Chair of Drilling and Completion Engineering

Master's Thesis

Machine Learning Application in Early Stuck Pipe Sign Detection by Real-time Monitoring Surface Drilling Parameters

Viacheslav Kobets

June 2021

MONTANUNIVERSITÄT LEOBEN
www.unileoben.ac.at

AFFIDAVIT

I declare on oath that I wrote this thesis independently, did not use other than the specified sources and aids, and did not otherwise use any unauthorized aids.

I declare that I have read, understood, and complied with the guidelines of the senate of the Montanuniversität Leoben for "Good Scientific Practice".

Furthermore, I declare that the electronic and printed version of the submitted thesis are identical, both, formally and with regard to content.

Date 14.06.2021

Signature Author
Viacheslav Kobets

MONTANUNIVERSITÄT LEOBEN
www.unileoben.ac.at

Viacheslav Kobets
Josef-Heissl-Strasse 26
8700 Leoben

To the Dean of graduate Studies of the Montanuniversitaet Leoben

Declaration of Approval for the Digital Publication of Scientific Theses

I am aware that the thesis entitled "Machine Learning Application in Early Stuck Pipe Sign Detection by Real-time Monitoring Surface Drilling Parameters " will be subject to a plagiarism assessment and may be stored by Montanuniversität Leoben for an unlimited period of time.

I agree that the University Library of Montanuniversität Leoben may publish the thesis open access in the World Wide Web. For embargoed theses this will be done after the embargo expires.

Note: in case you refuse the open access publication in the World Wide Web, the thesis will only be published in printed form (after a possible embargo has expired) in the University Library (dissertations also in the Austrian National Library).

I hereby agree with the open access publication of my thesis on the World Wide Web:

⊗ Yes

O No

Date 14.06.2021

Signature Author

Viacheslav Kobets

Master Thesis 2021 supervised by
Univ.-Prof. Dipl.-Ing. Dr.mont. Gerhard Thonhauser
Co-supervisor: Dipl.-Ing. Asad Elmgerbi

Machine Learning Application in Early Stuck Pipe Sign Detection by Real-time Monitoring Surface Drilling Parameters

DEPARTMENT PETROLEUM ENGINEERING

MONTAN UNIVERSITÄT

Dedicated to my friends and family.

Affidavit

I declare in lieu of oath that I wrote this thesis and performed the associated research myself using only literature cited in this volume.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich diese Arbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfsmittel bedient habe.

Name, 14 June 2021

Abstract

Early detection of stuck pipe incidence in real-time will lead to reducing the downtime and costly corrective actions. Therefore, in the last decades, intensive efforts have been made to develop methods to identify the stuck pipes early symptoms. One of these methods is torque and drag (T&D) modeling. Despite numerous advantages, the modeling-based method has, nowadays, T&D modeling is not common because it requires real-time merge and contextual data of heterogeneous frequency and quality as well as recalibration. The alternative approach is to apply data-driven algorithms to solve real-time torque and drag issues. This approach can be judged to be more robust than the first because real-time drilling data can be used. Besides, the data-driven algorithm does not limit the number of input parameters. From this perspective, this thesis proposes a model based on a machine learning concept focused on real-time detecting the early signs of impending stuck pipe by real-time monitoring the pertinent surface drilling parameters.

The newly proposed method uses machine learning linear regression algorithms to predict several parameters such as standpipe pressure, hook load, surface torque. The model can train from the data gained and adjust its performance automatically. The predicted parameters help the model to evaluate the behavior of real-time data. This comparison results in an alert level table for every predicted parameter. The alert level table allows the model to calculate stuck probability and generate an alarm when the probability exceeds the preliminary set limits. As a result, the model is suited to use real-time drilling operation data as input, calculate stuck pipe probability and notify the driller when the risk of stuck pipe is out of threshold and may be recognized as dangerous.

The first part of the thesis provides an overview of the problem, the current thesis challenges, and objectives. The main machine learning algorithms and data-based approaches application for early stuck pipe sign detection are denoted in the second part of the thesis. Finally, the third part of the work covers the methodology of the proposed approach in detail. Two case studies of the approach implementation using real-time data are also presented here.

In conclusion, the results of the work are presented and summarised. The features and shortcomings of the proposed approach are also discussed. The resulting model proves its ability to implement real-time data monitoring to avoid costly and time-consuming stuck pipe incidents.

Zusammenfassung

Die frühzeitige Erkennung des Auftretens von Steckrohren in Echtzeit führt zu einer Reduzierung der Ausfallzeiten und kostspieligen Korrekturmaßnahmen. Daher wurden in den letzten Jahrzehnten intensive Anstrengungen unternommen, um Methoden zur Identifizierung der festsitzenden und frühen Symptome zu entwickeln. Eine dieser Methoden ist die Drehmoment-und Schleppmodellierung (T&D). Trotz zahlreicher Vorteile ist die modellierungsbasierte Methode heutzutage nicht mehr üblich,da sie Echtzeit-Merge-und Kontextdaten heterogener Frequenz und Qualität sowie Neukalibrierung erfordert. Der alternative Ansatz besteht darin, datengesteuerte Algorithmen anzuwenden, um Drehmoment-und Schleppprobleme in Echtzeit zu lösen. Dieser Ansatz kann als robuster als der erste beurteilt werden, da Bohrdaten in Echtzeit verwendet werden können. Außerdem begrenzt der datengetriebene Algorithmus die Anzahl der Eingabeparameter nicht. Aus dieser Perspektive schlägt diese Arbeit ein Modell vor, das auf einem maschinellen Lernkonzept basiert, das sich darauf konzentriert, die frühen Anzeichen eines drohenden feststeckenden Rohrs in Echtzeit zu erkennen, indem die relevanten Oberflächenbohrparameter in Echtzeit überwacht werden.

