Western Australian School of Mines: Minerals, Energy and Chemical Engineering

Rule-based operator decision support for grinding circuit control

Jacques Olivier

0000-0002-4837-2175

This thesis is presented for the Degree of
Master of Philosophy
of
Curtin University

July 2021

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in my university.

Signature:

Date: 26/07/2021

Abstract

Grinding circuits are renowned for inefficient operation and disproportionate contributions to the energy expenditures on mineral processing plants. This, along with the pronounced effect of grinding on downstream beneficiation, has prompted considerable efforts from industry and academia to improve the operation of grinding circuits through effective process control. Consistent with the global trend towards automation, many practitioners are seeking to automate grinding circuit control using advanced process control systems such as model predictive controllers and expert control systems.

However, these efforts are complicated by the general complexity of the grinding task and frequent disturbances that cannot all be addressed during the controller design stage. Consequently, the implementation of automatic control systems has lagged other industry sectors, and advanced process control solutions are still not well established on most mineral processing plants. Therefore, grinding circuit performance is still largely dependent on operator decision-making processes through human supervisory control. Under these circumstances, there exists a strong motivation to provide operators with enhanced tools to support their decision-making, while still pursuing the longer-term goal towards automation.

Operator decision support for grinding circuit control may come in a variety of forms such as improved circuit visualisation tools, multivariate statistical process monitoring tools or rule-based decision support systems. The defining property of these decision support systems is that the information system supports the decision-maker, rather than replacing them. This results in a hybrid-intelligence control system, wherein decisions are made based on results of knowledge discovery processes from historical data, as well as the heuristic knowledge accumulated by human process operators.

In this work, machine learning methods were used to extract operating patterns from historical data. More specifically, decision tree algorithms were used to induce decision rules that could provide rule-based decision support to process operators. A methodology was presented which details the rule extraction procedure and metrics were proposed to analyse the utility of the decision rules for decision support. The methodology prioritises the

generation of small rule sets, containing interpretable and accurate rules that are sufficiently represented in the data set.

The methodology was applied to data sets from two industrial grinding circuits as case studies. The case studies demonstrated that decision trees are an effective approach to generating rule sets for operator decision support. The decision tree algorithms could induce intelligible rule sets without suffering excessive decreases in accuracy when compared with models of higher complexity.

The methodology presented requires expert knowledge of the grinding circuit under analysis rather than the data mining methods themselves and could easily be applied by plant metallurgists. Application of the rule sets in a decision support system could remove some of the arbitrariness associated with operator decision-making. This could lead to more efficient operation and significant financial benefits over longer-term periods on mineral processing plants.

Publications

The publications below were prepared because of this research. Attribution matrices for each of these works are included in Appendix A and Appendix B, respectively.

Refereed Journal Paper

Olivier, J. and Aldrich, C. (2021) Use of decision trees for the development of decision support systems for the control of grinding circuits. *Minerals* 11 (6), 595.

Refereed Conference Paper

Olivier, J., Aldrich, C., Shelley, P. and Davies, E. (2020) Decision support systems for SAG mill control: A case study. in Velasquez, C., Cisternas, L., Gutierrez, L., Jerez, O., Johnston, A., Kracht, W., and Kuyvenhoven, R. (eds) *2020 Procemin GEOMET*. Santiago, Chile: Gecamin Digital Publications, pp. 290–298.

The author presented the paper during the online conference.

Acknowledgements

I would like to express my appreciation to the following people with whom I have had the pleasure to associate with during the period of this work:

- My supervisor, Professor Chris Aldrich, for the invaluable guidance and advice throughout all stages of the project. I thank you for the introduction to this fascinating research area and the opportunity to work together to expand upon the field. Your efforts in the overall project administration are gratefully acknowledged.
- To Dr Paul Shelley and Eugene Davies from the Innovation team at Molycop, I am grateful for the practical experience and knowledge garnered throughout this project. The ongoing financial support from Molycop is sincerely appreciated. The perpetual pursuit of nice spots through the Murchison region of Western Australia have been a highlight throughout this experience.
- To friends and family whose support I could count on every step of the way.
- Finally, to my parents, for their role in allowing this opportunity to come to fruition and the unwavering support and encouragement throughout this period. I am eternally grateful to both of you.

Table of Contents

Nomenclature

AG	Autogeneous Grinding
ANN	Artificial Neural Network
APC	Advanced Process Control
CART	Classification and Regression Trees (algorithm)
DCS	Distributed Control System
DDC	Data-driven Control
DIKW	Data-Information-Knowledge-Wisdom
DNF	Disjunctive Normal Form
DSS	Decision Support System
ECS	Expert Control System
HMI	Human-Machine Interface
I.I.D.	Independent and Identically Distributed Random Variables
KPI	Key Performance Indicator
ML	Machine Learning
MPC	Model Predictive Control
MSE	Mean Squared Error
OOB	Out-Of-Bag
PCA	Principal Component Analysis
PID	Proportional-Integral-Derivative
PSD	Particle Size Distribution
RF	Random Forest
ROM	Run-of-mine
SABC	SAG, Ball mill and Pebble Crusher grinding circuit
SAG	Semi-autogeneous Grinding
SVM	Support Vector Machine

1. Introduction

1.1 Background

Comminution circuits have been estimated to account for between 2-3% of all electrical energy consumed on the planet (Fuerstenau and Abouzeid, 2002; Napier-Munn, 2015). This reportedly translates to between 30-50% of the energy consumption of mineral processing plants, depending on the hardness of the ore to be comminuted (Napier-Munn, Morrell, Robert, *et al.*, 1996). Grinding circuit installations have taken many different forms and configurations, but these circuits generally constitute the final stages of the comminution process. Regardless of the configuration, autogeneous grinding (AG), semi-autogeneous grinding (SAG), rod and ball mills in grinding circuits are usually the largest energy consumers on mineral processing plants.

However, grinding circuits are notorious for their inefficient operation and only a small fraction of energy consumed is utilised for the grinding task. Depending on the specific definition, the energy efficiency of grinding mills are estimated to range between 1-15% (Fuerstenau and Abouzeid, 2002). These inefficiencies arise because of a lack of guarantees that energy expended are utilised for the creation of new particle surfaces through breakage. Impact and attrition events are imparted almost randomly to the mill charge, breaking already liberated and unliberated particles alike (Wills and Napier-Munn, 2006). Large amounts of the energy imparted by the tumbling drum to the mill charge is also lost as heat and noise.

These statistics suggest that significant opportunities exist to improve the efficiency of comminution operations, especially through improvements to the grinding circuit. Current improvement efforts generally focus on the development of novel comminution machines with increased breakage efficiency, or more incremental improvements to current technologies through process control and optimisation. High pressure grinding rolls, (Schönert, 1988), have emerged as an alternative technology which has demonstrated increased energy efficiency in certain crushing-grinding applications (Danilkewich and Hunter, 2006). The conjugate anvil hammer-mill, (Nordell and Potapov, 2015), is another novel machine currently being prototyped, claiming to halve the energy of the grinding task (Li *et al.*, 2019). However, uptake of new technology is rather slow in the mineral processing

industry, and retrofitting existing operations are expensive. Thus, conventional grinding circuits will likely remain the dominant technologies for years to come. This has prompted significant industry and academic focus towards increasing grinding circuit efficiency through improved process control.

As with most industrial processes, grinding circuits are highly reliant on effective process control to achieve circuit and plant key performance indicator (KPI) targets. Historically, this has proved quite challenging, owed to the general complexity of the grinding task. It is well known that management of the mill load dynamics is key to efficient grinding circuit control (Nierop and Moys, 1996), yet this is hampered by the lack of direct online measurements of the load. Additionally, the mill load is constantly disturbed with varying ore compositions, competencies, and particle size distributions (PSD) in the mill feed. Changing feed stocks alter the mill load, ultimately affecting the grinding rate and grind size, which in turn again alters the load in a nonlinear feedback loop (Powell, van der Westhuizen and Mainza, 2009). Feed disturbances often go unmeasured, and their effect on the charge not always well understood. The intricacies of the grinding process are further exacerbated in AG/SAG mills, where the ore itself takes on the role of grinding media.

Throughout the industry, the majority of regulatory control functions on mineral processing plants, i.e. keeping process variables at specified set points, are executed by Proportional-Integral-Derivative (PID) control loops (Wei and Craig, 2009; Zhou *et al.*, 2016). PID controllers are generally coupled in single-input-single-output (SISO) configurations, with one manipulated variable controlling a controlled variable (Hodouin, 2011). However, these decentralised controllers are unable to account for the coupled interactions between individual variables, usually resulting in more conservative controller tuning (Hodouin, 2011). In other industries, these interactions are usually addressed with the addition of multivariable controllers or decouplers in the control architecture, derived from process models. However, the uncertainties associated with the complexities of the grinding task and the practically chaotic system found in the mill charge have hampered the development of reliable mathematical models.

Supervisory control functions, such as determining set points and enforcing adherence to process constraints, are entrusted to either advanced process control (APC) systems or human process operators. APC systems attempt to address the coupled, nonlinear

interactions between variables and predict the optimal control response in an automated fashion. APC systems can be broadly characterised into model-based, intelligent or hybrid systems. Since their introduction in the 1970's, significant strides have been made in the use of model-based systems such as model predictive controllers (MPC) (Muller and De Vaal, 2000; Ramasamy, Narayanan and Rao, 2005; Botha, le Roux and Craig, 2018); and intelligent systems such as expert control systems (ECS) (Bearman and Milne, 1992; Chen *et al.*, 2009; Hadizadeh, Farzanegan and Noaparast, 2018). These systems have demonstrated successful control near nominal operating conditions through numerous case studies. However, these systems do not always achieve satisfactory results when facing heavy disturbances, the majority of which cannot be addressed during the controller design phase (Zhou *et al.*, 2016). Additionally, MPC parameter tunings are model fit during circuit surveys in an environment wherein ore type and PSDs continuously vary over both short – and long term. Implementations of APC systems are further hampered due to the fact that many important process variables are still measured manually by process operators (Olivier and Craig, 2017).

Alternatively, data-driven control (DDC) methodologies have been receiving increasing attention in the minerals processing industry (Ding, Yang and Chai, 2017). In contrast to MPC and ECS, data-driven models learn by example from historical process data through machine learning (ML), instead of domain knowledge. Unlike first principle and empirical models, data-driven models are typically non-parametric, making them increasingly attractive when process mechanisms are not well understood. Recently, large amounts of research have focused on implementations of data-driven models in minerals processing, usually as soft sensors or machine vision systems, which are used as input to existing ECSs or other APCs (Zhou *et al.*, 2016; McCoy and Auret, 2019). Fully data-driven controllers have yet to obtain widespread acceptance in the industry, but have found some success in academic literature (Conradie and Aldrich, 2001; Zhou, Chai and Sun, 2013).

Despite the recent progress in instrumentation and automation, operators still play a critical role in the operation of many mineral processing plants. A recent survey by Olivier and Craig (2017) found that process operators still frequently intervene in the process with important actions. Many of these actions concerned supervisory control functions, such as adapting the process to frequently changing operating conditions. Additionally, the survey found that operators frequently disable controllers, owed to poor controller performance in the face of

large disturbances (Hodouin, 2011; Olivier and Craig, 2017). Thus, in many cases the performance of the processing plant remains largely dependent on the operator decision-making process. Li *et al.*, (2011) also remarked that in their experience, increased automation in control systems did not minimise the contribution of human operators, but rather increased the workload and heightened the skills required from operators. In such an environment, there is a strong motivation for development of improved decision support structures for the operator decision-making process, while remaining in pursuit of the longer-term goal of fully automatic control systems.

Decision support systems (DSS) are information systems that support decision-makers in decision-making activities. In multiple highly-cited surveys, Eom *et al.*, (1998, 2006) defines DSS with the following:

> A DSS is defined as a computer-based interactive system that support decision-makers rather than replaces them; utilizes data and models; solves problems with varying degrees of structure; and focuses on the effectiveness rather than the efficiency of decision processes (facilitating decision processes).

Accordingly, DSS consist of databases, a model, and a human-machine interface (HMI). Thus, DSS are very reliant on the heuristic or sometimes tacit knowledge of the decision-maker and retains the analytical and problem-solving ability of human operators. Surveys have found that most DSS applications in academic literature focus on the corporate functional management areas, such as production and marketing (Eom *et al.*, 1998; Eom and Kim, 2006). Applications in the resources areas were encountered much less frequently.

Operator decision support may come in a variety of forms. For grinding circuit control these may include, but are not limited to, providing increased circuit visualisation tools (Powell, van der Westhuizen and Mainza, 2009; Aldrich *et al.*, 2014), multivariate statistical process monitoring tools (Choi and Lee, 2004; Groenewald, Coetzer and Aldrich, 2006; Olivier and Aldrich, 2020) and rule-based DSS (Saraiva and Stephanopoulos, 1992; Aldrich *et al.*, 1997). These methods attempt to extract knowledge from raw data and present the knowledge to operators in an explicit manner. In the past, this could be done through manual analysis and interpretation of operational data, but the sheer volumes of data collected on modern processing plants render this impractical. ML methods provide a powerful alternative to the knowledge discovery process (Fayyad, Piatetsky-Shapiro and Smyth, 1996). ML methods, such

as classification, regression, and clustering models, are particularly useful in extracting patterns from large amounts of data, which are sometimes imperceptible to humans.

1.2 Objectives of the study

In this work, rule-based ML methods are used to extract operating patterns and knowledge from grinding circuit data. Specifically, decision tree models such as those proposed by Breiman *et al.*, (1984), are induced from the data during classification and regression tasks. Rules are extracted from the trees and represented as "if-and-then" decision rules. Specific focus was given to extracting a small amount of relatively accurate rules, with high amounts of support in the data set, and rules that are short enough to easily be interpreted by humans. The knowledge inferred through the rules could be used either in process improvement operations or as a form of rule-based decision support for operators in the control room.

Application of the methodology is demonstrated in the form of case studies on data sets obtained from industrial grinding circuits. The rule sets extracted in each case study were examined, and rules deemed of high utility were analysed for their practical application as operator decision support. Since current operator practices are naturally embedded in the process data, the methodology could discover common operator decision-making patterns, and formalise these patterns mathematically. Formalising these patterns could result in less variability in the operator decision-making process, possibly leading to increased production and stability in the grinding circuit.

The following specific objectives were pursued to achieve the research aim outlined above:

a) A critical literature review on current knowledge and knowledge discovery methods present in the minerals processing industry, concentrating on applications in grinding circuits. Specific focus was given to rule-based ML methods and their applicability for decision support
b) Presenting a methodology to extract rules from decision trees suitable for interpretation by human operators, as well as a method to evaluate the utility of rules extracted.
c) Demonstrating the rule extraction procedure on representative, industrial grinding circuits and analysing the knowledge contained in the rules

1.3 Thesis outline

Section 2 provides a broad review of the literature on the current control landscape encountered in mineral processing plants, with a specific focus on the grinding circuit. Current trends in the literature are identified, and the role of human operators are explored.

Section 3 provides an overview of decision trees and their induction using decision tree algorithms. A methodology for the induction of rules interpretable by humans is presented and statistics to evaluate the utility of extracted rules are introduced.

Sections 4 and 5 present two case studies demonstrating the application of the rule extraction methodology to data sets from two different industrial grinding circuits. Rules with high utility are analysed and their practical implications are discussed.

Chapter 6 contains some general discussions on the methodology and summarises the conclusions drawn from the study.

2. Knowledge discovery in mineral processing systems

This section explores the sources of data and information in mineral processing systems, and how knowledge is extracted from such sources for process control. Specific focus is given to data-mining methods with explicit knowledge representations, such as rule-based algorithms, and how these methods can support human decision-making processes. Where possible, the scope of the literature reviewed is limited to grinding circuits or the broader mineral processing industry. Finally, the section attempts to contextualise the role of the human process operator among process control tools and how rules extracted from decision trees could aid the operator as part of a DSS.

2.1 From data to wisdom in mineral processing systems

The concept of knowledge is extremely complex and remains the subject of vigorous debate among philosophers and academia. Knowledge is usually defined in terms of information and data; a relationship contextualised by the famous data-information-knowledge-wisdom (DIKW) hierarchy. This hierarchy is commonly represented as a pyramid, as shown in Figure 1. Ackoff (1989) offers the following definition of knowledge, which is frequently cited alongside descriptions of the DIKW hierarchy:

> Knowledge is know-how, and is what makes possible the transformation of information into instructions. Knowledge can be obtained either by transmission from another who has it, by instructions, or by extracting it from experience.

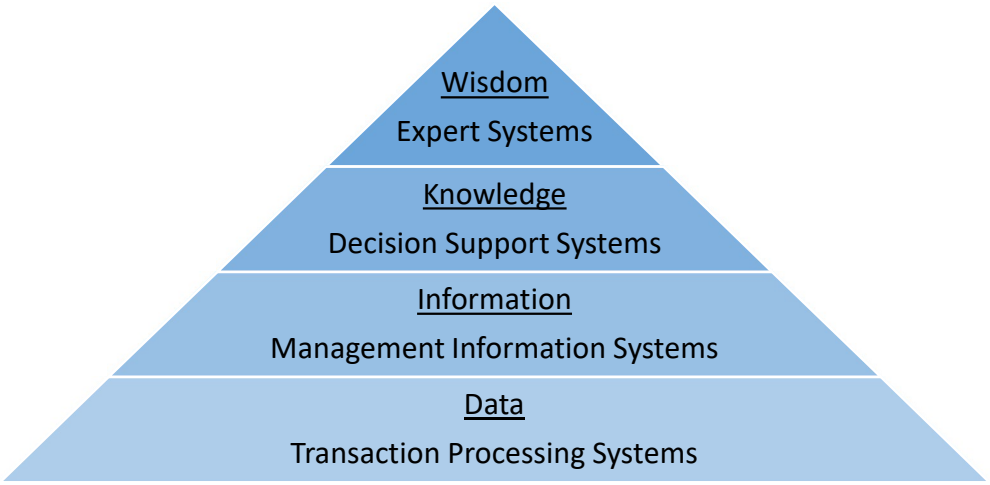

Figure 1: DIKW hierarchy, with different levels of information systems mapped into the hierarchy

Descriptions of the bottom two levels of the hierarchy, data and information, share more agreement between practitioners in the knowledge management field (Rowley, 2007). Data is seen as mere observations, without structure or context; whereas information is data transformed into a functional form, able to answer specific questions. From Ackoff's definition, we could interpret knowledge as a further transformation of information into sets of instructions, which could be actionable in certain cases. Rowley (2007) noted that the notion of wisdom is mostly not defined in information systems literature. This could be a result of the breadth of the fields to which the concept is applied. To provide an example, Awad and Ghaziri (2004, p. 40) suggested that "Wisdom is the highest level of abstraction, with vision foresight and the ability to see beyond the horizon".

Rowley (2007) also noted that the DIKW hierarchy could aid in role descriptions of different types of information systems, such as the systems defined in more classical literature on the subject (Laudon and Laudon, 1988). These definitions are included on the pyramid in Figure 1, since the hierarchy is somewhat analogous to the information systems available to process operators, metallurgists and control systems in mineral processing plants. These concepts are explored in Figure 2, and each layer is addressed individually below.

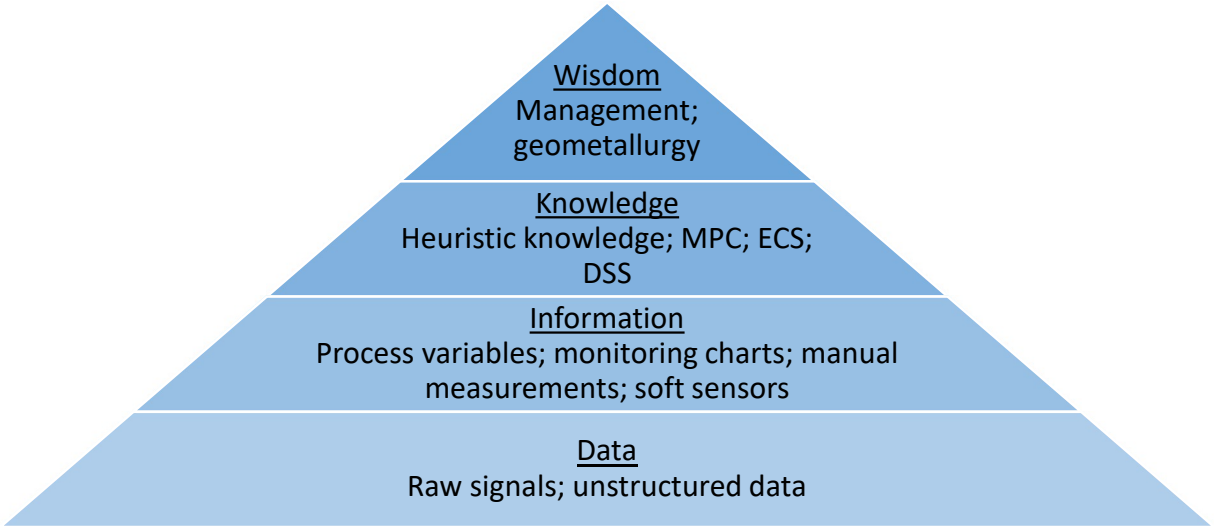

Figure 2: DIKW hierarchy of information systems in grinding circuits

All raw signals generated exist at the bottom level, the data level, of the hierarchy. Raw digital signals are collected from magnetic flowmeters, conveyor belt weightometers, load cells, density gauges etc. Sounds and vibrations emitted by the mill shell are often measured with microphones and accelerometers. Unstructured data sets, such as pictures and video feeds,

are also increasingly encountered. These data collectors can be found near feed conveyor belts or the hydrocyclone underflow to determine PSD estimates.

The raw digital signals are usually of high frequency and bear little relation to the physical phenomena observed around the circuit. To increase the utility of data collected for end-users, the data needs to be transformed into information. Digital signals need to be calibrated and transformed into meaningful process variables including flow rates, conveyor tonnages, mill load tonnages, bearing pressures and densities. Microphone and accelerometer measurements are processed into variables describing the impacts inside the mill, sometimes portrayed through an indication of the mill charge toe and shoulder positions (Gugel and Moon, 2007; Pax and Cornish, 2015). While video camera feeds are certainly useful to an extent simply through visual inspection, computer vision algorithms can transform the image contents into interpretable statistics useful for monitoring. Such applications include characterisation of the mill feed PSD into distinct classes (Olivier, Maritz and Craig, 2019) or automatic detection of the operational state of a hydrocyclone cluster (Giglia and Aldrich, 2020). All the information sources mentioned can be provided to either process operators, in the form of monitoring charts, or control systems to make control decisions.

At the knowledge level, information is transformed into sets of instructions, and decisions regarding the next control action is taken. Process operators use the information provided in monitoring charts, along with their own heuristic knowledge gained through experience, and make a subjective assessment of the best course of action. Automatic controllers rely on a knowledge base (ECS) or process model (MPC) pre-programmed into the system to determine whether a control action is required.

DSS are situated somewhere between the two extremes of manual and fully automatic control. Operators are provided with knowledge extracted from data to augment their own existing knowledge base, but the operator remains the final decision-maker. Thus, DSS takes on the form of a hybrid intelligence model, drawing on both the heuristic and tacit knowledge of human operators, as well as hidden knowledge patterns extracted through data mining methods. However, the reliance on human operators does introduce a certain arbitrariness that is not present in automatic systems. Human operators make subjective assessments of the circuit, and there are bound to be different evaluations between different individuals, or

different evaluations of similar operational states. A DSS needs to be designed to remove as much of this arbitrariness as possible.

Finally, the top level of the hierarchy in Figure 2 combines all of the previous levels into wisdom. Like the knowledge management and information sciences, this concept has been less studied in mineral processing literature. Generally, this level of abstraction is believed to be non-automatable and could only ever be performed by humans. Ding, Yang and Chai, (2017) and Zhou *et al.*, (2016) have proposed similar hierarchies wherein these top levels fall outside the scope of technical engineers, but rather to planning and scheduling departments or executive management.

To contextualise this notion of wisdom to the level of the technical engineer, it could be postulated that this top level encompasses the realm of geometallurgy. Metallurgists, process managers, and geologists use foresight to determine which ore sources or blends would allow them to achieve operational targets set by executive management. This type of evaluation requires expert knowledge of plant operation and thorough understanding of how the plant reacts to different ore sources. This type of wisdom would be difficult to formalise and automate since it relies heavily on the tacit knowledge and instincts of the expert.

2.2 Knowledge discovery methods in mineral processing systems

Effective process control requires knowledge built on data and information, as well as previous knowledge of the fundamental principles governing plant operation. This practice of finding existing knowledge and patterns from physical systems is often called knowledge discovery. While this term has recently become more synonymous with knowledge discovery through data (KDD) (Fayyad, Piatetsky-Shapiro and Smyth, 1996), all practitioners (manual and automatic) need to undergo some process of knowledge discovery to effectively control mineral processing systems. These processes for different control practitioners are depicted in Figure 3. Each of the knowledge discovery methods are briefly discussed below in the context of grinding circuit control.

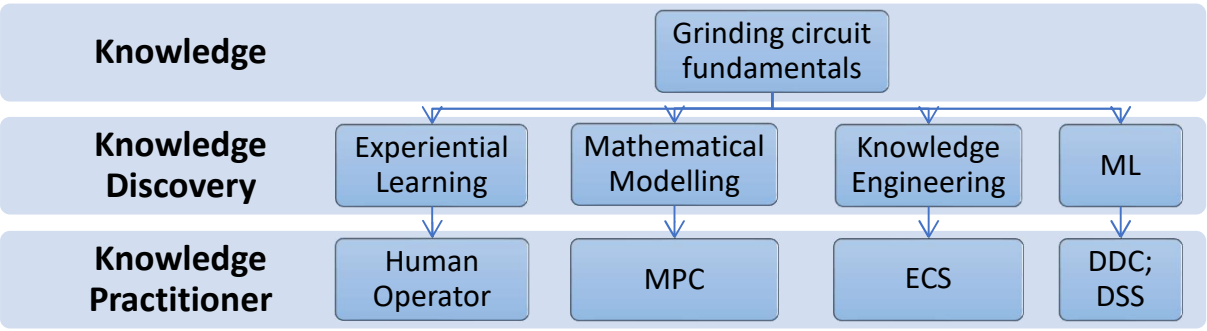

Figure 3: Knowledge discovery methods for different types of control practitioners

Human control room operators learn to control the grinding circuit through experience. Li *et al.*, (2011) noted that operators mostly receive hands-on training from more experienced operators inside the control room. Training literature is often provided with an introduction to the fundamentals of the process and common control actions, and the reasoning behind the actions, are presented in flowcharts. Through trial and error, operators accumulate experience and build up their own heuristic knowledge base. However, there are often great disparities between the experience levels of different operators; and many operators do not acquire adequate knowledge of the process and control systems during their time working on a mineral processing plant (Li *et al.*, 2011). Systematic training procedures could be of significant benefit to aid in overcoming some of these disparities.

For MPC, mathematical – and process models facilitate the knowledge discovery process. Controller designers derive mathematical models describing the grinding process, and model parameters are usually fitted to data from plant surveys. Despite the general complexity of the grinding task, MPC have made significant strides towards robust control of grinding circuits (Muller and De Vaal, 2000; Coetzee, Kerrigan and Craig, 2010; Salazar *et al.*, 2014). However, in contrast to areas such as the petrochemical industry, these methods have not reached widespread use in minerals processing (Olivier and Craig, 2017). This is generally attributed to the unsatisfactory performance of these algorithms in the presence of continuous, large disturbances (Zhou *et al.*, 2016). The MPC schemes are also hampered by the large process time lags inherent to the milling process (Coetzee, Kerrigan and Craig, 2010). Maintenance of the controller in the face of longer-term process deviations are also complex and requires additional work from controller designers.

ECS, sometimes referred to as knowledge-based intelligent control systems (Linkens and Minyou Chen, 1995), generally consist of a knowledge base and an inference engine. The knowledge base usually consists of sets of "if-then" rules, extracted during knowledge engineering (Åström, Anton and Årzén, 1986; Johannsen and Alty, 1991). During knowledge engineering, the "if-then" rules are obtained directly from the heuristic knowledge of process operators or experts, such as metallurgists. The process is generally facilitated through interviews, questionnaires or observation of the operation by the knowledge engineer (Johannsen and Alty, 1991; Sloan *et al.*, 2006). The inference engine receives real-time plant data and searches for applicable control rules to implement automatically. Thus, ECS attempt to model the knowledge and procedures used by human experts during decision-making processes. ECS are viable alternatives in areas where problems are more easily solved by knowledge and reasoning, rather than analytical solutions (Linkens and Minyou Chen, 1995). ECS have demonstrated various successful applications for supervisory grinding circuit control (Bearman and Milne, 1992; Sloan *et al.*, 2006; Chen, Li and Fei, 2008; Chen *et al.*, 2009; Hadizadeh, Farzanegan and Noaparast, 2018).

The success of ECS are highly reliant on the knowledge engineering process and the resulting knowledge base. Clancey (1986) remarked on the difficulty of the knowledge engineering process since it does not just involve transferring knowledge from experts into a knowledge base, but also involves the formalisation of very complex domains. These struggles arise from the fact that the experts remain human. Experts are known to sometimes have faulty memories, provide inconsistencies and suffer cognitive biases such as overconfidence in their analysis (Johannsen and Alty, 1991). Additionally, some expertise such as the expert's tacit knowledge, are very difficult to elucidate and in some fields there are no experts to generate an effective knowledge base (Åström, Anton and Årzén, 1986).

ML tools, such as those used in DDC and DSS, are an alternative method to extract knowledge from process data through learning algorithms. Since data is generally an unbiased realisation of physical phenomena, it contains a wealth of knowledge hidden as patterns. While human operators or experts easily identify some of these patterns, ML algorithms can unveil more complex patterns and inferences, well beyond that of human capabilities. While these methods have not been yet been widely adopted in mineral processing plants (Wei and Craig, 2008), several successful machine learning based control methodologies have been reported

in literature (Zhou, Chai and Sun, 2013; Zhou *et al.*, 2016). However, these methods would likely encounter the same problems as other APC methods in unfamiliar situations.

Instead of developing fully DDC solutions, ML models are more often used in the generation of new sources of information for operator decision-making or existing APC systems. ML models have been used for many soft sensor calculations, as well as the processing of unstructured data sources such as acoustic measurements and video camera feeds (McCoy and Auret, 2019). Since this work focuses on the use of ML tools to build DSS for process operators, the knowledge discovery processes in these methods are further discussed in the next section.

2.3 Data-driven knowledge discovery in mineral processing systems

2.3.1 Knowledge discovery goals and knowledge representations

The goal of knowledge discovery in data sets are usually predictive, descriptive, or a combination of both (Fayyad, Piatetsky-Shapiro and Smyth, 1996). Predictive goals will focus on generating models with the highest accuracy in future predictions. If it has sufficient predictive power, the models do not necessarily need to be grounded in reality or connected to observable physical phenomena. For example, autoregressive models of financial time series data can be very accurate but provide limited knowledge of the data-generating process. Such models would be of limited use in control applications, since predictions cannot be related to the specific variables causing deviations, and thus do not result in suggestions for control action. Contrasting to this, descriptive goals result in models that build patterns and inferences from real physical observations and are interpreted as a reflection of reality. With descriptive goals in mind, the representation of knowledge captured in these models are of critical importance.

Knowledge representations are commonly classified into implicit – or explicit representations (Nonaka and Takeuchi, 1995; Bandaru, Ng and Deb, 2017). Explicit knowledge representations have an unambiguous mathematical notation enabling knowledge to be expressed concisely (Faucher, Everett and Lawson, 2008). Implicit knowledge representations lack this formal mathematical notation. This implies the knowledge contained cannot be articulated unambiguously, thus requiring specific experience to understand and enabling subjective interpretations. This classification is demonstrated for five types of ML methods in Figure 4.

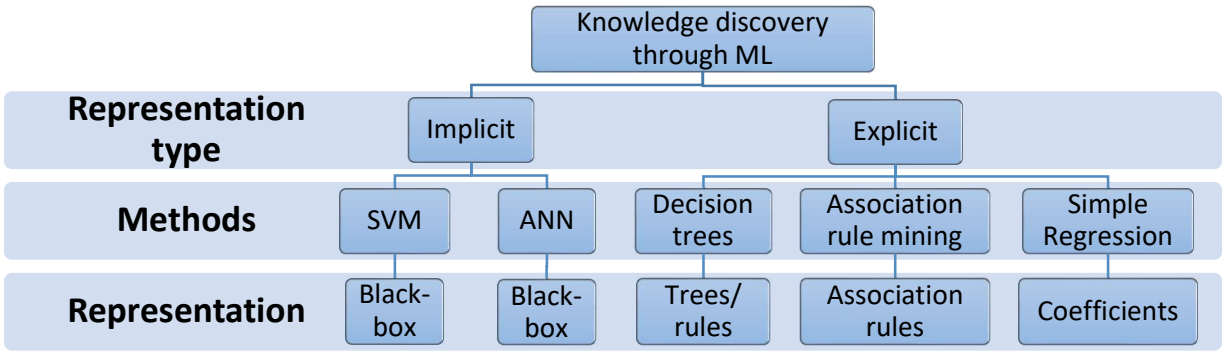

Figure 4: Examples of ML knowledge discovery methods and model representation

McCoy and Auret (2019) categorised ML methods in mineral processing systems according to the specific application of the methods. The applications most frequently encountered were data-based modelling, fault detection and diagnosis, and machine vision. The discussion here is focused on data-based modelling since this is one of the main tools for knowledge discovery from data. In mineral processing systems data-based modelling is often applied to produce "soft sensors", where by frequent plant measurements are used to predict certain variables which are difficult to measure or often remain unmeasured (McCoy and Auret, 2019).

The following subsections explore some data-based modelling methods applied in the mineral processing industries according to the representation of knowledge in the models, as depicted in Figure 4.

2.3.2 Implicit knowledge representations and black-box models

Figure 4 indicates that support vector machines (SVM) and artificial neural networks (ANN) are included among ML models with implicit knowledge representations. These models are commonly referred to as black-box models, since the process of transformation from inputs to outputs is obfuscated, and generally incomprehensible. The model parameters are sets of weights that facilitate this transformation from input to output. The values of the weights are determined during a learning stage, whereby the weights are optimised according to some loss function. For a classification task, this loss function maximises the separation between classes in a SVM, while neural networks are trained to minimise the prediction error.

SVMs and ANNs have found numerous successful implementations in grinding circuit control applications. Stange (1993) used ANNs to predict the hydrocyclone overflow particle size in a

grinding circuit controller. Conradie and Aldrich (2001) developed an ANN controller using evolutionary reinforcement learning that managed robust performance in the grinding circuit, even in the presence of large disturbances. Aldrich *et al.* (2014) used ANNs to predict the mill loads while Bardinas, Aldrich and Napier (2018) demonstrated how ANN's could be used to identify the operational states in grinding circuits. Zhao *et al.* (2010) proposed a SVM based soft sensor to predict the mill load from vibration frequencies. Jemwa and Aldrich (2006) used one-class SVMs to estimate nonparametric confidence limits for statistical process control methods on mineral processing plants.

While these methods have proved successful in automatic control applications, the obfuscation of knowledge as weights in the models is undesirable for DSS. Aside from SVMs and ANNs, visual data-mining methods such as principal component analysis (PCA), clustering diagrams, linear discriminant analysis and multidimensional scaling etc., also demonstrate implicit knowledge representations. These methods are capable of summarising and displaying high dimensional systems on two or three-dimensional plots. While such methods are more appropriate for DSS, these graphics generally require experience to interpret, and are often difficult to connect to a specific control action.

2.3.3 Explicit and implicit regression models

Figure 4 includes simple regression models as data-based modelling methods with explicit knowledge representations. Parametric regression models such as polynomial or Gaussian regression models use explicit mathematical models and a formal notation that is interpretable. The knowledge discovered through these models are represented explicitly through model coefficients. The coefficients can be interpreted as the relative contributions of various variables to the model, and their respective influences on the output. However, this is only true for lower complexity models such as linear, exponential or simple polynomial regression where the number of contributing terms does not obfuscate the input to output mapping.

These types of regression models with explicit notations have had widespread application in the minerals processing industry. Xiao (2001) developed simple multivariable regression models to predict the recovery of gravity recoverable gold from grinding circuits. The models could predict the recovery well within the precision limit imposed by physical measurements. Casali *et al.* (1998) developed a nonlinear regression model with ARMAX (autoregressive

moving-average with exogenous inputs) structures in a soft sensor application to predict the product PSD from a grinding circuit. Similarly, Minchala-Avila *et al.* (2016) used multivariable linear regression to predict the product PSD in cement grinding

The models produced in these examples are much more interpretable than regression models built from SVMs and ANNs, wherein the knowledge is stored implicitly. More complex models with higher capacity (ANNs, SVMs, random forests) can be more accurate, but the increased model capacity is hard to represent in an intelligible manner to humans. This trade-off is commonly encountered in data science applications.

The general trend in industry and academia seems highly favoured towards sacrificing model representation for increased accuracy. More recently, increasingly complex models such as recurrent neural networks (Inapakurthi, Miriyala and Mitra, 2020), long-short term memory networks (Miriyala and Mitra, 2020) and gated-recurrent units (Avalos, Kracht and Ortiz, 2020) have been applied for regression tasks pertaining to grinding circuits. These studies successfully demonstrate the applications of novel deep learning methods to mineral processing systems. However, results are often not compared to simpler models, making it difficult to grasp the improved performance because of these complex methods. Studies that have managed to compare neural networks to methods such as simple linear and nonlinear regression have not confirmed significant benefit using higher complexity models (Massinaei and Doostmohammadi, 2010; Nakhaei *et al.*, 2012; McCoy and Auret, 2019).

2.3.4 Rule-based representations

Rule-based models are another powerful alternative resulting in explicit model representations. Decision tree algorithms and association rule mining, as shown in Figure 4, are two examples of ML algorithms that deliver rule-based knowledge representations.

