University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: **Computer Science/ Reliable and secure systems**	Spring semester, **2021** Open access
Writer: **Maryam Farzad**	[illegible] (Writer's signature)

Faculty supervisor:
Tomasz Wiktorski

External supervisor(s):
Ekaterina Wiktorski

Thesis title:

The application of big data analysis and machine learning for kick detection and kick prediction during drilling operations

Credits (ECTS): **30**

Key words: **Time-series forecasting, Big data analysis, Machine learning, well control issues, kick and loss circulation, OpenLab simulator, Neural Network**	Pages: **98** Stavanger, **June 2021**

University of Stavanger

UNIVERSITY OF STAVANGER

The application of big data analysis and machine learning for kick detection and kick prediction during drilling operations

---

Author:
Maryam Farzad

Department:
Science and Technology

Major:
Computer Science

Supervisors:
Tomasz Wiktorski
Ekaterina Wiktorski

June 7, 2021

Abstract

Drilling operations for oil and gas extraction is a complex and risky process. Workers are not able to start the drilling operations unless they carefully accomplish some pre-drilling activities such as choosing a proper site wisely and arranging the rig equipment. But no matter how much caution the drilling setup is done, hazards and unplanned events are likely to happen anyway. ***"kick"*** is a down-hole phenomenon that occurs when an uncontrolled amount of formation fluid flows into the wellbore when formation pressure is more than hydrostatic and fractional pressure in the wellbore. There are several procedures when experiencing a kick to minimize the danger. However, small kicks are probable to turn into catastrophic blowouts due to improper handling.

Managing drilling hazards is a globally crucial task in the oil and gas industry to prevent incurable consequences such as fatal accidents, devastating injuries, loss and huge damages to the rig assets, and destructive influence on the environment. Due to the complexity of downhole conditions, finding the right and perfect way to reach the desired depth fast is extremely challenging. Drilling operations are directly accompanied by dealing with a huge volume of data. Therefore, it has become a major concern for many oil and gas companies to manage the datasets obtained from rigs to gain valuable operational insight. Data analysis in this area assists in effective decision-making in different activities while balancing the operational complexities with the associated risks.

Performance enhancement is the main wish in the oil and gas industry. However, traditional data preparation and analysis methods are not sufficiently capable of rapid information extraction and clear visualization of big complicated datasets. Data mining can make a big difference in operational performance by exploring an existing massive dataset to uncover unknown trends and identify relationships and anomalies in the dataset. Then, with machine learning, computers are able to learn the patterns by developing fast and efficient algorithms to make predictions about new datasets.

The main focus of this thesis is on mitigating hazardous events in drilling operations by enabling the identification and prediction of kicks ahead of time. In this study, firstly, the raw synthetic data is generated with the OpenLab drilling simulator created by NORCE. During drilling a well a large number of parameters are being monitored and saved which are processed through data cleaning, feature scaling, outlier detection, data labeling, and dataset splitting. Feature engineering and feature selection are the next vital steps in the dataset preparation phase to decide on which feature combinations are suitable to be applied. The data samples are proportionally split into train set and test set to be measured by some evaluation metrics for accuracy, precision, recall, and F1-score estimation. Then, Machine Learning models are created and developed to train the data. Finally, the models are going to be evaluated to optimize the whole process of kick identification and prediction. The proposed model can potentially be improved in terms of accuracy and efficiency.

Acknowledgements

Throughout the writing of this dissertation, I have received a great deal of support and assistance.

I would first like to express my sincere gratitude and appreciation to my supervisors, Tomasz Wiktorski and Ekaterina Wiktorski, for their valuable advises and feedback throughout my work on this thesis. They pushed me to sharpen my thinking and brought my work to a higher level with their insightful comments and opinions. I would also like to thank Professor Dan Sui for introducing this topic.

I am particularly grateful for the assistance given by Nils Braun, the designer and developer of TSFRESH Python library who answered all my questions with patience and guided me through the process of working with this Python package. I could not have completed this dissertation without the tremendous support he gave me.

My special thanks are extended to the technical and support staff in Computer Science and Petroleum departments at University of Stavanger and I would particularly like to single out the atOpenLab simulator mentors who provided me with the required access and guidance during my work.

I would also wish to acknowledge the unfailing support and great love of my family, friends and especially my husband, Behfar, who gave me continuous encouragement throughout my years of study and through the process of researching and writing this thesis.

Contents

List of Abbreviations **viii**

1 Introduction **1**
1.1 Overview 1
1.2 Motivation and Objective 2
1.3 Limitations 3
1.4 Methodology 3

2 Background **5**
2.1 Drilling Operation 5
2.2 Potential Risks in Drilling 9
2.3 Abnormal trends in drilling 13

3 State Of the Art **20**
3.1 Artificial Intelligence in Drilling 20
3.1.1 Data Mining 21
3.1.2 Machine Learning 22
3.1.3 Neural network 23
3.2 CRISP-DM 24
3.3 Literature Review 27

4 Big Data Analytics **31**
4.1 Big Data 31
4.1.1 Benefits 32
4.1.2 Challenges 33
4.2 Time-series Analysis and Forecasting 34

5 Data collection and preparation **37**
5.1 OpenLab Simulator 37
5.1.1 Openlab tabs 38
5.1.1.1 Hole Section 38

5.1.1.2 Wellpath 39
5.1.1.3 Fluid (Mud) 40
5.1.1.4 Drillstring 40
5.1.1.5 Geology 41
5.1.1.6 Rig 42
5.1.2 Strength 42
5.1.3 Weakness 43
5.2 Understanding the problem 43
5.3 OpenLab Data 45
5.3.1 Data Collection 46
5.3.2 Data Preparation 47
5.3.2.1 Data Cleansing 48
5.3.2.2 Data Transformation 50
5.3.3 Feature Engineering and Feature Selection 52
5.3.3.1 Manual vs. Automated 53
5.3.3.2 TSFRESH package 54
5.3.3.3 Rolling dataframe for time-series forecasting 57

6 Machine learning models selection 61
6.1 Steps to choose the right ML model 62
6.2 How to evaluate the chosen ML model 66

7 Models implementation and evaluation 70
7.1 Tools 70
7.1.1 Hardware 71
7.1.2 Sensors 71
7.1.3 Programming language 71
7.2 Data pre-processing 74
7.2.1 Kick detection 76
7.2.2 Kick prediction 77
7.3 Window features (lag) 77
7.4 Implementation 78
7.4.1 Detection 79
7.4.1.1 DecisionTreeClassifier 79
7.4.1.2 RandomForestClassifier 80
7.4.1.3 KNeighborsClassifier 81
7.4.1.4 Evaluation for detective models 82
7.4.2 Prediction 84
7.4.2.1 DecisionTreeClassifier-Prediction 85

7.4.2.2 RandomForestClassifier-Prediction 86
7.4.2.3 KNeighborsClassifier-Prediction 86
7.4.2.4 Evaluation for predictive models 87
7.4.3 Improve the predictive ML model 89
7.4.3.1 Detection 89
7.4.3.2 Prediction 90
7.4.4 Long short-term memory (LSTM) 92

8 Conclusion and future work **96**

List of Figures **100**

List of Tables **104**

List of Snippets **105**

Bibliography **106**

List of Abbreviations

AI	**A**rtificial **I**ntelligence
ANN	**A**rtificial **N**eural **N**etwork
API	**A**pplication **P**rogramming **I**nterface
ARIMA	**A**uto **R**egression **I**ntegrated **M**oving **A**verage
AUC	**A**rea **U**nder **C**urve
BH	**B**enjamin and **H**ochberg
BOP	**B**low **O**ut **P**reventor
BP	**B**ritish **P**etroleum
CBHP	**C**onstant **B**ottom **H**ole **P**ressure
ECD	**E**quivalent **C**irculation **D**ensity
ESD	**E**quivalent **S**tatic **D**ensity
FDR	**F**alse **D**iscovery **R**ate
HPHI	**H**igh **P**ressure **H**igh **T**emprature
HSE	**H**ealth **S**afety **E**nvironment
KNN	**K** **N**earest **N**eighbor
LSTM	**L**ong **S**hort **T**erm **M**emory
LWD	**L**ogging **W**hile **D**rilling
MD	**M**easured **D**epth
MGS	**M**ud **G**as **S**eperator
ML	**M**achine **L**earning
MPD	**M**anage **P**ressure **D**rilling

MSE	**M**ean **S**quared **E**rror
MWD	**M**easured **W**hile **D**rilling
NN	**N**eural **N**etwork
PWD	**P**ressure **W**hile **D**rilling
RF	**R**andom **F**orest
RMSE	**R**oot **M**ean **S**quared **E**rror
ROC	**R**eceiver **O**perator **C**haracteristic
ROP	**R**ate **O**f **P**enetration
RPM	**R**otation **P**er **M**inute
SMO	**S**equential **M**inimal **O**ptimization
SPP	**S**tand **P**ipe **P**ressure
SVM	**S**upport **V**ector **M**achine
TD	**T**otal **D**epth
TVD	**T**rue **V**ertical **D**epth
WOB	**W**eight **O**n **B**it

Dedicated to my younger brother, Alireza, who I have passed all the ups and downs of my life with and I couldn't imagine my life without him, and to my spouse, Behfar, for giving me strength to reach for the stars and chase my dreams.

Chapter 1

Introduction

1.1 Overview

In this data-driven era, understanding what to do and how to use the vast amount of data brings promising solutions to help experts in decision-making process and desired outcomes. Big data is a field dedicated to advanced analytic techniques to deal with complex and extensively large datasets. Big data methods and techniques with the capability of managing data familiarize users with unknown patterns, correlations, and deep insights.[1]

The oil and gas industry heavily relies on big data analysis. With the use of big data, oil companies can improve productions while optimizing costs. Oil well drilling operation is inherently associated with unstructured raw data. The better the oil companies interpret various types of data and unearth unknown trends, the more they become profitable and efficient. Extracting, investigating, and examining hidden patterns is considered a boon in regard to risk management in this field. Drilling operation hazards would be lied secretly behind these hidden patterns.[2]

During the deep-water drilling process, the subsurface challenging environment poses a significant threat in every drilling and well activity and a lot of work has been devoted to increase safety. Well control has become the main concern among oil and gas companies to protect lives and also prevent loss of natural resources. One of the most common risks is well kick which typically occurs when the formation pressure exceeds the wellbore pressure. In this situation, an intrusion of unwanted fluids runs into the wellbore. Being able to forecast kick incidents in advance will definitely diminish the probability of fatal blowout occurrence like the Deepwater Horizon oil spill which happened in the Gulf of Mexico in 2010.

1.2 Motivation and Objective

As the world is moving toward digitalization, the oil and gas industry does not seem to be far behind. During the drilling operation, a permanent concern is kick prediction and well control. The ever-growing amount of data generated by oil and gas companies raise the demand for advanced real-time predictive analytics. Data mining and machine learning are two methods integrated with drilling recently, with the objective of preventing getting overwhelmed or lose track of data. Due to some shared characteristics, the line between these two terms sometimes gets blurred. However, their success in challenging domains -like drilling- makes them popular.

Drilling operations generate a sequence of data continuously on a time-series basis. The abundance of sequence data has motivated interest to employ reliable techniques and develop machine learning models for reservoir management, well control, and people safety during drilling while data is continually being collected and stored. Kick prediction is categorized as time-series forecasting where predicting future values is based on previously observed values. In time-series observations, a time dimension is added as an explicit order dependence between observations. Time series forecasting is an important area of machine learning and the appropriate forecasting methods depend largely on what data are available.

The main goal of this thesis is to estimate how the sequence of observations will continue in the future in kick occurrence. Although the application of machine learning techniques for time-series forecasting is not straightforward, a strong data analysis will provide an effective dataset for the model. Improving the kick prediction process relies on the data fed into the model. The occurrence of the kick is detected by locating sensors to monitor drilling parameters. However, sensor-generated data are not directly ready for the model unless they get cleaned and prepared beforehand. Outliers and missing values are frequently encountered in the process of exploring and exploiting oil and gas data collections. After data cleansing, the large volume of data still needs to be customized by feature selection to make the access time fast and the total computation time less expensive. Since time series data have very high dimensionality, an efficient feature selection algorithm selects a smaller subset of raw data with relevant features with aim of reducing computational overload.

This study shows how to develop machine learning algorithms from the data collection phase to the real-time algorithm implementation phase. After data preparation, the type and kind of data play a key role in deciding which algorithm to use. Evaluating some models that might be the best fit for certain situations and then finalizing the model is the best approach that empirically gives you the best results. In a time-sensitive prediction model like the kick prediction model, finding the best parameters for the model is a comprehensive guide for model optimization techniques.

1.3 Limitations

Instead of running the OpenLab simulator only once and get as many as kicks desired for this work, multiple simulations are run for accomplishing this thesis and therefore, the data is spread across multiple tables and must be gathered into a single dataframe. Combining multiple simulations disintegrates the time-series consistency property of the data since multiple shorter time-series are attached together to generate the demanded number of kicks. To detect abnormal and rare behavior in a set of observations with acceptable accuracy, it is required to have as many data samples to achieve a desired level of performance. However, since suppressing a kick during drilling operations is not an effortless action, it is challenging in the OpenLab simulator similarly. Thus, the simulations need to be stopped and start again after a kick occurs.

1.4 Methodology

The following contributions are made in this thesis:

- Get familiar with Artificial Intelligence and Data Mining and their roles in drilling operations
- Understanding big data
- Analyzing big data
- Generate new optimal hyperparameter
- Select optimal hyperparameter combination
- Evaluate training/test dataset required to obtain predictability with the lowest error
- Identify and build machine learning models with the best performance regarding the issue
- Evaluated machine learning models accuracy, Precision, recall and other performance metrics

- Select the best-fit machine learning model to detect misbehavior with lowest False Positive and False Negative flag

The remainder of this thesis is outlined as follows:

Chapter 2: gives a general introduction to the drilling operations and the potential risk and hazards in offshore drilling as this process is considered as one of the five most dangerous professions in the world. Then by taking a deeper look at the abnormal trends in drilling operations, it would be obvious that a safe and effective drilling operation is not possible unless by a clear visualization and abnormal trend analysis.

Chapter 3: introduces Artificial Intelligence and its two primary applications named Data Mining and Machine Learning. This chapter is then followed by CRISP-DM model which is necessary for understanding the steps towards solving this forecasting problem. The chapter finally ends with literature review subsection.

Chapter 4: focuses on Big data and Big data analysis since drilling operation is one of the fields always dealing with a massive amount of data. So, ‘making use’ of data to provide insights that lead to improved performance is challenging. Moreover, the "time" component adds extra complexity to this forecasting problem since the dependencies between observations in inevitable. Therefore, coming through all the previous chapters, chapter 4 categorized the kick detection and kick prediction as time-series multivariate forecasting.

Chapter 5: After becoming acquainted with the main problem and its characteristics, it is time to start working with data. The OpenLab simulator is introduced thoroughly in this chapter. Then the data created via the simulator is analyzed and prepared for further stages. The very important step named feature Engineering and feature selection is applied by use of a useful package TSFRESH from Python.

Chapter 6: examines the factors for choosing the best machine learning model. This chapter also introduces the performance metrics for model assessment.

Chapter 7: contains the implementation and evaluation of the selected machine learning models.

Chapter 8: gives a conclusion and future work.

Chapter 2

Background

2.1 Drilling Operation

The entire global population would experience a better life with oil and natural gas production as a leading industry. Since energy has been a key enabler of living standards, meeting the energy needs of a growing world population will be critical to achieving a greener planet. Oil has become the world's most important source of energy since the mid-1950s, while environmental concerns and new technologies led another energy source to shift from coal to oil. When oil emerged as the preferred energy source, it has been a provider of some of the most essential elements of modern life improving. Primarily, the electric light bulb and the automobile were two key drivers that changed the world. However these days, within our daily lives oil is used almost everywhere like generating heat, fueling vehicles and airplanes, manufacturing almost all chemical products such as plastics, detergents, paints, and even medicines and food.[3]

The drilling process can begin after determining a locality has enough resources to explore. Petroleum reservoirs can be found beneath land or the ocean floor and can be extracted by drilling operations. Drilling is a machining process, involved with the creation of holes by using a twist drill bit. The act of cutting and removing material to create holes is extremely complex as the unknown condition of the seabed. A combination of complexity and a large number of unknown parameters of subsurface layers makes deepwater drilling a challenging business. With a strong focus on areas relating to safety, environment, and cost-effectiveness, drillers may achieve efficient results and highly functional oil and gas benefits.

Parameters involved in drilling play an important role in indicating the quality of the drilling process, in terms of the produced hole quality, tool life, and energy

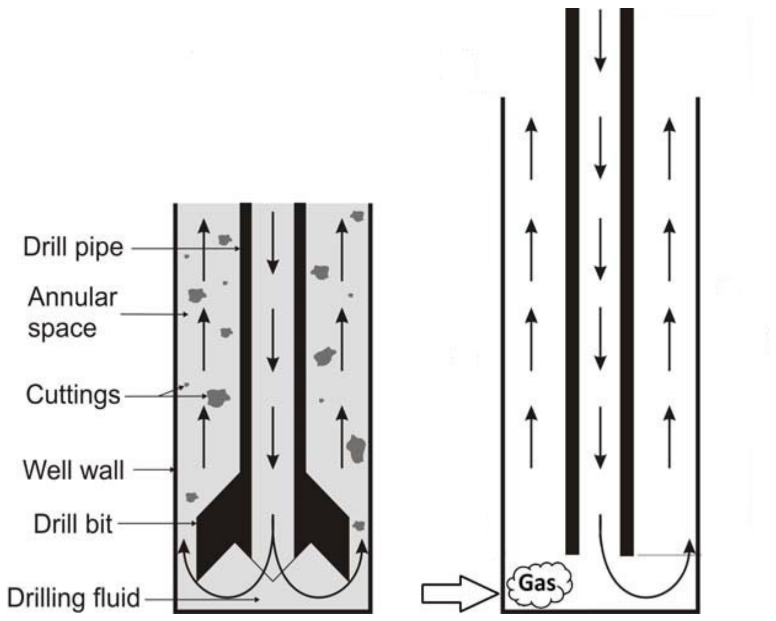


FIGURE 2.1: Illustration of the drilling fluid circulation in the wellbore consists of drill pipes and a heavy drill bit. Gas emission can occure during circulation.[4]

consumption. After the geological location for the well has been selected, the crew sets up the rig and equipment. The drilling process of a well consists of boring a hole by rotating a bit. Drill pipes are durable steel pipes that conduct the force to the drill bit (Figure 2.1). This method uses a heavy bit with the hardest material and sharp teeth that are repeatedly lifted and dropped that crushes and breaks the formation. As the drilling bit passes through a producing formation, gas can emit from the oil in the formation. The excavation of several kilometers deep into the earth's crust continues until a reservoir is reached. During this period, drilling bit needs to be replaced frequently as they get damaged and worn after several days or weeks of using. Drill bit replacement will help to achieve the desired penetration rate and long-term productivity.

Drilling operations rarely leave a clean hole suitable for long-term production, and that is the reason why drilling is performed in the presence of drilling mud. Drilling mud, also called drilling fluid, are mixtures of natural and synthetic chemical compounds which is pumped through the drill bit nozzles at relatively high velocity in order to circulate the fluid. To reach a deeper depth, rock cuttings must be transported out of the wellhole during the drilling process. Therefore, drilling mud is used throughout the process to clean the bottom-hole and carry the rock cuttings to the surface. Another responsibility of drilling mud is lubricating and cooling the drill bit because the bit and drillstring rotation produce a relatively high temperature at the bottom-hole during operations. The drilling mud, moreover, helps prevent the collapse of the openhole -the

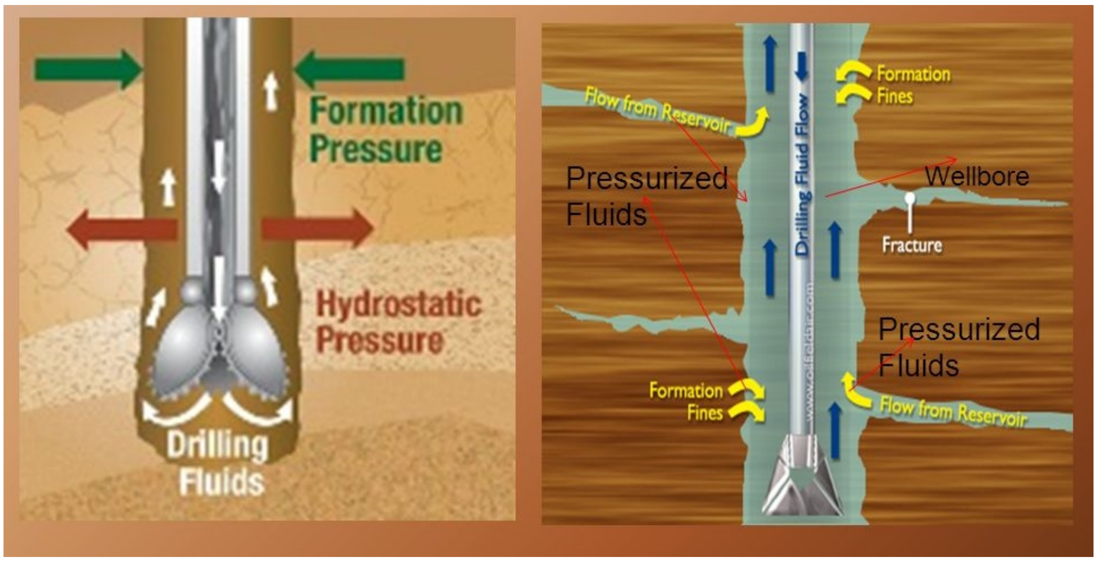


FIGURE 2.2: How a balance between formation pressure and hydrostatic pressure keep the wellbore stable [5]

part of the well that is exposed to the reservoir and not protected by casing- by exerting hydrostatic pressure. This pressure increases in proportion to the depth measured and should balance or exceed the natural formation pressure to help prevent an influx of gas or other formation fluids. A kick or blowout is physically caused by the pressure in the wellbore is less than that of the formation fluids. Therefore, as the formation pressure (pore pressure) increases with depth rapidly, the density of the drilling mud is increased to help maintain a safe margin and prevent kicks or blowouts [6]. As it is shown in Figure 2.2, careful fluid design based on testing and sampling can preserve the wellbore stability and minimize the potential formation damage. Whenever the bit reaches the depth of the targeted zone, the casing and cementing phase begins.[7]

A drill bit is inserted into the well via a drillstring after running the casing (Figure 2.3) and before cementing of well. The drilling fluid is then sprayed out through the nozzles and circulated for a certain amount of time to remove any remaining cuttings from the well for a trouble-free operation[9]. Cementing and running the casing would have unexpected difficulties in presence of cuttings or mud with undesirable properties. When precautions were taken, a cement slurry is then pumped into the well and allowed to harden to permanently fix the casing in place. Since the drilling process is sectional, the well is drilled to a certain depth, cased and cemented, and then the well is drilled to a deeper depth, cased and cemented again, and so on [10]. Displacement fluid such as drilling mud, brine, or water from the casing interior and borehole is a fundamental task cementing is responsible for. Prior to that, placing a cement sheath in the annulus between the casing and the formation provides a barrier to the fluids flow from or into the formation, and bonds the casing to the formation to

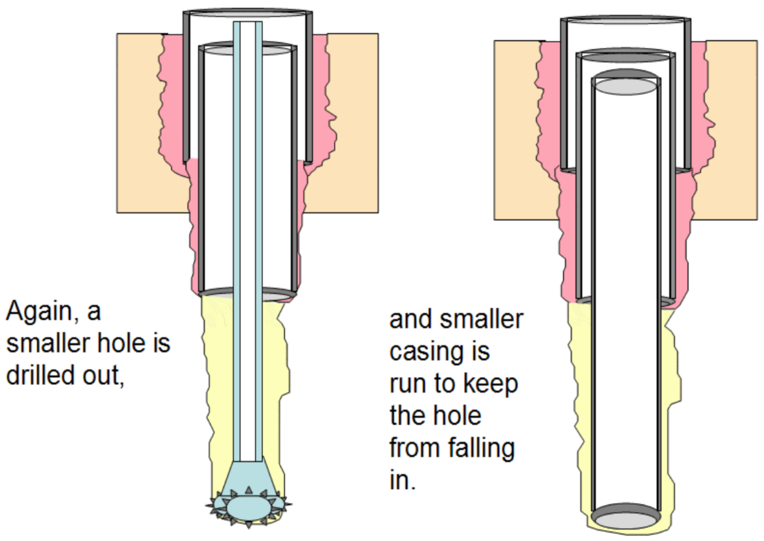

Again, a smaller hole is drilled out,
and smaller casing is run to keep the hole from falling in.

FIGURE 2.3: Runnig casing process is followed by drilling to deeper depths and by continuously adding lesser sized casings until the desired drilling depth is reached [8]

protect the steel casing against corrosion by formation fluids. In addition, the cement sheath provides a hydraulic seal that establishes zonal isolation. finally, testing some properties such as hardness, alignment, and a proper seal can be tested after letting the cement harden. Cementing is a critical procedure in the well construction process and failure to achieve these objectives may severely limit the well's ability to reach its full producing potential [11].

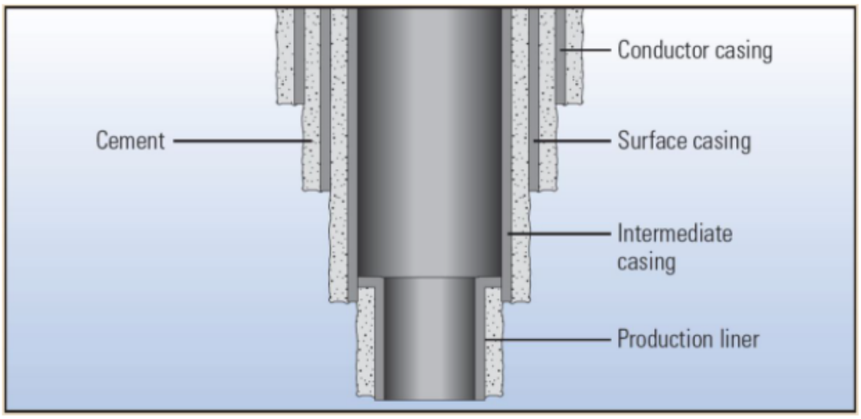


FIGURE 2.4: Typical casing and cementing program. Multiple strings from large to small diameter conductor may be required to reach the target producing zone. [12]

Figure 2.4 demonstrates the process of tripping, casing, and cementing where these processes are continued and repeated for each of the planned casing points until the well reaches the total depth (TD) of the well. Meanwhile, adding new sections (joints) of drill pipes and multiple strings of the intermediate casing may be required to reach the target producing zone as the hole gets deeper. When it comes to drilling operations, it is required to have speed and precision in the process while keeping track of well conditions. Drilling and well operations are characterized by considerable complexity. There are several key causes of well control incidents and vital requirements for monitoring safety and the ability for continuous well control in all phases of a well's lifetime. [13]

2.2 Potential Risks in Drilling

Drilling in any environment is potentially hazardous, and oil and gas well drilling operations are not exceptions. presents additional risk factors. Drilling continues in several stages and servicing activities involve many different types of equipment and materials. There are hazards associated with the preparation of camps, worksites, and drill pads, as well as those specific to the operation of drill rigs[14]. Although from the very early beginning of the drilling process, the operators have always been seeking to reduce the drilling costs mainly by increasing the drilling speed, an efficient, well-designed, and well-operated drill job must ensure maximum safety for drillers and workers, minimum damage for the formation and environment. It is almost certain that many problems and difficulties encounters in different types of hydrogeological formation. Due to the nonhomogeneous formations, problems will occur while drilling a well as the depth and diameter of the hole increases, even in very carefully planned wells. Here, it is vital to take a few precautionary measures and apply wisdom intelligently.[15]

These unplanned events can either lead to minor impacts to the drilling like small amounts of fluid loss, or to catastrophic wellbore failure like the disaster that happened in 2010 due to the explosion on the Deepwater Horizon oil rig in the Gulf of Mexico and caused economic loss, environmental pollution, injuries, and deaths. Although the crew encountered multiple hazards and warnings, a series of decisions that increased risk and failure to consider all risks associated with the operation was a contributing cause of the Macondo blowout[16]. The more the drillers are aware and prepared for the drilling hazards, the better they can recognize and control the troubles to prevent such tragedy. The following are some of the most prevalent problems faced in drilling [17]:

- **Loss of Circulation**
- **Kick and Blowout**

- **Borehole Instability**
- **Pipe Sticking and Drillpipe Failure**
- **Hole Deviation and Hole Cleaning**
- **Mud Contamination**

Offshore drilling is considered one of the five most dangerous professions and the risk is unavoidable. Many drilling operations take place in remote locales with rough seas and harsh conditions and workers are on shift for an average of 12-hours a day dealing with combustible materials and heavy machinery. With seven to 14-days on the rig at a time and isolated hundreds of miles off the coast, make a challenging environment for emergency situations. There is significant value in anticipating drilling hazards and a thorough understanding of the drilling hazard database[14]. Although drilling operation is subjected to multiple factors that consequently cause major or minor drilling hazards, most of these factors are linked to the drilling mud which its characteristics are of high importance and its optimization could lead to risk mitigation.

Kick is referred to as an uncontrolled and unexpected entry of water, gas, oil, or other reservoir fluid into the wellbore during drilling due to an under-balanced condition. One of the objectives of the well control is to make a balance between the mud hydrostatic pressure and the formation pressures. It means the drilling mud must be heavy enough to hold back the formation pressure but not so heavy to cause the formation to fracture. Figure 2.5 shows the expected balance of mud weight. If the pressure at the bottom-hole is maintained at a value slightly greater than the formation pressures, further influxes of formation fluids into the wellbore can be prevented. The unwanted flow -kick- is physically caused when the pressure in the wellbore is less than that of the formation fluids.

The worst kind of kicks are gas kicks and it is recommended to treat all the kicks as gas kicks. The high mobility of gas makes the gas kicks riskier than fluid kicks in the wellbore. They will be pumped up the well, expanding uncontrollably due to less density, displacing fluid from the well and furthermore reducing pressure and allowing more kick to enter. Water kicks may be troublesome, but they rarely constitute a significant threat to the safety of the crew and environment. An oil kick is generally less dramatic than gas kicks as it does not migrate upward and expand as a gas kick does, giving personnel little or enough time to react.

As it is mentioned before, one of the main responsibility of drilling mud is preventing well control issues. The drilling mud also should comply with established health, safety, and environmental (HSE) requirements so that personnel is not endangered and

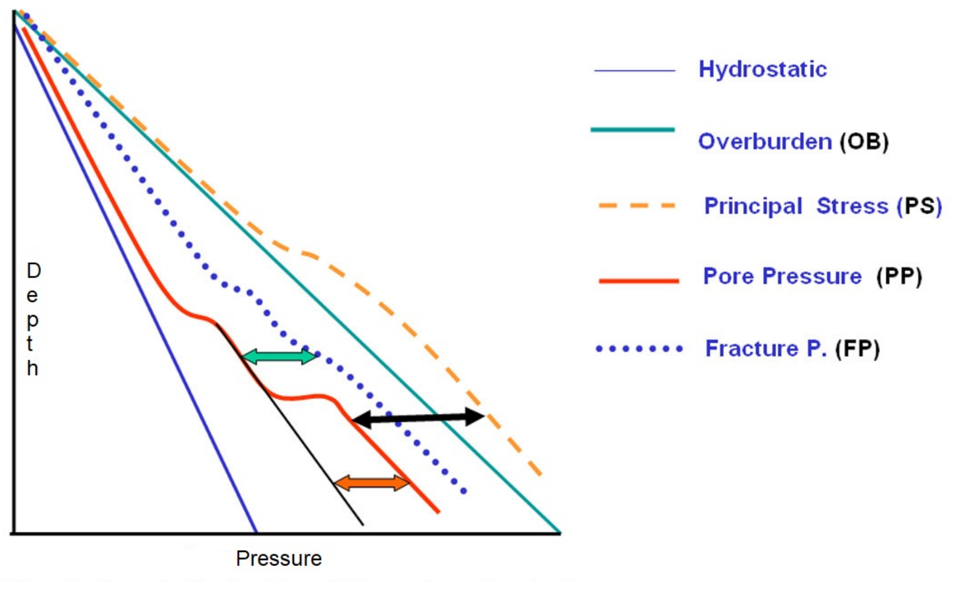


FIGURE 2.5: The right range for drilling formation pressure (pore pressure as red line) to balance the bottom-hole pressure [18]

environmentally sensitive areas are protected from contamination. Mud is also the only component that is in constant contact with the wellbore as it is circulated down the drill string, sprayed out the bit nozzles, and backs up the annulus to the surface throughout the entire drilling operation. Once mud continues to flow to the surface with some loss to the formation, partial loss circulation happens. On the other hand, total loss circulation occurs when the mud flows into a formation with no return to the surface. Therefore, the drilling mud is a major factor in the success of the drilling program and deserves careful study.