Die neu vorgeschlagene Methode verwendet lineare Regressionsalgorithmen für maschinelles Lernen, um mehrere Parameter wie Standrohrdruck, Hakenlast und Oberflächendrehmoment vorherzusagen. Das Modell kann aus den gewonnenen Daten trainieren und seine Leistung automatisch anpassen. Die vorhergesagten Parameter helfen dem Modell, das Verhalten von Echtzeitdaten zu bewerten. Dieser Vergleich führt zu einer Tabelle mit Warnstufen für jeden vorhergesagten Parameter. Die Warnstufen-Tabelle ermöglicht es dem Modell, die Wahrscheinlichkeit zu berechnen und einen Alarm zu erzeugen, wenn die Wahrscheinlichkeit die vorläufig festgelegten Grenzen überschreitet. Infolgedessen eignet sich das Modell, um Bohroperationsdaten in Echtzeit als Eingabe zu verwenden, die Wahrscheinlichkeit eines festsitzenden Rohrs zu berechnen und den Bohrer zu benachrichtigen, wenn das Risiko eines festsitzenden Rohrs außerhalb des Schwellenwerts liegt und als gefährlich erkannt werden kann.

Der erste Teil der Arbeit gibt einen Überblick über das Problem, die aktuellen Herausforderungen und Ziele der Arbeit. Die wichtigsten maschinellen Lernalgorithmen und datenbasierten Ansätze Anwendung für die frühe Stuck Pipe Sign Detection sind im zweiten Teil der Arbeit bezeichnet. Schließlich behandelt der dritte Teil der Arbeit die Methodik des vorgeschlagenen Ansatzes im Detail. Hier werden auch zwei Fallstudien zur Umsetzung des Ansatzes anhand von Echtzeitdaten vorgestellt.

Abschließend werden die Ergebnisse der Arbeit vorgestellt und zusammengefasst. Die Merkmale und Mängel des vorgeschlagenen Ansatzes werden ebenfalls diskutiert. Das resultierende Modell beweist seine Fähigkeit, eine Echtzeitdatenüberwachung zu implementieren, um kostspielige und zeitaufwändige Vorfälle mit feststeckenden Leitungen zu vermeiden.

Acknowledgments

I would like to express my sincere gratitude to supervisors Prof. Dipl.-Ing. Dr.mont. Gerhard Thonhauser & Dipl.-Ing. Asad Elmgerbi for providing their invaluable guidance, comments, suggestions throughout the project, the help during model building and validation, and giving the knowledge about stuck pipe early signs detection techniques.

Contents

Chapter 1 Introduction

1.1 Overview

Drill string stuck forecasting and mitigation strategies have attracted and will continue to attract attention in drilling as more complex wells are drilled in depleted zones to reach deeper formations. Detecting stuck pipe early and reducing the likelihood of pipe stuck in real-time helps prevent stuck pipe occurrence and assists in making informed decisions about the appropriate release mechanism to be adopted if it eventually occurs. It is reported that stuck pipe incidents take about 25% of total non-productive time (NPT) (Muqeem et al., 2012).
A stuck pipe event occurs when it is impossible to move or rotate a drill string in the wellbore. In such cases, a drill string cannot be retrieved without damaging the pipe or bottom-hole assembly (BHA). Two main mechanisms are responsible for stuck pipe events: pressure differences and mechanical blockage. In the first case, if annular pressure is greater than the pressure in the formation being drilled, the drill string "sticks" to the wellbore wall due to such pressure difference. The contact surface area also plays a role in such kind of sticking.
Conversely, the concept of mechanical sticking describes the restriction or prevention of drill string movement for other reasons, for example, debris in the wellbore, anomalies in the geometry of the wellbore, key-ways, seal formation due to poor hole cleaning, unfavorable properties of the drilled formation.

Early detection of stuck pipe incidents in real-time will lead to reducing the downtime and costly corrective actions. Therefore, intensive efforts have been developed to develop methods to identify the stuck pipes' early symptoms in the last decades. One of these methods is torque and drag (T&D) modeling. Nowadays, T&D modeling is not standard because it requires real-time merge and contextual data of heterogeneous frequency and quality and recalibration. These results are not easily accessible to the user. The fundamental weakness of the current torque and drag models is its constant curvature wellbore trajectory. These trajectories mean that the drill string's bending moment does not change smoothly at survey points, which means that some contact forces and Axial loads are not present in the model.

The alternative approach is to apply neural networks (deep learning) and machine learning (ML) algorithms to solve real-time torque and drag issues. This approach can be more robust because the inputs can be automatically entered in real-time into a model that will generate the appropriate outputs.

1.2 Challenges

If early signs are detected in most stuck pipe events and measures are taken, an incident is preventable. Time is critical in such cases to take avoidance or remediation measures. For this reason, suck pipe early signs detection is of great importance. Several attempts to detect preliminary signs and prevent developing stuck pipe incidents using machine learning were made.

Recently, attention was drawn to the use of different data-driven techniques for stuck pipe prediction. The most commonly used techniques are statistical methods, ML models (based on Artificial Neural Network (ANN)), Support Vector Machines (SVM), Linear Regression), and hybrid approaches.

There were several attempts to use statistical learning methods to predict and classify torque and drag during drilling. Usually, such a predictive model does not need the new BHA or bit data but only surface parameters. However, such models require large amounts of input data.

Despite the advantages of the ML models, they still have few remarkable shortcomings, which can be solved using this thesis's approach. The first issue is related to the data set used to build the predictive model, and the existing methods require the historical stuck pipe incidents data for building the models. The other issue is related to the data set for training the model; most of them used big data to significantly impact the data's model performance. For the method that uses classification algorithms, selecting the data windows for training the algorithms can be a big challenge.

1.3 Objectives

Based on the disadvantages denoted in the previous section, the main goal of the thesis is to develop a model based on a machine learning concept focused on real-time detecting the early signs of impending stuck pipe by real-time monitoring the pertinent surface drilling parameters.