Rules are popular knowledge representations in knowledge-based systems, including DSS (Cios, Pedrycz and Swiniarski, 1998; Eom and Kim, 2006). Rules also form the knowledge base in ECS, although these rules do not need to be interpreted by humans. Rule sets, along with inductive logic programming, fuzzy logic and case-based reasoning form part of the broader group of symbolic artificial intelligence applications. Their main advantages are their modularity and readability. Rules are easily interpretable since they are considered a visual

representation of human reasoning and decision-making processes. Rules can be added and deleted from a knowledge base quite easily, making a rule set highly modifiable.

Rule-based algorithms are generally quite versatile and can present decision rules with different levels of granularity. Numeric descriptors, such as those generated by regression tree algorithms, represent the highest level of granularity (Cios, Pedrycz and Swiniarski, 1998). The level of granularity can be reduced by applying rules to describe a system over pre-defined intervals, or further combined with fuzzy set theory (Zadeh, 1979) or rough sets (Pawlak, 1984).

The next section considers different approaches to rule induction in more detail. The applicability of each of the methods for decision support is also assessed.

2.4 Approaches to rule induction

Rule-based inductive learning models are expressed as either rule sets or decision trees. Decision rules generated from rule induction algorithms and decision tree models are usually expressed in some variant of the form of equation (1) below, where the # indicates a relational operator. The rules consist of a sequence of statements combined by logical operators. The type of rule induction algorithm determines the relational operators possible between attributes and values. A brief description of popular ML rule induction algorithms is given in the subsections below.

$$\begin{aligned} &IF\ (attibute_1\ \#\ value_1)\ and\ (attribute_2\ \#\ value_2)\ and\ \dots \\ &\dots and\ (attribute_n\ \#\ value_n); THEN\ (decision\ \#\ value) \end{aligned} \tag{1}$$

2.4.1 Classical rule induction algorithms

There exists a large range of rule induction algorithms such as the LEM1 and LEM2 algorithms, from the LERS (Learning from Examples using rough sets) data mining system (Grzymala-Busse, 1992), and the AQ family of algorithms (Michalski, 1969; Cios, Pedrycz and Swiniarski, 1998). These types of algorithms extract rules from data, usually represented as decision tables, using concepts from rough sets and variable-value logic calculus (Maimon and Rokach, 2005, chap. 13). However, this family of algorithms exclusively deal with ordinal or symbolic datatypes, and numerical attributes are required to be discretised before rule induction. This necessarily implies that these algorithms are not suited to regression tasks. An example of a

rule set generated using LEM1 to identify flu in medical patients (Maimon and Rokach, 2005, chap. 13), is given below.

$$(Temperature, very\ high) \rightarrow (Flu, yes)$$
$$(Nausea, yes) \rightarrow (Flu, yes) \quad (2)$$

Rule sets generated by these types of algorithms have been used in rule-based expert systems (Maimon and Rokach, 2005, chap. 13) to classify new, unseen cases. Some of these expert classification systems order the rules in a decision list, and the first rule matching a new case classifies the outcome (Rivest, 1987). In LERS, these rules are ordered according to their strength, specificity, and support. The rule strength indicates the number of cases correctly classified by the rule during training, while specificity refers to the number of attribute-value pairs in the antecedent (left-hand side) of the rule. The support of a rule is the product of its strength and specificity (Grzymala-Busse, 1992).

2.4.2 Association rule mining

Association rule mining constitutes another class of rule induction algorithms, first proposed by Agrawal, Imielinski and Swami (1993) in relation to "market basket analysis". In this work, the authors attempted to mine customer transaction data, to find product associations such as "90% of customers who purchase bread and butter also purchase milk" (Agrawal, Imielinski and Swami, 1993), as proposed by the rule in equation (3) below. The utility of association rules are evaluated according to the confidence and support of the rule. The support of a rule is defined as the fraction of data samples that satisfy the union of items in the antecedent and consequent of the rule (second number in curly braces in equation (3), 20%). The confidence in the rule is denoted by the 90% mentioned above, i.e. 90% of cases including the items in the antecedent, also include the items in the consequent of the rule.

$$[Bread, butter] \Longrightarrow [Milk]\ \{90, 20\} \quad (3)$$

The main difference between association rules and general decision rules is that both sides of the rule can contain any attributes from the original data set. Association rules are not concerned in predicting any single target attribute and focus on finding interesting relations between all attributes. For this reason, association rule mining is considered an unsupervised learning task to find patterns between variables in a data set.

Association rule mining algorithms have not been reported for classical DSS, but have been implemented as data-mining tools for APC systems in mineral processing plants. Abou and Dao (2009) have implemented association rule mining algorithms to find the optimal membership functions in a fuzzy logic controller. The controllers were reported to improve nominal performance of grinding circuits, but could not guarantee stability in the circuit (Abou and Dao, 2009, 2010). Ding *et al.*, (2012) proposed a knowledge-based feedback regulation for plant-wide optimisation and control. Rough-set theory was used to select appropriate control variables, and association rule mining was used to discover rules from operational data of the sets of variables. The rules were successfully implemented as decision support for plant-wide supervisory controllers (Ding *et al.*, 2009, 2012).

2.4.3 Decision rules from decision tree algorithms

The family of decision tree algorithms constitute another set of methods for rule induction. Decision trees are one of the most popular methods to explicitly represent patterns and knowledge. Decision tree algorithms recursively partition the input data space, allowing the knowledge to be represented in a hierarchical tree structure, as shown in Figure 5. The root nodes and internal nodes represent a test on a specific attribute, with the branches from the node representing the outcome of the test. Each leaf node provides a prediction for a specific pathway through the tree.

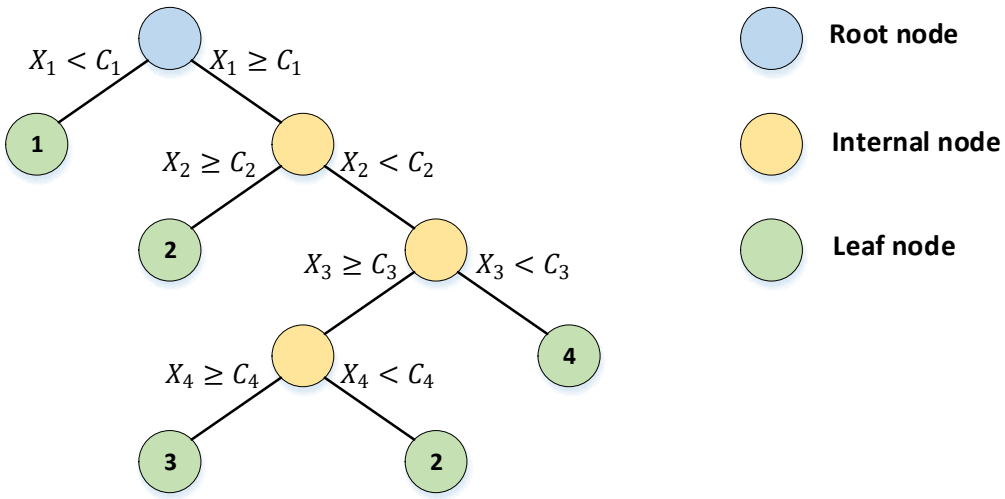

Figure 5: General classification tree diagram

Decision tree induction is closely related to the rules induced by the previously mentioned algorithms. Rules are readily extracted from decision trees by following the path from the

root node to the prediction in the leaf node. An example rule from the tree in Figure 5 is extracted in equation (4). A major difference between decision trees and general rule induction algorithms is that decision trees induce a disjunct set of decision rules, wherein the relationships between different rules are revealed through the tree structure. Most decision tree algorithms are capable of handling numerical data directly, and some can even be extended to regression tasks.

$$IF\ (X_1 \geq C_1)\ and\ (X_2 \geq C_2); THEN\ (Class = 2) \tag{4}$$

The most popular decision tree induction algorithms, such as ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993) and CART (Breiman *et al.*, 1984), construct decision trees in a top-down, recursive manner. During each iteration of the recursive process, the algorithms partition the data to satisfy some criterion in both resultant partitions. This criterion might be the largest reduction of impurity, the largest amount of information gain or other distance measures and depends on the specific algorithm used.

Decision trees and rules have had success in DSS in various industries. For example, Leech (1986) developed a knowledge base from rules derived from decision trees to predict pellet quality of uranium dioxide powders for nuclear fuels. De Oña, López and Abellán (2013) applied decision trees to traffic reports to identify common conditions leading to road accidents. Saraiva and Stephanopoulos (1992) proposed an on-line DSS using decision trees for process improvement in the processing industries.

Decision trees have also found some implementation in the mineral processing industry. In Gouws and Aldrich (1996) and Aldrich *et al.*, (1997), decision tree algorithms were used to induce rules tasked to classify the state of surface froths in flotation cells. Both studies found that the decision rules could classify the froths at least as well as human experts, and achieved results comparable to that of higher capacity models such as ANNs. Reuter *et al.*, (1998) developed a rule base to predict the manganese grade in an alloy from slag characteristics in a ferromanganese submerged-arc furnace. Applications to grinding circuits have not been encountered in available literature.

2.4.4 Other intelligent rule induction methods

Decision trees and rules have also been extracted through other intelligent algorithms. Schmitz, Aldrich and Gouws (1999) proposed a novel algorithm extracting decision trees from ANNs, called ANN-DT. The algorithm displayed favourable results compared to classical decision tree algorithms, such as CART, in a variety of applications.

Evolutionary computing, such as genetic algorithms, provides another avenue to rule induction. Explicit rule structures are encoded into genes, usually through a binary representation. Based on the principles of natural selection, these encoded rules are adaptively optimised to satisfy some fitness function through reproduction, crossover and mutation operations. Decision rules extracted from genetic algorithms (Schultz and Grefenstette, 1990; Carvalho and Freitas, 2004; Tseng *et al.*, 2008) and genetic programming applications (Benyahia and Potvin, 1998; Fallah-Mehdipour, Bozorg Haddad and Mariño, 2013) have been utilised successfully in various fields. Pertaining to minerals processing, Aldrich, Schmitz and Gouws (2000) developed a DSS consisting of heuristic rules from a domain expert, as well as fuzzified rules generated from decision tree algorithms. The membership functions and weights of the fuzzified rules were optimised using a genetic algorithm. The rules could successfully classify the grade of the floated minerals from features extracted from images of the surface froth in a flotation cell; and the fuzzified rules were used in a DSS.

2.4.5 Comparing rule induction algorithms for process control decision support

A summary of the factors considered for the implementation of each of the rule induction algorithms for decision support is presented in Table 1.

As discussed above, classical rule induction algorithms and association rule mining algorithms are unable to deal with continuous variables. Consequently, such variables need to be discretised before rule induction and are only capable of classification learning tasks. While discretised operational variables are commonly encountered in ECS or fuzzy-logic systems, some decision tree algorithms have the added versatility to extract rules from continuous variables. This enables the use of some decision tree algorithms for regression learning tasks to extract rules with maximum granularity from a data set.

As shown in Table 1, decision tree algorithms require increased hyperparameter tuning when compared to classical rule induction algorithms. Certain elements regarding the tree structure need to be determined before the induction process can commence. While this increases the complexity of the usage of such models, it also provides the flexibility required in the model to extract comprehensible decision rules. Auto-intelligent decision tree algorithms, such as those based on neural networks, or algorithms based on meta-heuristics such as genetic algorithms generally have much more parameters and are more sensitive to the specific parameter tunings. Decision trees have the added benefit of creating disjunct rule sets, with the association between rules determined by the overall tree structure.

Table 1: Comparison of practical considerations of rule induction methods for control decision support

Rule induction paradigm	**Datatypes**	**Learning task**	**Parameter setting**
Classical rule induction algorithms	Nominal/ordinal/ discrete	Classification	Minimal
Association rule mining	Nominal/ordinal/ discrete	Unsupervised variable associations	Minimal
Decision trees	Nominal/ordinal/ discrete/continuous	Classification and regression	Moderate tuning required
Auto-intelligent decision rules/trees	Nominal/ordinal/ discrete/continuous	Classification and regression	Complex tuning may be required

Based on these considerations decision tree models were chosen for rule induction. This allows using quantitative and qualitative data sets to be utilised for the extraction of classification or regression rules. Moderate tuning of the hyperparameters are required, but the trees generated are more robust towards parameter tuning than those generated with meta-heuristic algorithms.

2.5 Decision trees for rule-based decision support of grinding circuit operators

In this subsection the focus is shifted towards grinding circuits and the grinding circuit operator. The section briefly reviews the operation of a representative grinding circuit and the most important considerations for process control. The role of the process operator is

explored to make a pragmatic case for decision support and why decision rules from decision trees would be an effective manner to achieve this.

2.5.1 A brief overview of grinding circuits

Grinding circuits form the final stage of the comminution process, producing a product of the desired particle size for downstream enrichment processes. Depending on the specific ore to be comminuted, grinding is usually preceded by a crushing circuit to reduce run-of-mine (ROM) ore to a PSD suitable for grinding.

Historically, grinding circuits have taken form in many different configurations. These include primary-crushed single-stage SAG/AG/ball mills coupled with hydrocyclones for classification, tertiary-crushed feed to a closed ball milling circuit, South African style ROM SAG/ball mills or secondary-crush-HPGR ball milling (Wills and Napier-Munn, 2006; Metso, 2015). Selection of the optimal grinding circuit involves an economic analysis based on ore characteristics of the orebody from laboratory test work.

AG/SAG circuits have proliferated in recent years owed to the reduced operating costs required to operate these mills, with less grinding media required than ball mills (Napier-Munn, Morrell, Morrison, *et al.*, 1996). In the case of AG circuits, no such grinding media is required as the larger impact forces required are provided by the fraction of larger rocks from the ROM ore. AG/SAG circuits are able to treat a wider range of ores and feed PSD's, as well as ores with significant amounts of clay material (Napier-Munn, Morrell, Morrison, *et al.*, 1996). However, this robustness comes at the cost of increased complexity in circuit operation and control, with AG/SAG mills frequently operating unstably and with large amounts of variability. Figure 6 depicts a modern SAG, ball mill and pebble crusher (SABC) grinding circuit containing a reverse configuration ball mill and cyclone. The operation of a grinding circuit is discussed below with reference to the circuit in Figure 6.

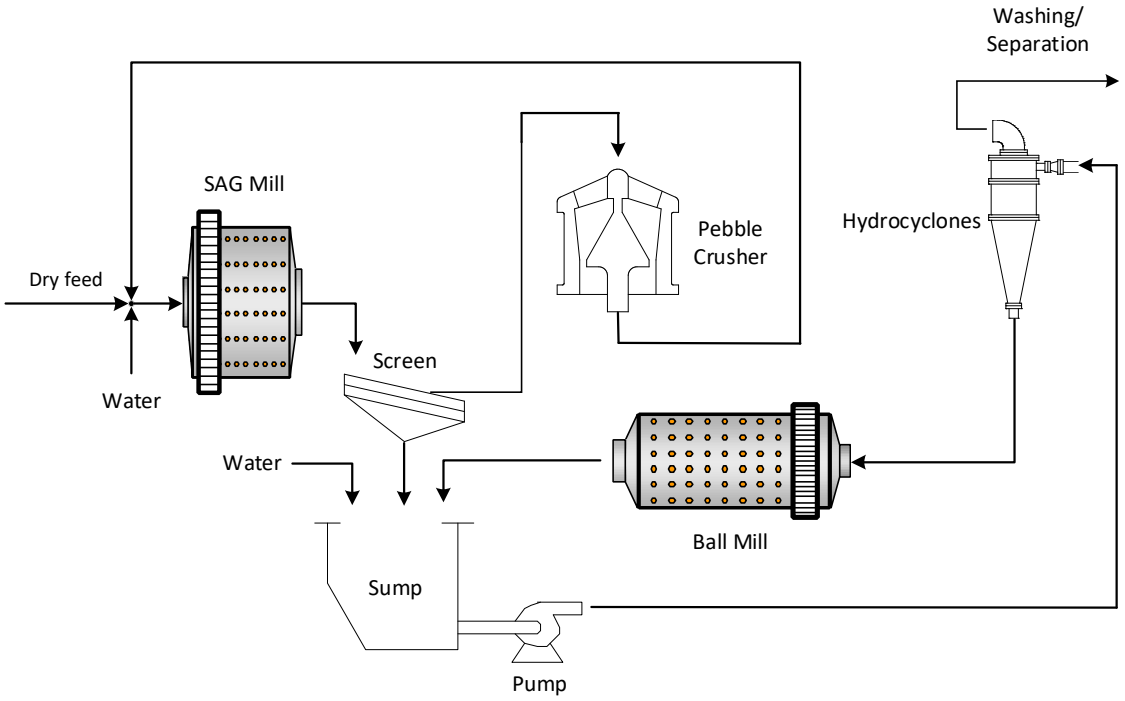

Figure 6: Diagram of a SABC grinding circuit with a reverse configuration ball mill

In the SABC circuit in Figure 6 dry ore and water are fed to the SAG mill. The PSD of the feed ore depends on the preceding crushing stage. For a primary crushed feed, a representative F80, indicating the size of the screen aperture through which 80% of the feed passes, is around 150 mm. The tumbling motion of the drum imparts impact forces to break down larger rocks and attrition forces to produce fine SAG mill product.

The SAG discharge is classified into under - and oversize material with a screen. The screen undersize has the desired PSD for ball milling and flows into a sump. A representative transfer size, T80, in a SABC circuit is around 2mm. The screen oversize mostly consists of so-called critical size material, denoting competent pebbles with a size between 20 – 60 mm. The pebbles are crushed to about 10 mm before joining the fresh feed for regrinding. The pebble crusher is a common feature in modern SAG circuits because of the increases in mill throughput achievable by removing and crushing pebbles before regrinding.

Screen undersize material is combined with more water in the sump before being pumped to a hydrocyclone cluster. The hydrocyclones classify the slurry into a fine overflow stream of liberated particles ready for downstream separation and a coarser underflow requiring further grinding. The PSD of the cyclone overflow is carefully controlled to maximise metal recoveries in downstream separation processes. For a gold-bearing ore requiring cyanidation,

a typical cyclone overflow PSD would contain 75% of particles passing a 75-micron sieve. This cut point in the cyclone is strongly affected by the cyclone feed PSD, pressure and solids loading, as well as the cyclone geometry.

The coarser cyclone underflow is directed through gravity to a ball mill for further grinding. In contrast to SAG mills, ball mills operate at a higher fill level, with a higher grinding media load and smaller grinding media sizes. These factors enable grinding to finer sizes through attrition and abrasion forces. The ball mill discharge is fed back to the sump to be reclassified by the hydrocyclones.

2.5.2 Considerations in grinding circuit control

Grinding circuit control is frequently implemented to achieve one of the following objectives (Wills and Napier-Munn, 2006):

a) Achieving maximum throughput at a constant product PSD
b) Maintaining constant throughput within a small range of product PSD
c) Achieving maximum throughput in conjunction with downstream recovery

In grinding circuits, the mills are responsible for breaking ore particles from a feed PSD to a target discharge PSD. Consequently, effective mill control is critical to the success of the grinding circuit objectives. Breakage is accomplished through impact, compression, attrition and abrasion forces between the ore particles themselves and collisions with grinding media. Thus, the breakage occurring inside the mill is largely dependent on the volumetric loading of the mill charge, commonly referred to as J_C, and the fraction of the charge consisting of grinding media, J_b/J_C.

Mill control generally focuses on maintaining the mill fill level, J_c, at the optimal level to produce the desired discharge PSD, while also making efficient use of the energy provided by the mill drive system. The relationship between the fill level, throughput, power and discharge PSD are commonly represented as grind curves (Powell and Mainza, 2006; Powell, van der Westhuizen and Mainza, 2009), as shown in Figure 7. The nonlinear relationships between the three parameters and the mill filling is particularly noticeable in Figure 7.

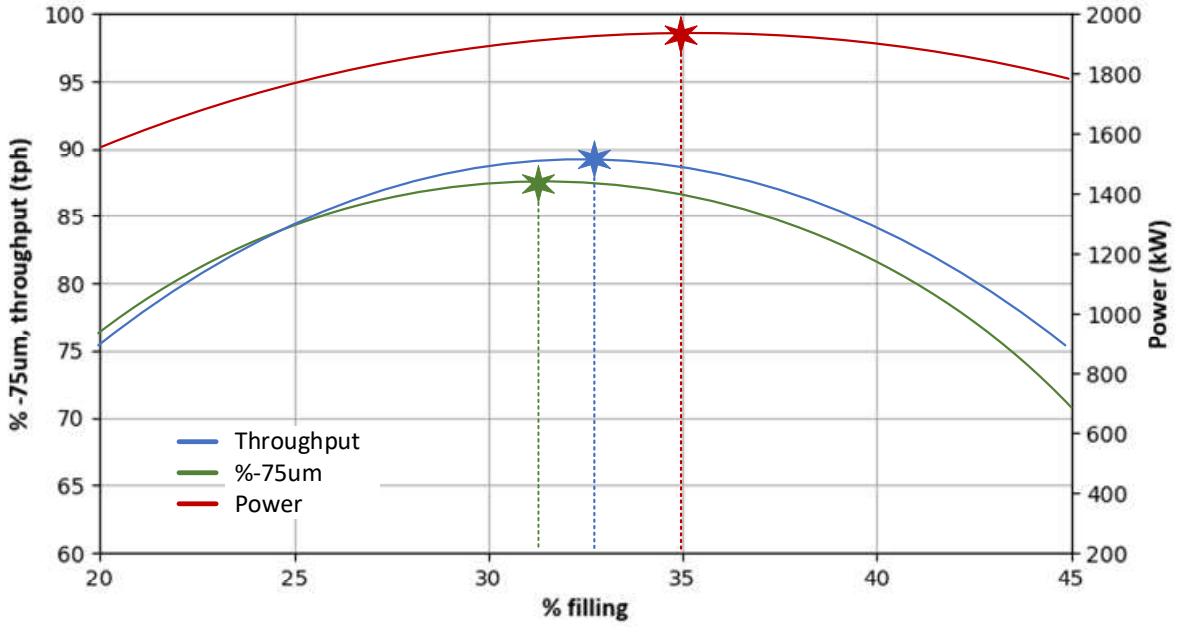

Figure 7: Representative grind curve for a fixed mill speed. Figure reproduced from (Powell, van der Westhuizen and Mainza, 2009).

While there are many variants of mill control philosophies, these all focus on maintaining the mill load in the region of the three peaks in Figure 7. Generally, process operators or APC systems manipulate the mill speed, feed rate or water addition rate to control the mill load. The mill speed is often set to control particle trajectories inside the mill to minimise hits to the shell and noise emanating from the mill. The dry feed rate is usually considered the primary manipulated variable to control the mill filling level. Water addition is frequently operated at a set ratio of the dry feed to maintain a constant solids density inside the mill. However, the water and mill speed can also be used to control the residence time of particles inside the mill.

An example of such a control philosophy is depicted in the control chart in Figure 8. While this chart is specifically focused towards operating a single-stage AG/SAG mill (single mill closed with cyclones), the control philosophies remain relevant to most circuit configurations.

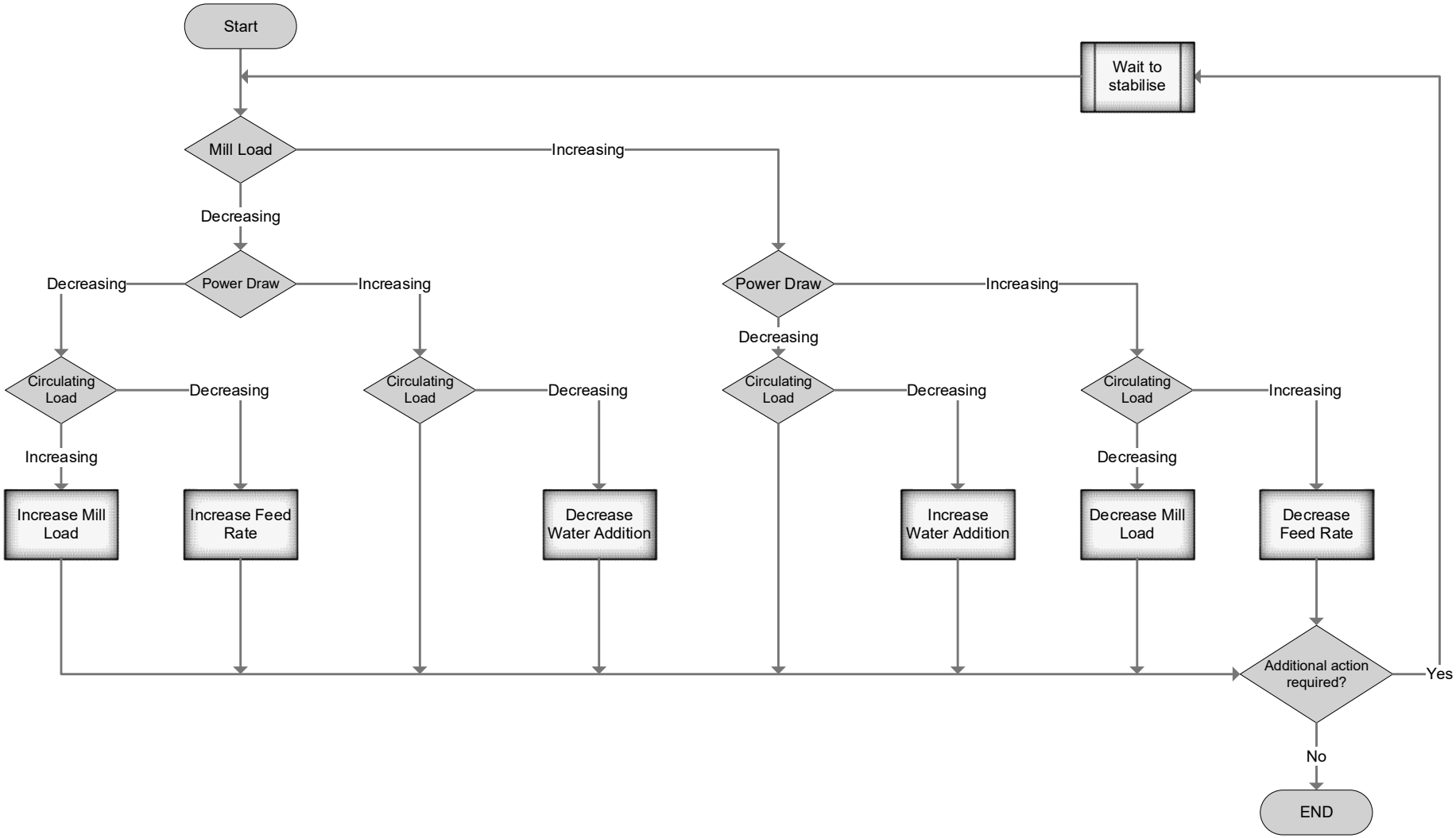

Figure 8: General single-stage SAG control chart adapted from Putland, Kock and Siddall (2011)

2.5.3 The role of the control room operator

In the absence of APC systems, the control room operator is responsible for supervisory control tasks. Control charts, such as the chart depicted in Figure 8 are often provided to operators during training or used as a reference in the control room. The chart leads operators to identify common operational states, and then suggests a control action. While such charts are useful to guide an operator in the right direction in most situations, it still leaves important questions up for subjective interpretations, such as:

- How much can each of these variables increase/decrease before it is time to act?
- How much should the recommended manipulated variable be increased/decreased in this situation?

These unanswered questions lead to major source of variability in the circuit operation. This is demonstrated in with some typical operator behavioural patterns observed by the author.

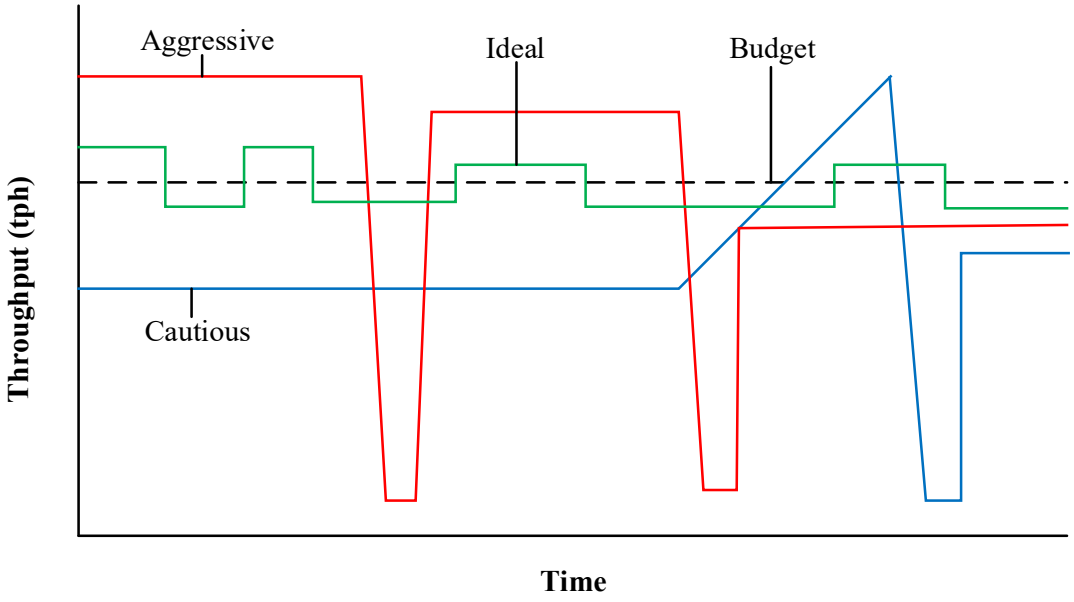

Figure 9: Common observed operator behavioural patterns over the course of a shift

Each operator has a specific appetite for risk while running the circuit. As demonstrated in Figure 9, some operators choose to run the circuit rather cautiously, maintaining relatively lower tonnages to ensure the circuit is not pushed to equipment limits. While this mostly alleviates the risk of equipment failures such as tripping the mill or blocking conveyor chutes, available production capacity remains unutilised. To achieve the budgeted tonnages per shift,

the operator sometimes feels the need to overcompensate, eventually overloading the mill and requiring drastic action, such as a “grind out”.

On the other hand, some operators are more comfortable exploring these upper equipment limits to maximise production and operate the circuit more aggressively. However, operating at these limits increases the likelihood of the operator losing control of the circuit, resulting in situations requiring drastic action to minimise equipment overload and damage. While this operator is taking on more risk, it is not clear whether this results in actual production increases over the longer term. Ideally, operators should be encouraged to make more frequent, but smaller changes to maintain circuit production around the budgeted amount.

The addition of diagnostic models to the operator’s suite of support tools could nudge the more cautious operator to increase production, while recommending the more ambitious operator to pull back. Such diagnostic models could be designed for anomaly or change-point detection, fault diagnosis or include sets of “if-and-then” decision rules as proposed in this work. The diagnostic models could provide the operator with a higher-level interpretation of the data, thereby improving immediate understanding of circuit conditions and possibly reducing the variability in decision-making.

2.5.4 The case for decision support in grinding circuit control

Based on the discussions in this section, human operators will continue to maintain a critical role in the success of grinding circuit operation. Computer-based APC systems have reported significant advances in literature since their introduction 40 years ago. However, as is the case in academic literature in most fields, for the most part only successful cases are reported. Li *et al.* (2011) remarks that successes are often reported in the early days of an APC implementation, when developers are still actively involved on site. However, the performance of these systems are known to deteriorate with time, yet such events remain unreported (Li *et al.*, 2011). This is reinforced in the survey done by Olivier and Craig (2017), which indicated that in 60% of the respondents’ plants, operators disable the controllers and take manual action at least once per shift. While these needs for manual intervention sometimes arises from faulty instruments and actuators, a significant portion of these events are caused by longer-term process drift from the controller models.

In their field study, (Li *et al.*, 2011) observed that the main impediment towards better usage of process control methods was that the information collected was not represented effectively to support the operator for supervisory control tasks. They suggested that an increased focus on human factors on mineral processing plants, aiming to achieve better integration between humans and technologies, could aid in achieving the increased plant performance expected with process control systems. This can be achieved through the development of effective HMIs and alarms, and restructuring operator training to build a better understanding of using these systems (Li *et al.*, 2011).

Accordingly, in this work a DSS is proposed to aid operators in their decision-making. The overall control system exists as a hybrid intelligence system, drawing knowledge extracted from data from the DSS, as well as heuristic and tacit knowledge from the process operator. There are some inherent advantages to this type of hybrid-intelligence system, three of which are discussed below:

a) Human operators can to the best of their ability deduce appropriate responses to unfamiliar situations. Automatic control systems lack this "common sense" and can only be expected to react reasonably in situations addressed during the controller design stage. Once an unfamiliar situation arises, operators may choose to draw on the guidance of a DSS, whereas an automatic system could extrapolate incorrectly to the unfamiliar situation, and act incorrectly before being temporarily disabled.
b) Operators on mineral processing plants are frequently required to take manual measurements of important variables, since the variables are not measured on-line (Olivier and Craig, 2017). Such measurements could include:
    - Taking a sample of a slurry stream and calculating the solids density with a Marcy scale
    - Determining the %-passing (P80) of a hydrocyclone overflow by taking a stream sample and manually separating the solids with a sieve tray
    - "Measuring" the slurry rheology by feeling the slurry stream with the hand (the author has observed numerous operators doing this to control water addition to the circuit)
    - Visual inspections of the hydrocyclone underflow to determine whether it is "roping", usually because of a high solids loading

If these variables are measured manually, they are often not loaded into the plant distributed control system (DCS) and are thus unavailable to automatic systems. The operator is required to act on the information gathered manually.

c) Operators are susceptible to a wider variety of environmental stimuli, often non-programmable, to their decision-making processes than automated systems. Such inputs might include finer knowledge of the ore being processed on the day as communicated by management, or certain restrictions on the circuit due to maintenance or equipment issues, or altered production targets from management etc. An operator, armed with their own knowledge base and DSS, might prove to be more effective and robust than automatic systems in these situations.

The overall control philosophy is depicted in Figure 10. Raw measured signals and unstructured data types are collected in the plant historian and DCS. Here, the data sources are processed and transformed into information before being displayed to the operator on a HMI. The DSS model resides as a submodule in the DCS. The DSS model analyses the real-time information and visually produces the results of a diagnostic model to the operator. Operators draw on the combined knowledge in the HMI, as well as their own heuristic knowledge, to decide on the appropriate control action. This control philosophy could provide improvements to process control in grinding circuits wherein supervisory control functions are mostly left to human operators.

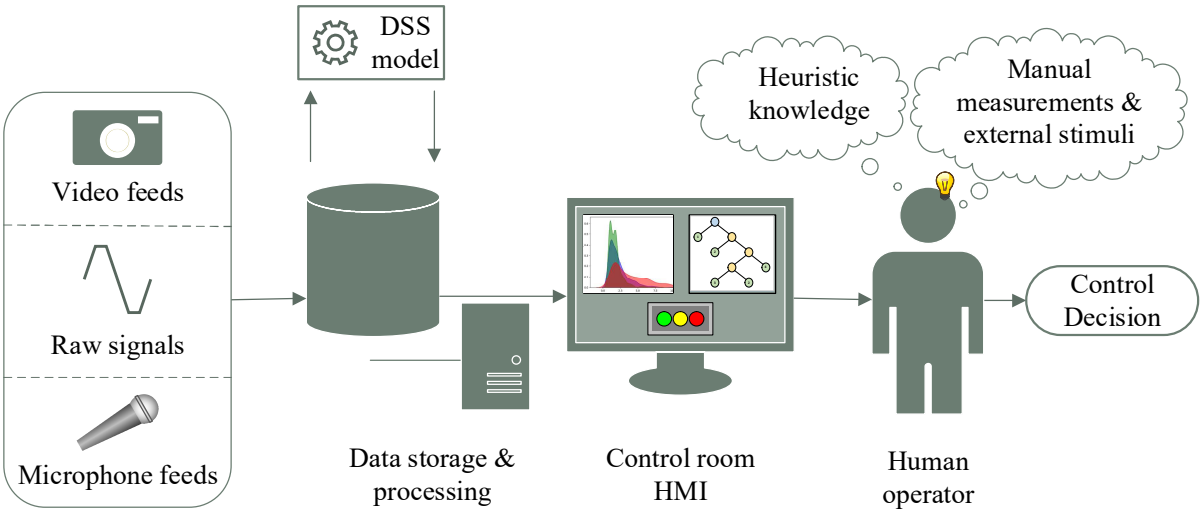

Figure 10: Hybrid intelligence control philosophy based on a DSS built from decision rules extracted from decision trees

2.5.5 Decision trees as a diagnostic model for decision support

While there are various methods to extract knowledge from data, the DSS proposed here is composed of rules extracted from decision trees. Thus, the diagnostic model referred to in Figure 10 would take the form of "if-and-then" decision rules. The rules could be displayed on the HMI as either a decision tree, in verbal rule form, or other common HMI graphics such a traffic light type system. Decision trees are powerful ML algorithms capable of addressing the nonlinear interactions between variables encountered in grinding circuits. Decision tree algorithms such as CART are capable of handling categorical or numerical variables, or a combination of the two, and is easily applied to both classification and regression problems.

Decision tree structures are a natural way to represent the decision-making process of operators running a grinding circuit. This is demonstrated by the control chart in Figure 8, which depicts common control decisions made by operators when controlling a single-stage SAG circuit. Knowledge in the chart is also easily expressed in the form of rules, as demonstrated in equation (5).

$$\begin{aligned} &IF\ (Mill\ load\ is\ increasing) \\ &\quad and\ (Power\ draw\ is\ decreasing) \\ &\quad and\ (Circulating\ load\ is\ decreasing); \\ &\quad THEN\ (Increase\ water\ addition\ rate) \end{aligned} \tag{5}$$

The rule in equation (5) designates an operational state wherein usually the mill is retaining fine material, causing the mill to overload. The control chart and rule suggest that the best course of action is to increase the water addition rate to flush out some of the fines.

Control charts and rules such as Figure 8 and equation (5) are the result of eliciting the heuristic knowledge of an expert, and representing this knowledge explicitly. Decision trees have a similar hierarchical structure as the flowchart in Figure 8, but knowledge in the trees are extracted directly from process data. Rules extracted from decision trees can formalise some of the most common behaviours exhibited by operators and be utilised to decrease the variability caused by inconsistent decision-making. Decreasing this variability could lead to increased production, reduced wear and increase the general stability of the grinding circuit.