Because kicks have been always one of the most challenging concerns facing the industry, taking necessary measures in controlling and handling them is of paramount importance. This is often attributed to the use of experts and experienced personnel in key positions. If the unwanted flow is controlled, the kick is considered killed. However, an uncontrolled kick that increases in severity lead to catastrophic issues what is known as a blowout[19]. Therefore all personnel must be aware of kick indicators and be prepared to take immediate action if any indicator or warning sign of a kick appears. Since kick indicators in well control are of the utmost importance to prevent a well control incident, it is crucial to have a careful observation and positive reaction to these signs. The most common causes of kicks are:

- Insufficient mud density
- Lost circulation
- Abnormal pressure

- Poor tripping practices
- Cementing operations
- Swabbing/surging

Any change from established trends usually indicates downhole problems. The alertness in monitoring parameters and recognizing signs and warnings is of the utmost importance to prevent a well control incident. Determining early possible kick indicators in well control could be sufficient to assist companies and regulators in preventing another Macondo type incident. If the interaction between crew and system (indicators, signs, warnings) is to be focused on constantly, the part of the accident related to the operators' ability would decrease enormously. For instance, in the Macondo disaster, bad management and a communications breakdown by BP and its partner about unexpected results from a critical negative pressure test on the rig were the main causes for the incident. The misreading of that pressure test and the decision to move ahead with an overbalanced condition led to one of the world's worst offshore oil rig disasters. The first well control actions were to observe well kick indications and to react promptly to minimize the influx and the blowout probability by closing the BOP (blowout preventer) and then routing the fluids exiting the riser to the Deepwater Horizon mud gas separator (MGS) system. Poor decisions by management were the real cause that an entirely preventable disaster changed to loss of lives and waste of oil.[20]

The various parameters that have been recorded throughout the drilling give an effective indication of the quality of the drilling process, in terms of the produced hole quality, tool life, and energy consumption. This process, which is sometimes referred to as measuring while drilling (MWD) or logging while drilling (LWD) involves monitoring the drilling process by mounting several sensors on a high-cost environment like rotary destructive drilling rigs where wireline logging is difficult and time-consuming. MWD/LWD offers advantages to measure drilling parameters such as fluid density, torque, drilling speed, flow-in and flow-out differential (delta flow), stand pipe pressure (SPP), rate of penetration (ROP), weight on bit (WOB), and so many other factors like pressure and temperature. The outcome of the process is a set of logs that describe the variation of the drilling parameters with depth and indicate any deviation from standard trends and patterns in order to help analysts establish the correct orientation of the drilling system, avoiding costly mistakes in the process. Changes in any of the above-mentioned parameters can indicate pressure changes in the well and potentially that the well may be kicking.[21]

2.3 Abnormal trends in drilling

Throughout the drilling process, a huge amount of data in form of sensor measurements is produced over time. This data contains the main source of detailed information about drilling operations. Each of these drilling operations has a specific pattern in rig sensor measurements. To minimize the consequences of undetected kicks, it is desirable to analyze the existing patterns so as to identify abnormal drilling situations. Abnormal trend detection is going to be widely used in drilling activities for detecting one or more hazards with the aim of providing a safe and effective drilling operation as any of the one or more hazards could be avoided. In this regard, several indicators might be defined to accelerate the abnormal trend analysis. Moreover, one or more thresholds could be outlined to support the safety process. When a trend analysis indicates that a threshold has been reached or exceeded, an alarm would be triggered. Then, a drilling operation should be altered or aborted decided by the drilling crew. This technique could reduce drilling costs and minimize the probability of encountering problems due to working with optimized parameters.[22]

In well control, since kick poses the highest risk to the safety of the wellbore, kick indicators are of the utmost importance to show what indicators are positive evidence of a kick or what are the warning signs. To monitor the down-hole conditions, while some experts focus on analyzing the returning drilling fluid at the surface, others rely on downhole monitoring along with surface monitoring. The exclusive surface monitoring has several limitations, such as a delay due to lag time and thus losing precious reaction time available to the crew to take actions to the kick. However, the downhole monitoring itself is prone to many challenges, including the possibility of excessive false alarms. Due to the fact that kick detection is complicated, the possibility of blowout prevention could be high if multiple kick indicators will be monitored simultaneously. Reading and interpreting a mixture of sensor data together could limit the frequency of false alarms significantly.[23]

In general, the rig activities have very clear and straightforward patterns in real-time measurement. When deteriorating trends are observed for some drilling parameters, this could be a sign of potential unwanted drilling conditions. Sensors are placed within the bottom-hole assembly to reveal what is happening close to the bit, while the surface sensors indicate what is happening with the drilling fluid properties. By distinguishing between normal and abnormal operation trends, a diagnostic procedure is established to identify borehole changes up to several hours in advance and take preventive action[24][25]. But, there is one question that might occupy investigators, leaders, and safety professionals' mind:

“Why did the crew fail to see what was going to happen?" or "How could they have not recognized the signs of the hazard?"

The answer appears to be found in the difference of "errors" and "failures" [26]. Detecting process deviations -known as symptoms- during the drilling procedure, will lead to capturing a probabilistic understanding of the downhole process. Then, the recognized symptoms are used as input parameters are translated into understandable concepts, and then interrelated through cause-effect relationships in pathways linking causes to related failures and which errors are causing the failure. Not all the symptoms are signs of probable hazards or uncontrolled situations. The observations of different experiments have suggested only a few deviations from the standard can be a sign of a kick situation. For example, the existence of different flow patterns is sometimes based on hole cleaning, pipe movements, tripping. Although changes in flow rate are powerful signs of kick, it is essential to first establish a certain threshold and stop the process when deviation from the normal trend is above the threshold [27]. It is recommended that in experiencing symptoms, the driller should consider the potential for the well to kick and check that if everything is as it should be before closing the well.

Early detection of kick requires the crew to notice any subtle changes in established patterns of rig activities. By investigating the sources, drilling engineers can come up with improvements for the rig performance and get rid of the abnormality. However, the various signs that have been recorded as early warning indicators may change from well to well so they are not consistent in all situations. Generally, as it is explained before in section 2.2, a kick is defined as an unexpected influx of formation fluids into a borehole. The pressure window seems to be wide near the surface but as drilling proceeds in deeper depth, the pressure window becomes narrow. The kick could occur when the formation pressure exceeds the wellbore pressure and fluids begin to flow from the formation into the wellbore. In other words, when the pressure exerted by the drilling mud column is not great enough to overcome the pressure exerted by the fluids in the formation drilled, a kick can happen. Some of the warning signs of kick can be found from the following list:

- **Changes in flow**
- **Increase in drilling rate of penetration**
- **Pit gain or loss**
- **Increase torque and drag**
- **Mud property changes**

- **Changes in Shape and Size of cuttings and rocks**
- **Changes in background gas**
- **Changes in return mud temperature**
- **Decrease in D-exponent**
- **Drilling fluid density reduction**

In normal drilling, there are some fluctuations that are categorized into normal trends. However, any deviation from the normal trend of a parameter is called abnormality and should be detected before losing well control. To evaluate downhole conditions while drilling numerous techniques are being utilized including drilling and logging indicators. It is vital to interpret all indicators in a group since considering them individually regardless of their relation to other parameters would result in invalid decision-making. The rest of this chapter is dedicated to reviewing some of the important signs of kick.

Using a variety of equipment and laboratory techniques, mudloggers are responsible for collecting and monitoring information from drilling operations and rock samples. This information is then interpreted by the driller team for operational purposes.

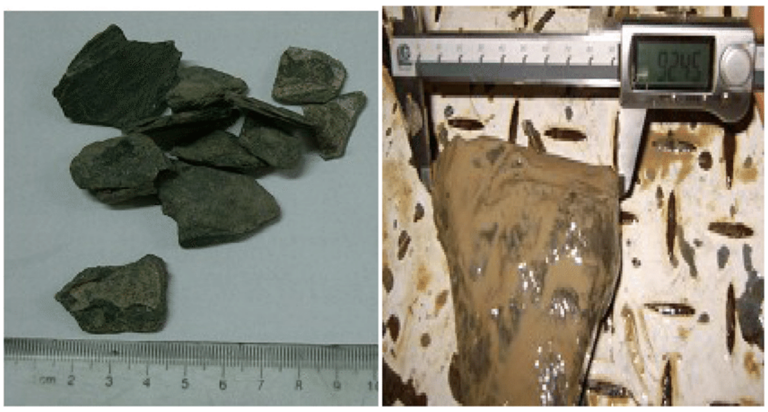


FIGURE 2.6: Any change in drilled cuttings size and shape might indicate well control problematic issues [28]

For example, drilled cuttings brought out of the well should be rather consistent in size and shape. (Figure 2.6). Normally, pressured formation produces small rocks that are flat with rounded edges. Under-balanced situations developed by abnormal high pore pressure zones can cause the formation to break. The broken cuttings are more sharp and big in comparison to those which are cut with a drill bit. Therefore, larger irregular pieces of cutting could indicate a well control situation. Changes in cutting shape and cutting load need to be monitored at the surface. However, among all parameters, the one which can be measured by sensors and not human are of utmost interest in this work.[28]

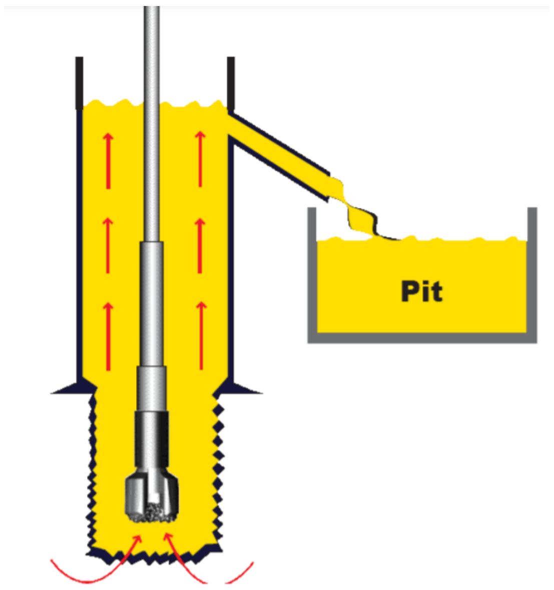


FIGURE 2.7: Pit level is constant during normal drilling and circulation [21]

When a kick enters a wellbore, it means the mud weight has failed to meet the requirement to be at an adequate level to overbalance pore pressure. Therefore, the influx begins finding its way to the surface and shows up as a gain in the volume of mud at the surface and also an increase in the flow-out rate of the well. In normal circulation, the flow-in and flow-out of the well are in a steady state condition. As illustrated in Figure 2.7, the pit level is constant during normal drilling. A kick breaks this balance since the fluid system in a drilling rig is a closed system and increases in the amount of flow-out will cause increases in pit level. A kick is an unwanted and unexpected amount of fluids entering the wellbore that displace an equal volume of mud resulting in pit gain (Figure 2.8). Adversely, a decrease in pit volume indicates a primary indicator for loss circulation in drilling as it shows a leakage or fluid loss to the formation.

In the drilling process, flow-rate change might happen with drill pipes pulled up to do connection or placed back for further drilling. In some cases, a small number of changes in flow-rate are not always dangerous, but they still need to be recognized based on their peak flow-rate. Traditional alarm systems were sensitive to very small and simple changes and consequently generated a large number of false alarms. Probability computations and threshold set up is a way to reduce false alarm in this regard. Based on the work has been done in [29], modern ML algorithms can reduce false alarm while maintaining tight alarm threshold as it is shown in Figure 2.9. The ML algorithms can

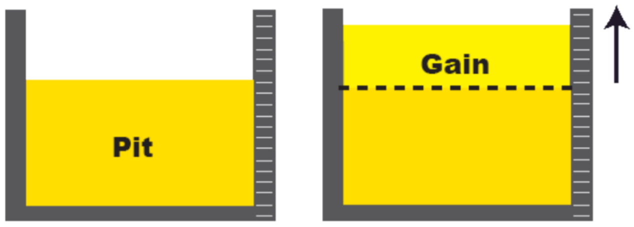


FIGURE 2.8: Pit gain (loss) is a relatively common problem in deep-water wells due to the entrance (exit) of unwanted and uncontrolled amount of fluids into (from) the wellbore. [21]

recognize any unusual gain or loss immediately. False alarm in a system influences the tendency of not taking alarms very seriously even when an influx or a threat happens. However, it is highly suggested to respond to the flow-rate changes or any other changes to be able to take corrective action in a timely manner rather than ignoring the alarm that indicates the danger.

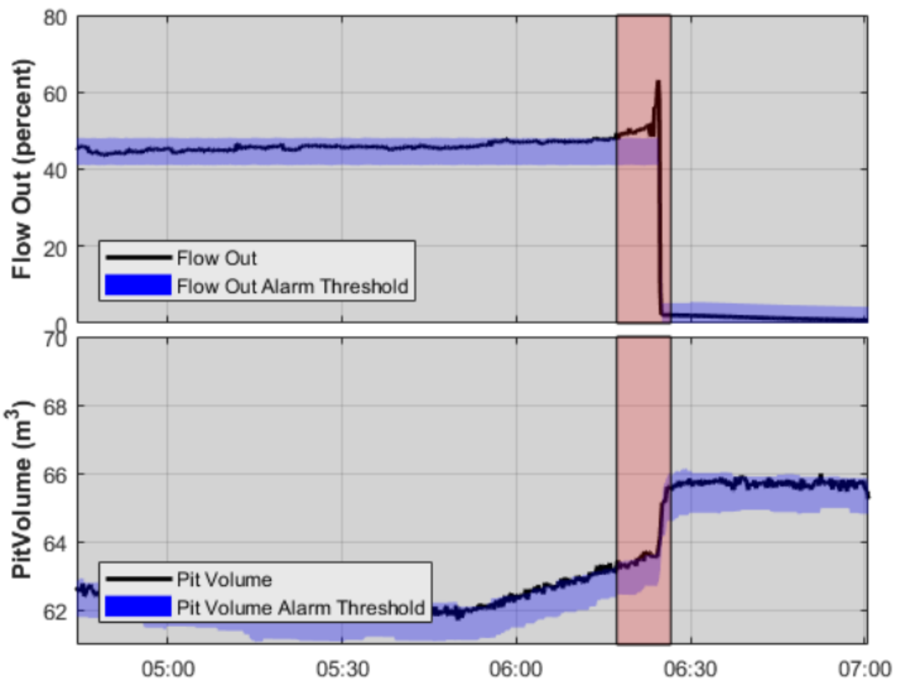


FIGURE 2.9: Abnormal behaviour in flow-rate and how ML algorithms can reduce false alarm by maintaining tight alarm threshold [29]

Another factor that indicates kick is penetration rate which is the speed at which the drill bit can break the rock and thus deepen the wellbore. Generally, as drilling progress, a normal trend is the slight decrease in the penetration rate as the well is being drilled ahead. In this state, a sudden increase in ROP might show downhole abnormal conditions which should be taken under consideration. The change in the rate of penetration is known as a Drilling Break(Figure 2.10) and might happen either when the soft formation is being penetrated or abnormal high pore pressure zones are encountered. Although the former result can be ignored, the latter result indicates the difference between hydrostatic and formation pore pressure goes down and the influx could have entered the well.

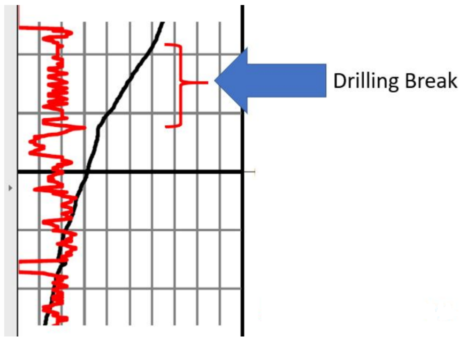


FIGURE 2.10: Any change in drilling normal trends can be warning indications of occurrence of a well control situation. For example a change in rate of penetration is called drilling break and it is a sign of potential problems. [21]

Generally, the density of formation increases with depth. The mud column is providing the hydrostatic pressure in the wellbore and this is the primary means of preventing a kick. The circulating mud controls pressure by adjusting its weight at each depth. Entering an abnormal high-pressure zone disturbs the balance and consequently, a decrease in the formation density occurs as illustrated in Figure 2.11. The decrease in the rate of compaction of shale in this trend shows an abnormality in pore pressure. This happens because an increase in pore pressure within the formation prevents compaction.

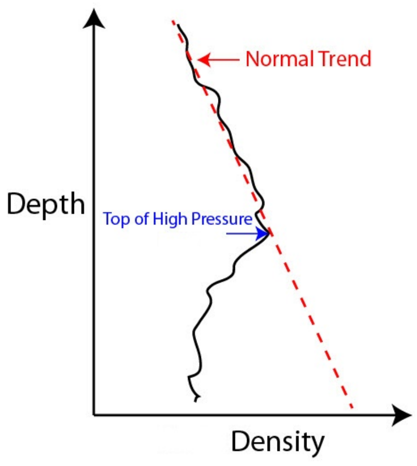


FIGURE 2.11: A deviation from a normal trend in formation pressure indicates a high-risk incident is going to happen [30]

In drilling parameter measurements, the downhole mud weight data such as ECD (equivalent circulation density) or ESD(equivalent static density) monitoring could be possible ways of kick detection, since a kick can be identified from the pressure while drilling (PWD) measurements like an abrupt increase in downhole pressure and temperature (Figure 2.12). Taking into account that the temperature will normally take a sharp increase in transition zones, abnormal changes in temperature could also be caused by many other factors including circulating rate, mud volume, hole size, and especially drilling into transition zones. Usually, the formation fluid flowing into the wellbore has a higher pressure than the mud pressure and a higher temperature than the mud temperature. Therefore, considering downhole measured mud pressure and temperature can give an early indication of kick.

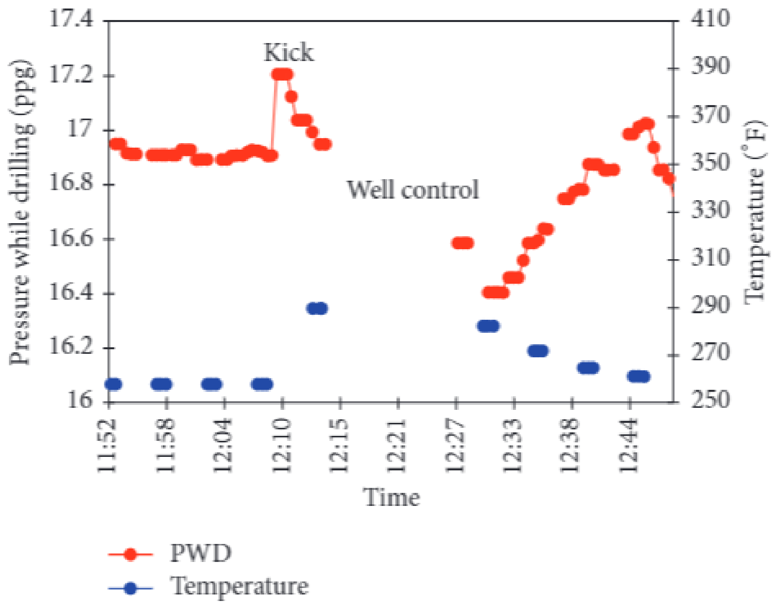


FIGURE 2.12: Well influx/kick detected from pressure while drilling showing an abrupt increase in measured downhole pressure (PWD or ECD)and temperature. [31]

Chapter 3

State Of the Art

3.1 Artificial Intelligence in Drilling

During the last decades, there have been major changes in well requirements and drilling capabilities. There are many different factors involved during drilling such as fluid formulation, property determination, its performance in the well, and its relationship with other wellbore drilling parameters while the relationship among them is complex yet advantageous. While new capabilities allow drillers to realize more ambitious well objectives, they also face new challenges. Monitoring the characteristic of each parameter especially at downhole conditions is a challenging task requiring advanced modeling techniques as well as human intuition and experience. Traditional data analysis methods were not sufficiently capable of rapid assessment of extensive datasets from a rig. Today, artificial intelligence (AI) is making a difference in a discipline looking for a significant improvement in the drilling process because the value of AI is making better decisions than what humans alone can do.

Although humans are naturally prone to making mistakes, these mistakes can have particularly devastating and long-lasting effects in some areas. As discussed before, human error is one of the greatest causes of unwanted events due to false data analysis in the oil and gas industry. Around 80 percent of accidents in the offshore oil and gas industry are blamed on human error. Where human error originates in a lack of knowledge or focus, AI which is considered as a remedy for human errors has replaced human intelligence, solving complex problems with a level of consistency and speed that's unmatched by human intelligence and the revolution begins. AI has emerged as an enabling technology for several purposes and addressing more complicated problems. However, it can not replace human intelligence where it is needed actually. AI exists to empower human intelligence to tackle high-level issues. For example, in offshore situations, once a problem is detected, collecting relevant rig data and designing the right solutions are all done by humans. Afterward, the AI come up with creating the right processes for the AI solutions to adapt, learning from feedback, and producing results. [32]

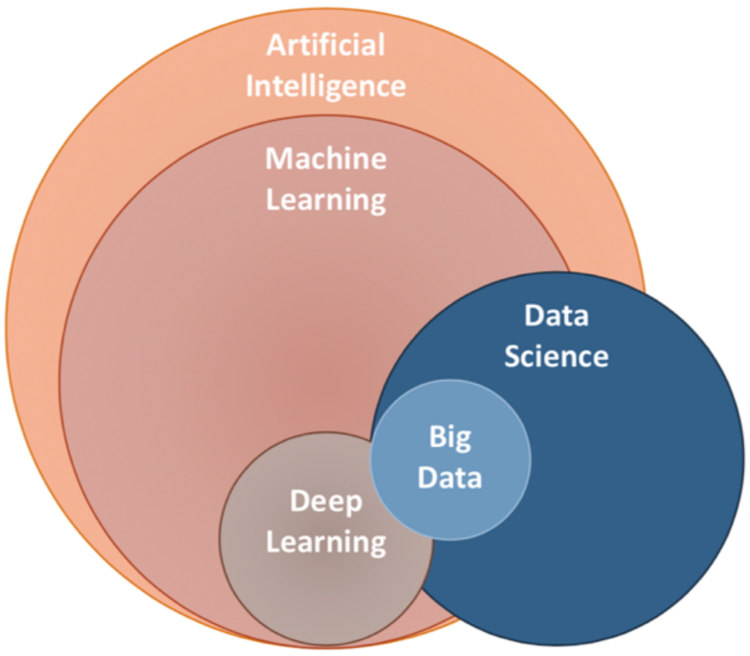


FIGURE 3.1: A clear visualization of overlapping Artificial Intelligence-related terminology. [32]

Digitalization of oil and gas technological processes based on the use of AI methods is among the prevailing trends of the 21st century. AI is a term used to cover different types of analytic. Although AI is a diverse field, within the oil and gas industry there are two primary applications of this technology: machine learning and data mining. 3.1 From initial exploration activities all the way through to the end, AI applications indicate how the technology has helped cut operational costs and enhance efficiencies across the industry. it's no surprise that a recent EY (Ernst and Young) survey showed more than 92 percent of oil and gas companies are either currently investing in AI or plan to in the next two years.[33]

3.1.1 Data Mining

In drilling operations, only the data can reveal the right and the wrong about the procedure. The continuous and ever-growing volume of data poses a serious challenge for companies narrowing their efforts on cleaning up large datasets to uncover valuable new insight with the aim of improving reservoir production. The complexity and uncertainties of drilling data make an urgent need to summarize signal processing and sensor-data analyzing so that practitioners in this field can understand each other in order to enhance oil and gas drilling functions. Data mining is a process used by companies to turn raw data into useful information in meaningful patterns and trends. "The parameters recorded for drilling optimization are critically important to be representative of the data they are meant to reflect" [34]. With the help of data mining, previously unknown and possibly useful relationships in data are discovered in order to provide the data in an easy-to-share format like graphs and tables. Since companies are dealing with significantly larger sets of data with more varied content, discovering useful structures such as patterns, models, and relations in data needs a big data strategy. The major concern regarding this application is using only selected information, which is not representative of the overall sample group to prove a certain hypothesis. [35]

3.1.2 Machine Learning

Machine learning (ML), is a computer system that can learn to perform automated tasks and think for itself through an algorithm that absorbs new data and experiences. Following human-written algorithms, ML allows users to recognize patterns in vast assortments of data to predict possible outcomes, to plan and take action for the future. ML is the best choice to analyze large data sets with algorithms and make predictions based on the insights gained from the data. This important branch of AI impacts the world around you in both obvious and obscure ways. In simple words, artificial intelligence is the study of creating computer systems that operate with human-like intelligence to accomplish repetitive or tedious tasks, and then machine learning is a prominent way of achieving that goal. The impact of AI and ML on the oil and gas industry goes beyond cost savings, offering several potential advantages.

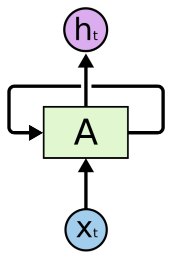
ML has changed the way the oil and gas industry runs. Not because it learns how to replace humans, but because it reshapes the oil and gas exploration and production landscape. It is now possible to teach machines to enhance the skills, performance, and cognitive power of humans. Within the offshore oil and gas industry, injecting ML-based techniques into workflows allows computer systems to learn from and interpret data without human input, refining the process through iterations to produce programs tailored to specific purposes. To make better use of the data, first, data mining is applied to clean and enrich the density of well-log datasets and avoid monitoring complex internal operations and be able to respond quickly to concerns that human operators may not have been able to detect. Then, prepared data from well logs can be fed into ML algorithms to produce accurate models to predict the probability of future events or accidents. The specialists can use AI in data science to accelerate their interpretation of complex data and more importantly provide a robust feature detection tool with a more exhaustive approach than manual efforts to make exploration and production more accessible. ML can also be used to run simulations, using predictive data models to discover patterns based on a variety of inputs and discover new exploration opportunities based on existing infrastructures. Therefore, ML does not necessarily replace people, but it only enables talented people to work better, safer and smarter.

Although the chosen algorithm for solving a problem must be relevant to the problem and the data, None of the ML algorithms works best for every problem. There are many factors affecting the process of choosing a proper algorithm among all the present algorithms such as the structure of the datasets and the nature of the problem. As a result, many different algorithms should be taken into consideration to evaluate their performance and choose the final winner. In Chapter 6, I will discuss the steps toward choosing the best ML algorithm among the common ones.

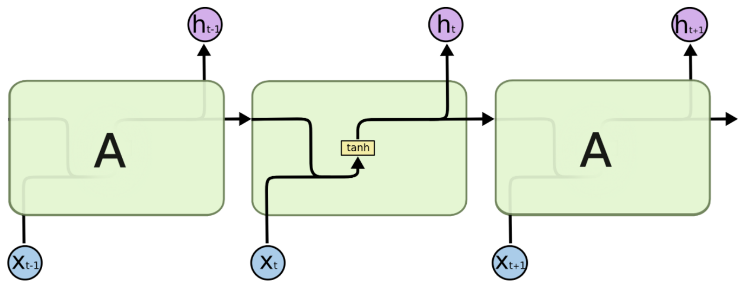
3.1.3 Neural network

Among the common ML algorithms, the Neural Network (NN) has the advantage of finding complex patterns and nonlinear relationships between inputs and output. Due to the sequence dependence among the input variables in time-series prediction problems, extra complexity is added to this kind of problem. A powerful type of Neural Network called Recurrent Neural Network (RNN) is able to handle sequence dependence in complex areas of deep learning.[36] Mostly, RNN designed to recognize data sequential characteristics and use patterns to predict the next likely scenario for problems that are not solvable with another ML modeling.

Traditional neural networks have a major shortcoming in understanding previous observations and informing the upcoming observation about what happened in the past. RNN addresses this issue with the loops that exist in the structure that allows information to persist. Figure 3.2a shows that once the output of the network is produced, it is copied and returned to the network as input. RNN preserves the context of previous inputs and thus, both current input and output are analyzed as the network trains. In Figure 3.2b, $tanh$ is the activation function that can be replaced by any other activation function as well.[37] Although the main and most important characteristic of RNN is the Hidden state in vanilla RNN, the hidden state is constantly being rewritten. The Hidden state remembers some information about a sequence, but it is extremely difficult for RNN to learn to preserve information over many timesteps. Therefore, the main disadvantages of RNN are vanishing gradient and exploding gradient problems.[38]



(A) RNN loops



(B) The repeating module in standard RNN contains a single layer

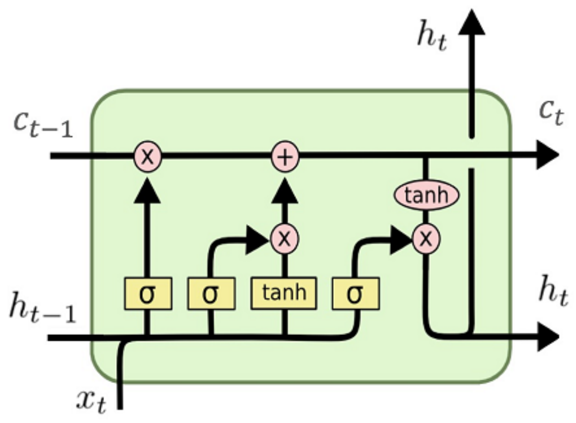
FIGURE 3.2: Recurrent Neural Network [39]

Long Short-Term Memory (LSTM) networks are a modified type of recurrent neural network capable of solving the underlying problems. Since RNN suffers from Gradient vanishing and exploding problems, LSTM resolves the problem by remembering past data in memory. LSTM units include a memory cell that can maintain information in memory for very long periods of time and thus LSTM is capable of learning long-term dependencies.[40] Moreover, a set of gates is used to control the flow of information regarding when information enters the memory when the data is output, and when the data is forgotten as it is depicted in Figure 3.3b.

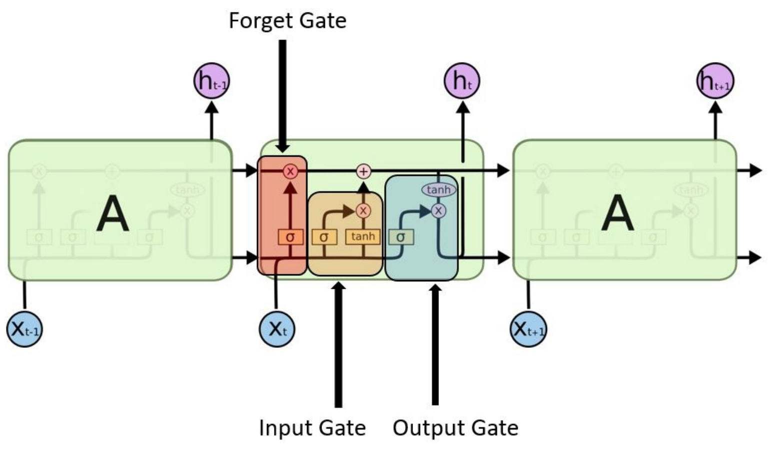
LSTMs are often referred to as fancy RNNs as they utilized various gates. Gates intelligently distinguishes which inputs in the sequence are important and which inputs are of lower importance. The gates are capable of storing useful information in the memory unit to be passed to the following cells. Vanilla RNNs do not have a cell state. Instead, they maintain memory-like arrays called hidden states and those hidden states serve as the memory for RNNs which store information extracted from a long input sequence. Meanwhile, LSTM has both cell states and hidden states.[41] The cell state, which is regulated by "gates", has the ability to remove or add information to the cell. Because of this cell state, in theory, LSTMs are able to handle the long-term dependency which is difficult in practice.[42]

3.2 CRISP-DM

Any good project starts with a deep understanding of the problem initially and followed by providing an efficient and effective solution to resolve the business issues. In typical analytics projects which involve multiple steps like data cleaning, preparation,



(A) LSTM



(B) LSTM Gates

FIGURE 3.3: Long Short-Term Memory (LSTM) network [39]

modeling, and model evaluation, a framework for recording experience and is needed to allow projects to be replicated. CRISP-DM stands for *Cross Industry Standard Process for Data Mining* and is an open-source and widely used methodology created to shape Data Mining projects. This comprehensive methodology provides anyone with a complete blueprint for conducting a data mining project. The process breaks down the lifecycle of a data mining project into six phases to encourage best practices and help to obtain better results overall.(Figure 3.4)

CRISP-DM model is an idealized sequence of events that creates a long-term strategy by structuring a basic and simple but still "good enough" model during the first iteration and improved the model in further iterations. Following CRISP-DM guidelines, a leading approach for managing data mining and predictive analytic for big data is

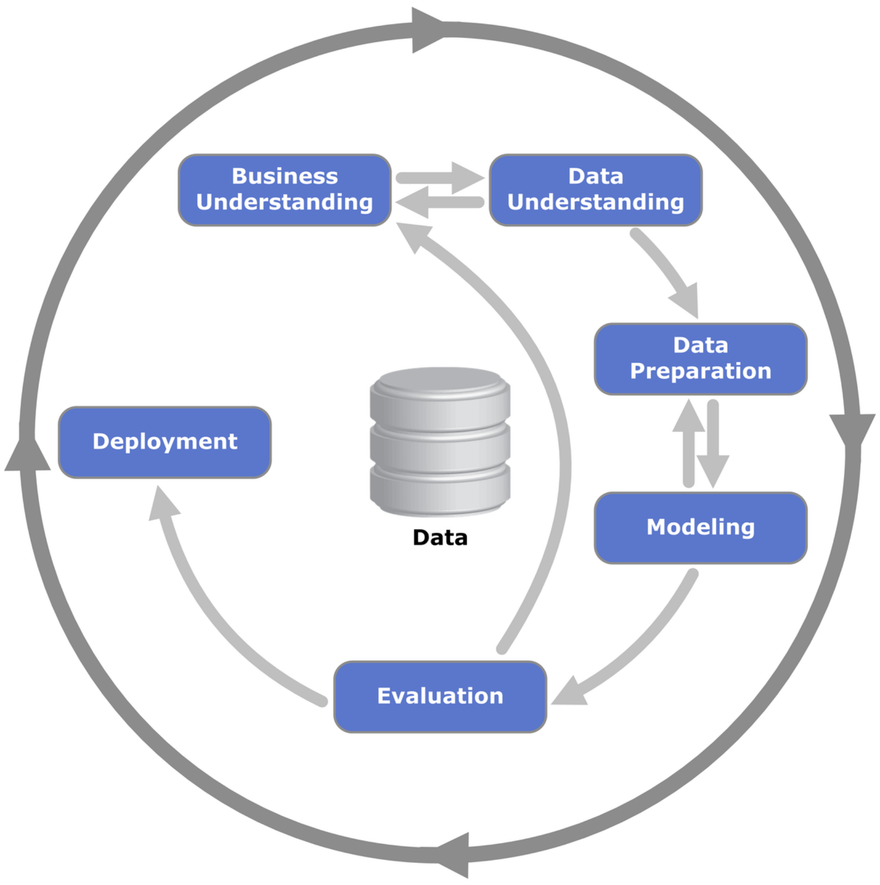


FIGURE 3.4: CRISP-DM overview and how a data mining project breaks into six phases to obtain better results [43]

available. Based on the preferences and needs, the following milestones are beneficial to implement data-driven analytics: [44]

1. Collect: gather all input data
2. Organize: store and organize data to be ready for analytics
3. Analyze: build data science and machine learning models
4. Deploy: deploy predictive analytics and data science models
5. Validate: validate the predictive analytics and machine learning
6. Trust: trust the predictive output and result

3.3 Literature Review

Different methods from different disciplines are being used nowadays in drilling activities in order to obtain a safe, environmentally friendly, and cost-effective well construction. Inevitably, offshore drilling involves high-risk accidents due to the harsh and complex offshore environments. One of the greatest risks in deepwater is the risk of losing well control which can lead to fatal consequences. Hazardous and dangerous conditions are just part of the job so that "safety first" is a slogan many oil and gas companies are striving to make a reality. Mitigating hazardous accidents put the issue of risk in offshore drilling operations into stark focus.