The basic methodology of the proposed model is demonstrated in Figure 1.

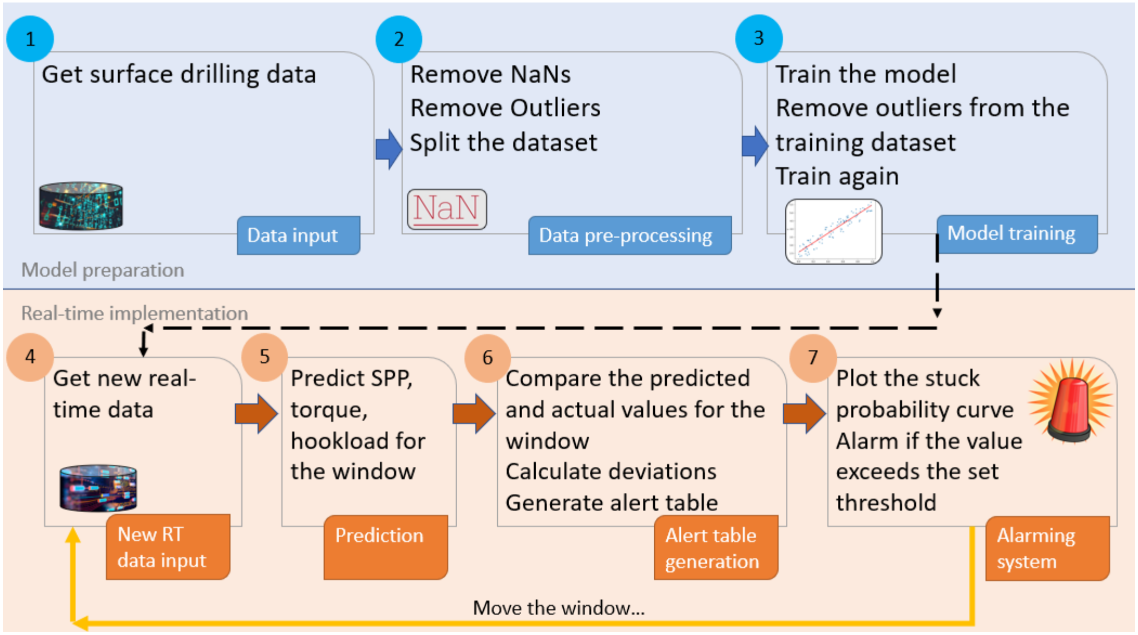

Figure 1: Basic Working Principle of the Developed Model

1.4 Thesis Structure

To achieve the defined goal of the thesis, the following steps are developed to be the main focus of the thesis:

- Literature review of main Machine learning algorithms.
- Literature review, analysis, and comparison of methods and models used for real-time stuck pipe detection approach based on data-driven technique.
- Data preparation to build the predictive models.
- Model building, tuning, performance evaluation.
- Real-time stuck pipe probability calculation, alarming system development.
- Model validation.

Chapter 2 Machine Learning Application on Stuck Pipe Events Detection

2.1 Overview

Machine Learning is an extensive subset of artificial intelligence that studies methods for building learning-capable algorithms. Commonly, there are two main types of training. Inductive training is based on identifying dominant patterns from particular empirical data. Deductive learning involves formalizing the knowledge of experts and transferring it to a computer in the form of a knowledge base. Deductive learning refers to expert systems, so machine learning and use case learning can be considered synonymous. Machine learning is at the intersection of mathematical statistics, optimization methods, and classical mathematical disciplines and has its specificity associated with computational efficiency and overfitting problems. Many inductive learning methods have been developed as an alternative to classical statistical approaches. Many methods are mostly related to the extraction of information and data mining.

The theoretical sections of machine learning are combined into a separate area, the theory of computational learning (Computational Learning Theory, COLT). Machine learning is not only a mathematical but also an engineering discipline. Pure theory, as a rule, does not immediately lead to methods and algorithms applicable in practice. However, to make them work well, additional heuristics have to be invented to compensate for the discrepancy between the assumptions made in the theory and the real problems. Furthermore, almost no research in machine learning is complete without an experiment on a model or actual data, confirming the practical workability of the method.

This chapter will start by giving a short introduction to the most common type of machine learning algorithms. Then, the recent data-driven applications for the stuck pipe detection approaches are discussed. In general, they can be classified into four categories:

1. Statistical learning methods
2. Linear regression methods
3. ANN approaches
4. Hybrid approaches

2.2 Supervised Learning Algorithms

2.2.1 Linear Regression

In mathematical statistics, linear regression is a method for approximating the dependencies between input and output variables based on a linear model. It is part of a broader statistical technique called regression analysis.

In regression analysis, input (independent) variables are also called predictor variables or regressors, and dependent variables are called criterion variables.