2.6 Summary

From the discussion in this section, it is clear that there is a great wealth of data and information being collected in grinding circuits and the broader mineral processing industry. However, the pathways to utilise this information and transforming it into knowledge are not yet well established. There is currently significant research focus on utilising these data sources to improve process control. Much of this research seems to be focused on predictive rather than descriptive goals. Increasingly complex modelling approaches are applied to industrial data sets, but it is not clear how these methodologies are building on our current understanding of the underlying data-generating processes. Many of these studies focus on the application of these complex modelling approaches, and not the data set itself, or the insights discovered from the data using the model.

This would not necessarily be a point of concern if fully automatic control systems were the norm. However, such systems are severely hampered by the lack of instrumentation to measure process disturbances, such as changes in feed PSD and ore competencies. While we remain unable to measure such disturbances, APC models are at best reactive and no feedforward control is possible to compensate for the variation in these inputs. Currently, there is a large industrial and academic focus to overcome these deficiencies in instrumentation.

With the advent of significantly increased computational power and data collection devices, the focus on human decision support seems to have decreased in the last decade compared to their prominence in the 1990's. However, in an industry such as the mineral processing industry where operators continue to have a significant role to play, further development of such systems seem necessary to improve process control until fully-automatic systems are more well-established.

Rule-based support systems are an especially attractive option when the fundamental process dynamics are not well understood. ECS have made significant strides since their introduction in the 1970's, but these systems are difficult to maintain over longer periods. Limited work has been done on the development of DSS using rule induction algorithms, while even fewer have focused on inducing decision rules that are interpretable for humans. Such an approach is investigated in this work to support grinding circuit operators.

3. Methodology

This section presents a methodology to extract decision rules using decision tree algorithms. The methodology focuses on the extraction of rules from grinding circuit data that are interpretable by humans. Decision tree algorithms have had widespread use and are well established in literature. Their use to extract decision rules intended for use by humans are encountered less frequently.

The basic methodology is outlined in Figure 11. Data collected from grinding circuits on mineral processing operations is used to construct a decision tree model. Successive instances of the tree are pruned until a tree structure is produced that is interpretable by humans, while remaining sufficiently accurate. A rule set for decision support is extracted from the pruned trees.

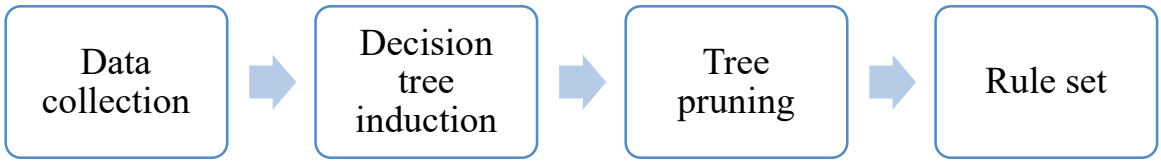

Figure 11: Basic methodology to interpretable extract decision rules from pruned decision trees for operator support

This section discusses the basics of decision trees and the algorithms to build trees. Pruning methods to increase the interpretability of tree structures are explored. A general methodology for the rule extraction procedure is presented, and a set of metrics to evaluate the utility of rules is discussed.

3.1 Decision tree models

Decision tree models are machine learning algorithms that split a dataspace into hyper-rectangular subspaces. Each subspace is associated with a single class label for categorical data, or numerical value for continuous data. These subspaces are identified by recursively searching for partitions based on input variables that cause the largest reduction in impurity of the output variable in proposed child partitions. Decision tree models are trained through supervised learning, during which the model learns to map a set of input data, $\mathbf{X} \in \mathbb{R}^{N \times M}$ (N samples with M variables each), to a response vector $\mathbf{y} \in \mathbb{R}^{N \times 1}$. Decision tree algorithms are

also considered non-parametric, in that the model does not require any assumptions regarding the distributions of input and target variables.

An illustrative example of the rectangular subspaces generated by decision tree models is given in Figure 12. A classification tree was fit to Fisher's renowned iris data set(Fischer, 1936) to predict the specific species of iris flower from multivariate input features such as the flower sepal length width and length. While the data set contains four input features, only two of these features were used for the classification task to reduce the dimensionality for visualisation. In two dimensions, the subspaces take on the shape of normal rectangles, as shown in Figure 12.

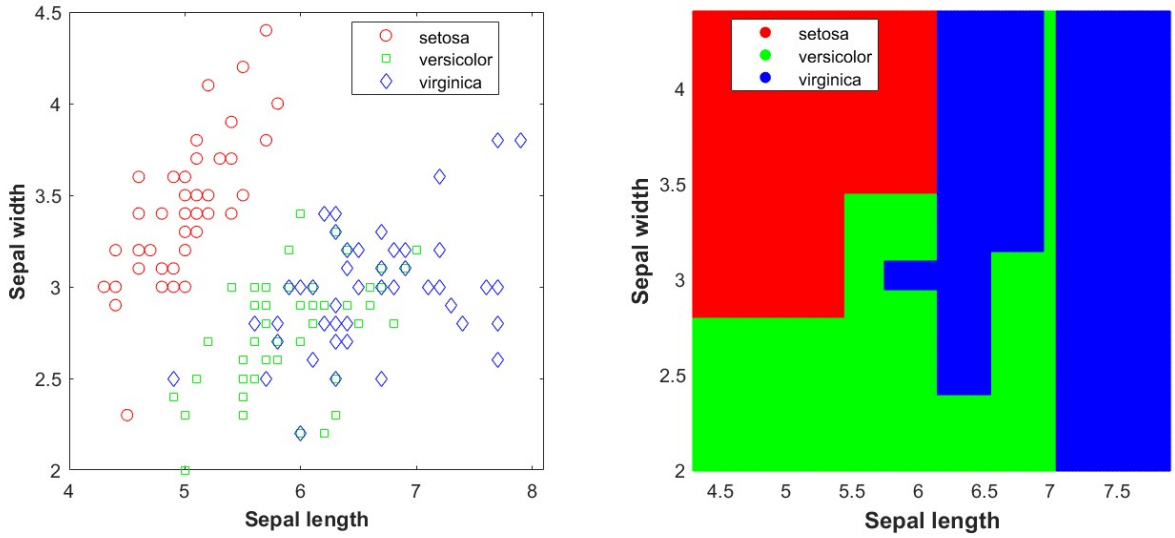

Figure 12: (Left) Scatterplot of the 2-D feature space of Fisher's iris data set with the different iris species (classes) superimposed with different colours and markers. (Right) 2-D Decision surface created by a classification tree tasked to classify the samples into each of the three iris classes

Decision tree models can be visualised as the decision tree structures in Figure 5 and Figure 13 below. These structures consist of nodes and branches (or edges) connecting the nodes. The top node is called the root node and contains all the examples in the data set. All nodes except for the root node have exactly one incoming edge. Nodes with outgoing edges are called internal nodes or decision nodes. Each internal node splits the dataspace into two or more subspaces according to some discrete function of the input values. Terminal nodes, or leaf nodes, have no outgoing edges and contain the numerical prediction or class membership for data that conforms to the conditions set out by nodes on the pathway from the root to the specific leaf node. In a classification tree, this prediction is the label of the class present

in the highest proportion in the node. For regression trees, each terminal node uses the average of the training sample responses belonging to the node as a prediction. The decision tree generated for the iris classification problem depicted in Figure 12 is shown in Figure 13. Decision tree models can be interpreted with relative ease using the decision tree structures.

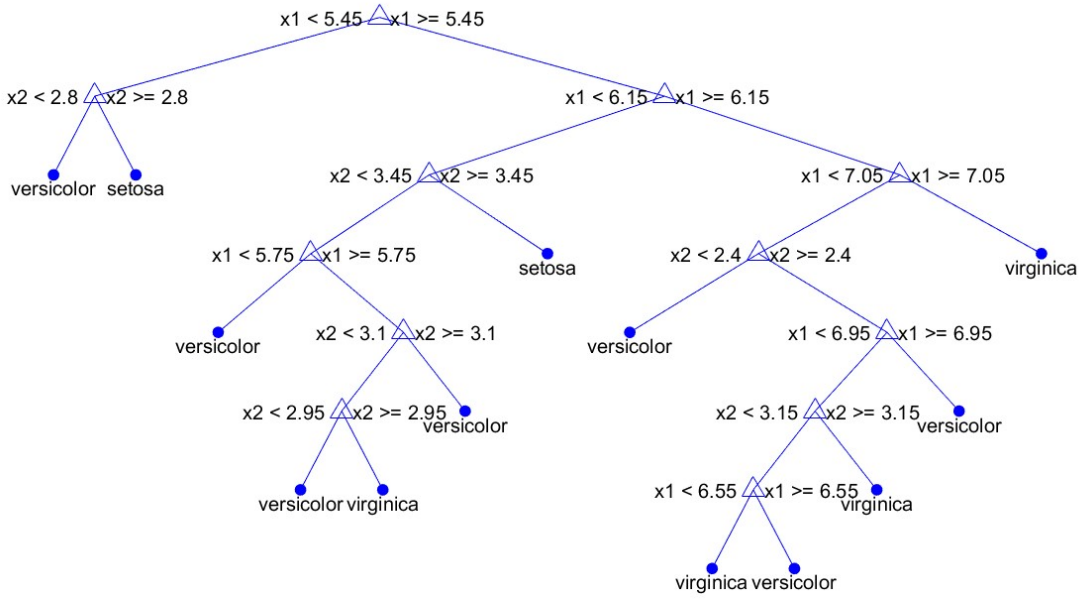

Figure 13: Decision tree model to classify samples from Fisher's iris data set into three classes designating different iris species

Decision trees are easily converted into a disjunctive normal form (DNF) rule set (Weiss and Indurkhya, 1993). In a decision tree, each path from the root node to a terminal node can be represented as a rule consisting of the conjunction of tests on the internal nodes on the path. The outcome of the rule is the class label or numerical value in the leaf node. Such a rule is extracted for each terminal node in the decision tree. A DNF rule set is generated by the conjunction of all the rules with an OR operator. A decision tree rule set is a special case of a DNF rule set where all rules are mutually exclusive, where only a single rule can be applied in each case. Both representations as decision trees or decision rules have made decision tree models attractive for DSS or expert systems since this type of logic is easy to program into a computer.

3.2 Decision tree induction

3.2.1 Decision tree induction algorithms

Decision tree algorithms automatically construct decision tree models from a data set. Usually the algorithm's goal is to find the optimal decision tree that minimises the model generalisation error on a test set. However, induction of the optimal decision tree from a given data set is a difficult task. Naumov (1991) showed that generating an optimal decision tree from decision tables is NP-hard, while Hancock *et al.* (1995) proved that finding a minimal tree consistent with the training set is also NP-hard. While it has yet to be proven (Cook, 2006), it is suspected that there are no polynomial-time algorithms for NP-hard problems. This indicates that the construction of the optimal decision tree would only be computationally feasible on smaller data sets. Consequently, decision tree algorithms employ heuristic methods for tree induction instead of an exhaustive search through all possible tree structures.

Decision tree algorithms such as ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993) and CART (Breiman *et al.*, 1984) employ a top-down, recursive induction process. During each iteration, the algorithms split the data subspace into further partitions based on some discrete function of the input features. This function manifests as a hyperplane perpendicular to one of the coordinate axes of the dataspace defined by the input features. The specific hyperplane is selected according to algorithm-specific splitting measures. Once the optimal split is identified, the recursive partitioning process continues through the newly created subspaces. This procedure is repeated until none of the new proposed splits lead to a gain in the splitting metric, or additional imposed stopping criteria are satisfied. A typical algorithmic framework, adapted from Maimon and Rokach (2005, chap. 9), for a classification tree induced with a top-down procedure is given in Figure 14.

The classification and regression trees (CART) algorithm developed by Breiman *et al.* (1984) is one of the most widely used decision tree algorithms. CART is characterised by the fact that the algorithm generates binary trees, since each node has exactly two outgoing edges. CART can generate classification and regression models for both categorical and continuous variables. CART can also implement custom misclassification costs and allows users to incorporate prior probability distributions into the model. To reduce overfitting, the decision

tree can be pruned using the cost-complexity pruning procedure to balance the model complexity and generalisation error. The Gini impurity is commonly used with CART for split selection in classification tasks. For regression tasks, CART selects splits that minimise the squared prediction error on the target vector.

Algorithm: $TopDownClassificationTreeInduction(\mathbf{S}, \mathbf{X}, \mathbf{y})$

Where:
$\mathbf{S}$ – Training set
$\mathbf{X} \in \mathbb{R}^{\mathbf{N}\times\mathbf{M}}$ – Input features
$\mathbf{y} \in \mathbb{R}^{\mathbf{N}\times\mathbf{1}}$ – Response

begin
1. Let T be a tree with a single root node
2. **if** a stopping criterion is fulfilled **then**
3. Mark the node in T as a leaf node with the most common value in $\mathbf{y}$ as its prediction
4. **else**
5. Find a discrete function $f(\mathbf{X})$ of the input features such that splitting $\mathbf{S}$ according to $f(\mathbf{X})$ results in a set of outcomes $(\mathrm{v}_1, \dots \mathrm{v}_\mathrm{n})$ that leads to the best splitting metric
6. **if** best splitting metric > threshold **then**
7. Label t with $f(\mathbf{X})$
8. **for** each of the outcomes v_i of $f(\mathbf{X})$ **do**
9. Let $Subtree_i = TopDownClassificationTreeInduction($ $\mathbf{S}(f(\mathbf{X}) = \mathrm{v}_\mathrm{i}), \mathbf{X}, \mathbf{y})$
10. Connect the root node of t_T to $Subtree_i$ with an edge labelled v_i
11. **end for**
12. **else**
13. Mark the node in T as a leaf node with the most common value in $\mathbf{y}$ as its prediction
14. **end if**
15. **end if**
16. **return T**

end

Figure 14: Algorithmic framework for top-down classification tree induction

The ID3 and C4.5 family of decision tree algorithms developed by Quinlan (1986, 1993), have also found widespread use. ID3, or iterative dichotimizer, was the first of the implementations by Quinlan and considered a very simple decision tree algorithm. ID3 cannot handle numerical data or missing values and does not apply any pruning. C4.5 was developed to address these issues and is much more like CART. Contrary to CART, ID3 and C4.5 allow multiple splits at each node. ID3 uses the information gain as a splitting criterion, while C4.5 employs a gain ratio.

Other popular decision tree inducers include the CHAID (Chi-square-Automatic-Interaction-Detection) (G.V.Kass, 1980), QUEST (Quick, Unbiased, Efficient, Statistical Tree) (Loh and Shih, 1997) and MARS (Multivariate adaptive regression splines) (Friedman, 1991) algorithms. However, these algorithms have not received as widespread usage as Quinlan's family of algorithms and CART.

A thorough comparison of the accuracy of classification algorithms, including most of the algorithms mentioned above, was conducted by Tjen-Sien, Wei-Yin and Shih, (1992). The authors found that C4.5, CART and QUEST performed the best among decision tree algorithms in terms of both error rate and speed. The authors also noted that the mean error rates between many different algorithms are sufficiently similar that these are unlikely to produce statistically significant differences in practical implementations. C4.5 was also observed to produce trees with twice as many leaves as trees generated with CART and QUEST (Tjen-Sien, Wei-Yin and Shih, 1992).

Based on the discussion above, CART was selected as decision tree inducer. Besides achieving classification results matching that of the top decision tree algorithms, CART is the sole algorithm capable of generating regression trees. This flexibility allows for the analysis here to be extended to the generation of regression rules for decision support. Generating rules at the top level of granularity of variables could provide some additional advantages in decision support. For this reason, further discussions focus on CART and the specific implementation of the algorithm found in MATLAB®, which was used during the analysis.

3.2.2 Split selection in CART

In line 5 of the general algorithmic framework in Figure 14, the decision tree algorithm searches for the optimal feature upon which to split, according to a splitting metric. In CART, this discrete splitting function is univariate, meaning the internal node splits the dataspace based on a specific value of a single feature. The splitting criteria for CART in classification and regression tasks are discussed below.

For classification purposes, CART calculates the Gini Index at each split point. The Gini Index is a measure quantifying the probability that a random chosen sample is misclassified if the class label is assigned randomly from the set of possible classes. Consider a classification task

consisting of C classes, where the proportion of class k, at node η, is given by $p(k|\eta)$. The Gini Index, $i(\eta)$, at node η is given by equation (6).

$$i(\eta) = \sum_{k=1}^{C} p(k|\eta)(1 - p(k|\eta)) = 1 - \sum_{k=1}^{C} p(k|\eta)^2 \quad (6)$$

The Gini Index increases as the impurity or mixture of classes increases at the node. This concept is demonstrated in Figure 15 for a binary classification problem. The Gini Index reaches a maximum when both classes are of equal proportion in the node, $p(y = 1|\eta) = 0.5$, while a completely homogeneous node has a Gini Index of zero.

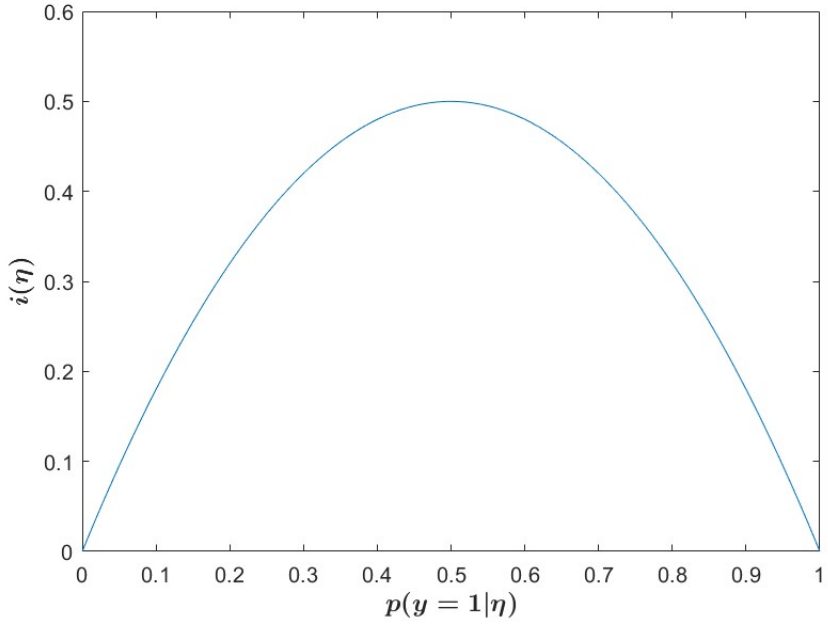

Figure 15: Gini impurity range for a binary classification problem

The reduction in impurity for a proposed split position, ξ, depends on the impurity of the current node, the impurity of proposed left and right child nodes (η_L and η_R), as well as the proportion of samples reporting to each child node (p_L and p_R), according to equation (7). The split position resulting in the largest decrease in the Gini Index is selected to create the next partitions.

$$\Delta i(\xi, \eta) = i(\eta) - p_R \times i(\eta_R) - p_L \times i(\eta_L) \quad (7)$$

In regression trees, splits are selected to minimise the mean squared error from predictions of the child nodes, $\hat{y}_{\eta_L}$ and $\hat{y}_{\eta_R}$, respectively as shown in equation (8). Therefore, in line 5 of

the algorithm in Figure 14, CART searches for the variable and split value resulting in the largest reduction in the sum of squared errors in the proposed child nodes.

$$\Delta i(\xi, \eta) = \sum_{X_i \in \eta_L} \left(y_i - \hat{y}_{\eta_L}\right)^2 + \sum_{X_i \in \eta_R} \left(y_i - \hat{y}_{\eta_R}\right)^2 \quad (8)$$

3.2.3 Overfitting and stoppage criteria

The partitioning procedure in decision tree algorithms continues until all examples in a node belong to the same class, have the same response value or the node contains only a single training example. This makes decision tree algorithms very powerful, since they can fit almost any data distribution. However, this also leads to one of the disadvantages of decision trees, as the models are prone to overfit noisy data sets. Overfitting is characterised by an increase in model accuracy on the training data set, but decreasing accuracy on an independent, unseen test data set. This phenomenon is demonstrated in Figure 16.

Figure 16 shows a sinusoidal data set with randomly generated noise added at every five data samples. Two different decision tree models were trained using CART. The first tree could make a maximum of five splits in total, while the second tree could make 20 splits. While the first tree is attempting to fit the sinusoidal function, the second tree with increased capacity is overfitting to some of the artificial noise. Given an unseen X-value, the tree with 20 splits might predict an erroneous y-value corresponding to noise, rather than an approximation of the smooth function. Even though the first tree model is of reduced complexity, the model generalises better to the sinusoidal function shape. Figure 16 also demonstrates how the discontinuous nature of decision trees is disadvantageous when attempting to approximate smooth functions.

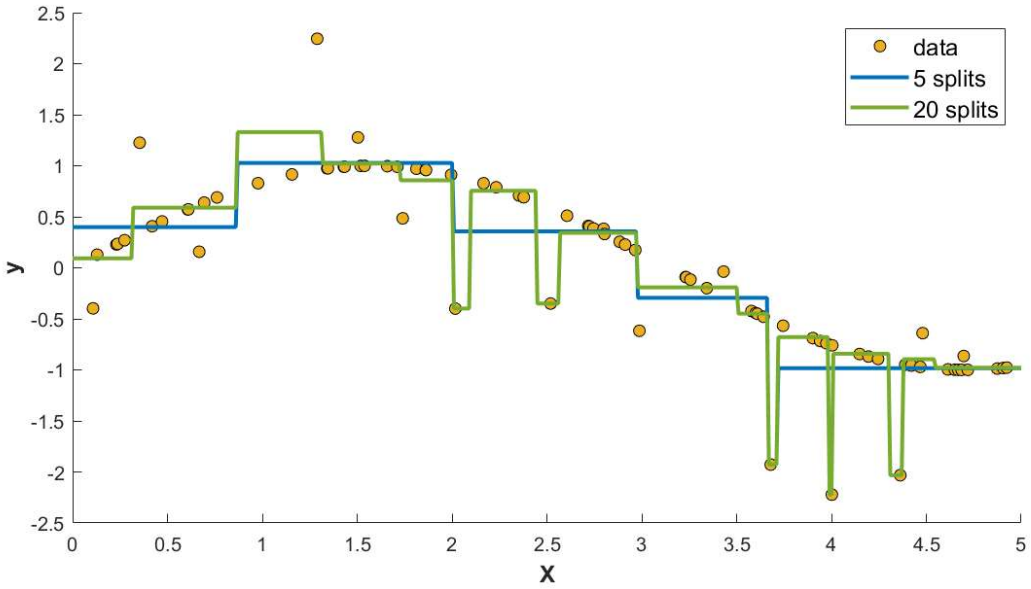

Figure 16: Sinusoidal data set with outlier values at every five data points. Predictions of two decision tree models with different model capacities, enforced through different stoppage criteria, on the data set is also shown.

When training decision tree models it is important to monitor the model performance on the training set, as well as an independent test set to quantify the model generalisation ability. This allows practitioners to observe whether adding additional splits is improving the model, or just overfitting noisy training data. Stoppage criterions are used to prevent decision tree models from overfitting noisy data sets. Some common stopping criteria include:

- Specifying a maximum tree depth or maximum number of splitting nodes
- Enforcing a minimum number of samples required to belong to a node for the node to be considered for another split
- Enforcing a minimum number of samples required to form a new leaf node
- Specifying a maximum number of leaf nodes
- Enforcing the best splitting criteria to be larger than some threshold value to create another split (as shown in the algorithm in Figure 14)

The stoppage criteria mentioned above are sometimes referred to as “pre-pruning” of the decision tree. Pre-pruning prevents the complete induction, where no more splits are possible, of the decision tree. Post-pruning refers to the set of pruning methods wherein the decision tree is first allowed to overfit the training data through a complete induction. The full tree structures are then cut back, usually in a bottom-up fashion, by removing subtrees

that do not contribute to the generalisation accuracy. Common pruning algorithms include cost-complexity pruning, introduced by Breiman *et al.* (1984), or reduced error pruning (Quinlan, 1987).

While some subtler differences between pre – and post pruned trees may exist, both methods aim to increase the generalisation ability of the tree through simplification of the tree structures. Another benefit of the pruning procedures is that the simplified trees are naturally more comprehensible. Rule sets extracted from pruned trees will contain a smaller number of rules, will contain a smaller number of antecedents in each rule and rules would be applicable to a larger amount of cases. These factors are important considerations for constructing successful trees for DSS.

3.2 Decision rule extraction procedure

This section presents the methodology used to extract useful rules from decision trees to support operator decision-making. The overall procedure is displayed in Figure 17. A short discussion of each step is given below and the considerations for practical application of each step to a real data set is presented. The knowledge discovery process is naturally iterative, since different data sets, different objectives and difference configurations of the decision tree models will lead to different rule sets. After each step, the practitioner evaluates the outcome and decides whether to proceed, or revisit previous steps. Examples of such decisions are also presented in Figure 17. In practice, a practitioner would repeat the process numerous times until a rule set is found which provides them with sufficient utility or finds a solution to a specific problem.

3.2.1 Data collection and exploration

In the first step of the methodology operational data is collected from the grinding circuit. Ideally, this data set would span a period of operation capturing some variation or drift in the process. This might include periods wherein metallurgists identified that the circuit was running sub optimally to identify the operational patterns that led to this operation. Alternatively, one could include periods wherein the circuit was running satisfactorily, where the experts want to formalise the operational patterns leading to this type of operation. This naturally requires intimate details of the circuit goals and specific operational paradigm. To

successfully evaluate the utility of identified rules, the practitioner needs to understand why operators choose to make certain decisions.

Next, an exploratory analysis of the collected data should be conducted. Dimensionality reduction methods, such as PCA, can help to reveal the presence of frequently recurring operating states in the multivariate data set. Tying decision rules to specific operational states could provide guidance to move from less, to more favourable states. The practitioner needs to decide whether the data sets contains enough variation, i.e. moving between different states, to proceed. If this is not the case, the data collection step is repeated with a larger data set or a data set spanning a different period.

3.2.2 Modelling problem formulation and assessment of predictability

The modelling problem needs to be carefully formulated to ensure rules are extracted to address a specific variable that can solve an existing problem or address specific operational patterns. The data set needs to be transformed to suit the supervised learning required by decision trees. This requires the identification of candidate input, $\mathbf{X}$, and target variables, $\mathbf{y}$, for the decision tree algorithm. Some examples for a grinding circuit might include:

a) Modelling the circuit dry feed rate, $\mathbf{y}$, as a function of other operational variables (feeder selection, feed PSD, mill power, water addition, pebble discharge, discharge densities etc.), $\mathbf{X}$, to identify patterns leading to lower or higher throughputs
b) Modelling the mill specific energy (kWh/t), $\mathbf{y}$, as a function of mill inputs (feed blend, feed PSD, water addition, grinding media addition etc.), $\mathbf{X}$, to discover patterns leading to increased milling efficiencies
c) Modelling the cyclone overflow density (% solids), $\mathbf{y}$, as a function of operational variables (discharge sump water addition, cyclone pressure, number of cyclones used, ball mill power draw. cyclone underflow density etc.), $\mathbf{X}$, to identify patterns resulting in satisfactory overflow densities for downstream processes

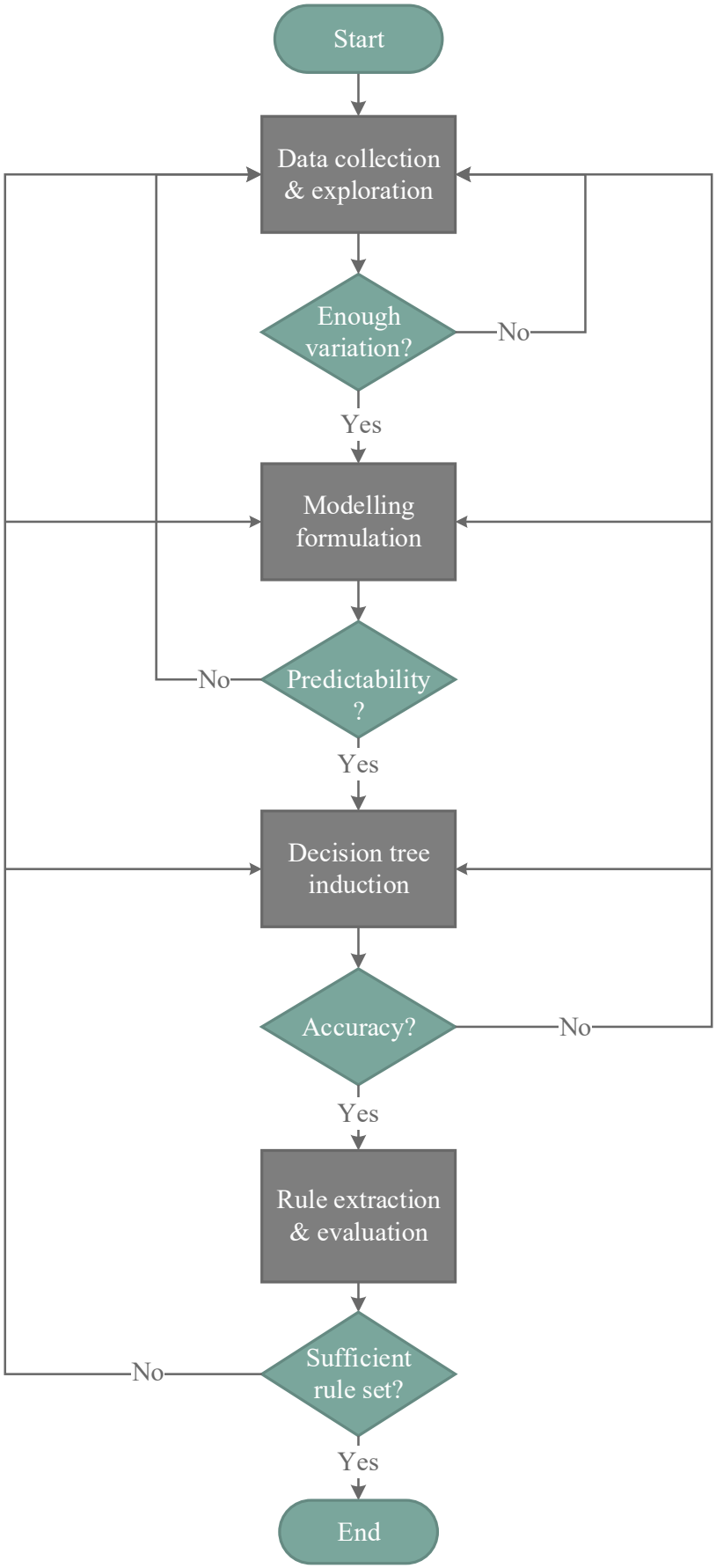

Figure 17: Extended methodology for extracting intelligible rules from decision trees

Both controlled and manipulated variables are suitable for knowledge discovery. Decision tree models constructed with manipulated variables as the target leads to rules with a direct control action as its prediction, i.e. increase or decrease the dry feed rate as in the first example above. Modelling a controlled variable does not have this feature but serves to discover common operational patterns leading to different operational states, i.e. identifying which patterns of operational variables increase the specific energy.

In addition to the input and output variables mentioned, the role of the operator is embedded as a latent variable into the data set. The operator's contribution will usually be revealed as a set point change in the manipulated variables. Thus, in many situations, rules are describing common decision-making by operators. With the rules induced, good and bad decision-making can be identified and addressed. Good decision-making patterns can be formalised and emphasised for future decisions.

Once the modelling problem has been formulated, the general predictability of the chosen target variable as a function of input variables should be assessed. While decision trees are interpretable, the model capacity is generally lower than that of ANNs or random forest (RF) models. In this study, RF models were used to determine the upper limit of predictive performance based on the model specification. Consequently, an overview of RF models are given later in the section.

To generate such models, 80% of samples in the original data set was used for training and the remaining 20% was used as a test set to quantify the generalisation performance of each of the models. The data was randomly divided into training and test sets, thereby introducing the assumption that the data samples are independent and identically distributed (I.I.D.) random variables. While this assumption may prove reasonable over long-term periods of steadier operation, it is less realistic during transient periods of operation.

Model accuracy on the training and test set is commonly quantified using the classification accuracy and R^2-score for classification and regression, respectively. The classification accuracy simply denotes the fraction of correctly classified samples, as indicated in equation (9). The coefficient of determination, R^2, denotes the fraction of the variation in the target variable accounted for by the fitted model as shown in equation (10). Alternatively, the R^2-score could also be described as the squared of the Pearson's correlation coefficient between

the observed target variable values, y_i, and the fitted values, $\hat{y}_i$. Accordingly, it is desired to achieve values close to one using both metrics. Large disparities between the training and test set accuracies indicate that the model is overfitting the training set and could indicate that a lower capacity model may be preferable.

$$classification\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions} \tag{9}$$

$$R^2 = 1 - \frac{explained\ sum\ of\ squares}{total\ sum\ of\ squares} = \frac{\sum_i^N \left(\hat{y}_i - \overline{\hat{y}_i}\right)^2}{\sum_i^N (y_i - \overline{y})^2} \tag{10}$$

If the predictive performance of the high capacity model is too low, the model specification can be revisited, or a larger data set gathered. The predictive performance of the higher capacity model also serves as a useful comparison for the performance of the decision trees generated in the next step of the procedure. Satisfactory generalisation performance is necessarily strongly linked to the specific model formulation and would require a subjective assessment in each case. As a general rule-of-thumb, a descriptive model with a classification accuracy or R^2-score of 0.8 may be considered satisfactory. However, models with less generalisation capability could still prove to be useful, such as in cases when attempting to identify rare fault conditions. In such cases the patterns identified will require expert scrutiny to ensure the validity of insights extracted.

3.2.3 Decision Tree Induction

Next, classification or regression trees are induced with the CART algorithm. The model is trained on a training partition of the data set and the generalisation accuracy calculated from predictions on a test set in a similar manner to that outlined in the previous section.

As a first attempt, the decision tree is induced with minimal restrictions on branch growth and complete induction is allowed. In this configuration the model is very likely to overfit the training set. The number of nodes in the unpruned tree is identified, and the generalisation accuracy is calculated on the test set. The tree performance is compared with that of the higher capacity model trained in the previous step.

Next, the tree induction is repeated with a progressively smaller number of nodes in the tree and the train – and test set accuracy is recorded. This restriction is enforced through either a

hard limit on the number of nodes allowed, or a limit on the maximum tree depth. As will be demonstrated, at some point the addition of more nodes does not further increase the model generalisation performance.

The final tree selected for rule extraction will contain enough nodes to not be dramatically worse than unpruned trees yet has a small enough number of splits to remain interpretable for decision support. This trade-off will be demonstrated in the case studies. The most influential variables to the decision tree model are identified from the variables used to split the data in the tree structure. These variables are analysed to determine whether they coincide with expert knowledge and circuit heuristics. If the tree accuracy is too low, previous steps are revisited and the induction process is repeated.

3.2.4 Rule extraction and evaluation

If the predictive performance of the selected tree is deemed satisfactory, the tree is converted into a DNF rule set. A DNF rule is extracted for each leaf node in the decision tree. Each rule is evaluated with a set of metrics, as specified in the following section, to determine the utility of the rule. Rules with high utility or importance are critically analysed since these rules have the potential to be utilised for operator decision support. These rules need to be evaluated by an expert familiar with the circuit, to determine the validity of the rule and how it can practically be used for decision support.

With a rule set consisting of high utility rules, the practitioner needs to decide whether the rule set is sufficient to address the problem identified during the model specification step. If this is not the case, previous steps are revisited to further refine the rule set.

3.3 Evaluating the utility of decision rules

As described above, extracted decision rules are evaluated according to the utility that they can provide in decision support. This implies that rules are significantly represented in the data set, referred to here as the support of the rule, while remaining sufficiently accurate and of low complexity to remain comprehensible. The utility of an individual rule can be considered a function of these three factors:

$$utility(rule) = f(support, accuracy, complexity) \quad (11)$$

Conceptually, specific configurations of these factors exist that can maximise the utility of a rule. This concept is explored qualitatively in the case studies. Each of the three metrics are briefly discussed below, and the interactions between the metrics are explored.

3.3.1 Supporting samples in the data set

For a rule to be of any utility, it needs to be significantly represented in the data set. A rule that is rarely applicable is not of much use for decision support. Thus, the higher the number of samples in the data set belonging to a specific rule, the higher the support is for the rule. This definition of support is analogous to the support metric of association rules (Agrawal, Imielinski and Swami, 1993). In contrast to ECS, the focus here is to generate a small number of rules with high support.

Common operating patterns, whether good or bad, will show up with high support in a rule set. Generally, the analysis proposed will focus on the subset of rules with the largest amount of support. However, sometimes the practitioner may want to diagnose the conditions surrounding rare events, such as fault conditions or operator mistakes detrimental to the operation. In such cases, the rules of interest could have less support.

3.3.2 Rule accuracy

A rule of high utility needs to provide an accurate prediction for the data samples belonging to the rule. The accuracy of the rule refers to the dispersion of target values of the data belonging to the rule. For classification trees, the accuracy of a rule is represented by the proportion of samples belonging to a rule that is correctly classified by the predicted class of the rule. Highly accurate classification rules would thus have a high proportion of samples correctly classified by the rule's predicted class. For regression trees, the accuracy is well represented by the standard deviation of the target values of samples belonging to the rule. Low standard deviation of the target values of samples indicates a relatively accurate prediction by the regression tree, with sample target values close to the predicted mean.

Both accuracy measures are closely related to the impurity measures used during construction of the trees. During this analysis, emphasis is generally placed upon rules with high accuracy.

3.3.3 Rule complexity

For a rule to be of utility for decision support, the rule must remain interpretable by humans. While this notion of interpretability is naturally subjective, longer rules with a large number of splits are difficult to interpret and tie to physical phenomena. The shorter the rule, the easier it is to interpret and possibly act upon. Here, an emphasis is placed on the generation of shorter rules. Rules with up to four or five conditions in the rule antecedent could be considered interpretable. The decision tree algorithm can be forced to generate shorter rules, and a fewer number of rules, by specifying the maximum number of splits allowed in the tree.