Conventional gas kick-detection approach is discussed in [45] and [46] by implementing a real-time distributed acoustic signal propagating through the drilling mud and the reflected acoustic energy is displayed as a reflection of possible kick. In wellhead sonar system [47], similar to acoustic approach, no downhole equipment is needed and the system functions properly even in absence of the drillstring. The bottom-hole reflection will occur if any free gas is present in the well. According to [48], because of the large amount of attenuation of sound waves in the drilling mud and the new directional drilling, the sound wave signals received after a long trip through the mud might be too weak for data processing and also be affected by other factors. Since it is difficult to apply the traditional ultrasonic technique in flow measurement, an Ultrasonic with Doppler Effect is proposed for early kick detection in this paper, even though acoustic principle methods are faster.

[29] exclusively focuses on situations when a rig experiencing kick or lost circulation as they could be worst-case scenarios. Then, by introducing adaptive alarm system and calculating alarm thresholds in real-time false alarms are dramatically reduced. Perhaps, the most promising aspect of adopting machine learning algorithms for drilling hazards detection is that they can significantly reduce false alarms, hence crew fatigue in a loud and stressful work environment will not affect the performance. Although in conventional drilling, the only widely accepted response to gas kicks is to shut the well in BOPs, a new Managed Pressure Drilling (MDP) technology utilizes a variant named constant bottom-hole pressure (CBHP) which circulates small and medium-sized influx out of the well with no more need to shut the well. However, [49] and [50] share their concerns regarding the definition of small or medium-sized influx that has not been fully addressed previously. Therefore, a reliable early kick detection algorithm is proposed by them based one simulation results and case study data. The algorithm relies on monitoring flow in and out and also exploiting pressure sensors. [51] describes new research on the subject of drilling safety. In this work, the combination

of downhole differential pressure, mudline(seabed), and surface fluid is utilized in the early identification of kick detection. [52] introduces a model-based estimation which uses surface measurements available in MDP to make estimation pore pressure and reservoir inflow rates during gas kick events.

In every competitive market field, AI-based devices and tools have provided tremendous potential for generating accurate analysis and results from large datasets, and the oil and gas industry is no exception to it. AI techniques are developed and deployed worldwide aiming to improve decision making as more data is fed into the system [53]. [54] and [55] review and analyze different successful applications of AI and ML that as related to the major aspect of the oil and gas industry namely ANNs, Fuzzy Logic, Genetic Algorithm, SVM, RF and etc and also reveals their advantages, disadvantages and purposes. These techniques have made a huge impact on different AI applications leading to saving time, minimizing risk ,and saving cost. According to [56], traditional data analysis is not capable to extract and process big complicated datasets. With several supervised learning techniques, the paper demonstrates how to manage interpreting trends, to detect failure patterns ,and to execute remedial actions to mitigate malpractice.

As a kick can pose a significant threat to safety drilling, high prediction accuracy and repeatability are of utmost importance in drilling risk management. [57] constructs a prediction model based on the parameters and the time-series analysis method (ARIMA) reveals the autocorrelation point of the parameter with a high fitting degree. This method has been applied on characteristic parameters such as pit gain and casing pressure as they indirectly reflect the bottom-hole condition and changes in them with time can be used to determine the severity of the kick. [58] deployed five different machine learning models to optimize kick detection: Decision Tree, K-Nearest Neighbor (KNN), Sequential Minimal Optimization (SMO) Algorithm, Artificial Neural Network (ANN), and Bayesian Network. The models have been trained based on surface parameters and among all five, Decision Tree and K-Nearest Neighbor outperformed the rest. [59] has presented a methodology for using supervised learning in facilitating early kick detection by combining a simple Artificial Neural Network (ANN), binary classifier and downhole monitoring of drilling flow parameters to build data-driven kick detection models. The authors were further motivated to look for simple ANN models that would be sufficient for the kick detection problem because numerous papers report the use of complex neural networks to solve the problem without justifying the need for such complex solutions. To differentiate among the different anomalous drilling events a more complex neural network (NN) architecture will be required.

[60] has investigated a data-driven Bayesian Network to solve kick problems in complex

systems where the knowledge about the system is not adequate to apply a model-based method. Downhole parameters are used for early kick detection. [61], similarly, experienced monitoring downhole parameters led to early kick detection. It was observed that gas kick effect directly on dynamic parameters including downhole pressure, mud density, mass flow rate, volume flow rate, dynamic weight on bit ,and rate of penetration. A dynamic neural network model is presented in [62] with different normalization methods and no false alarm happened during prediction. A major challenge in early kick detection is the increase in false alarm. An assembled long short-term memory recurrent neural network (LSTM-RNN) with d-exponent and stand pipe pressure data is proposed as a methodology for early kick detection without false alarm [36]. The methodology involves obtaining the sloping trend of the d-exponent data and the peak reduction in the standpipe pressure data for training the LSTM-RNN model for kick detection. However, only two kick indicators were investigated: the drilling parameter group (DPG) and the flow parameter group (FPG). The DPG includes four (4) input features: pit gain, Rate of Penetration (ROP), Rate per Minute (RPM) ,and Weight on Bit (WOB). The FPG includes three (3) input features: Flow-out, Flow-in ,and Standpipe Pressure (SPP). In total, there are seven (7) input features that are used in the LSTM model development, which does not include several extremely important features, such as Bottom Hole Pressure (BHP), Differential Flow Out (DFO), Weight on Hook (WOH) and etc. Furthermore, these neglected parameters contribute to gas kick detection in a less accurate and reliable manner.

For kick detection and also influx size estimation during drilling operations, [63] developed a new model of LSTM-RNN using OpenLab data. Detecting and quantifying the influx of fluids between fractured formations and the wellbore with high accuracy is the main goal of this paper. [64] modeled a total of 86 experiments in which three groups of complications (stuck-pipe or sticking, loss circulation, kick or gas-oil-water occurrence) and standard drilling operations were simulated to minimize the number of false alarms. The NN models are trained more efficiently when using not only the input values of drilling parameters but also the output results of some auxiliary machine learning models. [65] worked on three supervised learning algorithms based on different indicators. Then an LSTM-RNN is initialized and after evaluation and testing the best model was selected and deployed. Lost circulation is one of the frequent challenges encountered during the drilling. [66] 385 field datasets and new models were developed to predict the lost circulation solution for vertical and deviated wells using ANNs and SVM. It is concluded that some input parameters such as losses rate and lithology type have a more significant effect on lost circulation solutions. However, the performance efficiency demonstrates the advantage of the SVM over ANNs. Another work in [67] has collected data from lost circulation events for 50 drilled wells in the South China Sea where lost circulation is severe. The ANN model is then evaluated by four metrics: accuracy, precision, f1 score ,and recall and concluded the model performed well enough to be applied to other fields if required data are available.

Chapter 4

Big Data Analytics

4.1 Big Data

Big data is a term that refers to datasets that are massive and complex and could be both structured and unstructured. The datasets are rapidly generated and transmitted from a wide variety of sources while it's difficult or impossible to store or process them efficiently using traditional methods. Examples of big data include Amazon.com's product list, New York Stock Exchange, and social media databases like the database of Facebook user profiles. Big data is a big deal for industries and companies as businesses and organizations are constantly struggling with making better decisions, reducing their costs, and understanding the value of their products and services which helps in adapting or redeveloping them if something goes wrong. The use of big data allows industries and companies to have transparent and simple insight into their challenges and new growth opportunities by uncovering market trends, hidden patterns, customer preferences to divulge valuable information to outperform their peers.[68] Big data is essentially characterized by four **V**s to gain insights and make predictions: [69]

1. **Volume** Data are collected from different sources like business transactions, social media platforms, networks, human interactions, and application logs. The flow of data is massive and continuous.

2. **Velocity** The speed of generation of data is fast and data must be dealt with in a timely manner. To mine data for potential insights, decision-makers must have the capabilities to harness datasets in real-time or near real-time. For example, the ability to instantly process rig data can provide managers and crew with potentially life-saving information in hazardous events like kicks and blowouts.

3. **Variety** Data flows in with all types of formats. Working with unstructured and inconsistent data is an unbreakable part of this area with the rise of big data. The data comes in an array of forms including text, emails, audio, videos, photos and it does not fit easily into a straightforward and traditional model.

4. **Veracity** The degree of accuracy or truthfulness of data is regarded as veracity and mostly it is traced back to the source of data. The more sources are combined to have diversity and variety, the data quality and accuracy would be in jeopardy more.

Big data is not always about the amount of data that important, but it is about what organizations do with the data. organizations may choose to use all their big data or determine upfront which data is relevant. Either way, they are generally utilizing those activities that involve ‘making use’ of data to provide insights that lead to improved performance. Big Data analytics is the use of advanced analytic techniques that discover new information, identify patterns, and unearth unknown trends by which the overwhelming volume of disparate information becomes a simple, clear decision point.[70]

Big data analytics assists in oil and gas operations, exploration, and production sectors. The technology refers to a new method that can be employed to handle large datasets consist of sensor-generated data. With the recent advent of an ever-increasing number of small and energy-efficient sensors, the oil and gas industry is considered as an intensive field in terms of data analytics. It is experienced that the best way to prevent future problems is learning from the past. To support real-time decision-making, managers and experts can perform strategies by a combination of big data and advanced analytics to reduce the risk of drill problems by knowing and planning for the potential hazards that lie ahead.[71]

4.1.1 Benefits

These massive volumes of data can be used to address petroleum problems that would not have been able to be tackled before. Just like other industries, the oil and gas industry needs to understand which data is valuable. As it is mentioned before, one of the main goals of this thesis is being able to predict future performance based on historical results to prevent kicks. By analyzing the previous patterns in near real-time and early identifying anomalies that would impact drilling, undesired and hazardous drilling events like kicks and blowouts can be prevented or controlled.[72] There are some other related areas where analytics can improve drilling and completion operations:[73]

- Early identification of risk to the products/services, if there is any
- The industry can utilize external intelligence while taking decisions
- Better operational efficiency, accuracy and optimization
- Cost savings - Time reductions
- Performance forecasting

4.1.2 Challenges

The handling of big data is very complex. While big data holds a lot of promise, each decision-maker has to know what they are dealing with exactly. Cleaning and preparing big data is the most time-consuming and the least enjoyable data science task among data scientists. [74]

- Principally, big data is "big". Most of the time, risk managers and other employees are overwhelmed with the amount of data that is collected. Although new technologies have been developed for data storage, managing thousands of interlocking datasets that takes place on a daily basis is a compelling task.
- After collecting and storing raw data, with so much data available, it is challenging to excavate data to access the most useful insights in a timely manner. In this stage, detecting anomalies and outliers and removing or changing them will aid in reducing challenges in this area. Anomalies detection is identifying the unusual, unexpected, surprising patterns or events which differ from the norm. Being able to detect abnormal and rare behavior in a set of observations not only helps in preventing hazardous situations but also maximizes the accuracy in big data analysis. Since meaningful data play a pivotal role in training and ML model creation, the presence of anomalies can cause serious issues during data mining.
- Data needs to be visually presented in graphs or charts for better interpretation, as the relationships of data that are too numerous or complicated are illustrated easily in less space. Pulling put information from multiple, disjointed sources and importing them into reporting tools is frustrating and time-consuming.
- Big data is not 100 percent accurate and nothing is more harmful to data analytics than inaccurate data. Good input directly affects the final result. When datasets are combined together with unstructured and inconsistent data from diverse

sources, real problems would arise. Asymmetrical data, missing data, inconsistent data, logic conflicts, and duplicates data all lead to significant negative consequences for data quality.

- Data analytics is getting harder as the amount of data grows at a rapid pace. processing power is expanded to accommodates rapid changes in the growth of data for creating reports in increasingly complex datasets. Scaling data allows the data science team to work together more effectively.
- Tight budget, data security, confusion and anxiety, shortage of skills, and insufficient understanding are some other challenges the risk managers are dealing with. Figure 4.1 shows how data scientists spend 50 to 80 percent of their time on data curation and data preparation before it can actually be used.[75]

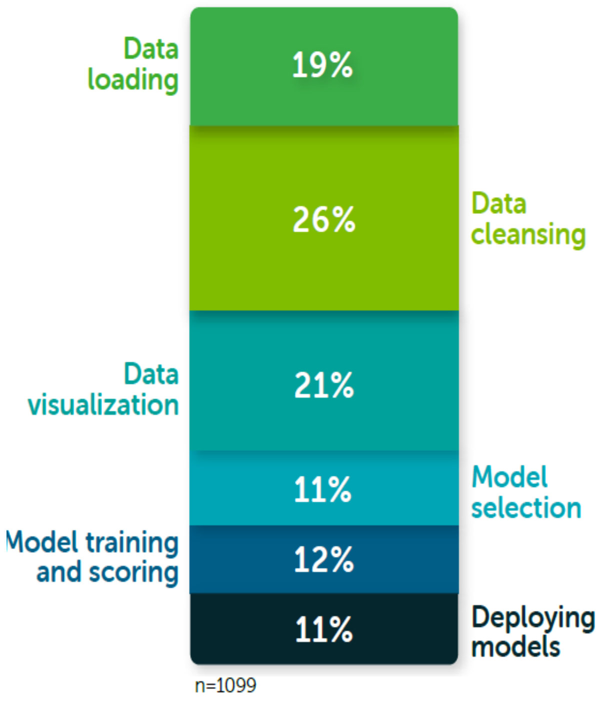


FIGURE 4.1: How data scientists spend their time [76]

4.2 Time-series Analysis and Forecasting

Time-series data is a sequence of time-based data obtained over time and often have equal time intervals between them. Time-series analysis helps to understand how past events influence the future. Consequently, in most of the fields including finance, economics, science, engineering, statistics, and public policy there is a large group of people who need to understand some basic concepts of time-series analysis and forecasting for managing their businesses. Time-series problems are more difficult to handle because of this "time" component that adds complexity to datasets. A proper time-series analysis, however, does provide an important contribution to more accurate forecasting. The skill of a time-series forecasting aids in determining a good forecasting model that best capture or describe an observed time-series in order to understand the underlying causes. The main aim of time-series forecasting is ultimately to estimate how the sequence of observations will continue into the future [77]. In the Oil industry, forecasting plays a major role in preventing unplanned future hazards or at least being aware of them to take precautions. Time-series forecasting is markedly leading to make positive impacts in the productions and mitigate the non-productive time.[78]

In drilling operations, the amount of data collected from the sensors of the well and on the rigsite has increased significantly. Rig crew and employees are constantly working with large volumes of data. Using advanced time-series analysis on the both high and low-frequency surface and downhole measurements provides drillers with active surveillance. On-time forecasting broadens the workers' vision and gives the driller advanced warning of the downhole abnormal conditions or rig equipment in presence of any failure. The difficulty of time-series analysis for drilling data stems from not being able to detect and analyze all the variables that cause a hazard in a real-time manner. The time-series model tries to extract all meaningful knowledge from input data to spot patterns and detect abnormalities and irregularities [79]. Regarding the aim of this thesis, in the ML model for kick prediction, multiple input variables are required in predicting the single output variable.

Time-series analysis can be either univariate or multivariate. In univariate forecasting, only one variable is varying over time, while in multivariate forecasting multiple variables are varying over time. For example, data collected from a sensor measuring the temperature of the downhole of a well every second provides a single observation recorder sequentially. Therefore, each second, response variable is influenced by only one factor and there is a one-dimensional value, which is the temperature. On the other hand, temperature, pressure, and WOB are three variables changing simultaneously over time as the drillstring goes deeper in a well. Multivariate time series data often have very high dimensionality. Classifying such high dimensional data poses a challenge because a vast number of features can be extracted over time and each of the parameters and elements influences the output result individually.[81]

"Time-series forecasting goes beyond 'just' time-series analysis. With time-series forecasting a model is being used to predict future values based on previously observed values over time."[82]

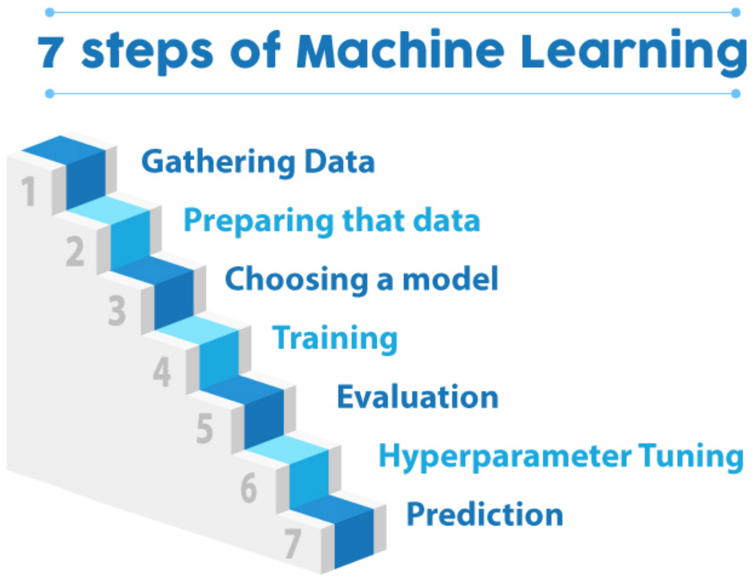


FIGURE 4.2: ML model steps [80]

ML is an area of high interest among tech enthusiasts, therefore it is mandatory to learn a programming language, preferably Python, along with the required analytical and mathematical knowledge to enter this area. As it is illustrated in Figure 4.2, the ML process is broken down to understand the steps and significance and function of each step. A more detailed representation of the ML model and involved steps is explained in the next chapters.

Chapter 5

Data collection and preparation

5.1 OpenLab Simulator

The drilling data for this paper was generated by the OpenLab Drilling simulator delivered by NORCE. The Drilling Well Modeling group of NORCE Energy has developed and managed this software in close collaboration with the University of Stavanger. This innovative and creative software consists of several software modules that implement specific functionalities within the system. OpenLab Drilling is the integration of the physical and virtual drilling and well operations intending to recreate the physical processes that take place during a drilling and well operation. The software uniqueness and novelty are of great importance to research environments and the business community who work with a large amount of data and seek methods to optimize the drilling process. It is not only useful for safer and more cost-effective drilling operations, but also it serves as an educational, developmental, and testing platform for students, lecturers, and researchers. [83] This chapter will give an overview of the simulator's key parameters and basic functionality in regards to drilling operations and influx simulation.

OpenLab comprises three systems, of which only the first one is finished and the rest are under development:

1. **Web-based drilling simulator**
2. **Drilling control room**
3. **Full-scale on-site operational drilling rig**

The OpenLab infrastructure has been developed with a high focus on user-friendliness and its interface offers a simple environment in which the user can either utilize numerous pre-made and optimized configurations and templates, with no need to change or can create new well configurations by modifying properties, parameters, design, and equipment. The templates can be modified later during the simulation according to the purpose of the simulation. Many of These factors have direct or indirect effects on the influx or loss in the well operations. After setting up the configurations, simulations run either in the web browser of OpenLab or through application programming interfaces (APIs) such as MATLAB or Python. The well configuration page consists of six tabs:[84]

- Hole Section
- Wellpath
- Fluid
- Drillstring
- Geology
- Rig

Moreover, simulations can be run in real-time, fast-forward, or sequence mode where the visual guide is provided for the end-users. Once the simulation starts, a sequence of inputs is supplied to a calculation module and in return, the state of the well is illustrated for every time step. The time step is 1 second in the physical application. "The results of the simulation are of two types: time-based or depth-based" [83]. All the setpoints and simulation results can be downloaded by users at the end of the simulation.

5.1.1 Openlab tabs

All the drilling parameters and elements are typed manually or imported by CSV files in the tables provided in each tab in order to simulate the well operation and drilling process. A quick start can be achieved by using pre-defined configurations and templates. Although the pre-made and optimized configuration is suitable for beginners, more experienced users can use editors to customize the design and properties.[84] [85]

5.1.1.1 Hole Section

The hole section consists of a riser, a set of casings and liners, and a definition of the open hole part. All wells have an open hole section primarily. It is the uncased part of a well that is exposed to instability and collapsing of formation into the wellbore. All numbers are measured depth in this section.(Figure 5.1)

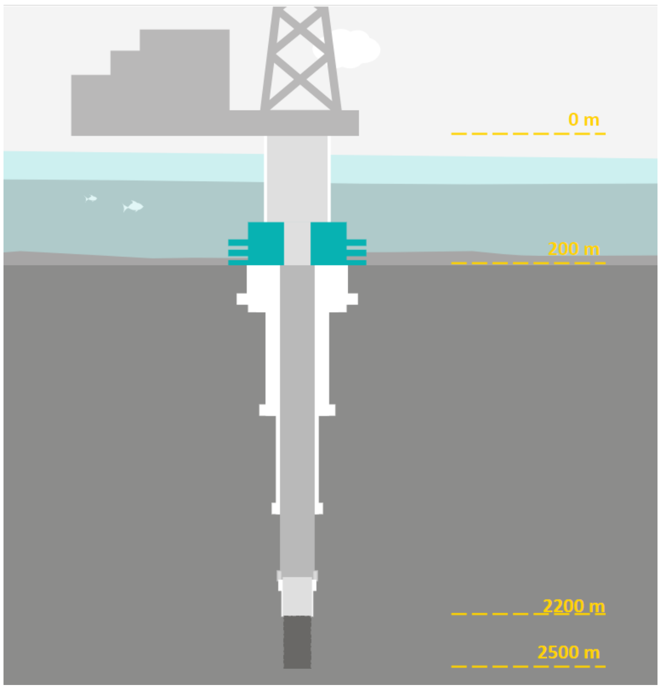


FIGURE 5.1: Hole section tab in OpenLab simulator [85]

5.1.1.2 Wellpath

As it is demonstrated in Figure 5.2, the visualization shows the wellpath structure in 3D. The wellpath can be completely customized on a meter by meter basis. It gives a comprehensive overview of the difference between the measured depth (MD) and true vertical depth (TVD). In addition, the inclination of wellpath creates a nonlinear relationship between MD and the pressure. From a simulation perspective, both drillstring mechanism and the cuttings transport are being affected by wellpath.

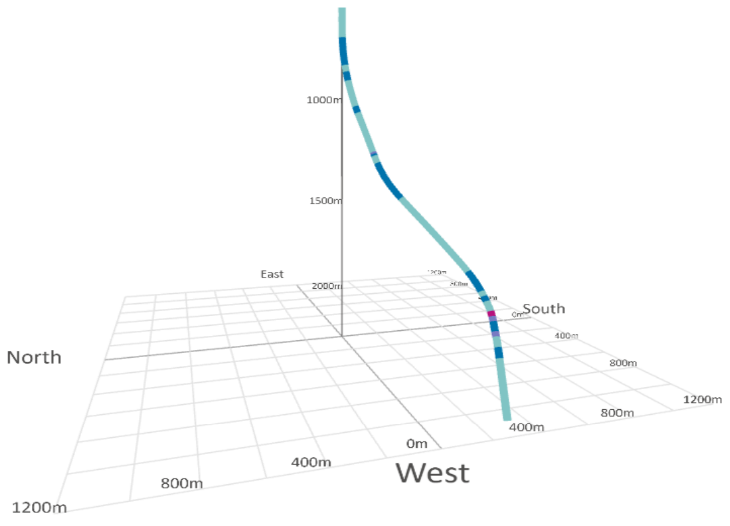
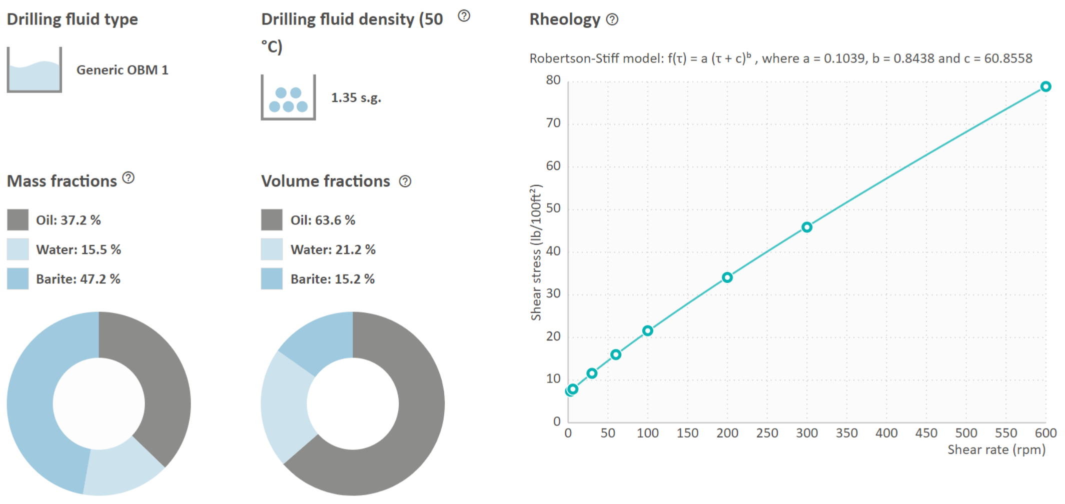


FIGURE 5.2: The visualization of wellpath tab structure in 3D [85]

5.1.1.3 Fluid (Mud)

Mud properties include mud type, density, oil-water ratio, gel strength, rheology and can be changed with a high degree of freedom. The mass fraction and volume fraction distribution are shown with pie charts, while a rheogram displays the flow behaviour of the fluid (Figure 5.3). Since the mud has a key role in stabilizing fragile formations, its properties must be assigned carefully. Thoroughly, the mud affects all aspects of the simulation.


Drilling fluid type

Generic OBM 1

Drilling fluid density (50 °C)

1.35 s.g.

Mass fractions

Oil: 37.2 %

Water: 15.5 %

Barite: 47.2 %

Volume fractions

Oil: 63.6 %

Water: 21.2 %

Barite: 15.2 %

Rheology

Robertson-Stiff model: $f(\tau) = a\,(\tau + c)^b$, where a = 0.1039, b = 0.8438 and c = 60.8558

FIGURE 5.3: The mud tab contains pie charts and rheogram chart [85]

5.1.1.4 Drillstring

The drillstring tab in OpenLab consists of drill pipes, bottom-hole assembly elements, and a drill bit (Figure 5.4). It is the OpenLab capability to variate the drill pipe inner and outer parameters. Different elements of the drillstring and their details are displayed by hover in the simulator.

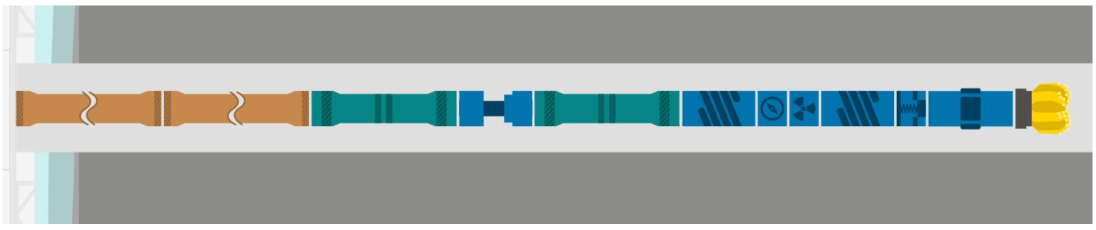
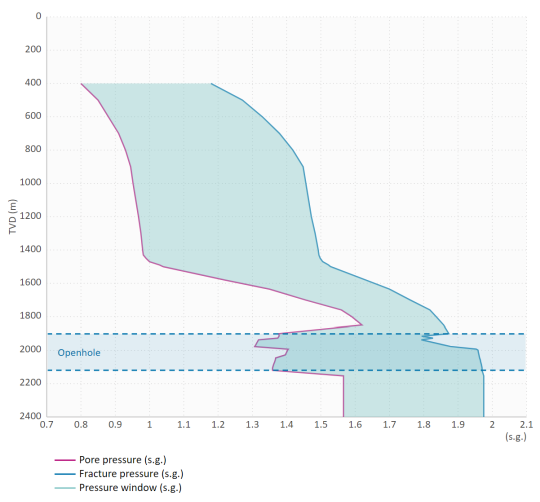


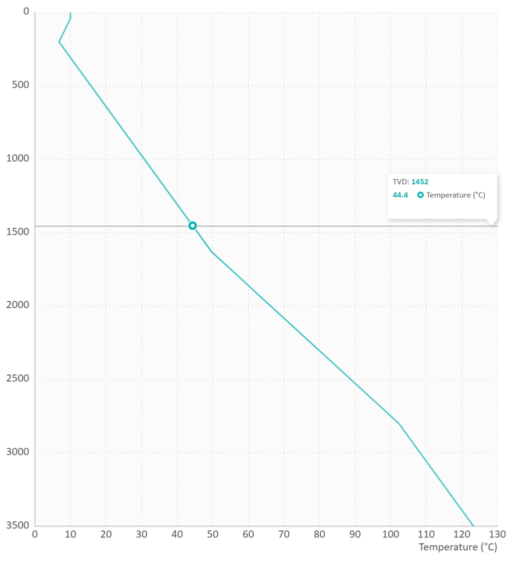
FIGURE 5.4: The drill pipe inner and outer parameters can be varied in drillstring tab [85]

5.1.1.5 Geology

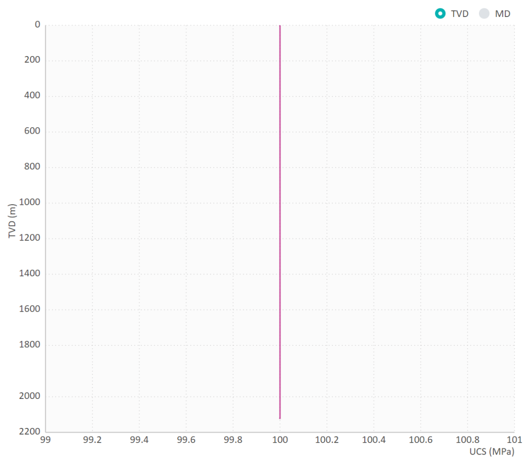
In the Geology tab, there are three subsections: Geopressures, Geothermal, and Formation (Figure 5.5). Due to the close margins between fracture gradient, mud weight, and pore pressure, a major threat is posed to support the wellbore walls for preventing influx and wellbore collapse during drilling. High-Pressure, High-Temperature (HPHT) wells mostly experience the simultaneous occurrence of losses and influx which can be simulated here [86]. A heavier mud pressure than the pore pressure results in wellbore stability in an open hole section and avoids fluid influx. Therefore, a pressure profile provides a great variety of influxes characteristics to create different cases. Also, as the temperature is important for several aspects of the well dynamic including the well pressure, a full thermal profile is essential to be set.



(A) Geopressure



(B) Geothermal



(C) Formation

FIGURE 5.5: Geology tab in OpenLab simulator [85]

5.1.1.6 Rig

The rig tab illustrated in Figure 5.6, displays a schematic of the rig layout. The rig parameters serve to guide towards a realistic drilling process by which operational limits can be implemented.

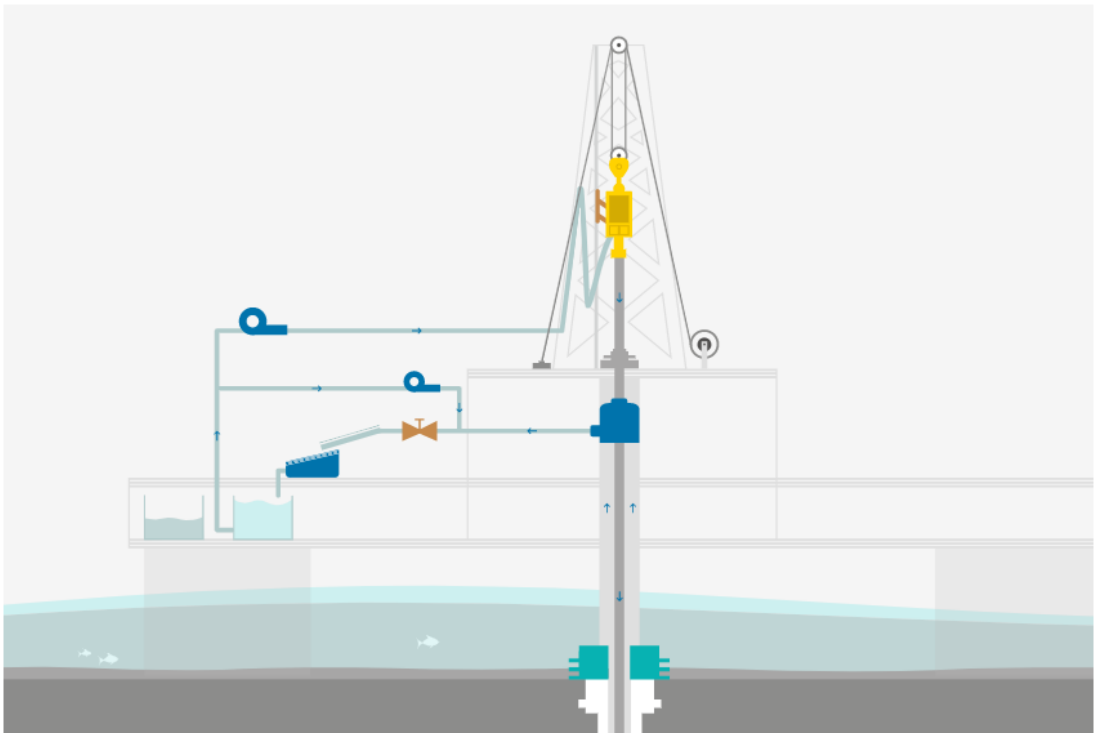


FIGURE 5.6: A schematic layout of the rig tab from OpenLab simulator [85]

5.1.2 Strength

The recent trend toward automation is a key driver in the drilling industry. Real-time analysis is a key driver in the drilling industry. The OpenLab software is a prime novel solution to build, test, and verify the physical and data-driven models. The OpenLab infrastructure came to life with the aim of facilitating education, research and innovation, and testing of new technology. Students, lecturers, and researchers are the main targets for OpenLab who are served with a user-friendly graphical interface and an easy-to-use API. Some of the strong points of OpenLab are summarized as:[85]

- Usable for both beginners and experts by means of intuitive interaction design and automated recommendations
- Enable multiple concurrent faster-than-real-time simulation runs

- Time/depth plots combined with interactive visualization data progress offers a simple environment for data management and data interpretation
- Cost-effective use of computational resources
- Secure handling of malicious inputs such as kick implementation

5.1.3 Weakness

Although the OpenLab application reshapes the way of how operations are done in the oil industry and gives a better understanding of the downhole physical processes, many challenges remain. This novel simulation technique is enabling manufacturers to benefit from optimization in top-drive controls and more effective drilling. However, unexpected things are always probable to happen in a rig. There are a lot of factors contributing to the safety of drilling operations that ignorance of them can lead to catastrophe in the real world. For example, as this work is mainly focused on kick and influx, there is no further applied noise or other data artifacts that are expected in a real drilling environment.[87]

Moreover, in order to have numerous simulated kicks for abnormal behaviour analysis, numerous individual simulations were run independently and inconsistent simulations cause inconsistent time steps in general. Since $\boldsymbol{time}$ is a vital component in time-series forecasting and $\boldsymbol{time}$ plays the most important role in the prediction of events through a sequence of time, the model would work better when the trends follow a consistent shape. Fortunately, this weakness is somehow solved by an introduction of a new Python package for time-series forecasting.