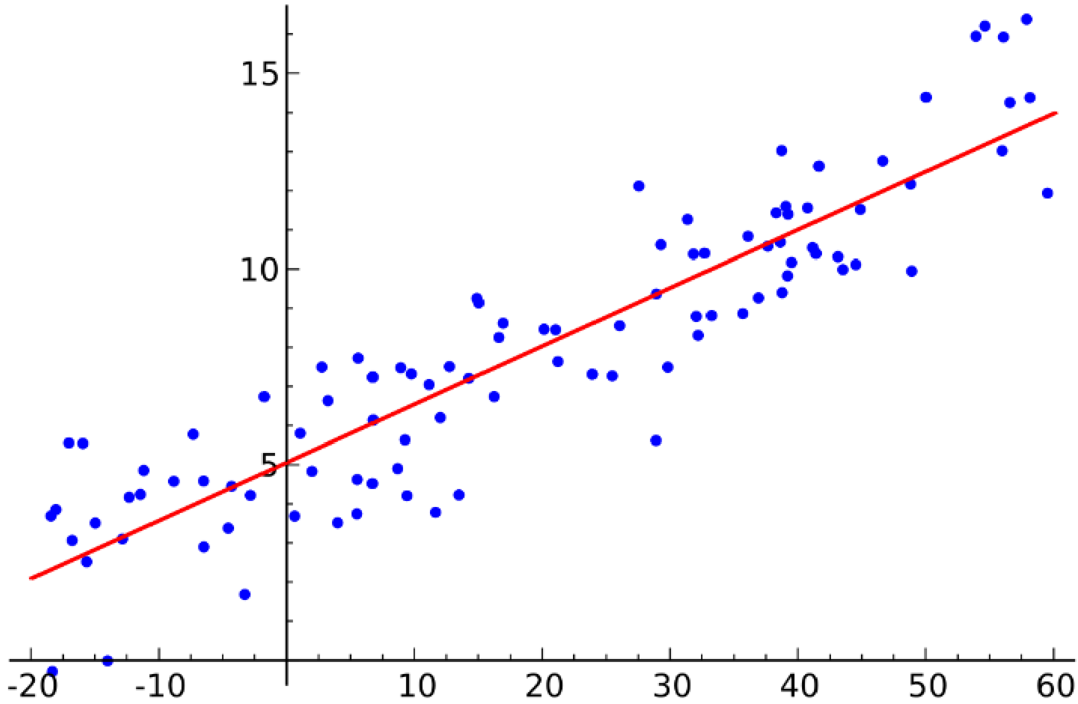

Figure 2: Linear Regression Example (Source: Wikipedia)

If the relationship between one input and one output variable is considered, then simple linear regression occurs. The regression equation is determined as y=ax+b. As a result, the corresponding straight line constructed is named the regression line.

Odds a and b, also called model parameters, are determined so that the sum of the squares of the deviations of the points corresponding to the actual observations of the data from the regression line would be minimal. Therefore, the odds are usually estimated using the least-squares method.

If a relationship is sought between inputs and one output variable, then multiple linear regression takes place.

Linear regression was the first type of regression analysis thoroughly studied and got widespread in practical applications. The linear regression gets popular because, in linear models, the estimation of parameters is more superficial and also because the statistical properties of the estimates obtained are easier to determine.

Linear regression has many practical uses. Most apps fall into one of two broad categories:

- If the goal is forecasting, linear regression can fit the model to the observed dataset.

- Suppose the goal is to explain the volatility of the output variable. In that case, linear regression analysis can be applied to quantify the strength of the relationship between the output and input variables.

2.2.2 Support Vector Machines

This method is used as a classification method to adapt to torque data and use non-linear classification boundaries to classify them effectively. Using kernels helps to map SVM input into large feature spaces implicitly. The efficiency of SVM classification depends on the kernel, kernel parameters, and field parameters. The effectiveness of SVM in characterizing text has been discussed by Joachims (1998). Using SVM for pattern recognition or classifying the torque trend can be helpful in drilling, especially for use as a predictive tool.

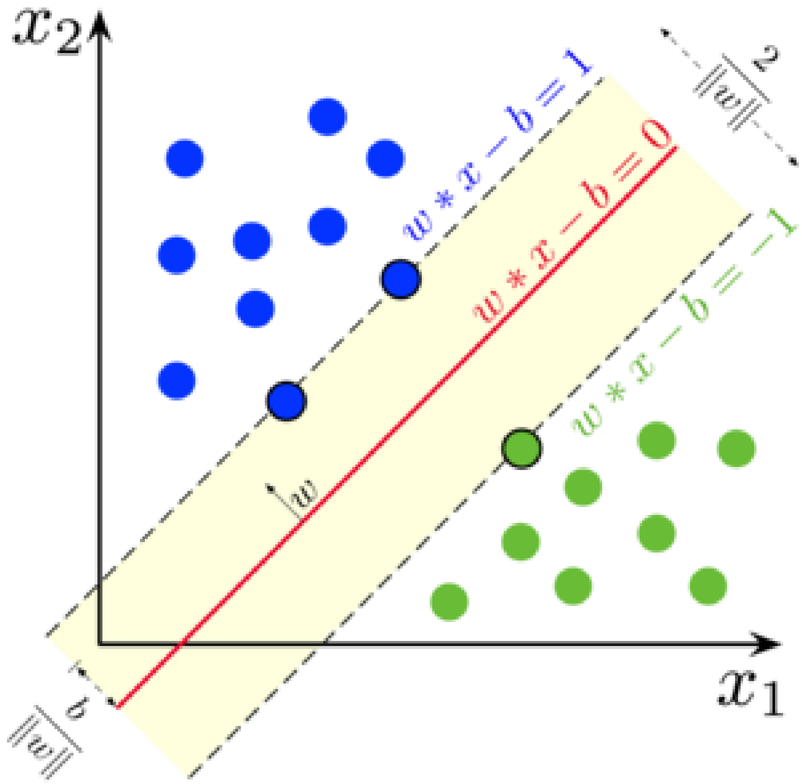

Figure 3: Support Vector Machine Example (Source: Wikipedia)

Support Vector Machines are a family of supervised learning-based binary classification algorithms that use linear hyperplane partitioning of feature space.

The method's main idea is to map the vectors of the feature space, representing the classified objects, into a space of a higher dimension. Because in the space of higher dimension, the linear separability of the set turns out to be higher than in the space of lower dimension. The reasons for this are intuitively clear: the more features are used for object recognition, the higher the expected recognition quality.

After being transferred to a space of greater dimension, a separating hyperplane is constructed in it. Moreover, all vectors located on one "side" of the hyperplane belong to one class, those located on the other - to the second. Two auxiliary hyperplanes are constructed on both sides of the main dividing hyperplane, parallel to it and at an equal distance from it. The distance between hyperplanes is called the gap.