However, these restrictions will come at the cost of decreasing the accuracy of the rules, since the capacity of the model has been decreased. Thus, there is a trade-off between the rule accuracy and its complexity. On the other hand, the decreased capacity of the model does have the beneficial effect that fewer rules, each with larger support, will be generated. Thus, while the limit to rule complexity decreases the accuracy, it might increase the support. The relationship between the support and accuracy is more dependent on the data set itself than the model specification.

3.4 An overview of Random Forest models

As depicted in Figure 17, the predictability of the target variable is assessed using a higher capacity model to obtain an upper limit of the predictive performance for the specific model on the collected data set. In this study, this was achieved by using RFs. While there are many high capacity models to choose from, such as ANNs and SVMs, RFs were chosen because of their close relation to decision trees. RFs consist of an ensemble of decision trees and have much higher capacity than any single tree. A brief discussion on RF models are presented below.

3.4.1 Random Forests

RFs (Breiman, 2001) grow many classification or regression trees using the standard CART algorithm. To classify or predict the regression output of an input sample $\mathbf{X_i}$, the sample is sent through each tree in the forest. Each tree predicts a response to the sample, called the "vote" of the tree. For classification, the RF chooses the class having the most votes from all the trees in the forest. For regression, the average of responses from all trees is the output of the forest. This procedure is depicted graphically in Figure 18.

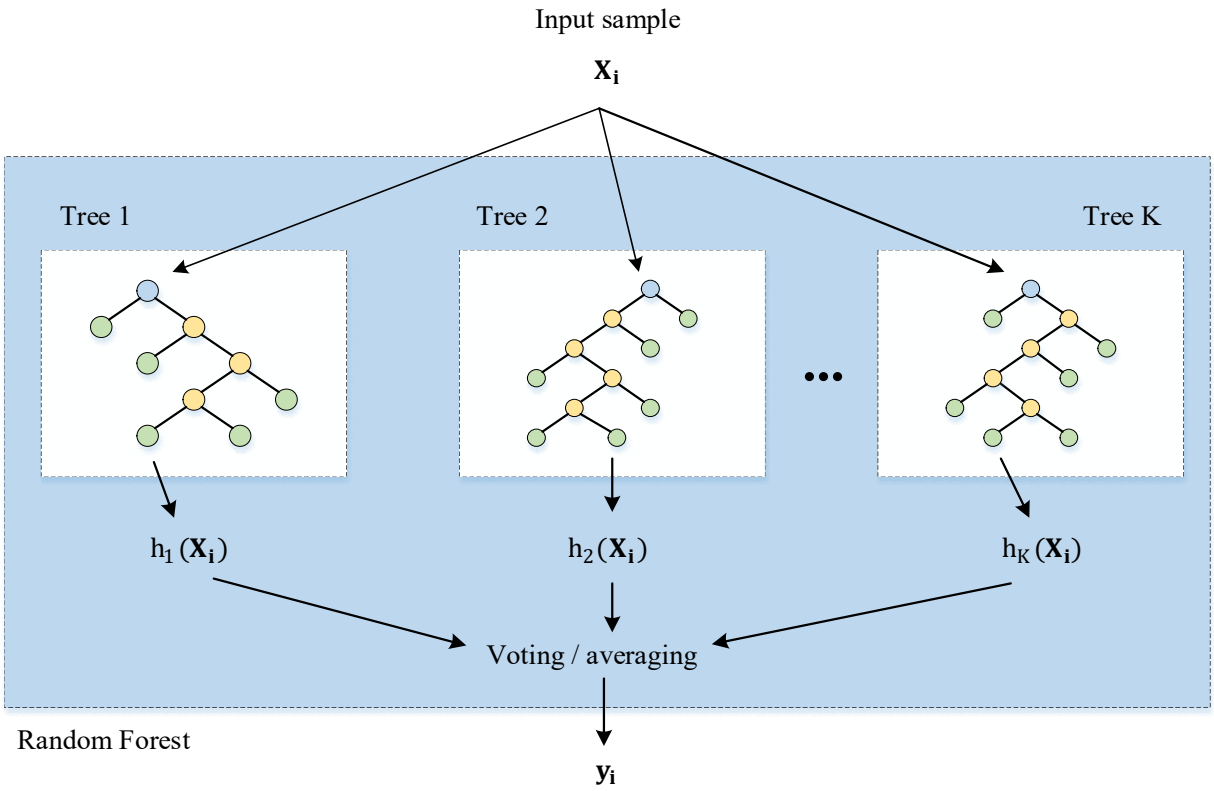

Figure 18: Schematic of decision trees predicting a response for an input sample in a random forest model (Reproduced from Aldrich and Auret (2013))

Each tree in the forest is grown with the following three steps:

1. Randomly sample n cases, $\mathbf{X_k} \in \mathbb{R}^{\mathbf{n \times M}}$, from $\mathbf{X} \in \mathbb{R}^{\mathbf{N \times M}}$, with $n < N$ with replacement (bootstrapping). $\mathbf{X_k}$ is the training set for growing the tree.
2. Select a number m ($m < M$), such that at each node, m variables are selected at random from the M variables available, and the best split from $\mathbf{X_k^{n \times m}}$ is selected to split the node. The value of m is constant throughout the forest growing procedure.
3. Grow each tree to completion without stoppage criteria applied. No pruning is applied afterwards.

Thus, each tree is grown on a different subset of the training data, with only a small subset of all the variables available for each split. While this decreases the strength of individual trees, the randomisation procedures also decrease the correlation of the trees in the forest, ultimately increasing the generalisation ability as the number of trees increase (Breiman, 2001). Breiman proved how these randomisation procedures inhibits random forests from overfitting data as more trees are added, but rather produce a limiting value of the generalisation performance.

The remaining samples not included in $\mathbf{X_k} \in \mathbb{R}^{\mathbf{n}\times\mathbf{M}}$ for the construction of each tree is used to quantify the test set error of each tree. In this manner, a test-set prediction is obtained for each sample in around a third of all the trees (assuming about two thirds of the data set is included in each bootstrap sample). Aggregating these predictions for each sample and testing against the ground-truth for the sample produces the out-of-bag (OOB) error estimate of the forest, which has proved an unbiased estimator of the forest generalisation performance.

Random forests usually require two model parameters: the number of trees K and the number of split variables at each node m. Since the random forest does not overfit, K can be selected to be sufficiently large so that the OOB error does not increase with the addition of more trees. Lower values of m increases the diversity between trees and increases the generalisation performance. However, a too small m values decreases the strength of individual trees and adversely affects generalisation. Liaw and Wiener (2002) suggested using m equal to the floor of the square root of M for classification and $M/3$ for regression.

4.4.2 Variable importance measures in random forests

While decision tree models are easily interpreted through the tree or rule structures, the forest of trees in RFs obscure the input-output mapping. The randomisation procedure in RFs also removes the ability of a single tree in the forest to sufficiently represent a data set. However, variable importance analyses can be used to discover the relative influence of each variable in the ensemble model.

One approach is to randomly permute the values of the predictor variables, one at a time, and determine the decrease in model accuracy because of the permutation of each of the variables. This procedure is called permutation variable importance analysis. Permutation of a predictor variable removes the variable's association to the response variable. If the model accuracy significantly decreases, the association between the predictor and response variable is significant. In RFs, the permutation variable importance can be calculated from the OOB samples after the growth of each new tree. The permutation variable importance $VI^{(k)}_{perm}(\mathbf{x_j})$ for variable $\mathbf{x_j}$ of M variables, for tree number k is calculated according to equation (12).

$$VI_{perm}^{(k)}(\mathbf{x_j}) = \frac{1}{n_{OOB}}\sum_{i=1}^{n_{OOB}}\left(y_i^{(k)} - \hat{y}_i^{(k)}\right)^2 - \frac{1}{n_{OOB}}\sum_{i=1}^{n_{OOB}}\left(y_i^{(k)} - \hat{y}_{j,i}^{(k)}\right)^2 \quad (12)$$

In equation (12):

- $y_i^{(k)}$ is the i'th response variable observation in the OOB samples of tree k
- $\hat{y}_i^{(k)}$ is the prediction of tree k for the i'th input vector in the tree's OOB samples
- $\hat{y}_{j,i}^{(k)}$ is the prediction of tree k for the i'th input vector in the tree's OOB samples, with variable j permuted in the input vector

The variable importance measure of each variable $VI_{perm}^{(k)}(\mathbf{x_j})$ is determined from the average of these measures over all individual trees in the forest:

$$VI_{perm}(\mathbf{x_j}) = \frac{1}{K}\sum_{k=1}^{K} VI_{perm}^{(k)}(\mathbf{x_j}) \quad (13)$$

The permutation importance measure also considers multivariate interactions between the permuted variable and other input variables. The permutation does not only destroy any association to the response variable, but also any association to other input variables (Aldrich and Auret, 2013, chap. 5). However, this implies that this measure is sensitive to the correlation structure of the data set.

The Gini importance measure is an alternative measure that estimates the influence of each predictor based on the changes in node impurities at each split in each tree of the RF where the predictor is involved. The Gini importance is based on the difference between a node's impurity and the weighted sum of impurities of child nodes descendent from the node. Summing these decreases in node impurities over all nodes in the forest gives the relative influence of a predictor variable (Aldrich, 2020).

Formally, the importance of variable $\mathbf{x_j}$ is determined by summing the decreases in impurity Δi at all nodes in the forest where the data space is split using $\mathbf{x_j}$**.** Within a single tree, this is calculated according to equation (14) (Aldrich, 2020).

$$VI_{gini}^{(k)}(\mathbf{x_j}) = \sum_{t \in T_k : v(s_t) = \mathbf{x_j}} p(t)\Delta i(s_t, t) \quad (14)$$

In equation (14), the summation is over all nodes t in the full set of nodes in the k'th tree in the forest, T_k. Additionally, the summation only considers nodes t where $\mathbf{x_j}$ is the variable, $v(s_t)$, used at split s_t. The summation is weighted by the fraction of samples $p(t) = \frac{n_t}{n}$ reaching node t. The average across all trees is known as the Gini importance of the variable:

$$VI_{perm}(\mathbf{x_j}) = \frac{1}{K}\sum_{k=1}^{K} VI_{gini}^{(k)}(\mathbf{x_j}) \tag{15}$$

In contrast to the permutation importance, the Gini importance is calculated according to impurity reduction on the training data set (not the OOB samples) and could show misleading importance measures when the tree is overfitting the training data. In the case of regression models, the $\Delta i(s_t, t)$ in equation is replaced by the mean squared error difference of parent and child nodes at the split.

3.4 Summary

In this section, the basic considerations of decision tree algorithms were discussed. While these methods are well established, their use for discovering rules interpretable by humans is less mature.

A methodology was presented to extract such a set of intelligible rules. The methodology is simple to apply but requires intimate knowledge of the specific operation. Metrics to evaluate the utility of each rule was also introduced. However, a circuit expert is still required to evaluate the knowledge contained in each of the rules for their practical application as part of a DSS. This is demonstrated in the case studies in the following sections.

4. Case study: Classification rules predicting operator decisions in a SAG circuit

In this section, the general methodology is demonstrated on a data set from an industrial SAG circuit. A classification tree is used to examine periods wherein the operator lost control of the circuit and had to take drastic action by bypassing the SAG pebble circuit. The case study focuses on identifying the operational patterns surrounding these events. This enabled predicting similar events based on historical data and allows metallurgists to address the conditions causing these events to reduce the frequency of such events in the future. The section starts with a brief description of the SAG circuit and its operation, after which the general methodology depicted in Figure 17 is followed to extract operational heuristics from the circuit.

4.1 SAG circuit description

A diagram of the SAG circuit is provided in Figure 19. Descriptions of each of the variables are summarised in Table 2.

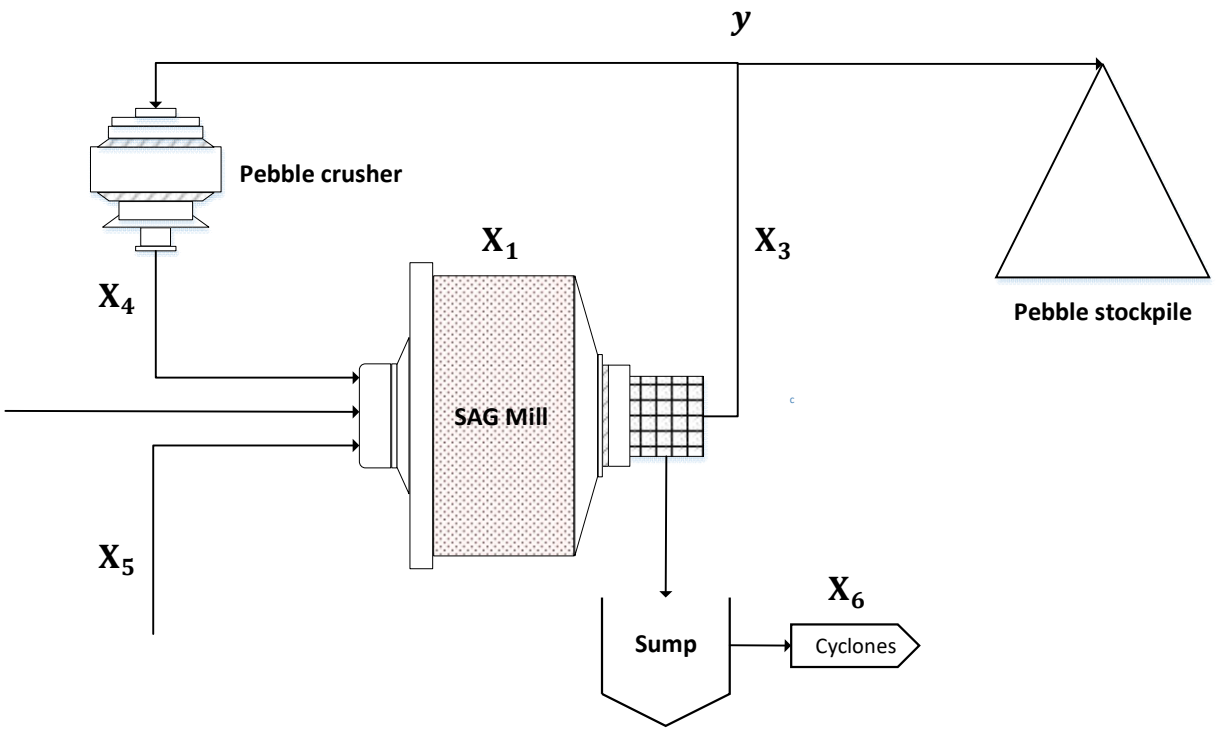

Figure 19: Diagram of the semi-autogeneous grinding circuit

The SAG circuit forms part of the common SABC grinding circuit configuration. Secondary crushed ore, with a fraction of primary crushed ore bypassing the secondary crusher, is fed to the mill from a crushed ore stockpile. Water addition to the mill is maintained at a ratio of the dry feed with a cascade controller. The mill is of a grate-discharge type with pebble ports to discharge critical size material. Fine SAG mill product flows through a trommel screen (undersize) into the SAG mill discharge sump, before being pumped to the cyclone feed hopper in a reverse configuration ball milling circuit. Pebbles, consisting of competent mid-size material building up in the mill, exit the mill along with slurry through the pebble ports. These pebbles are classified to the trommel screen oversize and fed to a cone crusher via a conveyor belt. The crushed pebbles are fed back on to the dry feed conveyor before re-entering the mill for further grinding.

Table 2: Description of SAG circuit variables in Figure 19

Variable identifier	Description	Unit
$\mathbf{X_1}$	Mill power draw	kW
$\mathbf{X_2}$	Dry feed rate	Tonnes/hour
$\mathbf{X_3}$	Pebble discharge rate	Tonnes/hour
$\mathbf{X_4}$	Pebble returns rate	Tonnes/hour
$\mathbf{X_5}$	Water addition rate	m^3/hour
$\mathbf{X_6}$	Cyclone pressure	kPa
$\mathbf{y}$	Pebble circuit bypass	Binary control variable

The analysis here is limited to the SAG portion of the circuit since the grinding circuit was generally SAG limited. The ball milling circuit consisted of two parallel ball mills with additional capacity available at the designed transfer grind size. A distinction is made between the pebble discharge rate and pebble returns, since operators have the option to divert the pebble discharge stream to a stockpile, effectively removing the pebbles from the circuit for near instantaneous control of the mill load.

4.2 SAG circuit model specification

To provide the necessary context to the raw data, the problem formulation and subsequent model specification is introduced before discussing the raw data in the next section. Consequently, step 2 of the methodology outlined in Figure 17 is presented before step 1.

In this case study, the focus was on gaining an understanding of when operators decide to bypass the pebble circuit. As mentioned above, operators have the option to divert the mill trommel oversize to a stockpile, instead of crushing the pebbles and returning them to the mill.

The SAG circuit is operated to achieve maximum throughput, by maximising dry feed rate, while operating within the power draw limits imposed by the SAG mill drive system. Operators continuously monitor the mill power draw and respond by changing the dry feed rate accordingly. Thus, the power draw is the main controlled variable, with the dry feed rate being the primary manipulated variable. There is no measurement of the mill load available. The water addition is set at a ratio of the dry feed rate, and operators can adjust this ratio.

A sequence of events was identified wherein the power draw very nearly approaches the limits of the mill drive system, implying the mill is overloaded. Operating in this region for too long risks tripping the mill by overloading the motor. In these events, operators would either completely stop the dry feed or bypass the pebble circuit, or in some cases take both actions. However, often this led to the pebble circuit being bypassed, thus immediately removing the mid-size fraction of material from the mill, and decreasing the load and power draw.

While this action effectively reduces the mill from tripping, it has various adverse consequences. Since the pebbles still contain a significant amount of valuable product, this material must be re-fed to the mill in the future. Pebbles are usually of higher competency since they had already once survived the milling process and refeeding this higher competency rock in the future essentially postpones the problem. The drastic reduction in mill load caused by dropping the mid-size fraction also results in periods of significantly higher grind sizes transferred to the subsequent ball milling circuit and temporarily overloading the ball mills. Removal of the mid-size fraction also resulted in large amounts of noise emanating from the mill, implying increased wear to steel liners and grinding media.

Metallurgists wanted to decrease the frequency of these events and understand the operating patterns leading to the decision to bypass the pebble circuit. Thus, decision tree models were constructed to predict the status of the pebble circuit. The status of the pebble circuit was represented as a binary "on/off" manipulated variable, as seen in Table 2. Since the target variable is in this case a categorical variable, classification trees were used to extract rules from the circuit. The input and output variables to the classification model are summarised in Table 3.

Table 3: Classification model specification to predict SAG circuit pebble bypass

Inputs	Output
$\boldsymbol{X_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_5, \mathbf{X}_6}$	$\mathbf{y}$

Notably missing from the inputs in Table 3 is $\mathbf{X_4}$, the pebble returns rate. This is because the pebble circuit bypass events correspond to normal tonnages on $\mathbf{X_3}$, the pebble discharge rate, but no pebble returns to the feed conveyor (zero on $\mathbf{X_4}$). While pebbles are continually being discharged from the mill, these are not recycled back to the mill feed. Bypass events are thus characterised by the relation between $\mathbf{X_3}$ and $\mathbf{X_4}$ and consequently the status of the pebble circuit can be perfectly predicted from knowledge of $\mathbf{X_3}$ and $\mathbf{X_4}$ alone. A model containing both $\mathbf{X_3}$ and $\mathbf{X_4}$ will contain a set of rules predicting the status of the pebble circuit using $\mathbf{X_3}$ and $\mathbf{X_4}$ only. While these rules will perfectly fit the data, it will not contain any logic useful to diagnose why these events occur or how they can be prevented. Thus, the relation between $\mathbf{X_3}$ and $\mathbf{X_4}$ had to be broken and $\mathbf{X_4}$ was removed from the input variables.

Modelling the status of the pebble circuit essentially attempts to model the operators' decision-making when deciding to bypass the pebble circuit. Decision rules induced during the analysis should identify the most common operational patterns leading to bypass events. Once these patterns are identified, the behaviours can be addressed to decrease the frequency of these occurrences and increase the longer-term throughput through the SAG. As mentioned above, the SAG circuit was generally the bottleneck in the grinding circuit and thus it was assumed that the ball mills would be able to process the additional tonnes coming from the SAG.

4.3 Raw SAG circuit data description and exploration

4.3.1 Raw data collection

Operational data spanning a period of six weeks was collected from the plant at a sampling frequency of five minutes. To obscure plant operating parameters, each variable was scaled into a range of zero to one using min-max normalisation. The normalised data samples are shown in Figure 20. For the **y**-variable, a pebble circuit bypass event is designated with the value "1", while normal operation of the circuit is denoted with the value "0". Of the 9500 data samples collected, 435 samples correspond to periods where the pebble circuit is bypassed. These periods corresponded to 141 unique bypass events. The raw data was cleaned to remove any plant downtime or periods wherein equipment maintenance occurred.

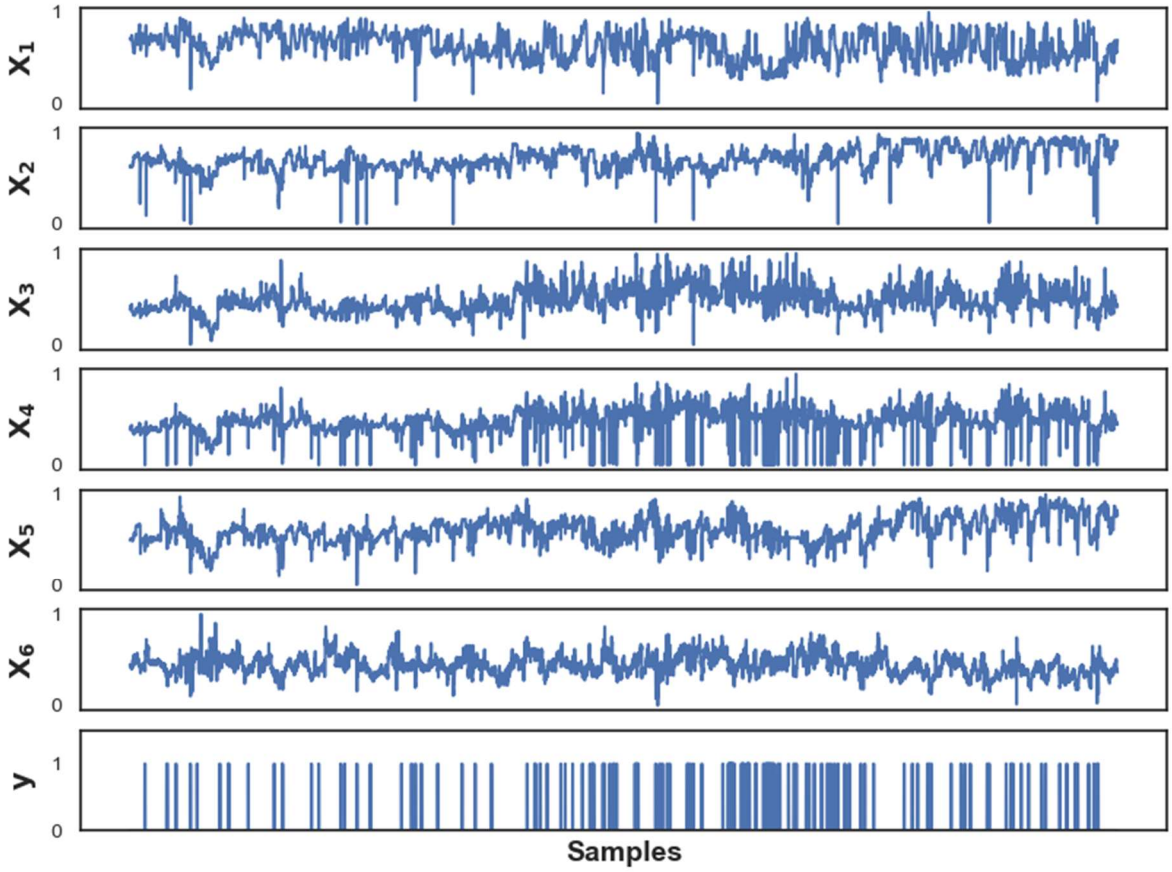

Figure 20: Normalised SAG circuit operational data spanning six weeks of operation

The six weeks of operation contains a large amount of variation in terms of feed ore blends, different operators, and different decisions made. However, these sources of variation could

be considered as normal operational variation and the data set is representative of normal operation at the plant. Rules extracted from the data set would be applicable to future periods of operation.

4.3.2 PCA of the SAG circuit data set

Principal component analysis (PCA) was used to reduce the dimensionality of the data set to obtain a lower dimensional representation of the variance experienced in the circuit over the six weeks of operation. The dimensionality reduction of multivariate data sets enables the identification of operational states as clusters in the lower-dimensional representation. A brief discussion on PCA is presented below, before the method is applied to the SAG data set.

PCA aims to find a set of principal components consisting of linear combinations of the original variables such that the first principal component accounts for the most variance in the data set of all such components. The second principal component then accounts for most of the remaining variance etc. During PCA, the data set$\mathbf{X} \in \mathbb{R}^{\mathbf{N}\times\mathbf{M}}$, is decomposed into the sum of the outer product of score – and eigenvectors (principal components), $\mathbf{t_i}$ and $\mathbf{v_i}$, respectively. The decomposition is presented in equation (16):

$$\mathbf{X} = \mathbf{t_1}\mathbf{v_1^T} + \mathbf{t_2}\mathbf{v_2^T} + \cdots + \mathbf{t_M}\mathbf{v_M^T} \tag{16}$$

The eigenvectors are calculated from the eigen decomposition of the covariance matrix of $\mathbf{X}$, according to equation (17):

$$cov(\mathbf{X})\mathbf{v} = \boldsymbol{\lambda}\mathbf{v} \tag{17}$$

Each pair of $\mathbf{t_i}$ and $\mathbf{v_i}$ are accompanied with an eigenvalue, λ_i, representing the amount of variance captured by the score-eigenvector pair. The score vectors are obtained by projecting the original data set onto each of the eigenvectors and represent the new positions of the data in the principal component space:

$$\mathbf{t_i} = \mathbf{X}\mathbf{v_i} \tag{18}$$

The dimensionality of the data set can be reduced by truncating equation (16) to retain only the $\mathbf{t_i}$ and $\mathbf{v_i}$ pairs representing significant variance in the data set, as denoted by λ_i. Often, the first two or three principal components capture a significant portion of the variance in the original data set.

PCA was applied to the set of operational variables $\mathbf{X_1}$-$\mathbf{X_6}$. The data set was projected onto the first three principal components, as shown in Figure 21. As indicated on each of the axes in the figure, the first three principal components collectively account for more than 83% of the variance in the data set. Thus, most of the variance in the data set is expressed in the three-dimensional space and the lower-dimensional projection can be used as a visual tool to inspect the data set. The pebble returns rate was superimposed on the principal component scores as a colour map. Since the data had been cleaned of periods of downtime, pebble return rates of zero correspond to the occurrence of a bypass event.

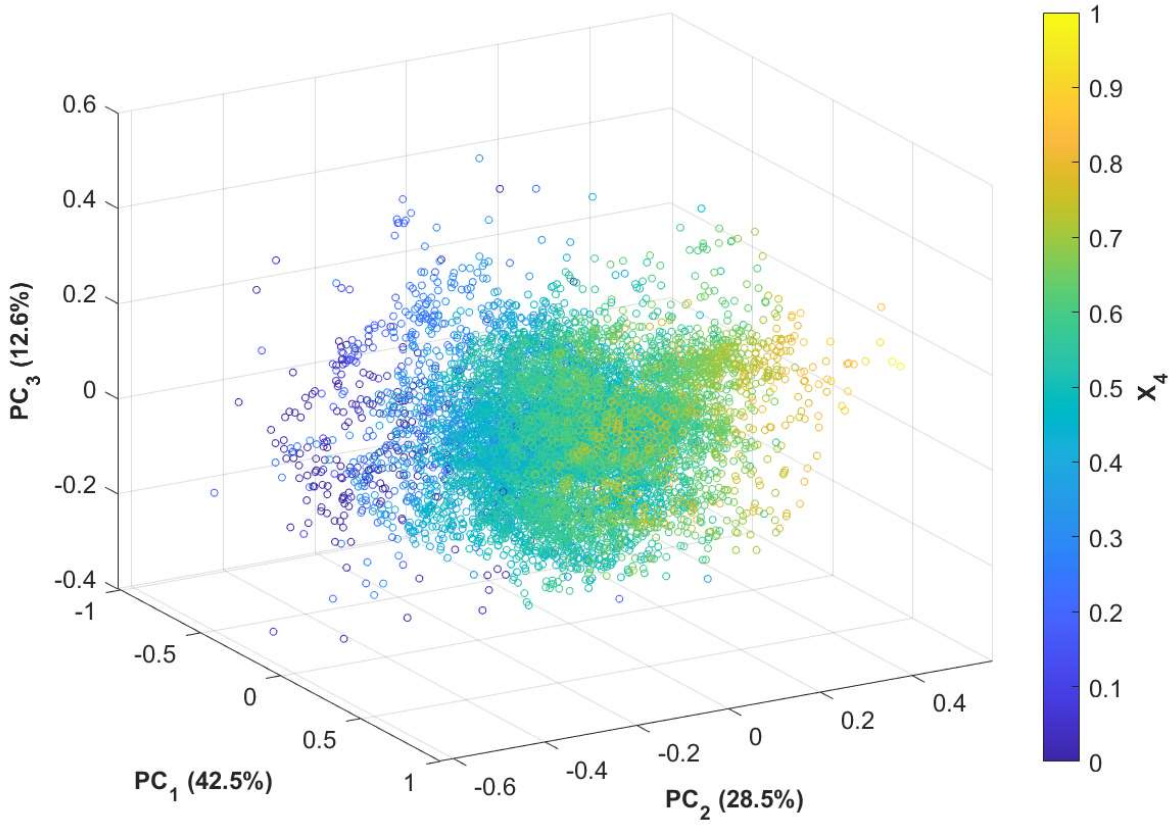

Figure 21: Principal component scores of SAG circuit operational variables on the first three principal axes. The percentage of variance captured by each principal component is indicated in brackets. The pebble returns rate is superimposed with a colour map, with a returns rate of zero corresponding to a bypass event.

Figure 21 demonstrates the effect of the pebble circuit bypass on the overall variance in the data set. From the figure, the pebble returns, and thus the pebble discharge rate, is a significant source of variance in the data set. These variables showed significant expression across the first principal component. Periods corresponding to bypass events lie outside the central cluster corresponding to normal operating conditions. The figure indicates that

reducing the frequency of such events would decrease the overall variability in the circuit, which could ultimately lead to financial benefits over the longer term.

4.3.3 Linear correlations between the SAG circuit operational variables

In another step of data exploration, the correlations between different variables can be inspected to gather further insight on the data set. Pearson's correlation coefficient, or bivariate correlation, quantifies the linear correlation between two variables using equation (19). A positive correlation indicates that the samples of two variables under consideration are often on the same side of their respective means. A negative correlation implies the samples are often on the opposite sides of their mean values.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{19}$$

Pearson's correlation was calculated for each pair of variables in the data set. The correlations are presented in the correlation matrix in Figure 22. Since the response is a categorical variable, it was not included in the analysis. However, understanding the linear associations between the circuit operational variables would aid in the interpretation of model results.

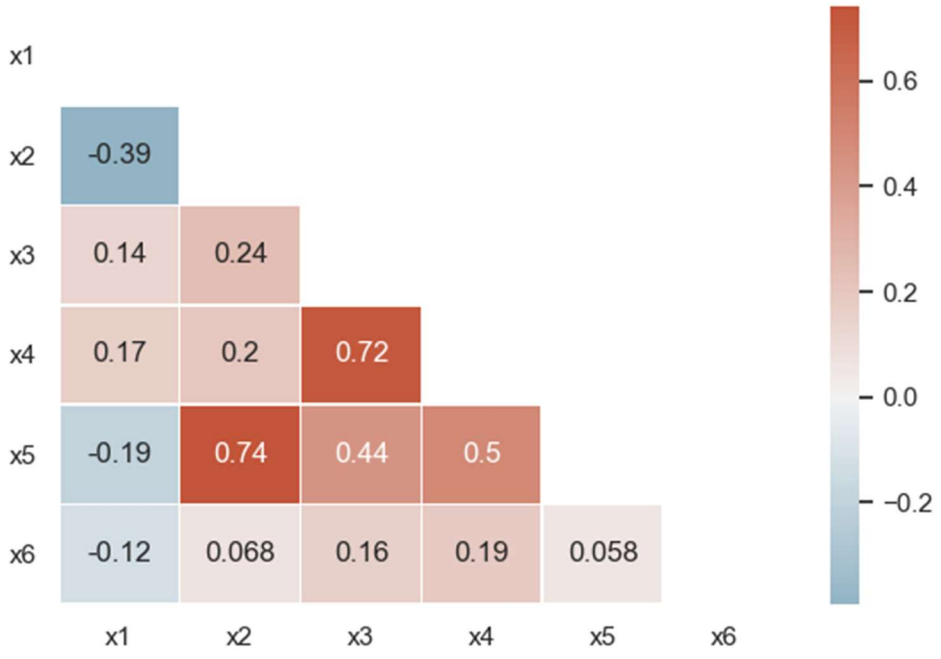

Figure 22: Pearson's correlation between all pairs of operational variables in the SAG circuit

Notable correlations from Figure 22 include:

- A relatively strong, positive correlation between the dry feed rate $\mathbf{X_2}$ and water addition rate $\mathbf{X_5}$. This is a result of the fact that the water addition is maintained at a fixed ratio of the dry feed rate to maintain a relatively constant SAG discharge slurry density
- A relatively strong, positive correlation between the pebble discharge rate $\mathbf{X_3}$ and the pebble returns rate $\mathbf{X_4}$. This is expected, since discharged pebbles are normally continuously crushed and returned to the mill. The variables are separated by a short time delay.
- A moderate, positive correlation between the pebble returns rate $\mathbf{X_4}$ and the water addition rate $\mathbf{X_5}$. This is likely the water addition increasing as more crushed pebbles are returned to the mill.

While the linear, bivariate correlations are useful to gain an understanding of the data set and aid the interpretation of model results, these measures are unable to capture any nonlinear correlations between variables. Nonlinear models are required to discover such patterns.

4.4 RF modelling of the SAG circuit operator

The next step in the procedure as shown in Figure 17 is to evaluate the general predictability of the data set according to the model formulation. In this section, a RF model is employed to determine the upper limit for predictive performance on the SAG data set.

4.4.1 The F-1 score for imbalanced data sets

As mentioned in the previous section, a pebble bypass event is rare compared to the number of samples where the pebble circuit operates as normal. This leads to the target variable being imbalanced, with the number of samples corresponding to normal pebble circuit operation (0) outnumbering bypass events (1) with a ratio of 20:1. Certain metrics do not deal well with this type of imbalance, and can show a bias towards the majority class. For example, a classifier that always predicts “0” (normal pebble circuit operation) will score a classification accuracy of 95% on this data set, when the classification accuracy is defined as the percentage of correct predictions as in equation (9).

Here, the F1-score was used to obtain a reasonable metric of the classification performance on the imbalanced data set, as defined below:

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{20}$$

In this context, the precision describes the fraction of samples correctly classified as bypass events (true positives) against the total number of samples classified as bypass events (true positives and false negatives):

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{21}$$

The recall describes the fraction of samples correctly classified as bypass events against the true number of bypass events:

$$recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{22}$$

True – and false positives and true - and false negatives are determined according to the logic in Table 4.

Table 4: General confusion matrix to determine F1-score

		Predicted class	
		No bypass (0)	**Bypass (1)**
True class	**No Bypass (0)**	True negative	False positive
	Bypass (1)	False negative	True positive

Ideally, high precision from a classifier, corresponding to few false positives, is expected. Yet, this precision cannot be too high without introducing a significant number of false negatives, leading to a decrease in the classifier recall. This trade-off between precision and recall is very commonly encountered when evaluating classifiers on imbalanced data sets.

The F1-score attempts to balance these two concepts by taking the harmonic mean of the two values. In this section, the F1-score is used to evaluate the classifier performance, where the highest F1-scores are desired.

4.4.2 Training a RF model on the SAG circuit data

A RF model was trained for the classification task on the SAG circuit data using the parameters summarised in Table 5. The number of predictors to sample was maintained at the default value suggested by Liaw and Wiener (2002). For classification RFs, a single sample can make a new leaf. The OOB scores of the model is shown in Figure 23. The figure shows a sharp decrease in the OOB error with the addition of up to 40 trees, after which the improvement plateaus with the addition of more trees. No significant improvement to the OOB score was observed beyond 200 trees.

Table 5: RF model parameters for the classification task on SAG circuit data

Parameter	Specification
Number of trees (K)	200
Predictors sampled at each split	$floor(\sqrt{M}) = 2$
Minimum leaf size	1

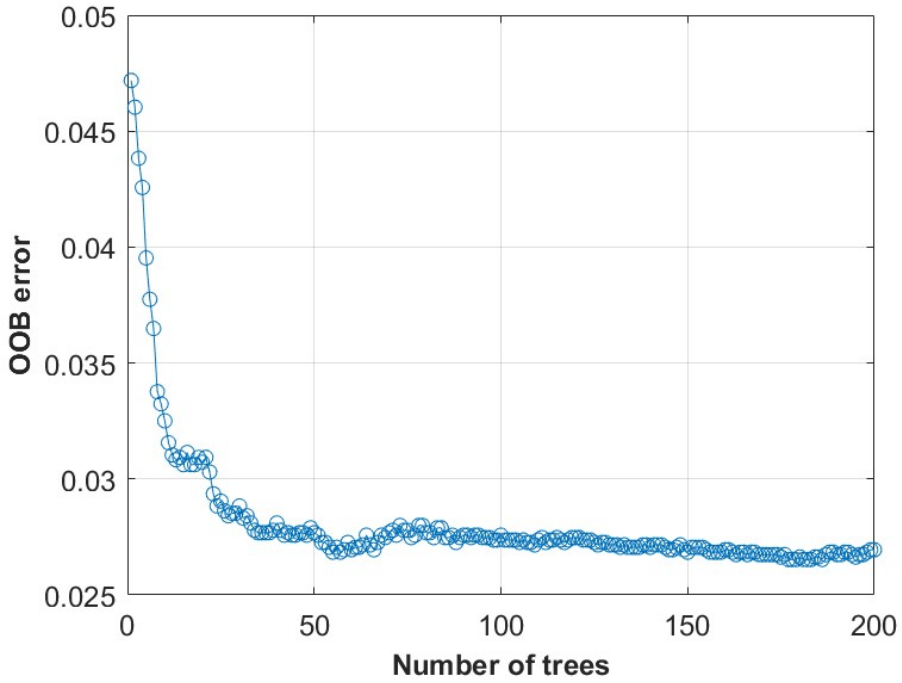

Figure 23: OOB scores of RF model of SAG circuit data with 200 trees

To evaluate the predictions of a typical RF model on a test set, a model was trained on a partition of 80% of the original data set and tested on the remaining 20%. The partitions were

stratified so that both partitions contain the same fraction of bypass events as in the original data set. The results are presented in the confusion matrix in Table 6. The results in the table correspond to a F1-score of 0.671.

