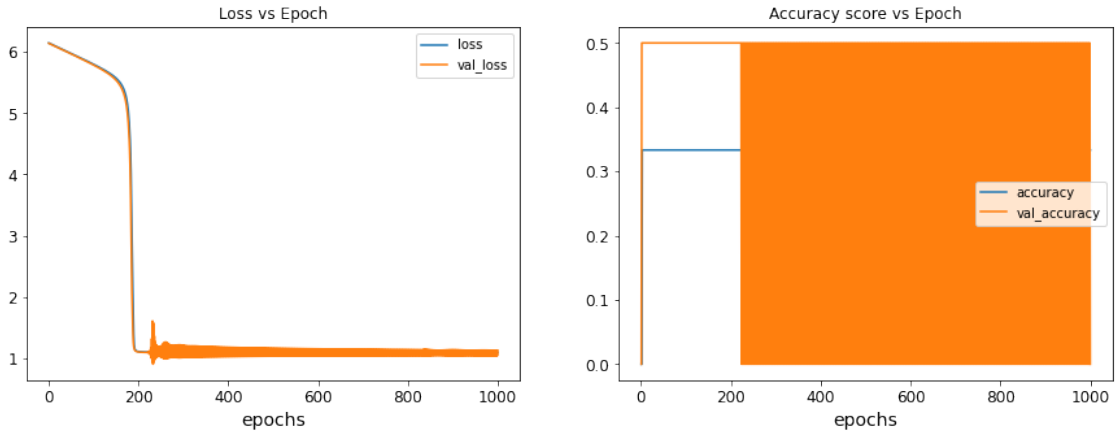


Figure 4.19: Loss vs epochs and accuracy score vs epochs for model 3. Blue and orange lines represent the training loss and validation loss, respectively.

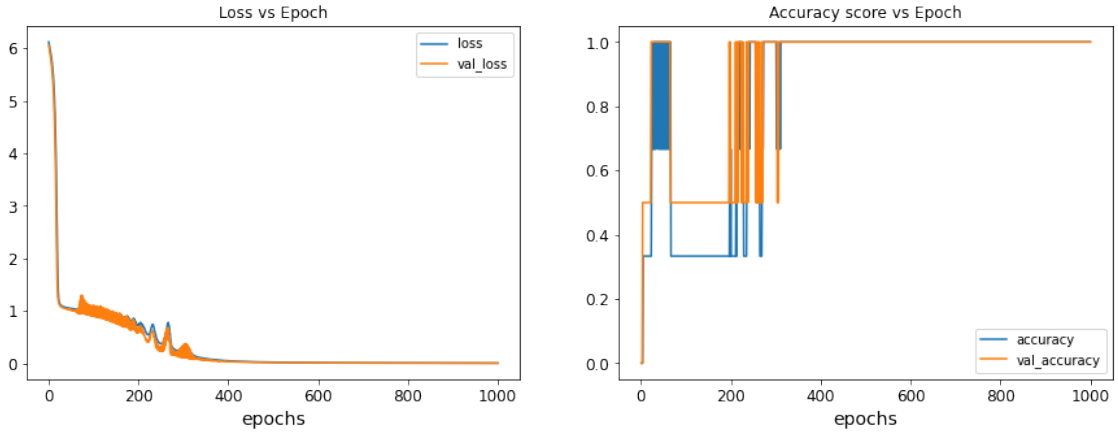


Figure 4.20: Loss vs epochs and accuracy score vs epochs for model 4. Blue and orange lines represent the training loss and validation loss, respectively.

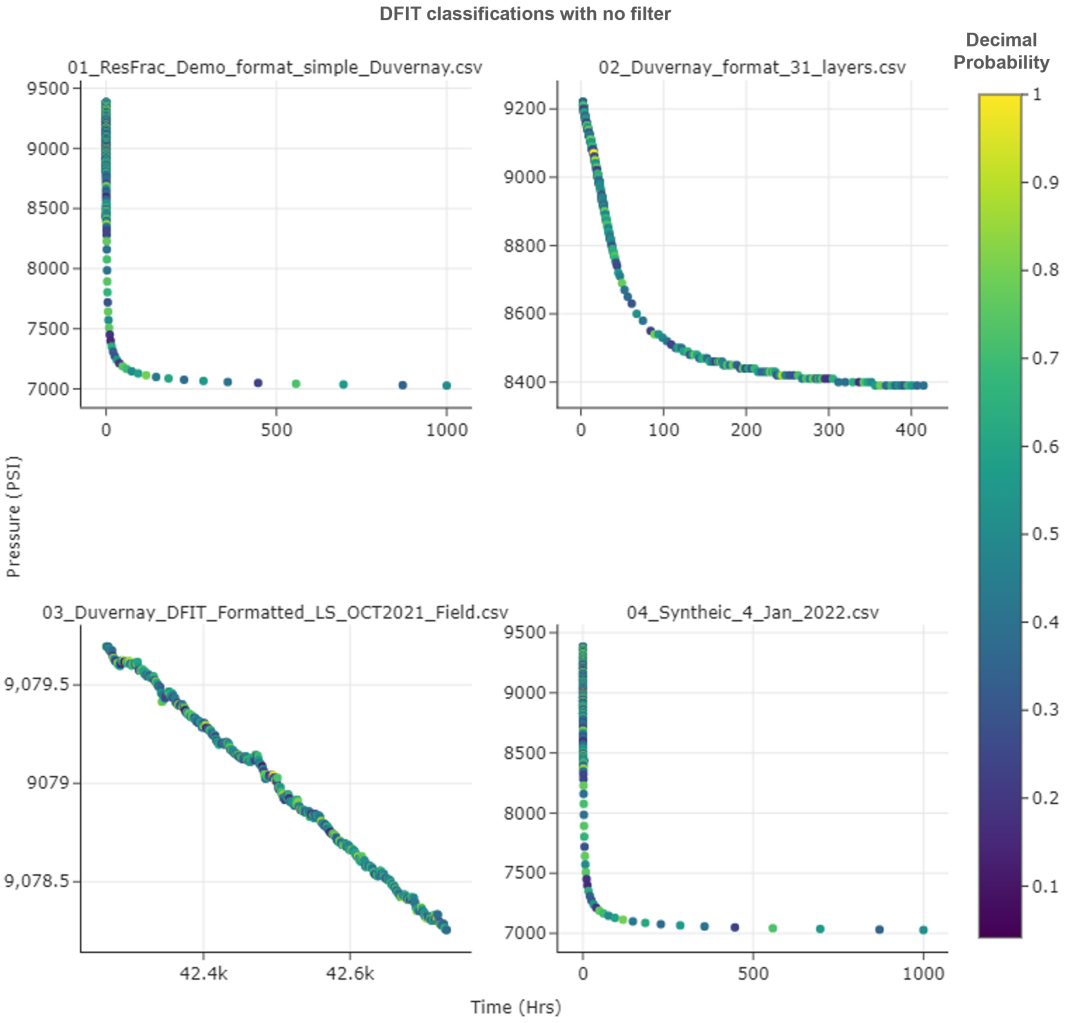


Figure 4.21: Resulting Sh_{min} classification probabilities from DNN model 1 as a continuous attribute superimposed on the Duvernay testing curve dataset.

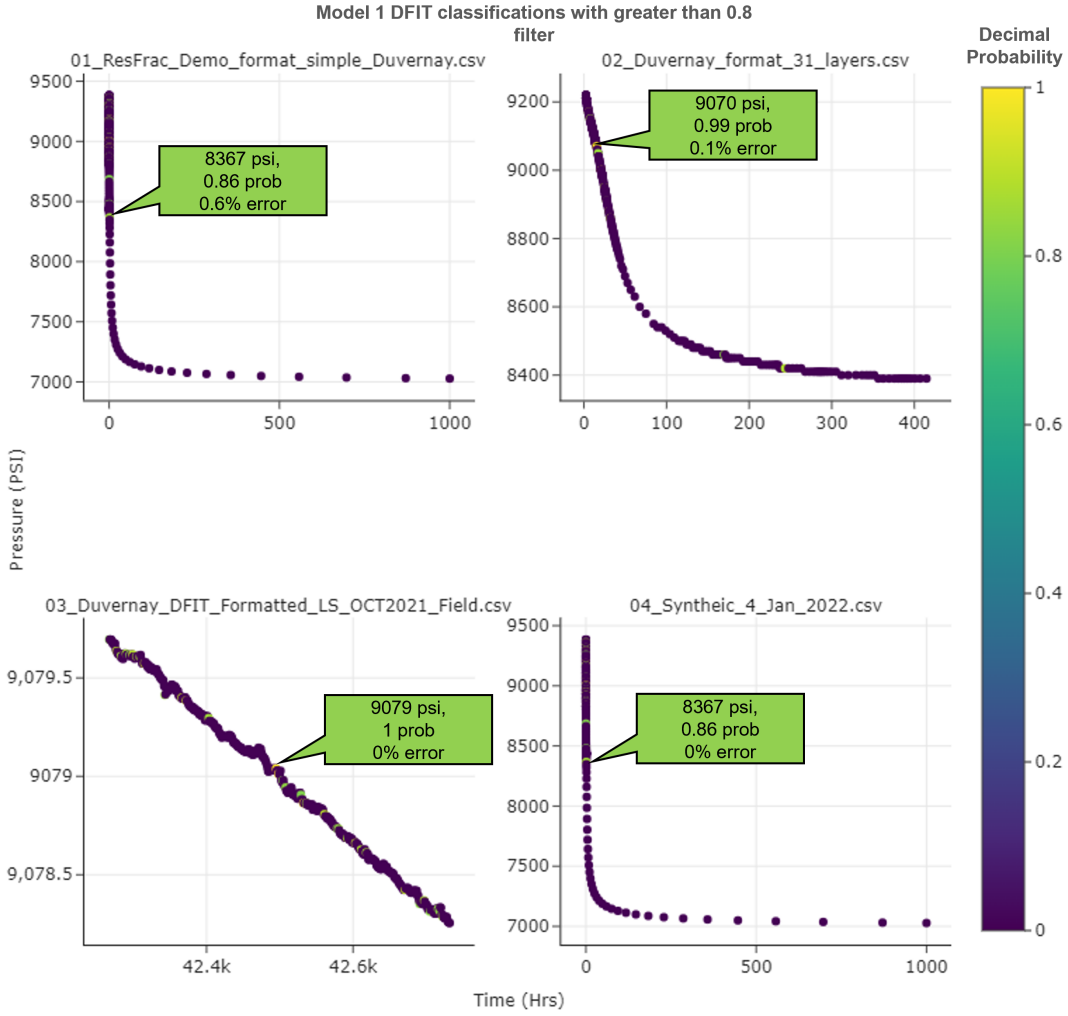


Figure 4.22: Resulting Sh_{min} classification probabilities from DNN model 1 as a continuous attribute superimposed on the Duvernay testing curve dataset. Probabilities in this case have been filtered to only show above 0.8 (80%).