5.2 Understanding the problem

The main point to consider when trying to solve a new problem is defining the problem. This thesis tries to solve the problem that the oil and gas industry has been coping with for a fairly long time which is *kick identification and kick prediction*. With a set of high-quality data available and the defined objectives, it would be less complex to explore potential underlying patterns which are hidden in the data. For the process of learning, some observations or samples are required for the systems (computer systems) to be learned automatically without human intervention.

ML algorithms are usually categorized as supervised or unsupervised learning.[88]

1. Supervised learning algorithms: are employed where the observations and samples are labeled. The ML algorithm analyses the input data and learns a function to map the relationship between the input and output values. With supervised learning, the algorithm uses the input to make a prediction and compares the prediction against the expected output. Labeled dataset means, for each set of observations, an answer or solution is provided as well. The training data has output variables corresponding to the input variables in this category. Supervised learning can further be classified into various sub-group named Regression, Classification, Forecasting, and Anomaly Detection.

2. Unsupervised Learning algorithms: are used when the training data does not have a response variable. The major difference between supervised and unsupervised learning is that there is no complete and clean labeled dataset in unsupervised learning and no labels are given to the learning algorithm. Instead, the algorithm will be left on its own to discover information and try to find the intrinsic pattern and hidden structures in the data to do a grouping of data and make a comparison to guess the output. Unsupervised learning is great when you have an intensive amount of data and also a lot of computing power to find patterns in the data. There are various types of unsupervised learning among which Clustering and Dimension Reduction algorithms are of high importance.

Although each machine learning project is different because the specific data at the core of the project is different, all of the ML problem starts with data, and to be more clear, lots of data. The raw data must be pre-processed prior to being used to fit and evaluate an ML model, however, prior to this step, the first phase is to decide what you want to predict and what is the problem you are trying to solve. As it is discussed in 1.2, the main goal of this thesis is to evaluate how some series of events can lead to one of the most high-risk well issues named kick to be able to predict such events to reduce kick occurrence. In simple words, there is prior knowledge of what the output values for the samples should be. The better you know about the predictive problem, the more accurate you can collect and prepare the data for the ML model. Equation 5.1 shows how the expected outputs are classified into two groups. Data labeling is a key part of data preparation for ML because it specifies which parts of the data the model will learn from.

$$label = \begin{cases} 1, & \text{kick.} \\ 0, & \text{No kick.} \end{cases} \tag{5.1}$$

This work aims to exploit supervised learning as the problem is considered as a classification issue. By taking a close look at the data and exploring the data for which the label is already known, the classification problem belongs to the category of supervised learning where the input data predict the likelihood that subsequent data will fall into one of the predetermined categories. Using summary statistics and data visualization illustrate how the data looks like and what kind of correlation is held by the attributes of data. Input variables are columns in the dataset provided to the model to make a prediction and output variable is a column in the dataset to be predicted by the model.

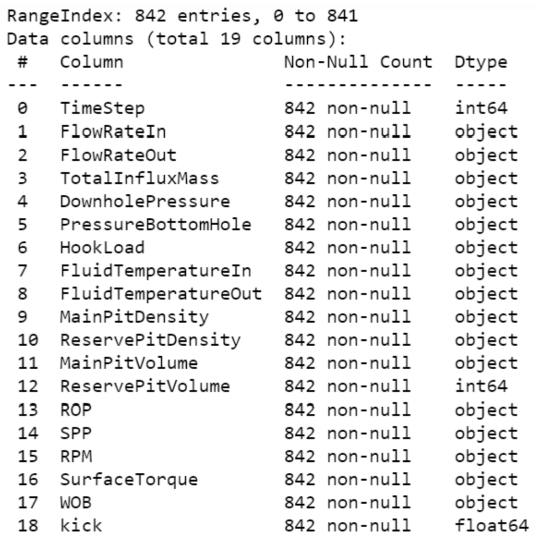
5.3 OpenLab Data

Time-series forecasting is an important area of ML that is often neglected. When it comes to forecasting time-series data, ML is an application of AI that strengths the ability to learn about patterns and anomalies and improves future prediction accuracy. Time-series forecasting is important because there are so many prediction problems that involve a time component. The purpose of this thesis is to create synthetic field data by the OpenLab drilling simulator, so that students, researchers, and industry experts can monitor and analyze trends by running simulations on virtual wellbores which are based on real well configurations. After determining the primary elements, the simulation starts. The plots are controlled during the entire simulation and the resulting data gathered in CSV files described later in this chapter. In this thesis, the main goal of using OpenLab is to design and implement a closed-loop well kick detection and circulation system to examine the kick attributes.

In this work, the synthetic data from OpenLab creates artificial datasets to analyze the characteristic of influx. According to an industry study in drilling, it is revealed that the accuracy and efficiency in drilling operations are commonly impaired by formation fluid influx and fluid losses. These events not only increase the non-productive time, but also cost the industry billions of dollars yearly. The main objective of forecasting in this industry is to leverage analytics to optimize output performance, improve production, make money, prevent hazards, and save lives and natural resources. High forecasting accuracy is not achievable without reliable and proper data. In the scope of reservoir simulation and forecasting, data quality has gained great attention. The issue of poor quality data results in both losing money and hindering organizations from exploiting their full potential. Data quality is a huge concern in time-series forecasting to the extent that this chapter is thoroughly allocated to data preparation for forecasting. As it is mentioned in section 4.1.2, although data scientists spend most of their time on data preparation and they view this task as the least enjoyable part of the work, the ML model would be useless without an explicit and precise data preparation.

5.3.1 Data Collection

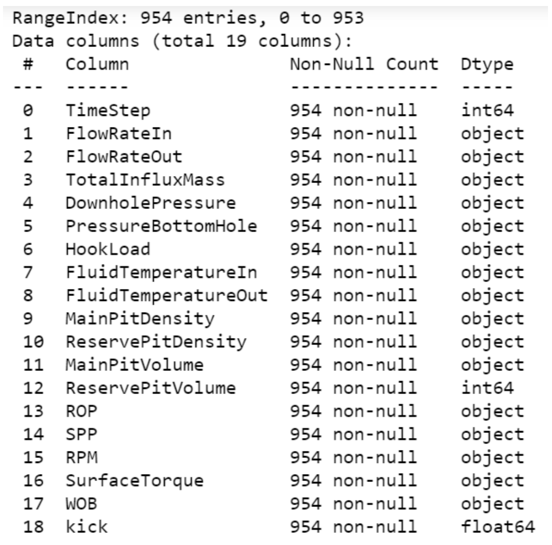
Before start digging into data, first, we need to know how the datasets are created. The initial drilling parameters are set manually in this work. Originally, the mud density is defined and once the simulation starts, the simulator gives time to -in this case- both flow and pressure to reach their steady state before the influx appears. Then, the occurrence of a kick would be applied by manipulating the flow rate. Alternating the flow rate will either cause a kick or suppress a kick, meanwhile, the mud density stays constant during the whole simulation. The same scenario is repeated for other simulations with different mud weights to study the behavior of parameters and evaluate the effectiveness of each indicator in identifying the influx. The resulting effects on all measurements are easily observable via plots in both time-based and depth-based visualization. After the simulation is done, the raw data from plots are imported into CSV files for further analysis. Since CSV format is portable, well understood, and ready for the predictive modeling process with no external dependencies, this is regarded as a standard practice to extract and save the data.



```
RangeIndex: 842 entries, 0 to 841
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   TimeStep             842 non-null    int64
 1   FlowRateIn           842 non-null    object
 2   FlowRateOut          842 non-null    object
 3   TotalInfluxMass      842 non-null    object
 4   DownholePressure     842 non-null    object
 5   PressureBottomHole   842 non-null    object
 6   HookLoad             842 non-null    object
 7   FluidTemperatureIn   842 non-null    object
 8   FluidTemperatureOut  842 non-null    object
 9   MainPitDensity       842 non-null    object
 10  ReservePitDensity    842 non-null    object
 11  MainPitVolume        842 non-null    object
 12  ReservePitVolume     842 non-null    int64
 13  ROP                  842 non-null    object
 14  SPP                  842 non-null    object
 15  RPM                  842 non-null    object
 16  SurfaceTorque        842 non-null    object
 17  WOB                  842 non-null    object
 18  kick                 842 non-null    float64
```

FIGURE 5.7: The first simulation sensor data and duration

Once the data created by OpenLab sensors, the data is primarily imported from OpenLab simulator into CSV files for further analysis. Each CSV file represents a different type of sensor data in columns and TimeStep in rows. Then these CSV files are integrated into one dataframe showing a single simulation. For example, Figure 5.7 and Figure 5.8 illustrate sensors for the first and second simulations respectively. While a



```
RangeIndex: 954 entries, 0 to 953
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   TimeStep              954 non-null    int64
 1   FlowRateIn            954 non-null    object
 2   FlowRateOut           954 non-null    object
 3   TotalInfluxMass       954 non-null    object
 4   DownholePressure      954 non-null    object
 5   PressureBottomHole    954 non-null    object
 6   HookLoad              954 non-null    object
 7   FluidTemperatureIn    954 non-null    object
 8   FluidTemperatureOut   954 non-null    object
 9   MainPitDensity        954 non-null    object
 10  ReservePitDensity     954 non-null    object
 11  MainPitVolume         954 non-null    object
 12  ReservePitVolume      954 non-null    int64
 13  ROP                   954 non-null    object
 14  SPP                   954 non-null    object
 15  RPM                   954 non-null    object
 16  SurfaceTorque         954 non-null    object
 17  WOB                   954 non-null    object
 18  kick                  954 non-null    float64
```

FIGURE 5.8: The second simulation sensor data and duration

simulation is running, alternating the flow rate could either cause a kick or suppress a kick. The ***kick*** column is the output label I am trying to accomplish *detection* and *prediction* about it [89]. The Snippet 5.1 converts the object data type to numeric since the ML mathematical models need numbers to work with.

SNIPPET 5.1: change the object type to numeric type for ML model

```
df = df.apply(pd.to_numeric)
```

5.3.2 Data Preparation

Knowing what you want to predict will help to decide which data may be more valuable to keep and which data are less useful. Data preparation also referred to as data preprocessing, is a significant step in the CRISP-DM discussed in 3.2 prior to ML modeling. Problems with ML modeling can stem from the way the data is prepared. Since predictive modeling involves learning from data, the more data you provide to the ML system, the faster that model can learn and improve. Aside from accuracy, data integrity also ensures deeper understanding and insights to manage a large amount of data. The likelihood of future outcomes based on historical data can be concluded by interpreting the insights and patterns hidden in the data. The ultimate task in the data preparation phase is the transformation of raw data exported from OpenLab into a form that is more suitable for ML modeling. Before feeding a dataset into an ML model, it is important to do some preprocessing to derive useful information from data which improves the ML model to learn. Actually, the foundation for trusted ML models could be ensured with proper data collection and preparation.

5.3.2.1 Data Cleansing

Based on the insights from data visualization, the data cleansing process can be applied to an entire dataset. During data exploration, there might be some incomplete or incorrect values. Although data cleansing may not be mentioned too often, it is a very critical step to avoid failing to yield ideal results. Data cleansing can be termed as a process of removing superfluous and repeated data records from raw data in order to feed the right data to machine learning algorithms. Cleaning of irrelevant and error-prone data will enhance the speed at which ML model trains [90]. Some of the best practices used for data cleaning in Machine Learning are:[91]

1. Filling missing values: impute zero for time-steps having missing values. (Snippet 5.2)

   SNIPPET 5.2: Filling missing values

   ```
   df.dropna(axis= 0 , how="all", thresh=None, subset=None,
       inplace=False)
   ```

2. Dealing with outliers and anomalies: these two terms are largely used in an interchangeable way and people occasionally argue that there is no difference between an outlier and an anomaly. The outlier is usually referred to as a single observation that lies far away from the mean or median in distribution, while anomaly is usually more than one observation and referred to data points that deviate from what is standard, normal, or expected. Regardless of the precise definition, the fundamental question is to include or exclude outliers/anomalies from a dataset, and the answer to it would be “it depends”.

   The primary difference between anomaly and outlier would be in the way of identification. Anomalies would be difficult to be spotted using visualization or detecting deviation from normal behaviour. Outliers on other hand would be merely extreme data points within the dataset that can be identified with basic statistical methods. Anomalous data can indicate abnormal incidents like a change in the normal behaviour of drilling operations. However, as the main goal of this thesis is to detect abnormal behaviour during drilling operations, Outliers can be very informative about the well control issues. Therefore, it is not always best to remove them from the dataset, because they might happen as a normal part of the study area.[92]

3. Dropping duplicates.

4. Dealing with columns that have a single unique value. Snippet 5.3 returns two columns named *ReservePitDensity* and *ReservePitVolume* which have unique

value in all simulations. Figure 5.9 shows that irrespective of these two columns, a kick would happen during drilling operation.

SNIPPET 5.3: Finding columns with unique value

```
for col in df.columns:
    if len(df[col].unique()) == 1 :
       print (col)
       df.drop(col, inplace=True, axis= 1 )
```

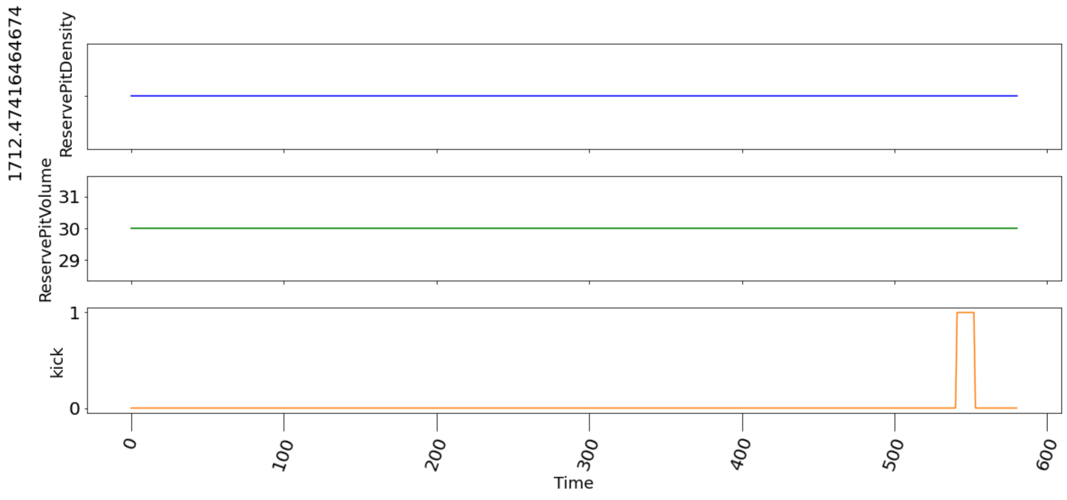


FIGURE 5.9: *ReservePitDensity* and *ReservePitVolume* columns have unique value in every simulation

5. Data reduction: this is helpful to reduce the data dimensionality as some of the parameters in a dataset are not going to make any contribution to the final result (Snippet 5.4). Moreover, working with extensive data causes several problems such as complex and time-consuming modeling. [93]

SNIPPET 5.4: Data reduction in order to solve the information overload problems

```
df = df[df.index % 2 == 0 ]
```

6. Dimensionality reduction by merging columns: reducing the number of input variables in a dataset has several advantages from an ML point of view as long as does not lead to some amount of data loss. This not only improves the interpretation of the parameters of the model but also reduces the time and storage space required. [93] Snippet 5.5 reduce the number of input by merging two variables *FlowRateIn* and *FlowRateOut* in one new feature named *DeltaFlow* as well as *FluidTemperatureIn* and *FluidTemperatureOut* in another new feature named *DeltaTemp*.

SNIPPET 5.5: Filling missing values

```
df['DeltaFlow'] =  abs(df['FlowRateIn'] - df['FlowRateOut'])
df['DeltaTemp'] =  abs(df['FluidTemperatureIn'] -
df['FluidTemperatureOut'])
```

5.3.2.2 Data Transformation

The process of changing the format, structure, or values of raw data into meaningful formats to be ready for analysis. Once the target column is set, the rest columns in the dataset are all the inputs. As it is shown in Figure 5.10, having more than one time-dependent variable as inputs, identifies this case as a *Multivariate Time-series* problem. The target variable is the feature of a dataset about which I want to gain a deeper understanding and the input variables are time-concurred series which depend not only on its past values but also have some dependency on other input variables as well.

Data Visualization plays an essential part in every step of ML in order to pinpoint the right direction to take. It can be very enlightening to plot the distributions of the numeric features or plot each column of the dataset in relation to the target column to achieve excellent exploratory data analysis. The advancement in data visualization empowers ML model creation.

There are two main data types from an ML perspective. All ML models are some kinds of mathematical models that need numbers to work with. Therefore, categorical data must be encoded to numbers before fitting and evaluating a model. The Snippet 5.6 depicts how numerical features are distinguished from categorical features.

SNIPPET 5.6: There are two data types in ML modeling: 1.Numerical 2.Categorical

```
categorical_features = list(set(df.columns)-set(df.
    _get_numeric_data().columns))

numerical_features = list(df._get_numeric_data().columns)
```

Standardization and Normalization are scaling techniques that come to the picture when input features of the dataset have varying scales. It is normal to be encountered different types of variables measured in different measurement units in the same dataset. Since variables that are measured at different scales may put different weights on the final model, feature scaling techniques modify features to lie between a given minimum and maximum value and lead the variables to contribute equally to the analysis so as to refuse to end up creating a bias. Transforming the data to comparable scales prevent

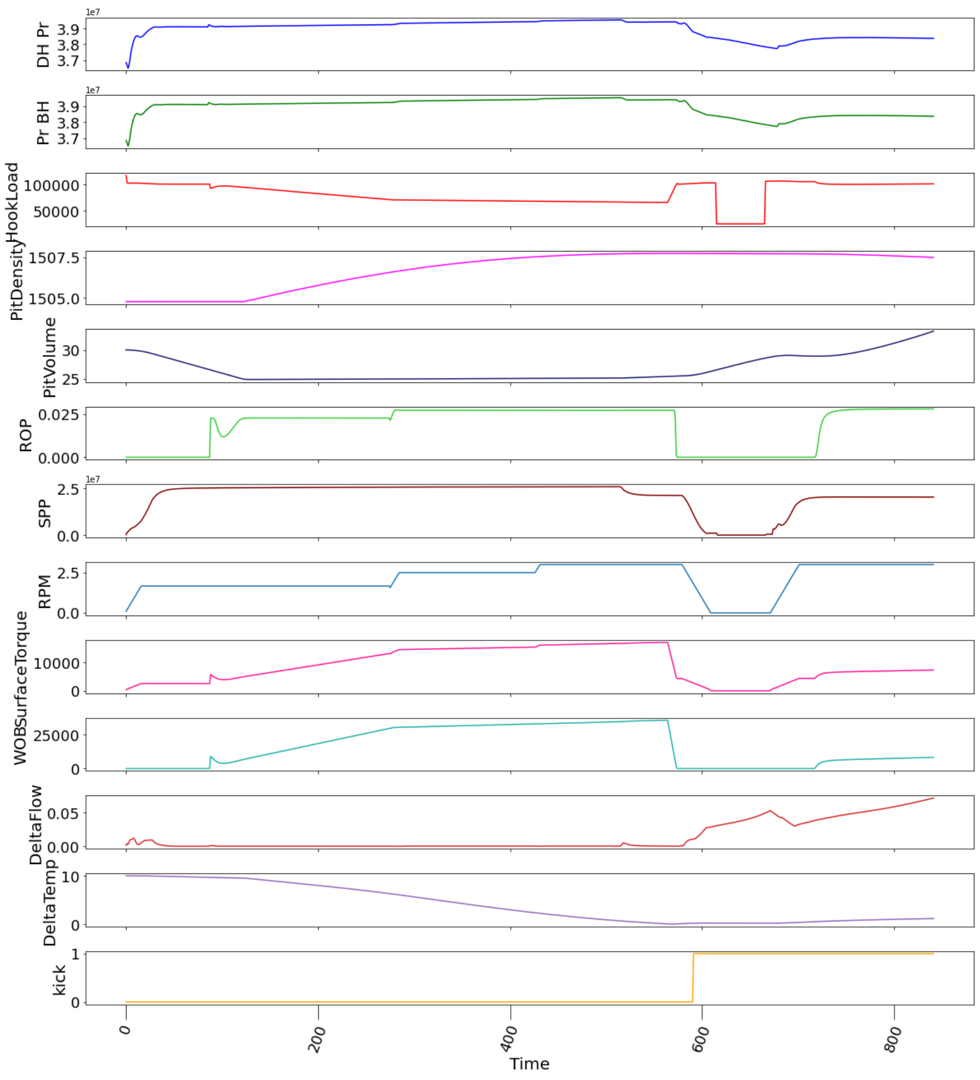


FIGURE 5.10: The kick as output variable and other data as input variables show this case is a *Multivariate Time-series* problem. The plots demonstrate how the sensor readings are changing over time (these plots refer to the first simulation)

variables with a larger range to start dominating over other variables and impacting the final results. Although there is no correct answer to when to use normalization over standardization and vice-versa, it is recommended to use normalization when the distribution is not Gaussian (a bell curve).[91]

SNIPPET 5.7: normalization as a scaling technique

```
scaler = MinMaxScaler(feature_range=( 0 , 1 ))
#kick and TimeStep are excluded
cols = df.columns.difference(['TimeStep','kick'])
df[cols] = scaler.fit_transform(df[cols])
```

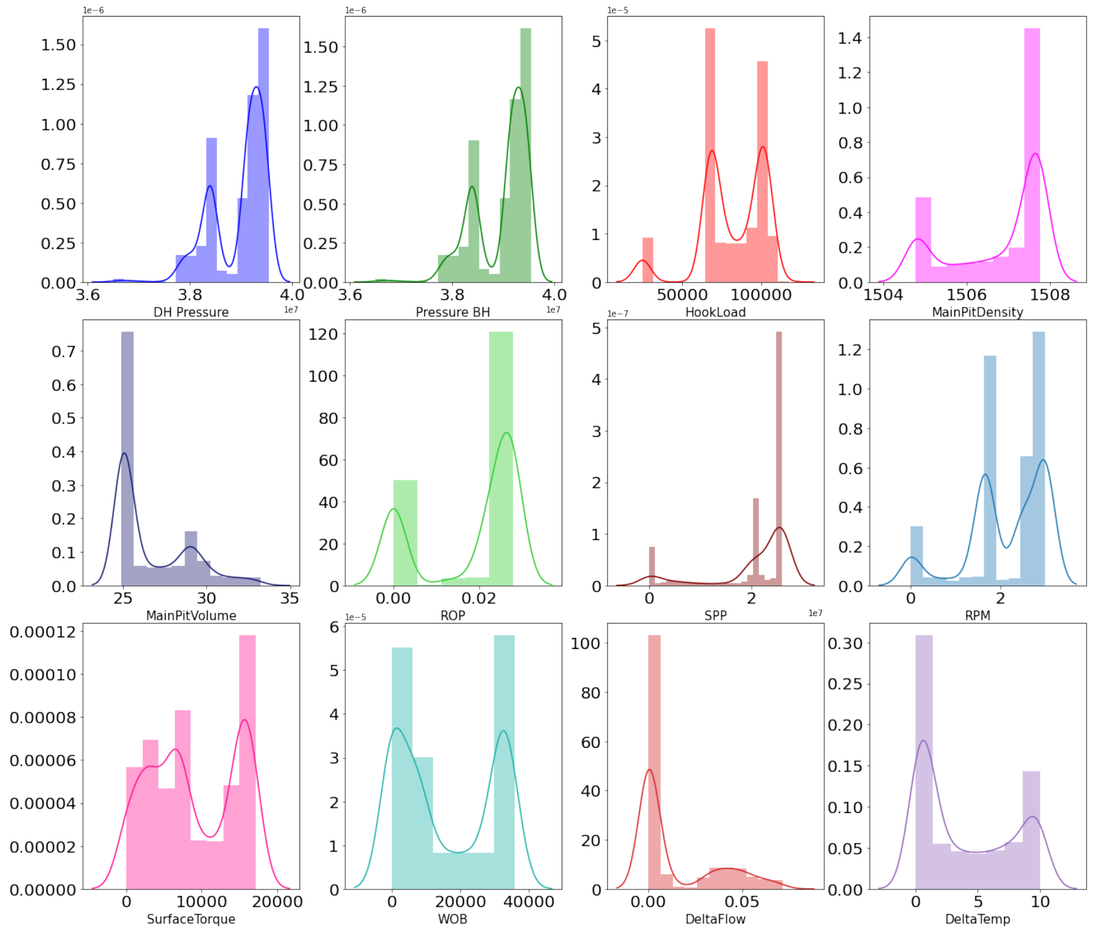


FIGURE 5.11: The input data distribution is not Gaussian(these plots refer to the first simulation)

The Snippet 5.7 applies normalization to the dataset for each simulation as the distribution is not Gaussian according to Figure 5.11.

All the above-mentioned steps regarding the data preparation must be applied for all the simulations to have a consistent dataframe for further analysis.

5.3.3 Feature Engineering and Feature Selection

One of the significant steps in data preparation for predictive modeling is the process of selecting a subset of the most relevant features from raw data. Feature engineering is transforming raw data into features that are more compatible with the machine learning algorithm requirements. Relying on the domain knowledge, feature engineering turns the excessive input data into useful features which improve performance and accuracy.[94] While feature engineering focuses on creating new features to make the most advantage of the previous data, feature selection keeps a subset of the original data and removes unimportant, redundant, or outright counterproductive to learning features. This step can be more important than the actual modeling because training a machine learning model for a prediction problem and forecasting results are highly dependent on how individual input features may correlate with the output. The better the features are prepared and chosen, the more accurate forecasting is achieved.[95] However, the question is:

How is it possible to get the most out of data for the predictive modeling?

It can be very tough to make a reasonably good model for a time-series problem as it is difficult to keep up with the pace of time. The "time" component presents some unique difficulties when trying to implement a robust ML system. Since identifying unusual and anomalous time-series is becoming increasingly common for organizations to identify abnormal behaviors, particular emphasis is given to feature engineering for time-series datasets in order to make stronger ML models in order to facilitate firmer decisions and market predictions.[96] What makes time-series problems different from the traditional ML problems is the dependency between each successive data point with its past values.

5.3.3.1 Manual vs. Automated

In this study, a vast amount of data are generated in a fraction of a second. All time-series data is a sequence of numbers that are ordered by a time index. After brainstorming and testing features, it is time to decide what new features to create. After feature creation, it is necessary to check how the new features work with the proposed model by use of some evaluation metrics. The feature engineering process can be an iterative process that interplays with feature creation and model evaluation until the satisfaction of model evaluation is fulfilled (Figure 5.12). Obeying simple statistical calculations, some standard features that one might take into account would be:[97]

- Means, Maximum, Minimum
- Standard deviations, Variance, Median
- Skewness, Kurtosis and Higher order moments

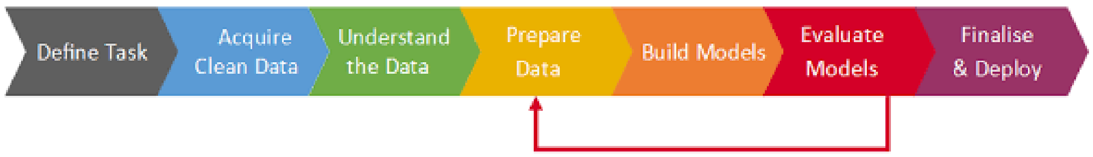


FIGURE 5.12: Once the model is built, it is possible to come back for further feature engineering to see if the performance can be improved. [98]

Due to some limitations discussed in 5.1.3, multiple simulations are run for accomplishing this thesis and therefore, the data is spread across multiple tables and must be gathered into a single dataframe named *df* with rows containing the observations over time and features in the columns. Therefore, Snippet 5.8 will concatenate all the CSV files generated by multiple independent simulations. Before concatenating, all the steps regarding the data preparation must be taken to have a consistent dataframe for further analysis.

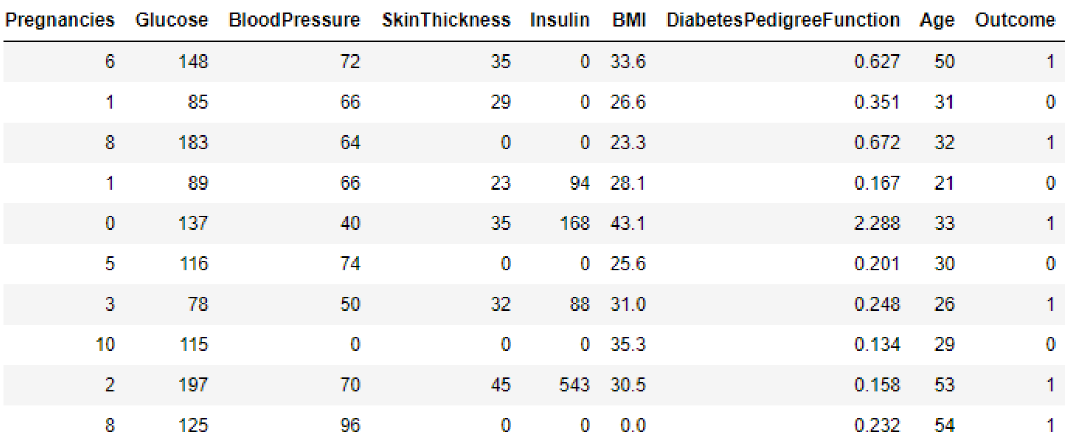
SNIPPET 5.8: Concatenate all individual simulations to have a unique dataframe

```python
import glob
import os
# use the path
path = r'C:\Users\maryam\PYTHON␣CODING'
# advisable to use os.path.join as this makes concatenation OS
    independent
all_files = glob.glob(os.path.join(path, "*.csv"))

df_from_each_file = (pd.read_csv(f) for f in all_files)
df_concatenated = pd.concat(df_from_each_file, ignore_index=True)
```

Furthermore, the desired features for ML are not only restricted to the above-mentioned simple statistical calculations. Instead, there are other mathematically complex and frequency-related features like Fourier Transform or Wavelet Transform or time-series related features like correlation coefficient and Euclidean Distance[97]. The traditional approach to engineer these complex features is not only a tedious and time-consuming task but also an error-prone process known as manual feature engineering. Human creativity and patience are obvious disadvantages of this manual method which can negatively affect the final result. Optimization the process of building and deploying an accurate ML model is not reasonably and computationally achievable unless automated feature engineering is applied.[99]

5.3.3.2 TSFRESH package

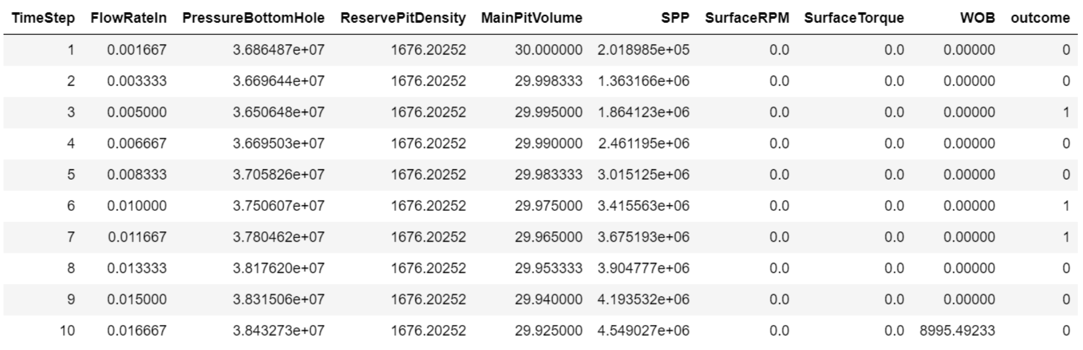
Since manual feature engineering is a tedious task and is limited by both human imagination and time restrictions, the promise of automated feature engineering is to surpass these limitations by automatically building hundreds of useful diverse new features using code that can be applied across all problems. This step can be more important than the actual model selection because an ML algorithm only learns from the data given to it. After creating many candidate features automatically, the best features can be then selected and used for training during the feature selection step. [97]



Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31.0	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0.0	0.232	54	1

TABLE 5.1: Diabetic Prediction as a sample of not time-series dataframe and it is time-independent

However, most ML algorithms are not time-aware and cannot be easily applied to time-series and forecasting problems. When a predictive algorithm is not time-aware, it typically looks at one row at a time when forming predictions as it is illustrated in Table 5.1). In contrast, in a time-series forecasting algorithm, as it is shown in Table 5.2), there is a time component that expresses the dependencies between observations. It is crucial to derive informative features based on past and present data in time. Forecasting is one of the hardest problems in predictive analytics because it is not always straightforward which input parameters are capable to explain the future output value. Furthermore, it is also tricky to define the number of consecutive observations per rolling window. The rolling window specifies how much recent history is required in order to make new predictions. In time-series forecasting, the time dimension itself adds an explicit order dependence among observations. All prior observations in a specific window are almost always treated equally.[100][101]



TimeStep	FlowRateIn	PressureBottomHole	ReservePitDensity	MainPitVolume	SPP	SurfaceRPM	SurfaceTorque	WOB	outcome
1	0.001667	3.686487e+07	1676.20252	30.000000	2.018985e+05	0.0	0.0	0.00000	0
2	0.003333	3.669644e+07	1676.20252	29.998333	1.363166e+06	0.0	0.0	0.00000	0
3	0.005000	3.650648e+07	1676.20252	29.995000	1.864123e+06	0.0	0.0	0.00000	1
4	0.006667	3.669503e+07	1676.20252	29.990000	2.461195e+06	0.0	0.0	0.00000	0
5	0.008333	3.705826e+07	1676.20252	29.983333	3.015125e+06	0.0	0.0	0.00000	0
6	0.010000	3.750607e+07	1676.20252	29.975000	3.415563e+06	0.0	0.0	0.00000	1
7	0.011667	3.780462e+07	1676.20252	29.965000	3.675193e+06	0.0	0.0	0.00000	1
8	0.013333	3.817620e+07	1676.20252	29.953333	3.904777e+06	0.0	0.0	0.00000	0
9	0.015000	3.831506e+07	1676.20252	29.940000	4.193532e+06	0.0	0.0	0.00000	0
10	0.016667	3.843273e+07	1676.20252	29.925000	4.549027e+06	0.0	0.0	8995.49233	0

TABLE 5.2: Kick Forecasting as a time-series dataframe that includes a time-related column

TSFRESH package (Time-Series FeatuRe Extraction on basis of Scalable Hypothesis tests) is an open-source Python library that automatically builds a large number of additional features out of existing time-series data [102]. There also exist other tools for time-independent data like *featuretools* package. In this work, I am attempting to solve a classification issue that involves when a sequence of events ends up causing a kick with regards to both identification and prediction. The lack of input data on forecasting problems or any other type of ML problems would jeopardize producing viable results.[103] Feature engineering is implemented by *feature extraction* method that involves finding and creating features for time-series data that can help understand, explain and predict the target variable. Moreover, *TSFRESH* package has a built-in filtering procedure called *feature selection* that keeps powerful and important features for the regression or classification tasks at hand. There are two functions in this package named *extract_features()* and *select_features()* that reframe the dataframe into a proper format for training and testing phase.[97]

It is advisable to keep this part of coding separate from the ML model coding as it helps to make it more maintainable and reusable. *TSFRESH* provides 63 time-series characterization methods, which computes a total of 794 time-series features by use of the *feature_calculators* module as they are comprehensively listed in [97]. This module which is thoroughly described in [102] takes time-series data as input and calculates the values of the features. Correlation, deviation, coefficient, entropy, quantile, variance, kurtosis, permutation, linear trend, and skewness properties are some of the examples of time-series features and they facilitate to fit a time-series dataframe into a range of time-series models. For example, the feature named:

$$PressureBottomHole\_\_cwt\_coefficients\_\_widths\_(2, 5, 10, 20)\_\_coeff\_14\_\_w\_5$$

denotes the value of the features form *feature_calculators.cwt_coefficients()* for the time-series *PressureBottomHole* under parameter values of widths=(2, 5, 10, 20), coeff=14 and w=5 [97]. Therefore, calling *extract_features()* function by Snippet 5.9 calculates a comprehensive set of features automatically.