Table 6:Confusion matrix of RF model predictions on an independent test set

		Predicted class	
		0	**1**
True class	**0**	1814	6
	1	40	47

In Table 6, entries in the diagonal from top-left to bottom-right correspond to samples correctly classified, while the other diagonal indicates samples that were misclassified. The table shows that 46% of bypass events were misclassified as normal pebble circuit operation. The results indicate an overall precision of $\frac{47}{47+6} = 0.89$, and a recall of $\frac{40}{40+4} = 0.46$. With the current model parameters, too many bypass samples are being misclassified as false negatives. In the next section, it is attempted to decrease this fraction of false negatives. For comparison, the classification accuracy as defined in equation (9) would result in a value of 0.976, thus overestimating classification results on the imbalanced data set.

4.4.3 Decreasing false negatives with custom misclassification costs

A custom cost matrix was implemented during the RF model training to reduce the number of false negatives of bypass events. This implies that a higher cost is associated with misclassifying a bypass event as normal pebble circuit operation, as indicated in Table 7. The table shows that misclassifying a bypass event has a cost 20 times higher than misclassifying a non-bypass event. The proportion of bypass events against normal operation determined this weighting factor. Correct classifications on the diagonal from top-left to bottom-right incur zero cost. The weighting factors are multiplied with the Gini index factors in equation (6) and equation (7) when determining the optimal split location. This procedure has a similar effect to inserting a prior probability distribution into the model before construction. An alternative way to deal with the false negatives is to randomly under-sample the majority class to the same number of samples as the minority class.

Table 7: Custom cost matrix for RF model construction to reduce false negatives

		Predicted class	
		0	**1**
True class	**0**	0	1
	1	20	0

The classification results of a RF model trained with the custom misclassification costs in Table 7 are shown in Table 8. The table shows a significant decrease in the number of false positives due to the addition of the custom misclassification costs. These costs have also not significantly increased the number of false negatives. The new RF models scored a precision of $\frac{58}{58+7} = 0.89$ and recall of $\frac{58}{58+29} = 0.67$, for an overall F1-score of 0.76. It was decided to adopt the proposed misclassification costs for all models going forward. The OOB errors for a RF with 200 trees and associated misclassification costs are shown in Figure 24.

Table 8: Confusion matrix of RF model with custom misclassification costs indicated in

		Predicted class	
		0	**1**
True class	**0**	1813	7
	1	29	58

While the OOB error in Figure 24 looks similar to the results of the RF model in Figure 23, the OOB error values are notably higher with about a factor of 10. This is a result of the higher cost associated with misclassifying bypass events as normal pebble circuit operation. Next, RF models with a varying number of trees were constructed on a training set consisting of 80% of the original data. The F1-scores of these models were calculated on an independent test set consisting of the remaining 20% of the data. The F1-scores as a function of the number of trees are shown in Figure 25.

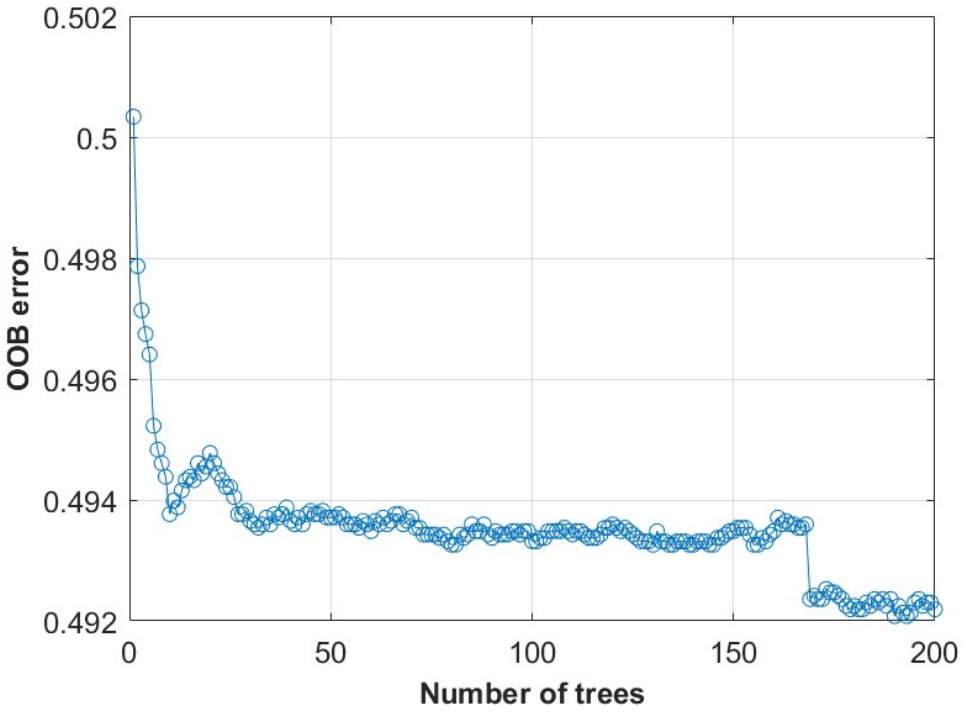

Figure 24: OOB scores of RF model of SAG circuit data with 200 trees and custom misclassification costs

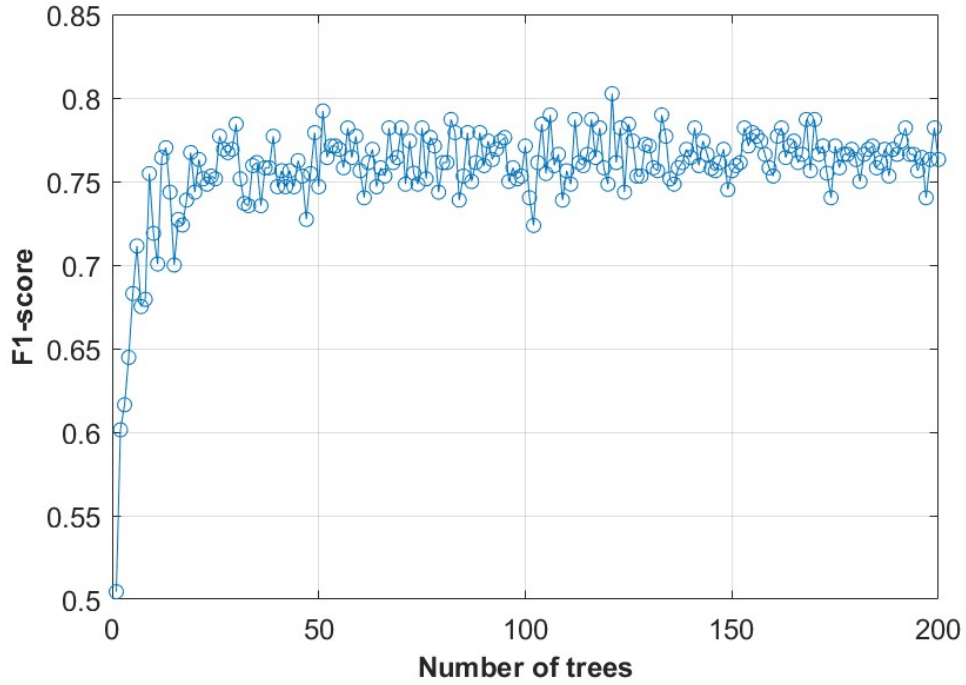

Figure 25: F1-scores of RF model of SAG circuit data with a varying number of trees and custom misclassification costs

Figure 25 shows that a single tree can achieve an F1-score of approximately 0.5. Adding 10 trees or more increases the F1-score up to between 0.75-0.8. The F1-scores will be compared to the scores achieved by single, simplified decision trees in the following section.

This section has demonstrated that the classification task is not trivial. Even with custom misclassification costs, a significant amount of bypass events is misclassified as false negatives. This could be a consequence of the fact that the model is attempting to predict human decision-making. While it is thought that there is a general pattern leading to the bypass events, the decisions made ultimately rely on subjective assessments of circuit conditions and inconsistent choices between different individuals. However, as shown above, a significant amount of these events can be predicted. The rule extraction procedure can be used to identify the most prominent behavioural patterns leading to these events.

4.4.4 Variable importance analysis of a RF model on the SAG circuit data

Variable importance measures were calculated to quantify the relative contribution of each variable to the RF model. Both the permutation importance and the Gini variable importance was calculated according to equation (13) and equation (15), respectively.

The variable importance measures were calculated for 30 instances of the model. An independent test set consisting of 20% of the data set was held out at each iteration to quantify the model generalisation. The test set was randomly sampled at each iteration, thereby generating 30 models trained on different partitions of the data set. Boxplot distributions of the permutation variable importance measures and Gini variable importance are shown in Figure 26 and Figure 27, respectively. As seen in the figures, the average F1-score on the test set over the 30 runs was 0.663.

As proposed by Auret and Aldrich (2012), a normally distributed random vector was added as an additional predictor column. Since the random variable has no association to the response and is not correlated to any of the variables, the importance measure of the random variable serves as a lower limit for a variable to be considered important. The random variable is denoted “R” in Figure 26 and Figure 27. As evidenced by the figures, the RF model deemed all variables more significant than the random variable.

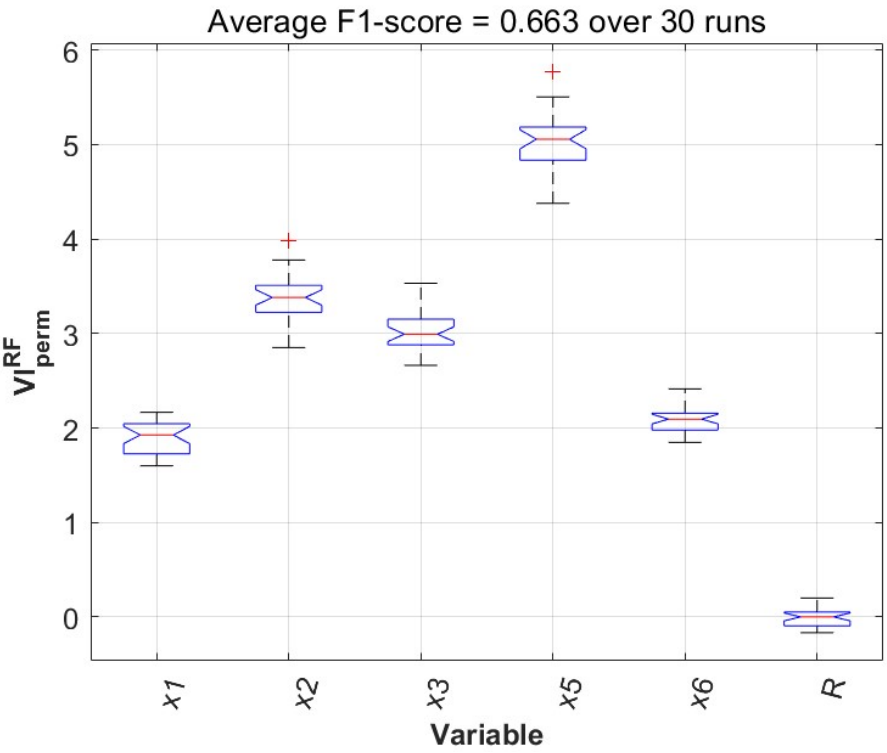

Figure 26: Distribution of permutation variable importance measures of SAG circuit RF model with 200 trees. Distributions were calculated over 30 runs of the model.

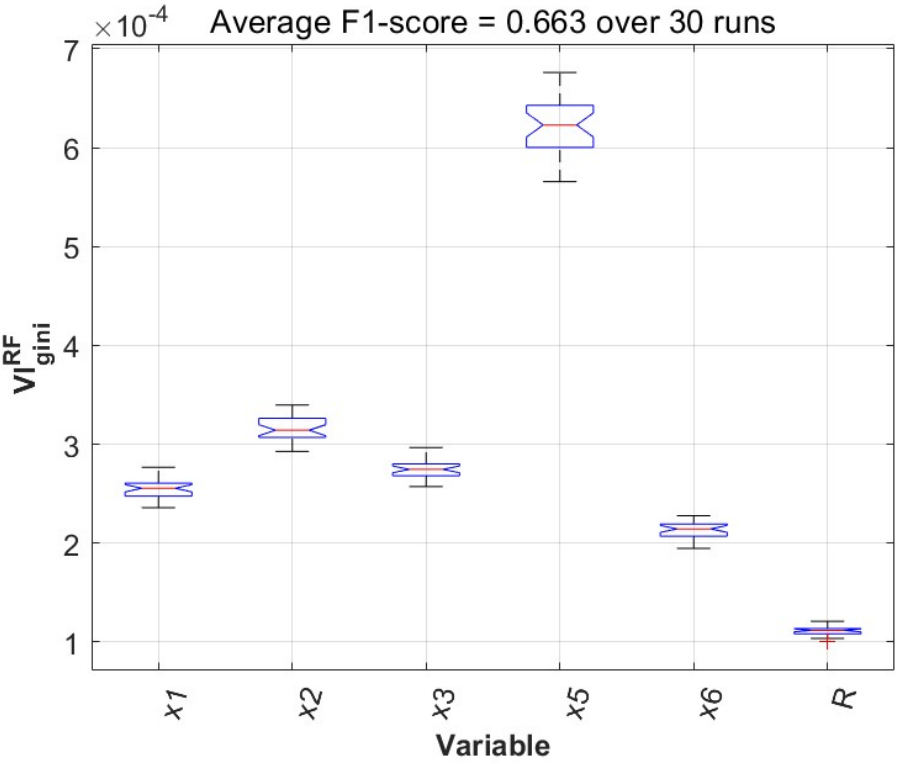

Figure 27: Distribution of Gini variable importance measures of SAG circuit RF model with 200 trees. Distributions were calculated over 30 runs of the model.

Both variable importance measures show markedly similar distributions over 30 instances of the model. Both importance measures deemed the water addition rate $\mathbf{X_5}$ as most influential for model performance. At this stage, this phenomenon was not completely understood, but was in accordance with the correlation analysis in section 4.3.3. The dry feed rate $\mathbf{X_2}$ was deemed second most significant by both measures, while the remaining variables had less influential contributions. Surprisingly, the pebble discharge rate $\mathbf{X_3}$ had a smaller contribution than previously thought, since operators had sometimes bypassed the pebble circuit in response to high loads on $\mathbf{X_3}$ that could have overloaded the pebble crusher. Although the pebble circuit bypass is thought to be a response to rapid increases in the power draw $\mathbf{X_1}$, this variable was not identified as very significant.

Overlooking the small discrepancies between the two measures, ideally the decision tree used for rule induction would show similar spreads among contributing variables as that in Figure 26 and Figure 27. Aside from the F1-score, a similarity in these variable importance distributions could serve as an additional indication that the structure of the tree is sufficiently representative of the higher capacity RF model.

4.5 Decision tree induction on the SAG circuit data

4.5.1 Classification tree induction and simplification

The previous section demonstrated that the status of the pebble circuit is indeed predictable from other operational variables. The RF model demonstrated that a F1-score of 0.5 could be obtained using a single decision tree. While this constitutes a considerable drop in accuracy from the full RF model, simpler decision tree models should still be able to extract simple rules describing the most common patterns leading to bypass events.

To further investigate the abilities of a single decision tree, a model was generated using the CART algorithm with the parameters in Table 9. The full tree is shown in Figure 28.

Table 9: CART model parameters for decision tree induction on SAG circuit data

Parameter	Specification
Minimum parent node size	10
Minimum leaf size	1
Predictors sampled at each split	All
Misclassification costs	Table 7

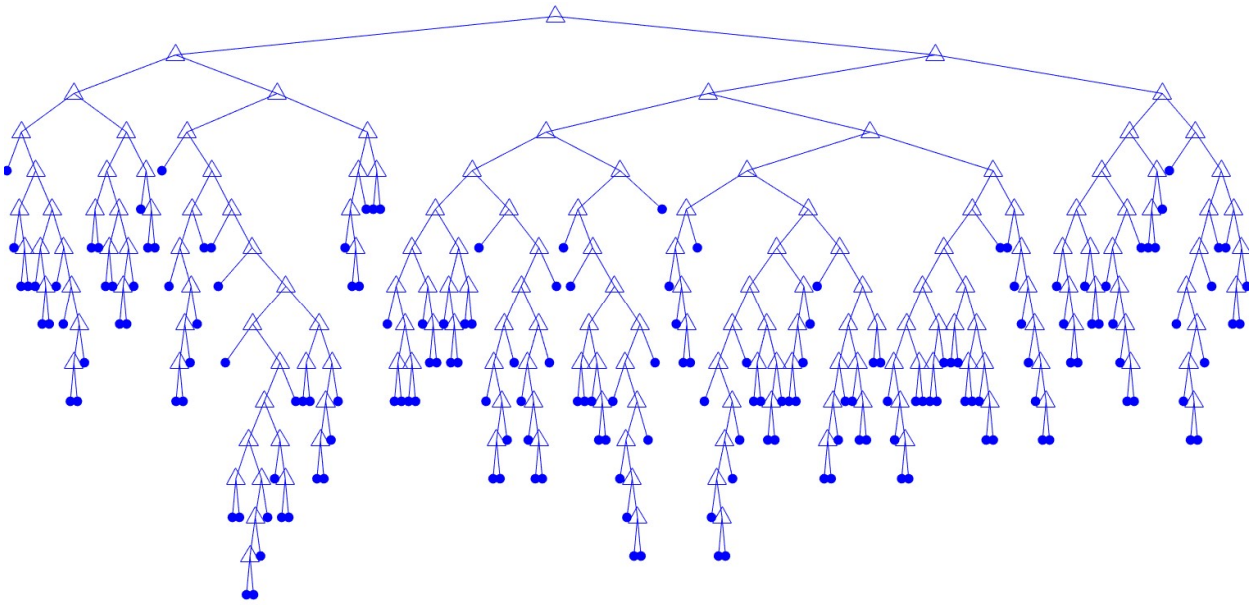

Figure 28: Decision tree predicting SAG pebble circuit status with no restrictions on branch growth capabilities

The tree in Figure 28 consists of 176 branch nodes and 177 leaf nodes, resulting in a F1-score of 0.44. The predictions of this tree on an independent test set is presented in the confusion matrix in Table 10.

Table 10: Confusion matrix of decision tree with parameters in Table 9 on an independent test set

		Predicted class	
		0	**1**
True class	**0**	1744	76
	1	41	46

As expected, using a single decision tree has resulted in an increase in misclassifications of bypass events, with a significant number of false negatives reappearing. Also, there has been

a significant increase in false positives. The reduced F1-score is thus an effect of decreased precision and recall. However, most bypass events remain correctly classified.

The large number of nodes in the tree in Figure 28 renders the decision rules extracted from the tree unsuitable for interpretable decision support. For example, at its maximum depth the tree contains 16 nodes, implying 15 terms in the rule antecedent. It needs to be investigated whether the large number of splits are contributing to the generalisation performance, or simply overfitting the training data set. This phenomenon is examined in Figure 29. Starting with the number of branch nodes in Figure 28, decision trees were trained with progressively less nodes allowed. This was enforced by constructing the tree with an additional parameter to that of Table 9, whereby the maximum number of branch splits is specified. The F1-score of the predictions on the training and test data set was noted for each tree.

Figure 29 demonstrates that up until the addition of 40 split nodes, there is a sharp increase in the tree accuracy with additional nodes. Upwards of 40 nodes, there is no significant improvement to the tree performance on the test data set, and additional nodes are overfitting the training data. However, using 10 and 20 splits leads to F1-scores around 0.3 and 0.35, respectively. This indicates that these trees contain a significant number of misclassifications. This stresses the importance of evaluating the accuracy of each rule extracted from such a tree.

To avoid the computational expense suffered in the generation of Figure 29, the procedure could be reversed. Starting with a single split, the number of splits could be progressively increased until the test set accuracy stops increasing with the larger number of splits. This is a similar procedure to the validation patience employed when training neural networks and will stop training once the model starts overfitting the training set.

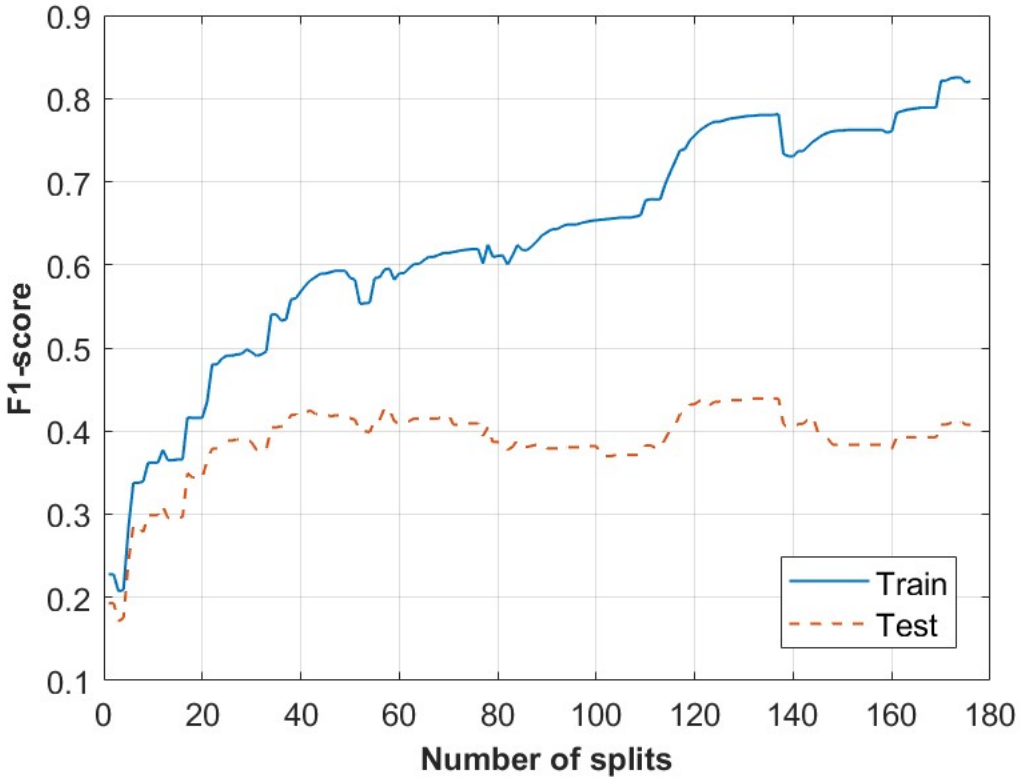

Figure 29: Classification tree F1-score on a training and test set as a function of the maximum number of split nodes allowed

Figure 29 indicates that the number of splits in the trees can be significantly reduced without suffering too much losses in accuracy on the test set. While some reduction in accuracy will necessarily occur, the tree structure and branches are significantly simplified. This is demonstrated in Figure 30 by visualising the tree structures with different limits on the number of splits allowed.

Figure 30 demonstrates that the F1-score increases as the number of split nodes increase, but so does the number of rules and average length of each rule in the rule set. The figure demonstrates that simple tree structures containing ten or twenty splits can achieve acceptable accuracies when compared to more complex trees with large numbers of splits. In this case study, it was selected to demonstrate the procedure on a tree with a maximum number of ten splits. This culminates in a rule set with a maximum amount of eleven rules, with the complexity of each rule limited to a maximum of four terms in the rule antecedent. The procedure can be repeated on a tree with a maximum of twenty splits, but this does result in some rules with five terms in the antecedent. Here, it is attempted to keep the number of terms in the antecedent to a minimum to increase interpretability.

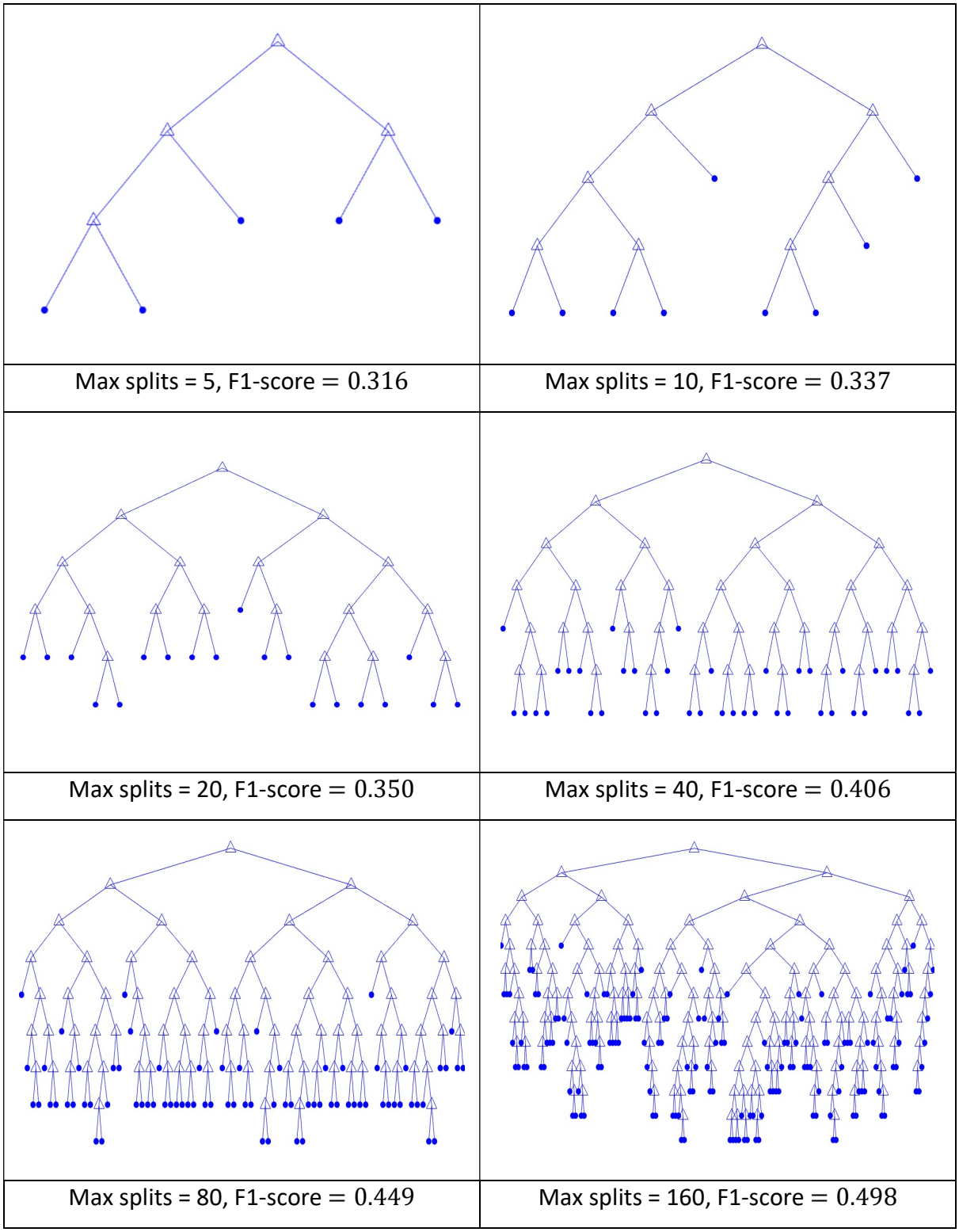

Figure 30: Decision trees on the SAG circuit data set with maximum number of splits imposed. F1-score of each tree on an independent test set is included.

4.5.2 Classification tree selection and variable importance analysis

The rest of the procedure will focus on decision trees with a maximum of ten splits allowed. These trees result in a small, but reasonably accurate rule set consisting of rules that are not

too complex and easily comprehensible. In this case, the tree with 20 splits would also be a good candidate, but a tree with ten splits was selected for demonstration.

The accuracy and variable contributions of such a tree was determined through 30 instances of the tree, using the parameters indicated in Table 9 and the additional restriction on the maximum number of splits. The permutation and Gini variable importance distributions over the 30 runs are depicted in Figure 31 and Figure 32, respectively. Again, a random vector was added as an additional predictor to serve as a threshold for what could be considered as important.

In contrast to the importance measure of the RF models, both measures do not regard the cyclone pressure $\mathbf{X_6}$ as more significant than the random variable $\mathbf{R}$. This variable must not have found expression in most of the 30 instances of the decision tree model. Other variables show similar importance distributions as in the unrestricted RF model. Again, both measures identify the water addition rate $\mathbf{X_5}$ as the most influential variable in the models.

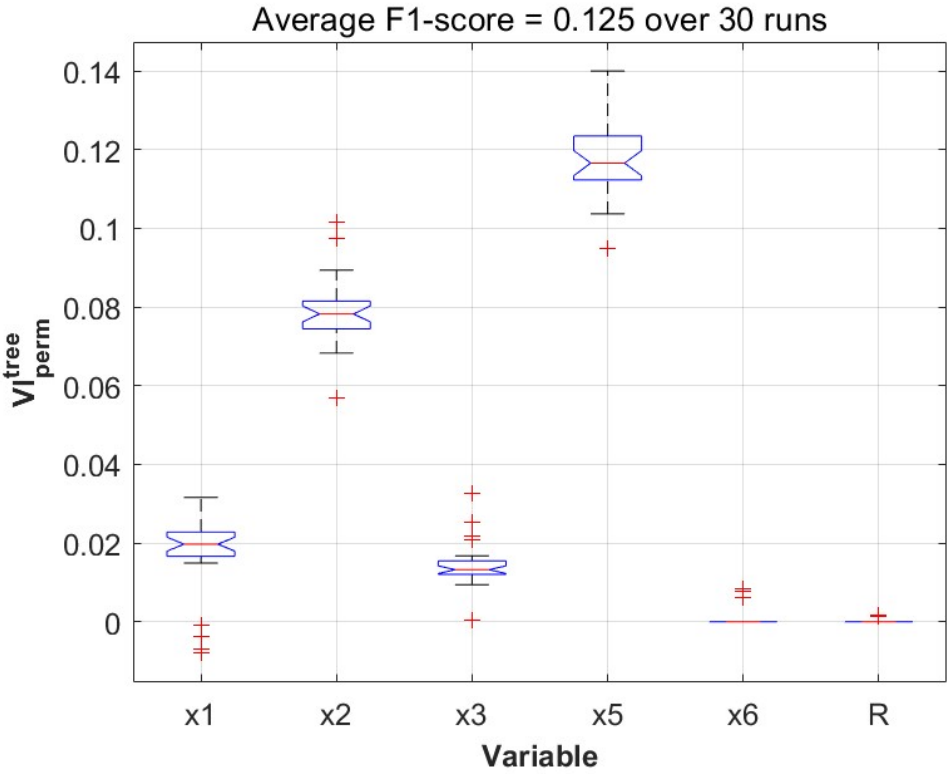

Figure 31: Distribution of permutation variable importance measures of SAG circuit decision tree models with a maximum of ten splits. Distributions were calculated over 30 runs of the model.

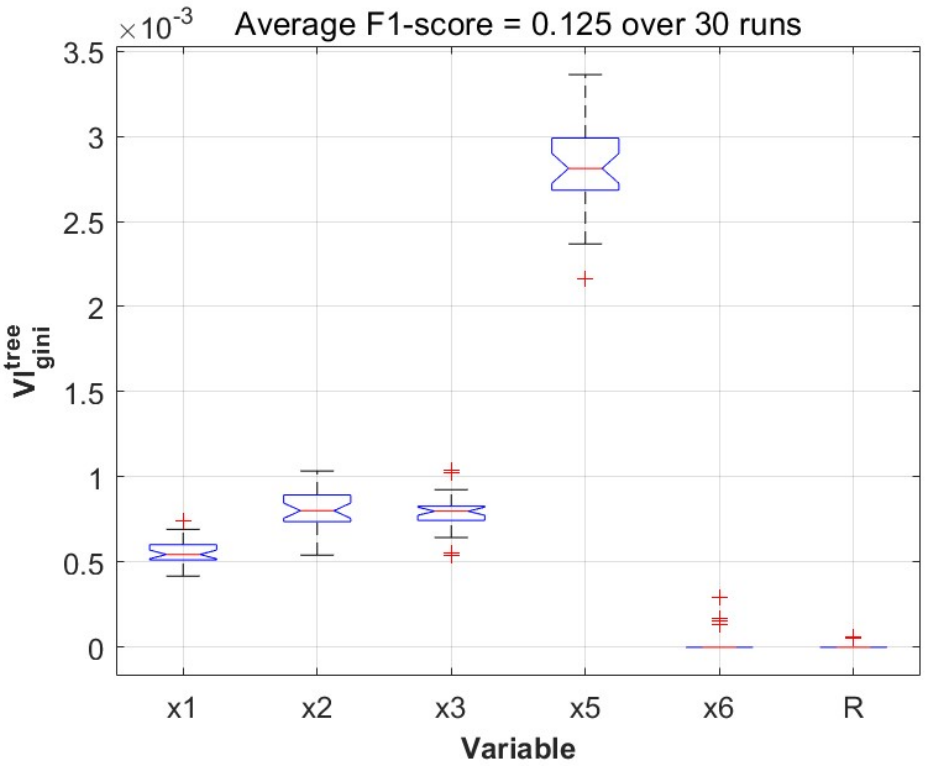

Figure 32: Distribution of Gini variable importance measures of AG circuit decision tree models with a maximum of ten splits. Distributions were calculated over 30 runs of the model.

The average F1-score over the 30 model runs is significantly lower than the previous results in trees with a maximum of ten splits. This demonstrates the instability of decision trees and the reliance of the tree structure on the specific partition of data used during training. Different instances of the decision tree model will generate different rule sets. After careful evaluation of the rules produced, rules from different trees could be combined into a rule set for decision support.

Here, the procedure is demonstrated on one such instance of the model. The final classification tree selected for rule extraction is show in Figure 33. The tree contains eight branch nodes leading to nine leaf nodes, with a maximum depth of five. Rules predicting pebble circuit bypass events are circled in red. The confusion matrix of predictions on an independent test set is given in Table 11.

Table 11: Confusion matrix of classification tree in .

		Predicted class	
		0	**1**
True class	**0**	1616	204
	1	29	58

The tree manages to score a precision of $\frac{58}{58+204} = 0.22$ and a recall of $\frac{58}{58+29} = 0.67$ for a total F1-score of 0.331. Clearly, the tree predicts a significant number of false positives. This will result in a decision support system wrongfully predicting a period of bypass, where the event might not occur. Operators will need to take this into account when evaluating an activated rule and augment the decision based on their heuristic circuit knowledge. However, the recall of the classifier is acceptable and manages to catch most bypass events, which can then be analysed and possibly avoided.

In this subsection, the variable importance measures were calculated to easily compare with the measures calculated for the RF model. However, a decision tree with ten splits such as in Figure 33 can be visually analysed to determine which variables play a significant role in the model. The most important variable contributions ($\mathbf{X_5}$, $\mathbf{X_2}$ and $\mathbf{X_1}$) can be identified in the top nodes in the tree in Figure 33. The cyclone pressure, $\mathbf{X_6}$, was not deemed important by the importance measures, and does not find expression in the decision tree.

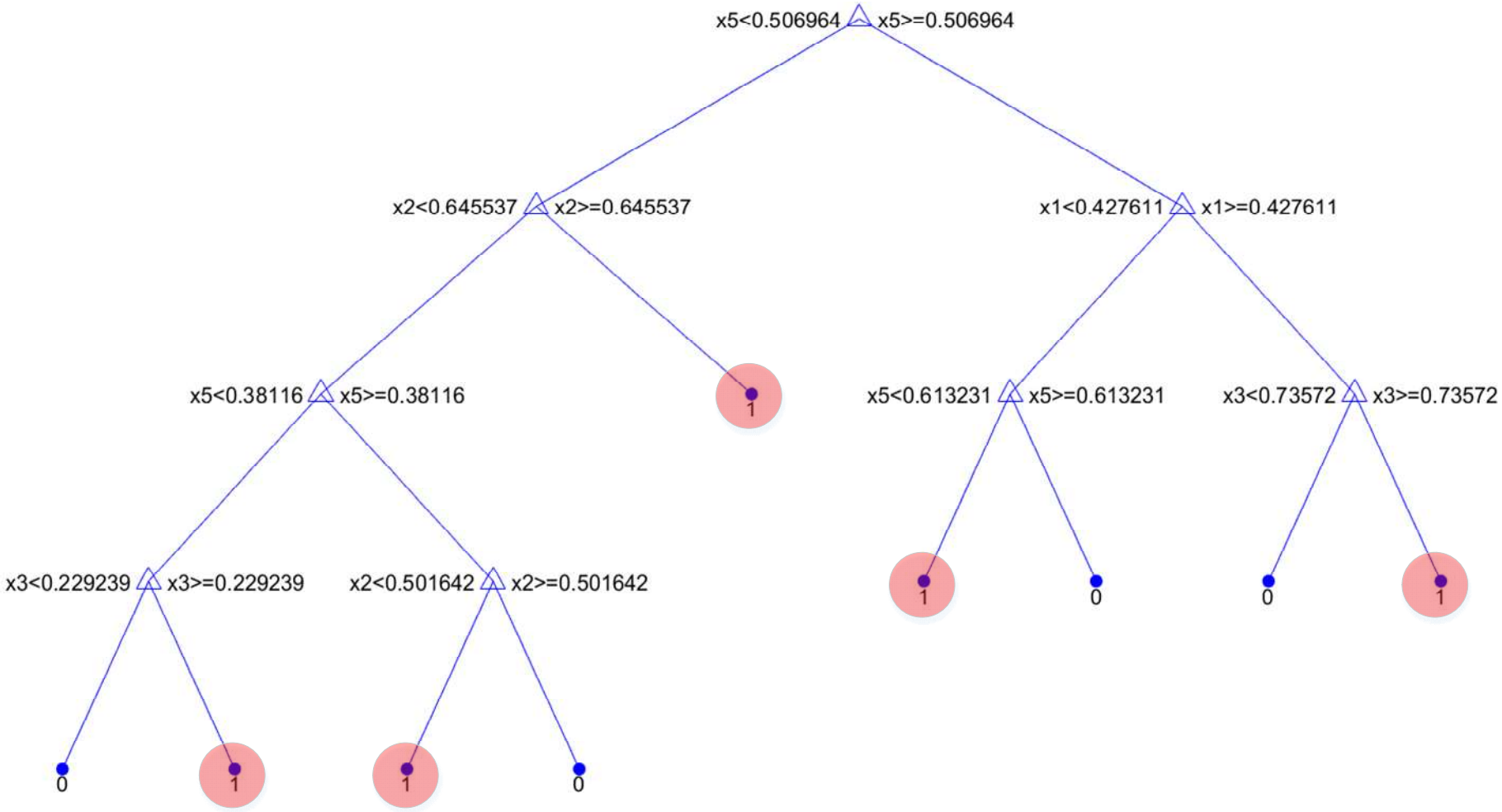

Figure 33: Final classification decision tree with a maximum of 10 node splits. Leaf nodes resulting in pebble bypass events are circled in red.