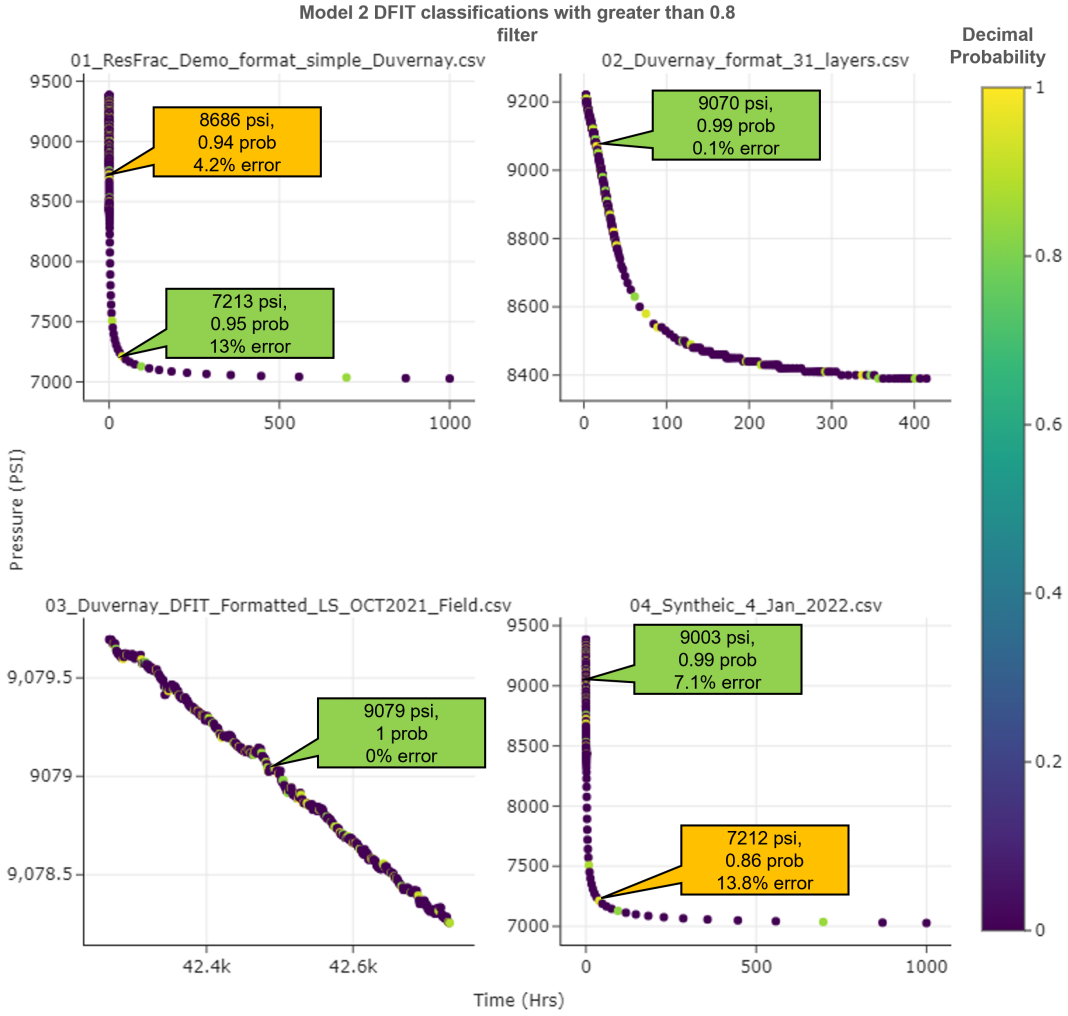


Figure 4.23: Resulting Sh_{min} classification probabilities from DNN model 2 as a continuous attribute superimposed on the Duvernay testing curve dataset. Probabilities in this case have been filtered to only show above 0.8 (80%).

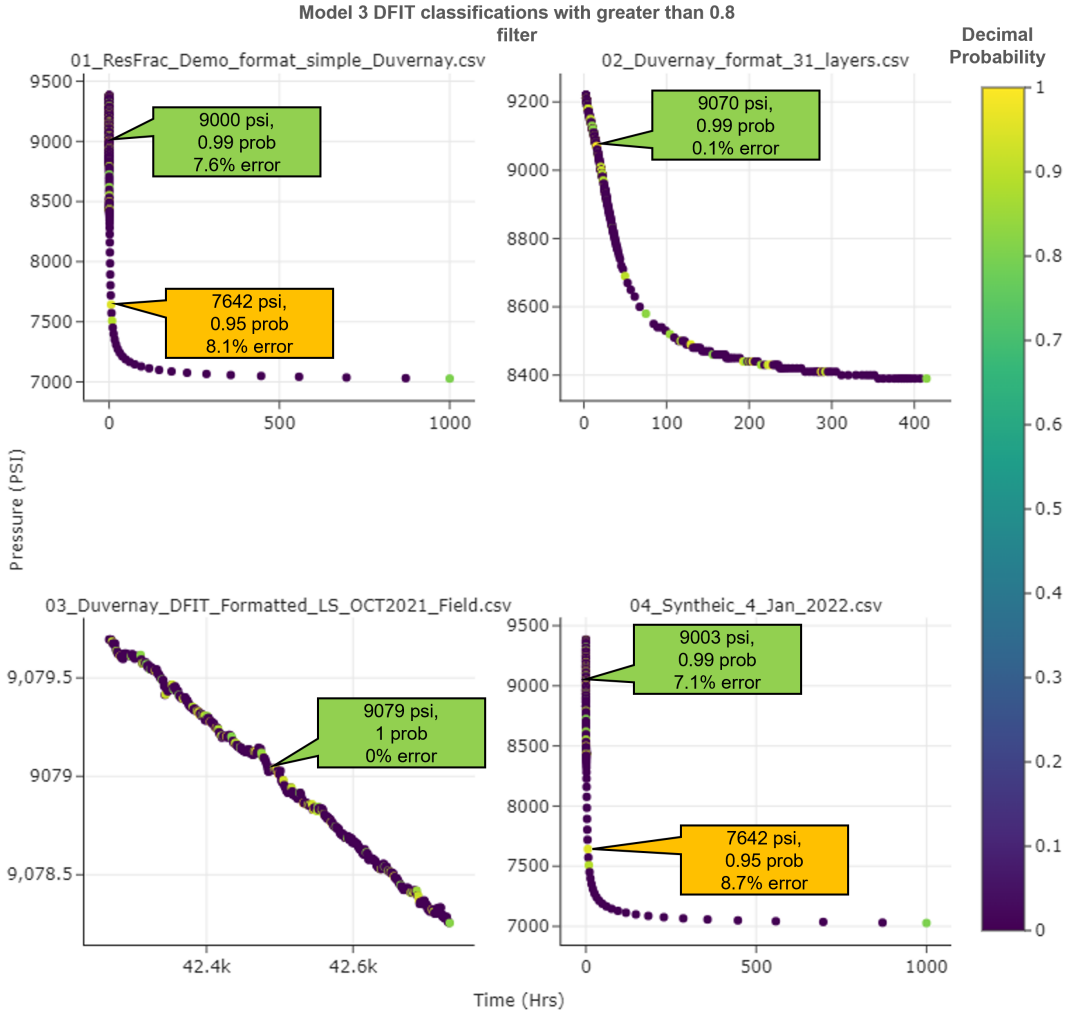


Figure 4.24: Resulting Sh_{min} classification probabilities from DNN model 3 as a continuous attribute superimposed on the Duvernay testing curve dataset. Probabilities in this case have been filtered to only show above 0.8 (80%).