SNIPPET 5.9: Extract a comprehensive set of features on *df* dataframe

```
from tsfresh import extract_features
extract_features(df, column_id="id", column_sort="time",
    column_kind="kind", column_value="value")
```

However, what is the proper format of the dataframe before being fed into the *extract_features()*?

There are four important columns which are of special interest to *TSFRESH*:[97]

1. column_id: This mandatory column indicates which entities each time-series belong to and separates your time-series from each other and also is the complicated one.
2. column_sort: This mandatory column specifies the sorting of the time-series data. It is not necessary to have equidistant time steps or the same time scale for the different ids and/or kinds. If this column is omitted, the dataframe is assumed to be already sorted in ascending order.
3. column_value: This mandatory column contains the actual values of the time-series. This corresponds to the measured values of different sensors on the simulations.
4. column_kind: This column is not mandatory and is out of the scope of this thesis.

Since multiple simulations are utilized in this work, it is essential to separate them by the most complicated column which is **"*id*"**. Each simulation is regarded as a separate **"*entity*"** which contains multivariate time-series. So, before concatenating simulations to each other to have a final dataframe, an extra column will be added to individual dataframes with various **"*id*"** numbers. For example simulation 1 will have id 1 and simulation 2 will have id 2 and so on. Even if there was only one simulation available for forecasting, it would also be crucial to have the **"*id*"** column but in this case, with a single unique value (1,2,3, etc or A,B,C, etc). Therefore, having this column is mandatory in all kinds of dataframe. Then, the **"*TimeStep*"** column corresponds to the sorting aspect which indicates the data are in time order when exported from the simulator. The **"*TimeStep*"** starts from zero point for each simulation and continues for a different amount of time in each of them. The column **kick** specifies the label/target which needs to be predicted with ML modeling. Lastly, other columns contain the value of the different sensors used for gathering both surface data and down-hole data during drilling operations.

5.3.3.3 Rolling dataframe for time-series forecasting

Before diving into *extract_features()* function, there is one more vital and inevitable step which is described as follows.

In a multivariate time-series prediction issue, there are two or more variables observed at each time step. Multivariate time-series analysis considers simultaneously multiple time-series and it is more complex than univariate time-series prediction. The persistence forecast involves using a number of previous observations to predict the next time step by shifting a cut-out window over the sorted time-series data. On each shift step,

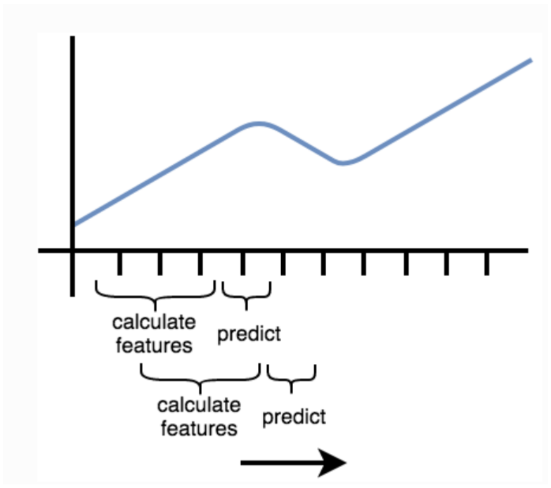


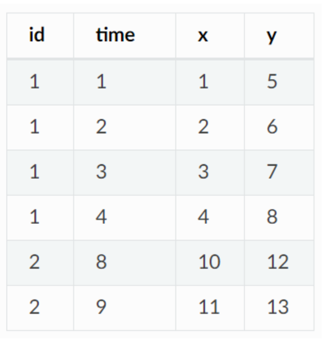
FIGURE 5.13: Rolling window in TSFRESH package [97]

the data is extracted through the cut-out window to build a new, smaller time-series and feature extraction is only done on this one. The use of prior time steps to predict the next time step is called the sliding window method or rolling window in which the chosen number ***n*** is the number of prior observations required for prediction and is called the window size.

SNIPPET 5.10: Rolling window in TSFRESH package

```
from tsfresh.utilities.dataframe_functions import roll_time_series
df_rolled = roll_time_series(df, column_id="id", column_sort="time"
    )
```

TSFRESH package handles the process of shifting a cut-out window over the datafarme to create smaller time-series cut-outs by use of $roll\_time\_series()$ function as depicted in Figure 5.13. The rolling process reshapes (and rolls) the dataframe into a form that can be fed into the usual *extract_features()* function. [97]

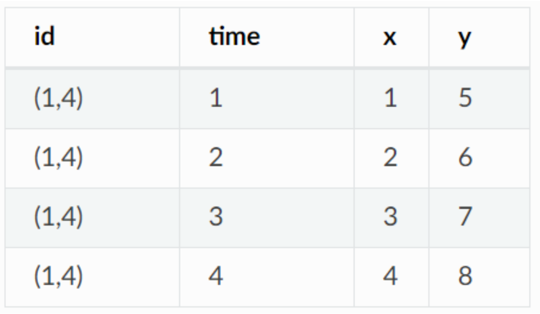


id	time	x	y
1	1	1	5
1	2	2	6
1	3	3	7
1	4	4	8
2	8	10	12
2	9	11	13

TABLE 5.3: Example of "id" column in a dataframe consists of two different entities (id 1 and 2) [97]

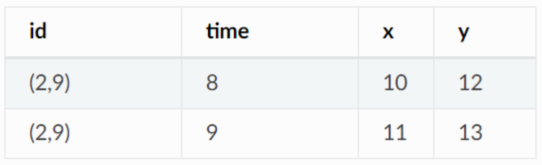
Snippet 5.10 shows how the rolling process is implemented in Python by use of $roll\_time\_series()$ function. This function automatically convert the dataframe to a new one with previous columns but with new format of **”*id*”** columns. A very simple dataframe(df) with the values from two sensors x and y for two different entities (id 1 and 2) in 4 and 2 time steps (1, 2, 3, 4, 8, 9) is shown in Table 5.3.

To get consecutive sub-time-series, Snippet 5.10 is applied and the new dataframe is exactly like the old one(df) except the **”*id*”** column. In this example, the features for the id=(1,4) are extracted using the data of id=1 up to and including t=4 (Table 5.4) and the features for the id=(2,9) are extracted using the data of id=2 up to and including t=9 (Table 5.5).



id	time	x	y
(1,4)	1	1	5
(1,4)	2	2	6
(1,4)	3	3	7
(1,4)	4	4	8

TABLE 5.4: Example of "id" after rolling [97]



id	time	x	y
(2,9)	8	10	12
(2,9)	9	11	13

TABLE 5.5: Example of "id" after rolling [97]

It is worth noting that “time” does not necessarily mean clock time here but could refer to any timestep instead. The “sort” column of a dataframe represents a sequential state to the property measurements. In the case of time series this could be the time dimension while in other cases, this would be a location, a frequency and etc. [97]

The size of expanding window grows as more observations are collected during the process for each **”*id*”**. There are two parameters useful for tuning the size of the window:

- $max\_timeshift$ defines how large the window is at maximum.(Snippet 5.11)
- $min\_timeshift$ defines how small the window is at minimum.(Snippet 5.12)

When the *max_timeshift* is set to ***n***, the current row plus ***n*** past rows are considered for feature engineering simultaneously. On the other hand, when the *min_timeshift* is set to ***n***, shorter time-series will be omitted from the beginning of the dataframe.

SNIPPET 5.11: Rolling window in TSFRESH package with *max_timeshift* parameter

```
df_rolled = roll_time_series(df, column_id="id", column_sort="time"
    , max_timeshift)
```

SNIPPET 5.12: Rolling window in TSFRESH package with *min_timeshift* parameter

```
df_rolled = roll_time_series(df, column_id="id", column_sort="time"
    , min_timeshift)
```

Now the dataframe is ready for the next steps. Eventually, Snippet 5.13 generates a comprehensive set of time-series features for each set of ***"id"*** in the rolled dataframe (*df_rolled*). If the *column_value* is not set to anything, the dataframe interprets both columns x and y as the actual values for *column_value*.

SNIPPET 5.13: Usual feature extraction on the rolled dataframe

```
from tsfresh import extract_features
df_features = extract_features(df_rolled, column_id="id",
    column_sort="time")
```

The process of identifying critical or influential variables regarding the target variable in the existing features set is feature selection. Snippet 5.14 evaluates the importance of the different extracted features. "For every feature, the influence on the target/label is evaluated by a univariate test and the p-Value calculation. Afterward, the Benjamini Hochberg procedure which is a multiple testing procedure decides which features to keep and which to cut off solely based on the p-values"[97]. The Benjamini and Hochberg step-up procedure (BH) was the first method proposed to control the False Discovery Rate (FDR).[104][105]

SNIPPET 5.14: Select the most relevent features out of all the features engineered for ML modeling

```
from tsfresh import select_features
df_selected = select_features(df_features, label)
```

Feature selection is primarily focused on removing non-informative or redundant features to reduce the number of input variables and dimensions. This process is mainly believed to improve the ML model accuracy in order to predict the target variable.

Chapter 6

Machine learning models selection

Although each ML project is unique, the steps on the path to a good or the best result are almost the same from project to project. ML approaches try to address some questions including what specific task should the model be automating and what information should be exposed to the user finally. The primary types of approaches in machine learning are supervised learning and unsupervised learning.

Supervised learning is where the presence of a supervisor as a teacher is evident. Supervised learning algorithms try to model the best relationship between input variables and output variables with the goal of finding the mapping function from input to output based on example input-output pairs. Supervised learning is a practical technique in which models are trained using labeled data which means data is already tagged with the correct answers. In other words, data labeling is a process of tagging data with one or more characteristics to provide context so that a machine learning model can learn from it. These meaningful and informative properties help to teach or train the model, so labeled data also called training data because they are used to generate the model. Supervised learning classified into two categories of algorithms:[106][107]

- Classification: A classification problem is when the output variable is a category, such as “happen” and “not happen” or “disease” and “no disease”.
- Regression: A regression problem is when the output variable is a real value, such as “price” or “weight”.

Unsupervised learning is another approach allowing the algorithm to act on that information without guidance and it relies on no human intervention at all. The model works on its own to discover information while it mainly deals with the unlabelled data and therefore, it performs more complex processing tasks. Even though there are no labels for data points, according to similarities, patterns, and differences, the algorithm attempts to find rules for groups of data points. Unsupervised learning itself includes clustering, dimensionality reduction, finding association rules, and anomaly detection.[108] Since this thesis mainly focuses on supervised learning due to the presence of labeled data, I refuse to dive deeper into the sub-categories of this approach.

6.1 Steps to choose the right ML model

As it is explained in the previous chapter, this thesis tries to exploit supervised learning since the problem is considered as a classification issue. Since the output data is labeled data in this work, it is categorized as a supervised learning problem. Moreover, the output of the model is binary classes (0 and 1), therefore it is a classification problem. In other words, by taking a close look at the data and exploring the data for which the label is already known, the classification problem belongs to the category of supervised learning where the model predicts the correct label for newly presented input data.

Time-series forecasting is an important area in many industries as most of the prediction problems involve a time component. Time-series forecasting involves selecting one or more than one model and then fit them on historical data. Afterward, it is time to use the model to predict future observations. There are many algorithms dedicated to time-series classification.

For picking the best algorithm among the shortlisted alternatives, testing the model on unseen data is a useful approach. Before building any model, to evaluate how well a classifier is performing, it is necessary to split the data into two parts:[109]

1. **Training set** is a subset of the samples used to build up predictive models. This subset is used to train and evaluate the model during the development stage.
2. **Test set** is the dataset excluding the training set. The trained model is now ready to make predictions on the unseen test set. This subset is used to assess the likely future performance of a model.

A typical train/test split would be to use 70% of the data for training and 30% of the data for testing. Depending on the amount of data available and the amount of data required, these percentages can be changed to 80-20 or 90-10 [110]. Snippet 6.1 depicts a function in Sklearn model selection that helps to split a dataframe into two subsets automatically.

SNIPPET 6.1: train test split

```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df, label,
    test_size=size, random_state= 1 , shuffle=False)
```

To have reasonable results from ML models, it is not only vital to feed in large quantities of data but also have to ensure the quality of data. With *extract_features()* function, the quantities aspect is guaranteed, and also with *select_features()* the qualities aspect is promised by keeping the informative and insightful data and removing the low-quality data in respect to the label variable. In time-series ML analysis, the observations are not independent, and thus it is not acceptable to split the data randomly and shuffle them as we do in non-time-series analysis. The default value is True for the shuffle object. Random_state can be 0 or 1 or any other integer. This object is initializing the internal random number generator which controls randomization during splitting. The default value is None for this object.

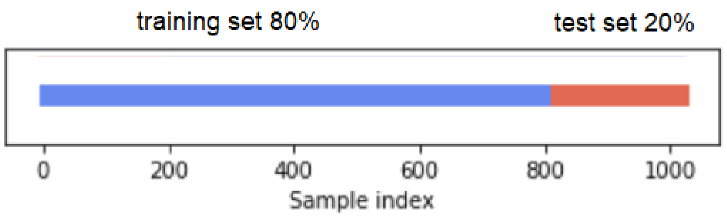


FIGURE 6.1: A common train/test split size [111]

A regular train/test split is shown in Figure 6.1. However, overfitting and underfitting are two modeling errors that lead to a negative impact on the performance of the model on new data.[112][113]

- overfitting happens when random error or noise instead of the underlying relationship. This error occurs when a model is excessively complex, such as having too many parameters relative to the number of observations, or when a function corresponds too closely to a particular set of data.
- underfitting happens when a model cannot capture the underlying trend of the data. This error refers to a model that can neither model the training data nor generalize to new data. Poor predictive performance is caused by Underfitting and it occurs when a model is too simple or informed by too few features.

SNIPPET 6.2: Cross validation

```
from sklearn.model_selection import KFold
cv = KFold(n_splits=n, random_state=None, shuffle=False)
for train_index, test_index in ts_cv.split(X):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

Cross-validation is a useful statistical method for improving the evaluation of ML models especially in cases where we need to mitigate the overfitting problem. By training several ML models on subsets of the available input data, each time, ample data is provided for training the model, and also ample data is left for testing. In k-fold cross-validation (Snippet 6.2 [111]), the input data is split into k subsets of data (also known as folds). Then the model is trained on all folds except one and a test is done on the remaining fold.[114]

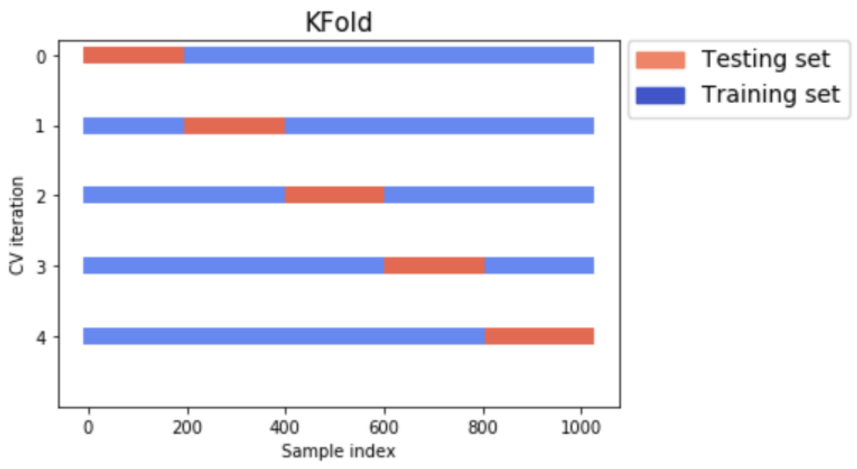


FIGURE 6.2: In KFold cross validation the size of training and testing remain unchanged in each split [115]

As it is illustrated in Figure 6.2, these steps will be repeated until the model is being tested on each of the folds. Ultimately, the final performance metrics will be the average of scores obtained in every fold to get the total effectiveness of our model. The number of folds in k-fold cross-validation is usually determined by the number of instances contained in the dataset [116]. However, the number of folds must be chosen in a way that the simulations do not be divided somewhere in the middle. Using 4-fold cross-validation means that 25% of the data is used for testing and this is usually pretty accurate. However, if the dataset size increases dramatically, like having more than 100,000 instances, it can be concluded that 10-fold cross-validation would lead to folds of sufficient yet proper length.

Although the most admitted technique toward building an ML model consists of randomly picking samples out of the available data and split them into train and test sets, in the case of time-series, the cross-validation is not trivial in its random foundation [117]. As it is discussed 5.1.3, due to some limitations in the OpenLab simulator, the dataset in this work is not a time-series simulation at a time but is a multiple concatenated time-series simulation.

"Since Shuffling reorders the observation occurrence order, the idea of shuffling the whole dataset completely ruins the time-based analysis. Because it makes no sense to use the values from the future to forecast values in the past. However, shuffling the order of independent dataset preserve the time-based relations among observations while the sequenced values inside each of the dataset remain unchanged."[115]

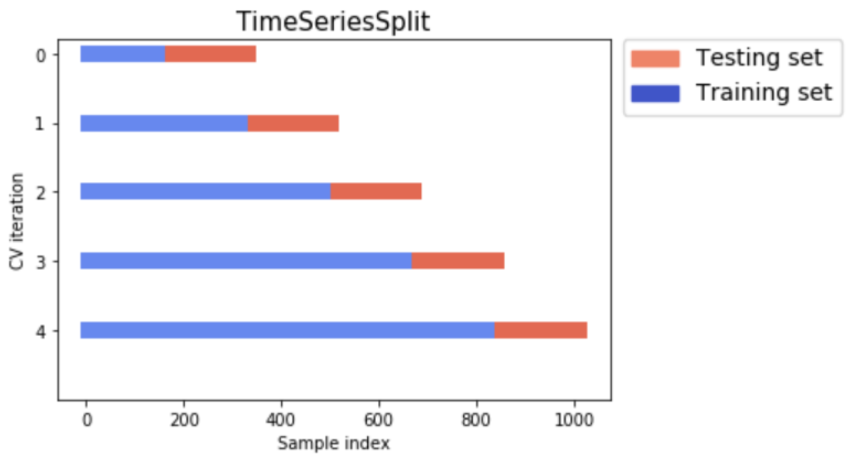


FIGURE 6.3: In timeseriesSplit, each split, the training size is getting bigger and successive training sets [115]

"It is not possible to choose the set sizes in the KFold method, but it is merely possible to choose the number of splits one would like to have. In this thesis, the k-fold cross-validation can be utilized if only the number of folds is chosen in a way that it splits the concatenated simulations but not the sequenced values inside each of the simulations."[115]

Another method that can be used for cross-validating the time-series model is cross-validation on a rolling basis. This cross-validation object (shown in Figure 6.3), is a variation of the KFold method but in each split, test indices are higher than before. This approach which is well-known in the time-series domain avoids future-looking when training the model. Thus, time-based splitting provides a statistically robust model evaluation and best representative of real-life scenarios. Snippet 6.3 shows a time-based cross-validation.[111]

SNIPPET 6.3: time-series train test split

```
from sklearn.model_selection import TimeSeriesSplit
ts_cv = TimeSeriesSplit(n_splits=n)
for train_index, test_index in ts_cv.split(X):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

6.2 How to evaluate the chosen ML model

ML algorithms or models can give different results due to either they are given different input datasets or the nature of the learning algorithm. There is no one approach or one solution that caters to all problems. How to choose the right ML model highly depends on the candidate model assessment. In order to choose the best model, there are many factors like speed or training time, number of features, type and size of data, and common metrics that narrow down the list of ML algorithm options. However, for classification problems, some performance metrics are mostly used to evaluate different ML algorithms like accuracy, recall, precision, and F1 score.

Supervised learning is of two types named classification and regression as it is described at the beginning of this chapter. The pool of accuracy and error metrics to choose from is different between classification and regression problems. In the latter, the goal is to predict one continuous value, and with classification, the goal is to predict discrete classes such as "kick" or "not kick". Although there is not one true accuracy or error measurement and they all have their strength and weaknesses, some measurements might distract us from the right path.

For example, since the labels for the two groups are arbitrary, the classification rate is more meaningful in this case than Mean Square Error (MSE) and MSE, so these metrics ought to be ignored. MSE or other error measurements like Root Mean Square Error (RMSE) tells how far your classifications are from the true values. Since the values assigned for groups are arbitrary, there is no sense of how far or close a classification is from the true value. In contrast, it is about either the output is correctly classified or it is not and the classification rate measures this properly.[118]

Whenever a model is built, there are several metrics that are helpful to figure out how well the model has performed. The Best metrics to measure and evaluate the performance of classification models are as following: [119]

1. Confusion Matrix: This is a popular way to represent the summarized useful findings. It is a kind of table (Table 6.1) that contains performance measurement metrics for ML classification. The binary classification has four possible types of results that the definition of the terms are:

   TP True Positive

   It is predicted positive and it is true. For example, it is predicted a kick happens and it actually happens.

		Predicted Value	
		Positive	**Negative**
Actual Value	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

TABLE 6.1: Confusion Matrix

TN True Negative
It is predicted negative and it is true. For example, it is predicted a kick does not happen and it actually does not happen.

FP False Positive
It is predicted positive, but it is false. For example, it is predicted a kick happens, but it actually does not happen.

FN False Negative
It is predicted negative, but it is false. For example, it is predicted a kick does not happen, but it actually happens.

2. Accuracy: This is the ratio of all correct predictions to total predictions that have been made and most of the time it is presented as a percentage by multiplying the result by 100. Accuracy could be a useful lead when dealing with a balanced (or approximately balanced) dataset when the class distribution is similar.

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+FP+TN}$$

3. Recall and Precision: Recall (aka Sensitivity) attempts to answer out of all the actual positive classes, how many are predicted correctly and it should be high

as possible. The Sensitivity (True Positive Rate) of a test in this work refers to how well a test identifies a kick that does happen in a drilling operation. Precision calculates what proportion of positive identifications was actually correct. Precision and Recall are two extremely important model evaluation metrics that are useful measures of success of prediction when the classes are very imbalanced.

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

4. F1 Score: F1 score (aka F score) is a way of combining the precision and recall of the model and calculates the weighted average of Precision and Recall. In a binary classification task, an F1 score fluctuates between 1 and zero where its best value at 1 and worst value at 0. The major difference between accuracy and F1 score is F1 Score might be a better measure to use if a balance between Precision and Recall is of importance and there is an uneven and imbalanced class distribution (a large number of Actual Negatives).

$$\text{F1} = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

5. Specificity: While the Sensitivity measure is used to determine the proportion of actual positive cases, which got predicted correctly, the Specificity (True Negative Rate) measure is used to determine the classifier's ability to identify actual negative cases, which got predicted correctly. The specificity of a test in this work refers to how well a test identifies a kick does not happen.

$$\text{Specificity} = \frac{TN}{FP + TN}$$

6. ROC Curves and AUC: The AUC-ROC curve is only restricted for binary classification problems and is a proper way to see how any predictive model can distinguish between the true positives and negatives. ROC stands for Receiver Operating Characteristic. This curve, as it is illustrated in Figure 6.4, does this by plotting sensitivity against 1-specificity to not only predict a positive as a positive correctly but also a negative as a negative. The area under the curve (AUC) is usually in the range [0.5,1] and shows a summary of the ROC curve. "A high AUC represents both high recall and high precision, where high precision relates to a low False Positive rate, and high recall relates to a low False Negative rate" [115]. The AUC score smaller than 0.5 indicates that a classifier performs worse than a random classifier. [120] snippet 6.4 contains the main packages for ROC-AUC measurement.

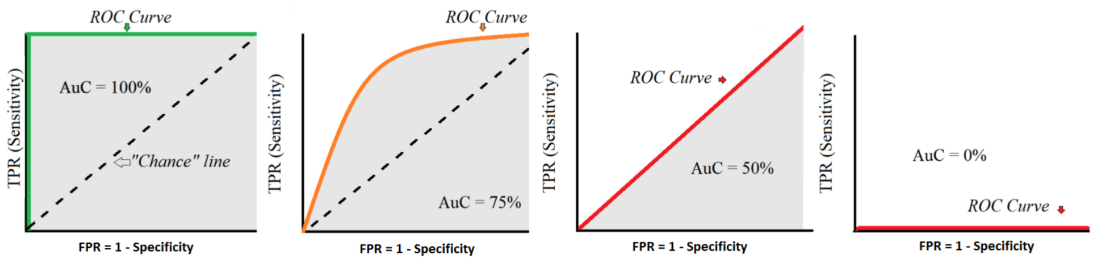


FIGURE 6.4: ROC VS AUC [121]

SNIPPET 6.4: Useful packages for ROC curve and AUC score

```
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score
from sklearn.metrics import precision_recall_curve
```

Chapter 7

Models implementation and evaluation

Models are fundamentally mathematical algorithms while Modeling in ML is an iterative phase where a data scientist continually trains and tests ML models to discover the best one for the given task. Among all the Popular algorithms that can be used for binary classification, in this thesis, **RandomForest**, **DecisionTree**, **K-Nearest Neighbors (KNN)**, and **Long Short-Term Memory (LSTM)** are built and compared to check how fit they are to solve the problem of *kick identification and kick prediction* during the drilling operation.

Now the data is in its usable shape through data preparation in chapter 5 and also the models are selected according to the business objectives through chapter 6. This chapter aims to estimate the accuracy and precision of a model on future and unseen data. Separating data into training and testing sets is an important part of evaluating ML models in order to prevent the model from overfitting and to accurately evaluate the model. In a dataset, a training set is implemented to build up and fit a model, while a test set is to validate the model built by providing an unbiased evaluation of a final model to qualify performance. The train/test split proportion to be divided is completely up to the task we face and is quite specific to the use case.

7.1 Tools

Having known the data as well as the complexity of the problem, a variety of tools have been used in this work.

7.1.1 Hardware

The configuration of the hardware is: CPU: intel Core i5 8th gen – RAM: 8GB – Hard disk: 256GB SSD

7.1.2 Sensors

As it is comprehensively described in 5.1, the synthetic data is created with an OpenLab simulator to analyze the characteristic of influx during drilling operations. Each sensor in the simulator is responsible to capture relevant information for a particular task. Since the amount of data collected from the sensors increases rapidly, building an ML model primarily involves monitoring and correctly interpreting all the data collected from the simulator. Although real-world sensor data often contains noise, the synthetic data in this work did not experience a high noise level.

7.1.3 Programming language

In this thesis, all the algorithms and analyses were developed in Jupyter Notebook which is a free, open-source, interactive web tool. Jupyter is an easy-to-use, interactive data science environment that is the best coding language for data mining and analysis. Jupyter is a great interface to the Python programming language that contains many powerful libraries, ranging from basic statistics to complex machine learning algorithms. Some of the commonly used libraries in this work are as follows:

- Pandas, Numpy: These are grouped as basic libraries which are used for underlying data analysis.
- TSFRESH: This library is used for feature engineering (extraction) and feature selection for time-series data.
- scikit-learn is used to build ML models.
  - sklearn.metrics: used for evaluation
  - sklearn.preprocessing: used for scaling
- Keras, TensorFlow: These two are referred to as deep learning libraries and help to build models like LSTM.
- Seaborn, matplotlib: These two are known as Python's most powerful visualization libraries.

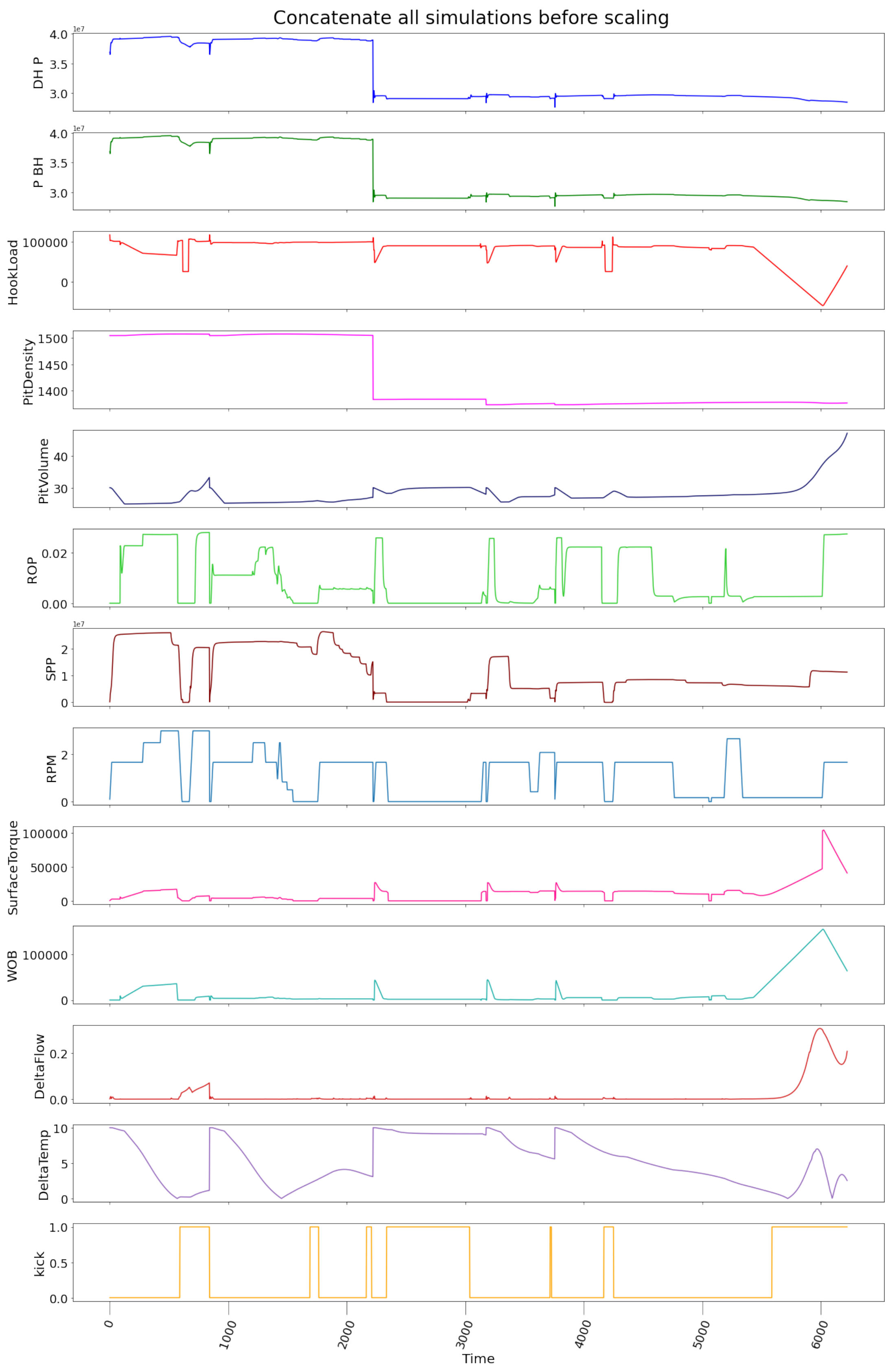


FIGURE 7.1: before scaling bigger font size

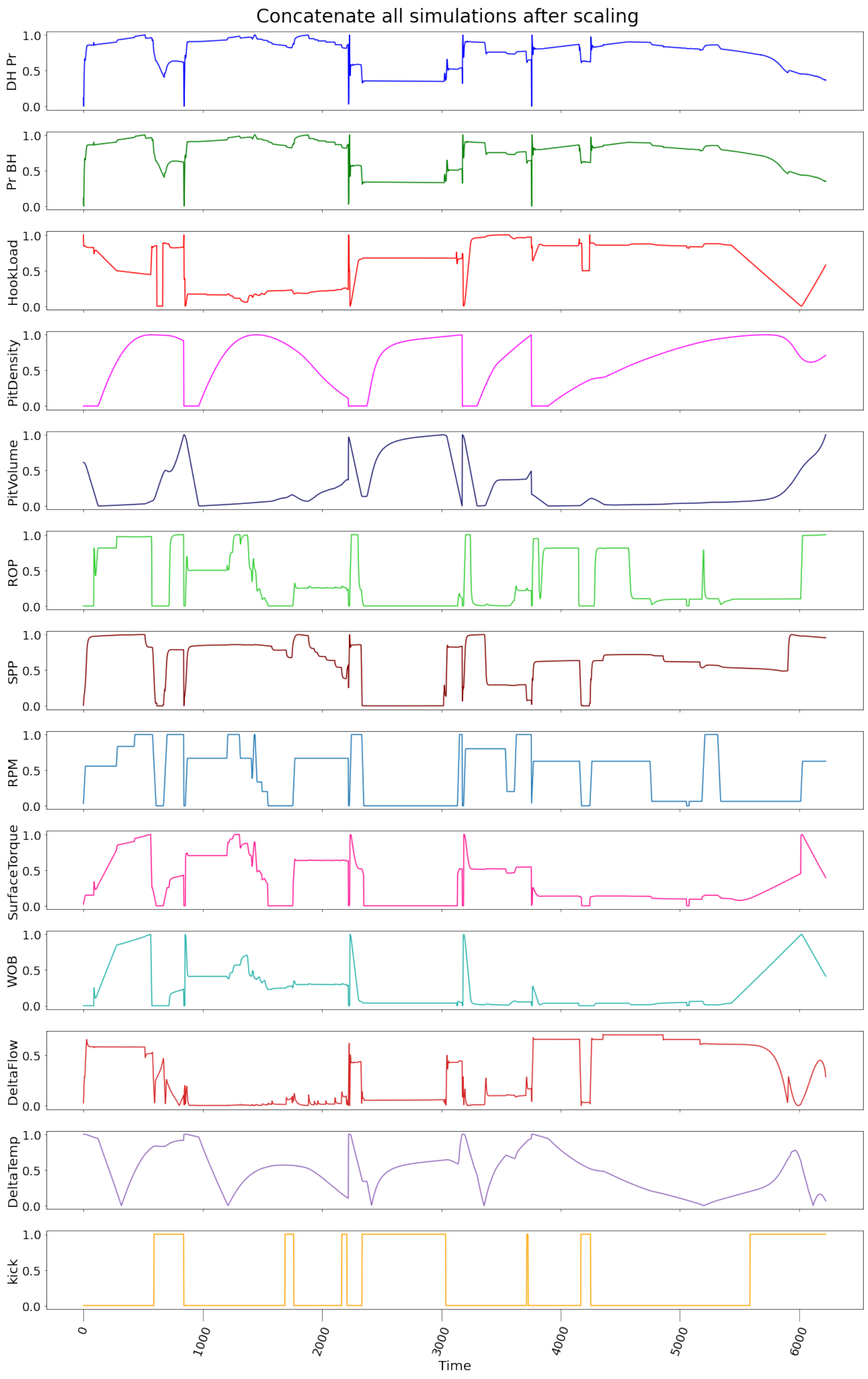
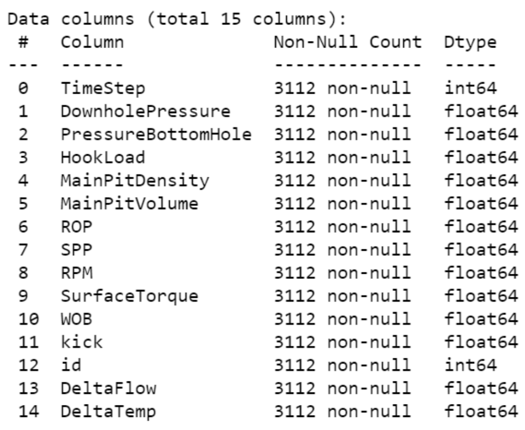


FIGURE 7.2: scaled features

7.2 Data pre-processing

To accomplish this thesis, multiple simulations are run for generating multiple kicks to have a variety of observations. Adequate observations lead the project to stepwise improvement. Although there is no must-follow rule toward predictive modeling, fundamental steps are the same in all projects. Drilling operations are constantly facing large volumes of data, so data exploration and visualization help to achieve a clear insight and identify trends, patterns, and outliers.