The task is to construct a separating hyperplane in such a way as to maximize the gap - the area of the feature space between the auxiliary hyperplanes, in which there should be no vectors. It is assumed that the separating hyperplane constructed according to this rule will provide the most confident separation of classes and minimize the average recognition error.

The vectors that fall on the boundaries of the gap (that is, they will lie on the auxiliary hyperplanes) are called support vectors (which gave the name to the method).

2.2.3 Random Forests

Random forest is a machine learning algorithm proposed by Leo Breiman and Adele Cutler, which uses a committee (ensemble) of decision trees. The algorithm combines two main ideas: the Breiman bagging method and the random subspace method proposed by Tin Kam Ho. The algorithm is used for problems of classification, regression, and clustering. The main idea is to use a large ensemble of decision trees, each of which by itself gives a poor quality of classification, but due to their large number, the performance is good.

A Random Forest is a composition (ensemble) of many decision trees, which reduces overfitting and improves accuracy compared to a single tree. The forecast is obtained by aggregating the responses of many trees. Trees are trained independently of each other (on different subsets), which solves the problem of constructing identical trees on the same dataset and makes this algorithm very convenient for distributed computing systems.

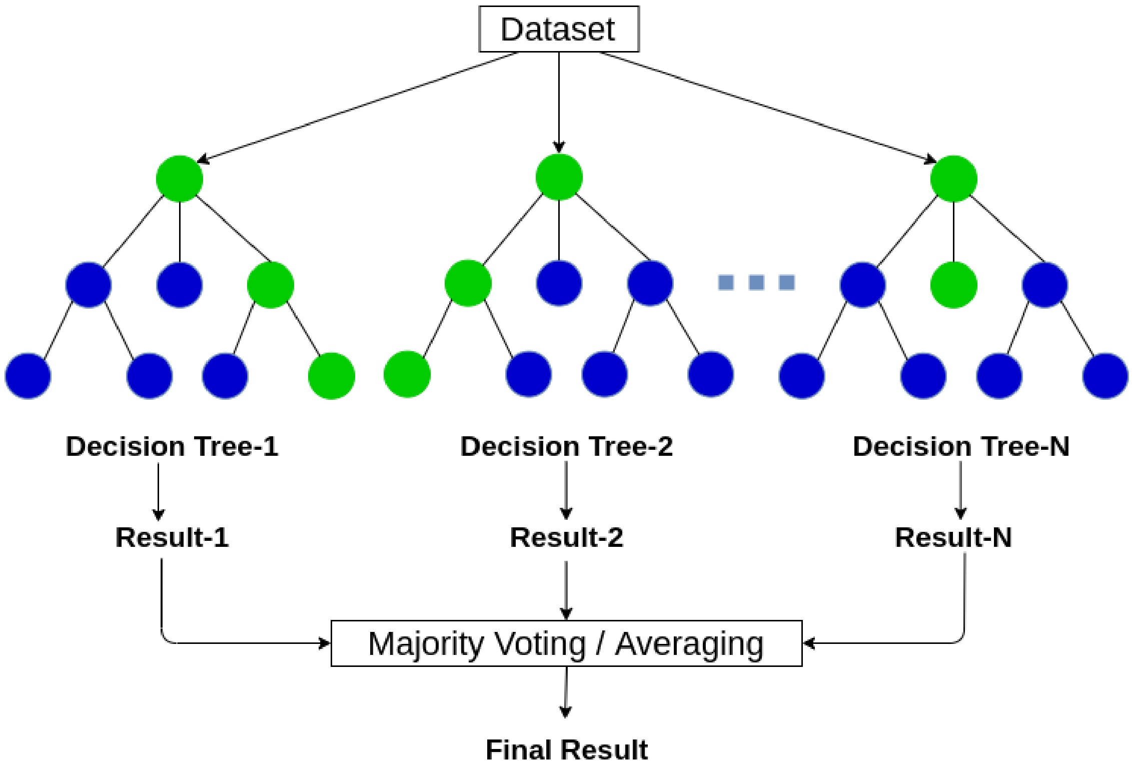

Figure 4: Random Forest Example (Source: analyticsvidhya.com)

For bagging (independent learning of classification algorithms, where the result is determined by voting), it makes sense to use many decision trees with a sufficiently large depth. During the classification, the final result will be the class for which most trees voted, provided that one tree has one vote.

Suppose that a model was formed with 500 trees in the binary classification problem. For instance, 100 trees indicate the zero class, and the remaining 400 indicate the first class. In this case, the model will predict the first class. If Random Forest is used for regression problems, choosing the solution that most trees voted for would be inappropriate. Instead, it chooses the average solution across all trees.

Random Forest (due to the independent construction of deep trees) requires many resources. Limiting the depth will damage accuracy (to solve complex problems, one needs to build many deep trees).

The training time for trees increases approximately linearly with their number. So, increasing the height (depth) of trees does not have the best effect on performance. However, it increases the efficiency of this algorithm (although at the same time, the tendency to overfitting increases).

2.2.4 Artificial Neural Networks

A neural network consists of artificial neurons connected in a certain way and the external environment through connections, each of which has a specific coefficient. The value arriving through it is multiplied (these coefficients are called weights).