4.6 SAG circuit rule extraction and evaluation

4.6.1 Classification rule extraction and simplification

From the decision tree in Figure 33, each path from the root node to a leaf node is represented as a rule consisting of the conjunction of tests on the internal nodes on the path. Individual tests on a specific path are connected by an "and" operator, since a data sample needs to pass all the tests on a path to belong to the specific leaf node. Rules extracted for each of the leaf nodes in Figure 33 are displayed in Table 19.

Table 12: Simplified classification rules from the decision tree in Figure 33

Number	Rule	Number	Rule
1	$If\ \mathbf{X_5} < 0.381$ $and\ \mathbf{X_2} < 0.646$ $and\ \mathbf{X_3} < 0.229;$ $Then\ \mathbf{y} = 0$	2	$If\ \mathbf{X_5} < 0.381$ $and\ \mathbf{X_2} < 0.646$ $and\ \mathbf{X_3} \geq 0.229;$ $Then\ \mathbf{y} = 1$
3	$If\ 0.381 \leq \mathbf{X_5} < 0.507$ $and\ \mathbf{X_2} < 0.502;$ $Then\ \mathbf{y} = 1$	4	$If\ 0.381 \leq \mathbf{X_5} < 0.507$ $and\ 0.502 \leq \mathbf{X_2} < 0.646;$ $Then\ \mathbf{y} = 0$
5	$If\ \mathbf{X_5} < 0.507$ $and\ \mathbf{X_2} \geq 0.646;$ $Then\ \mathbf{y} = 1$	6	$If\ 0.507 \leq \mathbf{X_5} < 0.613$ $and\ \mathbf{X_1} < 0.428;$ $Then\ \mathbf{y} = 1$
7	$If\ \mathbf{X_5} \geq 0.613$ $and\ \mathbf{X_1} < 0.428;$ $Then\ \mathbf{y} = 0$	8	$If\ \mathbf{X_5} \geq 0.507$ $and\ \mathbf{X_1} \geq 0.428$ $and\ \mathbf{X_3} < 0.736;$ $Then\ \mathbf{y} = 0$
9	$If\ \mathbf{X_5} \geq 0.507$ $and\ \mathbf{X_1} \geq 0.428$ $and\ \mathbf{X_3} \geq 0.736;$ $Then\ \mathbf{y} = 1$		

The rules in Table 12 were simplified in the sense that if a rule contained more than one occurrence of a single variable in the antecedent, the separate occurrences of the specific variable was simplified into a single expression. Two illustrative examples are given below in Table 13.

Table 13: Simplifying rules with multiple occurrences of a single variable in the rule antecedent

Number	Original	Simplified
1	$If\ \mathbf{X_5} < 0.507,$ $and\ \mathbf{X_2} < 0.646,$ $and\ \mathbf{X_5} < 0.381$ $and\ \mathbf{X_3} < 0.229;$ $Then\ \mathbf{y} = 0$	$If\ \mathbf{X_5} < 0.381$ $and\ \mathbf{X_2} < 0.646$ $and\ \mathbf{X_3} < 0.229;$ $Then\ \mathbf{y} = 0$
3	$If\ \mathbf{X_5} < 0.507,$ $and\ \mathbf{X_2} < 0.646,$ $and\ \mathbf{X_5} \geq 0.381$ $and\ \mathbf{X_2} < 0.502;$ $Then\ \mathbf{y} = 1$	$If\ 0.381 \leq \mathbf{X_5} < 0.507$ $and\ \mathbf{X_2} < 0.502;$ $Then\ \mathbf{y} = 1$

4.6.2 Evaluating the utility of SAG circuit decision rules

Each rule in Table 12 was evaluated according to the metrics proposed in section 3.3. The metrics for each rule are presented in Table 14, wherein rules predicting a bypass event have been highlighted in grey. For the rule complexity, expressions of the form "$0.381 \leq \mathbf{X_5} < 0.507$" are regarded as two splits, since it conceptually still requires evaluations of two conditions by the user.

In Table 14, the rules predicting bypass events all have an accuracy below 0.5. If the prediction were a simple majority vote of all the samples belonging to the rule, the rules would naturally predict normal operation of the pebble circuit. However, the higher misclassification cost imposed on misclassifying actual bypass events outweighs the cost of misclassifying normal operation samples. Thus, the higher misclassification cost allows for the identification of the operational states wherein these bypass events are most likely to occur. Intuitively, the accuracy is thus better interpreted as an indication of the probability of a bypass event occurring in the operational state specified by the rule.

The rules predicting bypass events, which are marked in grey, only account for relatively small portions of the data set. This was expected since such events are closer to a fault condition than normal operation. However, there is significant benefit in identifying these periods so that the frequency of such events can be decreased.

Table 14: Metrics to evaluate the utility of decision rules in Table 12

Number	Supporting samples (% of training data set)	Accuracy (probability of predicted class)	Complexity (number of splits)
1	1.45	1.000	3
2	2.24	0.246	3
3	0.48	0.189	3
4	9.82	0.988	4
5	5.79	0.360	2
6	2.35	0.196	3
7	6.21	0.970	2
8	68.17	0.989	3
9	3.49	0.102	3

The rules predicting bypass events only account for relatively small portions of the data set. This was expected since such events are closer to a fault condition than normal operation. However, there is significant benefit in identifying these periods so that the frequency of such events can be decreased.

All rules extracted contain between two and four expressions to evaluate in the rule antecedent. The restriction on the maximum number of splits allowed during tree growth prohibits the tree from growing complex rules. The pre-pruning of the tree effectively manages this criterion of the utility of a rule.

4.6.3 Analysing selected decision rules

A selection of rules from Table 12 is critically analysed for their use in decision support below. Consider a closer inspection of rule 5:

$$\begin{aligned} &If\ \textbf{Water addition rate} < 0.507 \\ &\quad and\ \textbf{Dry feed rate} \geq 0.646; \\ &\quad Then\ \textbf{Bypass pebble circuit}\ (Probability = 0.36) \end{aligned} \tag{23}$$

The rule states that at a higher dry feed rate coupled with a lower water addition rate, corresponding to an increased solids density in the SAG mill, there is a 36% chance the circuit is bypassed. The inadequate water addition is retaining too much material inside the mill,

leading to an increase in the mill load and power draw. In 36% of the cases, the mill must have overloaded, and the pebble circuit was bypassed.

Without complete knowledge of ore characteristics, it is difficult to determine the exact reason for this occurrence. However, softer ores were likely breaking down quicker and producing more fines. Coupled with the inadequate water addition, the mill retains the fines and starts overloading. To decrease the probability of this event in the future, metallurgists could reconsider the SAG discharge density targets given to operators based on such ore sources. Additionally, the decision rule could be displayed on a HMI, thus warning operators that the circuit is currently in a state that has historically led to bypass events.

Next, rule 9 is repeated and analysed below:

$$\begin{aligned} &If\ \textbf{Water addition rate} \geq 0.507 \\ &\quad and\ \textbf{Mill power draw} \geq 0.428 \\ &\quad and\ \textbf{Pebble discharge rate} \geq 0.736; \\ &\quad Then\ \textbf{Bypass pebble circuit}\ (Probability = 0.102) \end{aligned} \tag{24}$$

Rule 9 states that when the water addition rate and power draw are at medium levels and above, while the pebble discharge rate is high, there is a 10% chance that the pebble circuit would be bypassed. This situation could arise when the mill feed suddenly changes to a more competent ore source, or a larger portion of mid-size fraction material is being fed to the mill. The mill load and power draw are not necessarily high, but a large amount of competent mid-size material is being discharged from the mill. The amount of pebbles discharged could ascend to a level where the pebble crusher and circuit conveyors are unable to deal with the increased load.

Depending on the mill fill level and the amount of power available, operators might choose to draw a higher portion of large rocks from the stockpile to attempt to break down some of the mid-size material. Alternatively, operators may choose to draw an increased fraction of finer material from the stockpile to maintain the mill throughput while not further contributing to the generation of pebbles. Metallurgists could further investigate the particle size distributions received from the preceding crusher section to deal with these occurrences.

Rule 9 is contrasted by rule 8, which obtained the highest amount of support in the data set:

$$\begin{aligned} &If\ \textbf{Water addition rate} \geq 0.507 \\ &\quad and\ \textbf{Mill power draw} \geq 0.428 \\ &\quad and\ \textbf{Pebble discharge rate} < 0.736; \\ &\quad Then\ \textbf{Normal pebble circuit operation}\ (Probability = 0.989) \end{aligned} \tag{25}$$

As seen in the tree in Figure 33, Rule 8 and rule 9 split the data space according to the specific value of the pebble discharge rate. In contrast to rule number 9, rule 8 predicts that at lower pebble discharge rates, the circuit was only bypassed 1.1% of the time. Thus, the combination of the two rules discover the explicit value of the pebble discharge rate, such that when this value is exceeded, the operator is ten times more likely to bypass the pebble circuit. The rule can alert an operator when approaching this specific operational state, hopefully triggering faster control action, and avoiding the bypass event.

4.6.4 Rule implementation for decision support

A demonstration of the decision tree model predictions in the form of the decision rules outlined above is shown in Figure 34. The top pane of the figure shows 100 samples of operational data corresponding to more than eight hours of operation. The cyclone feed pressure has been omitted for clarity since this variable does not occur in any of the rules. The top pane clearly indicates two bypass events occurring corresponding to a non-zero pebble discharge rate but zero pebble returns. The bottom pane in Figure 34 shows a curve depicting the ground-truth data of the occurrence of bypass events, as well as the prediction of the decision tree model in Figure 33 and the specific rule activated by the decision tree model.

Figure 34 shows that the decision tree model can correctly classify the operation as a bypass event when these events are occurring, with rule 5 activating in both cases where the pebbles are sent to a stockpile. Rule 5 activates because of the low water addition rates during the events. This is an action triggered by the operator to start building up the mill charge since the mid-size fraction has quickly been removed. More significantly, rule 9 was activated before each of the events occurred. The model recognised that the combination of power draw, water addition rate and pebble discharge rate has historically resulted in an increased probability of bypass events occurring. In both cases the activation of this rule was followed by a bypass event. Rule 9 was activated 30 minutes and 70 minutes before each of the events,

respectively. This pre-emptive activation of rule 9 suggests that the event could have been avoided if the operator had received a warning.

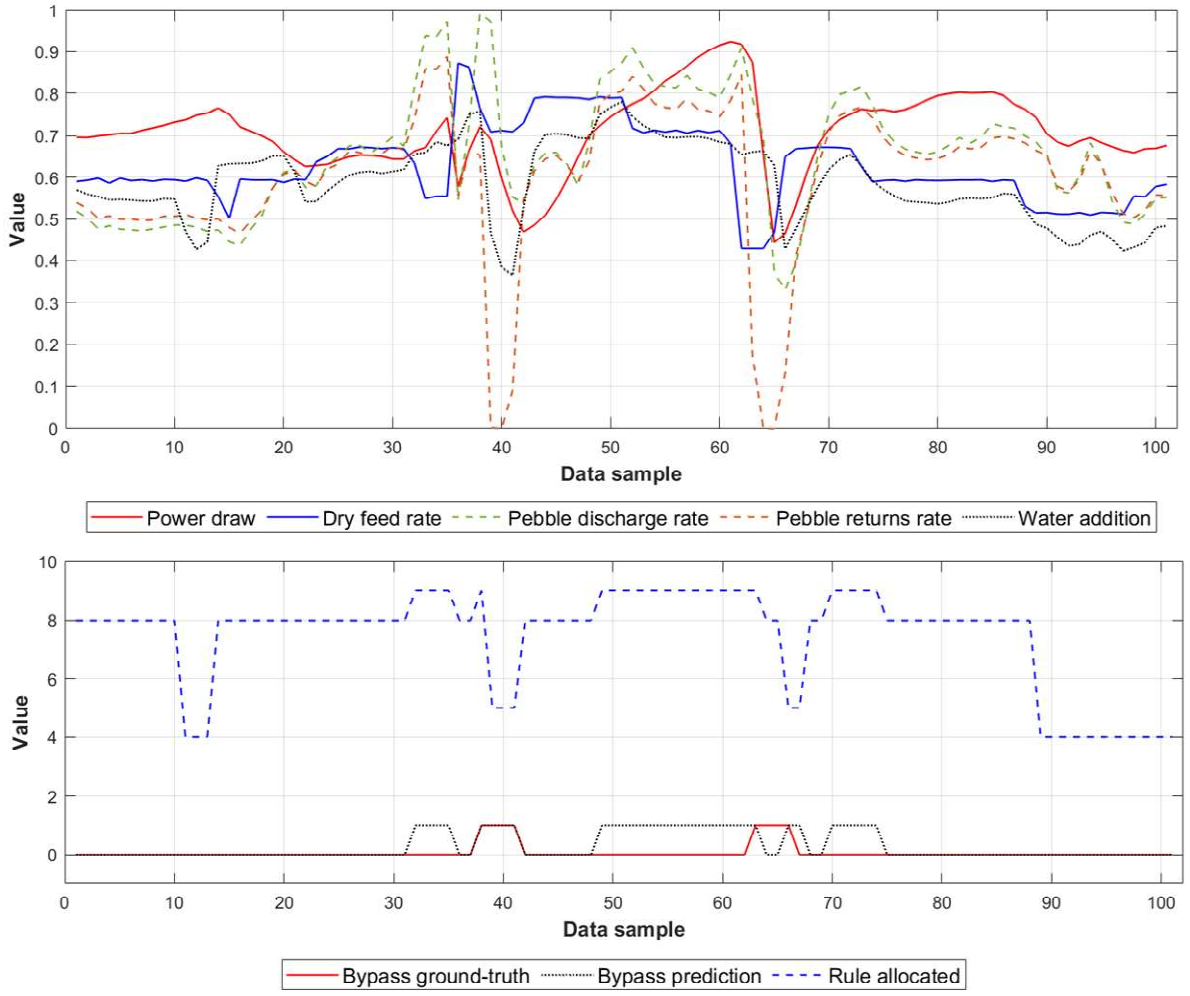

Figure 34: Top: SAG circuit operational data over 100 data samples. Bottom: Binary variable indicating ground truth of bypass events, Figure 33's decision tree model prediction of bypass events and specific rule activated in Figure 33 over the 100 data samples of operation.

While rule 9 can warn the operator that historically bypass events have occurred more often under these conditions, the correct action needs to be deduced by the operator based on the considerations mentioned earlier. During the first event there is still power available, and the operator could have chosen to draw coarser rock from the stockpile to attempt to break down some of the rapidly rising mid-size fraction material. In the second event, the mill has already reached the point of overload and a drastic step is required to reduce the load. However, the early activation of rule 9 could have provided the operator with sufficient warning to further reduce the dry feed rate to avoid the bypass event occurring.

In the example in Figure 34 the decision rules function as a predictive alert that the circuit is either approaching or already in an unfavourable state. The specific rule activated also provides some insight towards the factors causing the unfavourable state, i.e. the high pebble discharge rates causing the bypass in Figure 34. The alerts were triggered early enough to suggest that the bypass events could be avoided with the appropriate corrective action. The reduction in the frequency of such events, with more frequent control changes of smaller magnitude as depicted in Figure 9, could increase circuit production and stability over the longer term.

4.7 Summary

In this section, a case study was investigated to discover knowledge regarding a specific operator decision-making pattern. A classification tree was used to extract rules defining periods of operation wherein operators decided to bypass the pebble circuit in a SAG circuit. This was of interest to metallurgists, considering the adverse effect of this action on longer-term operation. The rule extraction procedure could successfully identify a small set of easily interpretable rules describing the most common operating patterns leading to bypass events.

Through the case study, it was recognised that simple decision tree models with hard restrictions on branch growth could achieve satisfactory results when compared to higher capacity ensemble models. While the complexity and accuracy of the model is decreased when compared to stronger models, the decision tree models have the added benefit that the models are interpretable as simple tree structures or decision rules. It was observed that the simple decision tree models could satisfactorily represent the structure of the ensemble models, through its ability to identify the same important variables and only small decreases in the model accuracy.

The classification problem proved to be non-trivial, with larger capacity RF models also struggling to predict when these decisions occurred. There was naturally a certain arbitrariness in some of these decisions. The modelling procedure was further complicated by the imbalanced data set, since bypass events only occurred a couple of times per day. However, it was demonstrated that with appropriate model parameters and a custom misclassification cost matrix, these rare events could be sufficiently identified by simple decision tree models.

The rules extracted are appropriate to provide decision support to process operators, either as explicit rules on an HMI, or process alarms alerting the operator when entering a state wherein bypass events have occurred historically. It was demonstrated in the case study that the decision tree model could predict certain occurrences of bypass events and provide early warnings to operators to act before the event occurs. While the decision rules do not necessarily provide operators with the correct control action, the variable tests in the rule antecedent provide guidance in diagnosing the current condition and determining the appropriate action.

5. Case study: Regression tree rules to predict the operational state in an AG circuit

In the second case study, the rule extraction methodology is applied to data from an industrial AG circuit. Rules were extracted from a regression tree characterising the operating patterns exhibited while processing two different feed blends. It is attempted to extract rules that could summarise operational heuristics in two periods leading to coarser and finer grind sizes, respectively. The information in such rules could be used as operational targets for future processing of similar blends. Similar to the previous case study, the section follows the general methodology depicted in Figure 17.

5.1 AG circuit description

A simple diagram of the AG circuit is shown in Figure 35. Fresh ore and water are fed to the AG mill. The mill discharge slurry flows into a discharge sump, from where the slurry is pumped to a wet screening operation. Screen oversize material is returned to the mill dry feed conveyor for regrinding. The screen undersize material is the final AG circuit product and sent along for downstream processing.

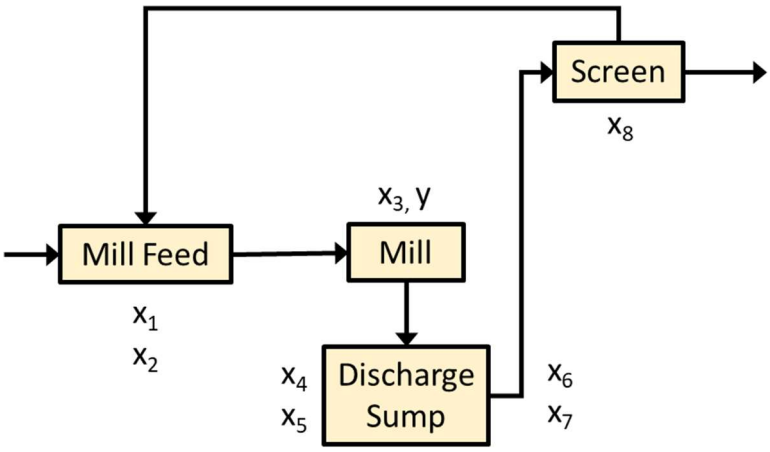

Figure 35: Simple diagram of the autogeneous grinding circuit

5.2 Raw AG circuit data description and exploration

5.2.1 Raw data collection

Operational data spanning four hours of circuit operation was collected from the AG circuit. The specific variables collected are summarised in Table 15.

Table 15: AG circuit operational variables collected

Variable identifier	Description
$\mathbf{X_1}$	Mill dry feed rate
$\mathbf{X_2}$	Mill water addition
$\mathbf{X_3}$	Mill load
$\mathbf{X_4}$	Discharge sump level
$\mathbf{X_5}$	Discharge sump outlet flowrate
$\mathbf{X_6}$	Discharge sump density
$\mathbf{X_7}$	Discharge sump water addition
$\mathbf{X_8}$	Classification screen current
$\mathbf{y}$	Mill power draw

The scaled plant data is displayed in Figure 36 below. The data samples were collected over five second intervals. Notably missing from the data is the recirculating load which was not measured at the time. Models need to account for the effect of this variable through the classification screen power draw.

The period of operation in Figure 36 depicts the transient response of the system to a changing ore blend. This is observed through the gradual increase in the mill power draw, $\mathbf{y}$, about halfway through the period of operation. This is primarily a result of the increase in the mill load, $\mathbf{X_3}$, around the same time. With a relatively constant dry feed rate, this was the result of a change in the feed ore blend to a more competent ore being processed. The ore with higher competence takes longer to break down to a size small enough to exit the mill through the grates, resulting in a higher load being help up in the mill.

The higher load means that more power is required to impart the rotating motion to the drum. A change in the feed PSD could result in a similar change to the mill load. AG mills are especially susceptible to such disturbances, since there is no steel grinding media in the mill and the rock itself is the only facilitator of particle breakage.

The period of operation in the higher power draw and load state delivered the desired transfer particle size from the milling circuit. At the lower power draw, the mill was

underloaded leading to a coarser transfer size to downstream processing. To avoid overloading the following ball milling circuit, operation in this state should have been avoided.

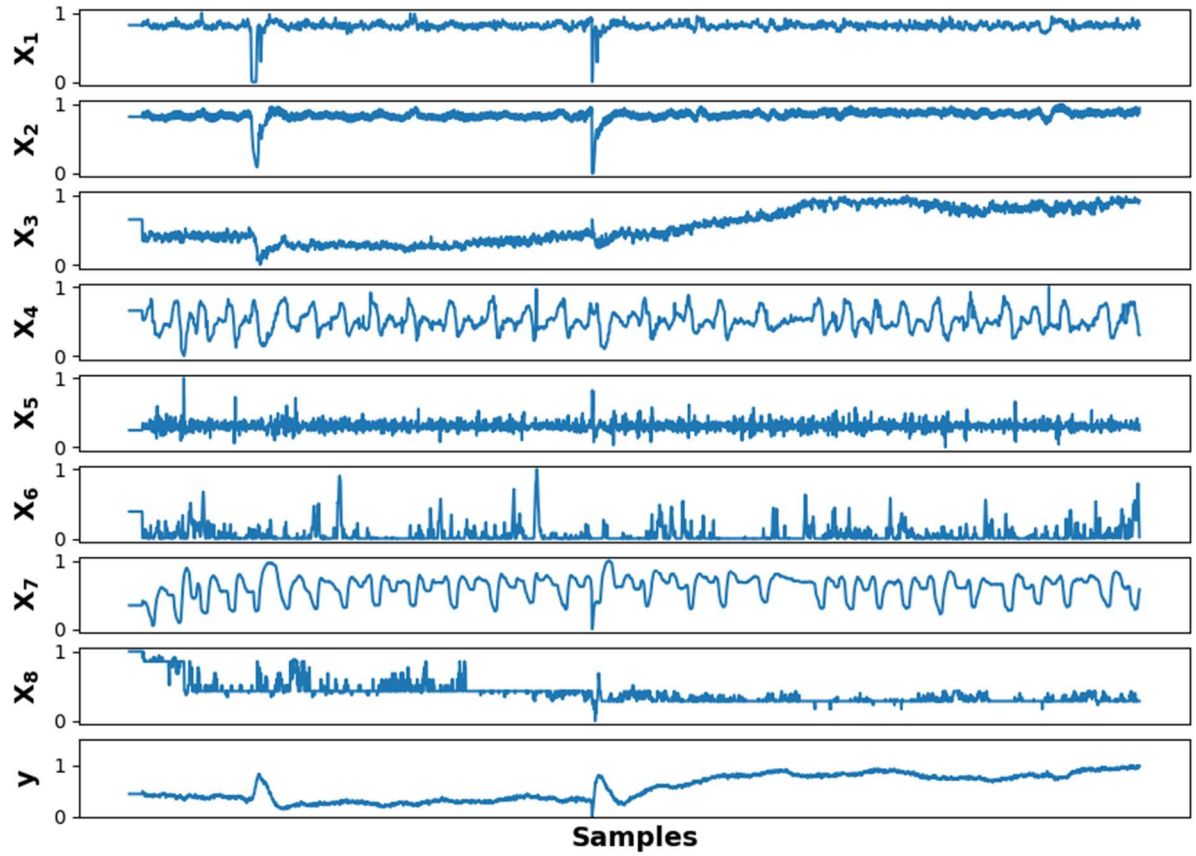

Figure 36: Normalised AG mill circuit data spanning four hours of operation

Based on the time series in Figure 36, the loading of the mill was not caused by operator action, but rather the system's natural response to the increased ore competency. Only small changes in the feed rate and mill water addition were observed. The lower mill load was accompanied by higher screen power draws, indicating higher levels of coarse material discharge. The discharge sump level is clearly controlled in a feedback loop using the discharge sump water addition.

In the absence of many operator control actions, decision tree algorithms could extract rules characterising the system's natural response to the ore change.

5.2.2 PCA of the AG circuit data

PCA was used to reduce the dimensionality of the nine-dimensional data set to visualise the variance exhibited in the circuit over the four hours of operation. The principal component

scores of the raw data in Figure 36 projected onto the first three principal components are shown in Figure 37. The figure shows that the first three principal components can collectively account for 75% of the variance in the original data set.

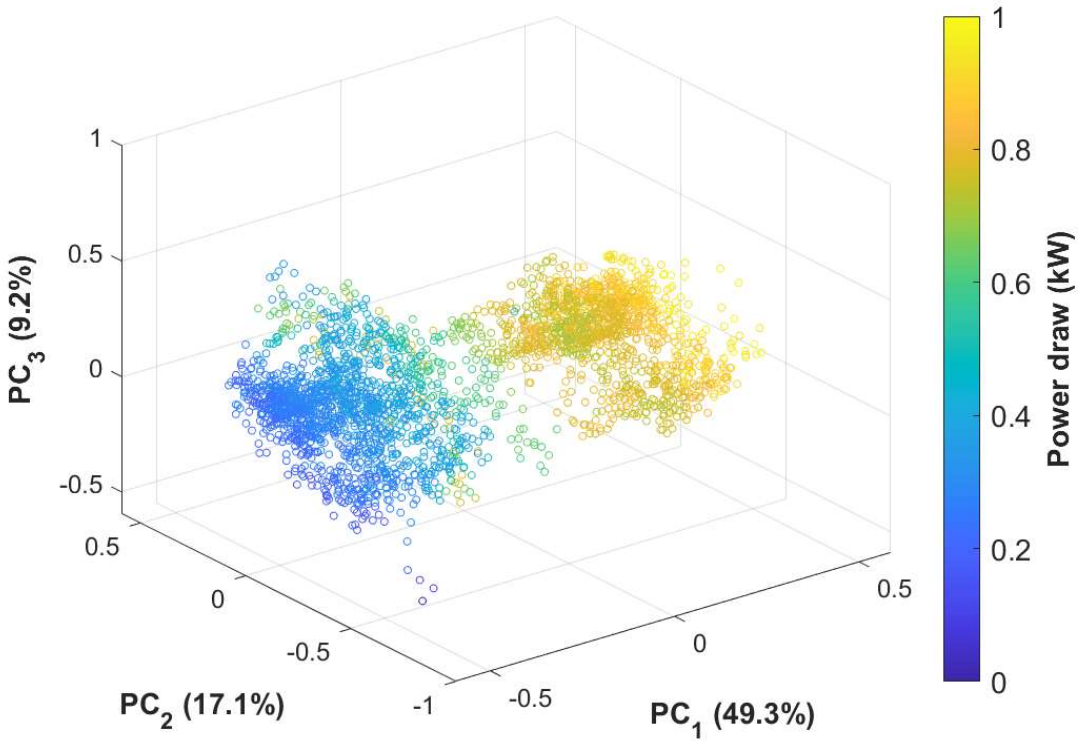

Figure 37: Principal component scores of AG circuit operational variables on the first three principal axes. The percentage of variance captured by each principal component is indicated in brackets. The mill power draw is superimposed with a colour map

Figure 37 shows the existence of two distinct clusters in the data, corresponding to the two operational states observed. As indicated by the superimposed power draw, the two states correspond to the coarser and finer product transfer size indicated by the lower and higher power draw states, respectively. Decision trees could be used to extract operating patterns characterising the circuit in each of these states.

5.2.3 Variable correlation analysis

As an additional exploratory step, Pearson's correlation was calculated for each pair of variables in the data set. The correlations are presented in the correlation matrix in Figure 38.

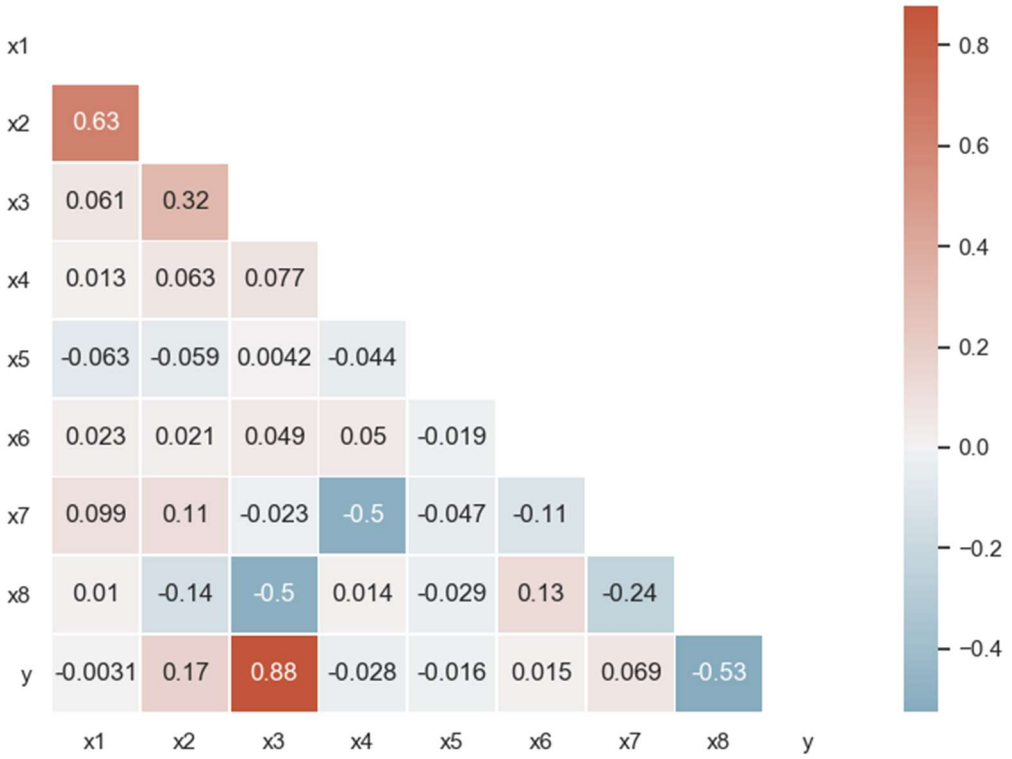

Figure 38: Pearson's correlation between all pairs of operational variables in the AG circuit

Notable correlations in Figure 38 include:

- A strong, positive correlation between the mill load $\mathbf{X_3}$ and power draw $\mathbf{y}$.
- A moderate, positive correlation between the mill dry feed rate $\mathbf{X_1}$ and the water addition rate $\mathbf{X_2}$. This is frequently encountered in grinding circuits, since some control on the mill discharge density is usually applied.
- A moderate, negative correlation between the mill discharge sump level $\mathbf{X_4}$ and the mill discharge sump water addition $\mathbf{X_7}$. This is realistic since the discharge sump water addition is manipulated to control the discharge sump level.
- A moderate, negative correlation between the mill load $\mathbf{X_3}$ and the screen power $\mathbf{X_8}$. An increased load indicates more material being help up in the mill, thus decreasing the solids discharge and lightening the load on the screen.
- A moderate, negative correlation between the screen power $\mathbf{X_8}$ and the mill power draw $\mathbf{y}$. This echoes the correlation between the mill load and screen loading.

5.3 AG circuit model specification

In this case study, a metallurgist could want to understand the operating patterns leading to the different transfer sizes from the AG mill to downstream processes. The higher power draw state in Figure 37 was associated with a finer resultant grind because of the higher specific energy (higher power draw for steady feed rate) and mill load. The lower power draw state was associated with a coarser grind.

To maximise production, metallurgists and process operators need to understand how to operate the circuit to process different feed blends from the available orebody. Different blends will require different operating parameters, such as feed rates, mill densities and recycle stream loadings. Summarising operational patterns occurring with different blends could aid metallurgists to identify potential process improvements during processing. This case study attempts to demonstrate how this could be achieved using decision trees.

The dependence of the product grind size on the mill load is well understood. A higher load at a constant feed rate implies longer residence times in the mill, increasing the probability of grinding to finer sizes. Additionally, the increased load implies more material is available for finer attrition breakage. What was less understood was the effect of the interaction between the load and other variables on the product grind size. It was sought to provide operators with explicit values of the most influential variables to characterising these two states. With this knowledge, operators would have explicit targets for these variables, to maintain the circuit in the state resulting in a finer grind.

There were additional benefits identified to operating in the higher power draw state. The mill load or filling level is a crucial consideration when trying to minimise mill liner wear. Higher load or fill levels can cushion the impacts from cataracting material onto the mill shell, reducing wear of the steel liners. Reducing excessive periods of wear can prolong the life of the mill liners.

As shown in Figure 37, the mill power draw can distinguish between the two states for the period of operation. Thus, the mill power draw was selected as the regression target variable $\mathbf{y}$, and all other circuit variables were used as inputs $\mathbf{X}$. This notation has been used in the previous figures and Table 15. If the dry feed rate had been more variable, the specific energy in kWh/tonne processed would have been more appropriate as a target variable.

The mill power draw is the primary controlled variable in the circuit, and thus rules extracted do not result in an explicit control action at the consequent (leaf) of the rule. However, the decision tree partitions can identify values of the most influential variables to maintain the power draw state, some of which can be manipulated to control the circuit.

5.4 Random forest model of the AG circuit power draw

Like the previous case study, a RF model was trained to serve as an indication of the upper limit for the regression task.

5.4.1 Fitting a RF model to the AG circuit data set

A RF model was fit to the AG circuit data set with input $\mathbf{x}$ and target variables $\mathbf{y}$ as indicated in Table 15. The model parameters are summarised in Table 16. The minimum leaf size is another parameter often imposed, borrowed from heuristics of decision tree growing. A minimum leaf size of five samples is generally recommended for regression trees (Aldrich and Auret, 2013, chap. 5) to stop individual trees from overfitting very noisy data sets. The OOB error of the trained model as a function of the number of trees included, is shown in Figure 39.

Table 16: AG circuit RF model parameters

Parameter	Specification
Number of trees (K)	100
Predictors sampled at each split	$floor(M/3) = 2$
Minimum leaf size	5

Figure 39 shows a fast decay in the OOB error as more trees are added to the forest, before slowly approaching the asymptotic limit of the generalisation performance. No significant improvements to the OOB error was observed beyond 100 trees.

The RF model was retrained with the same parameters, except only 80% of the data was used during training to test the generalisation performance on an independent test set. The test set constituted the remaining 20% of the data set. The predicted response on the test set is plotted against the actual response in Figure 40. In this instance, the coefficient of

determination R^2 was 0.923, implying the model could capture 92.3% of the variance in the response variable using the selected input variables.

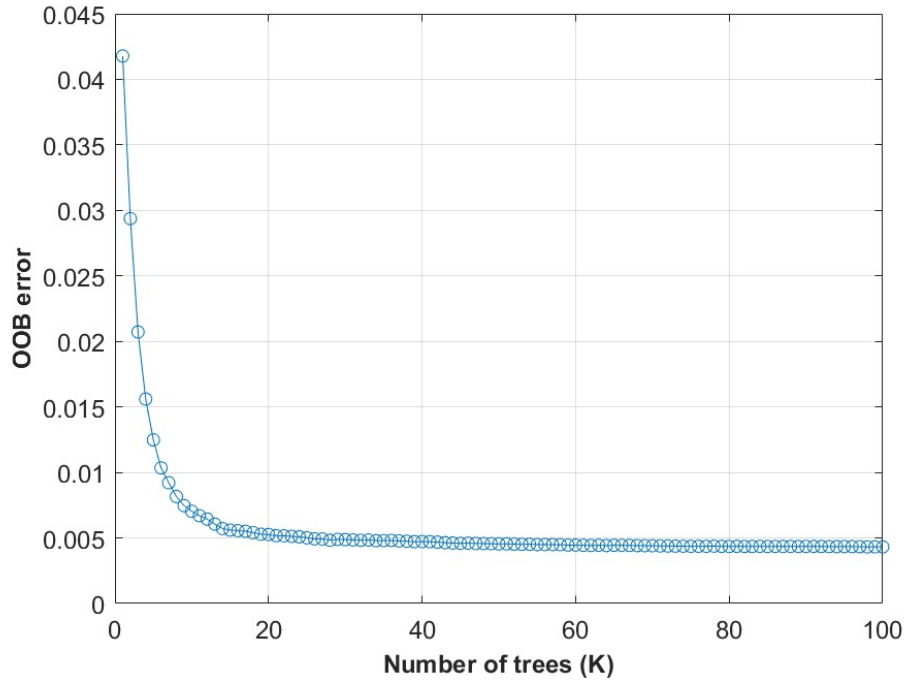

Figure 39: OOB error of the AG circuit RF model as a function of number of decision trees in the forest

For comparison, this procedure with an independent test set was repeated for RF models containing a single tree, up to 100 trees. The independent test set allows the calculation of the R^2- value as a function of the number of trees added, as shown in Figure 41. The results are similar to the OOB error results in Figure 39. Increasing the number of trees in the forest beyond 20 does not significantly increase the R^2-score. Encouragingly, a forest consisting of a single tree achieves an R^2-score over 0.88. However, this tree was induced without any restrictions on branch growth and is likely overfitting the training data set.