Table 7.1 gives a summary of ready-to-use sensor data. The primary steps toward ML model creation are based upon twelve input features and one output feature. The **”*TimeStep*”** represents the sorting aspect of data which indicates the data are in time order. The **”*id*”** column, which is described fully in 5.3.3.3, is a key component for the feature engineering process which is used in *extract_features()* and *select_features()* functions. The **”*id*”** column consists of positive integer numbers, ranging from 1 to 7, dedicated to different simulations as a way of distinguishing each simulation as an **”*entity*”**.



```
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   TimeStep            3112 non-null   int64
 1   DownholePressure    3112 non-null   float64
 2   PressureBottomHole  3112 non-null   float64
 3   HookLoad            3112 non-null   float64
 4   MainPitDensity      3112 non-null   float64
 5   MainPitVolume       3112 non-null   float64
 6   ROP                 3112 non-null   float64
 7   SPP                 3112 non-null   float64
 8   RPM                 3112 non-null   float64
 9   SurfaceTorque       3112 non-null   float64
 10  WOB                 3112 non-null   float64
 11  kick                3112 non-null   float64
 12  id                  3112 non-null   int64
 13  DeltaFlow           3112 non-null   float64
 14  DeltaTemp           3112 non-null   float64
```

TABLE 7.1: Total features overview in an analysis-ready format

Due to multiple independent simulations, the data is spread across multiple tables and must be gathered into a single dataframe. Figure 7.1 shows the final dataframe in which all the sensors data are concatenated in order of time sequence. Regardless of the simulation itself, simulation number 1 has come first, then simulation number 2 has come in the following, and so on. These visualization plots help to look at how the sensor readings are changing over time and how the kicks happen over time.

Getting the first look at Figure 7.1, it is obvious that data are suffering from unbalanced scaling. If the range of values varies widely, the feature with a higher value range starts dominating over lower scale features. Therefore, the very first step toward modeling is data scaling.

Feature scaling is one of the most important data pre-processing steps in ML and all scaling techniques try to achieve a similar target. Since the distribution of the data does not follow a Gaussian distribution, it is vital to use the normalization technique for scaling data. Normalization, also called scaling normalization, modifies features to lie between a given minimum and maximum value and leads the variables to contribute equally to the predictive analysis (Figure 7.2).

For further explanatory data analysis, histplot and boxplot are provided in Figure 7.3 and Figure 7.4 respectively to demonstrate the numerical data groups in a graphical manner.

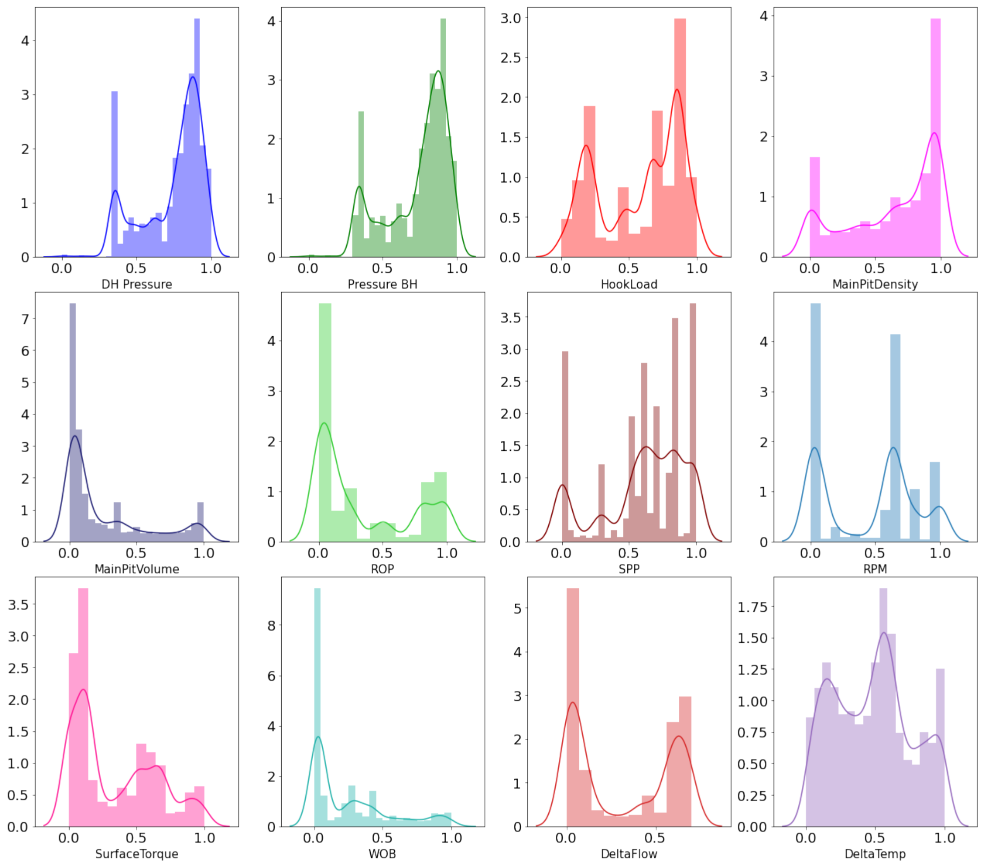


FIGURE 7.3: histplot of features

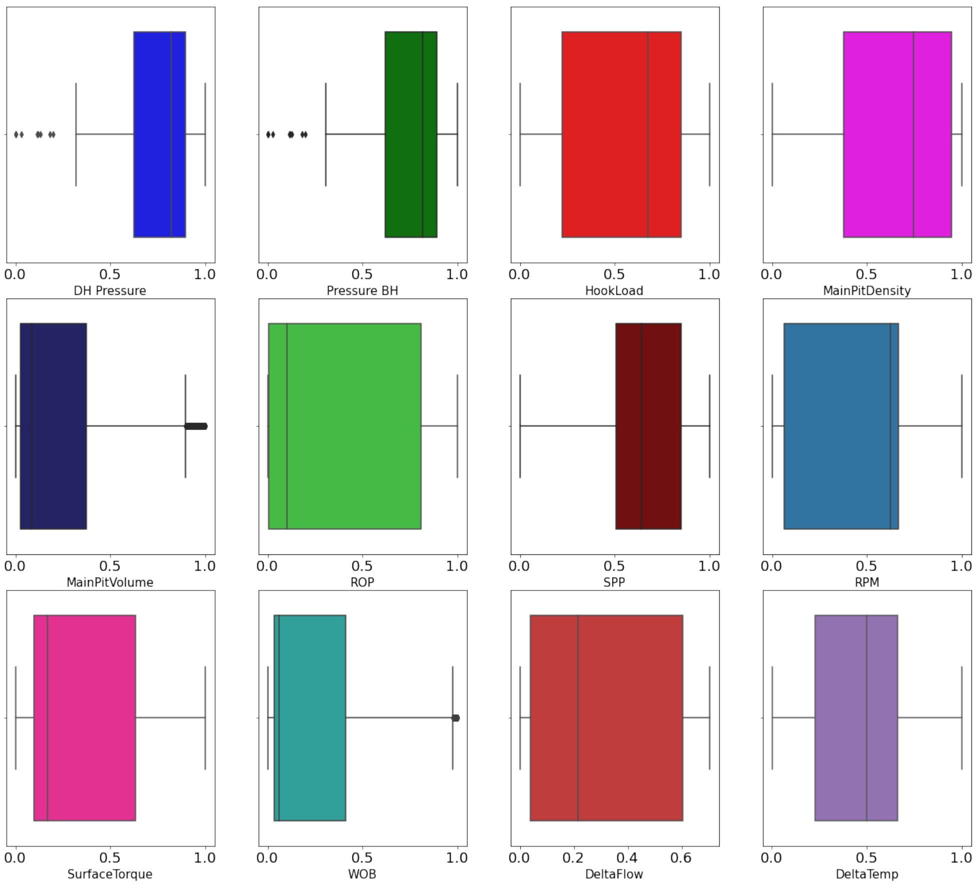


FIGURE 7.4: boxplot of features

7.2.1 Kick detection

Kick detection is categorized as an anomaly or outlier detection where rare events or observations raise the possibility of a potentially catastrophic incident. Figure 7.5 shows the artificial kicks generated by the OpenLab simulator over multiple simulations. Each simulation represents a series of observations over time. Therefore, each simulation consists of a time-series sequence of numerical data points in successive order while each of which is marked with a positive integer.

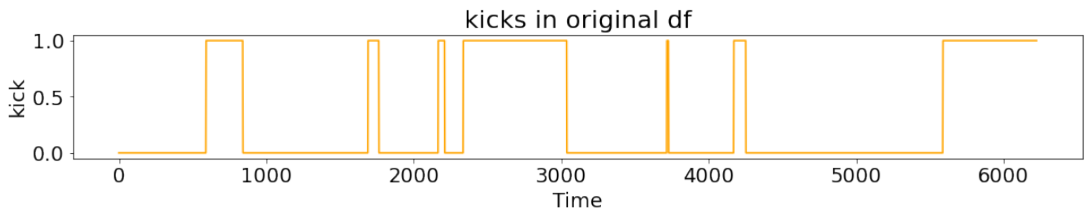


FIGURE 7.5: Original kicks

7.2.2 Kick prediction

Since the blowouts during drilling are very dangerous and life-threatening, the main focus during a drilling operation is to keep the wellbore pressure stable and prevent any type of influx of formation fluids. Early kick detection can prevent blowouts, therefore in the drilling industry, predicting the development trend of kick risk is of great importance than informative kick-detection notification. Focusing, the crew will have enough time to take action to eliminate or suppress the influx amount and prevent oil well blowouts. In this regard, Figure 7.6 shows that the label/target variable (kick) is shifted for different amounts of TimeStep to be evaluated how early the model is able to predict the kick happening.

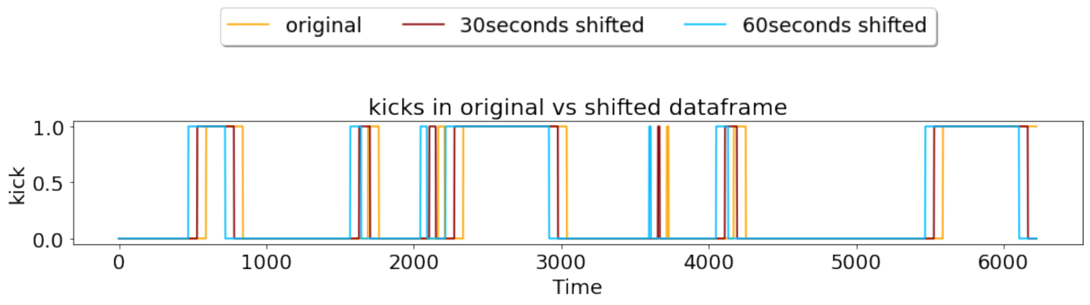
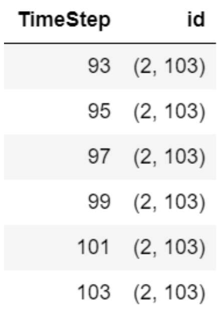


FIGURE 7.6: Original kicks vs shifted kicks

7.3 Window features (lag)

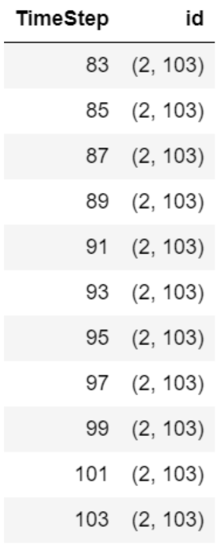
A time-series dataset must be transformed to a form that is comprehensive and interpretable for the ML model. Window features -aka lag features- are a summary of values over a fixed window of prior time steps in which every row contains data about one observation and also includes all previous occurrences of that observation. The sliding window method is the use of prior time steps to predict the next time step and the **“window”** is a fixed amount of passing time. In other words, the window features are basically the target variable but shifted over a period of time. The more we expand the window width, the more lagged features are included.

In section 5.3.3.3, it is described in details that how $roll\_time\_series()$ function transform the original dataframe(df) to the rolled dataframe (df_rolled) based on the **”*id*”** column. The $roll\_time\_series()$ function gives a new format of **”*id*”** to the (df_rolled) dataframe.



TimeStep	id
93	(2, 103)
95	(2, 103)
97	(2, 103)
99	(2, 103)
101	(2, 103)
103	(2, 103)

TABLE 7.2: When max_timeshift is set to 5, the current observation plus 5 prior time steps are all included as window features



TimeStep	id
83	(2, 103)
85	(2, 103)
87	(2, 103)
89	(2, 103)
91	(2, 103)
93	(2, 103)
95	(2, 103)
97	(2, 103)
99	(2, 103)
101	(2, 103)
103	(2, 103)

TABLE 7.3: When max_timeshift is set to 10, the current observation plus 10 prior time steps are all included as window features

7.4 Implementation

After rolling the dataframe, *extract_features()* function generates a large number of features automatically and also *select_features()* function selects the most relevant features which result in the best performing model. It is possible to control the size of the subsets with the parameters train_size and test_size. The test dataset size is set to 38% of the whole dataset for ML modeling and the rest is dedicated to the training dataset. Although the train-test split with more data in the training set will most likely give better accuracy, this number is achieved by error and trial in this thesis. The input parameters for $train\_test\_split()$ are described in the following subsections.

7.4.1 Detection

In Snippet 7.1, $df\_selected$ is the dataframe with the relevant features chosen form $df\_features$ and y is the final target for detection. The y used here is exactly the orange plot in Figure 7.5. The $df\_selected$ and y are the input parameters passed to $train\_test\_split()$ function afterwards.

SNIPPET 7.1: Select the most relevent features out of all the features engineered and kept in $df\_features$ based on y as the target

```
df_selected = select_features(df_features, y, fdr_level = 0.05)
```

7.4.1.1 DecisionTreeClassifier

Decision tree learning is predictive modeling that uses multiple algorithms to decide to split a node into two or more sub-nodes. Decision trees cover classification and regression problems, however, the tree in this work is called classification tree as the target is to classify label as "kick" or "not kick". Growing a tree involves deciding on which features to choose and what conditions to use for splitting, as well as the end of the branch where splitting stops and the decision/leaf is achieved.

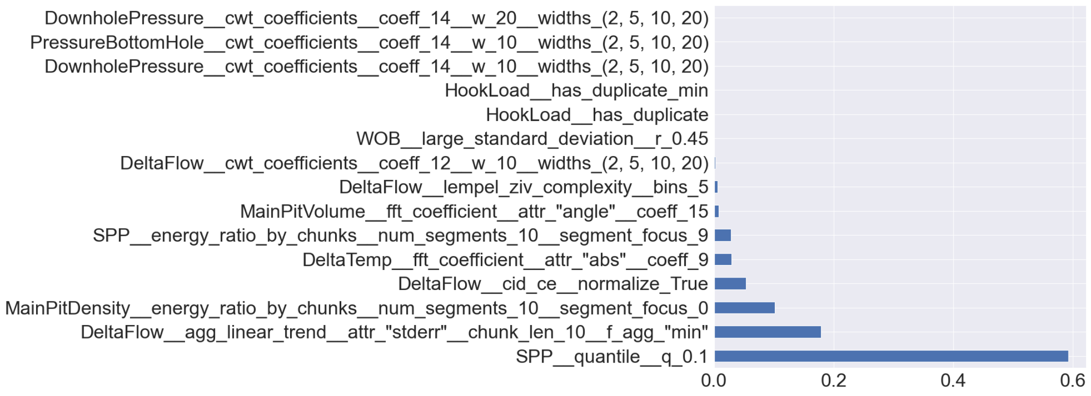


FIGURE 7.7: The most 15 relevant features in DecisionTreeClassifier

Since the deep decision tree is prone to overfitting, removing the branches that make use of features having low importance will reduce the complexity of the tree and improve accuracy. Although the number of selected features are hundreds of different generated variables, Figure 7.7 only shows the 15 most relevant features that are chosen by *select_features()* function via data selection.

Figure 7.8 demonstrates the prediction versus actual values by using DecisionTreeClassifier.

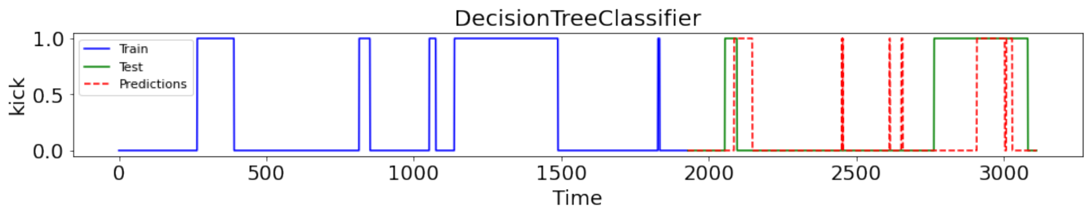


FIGURE 7.8: Predicted values of kicks in DecisionTreeClassifier

7.4.1.2 RandomForestClassifier

Random forest is a large number of individual decision trees and is one of the most used algorithms due to its simplicity and diversity. Multiple decision trees are built and merged together to get a more accurate and stable prediction model. Similar to the decision tree model, the random forest can be used for both classification and regression tasks as well.

One amazing advantage of the random forest model over the decision tree model is its ability to add additional randomness to the model while growing the trees. While the decision tree searches for the most important feature while splitting a node, random forest searches for the best feature among a random subset of features. Random forests overcome the overfitting problem that was one of the concerns in decision trees by creating random subsets of the features and building smaller trees using those subsets.

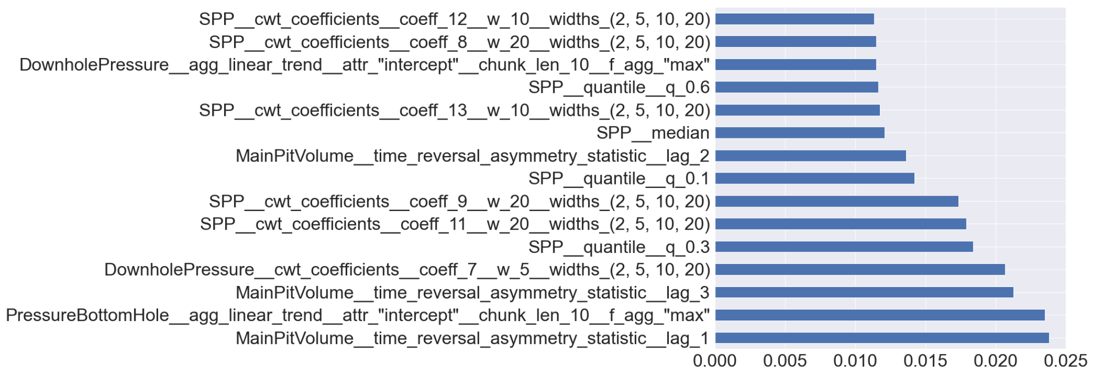


FIGURE 7.9: The most 15 relevant features in RandomForestClassifier

Although the number of selected features are hundreds of different generated variables, to avoid a mess in the bar plot, Figure 7.9 only shows the 15 most relevant features that are chosen by *select_features()* function in the feature selection process. By looking at the feature importance bar plot in Figure 7.9, it would be concluded which features to possibly drop because they don't contribute enough to the detection process.

Figure 7.10 demonstrates the prediction versus actual values by using RandomForest-Classifier.

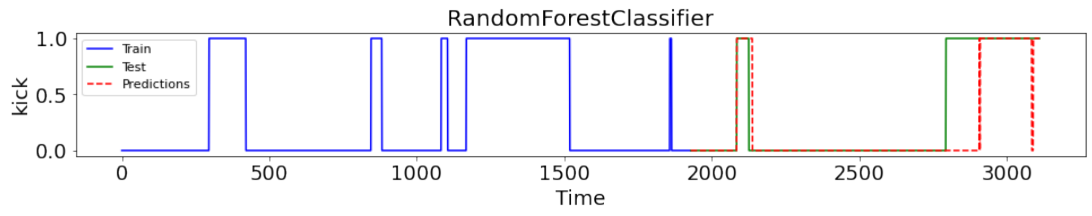


FIGURE 7.10: Predicted values of kicks in RandomForestClassifier

7.4.1.3 KNeighborsClassifier

The k-nearest neighbors (KNN) algorithm is a simple and easy to implement supervised ML algorithm and similar to the other two models, this one also can be used to solve both classification and regression problems. KNN looks for similarities between the new data and previous data and puts the new data into the category that is most similar to the available categories.

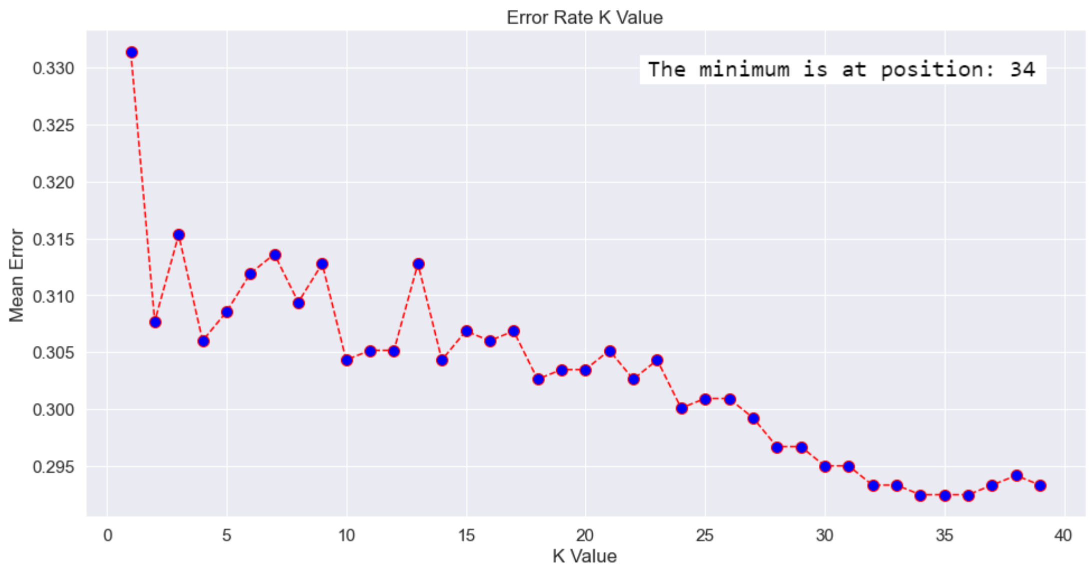


FIGURE 7.11: Find the optimal value of k in KNeighborsClassifier

One important step prior to the model creation is selecting the number ***"k"*** of the neighbors. Choosing the right ***"k"*** for our data is done by trying several Ks and picking the one that works best. Figure 7.11 illustrates out of 40 different ***"k"*** values, position 34 performs better for this issue. Since there is always a need to determine the value of ***"k"***, sometimes this could lead to complex calculations for some problems. Moreover, The computation cost is high in this model because the distance between the new data points and all the existing samples should be calculated to be able to decide which category the new data points belongs to.

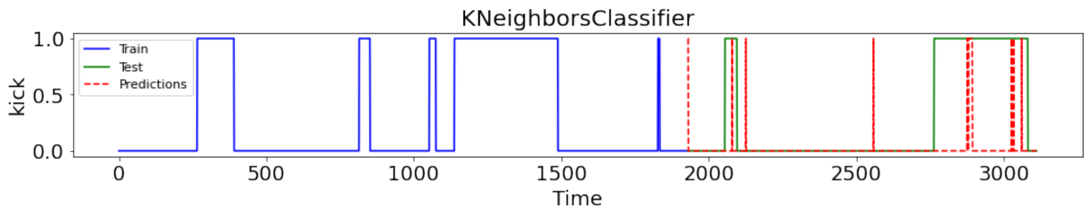
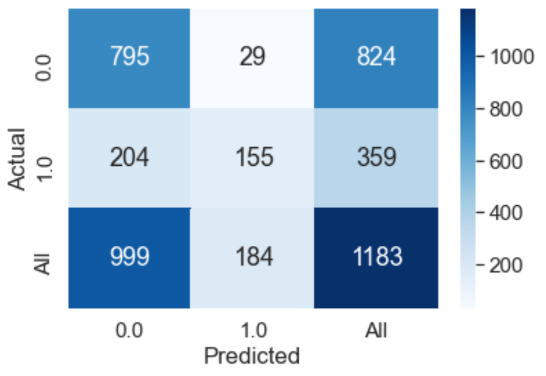


FIGURE 7.12: Predicted values of kicks in KNeighborsClassifier

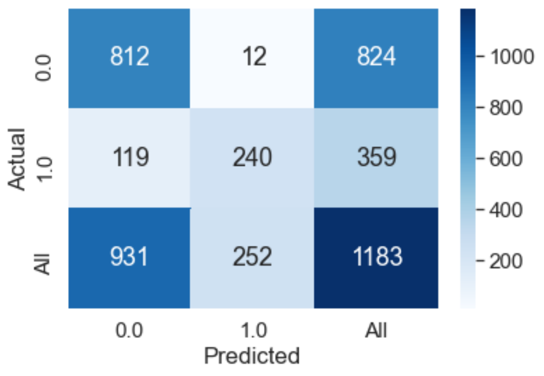
Figure 7.12 demonstrates the prediction versus actual values by using KNeighborsClassifier.

7.4.1.4 Evaluation for detective models

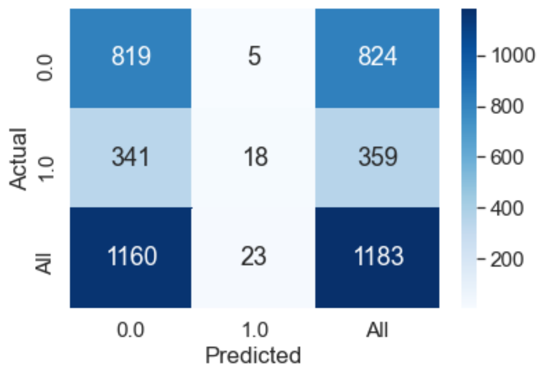
As it is discussed in section 6.2, it is worth mentioning again that some common metrics like MSE or RMSE are not suitable for classification problems like the problem in this thesis. Since the labels for the two groups are chosen arbitrarily, there is no sense of how far or close a classification is from the true value.



(A) DecisionTreeClassifier



(B) RandomForestClassifier



(C) KNeighborsClassifier

FIGURE 7.13: confusion matrices for detection modeling

For classification issues, a proper way of summarizing the performance of a classification algorithm is the confusion matrix (Figure 7.13). A confusion matrix is an N x N matrix that allows you to measure Recall, Precision, Accuracy, and ROC-AUC curve. In a confusion matrix, N is the number of target classes and this matrix gives direct comparisons of values like True Positives, False Positives, True Negatives, and False Negatives.

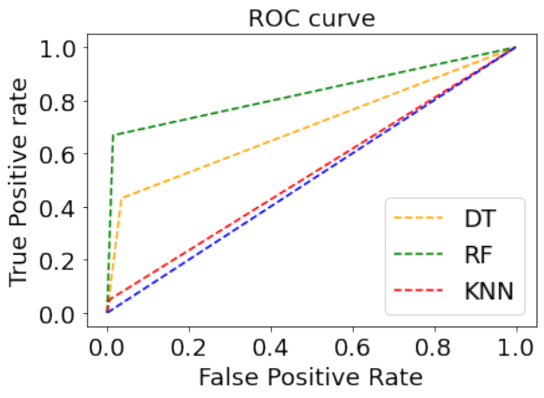
model name/metrics	Accuracy	precision	recall	f1	ROC AUC
DecisionTreeClassifier	0.81	0.881	0.432	0.571	0.703135
RandomForestClassifier	0.889	0.952	0.669	0.786	0.826980
KNeighborsClassifier	0.708	0.783	0.05	0.094	0.522036

TABLE 7.4: Evaluation measurements for detection modeling

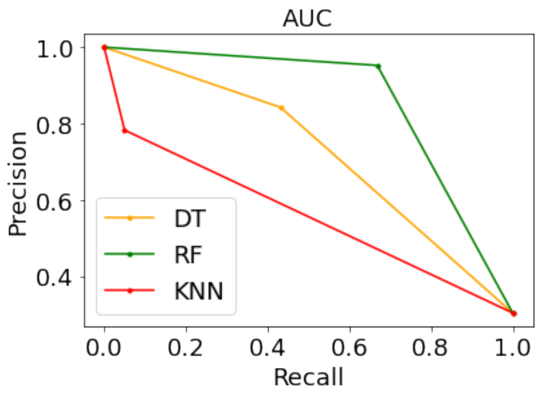
Table 7.4 summarizes the evaluation metrics for the three models for detection purpose. In classification problems, F1 score is the harmonic mean of precision and recall, thus it is a better measure than accuracy. For example, a model with an accuracy of 70% like the KNN detection model, might seems high enough accurate to some extent. However, a F1-score with 0.094 means the number of False Negative is comparably extensively high. The F1-score with a low amount shows that the KNN detection model does not perform successfully to detect the kicks in numerous cases.

There are two other useful measurements in classification problems named ROC curve and AUC. As it is illustrated in Figure 7.14a, the ROC curve shows the trade-off between sensitivity (or True Positive Rate) and specificity (1 – False Positive Rate), while AUC measures the ability of a classifier to distinguish between classes (Figure 7.14b). A Classifier that gives curves closer to the top-left corner indicates a better performance or in other words, a higher AUC indicates the better performance of the model at distinguishing between the positive and negative classes. AUC score around 0.7 to 0.8 is considered acceptable. However, the AUC score can also be equal to or smaller than 0.5, which indicates that a classifier performs equal or worse than a random classifier.

Figure 7.14 shows that, in kick detection analysis, RandomForestClassifier achieves better results than the other models.



(A) ROC



(B) AUC

FIGURE 7.14: ROC-AUC comparison for three detective models: DT(DecisionTreeClassifier), RF(RandomForestClassifier) and KNN(KNeighborsClassifier)

7.4.2 Prediction

For predictive analysis, Snippet 7.2 is applied with slightly different from Snippet 7.1. While Snippet 7.1 is used in detection problems, Snippet 7.2 is useful in prediction problems. This time, the *df_selected* is the dataframe with the relevant features chosen from *df_features* and *y_shifted* is the final target for detection.

SNIPPET 7.2: Select the most relevent features out of all the features engineered and kept in *df_features* based on *y_shifted* as the target

```
df_selected_shifted = select_features(df_features, y_shifted,
    fdr_level = 0.05)
```

The *y_shifted* used in Snippet 7.2 is exactly the red plot in Figure 7.15 where the original plot is shifted 30 seconds ahead. The *df_selected_shifted* and *y_shifted* are the input parameters passed to *train_test_split*() function then.