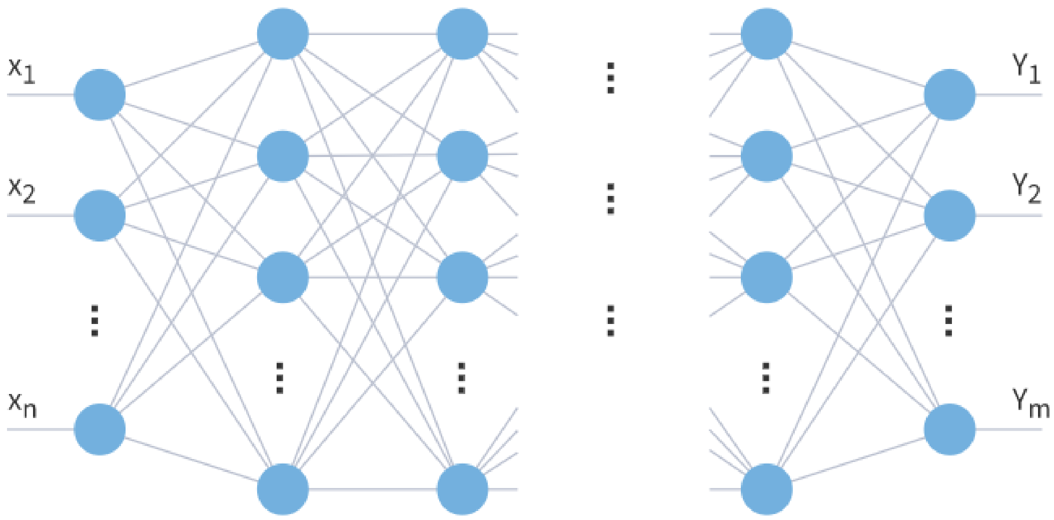

Figure 5: Typical Neural Network Architecture (Source: Wikipedia)

Neural networks can be implemented both in software and hardware (neurochips, neurocomputers).

In the process of functioning of a neural network, data is converted, the specific type of which is determined by the weights of intraneuronal connections, the type of activation function of neurons, architecture, and configuration of the network

Neural networks are machine learning-based models consisting of the iterative adjustment of the network weights according to the learning algorithm's rule. The rules can be used while constructing supervised learning (for multilayer perceptrons) and unsupervised (for Kohonen networks) neural networks.

Most often, neural networks are used to solve the following tasks:

- Function approximation - recovering functional dependencies from training data.
- Classification - determination of the belonging of the input object, represented by the feature vector, to one of the predefined classes.
- Clustering is a grouping of objects based on the proximity of their properties.
- Forecasting - predicting a value
- Optimization - finding a solution that satisfies the system of constraints and maximizes or minimizes the objective function.
- Associative memory is content-addressable memory used in superfast search systems.
- Control is the calculation of such an input effect on the system, at which it follows the desired trajectory.
- Speech recognition and machine translation.

Neural networks are widely used in data analysis, and neural network modules are part of almost any analytical platform. The main tasks solved using neural networks in data analysis are numerical prediction, classification, clustering, and forecasting.

The first working algorithm for learning artificial neural networks, the delta rule, was proposed in 1949 by Donald Hubb. The first practically functioning neural network (single-layer perceptron) was developed by F. Rosenblatt in 1958.

2.3 Classification Algorithms

2.3.1 Naive Bayes

Bayesian algorithms are a group of probabilistic classifiers used in ML. The method is based on applying Bayes' theorem. It is one of the first algorithms used for machine learning. The algorithm is suitable for binary and multiclass classification and allows for making predictions and forecast data based on historical results. A classic example is spam filtering systems. Such systems used Naive Bayes up till 2010 and showed satisfactory results. However, when Bayesian poisoning was invented, researchers started to find other ways to filter data. The Bayes theorem describes how the occurrence of an event impacts the probability of another event.

For example, this algorithm calculates the probability that a particular email is a spam based on the specific words. Thus, if the algorithm detects the specific words, there is a high possibility that the email is spam. However, the Naive Bayes algorithm assumes that the options are independent. Thus, the algorithm is called naive.

2.3.2 Multinominal naive Bayes

Multinomial Naive Bayes is usually applied for document classification based on the frequency of certain words present. However, bayesian algorithms are still used for text categorization and fraud detection. Moreover, they can be applied for machine vision (for instance, face detection), segmentation of the market, and bioinformatics.

2.3.3 Logistic Regression

Even though the name of this method might seem contra-intuitive, logistic regression is one of the classification algorithms. Logistic regression is a model that makes predictions using a logistic function to find the dependency between the output and input variables. Statquest made a great video explaining the difference between linear and logistic regression, such as obese mice.

2.3.4 Decision Trees

A decision tree is a simple way to depict a decision-making model represented by a tree. The method is easy to understand, interpret and visualize. Also, decision trees demand little effort for data preparation.

However, they also have a significant disadvantage. The trees can be unstable because of even the most negligible variance in data. Moreover, the over-complex trees do not generalize things well; it is called overfitting. Bagging, boosting, and regularization help to fix this problem.

The main elements of every decision tree are:

- Root node. The root node asks the main question. It has the arrows pointing down. However, no arrows are pointing to it.
- Branches. A subsection of a tree is termed a branch.
- Decision nodes. These are the subnodes for the root node. The decision nodes can be splitting into more nodes.
- Leaves or Terminal nodes. These nodes do not split. Instead, they represent final decisions or predictions.

Also, it is crucial to mention splitting, which divides a node into subnodes. Decision tree algorithms are referred to as CART (Classification and Regression Trees). These trees can work with both categorical or numerical data:

- Regression trees are primarily used when the variables have a numerical value.
- Classification trees can be implied when the data is categorical.

Decision trees are intuitive to understand and imply. For this reason, diagrams are commonly applied in a broad range of industries and disciplines.

2.4 Statistical learning Method to Detect Stuck Pipe Incident

A statistical model can be used to predict torque and drag while drilling. This model includes elements of statistical learning. Statistical Learning Method (SLM) is a data-driven model in which the forecast depends on the data available for the model.

One of the examples of statistical learning method implementation was done by Hedge et al. The dataset used in this article has been subdivided to assess the errors accurately. Typically, to avoid over-fitting the data, the dataset is divided into a training set, a validation set, and a test set. The training set is used to create the model. The validation set is responsible for cross-validation and limitation the number of parameters in the model to determine the most critical parameters. Finally, a test set is applied to "test" a model to establish its accuracy. The training and validation set is randomly divided into 0-3900 feet of data.