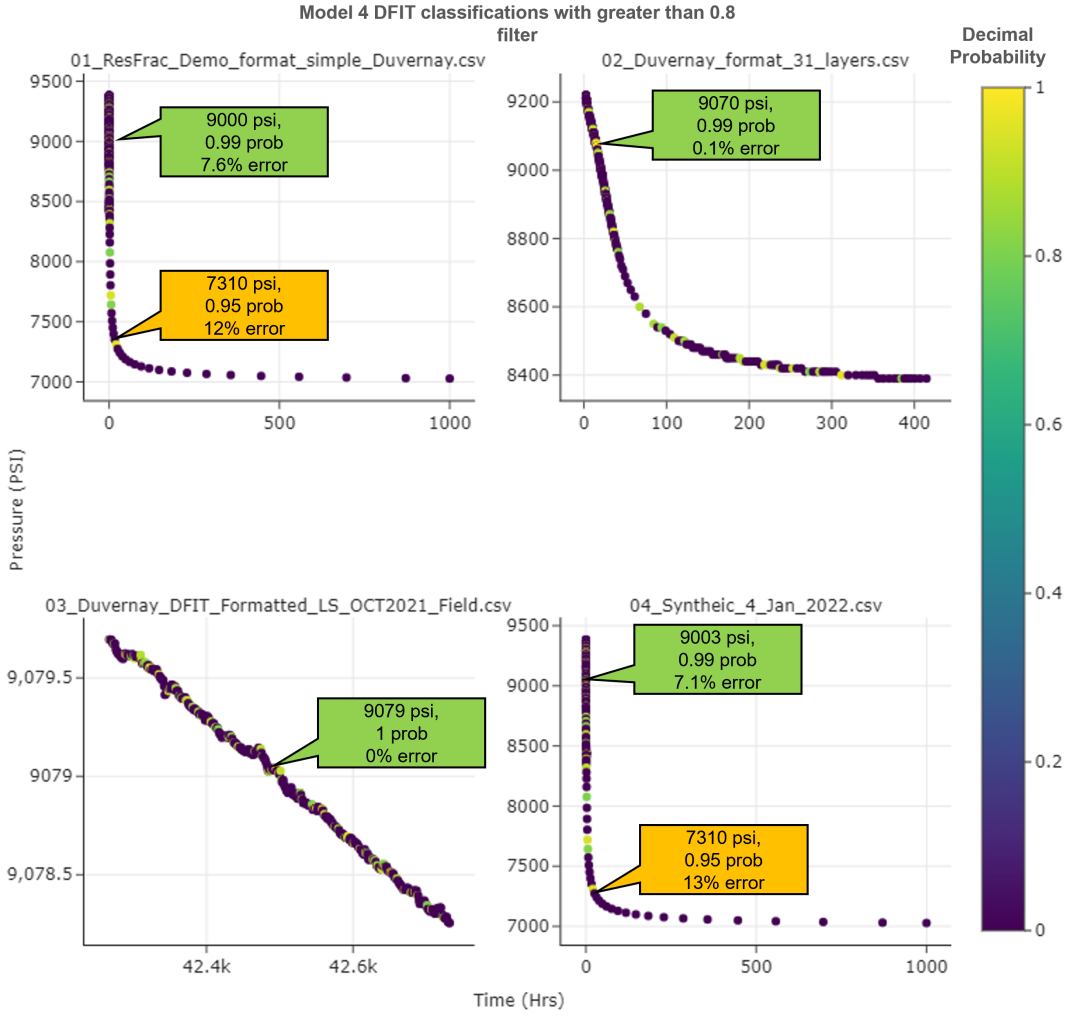


Figure 4.25: Resulting Sh_{min} classification probabilities from DNN model 4 as a continuous attribute superimposed on the Duvernay testing curve dataset. Probabilities in this case have been filtered to only show above 0.8 (80%).

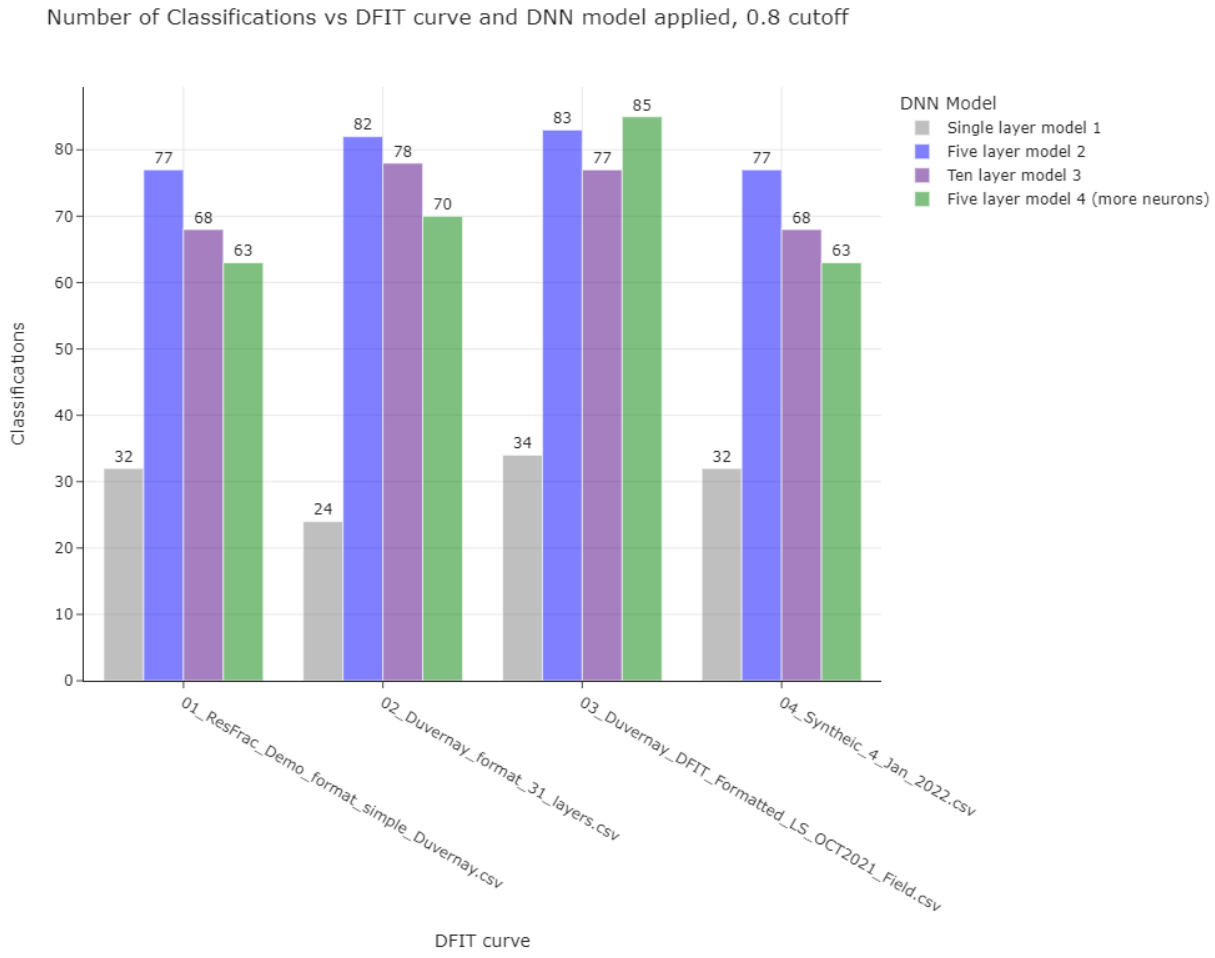


Figure 4.26: Number of potential Sh_{min} classifications vs Duvernay testing curve for each DNN model color coded.

4.4 Discussion

4.4.1 (1) Decision Tree and RF Classification

Inspection of Figures 4.15 and 4.16 portrays suboptimal results for the ability of Decision and RF tree based learning methods to identify Sh_{min}. As displayed in Figure 4.15, the Decision tree classification method identified one point as Sh_{min} on the Duvernay field case, however, the remainder of the testing DFIT curves remained unclassified. The point identified on the Duvernay field curve (9112psi) appears to be within reasonable error of the actual value (9079psi) with a 0.36% difference. Comparison with the RF method shows reduced classification effectiveness, with no classifications being successfully made (Figure 4.16).

To understand the poor performance of these methods, the generated classification models are interpreted in Figures 4.13 and 4.14. In both cases, the green classification area generally captures all Sh_{min} labels (green dots). The boundaries are however, rectangular and over extrapolated in some dimensions. For example, in the Decision tree model classification plot (Figure 4.13), the classification area of Sh_{min} appears to be interpolated far past the data points in an area where there is no data (150-1000hrs). This over extrapolation appears to correlate with the single classification made (9112psi) on the Duvernay field curve from the Decision tree learning model (Figure 4.15). For the case of the RF model in Figure 4.16, the classification boundary appears to have addressed this issue of over interpolation, however, this has the adverse effect of eliminating any classification potential for the testing data (Figure 4.16). This is an example of overfitting. To understand this overfitting phenomena, analysis focused on the training and testing datasets in Figure 2.2.

A comparison of the training and testing data in Figure 2.2 shows the adversities an overfit, non-generalized model can have on classification tasks. In this figure, the black training curves are clustered in a different region of the pressure vs time space when compared to the red testing set. Referring back to the classification model areas in Figures 4.15 and 4.16 one can note that the model boundaries would be concentrated around the black training curves,

particularly for the RF model. In other words, the training dataset used to build the model would not have enough information to extend this boundary further to the red testing curves. This is the issue of generality, a key problem that faces most ML methods where the model cannot classify what it has not seen before.

Interestingly, it appears that the Decision tree method handles the adversity of overfitting in this scenario better than the RF method. This observation contradicts the claim by multiple sources (Ho, 1995; Géron, 2019; Chollet, 2018) that RF methods tackle overfitting. Perhaps it is the simplicity of the Decision tree algorithm that allows it to to create 'rectangular over interpolation' (Figure 4.15) and therefore, a more generalized solution. At the same time, it can be observed that this method is still largely unsuccessful, classifying only 25% of the testing dataset.

Collectively, it is noted that the Decision and RF tree learning methods do not provide a universally generalisable approach for DFIT classification. For these methods to be effective, the user would require access to a large variety of permutated datasets with vastly differing Sh_{min} classifications. It is hypothesized that training the data using all training datasets at once, as two dimensional dataframe, does not allow the learning method to understand relationships between points belonging to the pressure decline. This leads to a grouped average classification which may not be generalized to other datasets. To address this adversity, the experimental idea was formulated that each curve in the training dataset needs to be learned independently, with emphasis on the complex non-linear relationships that exist between the points and their corresponding labels. The application of the DNN in the next section explores this method.