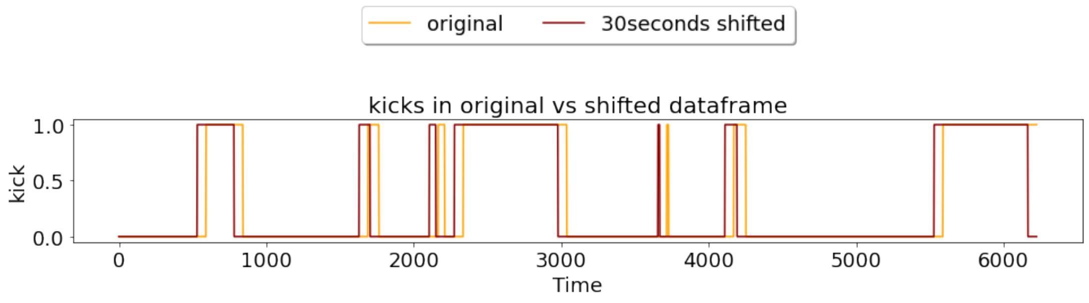


FIGURE 7.15: Original kicks vs shifted kicks for 30seconds

While *detection* and *forecasting* may sound similar to predictive analytics, understanding the difference between detection and prediction helps to analyze behaviour of trends in both scenarios and prevent potential drawbacks. Most probably in drilling operations, predicting a kick would be way more helpful than detecting it. Forecasting is a process of predicting or estimating future events based on past and present data and kick prediction is of great concern in wells. Uncontrolled incidents like sudden blowouts are consequences of the late kick detection which increases the possibility of injury and potential loss of life and equipment. On the other hand, early kick detection would provide enough time for the crew and managers to take some measurements and control the influx amount.

7.4.2.1 DecisionTreeClassifier-Prediction

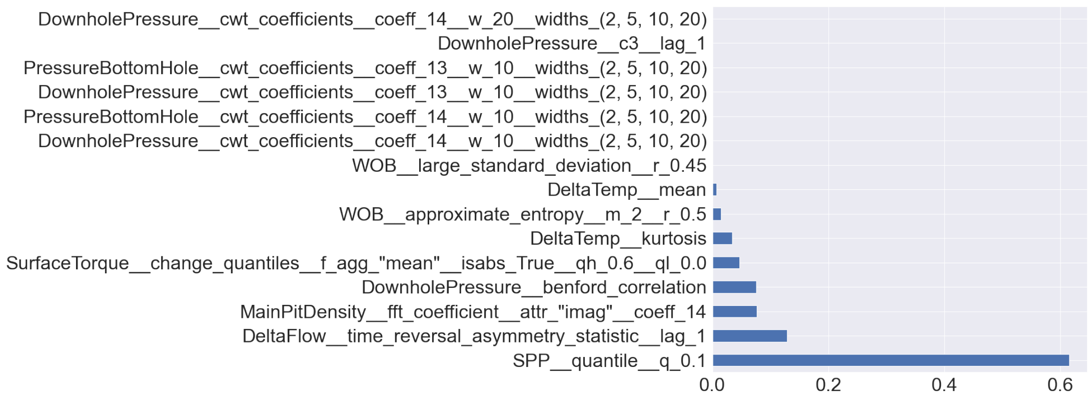


FIGURE 7.16: The most 15 relevant features for prediction by DecisionTreeClassifier

In contrast to Figure 7.7, Figure 7.16 shows that the decision tree modeling for prediction has some new features like $DeltaTemp\_mean$ and $DeltaTemp\_kurtosis$ that are among the most influential and dominant features.

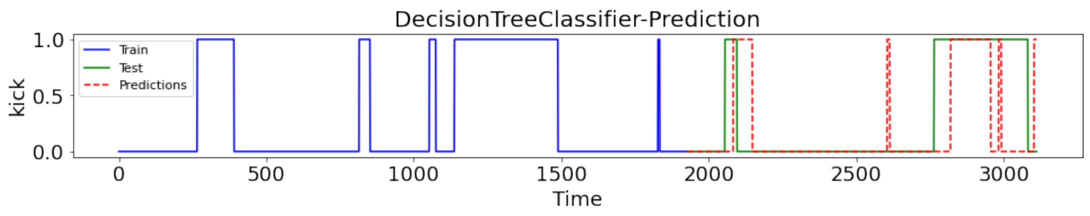


FIGURE 7.17: Predicted values of kicks in DecisionTreeClassifier with $y\_shifted$ target

Figure 7.17 demonstrates the prediction versus actual values by using DecisionTreeClassifier for prediction 30 seconds ahead of time.

7.4.2.2 RandomForestClassifier-Prediction

Similar to the Figure 7.9, in random forest modeling for prediction, the Figure 7.18 shows that the model is more affected by SPP and PressureBottomHole features more than other features.

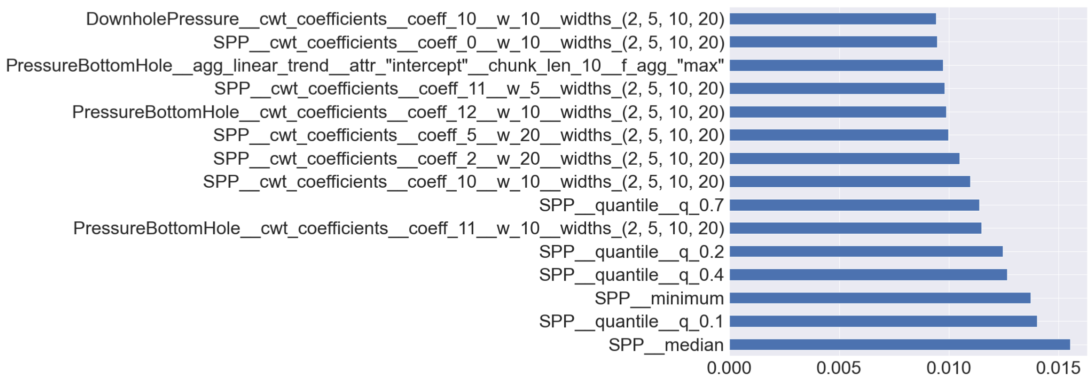


FIGURE 7.18: The most 15 relevant features for prediction by RandomForestClassifier

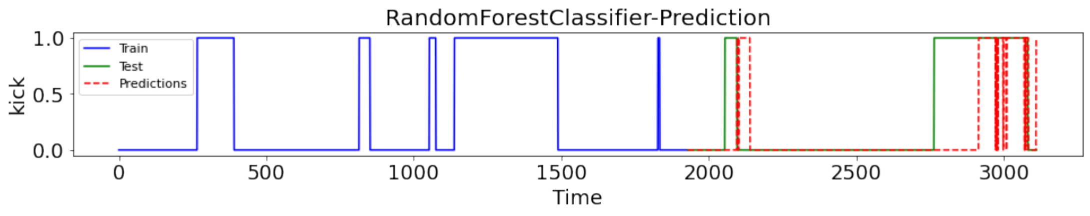


FIGURE 7.19: Predicted values of kicks in RandomForestClassifier with $y\_shifted$ target

Figure 7.19 demonstrates the prediction versus actual values by using RandomForestClassifier for prediction 30 seconds ahead of time.

7.4.2.3 KNeighborsClassifier-Prediction

In contrast to the Figure 7.11 related to detection, Figure 7.20 depicts that the mean error happens at position 8 in prediction scenario. Same as the previous KNN model for detection purposes, this model also results in high costs associated with the cost of computation. Since both detection and prediction modeling involve big data analysis, this model suffers from computing complexity overhead in terms of time and cost.

Figure 7.21 demonstrates the prediction versus actual values by using KNeighborsClassifier for prediction 30 seconds ahead of time. This plot looks so messy and although it has comparably high accuracy, it suffers from too many False Negative cases. Thus, it is unable to correctly interpret the trends to predict kicks ahead of time.

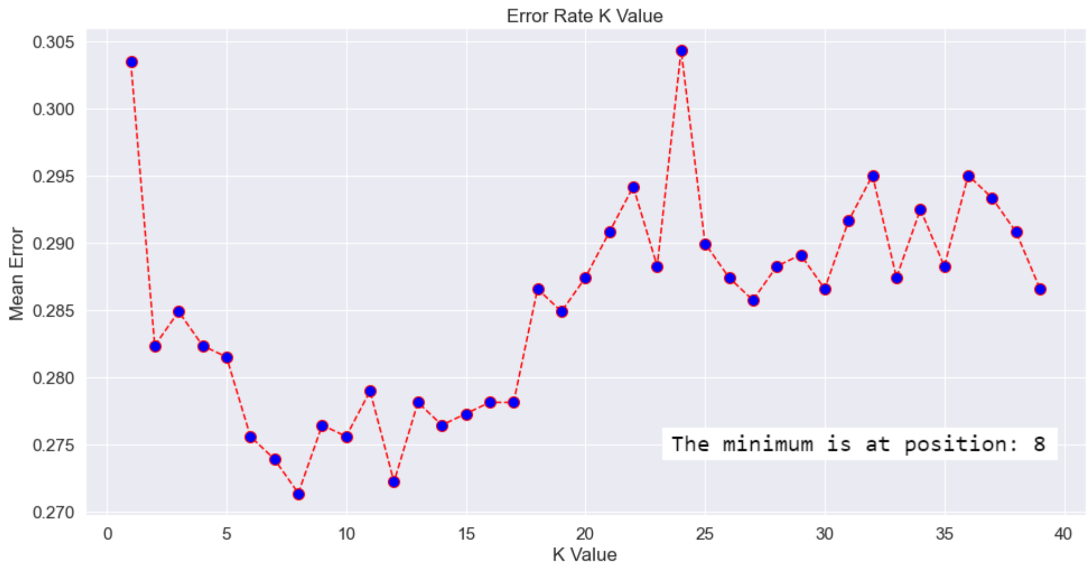


FIGURE 7.20: Find the optimal value of k in KNeighborsClassifier for prediction

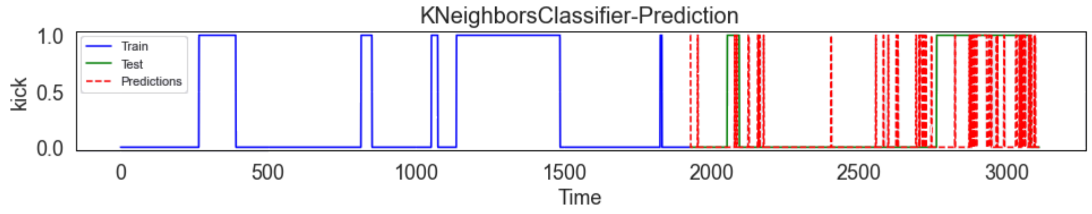
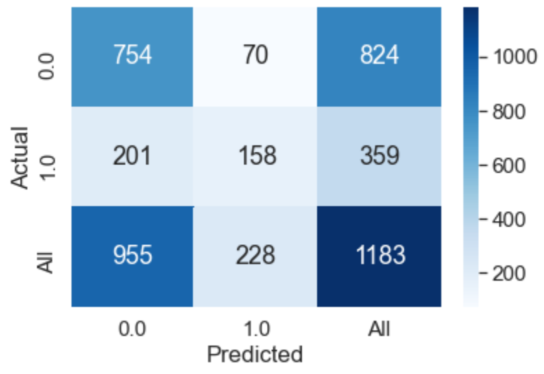


FIGURE 7.21: Predicted values of kicks in KNeighborsClassifier with $y\_shifted$ target

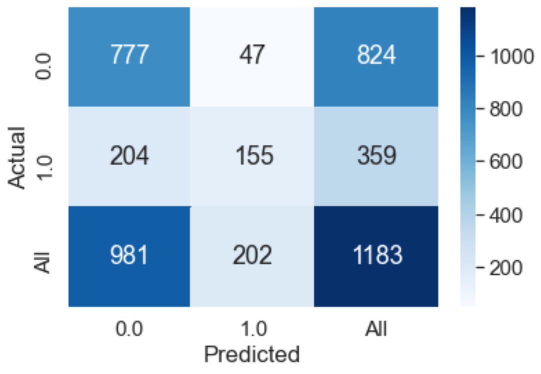
7.4.2.4 Evaluation for predictive models

Table 7.5 summarizes the evaluation measurements for three predictive modeling introduced earlier and the Figure 7.22 contains confusion matrices of three models. Considering Table 7.5 and Figure 7.22, it is obvious that two of the chosen models, DecisionTreeClassifier and RandomForestClassifier achieve almost similar outcomes. As False Negative and False Positive rates are errors in binary classification, a model with lower False Negative and False Positive brings better outputs. The RandomForestClassifier not only has a lower False Positive (47) but also has a higher True Positive (777) than other models.

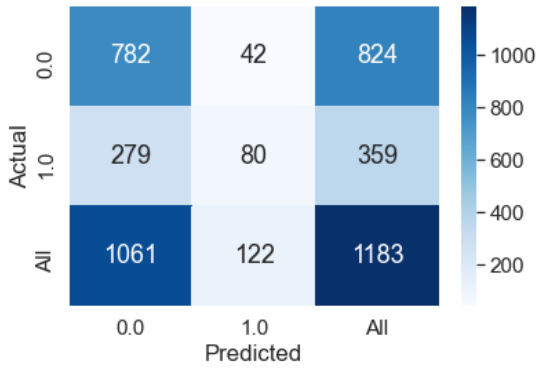
Moreover, although KNeighborsClassifier has a lower number of False Positive(42) compared to the other models, it has a high number of False Negative cases(279). Thus, KNeighborsClassifier performs poorly on predicting kicks 30 seconds ahead of time. In addition, KNeighborsClassifier outperforms the other two models in True Positive cases(782), this classifier still suffers from a low F1-score and recall. This case explicitly reveals that why accuracy is not solely enough for ML evaluation and assessment.



(A) DecisionTreeClassifier



(B) RandomForestClassifier



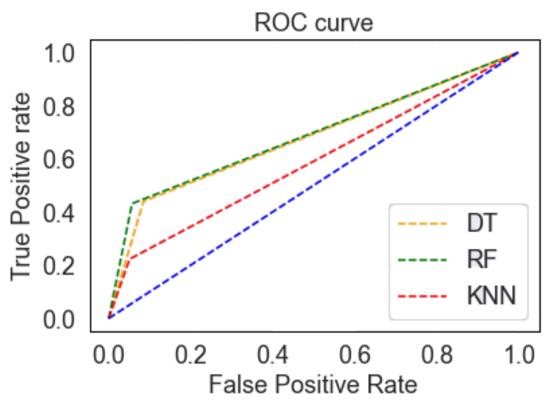
(C) KNeighborsClassifier

FIGURE 7.22: confusion matrices for prediction modeling

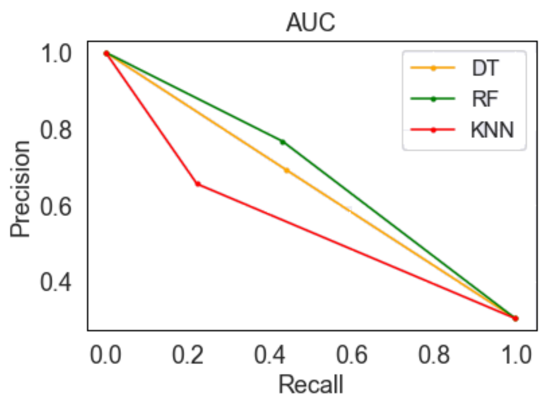
model name/metrics	Accuracy	precision	recall	f1	ROC AUC
DecisionTreeClassifier	0.771	0.693	0.440	0.538	0.677580
RandomForestClassifier	0.788	0.767	0.432	0.553	0.687358
KNeighborsClassifier	0.729	0.656	0.223	0.333	0.585935

TABLE 7.5: Evaluation measurements for predictive modeling

Figure 7.23 shows that, in kick prediction analysis, RandomForestClassifier and DecisionTreeClassifier achieve better results than the other model. Among these two, RandomForestClassifier represents better results with higher accuracy.



(A) ROC



(B) AUC

FIGURE 7.23: ROC-AUC comparison for three predictive models: DT(DecisionTreeClassifier), RF(RandomForestClassifier) and KNN(KNeighborsClassifier)

7.4.3 Improve the predictive ML model

7.4.3.1 Detection

The idea behind finding an optimum detective analysis and best ML modeling is to create the model with a variation of rolling window size and look over the functionality of the model. There are two parameters named $max\_timeshift$ and $min\_timeshift$ to adjust window length. The size of the window defines how many prior observations are needed to predict the next observation in the next time step. The window length is adaptive and is derived by the performance of the model in order to ensure certain reliability in the model performance.

For the RandomForestClassifier, 4-fold cross-validation is applied with the detection purpose improvement. However, in this stage, using different sizes of rolling windows for various ML models proved that shorter rolling window sizes tend to yield better performance than shorter sizes as is summarized in Table 7.6.

cross-validation for RandomForestClassifier				
metrics	fold1	fold2	fold3	fold4
accuracy	0.865	0.900	0.761	0.862
precision	0.550	0.991	0.202	1.000
recall	0.888	0.817	1.000	0.664
F1-score	0.679	0.896	0.336	0.798
ROC AUC	0.874322	0.904460	0.8727775	0.831761

TABLE 7.6: 4-fold cross-validation to improve the RandomForestClassifier for detection

Figure 7.24 shows a 4-fold cross-validation for RandomForestClassifier for kick detection and it demonstrates the primary order of simulations was not the best choice. Because with fold2, an accuracy of 90% and AUC score of 0.9 are obtained.

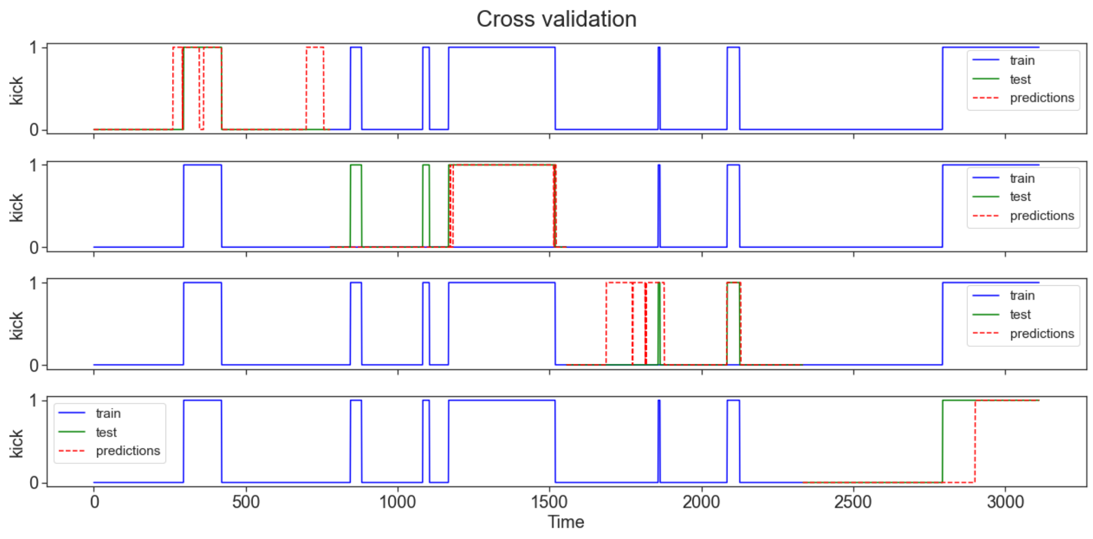


FIGURE 7.24: Improvement in RandomForestClassifier with 4-fold cross-validation for detection

It is worth mentioning that k-fold cross-validation is about estimating the accuracy, not improving it. Therefore, computing an overall evaluation score by taking the mean of the 4 folds scores would probably provide a more robust evaluation of the model.

7.4.3.2 Prediction

Choosing the right error or score metric helps the decision-maker to lead the model creation toward optimization. In the predictive evaluation, RandomForestClassifier and DecisionTreeClassifier have been concluded as the best fit for *kick prediction* problem during the drilling operation. However, the question is:

Is there any other way for improving the model at this stage?

The answer is yes. Although the ideal split for train/test is said to be 80/20 or 70/30 percent for training and testing sets respectively, there is no rule or the fixed number for applying this split. In this thesis, depending on the size of the dataset and parameter complexity, it is set to 62/38 percent. This distribution is reached randomly via error and trial. However, some approaches are introduced in 6.1 to not only pick the best algorithm but also improve the evaluation of ML models. Cross-validation is a useful statistical method for improving the evaluation of ML models.

For the DecisionTreeClassifier, a 4-fold time-series cross-validation is applied which is a variation of the k-fold method. In time-series cross-validation only the number of folds is chosen and the cross-validating is done on a rolling basis as it is depicted in Figure 7.25.

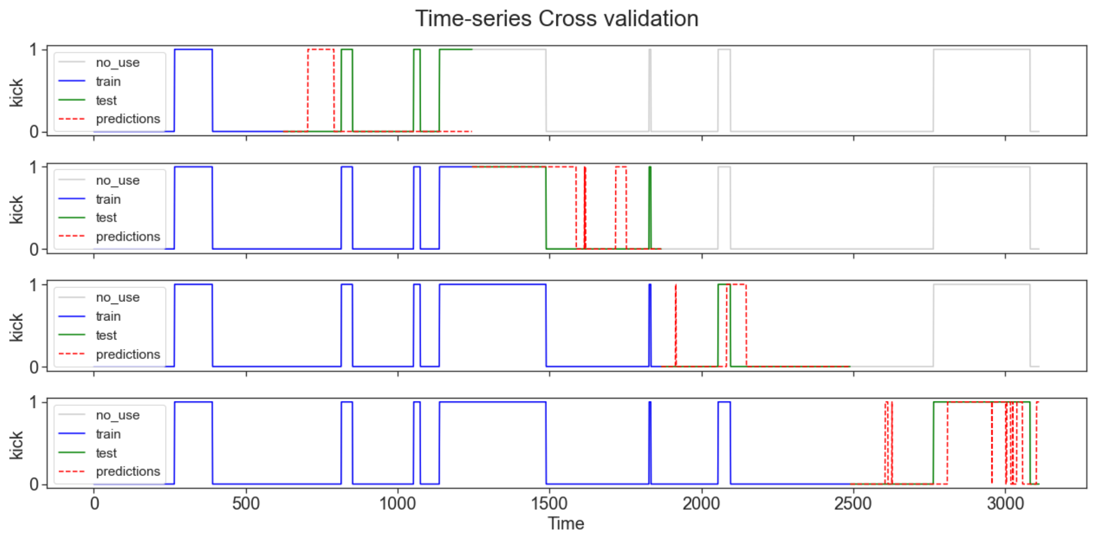


FIGURE 7.25: Improvement in DecisionTreeClassifier with 4-fold time-series cross-validation for prediction

Therefore, the DecisionTreeClassifier improves in accuracy, precision, recall, and F1-score and ROC-AUC in the last fold (fold4). Table 7.7 gives a summary of each fold performance.

time-series cross-validation for DecisionTreeClassifier				
metrics	fold1	fold2	fold3	fold4
accuracy	0.593	0.770	0.868	0.818
precision	0	0.639	0.194	0.922
recall	0	0.976	0.317	0.704
F1-score	0	0.773	0.241	0.799
ROC AUC	0.405495	0.656	0.612065	0.820951

TABLE 7.7: 4-fold time-series cross-validation to improve the DecisionTreeClassifier for prediction

In this part, it is worth mentioning that it is not necessary to stick with 4 folds. Generally, somewhere between 5 and 10 will give good results. The window length and number of folds are two factors with high influence in the output results.

7.4.4 Long short-term memory (LSTM)

As it is described in section 3.1.3, NN is mostly used when there are complex nonlinear relationships between inputs and output. LSTM networks are a type of RNN used for sequence classification.

Preparing the input values is a vital part of the LSTM time-series predictions since LSTM expects the input values in a specific 3D tensor format of test sample size by time steps by the number of input features.[122] Since the input layer shape is significantly in a different form compared to other ML models, so the Snippet 7.3 is applied to convert the sequence data that may be a 1D or 2D matrix of numbers to the required 3D format of the LSTM input layer.

SNIPPET 7.3: convert the sequence data from 1D or 2D matrix to the 3D format required for LSTM input layer

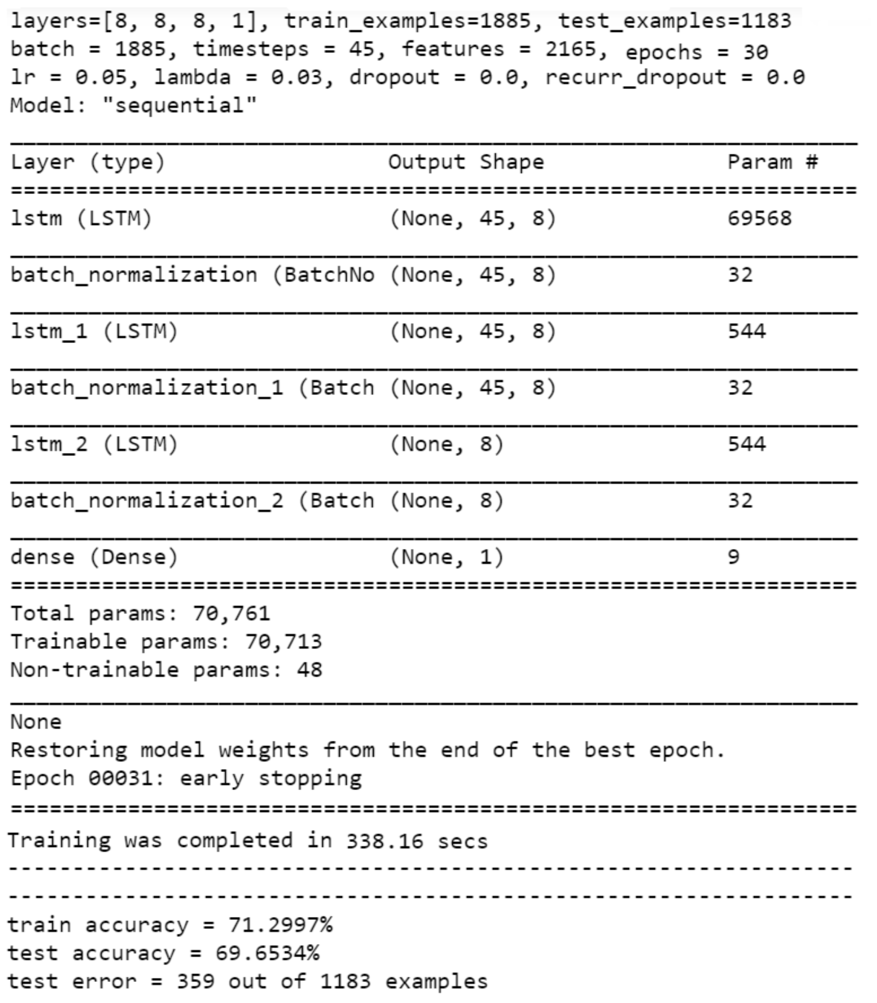
```
train_features , test_features , train_labels , test_labels  =
   train_test_split(df_selected, y, test_size= 0 .38, shuffle=False)
T = 45  # my choice of the rolling window
prepend_features = train_features.iloc[-(T- 1 ):]
test_features  = pd.concat([prepend_features, test_features ], axis=
    0 )

X_train, y_train = [], []
for i in range(train_labels.shape[ 0 ] - (T- 1 )):
    X_train.append(train_features.iloc[i:i+T].values)
    y_train.append(train_labels.iloc[i + (T- 1 )])
X_train, y_train = np.array(X_train), np.array(y_train).reshape(- 1 ,
    1 )
print(f'Train_data_dimensions:{X_train.shape},{y_train.shape}')

X_test, y_test = [], []
for i in range(test_labels.shape[ 0 ]):
    X_test.append(test_features.iloc[i:i+T].values)
    y_test.append(test_labels.iloc[i])
X_test, y_test = np.array(X_test), np.array(y_test).reshape(- 1 , 1 )

print(f'Test_data_dimensions:{X_test.shape},{y_test.shape}')
```

The Train_data_dimensions would be (1885, 45, 2165), (1885, 1) and Test_data_dimensions would be (1183, 45, 2165), (1183, 1). The meaning of the 3 input dimensions are:

- samples
- time steps

- features

Note that in my dataset I have multivariate input features to predict one-dimensional y values. Thus, in the LSTM input layer which is specified by the input_shape argument on the first hidden layer of the network, there are X_train.shape[2] input features, and there is only one unit in the Dense.



```
layers=[8, 8, 8, 1], train_examples=1885, test_examples=1183
batch = 1885, timesteps = 45, features = 2165, epochs = 30
lr = 0.05, lambda = 0.03, dropout = 0.0, recurr_dropout = 0.0
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 45, 8)             69568
_________________________________________________________________
batch_normalization (BatchNo (None, 45, 8)             32
_________________________________________________________________
lstm_1 (LSTM)                (None, 45, 8)             544
_________________________________________________________________
batch_normalization_1 (Batch (None, 45, 8)             32
_________________________________________________________________
lstm_2 (LSTM)                (None, 8)                 544
_________________________________________________________________
batch_normalization_2 (Batch (None, 8)                 32
_________________________________________________________________
dense (Dense)                (None, 1)                 9
=================================================================
Total params: 70,761
Trainable params: 70,713
Non-trainable params: 48
_________________________________________________________________
None
Restoring model weights from the end of the best epoch.
Epoch 00031: early stopping
=================================================================
Training was completed in 338.16 secs
-----------------------------------------------------------------
-----------------------------------------------------------------
train accuracy = 71.2997%
test accuracy = 69.6534%
test error = 359 out of 1183 examples
```

TABLE 7.8: key parameters while building an LSTM model

A few key parameters provided in Table 7.8 for determining the quality of the network.[123][124][125]

1. Epochs: is a hyperparameter that specifies the number of TimeSteps the data will be passed to the neural network.
2. Batch size: in general refers to the number of Training examples utilized per Iteration. This is not to be confused with the rolling window size used as time-series predictors. This is also called Steps per epoch.

3. Activations function: describing which activation function to use. This function is used in order to help the network learn complex patterns in the data. The activation function can either let no flow or complete flow of data throughout the gates. Comparing to the neuron-based model like the brain, the activation function filters what should be passed to the next level neurons.

4. Optimizer: Keras provides built-in optimizers in order to improve the accuracy of the model.

5. Dropout: may be implemented on any or all hidden layers and it is a simple way to prevent overfitting.

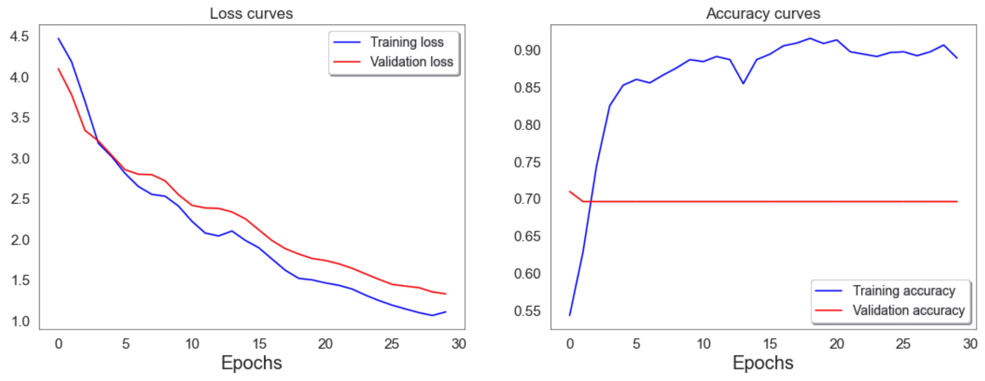


FIGURE 7.26: LSTM performance

Loss value implies how poorly or well a model behaves after each iteration of optimization while accuracy is the measure of how accurate the model's prediction is compared to the true data. Figure 7.26 shows a big difference between training and validation performance which indicates that the network is overfitting badly since, in the case of overfitting, the validation accuracy stops increasing. It is obvious that the validation accuracy is only 69% and remains unchanged during different epochs. The model is unable to generalize itself to get a validation accuracy above a certain threshold.

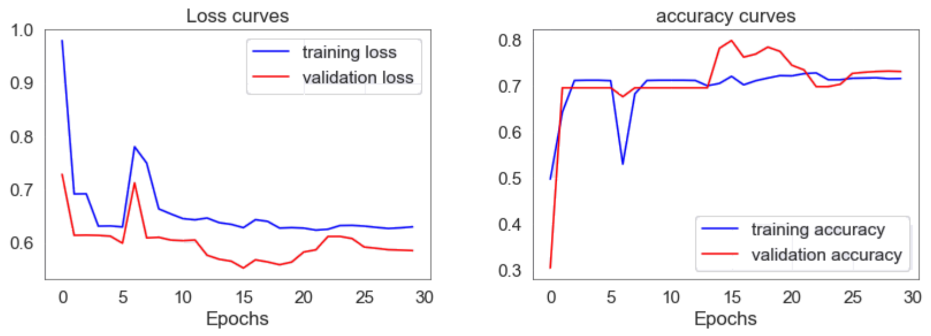


FIGURE 7.27: LSTM performance improvement by simplifying the model to overcome the overfitting issue

Validation loss and accuracy have improved with each epoch in Figure 7.27. Good accuracy with lower loss means low errors have been made on a few data. To improve the model, more layers can perform better but also harder to train, or sometimes a dropout layer randomly drops some of the connections between layers. Also, Changing the LSTM optimizer would result in better outputs and among different optimizers, "Adam" is generally chosen as the best overall choice. the trial and error approach for choosing the right amount of nodes and layers will provide the best results for any individual problem. Here, the fundamental model in Table 7.8 has simplified by means of neurons and batch size to resolve the overfitting issue.

After getting some intuition about how to choose the most important parameters, it is time to build the model couple of times. Testing various configurations and then evaluating the outcomes, the finalized model makes predictions with 74% accuracy and the output is demonstrated in Figure 7.28.

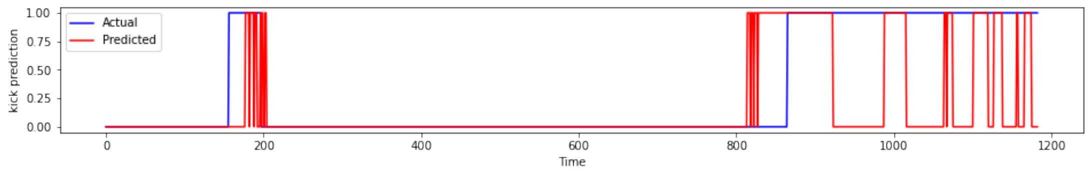


FIGURE 7.28: LSTM prediction results vs actual data

Chapter 8

Conclusion and future work

The main objective of this thesis was to build several ML models to detect and predict how a sequence of observations would threaten the well safety during the drilling operations. Using AI and ML facilitate detecting and predicting the well control issues such as kicks or loss circulations in high-risk industries like drilling. Although kicks are eventually controlled before causing blowouts most of the time, the main focus in this work was to predict kicks before happening in order to have enough time to take countermeasures to save lives and rig instruments. This task has been completed, and four different ML models were implemented, evaluated, and improved in this area. The work could be divided into two parts. The first part was to collect and prepare the data and the second part dedicated to ML models in terms of creation, evaluation, and modification.