The model provides low error rates and the ability to learn as more data is acquired. Since the goal is to accurately predict and classify torque and drag, measuring the error will estimate the model's accuracy. The root mean square error (MSER) of the model on the test data is the primary accuracy estimator in the case. The data was formatted using the *dplyr* toolkit developed by Wickham (2013), which provides an efficient way to clean and organize data. Tyler and Dakota formations are presented as an example of the working principle. In both reservoirs, machine learning regression (MLR) was performed using the same predictors. The Dakota Formation has 158 data points, and the Tyler Formation has approximately 875 surface measurement data points. In each case, the first 20% of the data was a training set, the next 10% was a validation set, and the rest of the data set was a test set. MLR results for predicting downhole torque using surface parameters are confirmed where the MSER for the Dakota formation was 18.53, and for the Tyler formation, it was 1430.575, as shown in Figure 6. The first plot is the MLR forecast for the Dakota Sandstone, and the second plot is for Tyler.

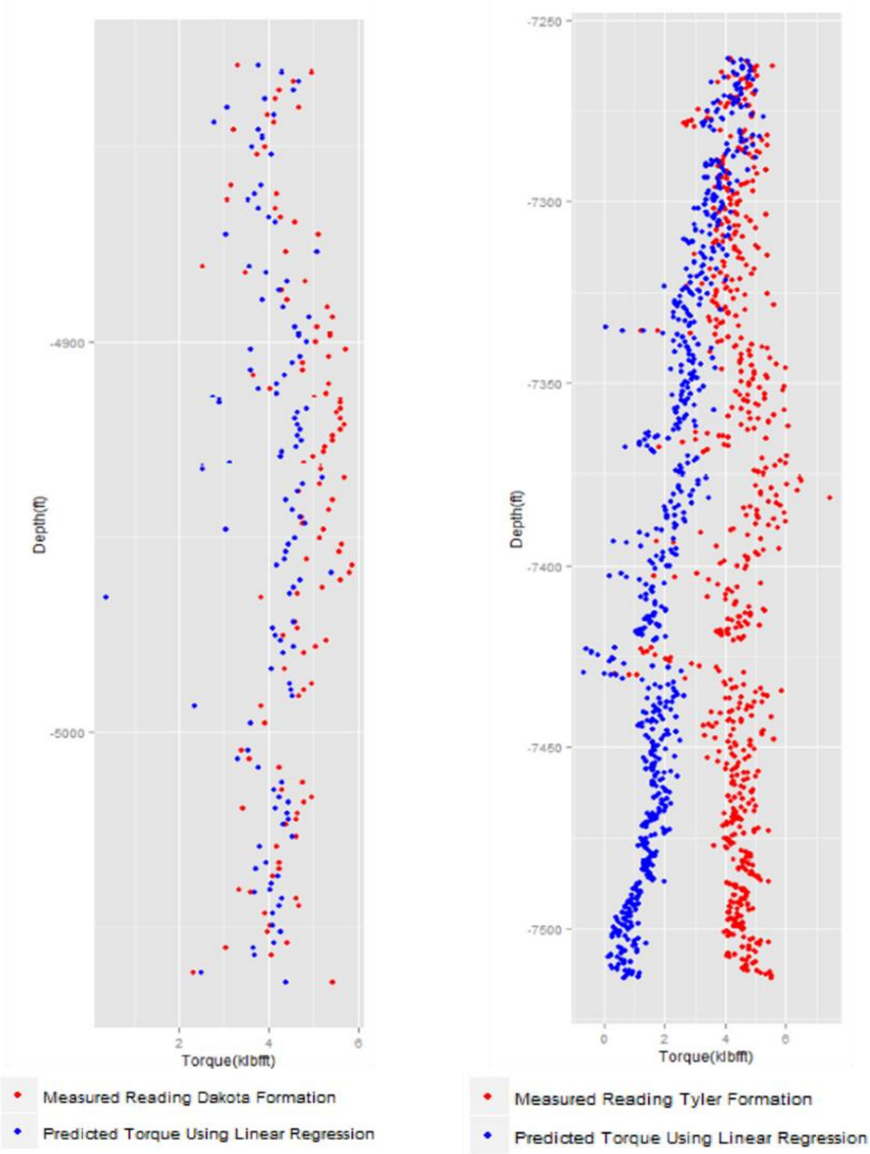

Figure 6: MLR Used to Predict Downhole Torque in Dakota and Tyler Formations (Hedge et al.)

Based on the results obtained in a limited field study, predicting torque during operations in real-time is possible using statistical learning methods. The use of statistical teaching methods has the advantage of predictive utility over conventional empirical methods. In addition, the SLM can adapt to changes during drilling and improve the model as the well is drilled. However, after making changes, the model will have to be re-trained.

Brancovic et al. developed three indicators related to various physical phenomena to recognize stuck pipes and events that precede them. Then, the researchers used these indicators (and other well logging inputs) to explore a statistical model to anticipate stuck pipe events. Various data are available for this task, namely time-based data annotated by drilling operators and logging data obtained using surface equipment. Unfortunately, the consistency of these data is sometimes questionable due to the subjectivity of operators' assessments.