4.4.2 (2) DNN Classification

As discussed in the methods section, the application of the DNN learning model is hypothesized to create a more flexible (generalizable) and quick Sh_{min} predictive method. This is achieved by passing each DFIT pressure decline in the training dataset through the learning network

independently, allowing for non-linear relationships between the points and their labels to be identified. Training the tested DNN models in Table 4.1 was relativity quick, with the largest model taking 20 seconds. In this table, there is a correlation with the number of parameters trained and time taken to train the model. For example, the single layer model takes the shortest time (18.9seconds) with 435,555 parameters to train and the five layer model with additional neurons take the longest time (20seconds) with 607,406 parameters to train. The relatively short training time allowed for hyperparameter adjustments to be made with ease. A reduced dimensionality associated with time series data is attributed to these time savings. Other studies by Ismail Fawaz et al. (2019) also discuss better and faster convergence of time series data when compared to higher dimensionality photos.

Figures 4.17, 4.18, 4.19, and 4.20 display the learning metrics associated with each of these models. Generally, all models converge after 400 epochs with the training and validation loss converging to zero. Of the four models, Model 1 appears to have the smoothest convergence to zero for the loss metric and the quickest convergence to an accuracy score of one. This appears to correlate to high accuracy classifications on the testing data in Figure 4.22. A further dissection of these classification results reveals the field and second 'simple' model to highest probability solutions that match the true interpretations of Sh_{min}, this is followed by the 'complex' 31-layer Duvernay model and the 'simple' Duvernay model. Overall, highest probability classifications appear to represent true Sh_{min} values with low error for the first DNN model. Model 1 also displays strong generality characteristics when geologic and curve segment attributes are varied. This is supported in Figure 4.22 where 100% of the testing datasets have an accurate Sh_{min} classification.

Increasing the number of hidden layers to five in Model 2 appears to yield an oscillating convergence to a loss of 1 in Figure 4.18 and a higher variation in accuracy before 400 epochs. Perhaps the addition of more layers in a more complex model has resulted in convergence instability when compared to Model 1. This results in similar Sh_{min} classifications to Model 1, however, with higher error. In this scenario, the field and 'complex' 31-layer DFIT curves

appear to have the closest classifications (Figure 4.23). In contrast, the two 'simple' DFIT curves appear to have poor classification results, with both the first and second highest probability classifications having high error (Figure 4.23). The generality of this model appears to be moderate, with only 50% of the training data being accurately classified.

Adding more complexity to the DNN model with 10 layers (Model 3) appears to yield rapid convergence to a loss of 1 at 200 epochs, followed by oscillation in Figure 4.19. Both the validation and training dataset accuracy appear to be low at 0.5 and 0.33, respectively. Past 400 epochs the model appears to lose stability with poor accuracy. Similar to Model 2, it appears that fitting a complex model to seemingly simple system has resulted in degraded classification results. In this scenario, classification results in Figure 4.24 closely mirror the observations made from Model 2 results. The generality of this model appears to be moderate, with only 50% of the training data being accurately classified.

Model 4 is the final model tested with 5 layers and additional neurons above the thresholds defined by Boger and Guterman (1997) and Berry and Linoff (2004). Model convergence and accuracy in Figure 4.20 appear to mimic the same observations made from Model 2. Again, this hints to the same observation that a complex model may not be adequate for modelling a simplistic data space. Classification results from Model 4 (Figure 4.25) appear to closely match those generated by Model 2 (Figure 4.23). This suggests the criteria outlined by Boger and Guterman (1997) and Berry and Linoff (2004) does not have substantial effects on this classification problem. On the topic of generality, this model appears to be moderately generalizable, with only 50% of the training data being accurately classified.

Collectively, the experiments performed for DFIT DNN classification optimization problem can be interpreted to make interesting conclusions. The first of these conclusions is that training DNN models using this dataset appears to be relatively quick, therefore, model optimizations with varying hidden layer combinations appear to be less necessary. As depicted, efforts to optimize the model with more layers appears to have lead to poorer and less generalizable classification results. This may be the result of working with lower

dimensionality time series data where the addition of hidden layers and 'complexity' appears to have degraded model convergence and accuracy. For this reason the single layer model (Model 1) has created optimal results. This observation is further portrayed in Figure 4.26 where the total number of potential high probability classifications for the single layer model is on average half the size of the other model combinations.

Finally, the criteria set by Boger and Guterman (1997) and Berry and Linoff (2004) appears to not be not essential to for accurate DFIT classification. This is supported by the classification produced by Models 2 and 4 having significant similarities. It is noted that DNN optimization for the DFIT classification problem should instead focus on the number of hidden layers in the model, this appears to have the most significant impact.

4.4.3 Comparison of the Two Supervised Methods

In this study, I have designed and tested (1) Decision and RF classifiers and (2) DNN supervised methods as an effort to save time and eliminate bias in the DFIT classification problem. Overall, the Decision and RF tree based classification showed the greatest flexibility for data input and model generation. The DNN model contrasted from this where care was required to match the size of the datasets, as well as ensuring selected curve segments contained an Sh_{min} value. Model training for both methods was relativity rapid with 20 seconds being the longest training time. For this reason, data formatting appears to be the key contributor in time taken to reach a solution from each model that would be contained in the supervised user interface of Figure 4.12.

In this experiment, data formatting for the tree based methods took in the order of minutes. This contrasts from the DNN method, where data formatting was in the order of hours. From this point of view the Decision and RF methods excel in time savings for the user.

Although time savings is an important aspect of ML method performance, so is the ability to identity the event of interest. This is were the Decision and RF methods show poor

performance and poor generality. The simplest of the DNN models contrast from this and produce high accuracy Sh_{min} classifications. Furthermore, the DNN model was generalizable to varying curve segment information and geologic conditions. This improved accuracy is believed to be attributed to the DNN method learning the point relationships in each curve independently opposed to the grouped learning approach for Decision and RF based learning.

Human bias reduction for both of these methods is achieved by allowing ML methods to derive modeled relationships for key parameters. Comparatively, analytical methods in Chapter 1 relied on line fitting procedures, subject to human variation. Additional bias reduction is achieved in these methods by eliminating fracture physics assumptions made when creating derivative curves for the analytical process. This is achieve by only using the pressure versus time data. It can be noted that bias can still be introduced in the training population selection for supervised methods. In the case of this study care was taken to vary parameters in the training set. That being said, it was observed the tree based methods still demonstrated biased classification results. In contrast, the DNN was able to produced generalizable solutions, suggesting an unbiased model.

After analysis of the two tested supervised methods, it appears the DNN learning method is the optimal supervised learning method for Sh_{min} classification. Arguably, to save time and improve Sh_{min} interpretations using the DNN method, this would still require a developed data formatting technique to input data into the trained model. A potential solution would be to develop a secondary program to perform this formatting.

4.5 Conclusions

In efforts of developing new methods to challenge the clustering method proposed by the *CREWES DFIT Clustering app* this chapter designed and tested two groups of supervised ML methods for Sh_{min} classification. These consisted of tree based and DNN learning methods. The goal of designing these methods was to save the end product user time by eliminating

the interactivity portion of the DFIT data science Venn diagram (Figure 1.9) and further eliminate human bias when interpreting these curves. This will allow for quick and accurate interpretation of Sh_{min} and therefore, efficient optimization of the hydraulic stimulation process to improve economics.

Based on the goals of this experiment, one of the key performance factors was time savings. It was found that the tree based methods: Decision and RF excel in simplistic data formatting with little data preconditioning required. The DNN method however required a more complex data structure as well as data clipping for input into the training model. The model training time was reasonably quick, taking a maximum of 20 seconds. For these methods, data formatting appears to be the longest time consumer. A possible solution for this is developing general formatting codes for input into the trained classifier model.

The goal of saving time must also be balanced with classification accuracy and generalizability of the trained model. From this perspective the DNN methods outperform the tree based methods with 100% classification of Sh_{min} for the top performing DNN model and only 25% for the top performing tree based method. This contrast in accuracy is hypothesized to be the result of the data formatting and input into the respective learning models. The DNN method looks at each DFIT curve independently and learns non-linear relationships between each point, tree based methods look at all training curves collectively and try to build an average classifier. For this reason DNN models produce relatively unbiased classifications.

Collectively, the DNN method appears to produce the best classifier model for this problem. Within the DNN architecture, it appears that less layers produce improved model convergence and accuracy. This is related to the reduced dimensionality of this data requiring simplified models. Future studies may look at the application of other neural network structures such as Convolutional Neural Networks (CNN) and Long Short Term Memory (LSTM) as alternatives.

Chapter 5

ML method comparison

Following the development of unsupervised and supervised methods, this chapter will look at contrasting strengths and weaknesses. This is achieved by looking at the aspects of (1) Ease of data input/training/prediction, (2) Interactivity and generalization, (3) Classification power, accuracy, and bias reduction, and (4) Overall method ranking. Using the findings from this process, recommendations will be made for future projects in the realm of DFIT event detection.

5.0.1 Ease of data input/training/prediction

Collectively, the four methods in this study have varying degrees of complexity at different stages of the learning and prediction process for DFIT key parameters. To contrast these complexities, Figure 5.1 displays the amount of time taken during the data formatting (input), model training/calibration, and prediction process for the analytical (Chapter 1), unsupervised clustering app (Chapter 3), supervised tree (Chapter 4), and supervised DNN methods (Chapter 4). The analytical method appears to take the largest amount of time to produce a DFIT interpretation and the supervised tree based method the shortest. The time difference of 27 hours between these two end member methods can be dissected by analysing the process times (Figure 5.1).