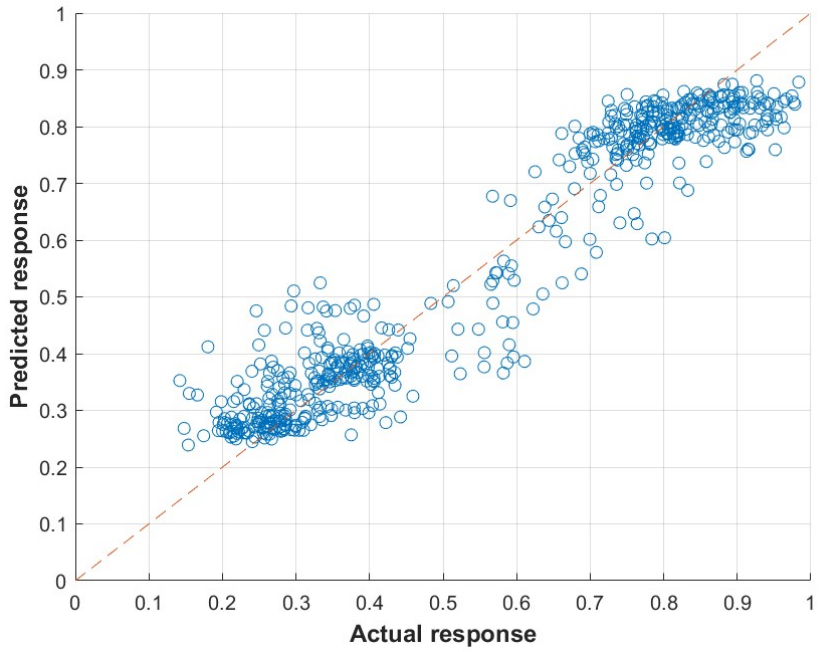

Figure 40: Actual response vs predicted response of a RF model with 100 trees on a held out test set of the AG circuit data

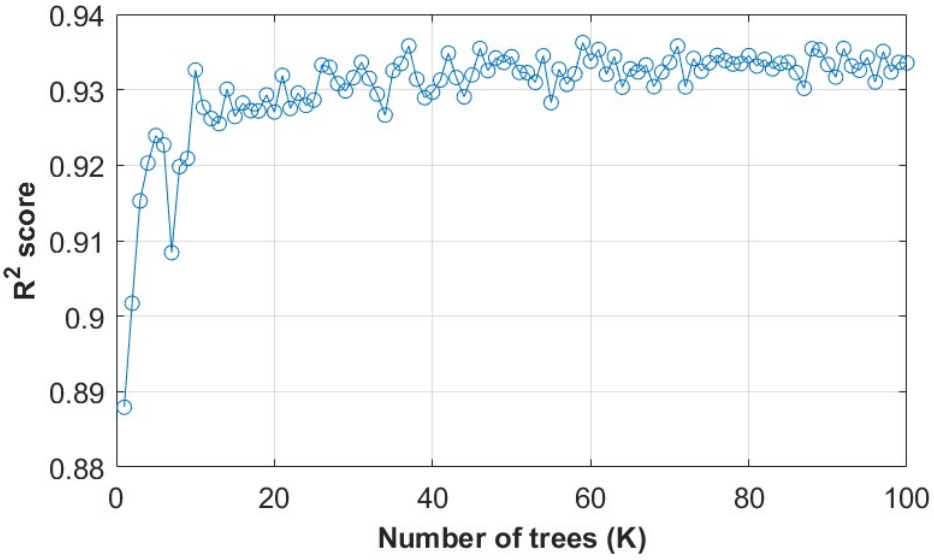

Figure 41: RF model R^2-score on an independent test set as a function of trees in the forest

Based on these results, the AG mill power draw is sufficiently predictable from the input variables to continue with the rule induction procedure. These results will be used to benchmark the result from single decision trees in the following steps in the procedure.

5.4.2 Variable importance measures of the RF model of the AG circuit data

Variable importance measures were calculated to quantify the relative contribution of each variable to the RF model. Both the permutation importance and regression variant of the Gini

variable importance was determined. As discussed in section 3.4, the regression variant calculates the change in mean squared error (MSE) from parent to child nodes. Thus, $\Delta i(s_t, t)$ in equation (14) is replaced with $\Delta MSE(s_t, t)$.

The variable importance measures were calculated for 30 instances of the model. Like in the previous section, an independent test set consisting of 20% of the data set was held out at each iteration to quantify the model generalisation. Again, the test set was randomly sampled at each iteration, thereby generating 30 models trained on different partitions of the data set. Boxplot distributions of the permutation variable importance measures and MSE variable importance are shown in Figure 42 and Figure 43, respectively. As seen in the figures, the average R^2-value on the test set over the 30 model runs was 0.933.

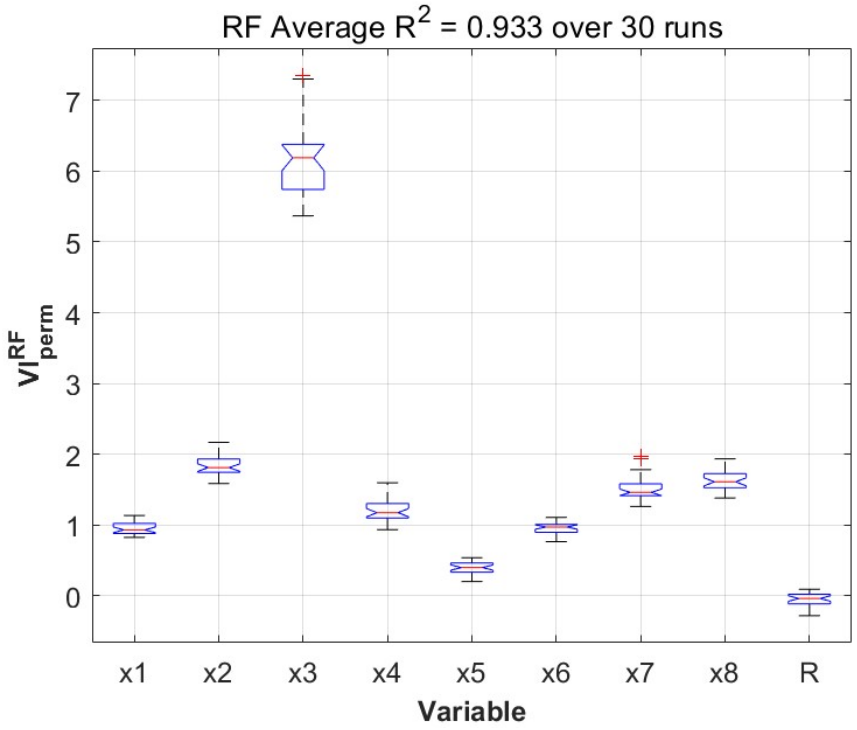

Figure 42: Distribution of permutation variable importance measures of AG circuit RF model with 100 trees. Distributions were calculated over 30 runs of the model

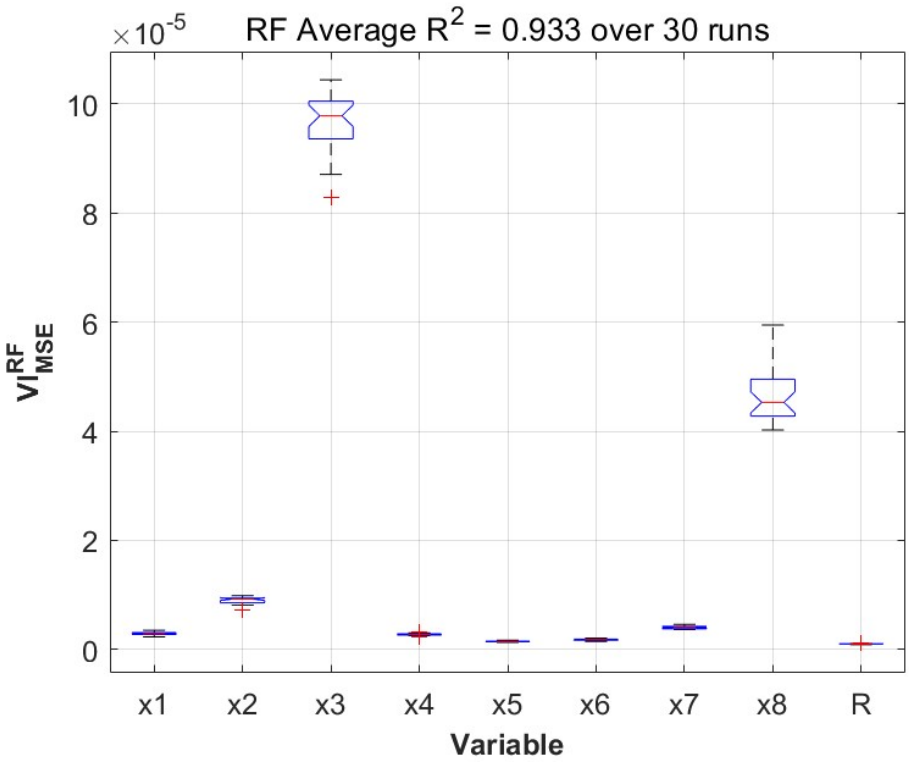

Figure 43: Distribution of MSE variable importance measures of AG circuit RF model with 100 trees. Distributions were calculated over 30 runs of the model

As evidenced by the figures, the RF model considers all input variables at least as important as the random variable. Overall, the two measures are relatively consistent with one another. Both importance measures highlight the large contribution of the mill load $\mathbf{X_3}$, with smaller contributions from the remaining variables. The mill dry feed rate $\mathbf{X_1}$ and variables relating to the discharge sump ($\mathbf{X_4}$-$\mathbf{X_7}$) show relatively minor contributions in both measures. The mill water addition rate $\mathbf{X_2}$ shows a small increase in contribution above the other minor contributors. Contrastingly between the two measures, the MSE importance identifies the classification screen power draw $\mathbf{X_8}$ as a larger contributor. These results are consistent with what was expected from heuristics, as well as the linear correlation analysis earlier.

5.5 Decision tree induction on the AG circuit data

5.5.1 Regression tree induction and simplification

The previous section demonstrated that the mill power draw can be accurately represented using a single decision tree model, with the RF model consisting of a single tree achieving an R^2-score above 0.88. However, these trees were constructed with minimal pruning and would be difficult to interpret for decision support.

To demonstrate this, a single decision tree was induced using the CART algorithm and the parameters listed in Table 17. A restriction on the minimum parent (branch) node size is usually imposed as a default setting in software packages to prevent the tree from growing separate branches for each training example. However, the minimum leaf size of one member still allows the tree to overfit the training data. The tree grown using these settings is shown in Figure 44 below.

Table 17: CART model parameters for decision tree induction on AG circuit data

Parameter	Specification
Minimum parent node size	10
Minimum leaf size	1
Predictors sampled at each split	All

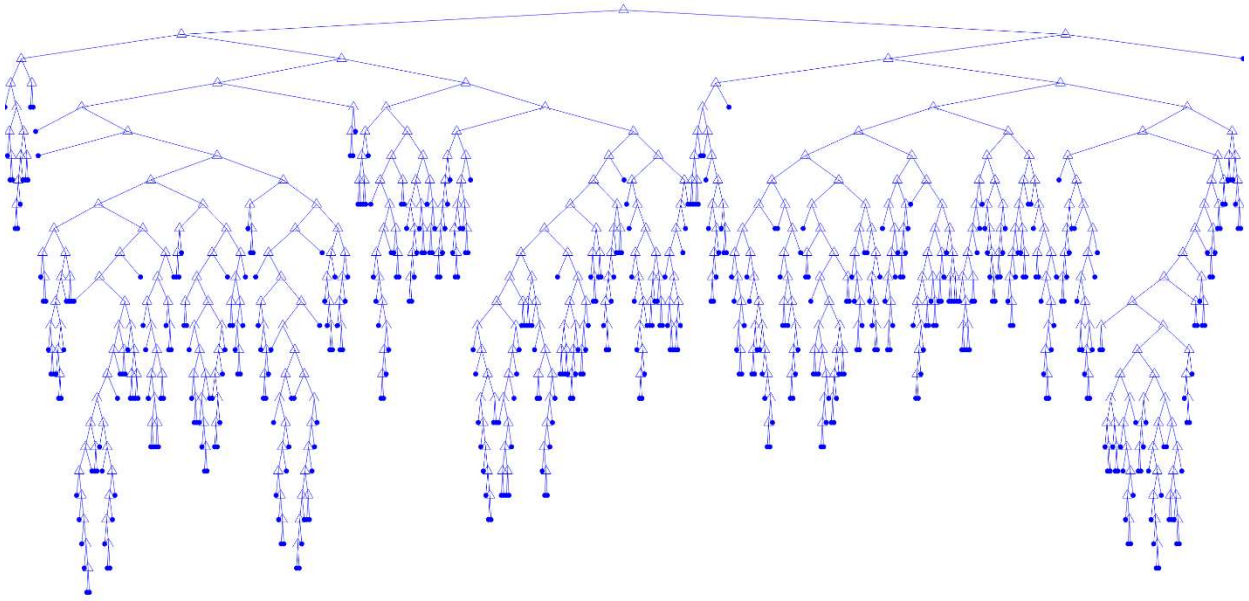

Figure 44: Decision tree predicting the AG mill power draw with minimal restrictions on growth capabilities

The tree in Figure 44 consists of 969 nodes of which 484 are branch nodes and 485 are leaf nodes. If rules were to be extracted from this tree, the result would be a rule set consisting of 485 rules, with up to 25 terms in the rule antecedent (corresponding to the maximum depth observed). Undoubtedly, such a rule set is impractical to provide interpretable decision support. The large number of splits allows the tree to closely fit the training data, but the interpretability of the tree and individual branches is lost.

The tree in Figure 44 achieved an R^2-score of 0.89 on a test data set consisting of 20% of the total data. While the tree is sufficiently accurate, it needs to be investigated how many splits are required to achieve sufficient generalisation performance on the test set. This is examined in Figure 45. Starting with the number of branch nodes in Figure 44, decision trees were trained with progressively less nodes allowed. This was enforced by specifying the maximum number of splits allowed, in addition to the parameters in Table 17. The R^2-value of the tree predictions with the training and test data set was noted for each tree.

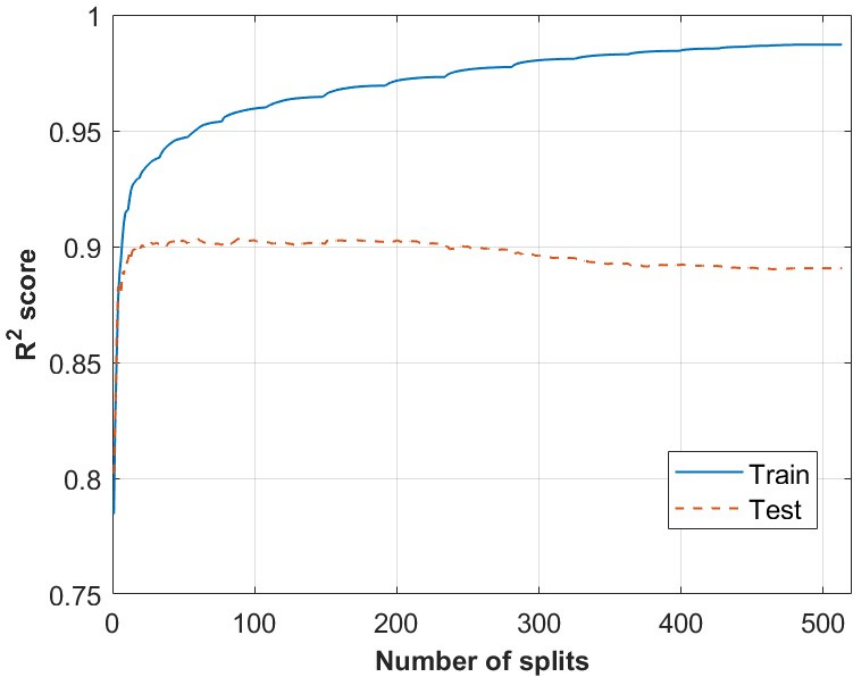

Figure 45: Regression tree accuracy as a function of the maximum number of split nodes allowed

Figure 45 demonstrates that many of the nodes are not contributing to the generalisation performance, but rather overfitting the training data. At first the tree accuracy increases sharply with the addition of more split nodes. However, between 30 – 200 splits this increase flattens and no significant improvement to the generalisation performance is observed. When more than 250 splits are used, the generalisation performance in the tree starts to decrease, indicating that the tree is now overfitting. The figure further indicates that as few as ten splits can be used to generate a tree with a R^2-score of 0.89.

This result again implies that we can significantly reduce the number of splits in construction of the decision trees. The progressive simplification of the tree structure with the enforced limit on the number of splits is demonstrated in Figure 46 .

It is worth noting the discrepancies in generalisation performance between the trees in Figure 46 and Figure 45, as well as the RF models in the previous section. The trees in Figure 46 are sometimes scoring higher R^2-values than even the RF models. However, the trees in Figure 46 have not been cross-validated and only tested on a single training set. This type of instability in the results is a notable drawback of decision trees. Small changes in the training samples can result in entirely different tree structures to emerge (Aldrich and Auret, 2013, chap. 5). It is thus important that decision tree models be cross validated on different partitions of the full data set.

Figure 46 shows that small, interpretable rule sets with high accuracy can be created with a small number of splits allowed in the tree. The tree with a maximum of ten splits results in nine rules with a maximum depth of four, implying four conditions in the rule antecedent. The tree with a maximum of 20 splits contains 17 rules, with a maximum of five terms in the antecedent. While this is naturally subjective, five terms in the antecedent could be approaching the limit for practically supporting operator decisions. Evaluating more terms every time a decision is made could be too confusing or time consuming. On the other hand, a tree with a maximum of five splits would be too simple. The tree would only identify the most critical parameter, the mil load $\mathbf{X_3}$, and would likely not contain any knowledge on the interaction between the most influential variables.

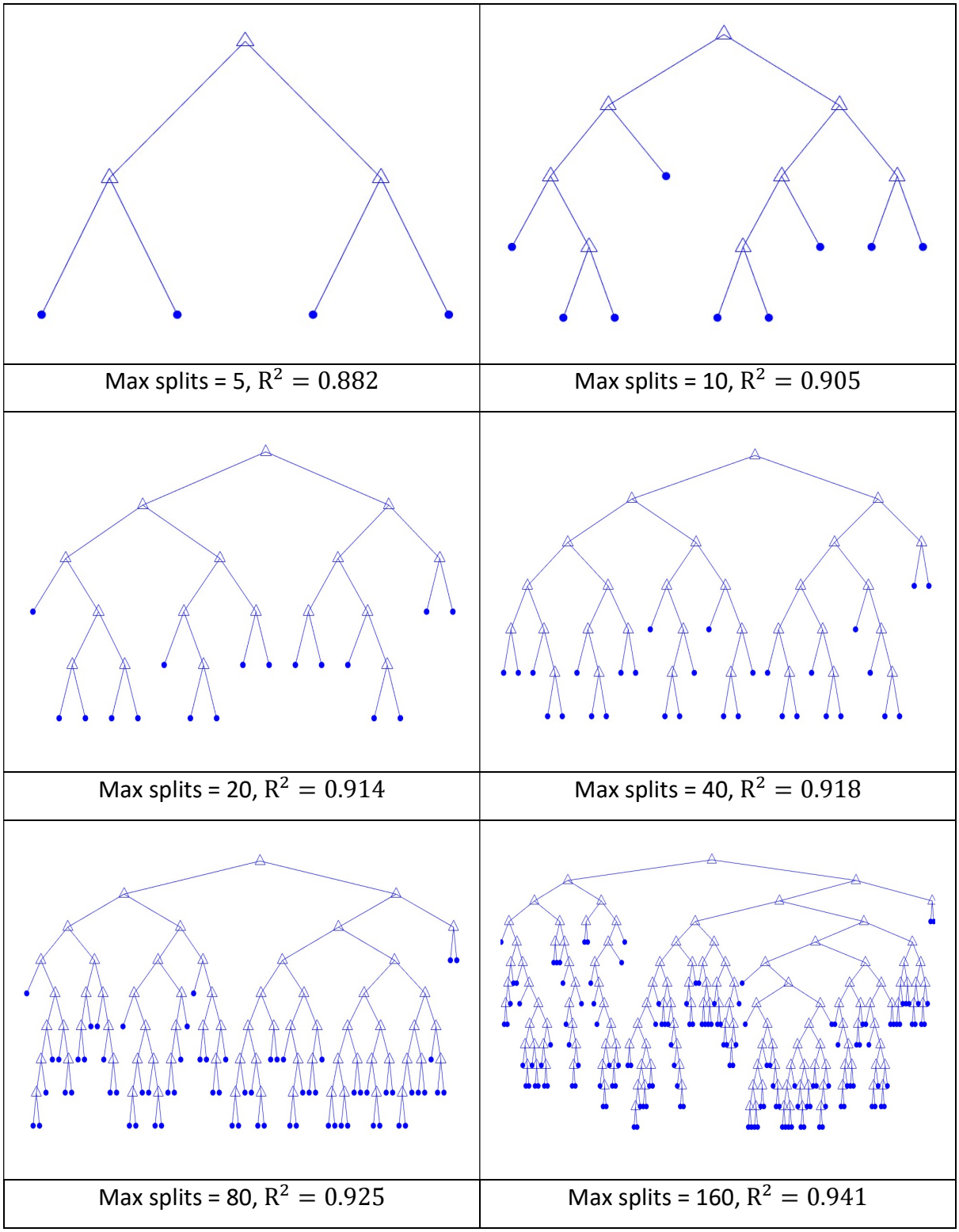

Figure 46: Decision trees on the AG circuit data set with maximum number of splits imposed. Accuracy of each tree on an independent test set is included.

5.5.2 Tree selection and variable importance analysis

The remaining steps in the procedure focus on decision trees with a maximum of ten splits for demonstration. The accuracy and variable contributions of such a tree was determined through 30 instances of the tree, using the parameters indicated in Table 18. The permutation

and MSE variable importance distributions over the 30 runs are depicted in Figure 47 and Figure 48, respectively. As shown in the figure, the trees achieved a cross validated R^2-score of 0.896 or could account for 89.6% of the variance in the response variable from the set of input variables.

Table 18: CART model parameters for decision tree induction on AG circuit data with restrictions on branch growth

Parameter	Specification
Minimum parent node size	10
Minimum leaf size	1
Maximum number of splits	10
Predictors sampled at each split	All

The importance measures in Figure 47 and Figure 48 show markedly similar distributions to that of the RF model in Figure 42 and Figure 43. Both identify the major contribution of the mill load $\mathbf{X_3}$ and a smaller contribution from the screen power draw $\mathbf{X_8}$. Even smaller contributions from the mill water addition $\mathbf{X_2}$ and mill dry feed rate $\mathbf{X_1}$ is identified in both measures. The remaining variables did not score significantly higher contributions than the random variable. The reduced capacity of the decision tree effectively filters out the variables with minimal contribution to the model. With the high average R^2-score and similar variable importance measures, it can be concluded that the tree is sufficiently representative of the larger capacity RF model for the rule extraction procedure to continue.

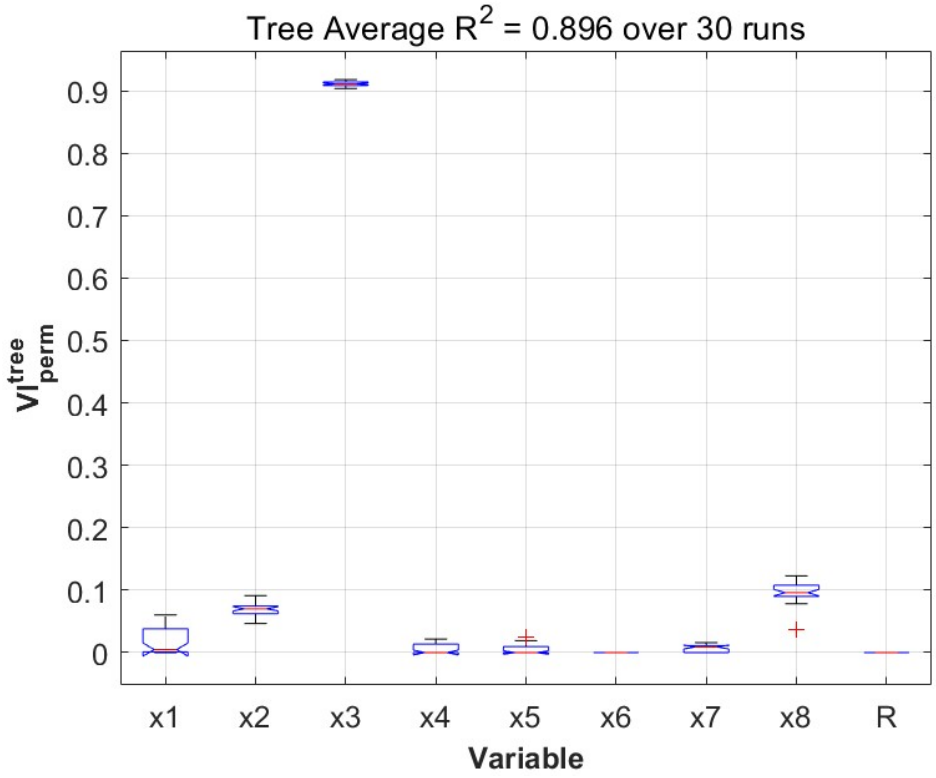

Figure 47: Distribution of permutation variable importance measures of AG circuit decision tree models with a maximum of ten splits. Distributions were calculated over 30 runs of the model.

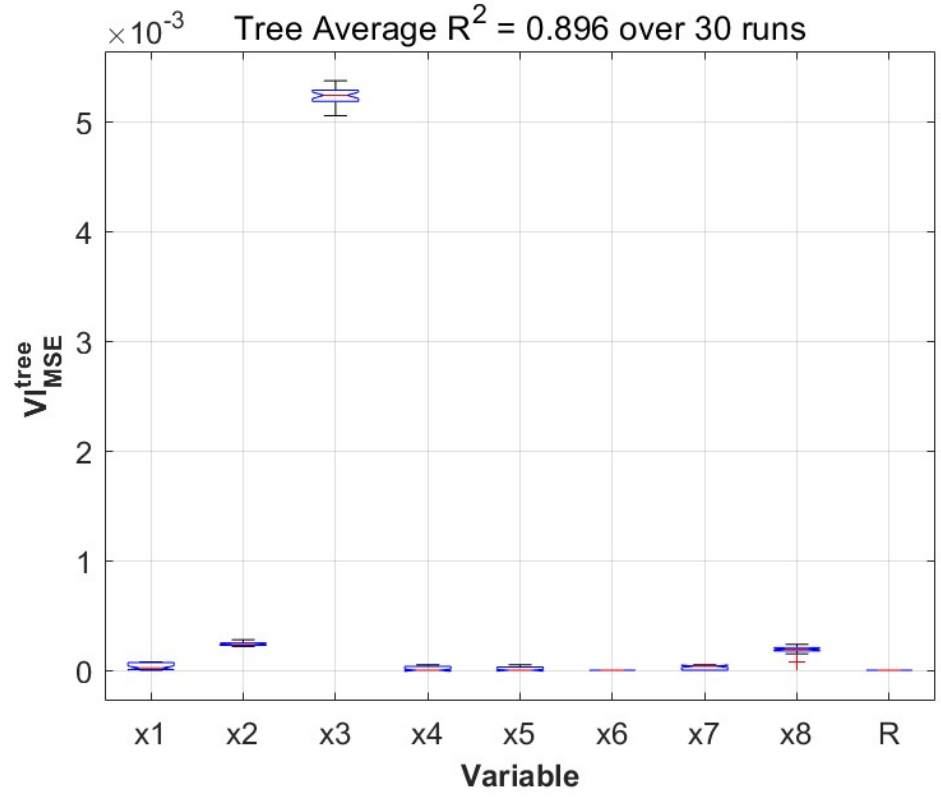

Figure 48: Distribution of MSE importance measures of AG circuit decision tree models with a maximum of ten splits. Distributions were calculated over 30 runs of the model.

The full tree is shown with all split variables and values in Figure 49 below. This tree achieved a R^2- value of 0.902 on an independent test set. The most important variable contributions ($\mathbf{X_3}$, $\mathbf{X_8}$ and $\mathbf{X_2}$) are identified in the top nodes in the tree. Also shown are branches containing splits on variables deemed unimportant by the variable importance analysis. However, during evaluation of individual rules it will be shown that while these rules are expressed in the tree structure, the support for these rules disqualify them for decision support usage.

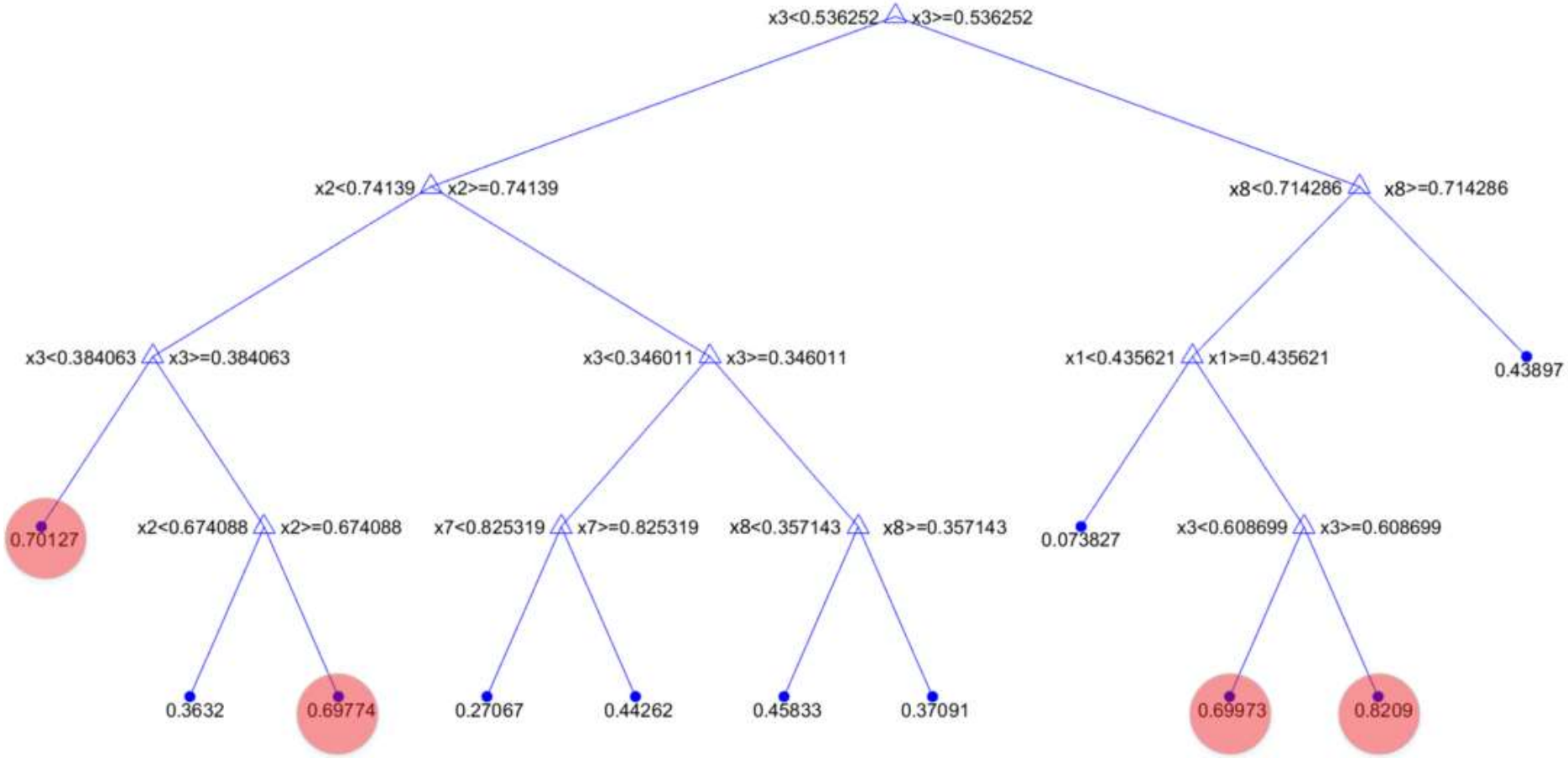

Figure 49: Full decision tree with a maximum of ten splitting nodes. Branch node splitting variables and values are shown, as well as leaf node predictions of the AG circuit power draw. Leaf nodes resulting in predictions for the higher power draw state are circled in red.

5.6 AG circuit decision rule extraction and evaluation

5.6.1 Rule extraction and simplification

In a similar fashion to the previous case study, decision rules are extracted for each leaf node in the tree in Figure 49. Where a variable appears twice in the antecedent of a single rule, the terms were simplified into a single expression for the variable. The rules are presented in Table 19, with the numbering consistent to the leaf nodes from left to right in the tree.

Table 19: Simplified decision rules extracted from the regression tree in Figure 49

Number	Rule	Number	Rule
1	$If\ \mathbf{X_3} < 0.384,$ $and\ \mathbf{X_2} < 0.741;$ $Then\ \mathbf{y} = 0.701$	2	$If\ 0.384 \leq \mathbf{X_3} < 0.536,$ $and\ \mathbf{X_2} < 0.674;$ $Then\ \mathbf{y} = 0.363$
3	$If\ 0.384 \leq \mathbf{X_3} < 0.536,$ $and\ 0.674 \leq \mathbf{X_2} < 0.741;$ $Then\ \mathbf{y} = 0.698$	4	$If\ \mathbf{X_3} < 0.346$ $and\ \mathbf{X_2} \geq 0.741$ $and\ \mathbf{X_7} < 0.825;$ $Then\ \mathbf{y} = 0.271$
5	$If\ \mathbf{X_3} < 0.346$ $and\ \mathbf{X_2} \geq 0.741$ $and\ \mathbf{X_7} \geq 0.825;$ $Then\ \mathbf{y} = 0.443$	6	$If\ 0.346 \leq \mathbf{X_3} < 0.536$ $and\ \mathbf{X_2} \geq 0.741$ $and\ \mathbf{X_8} < 0.357;$ $Then\ \mathbf{y} = 0.458$
7	$If\ 0.346 \leq \mathbf{X_3} < 0.536$ $and\ \mathbf{X_2} \geq 0.741$ $and\ \mathbf{X_8} \geq 0.357;$ $Then\ \mathbf{y} = 0.371$	8	$If\ \mathbf{X_3} \geq 0.536$ $and\ \mathbf{X_8} < 0.714$ $and\ \mathbf{X_1} < 0.436;$ $Then\ \mathbf{y} = 0.074$
9	$If\ 0.536 \leq \mathbf{X_3} < 0.609$ $and\ \mathbf{X_8} < 0.714$ $and\ \mathbf{X_1} \geq 0.436;$ $Then\ \mathbf{y} = 0.700$	10	$If\ \mathbf{X_3} \geq 0.609$ $and\ \mathbf{X_8} < 0.714$ $and\ \mathbf{X_1} \geq 0.436;$ $Then\ \mathbf{y} = 0.821$
11	$If\ \mathbf{X_3} \geq 0.536$ $and\ \mathbf{X_8} \geq 0.714;$ $Then\ \mathbf{y} = 0.439$		

5.6.2 Evaluating the utility of AG circuit decision rules

Each rule in Table 19 was evaluated according to the metrics described in section 3.3. The results are summarised in Table 20 below.

Table 20: Metrics to evaluate the utility of decision rules in Table 19

Number	Supporting samples (% of training data set)	Accuracy (std. deviation)	Complexity (number of splits)
1	2.13	0.095	2
2	0.43	0.101	3
3	0.09	0.125	4
4	23.49	0.053	3
5	1.40	0.149	3
6	5.57	0.134	4
7	20.09	0.073	4
8	0.13	0.065	3
9	3.11	0.071	4
10	42.13	0.068	3
11	1.23	1.69×10^{-16}	2

Table 20 indicates that rules 4, 7 and 11 have high amounts of support in the data set, collectively accounting for approximately 86% of samples in the training data set. The remaining rules received only minor support in the data set. Since the goal of this investigation was to focus on the most common operating patterns leading to the two operational states, these rules were selected for further analysis for decision support.

In addition to high amounts of support, the three rules have relatively high accuracies, with low standard deviations (less than 0.1) implying the spread of target values belonging to the rule is sufficiently low. Rule 6 was another candidate for analysis of less frequently occurring states, with 5.5% support in the data set. However, the accuracy of this rule was deemed too low considering the number of supporting samples.

5.6.3 Analysing rules with high utility

Rules 4, 7, and 10 were identified to hold high utility and is critically analysed to diagnose the operating patterns leading to each of the operational states.

Rule 4 is repeated with its variable descriptions below:

$$\begin{aligned} &If\ \textbf{Mill load} < 0.346, \\ &\quad and\ \textbf{Water addition rate} \geq 0.714, \\ &\quad and\ \textbf{Mill discharge sump water addition} < 0.825; \\ &\quad Then\ \textbf{Mill power draw} = 0.271 \end{aligned} \tag{26}$$

Rule 4 predicts a low power draw and characterises most of the operation in the lower power draw state. The rule identifies that the lower power operating state occurred at mill loads lower than 0.346 and high water addition rates. With a constant dry feed rate, the high water addition rate is flushing finer material from the mill, causing the mill load to drop. The softer ore fed at the time was breaking down faster than usual. Combined with the high water addition, this caused the reduced load resulting in a coarser grind size. This situation occurred throughout 20% of the data set but could possibly be remedied.