Building an ML model and designing relevant algorithms are not as straightforward as they seem and deploying them in real business environments is even harder. Before starting to gather data, it is vital to ensure that the domain of the problem is correctly understood. *"kick detection and kick prediction"* is categorized as a *supervised learning* and *binary classification* since models were being trained using labeled data which means that data is already tagged with the correct answers. Moreover, working with the oil and gas industry requires handling the ever-increasing amount of data and large datasets which are of high importance in this field. To handle such an amount of data, the Python programming language and scikit-learn library are utilized widely in this work.

Time component is a crucial element in sequential forecasting problems where present observation occurrence is dependent on previous observations. Time-series forecasting problems are more complex as the time component adds an extra dimension to the dataset. The next step towards building any ML model is to gather and organize data which is the most time-taking task in this process.

The work initially started with main operating parameters such as FlowRateIn and Out, Pressure, Density, ROP, SPP, and so on. From raw data to the ML model, an advanced method of automatic feature engineering and feature selection is applied in this work. A new Python package named *TSFRESH* is used to extract hundreds of new features out of the existing input features. This perfect open source library generates new characteristics of data assisting in new insights into time-series and their dynamics. Using *feature_calculators* module, the *TSFRESH* creates a total of 794 time-series features including deviation, correlation, entropy, quantile, kurtosis, and so many other properties.

Keep in mind that having such a vast database, the crucial point was to remove outliers and anomalies without losing valuable observations. Plotting and visualization played an important role in this work to figure out what features to keep and what features to omit. *ReservePitDensity* and *ReservePitVolume* had unique value during operation and had no impact on kicks. Moreover, *DeltaTemp* and *DeltaFlow* were generated to reduce dimensionality. Extracted features are directly influenced by different sizes of the rolling window and train/test split. Even with the models themselves, the input variables like number of nodes in LSTM or the maximum depth of the tree in DecisionTreeClassifier would change the final result to some extent.

DecisionTreeClassifier, RandomForestClassifier, KNeighborsClassifier, and LSTM are four models that have been implemented in this work and then, some techniques were used to boost the performance of the ML models. In the detection process, RandomForestClassifier intensively outperformed the other models, while in the prediction, RandomForestClassifier and DecisionTreeClassifier performed almost similarly and better than KNeighborsClassifier. In the detection model improvement, cross-validation enhanced the RandomForestClassifier model enhanced the accuracy by 2% and ROC AUC by 8%. On the other part, the DecisionTreeClassifier prediction model was boosted more successfully than others by time-series cross-validation. The accuracy, precision, and ROC AUC were enhanced by 4%, 23%, and 15% respectively. In the LSTM model, performance enhancement was accomplished by simplifying the model (lower number of neurons and batch size and also change the optimizer to "Adam") to overcome the overfitting problem. Finally, the LSTM performance improved by 5% in total.

For future work, since the data is generated by the OpenLab drilling simulator delivered by NORCE, real-world data analysis is lacking in this work. Walking through a real-world use-case like kick incident and having a more realistic view of the trends and patterns would assist in obtaining clearer insights into the procedure. In addition, this work mainly puts more focus on the most common ML algorithms like K Nearest

Neighbor, Decision Tree, and Random Forest and less on Neural Networks. The next thing that could be experimented further is to examine Neural Networks and Long Short-Term Memory behaviour in order to achieve more accurate performance out of these models. Future studies could also aim to replicate the results of this work by changing the primary configuration and settings like train/test split size and feature filtering parameters.

List of Figures

2.1 Illustration of the drilling fluid circulation in the wellbore consists of drill pipes and a heavy drill bit. Gas emission can occure during circulation.[4] 6
2.2 How a balance between formation pressure and hydrostatic pressure keep the wellbore stable [5] 7
2.3 Runnig casing process is followed by drilling to deeper depths and by continuously adding lesser sized casings until the desired drilling depth is reached [8] 8
2.4 Typical casing and cementing program. Multiple strings from large to small diameter conductor may be required to reach the target producing zone. [12] 8
2.5 The right range for drilling formation pressure (pore pressure as red line) to balance the bottom-hole pressure [18] 11
2.6 Any change in drilled cuttings size and shape might indicate well control problematic issues [28] 15
2.7 Pit level is constant during normal drilling and circulation [21] 16
2.8 Pit gain (loss) is a relatively common problem in deep-water wells due to the entrance (exit) of unwanted and uncontrolled amount of fluids into (from) the wellbore. [21] 17
2.9 Abnormal behaviour in flow-rate and how ML algorithms can reduce false alarm by maintaining tight alarm threshold [29] 17
2.10 Any change in drilling normal trends can be warning indications of occurrence of a well control situation. For example a change in rate of penetration is called drilling break and it is a sign of potential problems. [21] 18
2.11 A deviation from a normal trend in formation pressure indicates a high-risk incident is going to happen [30] 18
2.12 Well influx/kick detected from pressure while drilling showing an abrupt increase in measured downhole pressure (PWD or ECD)and temperature. [31] 19

3.1 A clear visualization of overlapping Artificial Intelligence-related terminology. [32] 21
3.2 Recurrent Neural Network [39] 24
(a) RNN loops 24
(b) The repeating module in standard RNN contains a single layer 24
3.3 Long Short-Term Memory (LSTM) network [39] 25
(a) LSTM 25
(b) LSTM Gates 25
3.4 CRISP-DM overview and how a data mining project breaks into six phases to obtain better results [43] 26

4.1 How data scientists spend their time [76] 34
4.2 ML model steps [80] 36

5.1 Hole section tab in OpenLab simulator [85] 39
5.2 The visualization of wellpath tab structure in 3D [85] 39
5.3 The mud tab contains pie charts and rheogram chart [85] 40
5.4 The drill pipe inner and outer parameters can be varied in drillstring tab [85] 40
5.5 Geology tab in OpenLab simulator [85] 41
(a) Geopressure 41
(b) Geothermal 41
(c) Formation 41
5.6 A schematic layout of the rig tab from OpenLab simulator [85] 42
5.7 The first simulation sensor data and duration 46
5.8 The second simulation sensor data and duration 47
5.9 *ReservePitDensity* and *ReservePitVolume* columns have unique value in every simulation 49
5.10 The kick as output variable and other data as input variables show this case is a *Multivariate Time-series* problem. The plots demonstrate how the sensor readings are changing over time (these plots refer to the first simulation) 51
5.11 The input data distribution is not Gaussian(these plots refer to the first simulation) 52
5.12 Once the model is built, it is possible to come back for further feature engineering to see if the performance can be improved. [98] 53
5.13 Rolling window in TSFRESH package [97] 58

6.1 A common train/test split size [111] 63

6.2 In KFold cross validation the size of training and testing remain unchanged in each split [115] 64
6.3 In timeseriesSplit, each split, the training size is getting bigger and successive training sets [115] 65
6.4 ROC vs AUC [121] 69

7.1 before scaling bigger font size 72
7.2 scaled features 73
7.3 histplot of features 75
7.4 boxplot of features 76
7.5 Original kicks 76
7.6 Original kicks vs shifted kicks 77
7.7 The most 15 relevant features in DecisionTreeClassifier 79
7.8 Predicted values of kicks in DecisionTreeClassifier 80
7.9 The most 15 relevant features in RandomForestClassifier 80
7.10 Predicted values of kicks in RandomForestClassifier 81
7.11 Find the optimal value of k in KNeighborsClassifier 81
7.12 Predicted values of kicks in KNeighborsClassifier 82
7.13 confusion matrices for detection modeling 82
(a) DecisionTreeClassifier 82
(b) RandomForestClassifier 82
(c) KNeighborsClassifier 82
7.14 ROC-AUC comparison for three detective models: DT(DecisionTreeClassifier), RF(RandomForestClassifier) and KNN(KNeighborsClassifier) 84
(a) ROC 84
(b) AUC 84
7.15 Original kicks vs shifted kicks for 30seconds 84
7.16 The most 15 relevant features for prediction by DecisionTreeClassifier 85
7.17 Predicted values of kicks in DecisionTreeClassifier with $y\_shifted$ target 85
7.18 The most 15 relevant features for prediction by RandomForestClassifier 86
7.19 Predicted values of kicks in RandomForestClassifier with $y\_shifted$ target 86
7.20 Find the optimal value of k in KNeighborsClassifier for prediction . . 87
7.21 Predicted values of kicks in KNeighborsClassifier with $y\_shifted$ target 87
7.22 confusion matrices for prediction modeling 88
(a) DecisionTreeClassifier 88
(b) RandomForestClassifier 88
(c) KNeighborsClassifier 88

7.23 ROC-AUC comparison for three predictive models: DT(DecisionTreeClassifier), RF(RandomForestClassifier) and KNN(KNeighborsClassifier) 89
(a) ROC 89
(b) AUC 89
7.24 Improvement in RandomForestClassifier with 4-fold cross-validation for detection 90
7.25 Improvement in DecisionTreeClassifier with 4-fold time-series cross-validation for prediction 91
7.26 LSTM performance 94
7.27 LSTM performance improvement by simplifying the model to overcome the overfitting issue 94
7.28 LSTM prediction results vs actual data 95

List of Tables

5.1 Diabetic Prediction as a sample of not time-series dataframe and it is time-independent 55
5.2 Kick Forecasting as a time-series dataframe that includes a time-related column 55
5.3 Example of "id" column in a dataframe consists of two different entities (id 1 and 2) [97] 58
5.4 Example of "id" after rolling [97] 59
5.5 Example of "id" after rolling [97] 59

6.1 Confusion Matrix 67

7.1 Total features overview in an analysis-ready format 74
7.2 When max_timeshift is set to 5, the current observation plus 5 prior time steps are all included as window features 78
7.3 When max_timeshift is set to 10, the current observation plus 10 prior time steps are all included as window features 78
7.4 Evaluation measurements for detection modeling 83
7.5 Evaluation measurements for predictive modeling 88
7.6 4-fold cross-validation to improve the RandomForestClassifier for detection 89
7.7 4-fold time-series cross-validation to improve the DecisionTreeClassifier for prediction 91
7.8 key parameters while building an LSTM model 93

List of Snippets

5.1 change the object type to numeric type for ML model 47
5.2 Filling missing values 48
5.3 Finding columns with unique value 49
5.4 Data reduction in order to solve the information overload problems . . 49
5.5 Filling missing values 50
5.6 There are two data types in ML modeling: 1.Numerical 2.Categorical 50
5.7 normalization as a scaling technique 51
5.8 Concatenate all individual simulations to have a unique dataframe . . 54
5.9 Extract a comprehensive set of features on *df* dataframe 56
5.10 Rolling window in TSFRESH package 58
5.11 Rolling window in TSFRESH package with $max\_timeshift$ parameter 60
5.12 Rolling window in TSFRESH package with $min\_timeshift$ parameter 60
5.13 Usual feature extraction on the rolled dataframe 60
5.14 Select the most relevent features out of all the features engineered for ML modeling 60

6.1 train test split 62
6.2 Cross validation 63
6.3 time-series train test split 65
6.4 Useful packages for ROC curve and AUC score 69

7.1 Select the most relevent features out of all the features engineered and kept in $df\_features$ based on y as the target 79
7.2 Select the most relevent features out of all the features engineered and kept in $df\_features$ based on $y\_shifted$ as the target 84
7.3 convert the sequence data from 1D or 2D matrix to the 3D format required for LSTM input layer 92

Bibliography

[1] Haiyan Xie, Arun Kumar Shanmugam **and** Raja RA Issa. “Big Data Analysis for Monitoring of Kick Formation in Complex Underwater Drilling Projects”. **in**: *Journal of Computing in Civil Engineering* 32.5 (2018), **page** 04018030.

[2] S Rana **andothers**. “Facts and data on environmental risks-oil and gas drilling operations”. **in**: *SPE Asia Pacific Oil and Gas Conference and Exhibition*. Society of Petroleum Engineers. 2008.

[3] McKinsey Quarterly. *The decoupling of GDP and energy growth: A CEO guide*. `https://www.mckinsey.com/industries/electric-power-and-natural-gas/our-insights/the-decoupling-of-gdp-and-energy-growth-a-ceo-guide`. April2019.

[4] Gabriel Merhy de Oliveira, Jonathan Felipe Galdino **and** Admilson T Franco. “PRESSURE PROPAGATION IN DRILLING FLUIDS DURING A KICK OF GAS”. **in**: ().

[5] OL Ayodele, JM van Bever Donker **and** M Opuwari. “Pore pressure prediction of some selected wells from the Southern Pletmos Basin, offshore South Africa”. **in**: *South African Journal of Geology 2016* 119.1 (2016), **pages** 203–214.

[6] Simon Haukanes. “State and Parameter Identification Applied to Dual Gradient Drilling with Water Based Mud”. mathesis. NTNU, 2015.

[7] Eric Bertet **andothers**. *Method of cementing deformable casing inside a borehole or a conduit*. US Patent 5,718,288. 1998.

[8] Bikram Singh. *Offshore Well Drilling : A General Overview*. `https://www.marineinsight.com/offshore/offshore-well-drilling-a-general-overview/`. Oct2019.

[9] The Government of Saskatchewan. “Casing and Cementing Requirements”. **in**: *Directive PNG005* (May2018).

[10] Dipal Patel **andothers**. “A review on casing while drilling technology for oil and gas production with well control model and economical analysis”. **in**: *Petroleum* 5.1 (2019), **pages** 1–12.

[11] Vivek Thakar **andothers**. “A Review On Casing While Drilling Technology For Oil And Gas Production With Well Control Model And Economical Analysis”. **in**: *AAPG European Region, 3rd Hydrocarbon Geothermal Cross Over Technology Workshop*.

[12] Erik B. Nelson. *Offshore Well Drilling : A General Overview*. `https://www.slb.com/resource-library/oilfield-review/defining-series/defining-cementing`. 2011.

[13] Wayne Nash. *Tips for Running, Cementing Casing on Drilling Jobs*. `https://www.nationaldriller.com/articles/91049-tips-for-running-cementing-casing-on-drilling-jobs`. Dec2017.

[14] G Hoetz, B Jaarsma, M Kortekaas **andothers**. “Drilling hazards inventory: The key to safer-and cheaper-wells”. **in**: *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers. 2013.

[15] Qadir Mehmood Soomrob F. Sherwanic Muhammad Aamird Muhammad Mujtaba Asada Razali Bin Hassana. “Identification of Hazardous Nature of Well Drilling Operation With Associated Potential Hazards at Oil and Gas Extraction Industries: an Explanatory Approach”. **in**: *4th Scientific Conference on Occupational Safety and Health* 13.2 (2016), **pages** 85–92.

[16] John Wihbey. *Final report on the causes of BP’s Macondo Well blowout*. `https://journalistsresource.org/studies/environment/energy/final-report-causes-macondo-well-blowout/`. Sep2011.

[17] Zeyad Hassan. “Common Drilling well problems (Reasons, indications, mitigation and prevention)”. **in**: (April2018).

[18] Selim Shaker. “The Double Edged Sword: The Impact of the Interaction between Salt and Sediment on Sub-salt Exploration Risk in Deepwater from Mahogany to Jack”. **in**: (2008).

[19] Derek Krieg. *What is blowout?* `https://oilfieldbasics.com/2018/10/11/what-is-a-blowout/`. Oct2018.

[20] Mariana Coelho. *Causes of Kicks*. `https://www.linkedin.com/pulse/causes-kicks-mariana-coelho/`. April2017.

[21] DrillingFormulas.Com. *Possible Kick Indicators in Well Control*. `http://www.drillingformulas.com/possible-kick-indicators-in-well-control/`. 2019.

[22] Tongqiang Xia **andothers**. "Fluid flow in unconventional gas reservoirs". **in**: *Geofluids* 2018 (2018), **page** 2178582.

[23] Hewei Tang **andothers**. *Automatic abnormal trend detection of real time drilling data for hazard avoidance*. US Patent App. 16/649,249. 2020.

[24] Fred NG. "Multiphase simulations enhance well designs, contingency planning". **in**: *Drilling contractor* 68.6 (2012).

[25] Inge Mosti, Bjørn-Tore Anfinsen **and** Anne Sofie Flatebø. "Impact of thermal expansion on kick tolerance should be part of pre-drilling risk assessment". **in**: *Drilling contractor* (2008), **pages** 104–106.

[26] Pål Skalle, Agnar Aamodt, Isak Swahn **andothers**. "Detection of failures and interpretation of causes during drilling operations". **in**: *Abu Dhabi International Petroleum Exhibition & Conference*. Society of Petroleum Engineers. 2016.

[27] Pål Skalle, Agnar Aamodt, Odd Erik Gundersen **andothers**. "Detection of symptoms for revealing causes leading to drilling failures". **in**: *SPE Drilling & Completion* 28.02 (2013), **pages** 182–193.

[28] Paul Fekete **andothers**. "Wellbore stability management in weak bedding planes and angle of attack in well planing". **in**: *SPE Nigeria Annual International Conference and Exhibition*. Society of Petroleum Engineers. 2014.

[29] Sean Unrau **andothers**. "Machine learning algorithms applied to detection of well control events". **in**: *SPE Kingdom of Saudi Arabia Annual Technical Symposium and Exhibition*. Society of Petroleum Engineers. 2017.

[30] DrillingFormulas.Com. *Possible Kick Indicators in Well Control.* https://www.drillingformulas.com/pore-pressure-evaluation-while-drilling-is-important-for-well-control/. 2015.

[31] Jincai Zhang **and** Shangxian Yin. "Real-time pore pressure detection: indicators and improved methods". **in**: *Geofluids* 2017 (2017).

[32] Naresh Thakur. *The differences between Data Science, Artificial Intelligence, Machine Learning, and Deep Learning*. https://ai.plainenglish.io/data-science-vs-artificial-intelligence-vs-machine-learning-vs-deep-learning-50d3718d51e5. 2020.

[33] Jeff Williams. *Is AI the fuel oil and gas needs?* https://www.ey.com/en_ro/applying-ai-in-oil-and-gas. June2019.

[34] M Enamul Hossain **and** Abdulaziz Abdullah Al-Majed. *Fundamentals of sustainable drilling engineering*. John Wiley & Sons, 2015.

[35] import.io. *Data Mining vs. Machine Learning: What's The Difference?* https://www.import.io/post/data-mining-machine-learning-difference/. 2017.

[36] Augustine Osarogiagbon **andothers**. "A new methodology for kick detection during petroleum drilling using long short-term memory recurrent neural network". **in**: *Process Safety and Environmental Protection* (2020).

[37] Jiahang Han, Yanji Sun, Shaoning Zhang **andothers**. "A data driven approach of ROP prediction and drilling performance estimation". **in**: *International Petroleum Technology Conference*. International Petroleum Technology Conference. 2019.

[38] Qishuai Yin **andothers**. "Field data analysis and risk assessment of gas kick during industrial deepwater drilling process based on supervised learning algorithm". **in**: *Process Safety and Environmental Protection* 146 (2021), **pages** 312–328.

[39] Christopher Olah. *Understanding LSTM Networks*. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Aug2015.

[40] Kenneth Omokhagbo Afebu **andothers**. “LSTM-based approach for predicting periodic motions of an impacting system via transient dynamics”. **in**: *Neural Networks* 140 (2021), **pages** 49–64.

[41] Yao Ming **andothers**. “Understanding hidden memories of recurrent neural networks”. **in**: *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE. 2017, **pages** 13–24.

[42] Abdullah AlBar **andothers**. “Optimizing the Number of Nodes in Artificial Intelligence Neural Network Using Parallel Computing Methods”. **in**: *Abu Dhabi International Petroleum Exhibition & Conference*. Society of Petroleum Engineers. 2018.

[43] Ignitedata.com. *How successful are your Data Science projects?* `https://ignitedata.com.au/perspectives/how-complete-are-existing-data-science-methodologies`. 2020.

[44] Data Science Project Management. *What is CRISP DM?* `https://www.datascience-pm.com/crisp-dm-2/`. 2020.

[45] Daniel Codazzi. *Kick detection during drilling*. US Patent 5,154,078. 1992.

[46] Roland E Chemali, Volker Krueger **and** Rocco DiFoggio. *Early Kick Detection in an Oil and Gas Well*. US Patent App. 11/841,527. 2008.

[47] Jon Bang **andothers**. “Acoustic gas kick detection with wellhead sonar”. **in**: *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers. 1994.

[48] Quan Zhou **andothers**. “The application of ultrasonic based on Doppler effect used in early kick detection for deep water drilling”. **in**: *2013 International Conference on Communications, Circuits and Systems (ICCCAS)*. **volume** 2. IEEE. 2013, **pages** 488–491.

[49] Ali Karimi Vajargah **and** Eric van Oort. “Early kick detection and well control decision-making for managed pressure drilling automation”. **in**: *Journal of Natural Gas Science and Engineering* 27 (2015), **pages** 85–92.

[50] Arne Handal **andothers**. “Safety barrier analysis and hazard identification of blowout using managed pressure drilling compared with conventional drilling”.

in: *IADC/SPE Managed Pressure Drilling and Underbalanced Operations Conference and Exhibition*. Society of Petroleum Engineers. 2013.

[51] ED Toskey **andothers**. “Kick detection at the subsea mudline”. **in**: *Offshore Technology Conference*. Offshore Technology Conference. 2015.

[52] Adrian Ambrus **andothers**. “Real-time estimation of reservoir influx rate and pore pressure using a simplified transient two-phase flow model”. **in**: *Journal of Natural Gas Science and Engineering* 32 (2016), **pages** 439–452.

[53] Opeyemi Bello **andothers**. “Application of artificial intelligence techniques in drilling system design and operations: a state of the art review and future research pathways”. **in**: *SPE Nigeria Annual International Conference and Exhibition*. Society of Petroleum Engineers. 2016.

[54] Opeyemi Bello **andothers**. “Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art”. **in**: *Journal of Artificial Intelligence and Soft Computing Research* 5.2 (2015), **pages** 121–139.

[55] Farshad jafarizadeh By Mohammad Hossein Motamedie. “An Overview on Applications of Machine learning in petroleum Engineering”. **in**: (MArch2020).

[56] Christine I Noshi, Jerome J Schubert **andothers**. “The role of machine learning in drilling operations; a review”. **in**: *SPE/AAPG Eastern Regional Meeting*. Society of Petroleum Engineers. 2018.

[57] Hu Yin **andothers**. “Kick Risk Forecasting and Evaluating During Drilling Based on Autoregressive Integrated Moving Average Model”. **in**: *Energies* 12.18 (2019), **page** 3540.

[58] Raed Alouhali **andothers**. “Drilling through data: Automated kick detection using data mining”. **in**: *SPE International Heavy Oil Conference and Exhibition*. Society of Petroleum Engineers. 2018.

[59] Somadina Muojeke, Ramachandran Venkatesan **and** Faisal Khan. “Supervised data-driven approach to early kick detection during drilling operation”. **in**: *Journal of Petroleum Science and Engineering* (2020), **page** 107324.

[60] Dinh Minh Nhat, Ramachandran Venkatesan **and** Faisal Khan. “Data-driven Bayesian network model for early kick detection in industrial drilling process”. **in**: *Process Safety and Environmental Protection* (2020).

[61] Idris O Sule, Faisal Khan **and** Stephen Butt. “Experimental investigation of gas kick effects on dynamic drilling parameters”. **in**: *Journal of Petroleum Exploration and Production Technology* 9.1 (2019), **pages** 605–616.

[62] Mohammadreza Kamyab **andothers**. “Early kick detection using real time data analysis with dynamic neural network: A case study in iranian oil fields”. **in**: *Nigeria annual international conference and exhibition*. Society of Petroleum Engineers. 2010.

[63] Andreas K Fjetland **andothers**. “Kick detection and influx size estimation during offshore drilling operations using deep learning”. **in**: *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE. 2019, **pages** 2321–2326.

[64] Sergey Borozdin **andothers**. “Drilling Problems Forecast System Based on Neural Network”. **in**: *SPE Annual Caspian Technical Conference*. Society of Petroleum Engineers. 2020.

[65] Qishuai Yin **andothers**. “Field data analysis and risk assessment of gas kick during industrial deepwater drilling process based on supervised learning algorithm”. **in**: *Process Safety and Environmental Protection* 146 (), **pages** 312–328.

[66] Ahmed K Abbas **andothers**. “Intelligent decisions to stop or mitigate lost circulation based on machine learning”. **in**: *Energy* 183 (2019), **pages** 1104–1113.

[67] Xinxin Hou **andothers**. “Lost Circulation Prediction in South China Sea using Machine Learning and Big Data Technology”. **in**: *Offshore Technology Conference*. Offshore Technology Conference. 2020.

[68] Abdelkader Baaziz **and** Luc Quoniam. “How to use Big Data technologies to optimize operations in Upstream Petroleum Industry”. **in**: *21st World Petroleum Congress, Moscow*. 2014.

[69] Mehdi Mohammadpoor **and** Farshid Torabi. “Big Data analytics in oil and gas industry: An emerging trend”. **in**: *Petroleum* (2018).

[70] Alex Bekker. *How To Benefit from Big Data Analytics in the Oil and Gas Industry?* `https://www.scnsoft.com/blog/big-data-analytics-oil-gas`. Feb2020.

[71] Avantika Monnappa. *What's the Big Deal About Big Data?* `https://www.simplilearn.com/whats-the-big-deal-about-big-data-article`. May2020.

[72] MILA JONES. *Effective Ways How Data Analytics Help to Make a Better Entrepreneur*. `https://towardsdatascience.com/how-data-analytics-is-helping-small-businesses-re-imagine-growth-opportunities-a33f3defe744`. Sep2019.

[73] Deena Zaidi. *Role of Data Analytics in the Oil Industry*. `https://towardsdatascience.com/here-is-how-big-data-is-changing-the-oil-industry-13c752e58a5a`. Oct2017.

[74] Ashish Kumar. *Top Big Data Analytics Challenges Faced By Business Enterprises*. `https://elearningindustry.com/big-data-analytics-challenges-faced-business-enterprises-7-top`. 2018.

[75] Debi Prasanna Acharjya **and** K Ahmed. "A survey on big data analytics: challenges, open research issues and tools". **in**: *International Journal of Advanced Computer Science and Applications* 7.2 (2016), **pages** 511–518.

[76] Alex Woodie. *Data Prep Still Dominates Data Scientists' Time*. `https://www.datanami.com/2020/07/06/data-prep-still-dominates-data-scientists-time-survey-finds/`. Jul2020.

[77] Jason Brownlee. *Role of Data Analytics in the Oil Industry*. `https://www.mygreatlearning.com/blog/what-is-machine-learning/`. Aug2018.

[78] Bilal Esmael **andothers**. "A statistical feature-based approach for operations recognition in drilling time series". **in**: *International Journal of Computer Information Systems and Industrial Management Applications* 5 (2015), **pages** 454–61.

[79] Akshay P Jain. *Time Series Forecasting – Data, Analysis, and Practice*. `https://neptune.ai/blog/time-series-forecasting`. March2021.

[80] Vaishali Advani. *What is Machine Learning? How Machine Learning Works and future of it?* `https://www.mygreatlearning.com/blog/what-is-machine-learning/`. March2021.

[81] Philippe Esling **and** Carlos Agon. “Time-series data mining”. **in**: *ACM Computing Surveys (CSUR)* 45.1 (2012), **pages** 1–34.

[82] John Galt. *Time series analysis and forecasting explained*. `https://blog.johngalt.com/time-series-analysis-forecasting-explained`.

[83] Research Council of Norway (RCN). *OpenLab Drilling*. `https://www.norceresearch.no/prosjekter/openlab-drilling`. 2019.

[84] Andreas Kvalbein Fjetland. “Kick Detection During Offshore Drilling using Artificial Intelligence”. mathesis. Universitetet i Agder; University of Agder, 2019.

[85] Nejm Saadallah **andothers**. “OpenLab: Design and Applications of a Modern Drilling Digitalization Infrastructure”. **in**: *SPE Norway One Day Seminar*. Society of Petroleum Engineers. 2019.

[86] Fred NG. “Kick handling with losses in an HPHT environment”. **in**: *World oil* 230.3 (2009).

[87] Attila Nagy. “Availability and Quality of Drilling Data in the Volve Dataset”. mathesis. University of Stavanger, Norway, 2019.

[88] Serafeim Loukas. *What is Machine Learning: Supervised, Unsupervised, Semi-Supervised and Reinforcement learning methods*. `https://towardsdatascience.com/what-is-machine-learning-a-short-note-on-supervised-unsupervised-semi-supervised-and-aed1573ae9bb`. June2020.

[89] Yuji Roh, Geon Heo **and** Steven Euijong Whang. “A survey on data collection for machine learning: a big data-ai integration perspective”. **in**: *IEEE Transactions on Knowledge and Data Engineering* (2019).

[90] Peng Li **andothers**. “CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks”. **in**: *36th IEEE International Conference on Data Engineering (ICDE 2020)(virtual)*. ETH Zurich, Institute for Computing Platforms. 2021.

[91] Stacey Ronaghan. *Data Preparation for Machine Learning: Cleansing, Transformation Feature Engineering*. `https://towardsdatascience.com/data-preparation-for-machine-learning-cleansing-transformation-feature-engineering-d2334079b06d`. Sep2019.

[92] Ira Cohen. *The three different types of outliers*. `https://towardsdatascience.com/outliers-analysis-a-quick-guide-to-the-different-types-of-outliers-e41de37e6bf6`. Oct2018.

[93] D Addison, Stefan Wermter **and** Garen Z Arevian. “A comparison of feature extraction and selection techniques”. **in**: *Proceedings of the International Conference on Artificial Neural Networks*. 2003, **pages** 212–215.

[94] Petter Søland **and** Mikkel Vatne Thue. “Machine learning for automated stratigraphy classification: an empirical study to label subsurface formations in the Johan Sverdrup field”. mathesis. 2019.

[95] Girish Chandrashekar **and** Ferat Sahin. “A survey on feature selection methods”. **in**: *Computers & Electrical Engineering* 40.1 (2014), **pages** 16–28.

[96] Jundong Li **andothers**. “Feature selection: A data perspective”. **in**: *ACM Computing Surveys (CSUR)* 50.6 (2017), **pages** 1–45.

[97] Nils Braun Maximilian Christ. *The documentation of tsfresh*. `https://tsfresh.readthedocs.io/en/latest/index.html`. 2018.

[98] Llewelyn Fernandes openclassrooms.com. *Create New Features From Existing Features*. `https://openclassrooms.com/en/courses/6389626-train-a-supervised-machine-learning-model/6398776-create-new-features-from-existing-features`. 2020.

[99] Hongtao Shi **andothers**. “Efficient and robust feature extraction and selection for traffic classification”. **in**: *Computer Networks* 119 (2017), **pages** 1–16.

[100] Rob J Hyndman **and** George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[101] Georgia Papacharalampous, Hristos Tyralis **and** Demetris Koutsoyiannis. “Univariate time series forecasting of temperature and precipitation with a focus on machine learning algorithms: A multiple-case study from Greece”. **in**: *Water resources management* 32.15 (2018), **pages** 5207–5239.

[102] Maximilian Christ **andothers**. "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)". **in**: *Neurocomputing* 307 (2018), **pages** 72–77.

[103] Joseph Walters **and** S Kyle Travis. "Time-Series Forecasting: A Theoretical Model for Predicting Performance Potential". **in**: ().

[104] JA Ferreira, AH Zwinderman **andothers**. "On the benjamini–hochberg method". **in**: *Annals of Statistics* 34.4 (2006), **pages** 1827–1849.

[105] José A Ferreira. "The Benjamini-Hochberg method in the case of discrete test statistics". **in**: *The international journal of biostatistics* 3.1 (2007).

[106] Jason Brownlee. *Difference Between Classification and Regression in Machine Learning*. `https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/`. Dec2017.

[107] Dr. Michael J. Garbade. *Regression Versus Classification Machine Learning: What's the Difference?* `https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7`. Aug2018.

[108] Ramadass Sathya **and** Annamma Abraham. "Comparison of supervised and unsupervised learning algorithms for pattern classification". **in**: *International Journal of Advanced Research in Artificial Intelligence* 2.2 (2013), **pages** 34–38.

[109] pythonbasics.org. *Learn Python Programming*. `https://pythonbasics.org/`. 2021.

[110] Jason Brownlee. *Train-Test Split for Evaluating Machine Learning Algorithms*. `https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/`. July2020.

[111] Lars Buitinck **andothers**. "API design for machine learning software: experiences from the scikit-learn project". **in**: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013, **pages** 108–122.

[112] H Jabbar **and** Rafiqul Zaman Khan. "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)". **in**: *Computer Science, Communication and Instrumentation Devices* (2015), **pages** 163–172.

[113] Will Koehrsen. “Overfitting vs. Underfitting: A Complete Example”. **in**: *Towards Data Science* (2018).

[114] F. Pedregosa **andothers**. “Scikit-learn: Machine Learning in Python”. **in**: *Journal of Machine Learning Research* 12 (2011), **pages** 2825–2830.

[115] scikit learn.org. *scikit-learn Machine Learning in Python*. `https://scikit-learn.org/stable/`.

[116] Soumya Shrivastava. *Cross Validation in Time Series*. `https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4`. Jan2020.

[117] Christoph Bergmeir **and** José M Benítez. “On the use of cross-validation for time series predictor evaluation”. **in**: *Information Sciences* 191 (2012), **pages** 192–213.

[118] DataQuest.io Christian Pascual. *Understanding Regression Error Metrics in Python*. `https://www.dataquest.io/blog/understanding-regression-error-metrics/`. Sep2018.

[119] Will Koehrsen. *Beyond Accuracy: Precision and Recall*. `https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c`. 2018.

[120] Jason Brownlee. *ROC Curves and Precision-Recall Curves for Imbalanced Classification*. `https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification/`. Jan2020.

[121] David Banys Daniel Kobran. *Artificial Intelligence Wiki*. `https://docs.paperspace.com/machine-learning/wiki/auc-area-under-the-roc-curve`. 2020.

[122] Jaroslaw Kurek **andothers**. “Automatic Identification of Drill Condition During Drilling Process in Standard Laminated Chipboard with the Use of Long Short-Term Memory (LSTM)”. **in**: *19th International Conference Computational Problems of Electrical Engineering*. IEEE. 2018, **pages** 1–4.

[123] Christian Garbin, Xingquan Zhu **and** Oge Marques. “Dropout vs. batch normalization: an empirical study of their impact to deep learning”. **in**: *Multimedia Tools and Applications* (2020), **pages** 1–39.

[124] Amir Farzad, Hoda Mashayekhi **and** Hamid Hassanpour. “A comparative performance analysis of different activation functions in LSTM networks for classification”. **in**: *Neural Computing and Applications* 31.7 (2019), **pages** 2507–2521.

[125] David Banys Daniel Kobran. *Artificial Intelligence Wiki*. `https://docs.paperspace.com/machine-learning/wiki/epoch`. 2020.