Process times within Figure 5.1 reveal a key paradigm shift in the decrease of data formatting time when comparing supervised processes to unsupervised and analytical processes. This is mostly the result of supervised algorithms requiring a predefined and generalized training dataset. Furthermore, Chapter 4 displayed that supervised DNN methods also require data formatting and windowing of datasets. This is in contrast to the analytical and unsupervised clustering app methods where the user is simply required to input pressure vs time data for the DFIT under investigation with more emphasis on the manual interpretation. It is important to note that data formatting algorithms can be developed to reduce the data formatting time to seconds for these supervised methods, however, not all field data is created equal. Therefore, it would be a difficult task to create a formatting algorithm that captures all data variations. Furthermore, the supervised methods tested only required the labelling of Sh_{min}, arguably, data labelling for a holistic result, containing ISIP and $P_{reservoir}$ would also be more time consuming.

The relative time saving from data formatting created by analytical and unsupervised clustering methods appear to be overshadowed by training/calibration and prediction times. It appears as though the aspect of having user interaction for the analytical and unsupervised clustering methods adds significantly more time to these processes when compared to supervised methods (Figure 5.1). For example, in Figure 5.1 the training time jumps from 0.2 seconds to 24+ hours for the analytical method. It is important to note that this training time for the analytical method would be dedicated to the human interpreter getting acquainted to the manual methods for DFIT interpretation and is also based on experience. In the case of this study, it took me roughly 24 hours to read literature, and build confidence to make manual interpretations. Following building the fundamentals to perform analytical methods, the prediction process is also performed solely by the trained human interpreter. This may be thought of as training the 'human' DFIT interpretation model, as opposed to the other three computer aided vision models. It appears that integrated computer vision in human models significantly reduced training and prediction times for the unsupervised

clustering app. In this scenario, unsupervised clustering is identifying the boundaries to the human subject to adjust and optimize appropriately. Looking back to Chapter 3, it was noted that unsupervised clustering methods merge the three domains of DFIT data science (Figure 1.9). In doing so, the addition of computer vision is the reason why we can reduce training and prediction time with respect to the pure analytical methods.

Using the idea of human analytical methods being slower for interpretation, one can understand why the supervised DNN and tree based learning is still faster than the unsupervised learning method. This is because these supervised methods are fully dependent on computer vision an no human input is required. The pseudo algorithm for supervised methods (Figure 4.12) further exemplifies this, where the user simply applies the computer trained model to produce a solution. This differs from the unsupervised algorithm where the user is required to update the model parameters if necessary (Figure 3.3).

The supervised tree method is the most time efficient followed by the supervised DNN, unsupervised clustering app, and analytical methods when evaluated in an end user product context. This is assuming data formatting is simple and repeatable for both supervised methods. With this in mind, one may imply from this that supervised methods are superior, however, it is also important to weigh the resulting costs this time saving has on interactivity, generalization, and the classification accuracy. This will be investigated in the next two sections.

5.0.2 Interactivity and generalization

As displayed in the dataflow diagram for supervised methods, (Figure 4.12) there is no user interactivity and only a final model from which classifications can be made from. This may be referred to as a non-fluid model as it cannot be easily changed after training. In contrast, the unsupervised and analytical algorithms allow for user model adjustment after the data has been inputted (Figure 3.3). This is further demonstrated in the DFIT data scientist venn diagram (Figure 1.9) where the interactive analytical domain is not present for supervised

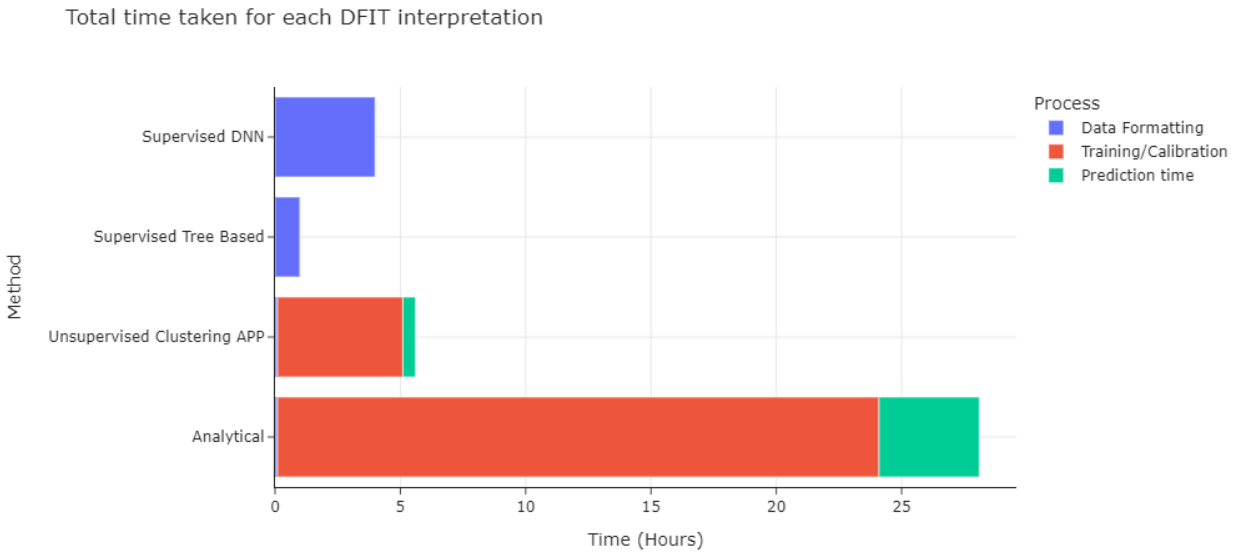


Figure 5.1: Total time required to learn each method in this thesis and make a prediction. Categories of time are split up into data formatting, training time, and prediction time.

methods. With this in mind, Chapters 3 and 4 display that the addition of interactivity appears to have two benefits: process understanding and fluid model adjustment for event identification optimization (generalization).

Efforts to understand classification trends created by ML methods is easiest when integrating interactivity into the learning method. For the case of the unsupervised clustering app, this interaction allowed for the user to test multiple variable combinations, use PCA plots, and quality control displays to understand the varying degree to which each parameter affects the classification. In contrast, users of supervised methods simply rely on the stagnant 'black box' model for classifications. This is an adversity that faces the supervised methods. The lack of interactivity diminishes the users understanding of the problem as no work is required.

Interactivity contributes to the optimization of key parameter identification for the unsupervised method and analytical methods. As displayed in Figure 3.10, alternate optimizations of the clustering algorithm allowed for alternate event identifications to be made. On the

other hand, the tree based methods were unsuccessful at making classifications of Sh_{min} on most instances of the training data in chapter 4 (Figures 4.15 and 4.16). In contrast, the supervised DNN with no interactivity showed promise for generalized solution potential by classifying all four instances of training data Sh_{min} occurrences in Figure 4.21.

The interactivity aspect appears to have improved the analytical and unsupervised clustering methods' ability to understand the classification process and generalize solutions to differing geologic instances. As for methods with no interactivity, the supervised DNN appears to show generalized model solutions as well for Sh_{min} classifications. Future studies may seek to expand this supervised DFIT classification method to classify multiple events at once for a closer comparison to the analytical and unsupervised clustering methods which already do so. The next subsection will shift focus to the classification accuracy and overall bias contained in each method.

5.0.3 Classification power, accuracy, and bias reduction

Another category of consideration when ranking the four clustering methods in this study is the ability to classify events accurately and reduce human bias. To investigate the aspect of classification accuracy, Figure 5.2 is generated as a summary of the probability of key parameters being identified by the optimized method and the associated error range for identified attributes (black bar). The supervised tree based methods have the least success at identifying events for varying DFIT curve conditions. This is also associated with the highest error produced classification.

For the remaining three tested methods, differences between probability of event identification and accuracy become more subtle. In Figure 5.2 it appears that the unsupervised clustering app has the lowest relative probability of identifying all events. This decreased event identification probability is the result of missing curve data making the clustering app unable to identify the $P_{reservoir}$. In contrast, the supervised method is not affected by this as it was only trained to identify Sh_{min}. With this in mind, a strength of the analytical method

is its ability to estimate reservoir pressure when the data is missing from the DFIT pressure decline. Future studies may investigate techniques of interpolating this missing data.

To create a more accurate comparison of these three methods, Figure 5.2 is adjusted to show events that the data allowed trained models to identify (Figure 5.3). This does not change the events (ISIP, Sh_{min}, and P_{res}) for the analytical method, however, will reduce the clustering app to ISIP and Sh_{min} and supervised method to Sh_{min}. Evaluation of this plot reveals all methods have equal probability of creating an event interpretation when the data is fully complete. This still leaves the accuracy of the event identified in question. To explore these differences the maximum value error bars can be analysed in Figure 5.3. Ranking the associated event error of these methods, the supervised DNN appears to have the lowest at 0.6%, followed by the unsupervised clustering at 1% and analytical method at 4%. This shows the strength of the unsupervised and supervised methods over the original analytical method. It further suggests the power of the supervised DNN. Future studies will expand its classifying power to all key parameters on the DFIT pressure decline.

After producing accurate event interpretations, one of the main goals of this experiment was to also reduce human user bias. Unsupervised and supervised methods achieve this by relying on mathematical inferences opposed to line fitting procedures from analytical methods. This is demonstrated in Figure 3.11 of chapter 3 where PCA analysis reveals the hidden mathematics of the clustering method. A potential bias remaining in the unsupervised method is the use of curves created by the Carter leak-off assumption. The supervised method eliminates this potential bias by only relying on the pressure vs time data. For the supervised methods, bias may exist in the training data used, it is the model builder's responsibility to create a generalized training set of varying geologic conditions.