A metallurgist analysing the conditions identified in rule 4 might conclude that the AG mill was operating at a sub-optimal density to process this ore blend. Based on the split values extracted, it could be suggested to operators to operate the mill at a higher solids density when processing this ore source again in the future. Alternatively, the mill could benefit from a higher dry feed rate. However, such periods were not captured in the original data set and is not expressed in any of the rules. The split on the water addition is of less utility. The split correctly identifies that periods of high water addition to the AG mill feed required a reduction in the mill discharge sump water addition. The rule is likely picking up the actions of an automatic sump-level control system.

Rule 7 is shown with its variable descriptions below:

$$\begin{aligned} &If\ 0.346 \leq \textbf{Mill load} < 0.536, \\ &\quad and\ \textbf{Mill water addition rate} \geq 0.741, \\ &\quad and\ \textbf{Classification screen current} \geq 0.357; \\ &\quad Then\ \textbf{Mill power draw} = 0.371 \end{aligned} \tag{27}$$

Based on the power draw and load partitions identified, this rule is describing the transient response of the system to the changing ore blend. The power draw is higher than in Rule 4, owed mainly to an increased load resulting from the introduction of higher competency ore. The circuit was found in this state with a classification screen power draw above 0.357 (above low) implying a higher load on the classification screen due to the more competent material increasing the screen oversize and the recirculating load. Again, the water addition was high

and the mill power draw could have been stabilised at a higher level if the water addition was decreased during this transient period to try and retain more material inside the mill. Since the rule is describing the operation during a transient period, the information contained is less useful when attempting to summarise the operational heuristics for a specific blend.

Rule 10 describes most of the operation in the higher power draw state:

$$\begin{aligned} &If\ \textbf{Mill load} > 0.609, \\ &\quad and\ \textbf{Classification screen current} < 0.714, \\ &\quad and\ \textbf{Dry feed rate} > 0.436; \\ &\quad Then\ \textbf{Mill power draw} = 0.821 \end{aligned} \tag{28}$$

Rule 10 captures the operating pattern in the higher power draw state which led to the desired grind size. The rule discovers explicit partition values of the dry feed and mill load required to maintain the power draw at the desired state. The split on the classification screen power draw insists that the screen power remain below a certain value, or else the mill is discharging too much material, and not holding up the charge to maintain the high power draw state and a finer grind. The split values at the partitions could be used to provide operational targets for operators to maintain the circuit in this state. Once the rule detects that the circuit is leaving this state, it can notify the operator, imploring them to diagnose what caused the deviation and take control action.

Rules 4 and 10 manage to capture and compactly summarise the operational conditions pertaining to each of the operational states. The decision tree algorithm identifies the variables most critical to differentiate between the two states and presents their relationship in a decision rule. The two rules could be analysed by metallurgists and used to provide target values of operational variables to process operators. An example of a target sheet, such as those frequently encountered in control rooms, for the two feed blends is given in Table 21.

In the case of the first ore blend the original data set did not contain any periods wherein the mill was run optimally. Consequently, rule 4 cannot provide exact variable targets with which to run the circuit. Ideally more data would be collected to identify periods of good operation using this feed blend. However, given the current data set the metallurgist can use rule 4 as the lower limit for each of these variables and instruct operators to operate above these limits. Based on the rule the metallurgist can diagnose the operation and identify that perhaps the mill load needs to be increased by lowering the water addition rate. These suggestions

are reflected in the values from rule 4 inserted into Table 21. Repeating the analysis with a larger data set could aid in determining the missing values currently present in the table.

Table 21: Operational targets extracted from decision rules for two ore blends

Blend 1 (based on rule 4)		**Blend 2 (based on rule 10)**	
Controlled variable targets		**Controlled variable targets**	
Mill power	>> 0.271	Mill power	0.821
Mill load	>> 0.346	Mill load	> 0.609
Screen power	-	Screen power	< 0.714
Manipulated variable setpoints		**Manipulated variable setpoints**	
Dry feed rate	-	Dry feed rate	> 0.436
Water addition rate	< 0.714	Water addition rate	-
Discharge sump water addition	> 0.825	Discharge sump water addition	-

In contrast, the pattern identified in rule 10 can be used to set operational targets since this operation resulted in a desired transfer particle size. While the rule did not provide explicit targets for all manipulated variables, an operator can utilise their heuristic knowledge in changing these variables to achieve the controlled variable targets. A larger data set, perhaps solely focusing on this period of operation, could be analysed to obtain explicit values for the manipulated variables.

5.7 Summary

In this section, a regression tree was used to discover common operating patterns encountered in two different operational states that occurred in an AG circuit. The resultant rule set was consistent with circuit heuristic knowledge and identified explicit values of variables describing the circuit within the two states. This type of knowledge can guide metallurgists when determining target variable ranges for operators or can aid the operator directly as part of a DSS on an HMI. Using the rules as guidance, operators have explicit targets discovered directly from process data, to maintain the circuit in the desired operating state more often to improve the resultant grind size.

While the case study demonstrated how this might be achieved, the results were limited by the size of the available data set. Ideally, more data would be made available to obtain a representative sample of circuit operation. The I.I.D. assumption imposed on the data set would also be less realistic during such a short and transient state of operation. When analysing similar transient periods, it should be considered to add moving averages of key variables to the list of inputs. This is a relatively simple method to account for the complex dynamics displayed in grinding mills. Alternatively, lagged versions of input variables could be added to the list of model inputs. However, this introduces a rather complicated additional step of determining the appropriate time lags and number of lagged variables to include.

The small period of operation analysed naturally reduces the representativeness of the data set for normal operation. Rules over this short period might not be applicable in future operation. However, the case study demonstrates how this type of analysis could be employed if such data sets were available.

The rule extraction procedure demonstrated could be repeated to expand on the current rule set. More data can be collected and analysed or the number of nodes in the tree can be increased to extract more rules. Alternatively, the problem could have been formulated as a classification task during the model specification step. In this case, the modelling problem would be a binary classification task, predicting to which operational state each sample belongs. However, the regression model offers increased granularity and is more specific as to where the circuit will end up when following a rule.

6. Discussion and conclusions

This section presents some general observations regarding the rule extraction methodology and their application to the case studies, after which the conclusions of the study are summarised.

6.1 Discussion

6.1.1 Rule extraction methodology

The two case studies aimed to demonstrate how decision tree algorithms could be used to extract small sets of simple, yet accurate rules, with high amounts of support in the data set. These rules could be used to aid process operators during decision-making processes in controlling grinding circuits.

In the first case study, classification trees were used to identify operational patterns associated with process operators deciding to bypass the SAG pebble circuit. The classification rules attempted to mimic the operator's decision-making process when certain patterns of events arose. This case study demonstrated the flexibility of the model specification step in the overall procedure, and how models can be formulated to identify and address certain events of interest. The case study also demonstrated how the procedure can be altered to specifically identify rare occurrences in the data set, and how decision rules of such events should be interpreted. The subjective decision-making by operators decreased the overall predictability of the modelling problem. The importance of the proposed utility metrics was emphasised in this case and demonstrated how these are useful for interpreting rules even when inaccuracies are present.

The case study demonstrated how the decision tree model could aid operators in the form of a predictive alert of possible bypass events approaching. Based on the pattern identified from historical data, the model could predict such events upwards of 20 minutes before occurring, implying these could have been avoided had the operator acted at the time the alert could have been sounded. While the alert does not provide a clear instruction to control action, the decision rule could help the operator diagnose the situation and deduce an appropriate response.

In the second case study, a regression tree was used to extract rules exemplifying common operating patterns encountered in two operational states of the grinding circuit. While classification trees have found some implementations in knowledge-based systems, regression trees have received markedly less attention in this area. The advantage of using a regression tree in this situation is that these operational states do not necessarily have to be predefined. With a classification tree, operational data needs to be discretised into distinct classes to discover knowledge regarding the classes. Embedding this type of prior knowledge into the data could inhibit the algorithms to discover certain unknown patterns in the data space. However, the discretisation approach can result in rules that are more interpretable. Olivier *et al.* (2020) investigated this approach by manually fuzzifying the resulting rule set with categorical linguistic terms. The authors found that meaningful rules could be extracted to infer the operational state of a SAG mill at different mill load levels.

The second case study demonstrated how the methodology could be used to extract rules compactly summarising the operation of the circuit in different conditions, and how this could be used to set explicit target values for operators. However, only a small data set was available in this case and the rules may not be representative in future situations. This case study did not result in the sort of automated decision support as demonstrated in the first case study but demonstrated the versatility of the methodology for knowledge discovery in grinding circuits.

A potential weakness of the case studies is the pseudo-steady state assumption introduced when analysing the data sets. This was introduced through the random partitioning of data sets under the I.I.D. assumption, as well as the disregard of the dynamic nature of the time series data. These assumptions are somewhat realistic in the first case study considering the large time span of the dataset and assuming that operators manage the mill relatively similarly. These assumptions are less realistic during the transient period of operation present in the second case study. When considering implementation, this case study should be repeated with a larger dataset for each ore blend individually. However, the case study manages to demonstrate how the methodology could be used to define operational targets when conditions are changing. It was also noted that including moving averages of key variables would provide a relatively simplistic manner to incorporate dynamics responses of such variables.

A big advantage of this type of knowledge discovery is the simplicity of the general methodology followed for rule induction. Plant metallurgists attempt to optimise plant operation daily by identifying inefficiencies or operational problems. Once metallurgists identify such problems they believe are solvable with historical data, they could formulate a decision tree model to answer specific questions regarding the problem. Decision trees and decision rules are easily induced using pre-packed, open-source CART or C4.5 implementations. The greatest inhibitor of extracting rules for successful decision support would be the unavailability of quality data sets, or a lack of site-specific knowledge to interpret and critically evaluate the patterns such rules discover. Neither of these would be of any concern to a plant metallurgist.

The decision rules extracted from decision tree algorithms are naturally comparable with heuristic operating rules in ECS. In the first case, knowledge is discovered through machine induction, while in the latter knowledge is elicited directly from a human expert. Incorporating induced decision rules into the knowledge base of an existing ECS would be a trivial task since logic is represented similarly in both cases. Induced decision rules could supplement the heuristic rule base in existing ECS, especially in cases where experts have trouble explicitly stating their thought processes and conclusions. However, it is unlikely that ML induced decision rules will solely be used to make up the knowledge base of an ECS. Unlike with an expert, there is no guarantee that decision rules contain valid or insightful knowledge, so the expert is still required to critically analyse each rule to ensure the rule set is reasonable. Experts are also required to introduce knowledge that are not necessarily reflected in the process data. These rules will usually pertain to process or equipment constraints and other variables that are not frequently measured.

As was demonstrated, rules induced with decision trees are suitable for use in automated decision support in the control room. However, in such cases it is critical that rule knowledge is communicated with effective HMIs to operators. Operators also need to receive additional training to effectively use the tools for process control. Li *et al.* (2011) identified a general distrust or passive use of automatic technology by process operators. This was owed to a lack of understanding of the tools and lack of training in how to effectively use the tools. Successful implementation of rule-based decision support will need to focus on such factors to better support the operator in human supervisory control tasks.

6.1.2 Possible extensions on the methodology

Throughout the study, a few possible extensions or variants of the general methodology was identified. These might be considered for future work on similar knowledge-based systems.

In this work, pre-pruned trees generated using stoppage criteria were used for rule induction. Pre-pruned trees naturally generate under-fitted tree structures. An alternative approach would be to allow the decision tree to overfit the training data set, where after post-pruning methods such as cost-complexity pruning (Breiman *et al.*, 1984) or reduced error pruning (Quinlan, 1987) are used to remove sub-branches that do not contribute to the generalisation accuracy. The initial, over fitted tree is completely accurate before branches not contributing to the generalisation accuracy are removed. However, it is unclear whether this would lead to significantly different structures at the top of the tree, especially considering the small number of nodes that are kept to maintain interpretability for decision support.

The hyper-rectangular nature of partitions in a decision tree can result in a decision surface with large discontinuities. As demonstrated by Aldrich, Schmitz and Gouws (2000), this can cause unstable decision-making by an automatic system when operating close to these discontinuities. Aldrich, Schmitz and Gouws (2000) proposed a further fuzzification of classification rules to produce a smoother response surface. The weights of the rule set and fuzzy membership functions were optimised using a differential evolution algorithm. The resultant response surface was notably smoother while also allowing for the intermediate calculation of a gradient at the position on the surface. Similar fuzzification of the decision rules extracted in this work could provide benefits to the overall stability of the circuit when used as decision support.

As investigated earlier, genetic algorithms have been widely applied for rule induction. This usually involves explicitly encoding rule structures as genes and allowing the genetic algorithm to optimise the decision variables and values through reproduction, crossover, and mutation operations. For decision support it was proposed that the utility of a rule is dependent on the support, accuracy and complexity of a rule, as stated in equation (11). In an alternative procedure, functions such as equation (11) could serve as a fitness function for genetic algorithms to optimise the rule set. With such a fitness function, the algorithm will not search for the most accurate rule set, but rather the rule set of most utility for operator decision support. Rewarding high support and accuracy, while penalising high complexity,

could result in finding the optimal rule set to support operators. Rules extracted using CART could be used as the initial population of the genetic algorithm. Such a procedure could result in a better rule set for decision support than traditional decision tree algorithms.

6.2 Conclusions

This work investigated the application of rule-based machine learning algorithms for knowledge discovery in industrial grinding circuit data. More specifically, decision rules derived from decision tree algorithms were analysed to determine the utility of such rules to support grinding circuit operators inside the control room.

A critical literature study was conducted investigating the current control landscape in grinding circuits, and broader mineral processing plants. This study revealed that while there are continuous developmental efforts toward automatic APC systems, a large portion of mineral processing plants operate under human supervisory control. The performance of these plants remains highly dependent on operator decision-making processes. Rule-based decision support systems were identified as a suitable hybrid-intelligence method of control; drawing both on the knowledge discovered from data using decision tree algorithms, as well as the operators' heuristic knowledge accumulated through experience.

A procedure to extract small rule sets, containing accurate, yet comprehensible rules with high representation in the data set was presented. It was proposed that the utility of rules be evaluated according to the amount of support, accuracy, and complexity of the rule. The methodology results in a procedure that is not complicated to implement but requires expert knowledge of the specific circuit operation. Thus, plant metallurgists could readily conduct this type of analysis.

The methodology was demonstrated in two case studies on data sets from industrial grinding circuits, each with their own goals for the analysis. It was demonstrated that decision tree algorithms offer an effective approach to extracting intelligible rules that could support grinding circuit operators. The first case study demonstrated that with careful model specification, decision trees could attempt to model operator decision-making processes. The decision tree algorithm induced rules formalising common operator behavioural patterns, which could be analysed and addressed for improvement. It was demonstrated how the decision tree model could provide automatic decision support in the form of predictive alerts.

In both case studies, the decision tree models could identify the most influential variables as reliable as the more complex random forest model.

In the second case study it was shown that decision tree models could predict circuit outputs with accuracies close to that of more complex models, such as a random forest. While the random forest is slightly more accurate, the decision tree is vastly more interpretable, and the knowledge contained is easily converted to a rule set. This case study demonstrated that the methodology could extract the most significant characteristics of periods of operation and compactly summarising these as decision rules.

In conclusion, the work in this thesis demonstrated that decision trees can extract informative and practically useful decision rules from complex dynamic systems. Coupled with effective HMIs, such decision rules could be used to support operators in the control room during human supervisory control tasks. These types of decision support systems could decrease the arbitrariness of operator control actions and improve the overall control capabilities, leading to significant financial benefit on mineral processing plants

References

Abou, S. C. and Dao, T. (2009) Fuzzy logic controller based on association rules mining: Application to mineral processing. *Proceedings of the World Congress on Engineering and Computer Science (WCECS 2009)*. San Francisco, USA, 20-22 October 2009.

Abou, S. C. and Dao, T. M. (2010) Association rules mining approach to mineral processing control. *Engineering Letters*, 18 (2), pp. 1–9.

Ackoff, R. L. (1989) From Data to Wisdom. *Journal of Applied Systems Analysis*, 16 , pp. 3–9.

Agrawal, R., Imielinski, T. and Swami, A. (1993) Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93*, pp. 207–216.

Aldrich, C. (2020) Process variable importance analysis by use of random forests in a shapley regression framework. *Minerals*, 10 (5), pp. 1–17.

Aldrich, C. and Auret, L. (2013) *Unsupervised process monitoring and fault diagnosis with machine learning methods*. London: Springer.

Aldrich, C., Burchell, J. J., De, J. W. and Yzelle, C. (2014) Visualization of the controller states of an autogenous mill from time series data. *Minerals Engineering*, 56 , pp. 1–9.

Aldrich, C., Moolman, D. W., Gouws, F. S. and Schmitz, G. P. J. (1997) Machine learning strategies for control of flotation plants. *Control Engineering Practice*, 5 (2), pp. 263–269.

Aldrich, C., Schmitz, G. P. J. and Gouws, F. S. (2000) Development of fuzzy rule-based systems for industrial flotation plants by use of inductive techniques and genetic algorithms. *Journal of The South African Institute of Mining and Metallurgy*, 100 (2), pp. 129–134.

Åström, K. J., Anton, J. J. and Årzén, K. E. (1986) Expert control. *Automatica*, 22 (3), pp. 277–286.

Auret, L. and Aldrich, C. (2012) Interpretation of nonlinear relationships between process variables by use of random forests. *Minerals Engineering*, 35 , pp. 27–42.

Avalos, S., Kracht, W. and Ortiz, J. M. (2020) Machine Learning and Deep Learning Methods in

Mining Operations: a Data-Driven SAG Mill Energy Consumption Prediction Application. *Mining, Metallurgy and Exploration*, 37, pp. 1197-1212.

Awad, E. M. and Ghaziri, H. M. (2004) *Knowledge Management*. Upper Saddle River, NJ: Pearson Education International.

Bandaru, S., Ng, A. H. C. and Deb, K. (2017) Data mining methods for knowledge discovery in multi-objective optimization: Part A - Survey. *Expert Systems with Applications*, 70 , pp. 139–159.

Bardinas, J., Aldrich, C. and Napier, L. (2018) Predicting the Operating States of Grinding Circuits by Use of Recurrence Texture Analysis of Time Series Data. *Processes*, 6 (2), p. 17.

Bearman, R. A. and Milne, R. W. (1992) Expert systems: Opportunities in the minerals industry. *Minerals Engineering*, 5 (10–12), pp. 1307–1323.

Benyahia, I. and Potvin, J. (1998) Decision support for vehicle dispatching using genetic programming, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans,* 28 (3), pp. 306–314.

Botha, S., le Roux, J. D. and Craig, I. K. (2018) Hybrid non-linear model predictive control of a run-of-mine ore grinding mill circuit. *Minerals Engineering*, 123 (October 2017), pp. 49–62.

Breiman, L. (2001) Random forests. *Machine Learning*, 45 , pp. 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984) *Classification and regression trees*. Monterey, CA: Wadsworth Int. Group.

Carvalho, D. R. and Freitas, A. A. (2004) A hybrid decision tree/genetic algorithm method for data mining. *Information Sciences*, 163 (1–3), pp. 13–35.

Casali, A., Gonzalez, G., Torres, F., Vallebuona, G., Castelli, L. and Gimenez, P. (1998) Particle size distribution soft-sensor for a grinding circuit. *Powder Technology*, 99 (1), pp. 15–21.

Chen, X., Li, S., Zhai, J. and Li, Q. (2009) Expert system based adaptive dynamic matrix control for ball mill grinding circuit. *Expert Systems With Applications*, 36 (1), pp. 716–723.

Chen, X. song, Li, Q. and Fei, S. min (2008) Supervisory expert control for ball mill grinding circuits. *Expert Systems with Applications*, 34 (3), pp. 1877–1885.

Choi, S. W. and Lee, I. B. (2004) Nonlinear dynamic process monitoring based on dynamic kernel PCA. *Chemical Engineering Science*, 59 (24), pp. 5897–5908.

Cios, K., Pedrycz, W. and Swiniarski, R. (1998) *Data mining methods for knowledge discovery*. Boston, MA: Springer.

Clancey, W. J. (1986) Transcript of plenary sessions: Cognition and expertise. in *1st AAAI Workshop Knowledge Acqutiition in Knowledge Based Systems*. Banff, Canada.

Coetzee, L. C., Kerrigan, E. C. and Craig, I. (2010) Nonlinear model predictive control of a run-of-mine ore milling circuit. *IEEE Transaction on Control Systems Technology*, 18 (1), pp. 222–229.

Conradie, A. V. E. and Aldrich, C. (2001) Neurocontrol of a ball mill grinding circuit using evolutionary reinforcement learning. *Minerals Engineering*, 14 (10), pp. 1277–1294.

Cook, S. (2006) The P versus NP problem. in Carlson, J., Jaffe, A., and Wiles, A. (eds) *The Millenium Prize Problems*. Clay Mathematics Institute; American Mathematical Society, pp. 87–107.

Danilkewich, H. and Hunter, I. (2006) HPGR challenges and growth opportunities. in *Fourth International Conference on Autogenous and Semiautogenous Grinding Technology*. Vancouver, British Columbia, Canada, pp. IV-27-IV–44.

Ding, J., Chai, T., Wang, H. and Chen, X. (2012) Knowledge-based global operation of mineral processing under uncertainty. *IEEE Transactions on Industrial Informatics*, 8 (4), pp. 849–859.

Ding, J., Chen, Q., Chai, T., Wang, H. and Su, C. Y. (2009) Data mining based feedback regulation in operation of hematite ore mineral processing plant. in *Proceedings of the American Control Conference*. St. Louis, MO, USA: IEEE, pp. 907–912.

Ding, J., Yang, C. and Chai, T. (2017) Recent Progress on Data-Based Optimization for Mineral Processing Plants. *Engineering*, 3 (2), pp. 183–187.

Eom, S. B., Lee, S. M., Kim, E. B. and Somarajan, C. (1998) A survey of decision support system applications (1988-1994). *Journal of the Operational Research Society*, 49 , pp. 109–120.

Eom, S. and Kim, E. (2006) A survey of decision support system applications (1995-2001). *Journal of the Operational Research Society*, 57 (11), pp. 1264–1278.

Fallah-Mehdipour, E., Bozorg Haddad, O. and Mariño, M. A. (2013) Developing reservoir operational decision rule by genetic programming. *Journal of Hydroinformatics*, 15 (1), pp. 103–119.

Faucher, J. B. P. L., Everett, A. M. and Lawson, R. (2008) Reconstituting knowledge management. *Journal of Knowledge Management*, 12 (3), pp. 3–16.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996) The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39 (11), pp. 27–34.

Fischer, R. A. (1936) The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7 (2), pp. 179–188.

Friedman, J. H. (1991) Multivariate adaptive regression splines. *The Annals of Statistics*, 19 (1), pp. 1–67.

Fuerstenau, D. W. and Abouzeid, A. Z. (2002) The energy efficiency of ball milling in comminution. *International Journal of Mineral Processing*, 67 (1–4), pp. 161–185.

G.V.Kass (1980) An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Applied Statistics*, 29 (2), pp. 119–127.

Giglia, K. C. and Aldrich, C. (2020) Operational state detection in hydrocyclones with convolutional neural networks and transfer learning. *Minerals Engineering*, 149 (January), p. 106211.

Gouws, F. S. and Aldrich, C. (1996) Rule-based characterization of industrial flotation processes with inductive techniques and genetic algorithms. *Industrial and Engineering Chemistry Research*, 35 (11), pp. 4119–4127.

Groenewald, J. W. de V, Coetzer, L. P. and Aldrich, C. (2006) Statistical monitoring of a grinding circuit: An industrial case study. *Minerals Engineering*, 19 (11), pp. 1138–1148.

Grzymala-Busse, J. W. (1992) LERS - A system for learning from examples based on rough sets. in R.Slowinski (ed.) *Intelligent Decision Support: Handbook of Applications and Advances in Rough Sets Theory*. Dordrecht, Boston, London: Kluwer Academic Publishers, pp. 3–18.

Gugel, K. S. and Moon, R. M. (2007) Automated mill control using vibration signal processing. *2007 IEEE Cement Industry Technical Conference Record*, pp. 17–25.

Hadizadeh, M., Farzanegan, A. and Noaparast, M. (2018) A plant-scale validated MATLAB-based fuzzy expert system to control SAG mill circuits. *Journal of Process Control*, 70 , pp. 1–11.

Hancock, T., Jiang, T., Li, M. and Tromp, J. (1995) Lower bounds on learning decision lists and trees. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 900 (0040), pp. 527–538.

Hodouin, D. (2011) Methods for automatic control, observation, and optimization in mineral processing plants. *Journal of Process Control*, 21 (2), pp. 211–225.

Inapakurthi, R. K., Miriyala, S. S. and Mitra, K. (2020) Recurrent neural networks based modelling of industrial grinding operation. *Chemical Engineering Science*, 219 , p. 115585.

Jemwa, G. T. and Aldrich, C. (2006) Kernel-based fault diagnosis on mineral processing plants. *Minerals Engineering*, 19 (11), pp. 1149–1162.

Johannsen, G. and Alty, J. L. (1991) Knowledge engineering for industrial expert systems. *Automatica*, 27 (1), pp. 97–114.

Laudon, K. C. and Laudon, J. P. (1988) *Management Information Systems*. Macmillan.

Leech, W. J. (1986) A rule-based process control method with feedback. *Advances in Instrumentation*, 41 , pp. 169–175.

Li, G., Roufail, R., Klein, B., Nordell, L., Kumar, A., Sun, C. and Kou, J. (2019) Experimental evaluation of the conjugate anvil hammer mill – Comparison of semi-confined to confined particle breakage. *Minerals Engineering*, 137 (October 2018), pp. 34–42.

Li, X., McKee, D. J., Horberry, T. and Powell, M. S. (2011) The control room operator: The forgotten element in mineral process control. *Minerals Engineering*, 24 (8), pp. 894–902.

Liaw, A. and Wiener, M. (2002) Classification and Regression by randomForest. *R News*, 2 (3), pp. 18–22.

Linkens, D. A. and Minyou Chen (1995) Expert control systems-I. Concepts, characteristics and issues. *Engineering Applications of Artificial Intelligence*, 8 (4), pp. 413–421.

Loh, W.-Y. and Shih, Y.-S. (1997) Split selection methods for classification trees. *Statistica*

Sinica, 7 (4), pp. 815–840.

Maimon, O. and Rokach, L. (2005) *Data mining and knowledge discovery handbook*. Tel-Aviv, Israel: Springer.

Massinaei, M. and Doostmohammadi, R. (2010) Modeling of bubble surface area flux in an industrial rougher column using artificial neural network and statistical techniques. *Minerals Engineering*, 23 (2), pp. 83–90.

McCoy, J. T. and Auret, L. (2019) Machine learning applications in minerals processing: A review. *Minerals Engineering*, 132 (December 2018), pp. 95–109.

Metso (2015) *Basics in minerals processing*. Available: https://www.mogroup.com/insights/e-books/basics-in-minerals-processing-handbook/

Michalski, R. S. (1969) On the quasi-minimal solution of the general covering problem. *Proceedings of the V International Symposium on Information Processing (FCIP 69)*. Bled, Yugoslavia, 8-11 October 1969, pp. 125–128.

Minchala-Avila, L. I., Reinoso-Avecillas, M., Sanchez, C., Mora, A., Yungaicela, M. and Mata-Quevedo, J. P. (2016) A comparative study of black-box models for cement quality prediction using input-output measurements of a closed circuit grinding. *2016 Annual IEEE Systems Conference (SysCon 2016)*. Orlando, Florida, USA, 18-21 April 2016, pp. 1-7.

Miriyala, S. S. and Mitra, K. (2020) Deep learning based system identification of industrial integrated grinding circuits. *Powder Technology*, 360 , pp. 921–936.

Muller, B. and De Vaal, P. L. (2000) Development of a model predictive controller for a milling circuit. *Journal of The South African Institute of Mining and Metallurgy*, 100 (7), pp. 449–453.

Nakhaei, F., Mosavi, M. R., Sam, A. and Vaghei, Y. (2012) Recovery and grade accurate prediction of pilot plant flotation column concentrate: Neural network and statistical techniques. *International Journal of Mineral Processing*, 110–111 , pp. 140–154.

Napier-Munn, T. (2015) Is progress in energy-efficient comminution doomed? *Minerals Engineering*, 73 , pp. 1–6.

Napier-Munn, T. J., Morrell, S., Robert, D. and Kojovic, T. (1996) *Mineral comminution circuits: their operation and optimisation*. Indooroopilly, Qld Australia: Julius Kruttschnitt Mineral

Research Centre, University of Queensland.

Naumov, G. E. (1991) NP-completeness of problems of construction of optimal decision trees. *Soviet Physics Doklady*, 36 (4), pp. 270–271.

Nierop, M. A. Van and Moys, M. H. (1997) Measurement of load behaviour in an industrial grinding mill. *Control Engineering Practive*, 5(2), pp. 257–262.

Nonaka, I. and Takeuchi, H. (1995) *The knowledge-creating company: How Japanese companies create the dynamics of innovation*. Oxford University Press.

Nordell, L. and Potapov, A. (2015) Novel comminution machine may vastly improve crushing-grinding efficiency. *6th International Autogenous Grinding, Semi-autogenous Grinding and High Pressure Grinding Roll Technology Conference (SAG 2015)*, Vancouver, British Columbia, Canada, 20-23 September 2015, pp. 1–16.

Olivier, J. and Aldrich, C. (2020) Dynamic monitoring of grinding circuits by use of global recurrence plots and convolutional neural networks. *Minerals*, 10 (11), pp. 1–28.

Olivier, J., Aldrich, C., Shelley, P. and Davies, E. (2020) Decision support systems for SAG mill control: A case study. in Velasquez, C., Cisternas, L., Gutierrez, L., Jerez, O., Johnston, A., Kracht, W., and Kuyvenhoven, R. (eds) *2020 Procemin GEOMET*. Santiago, Chile: Gecamin Digital Publications, pp. 290–298.

Olivier, L. E. and Craig, I. K. (2017) A survey on the degree of automation in the mineral processing industry. in *2017 IEEE AFRICON: Science, Technology and Innovation for Africa,*. Cape Town, South Africa: IEEE, pp. 404–409.

Olivier, L. E., Maritz, M. G. and Craig, I. K. (2019) Deep Convolutional Neural Network for Mill Feed Size Characterization. *IFAC-PapersOnLine*, 52 (14), pp. 105–110.

De Oña, J., López, G. and Abellán, J. (2013) Extracting decision rules from police accident reports through decision trees. *Accident Analysis and Prevention*, 50 , pp. 1151–1160.

Pawlak, Z. (1984) Rough classification. *International Journal of Man-Machine Studies*, 20 (5), pp. 469–483.

Pax, R. A. and Cornish, B. (2015) Determination of particle trajectories, toe and shoulder dynamics using a non-contact acoustic array on an industrial SAG mill. *6th International*

Autogenous Grinding, Semi-autogenous Grinding and High Pressure Grinding Roll Technology Conference (SAG 2015). Vancouver, British Columbia, Canada, 20-23 September 2015, pp. 1–15.

Powell, M. and Mainza, A. (2006) Extended grinding curves are essential to the comparison of milling performance. *Minerals Engineering*, 19 (15), pp. 1487–1494.

Powell, M. S., van der Westhuizen, A. P. and Mainza, A. N. (2009) Applying grindcurves to mill operation and optimisation. *Minerals Engineering*, 22 (7–8), pp. 625–632.

Putland, B., Kock, F. and Siddall, L. (2011) Single stage AG/SAG milling design. *Fifth International Autogenous Grinding, Semi-autogenous Grinding and High Pressure Grinding Roll Technology Conference (SAG 2011)*, Vancouver, British Columbiam Canada, 24-28 September 2011, pp. 1–11.

Quinlan, J. R. (1986) Induction of decision trees. *Machine Learning*, 1 (1), pp. 81–106.

Quinlan, J. R. (1987) Simplifying decision trees. *International Journal of Man-Machine Studies*, 27 (3), pp. 221–234.

Quinlan, R. (1993) *C4.5: Programs for machine learning*. Los Altos: Morgan Kaufmann Publishers Inc.

Ramasamy, M., Narayanan, S. S. and Rao, C. D. P. (2005) Control of ball mill grinding circuit using model predictive control scheme. *Journal of Process Control*, 15 (3), pp. 273–283.

Reuter, M. A., Moolman, D. W., Van Zyl, F. and Rennie, M. S. (1998) Generic Metallurgical Quality Control Methodology for Furnaces on the Basis of Thermodynamics and Dynamic System Identification Techniques. *IFAC Proceedings Volumes*, 31 (23), pp. 363–368.

Rivest, R. L. (1987) Learning decision lists. *Machine Learning*, 2 , pp. 229–246.

Rowley, J. (2007) The wisdom hierarchy: Representations of the DIKW hierarchy. *Journal of Information Science*, 33 (2), pp. 163–180.

Salazar, J. L., Valdés-González, H., Vyhmesiter, E. and Cubillos, F. (2014) Model predictive control of semiautogenous mills (sag). *Minerals Engineering*, 64 , pp. 92–96.

Saraiva, P. M. and Stephanopoulos, G. (1992) Continuous process improvement through

inductive and analogical learning. *AIChE Journal*, 38 (2), pp. 161–183.

Schmitz, G. P. J., Aldrich, C. and Gouws, F. S. (1999) ANN-DT: An algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks*, 10 (6), pp. 1392–1401.

Schönert, K. (1988) A first survey of grinding with high-compression roller mills. *International Journal of Mineral Processing*, 22 (1–4), pp. 401–412.

Schultz, A. C. and Grefenstette, J. J. (1990) Improving tactical plans with genetic algorithms. *Proceedings of the 2nd International IEEE Conference on Tools for Artificial Intelligence*, Herndon, VA, USA, 6-9 November 1990, pp. 328–334.

Sloan, R., Parker, S., Craven, J. and Schaffer, M. (2006) Expert systems on SAG circuits: Three comparative case studies. *Fourth International Conference on Autogenous and Semiautogenous Grinding Technology (SAG 2006)*, Vancouver, British Columbia, Canada, 23-27 September 2006, pp. II346–II357.

Stange, W. (1993) Using artificial neural networks for the control of grinding circuits. *Minerals Engineering*, 6 (5), pp. 479–489.

Tjen-Sien, L., Wei-Yin, L. and Shih, Y.-S. (1992) A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms. *Machine Learning*, 229 , pp. 203–229.

Tseng, M. H., Chen, S. J., Hwang, G. H. and Shen, M. Y. (2008) A genetic algorithm rule-based approach for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63 (2), pp. 202–212.

Wei, D. and Craig, I. (2008) Grinding mill circuits - A survey of control and economic concerns. *IFAC Proceedings Volumes*, 41(2), pp. 1000–1005.

Wei, D. and Craig, I. K. (2009) Grinding mill circuits - A survey of control and economic concerns. *International Journal of Mineral Processing*, 90 (1–4), pp. 56–66.

Weiss, S. M. and Indurkhya, N. (1993) Optimized Rule Induction. *IEEE Expert*, 8 (6), pp. 61–69.

Wills, B. A. and Napier-Munn, T. (2006) *Wills' Mineral Processing Technology*. Seventh ed. *Wills' Mineral Processing Technology: An introduction to the practical aspects of ore*

treatment and mineral recovery. Seventh ed. Elsevier; Butterworth-Heinemann.

Xiao, Z. (2001) *Developing simple regressions for predicting gold gravity recovery in grinding circuit*. McGill University.

Zadeh, L. A. (1979) *Fuzzy sets and information granularity*. *Advances in fuzzy set theory and applications*. Edited by M. M. Gupta.

Zhao, L., Tang, J., Yu, W., Yue, H. and Chai, T. (2010) Modelling of mill load for wet ball mill via GA and SVM based on spectral feature. in *IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*. Changsha, China: IEEE.

Zhou, P., Chai, T. and Sun, J. (2013) Intelligence-based supervisory control for optimal operation of a DCS-controlled grinding system. *IEEE Transactions on Control Systems Technology*, 21 (1), pp. 162–175.

Zhou, P., Lu, S., Yuan, M. and Chai, T. (2016) Survey on higher-level advanced control for grinding circuits operation. *Powder Technology*, 288 , pp. 324–338.

Appendix A

Refereed Journal Paper

Olivier, J. and Aldrich, C. (2021) Use of decision trees for the development of decision support systems for the control of grinding circuits. *Minerals* 11 (6), 595.

	Conception and design (%)	Acquisition of Data and Method (%)	Data Conditioning and Manipulation (%)	Analysis and Statistical Method (%)	Interpretation and Discussion (%)	Final Approval (%)	Total Contribution (%)
Jacques Olivier	60	70	100	100	70	40	60
I acknowledge that these represent my contribution to the above research output Signed:							
Chris Aldrich	40	30	-	-	30	60	40
I acknowledge that these represent my contribution to the above research output Signed:							
Total %	100	100	100	100	100	100	

Appendix B

Refereed Conference Paper

Olivier, J., Aldrich, C., Shelley, P. and Davies, E. (2020) Decision support systems for SAG mill control: A case study. in Velasquez, C., Cisternas, L., Gutierrez, L., Jerez, O., Johnston, A., Kracht, W., and Kuyvenhoven, R. (eds) *2020 Procemin GEOMET*. Santiago, Chile: Gecamin Digital Publications, pp. 290–298.

	Conception and design (%)	Acquisition of Data and Method (%)	Data Conditioning and Manipulation (%)	Analysis and Statistical Method (%)	Interpretation and Discussion (%)	Final Approval (%)	Total Contribution (%)
Jacques Olivier	60	50	100	100	70	40	60
I acknowledge that these represent my contribution to the above research output Signed:							
Chris Aldrich	40	30	-	-	10	20	20
I acknowledge that these represent my contribution to the above research output Signed:							
Paul Shelley	-	10	-	-	10	20	10
I acknowledge that these represent my contribution to the above research output Signed:							
Eugene Davies	-	10	-	-	10	20	10
I acknowledge that these represent my contribution to the above research output Signed:							
Total %	100	100	100	100	100	100	