5.0.4 Overall method ranking

The overarching goal of this thesis is to explore ML methods' capability of improving DFIT interpretation in terms of time saving and human bias reduction. To fully compare and contrast

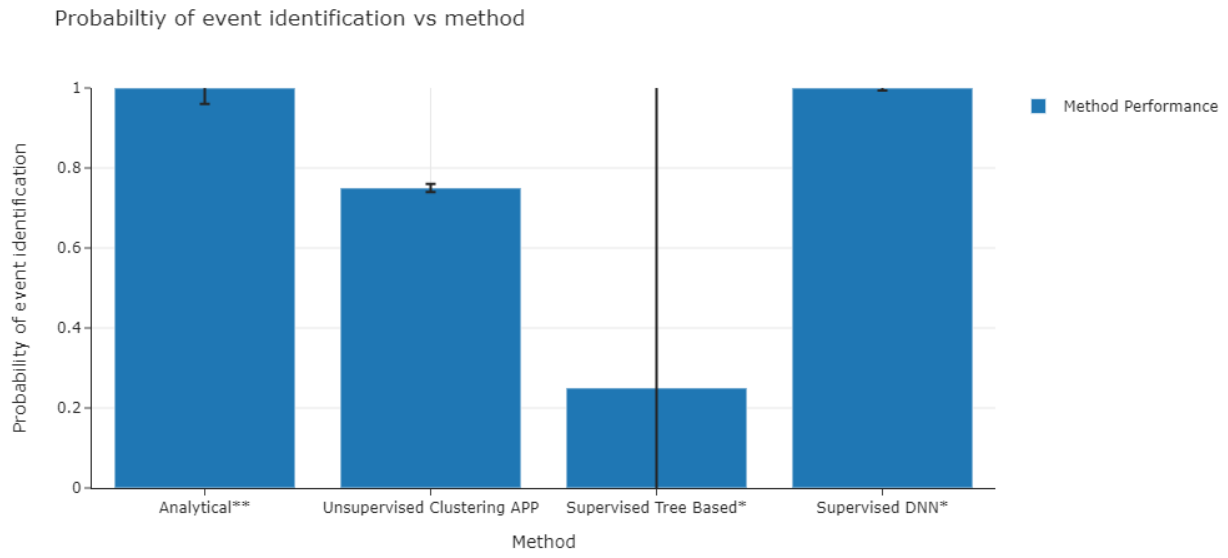


Figure 5.2: Probability of event identification vs method in this study. The maximum error bar range for the identified value is overlaid. Methods with a single star indicate averages when only one event of interest was targeted (Sh_{min}). Methods with two stars are subject to vary base on the interpreters experience.

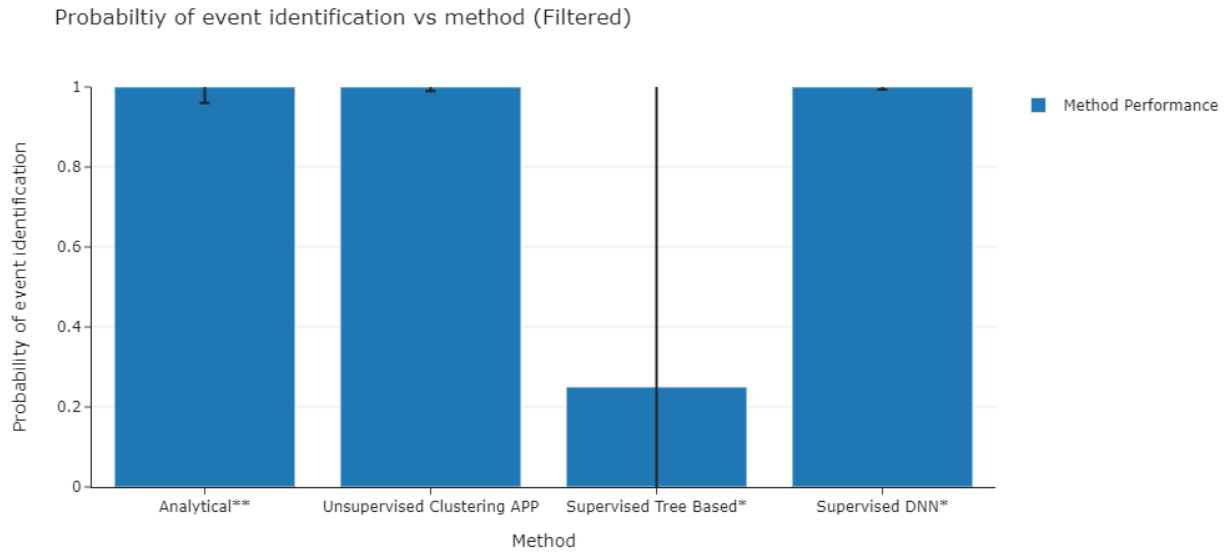


Figure 5.3: Probability of event identification vs method in this study. The maximum error bar range for the identified value is overlaid. Methods with a single star indicate averages when only one event of interest was targeted (Sh_{min}). Methods with two stars are subject to vary base on the interpreters experience.

the benefits and weaknesses of each method attributes such as: time savings, interactivity, generalization, accuracy, and human bias reduction were discussed in the prior sections. Table 5.1 displays a summary the strengths for each method. This table demonstrates that although supervised methods can all demonstrate time saving properties it does not mean the output result is satisfactory. This is particularity the case for the supervised tree methods. Of the methods tested, the unsupervised clustering app appears to meet all the desirable requirements in Table 5.1. This method allows for time saving and bias reduction while including aspects of interactivity observed in the analytical method. It should be noted that the supervised DNN method allows for a significantly reduced interpretation time when compared to analytical and unsupervised clustering methods. This also comes at the cost of losing interactivity and the users understanding of how the solution was formed.

With the observations summarized in Table 5.1 and overall method ranking can be derived for the DFIT data tested in this thesis. In order from optimal to suboptimal methods this would be (1) unsupervised clustering app, (2) supervised DNN, (3) analytical, and (4) supervised tree based methods. It is important to note that the top three methods portrayed in this ranking may vary in order based on data type and are all equally viable methods. Method selection should primarily be based on the users time and risk tolerance to human bias. Collectively, the ML methods tested in this thesis show great promise for DFIT interpretation optimization.

5.0.5 Recommended future work

Methods and adversities identified in this thesis leave significant room and promise for future work. For the supervised methods, studies testing other models and app building may improve interactivity. The DNN supervised method shows strengths in its ability to reduce interpretation time. This method should be further expanded to classify additional events including $ISIP$ and $P_{reservoir}$. To achieve a prediction of $P_{reservoir}$ a method will also have to be developed for estimating missing segments of the DFIT curve. Lastly, research may

Table 5.1: Stengths of each method

# Method	Time savings	Interactivity	Generalization	Accuracy	Human Bias Reduction
Analytical		✓	✓	✓	
Unsupervised clustering	✓	✓	✓	✓	✓
Supervised Tree	✓				
Supervised DNN	✓		✓	✓	✓

also focus on stress testing these methods with more varying geologic conditions. This will build further confidence in how generalizable these methods are.

Chapter 6

Conclusion

6.0.1 Future projects

The following avenues for future research have been identified from this project as follows:

- Further stress testing of the unsupervised app and supervised method
  - Sub-studies using alternative synthetic modeling software for curve generation and additional field tests
  - testing on non-ideal pressure declines
- Attribute investigation for ML method optimization
  - Frequency domain transforms and additional filtering to create additional attributes for classification
  - Hybrid supervised ML approach constrained by models such as Carter leak-off flow
- Development of unsupervised and supervised user interface
  - Inclusion of additional parameters such as probability filtering in the Gaussian Mixture model method for unsupervised clustering

  - Development of app interface for supervised method to invoke more interaction
  - Testing of new supervised methods such as CNN
- Additional event classification
  - Do unclassified cluster boundaries created by supervised methods have physical meaning in the fracture closure process?
  - Multi-event classification (ISIP, Sh_{min}, $P_{reservoir}$) for supervised methods
  - Event classification on pressure ramp-up cycle leading up to formation breakdown
  - Trail of methods on new data sets such as cyclic fracture stimulation data
- Interpolation of missing pressure decline data

6.0.2 Conclusion

Diagnostic Fracture Injection Tests (DFIT) are an important in-situ measurement method for deriving key subsurface parameters such as Sh_{min} and $P_{reservoir}$. This information can then be used to optimize fracture stimulation design. To achieve these optimization aspects, an effective and accurate method must be developed to obtain key parameters from an apparently smooth pressure vs time DFIT. Current analytical methods have developed a solution to this adversity by modeling flow behavior and translating this to interpretable derivative curves. This is time consuming, requiring a different method for each key parameter, and can be subjected to unintended human bias in line-fitting processes. This thesis tests Machine Learning methods (ML) as a mean of addressing timely and potentially biased processes.

To address adversities faced by analytical methods, a combination of unsupervised and supervised ML methods were applied to a dataset of synthetic and Duvernay Formation field tests. Results found that supervised methods where most effective at reducing interpretation time, however, required significant data formatting and eliminated the interactive aspect of interpretation. Unsupervised methods applied in an *R-Studio Shiny Web App®* environment

appeared to address this issue of interactivity, however, made the process on average more time consuming as the user can now iterate through possible solutions. Another benefit of using unsupervised methods is that exploration as to *why* classifications appear where they do can be done with ease in the interactive process. This exploration revealed a combination of dimensionality and point frequency to contribute to classification boundaries. This arguably creates a more mathematical solution opposed to the line fitting process that is encompassed in analytical methods. Lastly, by varying the geologic conditions and segments expressed on the DFIT curves for ML testing, it was found that both supervised DNN and clustering methods where generalizable, and could also be optimized to identify groups of key parameters.

Overall the ML approach to DFIT interpretation shows significant promise for time and bias reduction. This will allow the end-user to save time and make unbiased optimizations to stimulation design or risk analysis. Future projects will look at stress testing these methods with further varying DFITs, attribute optimization, interface development, additional event classification, and interpolation of missing data.

Bibliography

Agarwal, R. G. (1980). A new method to account for producing time effects when drawdown type curves are used to analyze pressure buildup and other test data. In *SPE Annual Technical Conference and Exhibition*. OnePetro.

Alatrach, Y., Mata, C., Omrani, P. S., Saputelli, L., Narayanan, R., and Hamdan, M. (2020). Prediction of well production event using machine learning algorithms. *Presented at the Abu Dhabi International Petroleum Exhibition and Conference*.

Baddeley, M. C., Curtis, A., and Wood, R. (2004). An introduction to prior information derived from probabilistic judgments: elicitation of knowledge, cognitive bias and herding. *The Geological Society of London*, 239:15–27.

Barree, R. D., Barree, V. L., and Craig, D. (2009). Holistic fracture diagnostics: Consistent interpretation of prefrac injection tests using multiple analysis methods. *SPE Production and Operations*, 24:396–406.

Berry, M. J. and Linoff, G. S. (2004). *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Boger, Z. and Guterman, H. (1997). Knowledge extraction from artificial neural network

models. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, volume 4, pages 3030–3035 vol.4.

Bond, C. E., Gibbs, A. D., Shipton, Z. K., and Jones, S. (2007). What do you think this is? "conceptual uncertainty" in geoscience interpretation. *GSA Today*, 17:4–11.

Bourdet, D., Ayoub, J. A., and Pirard, Y. M. (1989). Use of pressure derivative in well-test interpretation. *SPE Formation Evaluation*, 4:293–302.

Budowle, B., Bottrell, M. C., Bunch, S. G., Fram, R., Harrison, D., Meagher, S., Oien, C. T., Peterson, P. E., Seiger, D. P., Smith, M. B., Smrz, M. A., Soltis, G. L., and Stacey, R. B. (2009). A perspective on errors, bias, and interpretation in the forensic sciences and direction for continuing advancement. *Forensic Science*, 54:798–809.

Chen, D., Li, Y., Yang, X., Jiang, W., and Guan, L. (2021). Efficient parameters identification of a modified gtn model of ductile fracture using machine learning. *Engineering Fracture Mechanics*, 245:107535.

Chollet, F. (2018). *Deep Learning With Python*. Manning Publications.

Cios, K. J. (2018). Deep neural networks—a brief history. In *Advances in Data Analysis with Computational Intelligence Methods*, pages 183–200. Springer.

Clarkson, C. R., Jensen, L., and Chipperflid, S. (2012). Unconventional gas reservoir evaluation: What do we have to consider? *Journal of Natural Gas Science and Engineering*, 8:9–33.

Cramer, D. D. and Nguyen, D. H. (2013). Diagnostic fracture injection testing tactics in unconventional reservoirs. *SPE Hydraulic Fracturing Technology Conference*.

Dror, I. E. (2020). Cognitive and human factors in expert decision making: Six fallacies and the eight sources of bias. *Analytical Chemisty*, 92:7998–8004.

Dunn, L., Schmidt, G., Hammermaster, K., Brown, M., Bernard, R., Wen, E., Befus, R., and Gardiner, S. (2012). The duvernay formation (devonian): sedimentology and reservoir characterization of a shale gas/liquids play in alberta, canada. In *Canadian Society of Petroleum Geologists, Annual Convention, Calgary*.

Duong, A. N. (1989). A new set of type curves for well-test interpretation with the pressure/pressure-derivative ratio. *SPE Form Eval 4*, 02:264–272.

Ester, M., Kriegel, H. P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *kdd*, 96:226–231.

Farghal, N. S. and Zoback, M. D. (2014). Utilizing ant-tracking to identify slowly slipping faults in the barnett shale. In *SPE/AAPG/SEG Unconventional Resources Technology Conference*. OnePetro.

Fossen, H. (2016). *Structural Geology*. Cambridge University Press.

Froner, B., Purves, S., Lowell, J. D., and Henderson, J. (2013). Perception of visual information: the role of colour in seismic interpretation. *First Break*, 31:29–34.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. (2018). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*.

Geoffrey, E. (2018). *Cognitive Biases in Visualization*. Springer International Publishing.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.".

Good, I. J. (1992). *Rational Decisions*. Springer New York, New York, NY.

Goodfellow, I., Bengio, Y., and Courville, A. (2017). Deep learning (adaptive computation and machine learning series). *Cambridge Massachusetts*, pages 321–359.

Gorodov, E. Y. and Gubarev, V. V. (2013). Analytical review of data visualization methods in application to big data. *Journal of Electircal and Computer Engineering*, 2013.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

Hu, W., Yuchen, J., Wu, X., and Chen, J. (2021). Progressive transfer learning for low-frequency data prediction in full-waveform inversion. *Geophysics*, 86:R369–R382.

Ippolito, M., Ferguson, J., and Jenson, F. (2021). Improving facies prediction by combining supervised and unsupervised learning methods. *Journal of Petroleum Science and Engineering*, 200:108300.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963.

Jung, H., Sharma, M. M., Cramer, D. D., Oakes, S., and McClure, M. W. (2016). Re-examining interpretations of non-ideal behavior during diagnostic fracture injection tests. *Journal of Petroleum Science and Engineering*, 145:114–136.

Kiefer, J. and Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466.

Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020.

Knapp, L. J., McMillan, J. M., and Harris, N. B. (2017). A depositional model for organic-rich duvernay formation mudstones. *Sedimentary geology*, 347:160–182.

Li, J., Izakian, H., Pedrycz, W., and Jamal, I. (2021). Clustering-based anomaly detection in multivariate time series data. *Applied Soft Comutung Journal*, 100:106919.

Liu, G. and Ehlig-Economides, C. (2018). Practical considerations for diagnostic fracture test (dfit) analysis. *Journal of Petroleum Science and Engineering*, 171:1133–1140.

Liu, Y., Wu, K., Jin, G., and Moridis, G. (2020). Rock deformation and strain-rate characterization during hydraulic fracturing treatments: Insights for interpretation of low-frequency distributed acoustic-sensing signals. *SPE Journal*, 25:2251–2264.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, 1:281–297.

McClure, M., Bammidi, V., Cipolla, . C., Cramer, D., Martin, L., Savitski, A. A., Sobernheim, D., and Voller, K. (2019). A collaborative study on dfit interpretation: Integrating modeling, field data, and analytical techniques. *SPE/AAPG/SEG Unconventional Resources Technology Conference, Denver, Colorado, USA*, 123.

McClure, M., Kang, C., Medam, S., and Hewson, C. (2021). Resfrac technical writeup. *https://arxiv.org/abs/1804.02092v9*.

McClure, M. W., Jung, H., Cramer, D. D., and Sharma, M. M. (2016). The fracture-compliance method for picking closure pressure from diagnostic fracture-injection tests. *SPE Journal*, 04:1321–1339.

McGuinness, L. A. and Higgins, J. P. (2021). Risk-of-bias visualization (robvis): An r package and shiny web app for visualizing risk-of-bias assessments. *Research synthesis methods*, 12(1):55–61.

McKean, S. H., Priest, J. A., Dettmer, J., and Eaton, D. W. (2019). Quantifying fracture networks inferred from microseismic point clouds by a gaussian mixture model with physical constraints. *Geophysical Research Letters*, 46:11008–11017.

Mohamed, M. I., Mehta, D., Salah, M., Ibrahim, M., and Ozkan, E. (2020). Advanced machine learning methods for prediction of fracture closure pressure, closure time, permeability,

and time to late flow regimes from dfit. *SPE/AAPG/SEG Unconventional Resources Technology Conference*.

Molenaar, M. M., Hill, D. J., Webster, P., Fidan, E., and Birch, B. (2012). First downhole application of distributed acoustic sensing for hydraulic-fracturing monitoring and diagnostics. *SPE Drilling and Completion*, 1:32–38.

Müller, A. C. and Guido, S. (2016). *Introduction to machine learning with Python: a guide for data scientists*. " O'Reilly Media, Inc.".

Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., and Brown, S. D. (2004). An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 18(6):275–285.

Nolte, K. G. (1988). Principles for fracture design based on pressure analysis. *SPE Production Engineering*, 3:22–30.

Othman, A., Iqbal, N., Hanafy, S. M., and Waheed, U. B. (2021). Automated event detection and denoising method for passive seismic data using residual deep convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11.

Pandey, Y. N., Rastogi, A., Kainkaryam, S., Bhattacharya, S., and Saputelli, L. (2020). *Machine Learning in the Oil and Gas Industry*. Apress.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

Potocki, D. (2016). Understanding fracture stimulation complexity and reservoir performance at in-situ conditions using dfits. *Reservoir*, 03:14–18.

Prevos, P. (2019). *Principles of Strategic Data Science: creating value from data, big and small*. PACKT Publishing Limited.

Redner, R. A. and Walker, H. F. (1984). Mixture densities, maximum likelihood and the em algorithm. *SIAM Review*, 26:195–239.

Saikia, P., Baruah, R. D., Singh, S. K., and Chaudhuri, P. K. (2020). Artificial neural networks in the domain of reservoir characterization: A review from shallow to deep models. *Computers and Geosciences*, 135:104357.

Savalia, S. and Emamian, V. (2018). Cardiac arrhythmia classification by multi-layer perceptron and convolution neural networks. *Bioengineering*, 5(2):35.

Shen, L., Schmitt, D. R., and Haug, K. (2018). Measurements of the states of in situ stress for the duvernay formation near fox creek, west-central alberta. *Alberta Energy Regulator / Alberta Geological Survey*, 97:29.

Shen, Y., Cao, D., Ruddy, K., and Moraes, L. F. T. D. (2020). Near real-time hydraulic fracturing event recognition using deep learning methods. *SPE Drill and Compl*, 35:478–489.

Siddhamshetty, P., Wu, K., and Kwon, J. S. (2019). Modeling and control of proppant distribution of multistage hydraulic fracturing in horizontal shale wells. *Ind. Eng. Chem. Res.*, 58:3159–3169.

Sullivan, M., Zanganeh, B., Springer, A., and Clarkson, C. (2019). Post-fracture pressure decay: A novel (and free) stage-level assessment method. *SPE/AAPG/SEG Unconventional Resources Technology Conference, Denver, Colorado, USA*.

Switzer, S., Holland, W., Christie, D., Graf, G., Hedinger, A., McAuley, R., Wierzbicki, R., Packard, J., Mossop, G., and Shetsen, I. (1994). Devonian woodbend-winterburn strata of the western canada sedimentary basin. *Geological Atlas of the Western Canada Sedimentary Basin*, 4:165–202.

Tarrahi, M., Jafarpour, B., and Ghassemi, A. (2015). Integration of microseismic monitoring

data into coupled flow and geomechanical models with ensemble kalman filter. *Water Resources Research*, 51:5177–5197.

Valls, J., Tobías, A., Satorra, P., and Tebé, C. (2020). Covid19-tracker: a shiny app to analyze data on sars-cov-2 epidemic in spain. *Gaceta sanitaria*, 35(1):99–101.

Venieri, M., Weir, R., McKean, S. H., Pedersen, P. K., and Eaton, D. W. (2020). Determining elastic properties of organic rich shales from core, wireline logs, and 3-d seismic. *Journal of Natural Gas Science and Engineering*, 84:103637.

Wang, S. and Chen, S. (2019). Insights to fracture stimulation design in unconventional reservoirs based on machine learning modeling. *Journal of Petroleum Science and Engineering*, 174:682–695.

Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244.

Williams, G. J. (2018). *The Essentials of Data Science: Knowledge Discovery Using R*. Routledge, an imprint of the Taylor and amp; Francis Group.

Yang, F. and Ma, J. (2019). Deep-learning inversion: A next-generation velocity model building method. *Geophysics*, 84:R583–R599.

Yu, T. and Zhu, H. (2020). Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*.

Zanganeh, B., Clarkson, C. R., and Jones, J. R. (2018). Reinterpretation of flow patterns during dfits based on dynamic fracture geometry, leakoff and afterflow. *SPE Hydraulic Fracturing Technology Conference and Exhibition*.

Appendix A

DFIT Synthetic Parameters

#	File	WBS	NWT	S90	E0max	Shmin_Mpa	Pr_Mpa	Permeabilty_m2	Geomety	Res fluid
1 (base case)	B_ts	NA	0	7	0.00025	45	30	2E-19	Radial	Water
2	B_ts_NWT50	NA	50	7	0.00025	45	30	2E-19	Radial	Water
3	B_ts_NWT150	NA	150	7	0.00025	45	30	2E-19	Radial	Water
4	B_ts_NWT250	NA	250	7	0.00025	45	30	2E-19	Radial	Water
5	B_ts_S10	NA	0	10	0.00025	45	30	2E-19	Radial	Water
6	B_ts_S4	NA	0	4	0.00025	45	30	2E-19	Radial	Water
7	B_ts_S1	NA	0	1	0.00025	45	30	2E-19	Radial	Water
8	B_ts_E15	NA	0	7	0.00015	45	30	2E-19	Radial	Water
9	B_ts_E5	NA	0	7	0.00005	45	30	2E-19	Radial	Water
10	B_ts_E250	NA	0	7	0.0025	45	30	2E-19	Radial	Water
11	B_ts_E250_S10	NA	0	10	0.0025	45	30	2E-19	Radial	Water
12	B_ts_E250_S1	NA	0	1	0.0025	45	30	2E-19	Radial	Water
13	B_ts_E5_S10	NA	0	10	0.00005	45	30	2E-19	Radial	Water
14	B_ts_E5_S1	NA	0	1	0.00005	45	30	2E-19	Radial	Water
15	B_ts_WBS	Yes	0	7	0.00025	45	30	2E-19	Radial	Water
16	B_ts_WBS_NWT250_E250	Yes	250	7	0.0025	45	30	2E-19	Radial	Water
17	B_ts_WBS_NWT250_E5	Yes	250	7	0.00005	45	30	2E-19	Radial	Water
18	B_ts_WBS_NWT250_E250_PKN	Yes	250	7	0.0025	45	30	2E-19	PKN	Water
19	B_ts_WBS_NWT250_E250_g	Yes	250	7	0.0025	45	30	2E-19	Radial	Gas
20	B_ts_WBS_NWT250_E250_g_PKN	Yes	250	7	0.0025	45	30	2E-19	PKN	Gas
21	B_ts_pr33	NA	0	7	0.00025	45	33	2E-19	Radial	Water
22	B_ts_pr36	NA	0	7	0.00025	45	36	2E-19	Radial	Water
23	B_ts_sh42	NA	0	7	0.00025	42	30	2E-19	Radial	Water
24	B_ts_sh39	NA	0	7	0.00025	39	30	2E-19	Radial	Water
25	Simple_duvernay	Yes	250	7	0.00025	58	49	2E-19	Radial	Gas
26	Simple_duvernay_2	Yes	250	7	0.00025	58	49	2E-19	Radial	Gas
27	Complex durvernay 31_layers	Yes	250	7	0.00025	64	57	2E-19	Radial	Gas

Figure A.1: Input parameters for Resfrac synthetic modelling.

Appendix B

Clustering Methods

Table B.1: Hyperparamters for clustering methods

# Clustering method	Parameters
K-means	Number of clusters
DB-scan	Minimum points, Search radius
Hierarchical Clustering	Measure of distance between clusters, Number of clusters
Gaussian mixture model	Shape of probability distribution, number of clusters

Table B.2: Clustering methods

# Clustering method	Definition
K-means	Assigns clusters to centroids until stabilization
DB-scan	Groups points in densely packed, leaves outliers in low density areas
Hierarchical Clustering	splits clusters of similarity
Gaussian mixture model	Bayesian probability to assign clusters

Appendix C

Labeled Curves

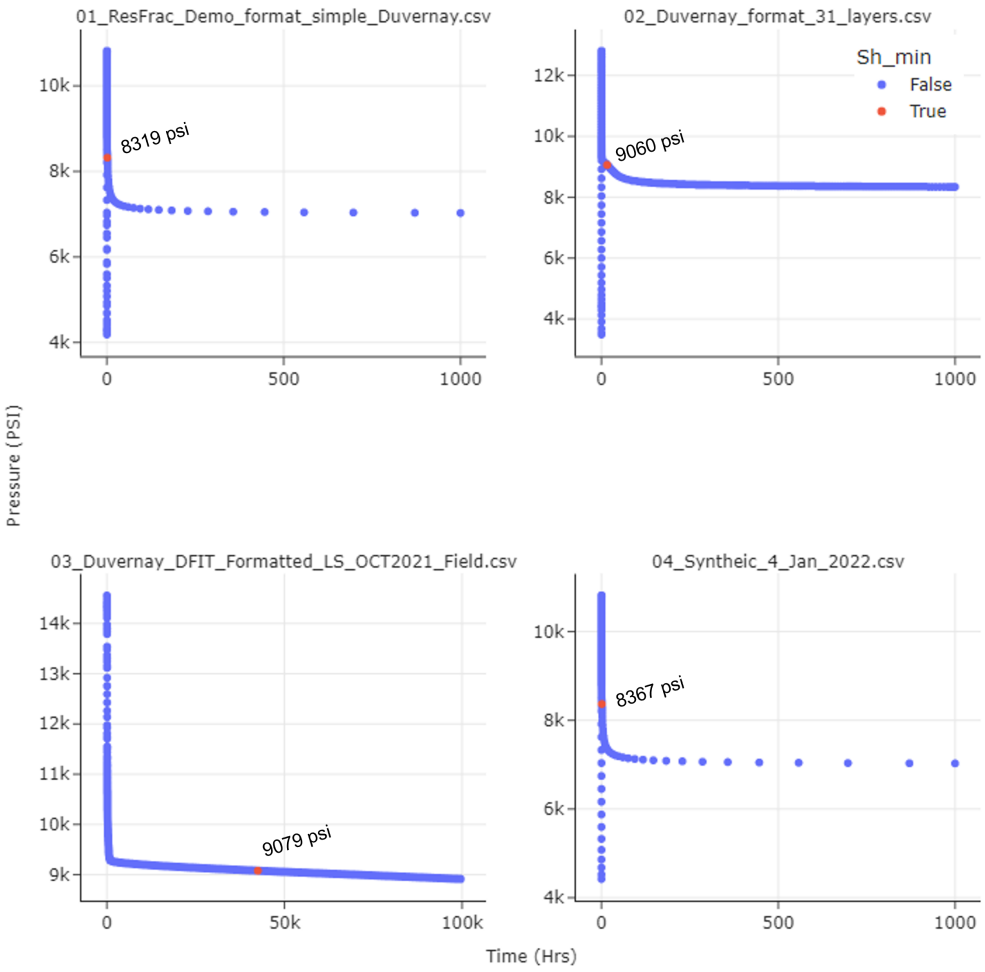


Figure C.1: Reference plot of each Duvernay testing curve with the actual Sh_{min} values labeled.

Appendix D

Library of the codes

The codes written for this thesis can be found at:

https://github.com/lukas-sadownyk/DFIT_ML_Library.git.