The mud-log reports the measurements of several parameters related to the drilling process (some variables are not measured but computed from other measurements), sampled every 5 seconds. Numerous variables related to the drilling depth are available. In this work, the following mud-log variables were used:

- DBTM - bit depth along the drilling direction
- BPOS - position (height) of the traveling block, which supports the drill pipe
- HKLA - (average) tension on the cable from the drawworks (hook load)
- TQA - (average) rotary torque applied to the drill string (taken at the rotary table or the top drive motor cable)
- RPMA - (average) rotary speed
- SPMT - total pump stroke rate
- SPPA - (average) pressure of the mud on the standpipe

The methodology of such an approach is the following. The three proposed detectors (standpipe pressure (SPP), Torque & RPM, Hook load & Block Position) can be used in post-processing to verify the wellbore problems. More importantly, these detectors can also be used while drilling to identify minor and significant drilling problems, the former also representing possible precursors of adhesion. In this section, the researchers tentatively discuss building a prediction model based on these detectors. The basic idea is to use the available drilling data to match detectors (and raw logs) with an artificially generated alarm by assigning the highest value for each documented stuck and gradually decreasing value backward in time and upwards in depth. Indeed, the system can sound the alarm (and increase in level) as drilling gets closer to the choice, either in time or in-depth. The inputs and outputs are aggregated into 4m bins based on depth and averaged before training to reduce variability and noise in the data.

Figure 7 shows a graphical representation of the predictive model results obtained at three successive points during the drilling phase, resulting in a particularly severe stuck pipe. As drilling goes, the model (trained on other wells and the previous phase of the same well) indicates initially a relatively healthy status of the wellbore (left picture) until the 200th bin is reached. Then, several depth bins are marked with a high level of risk (middle picture). Although subsequent actions could change the risk level of deep bins only slightly in this area, drilling was resumed (the bit was run down to bin 240), leaving a high-risk area behind. Unfortunately, it turned out to be a poor choice as a stuck pipe incident occurred. Sticking occurs when the bit is several meters below a critical region that is compatible with the position of the large drill string elements concerning the bit. By appropriately reworking the critical zone before resuming drilling, sticking could be avoided.

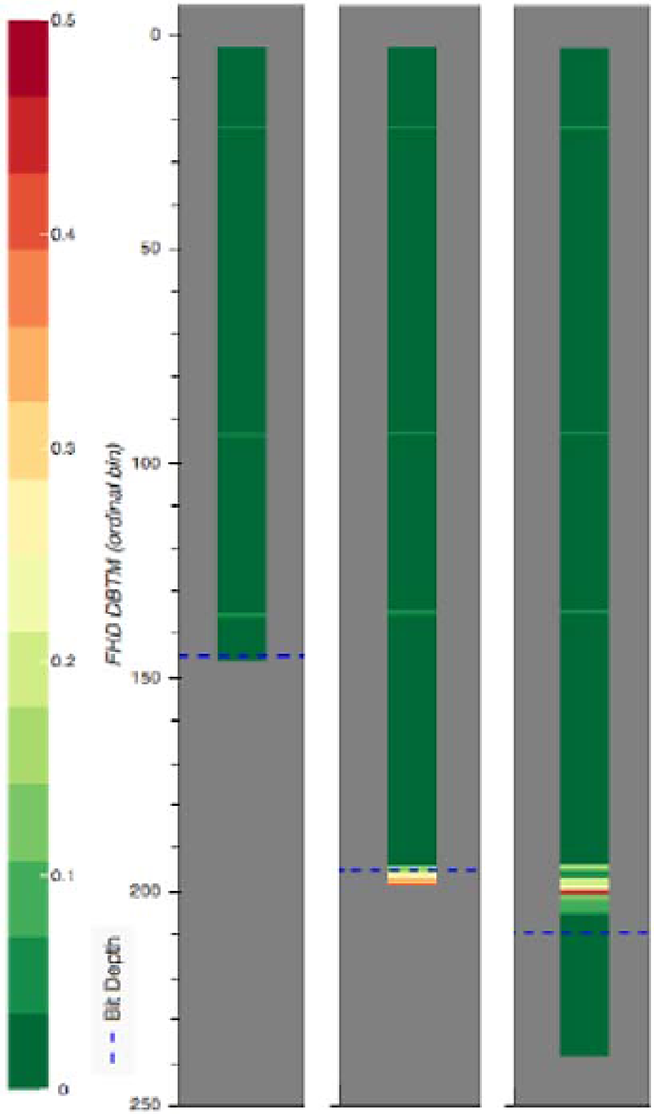

Figure 7: Screenshots of the web Application Showing the Predicted Well Status of the Wellbore at three Subsequent Timesteps (Brancovic et al.)

Based on the mud-log data, three different indicators have been developed to capture three different physical phenomena associated with the stuck occurrence. The first one is for detecting difficulties in linear movement of the drill string, and the second is for detecting problems with rotation. The third indicator detects unexpected pressure spikes in the annulus.

2.5 Linear Regression Method to Detect Stuck Pipe Incident

Ahmed et al. used a moving window-based regression machine learning approach to develop a model to detect early warning signs of a conventional pipe stuck mechanism during drilling operations. The unsupervised machine learning algorithm is programmed to automatically detect deviations in trends in drilling parameters in real-time and predict potential pipe sticking and preliminary transmission of observation results in the form of warnings to engineers for preventive, corrective action. In the first place, it invariably increases the chances of avoiding stuck pipe incidents and timely release of the pipe if possible. Since a stuck pipe accounts for a significant part of the NPT, time is of the essence in such cases, as the adverse reaction to a stuck pipe can easily make the situation worse.

Several signs are used in the model:

- Hook load peaks between connections
- Non-smooth block position curve
- Increased standpipe pressure
- Torque spikes
- Dogleg severity Indicator

In the case study, the data archived in real-time was passed through the model in replay mode. As seen in Figure 8, the model generated several sets of observations that should be detected by the model when running from the new BHA. As one gets closer to the stuck point, the severity and probability of a pipe stuck increases. The model generated two warnings before the pipe stuck. While these alerts were retrospective, they were generated in real-time data playback. It means that if this model were deployed while drilling the well, it would detect an abnormal trend and alert the driller to take actions to prevent pipe sticking. Thereby NPT could be possibly reduced.

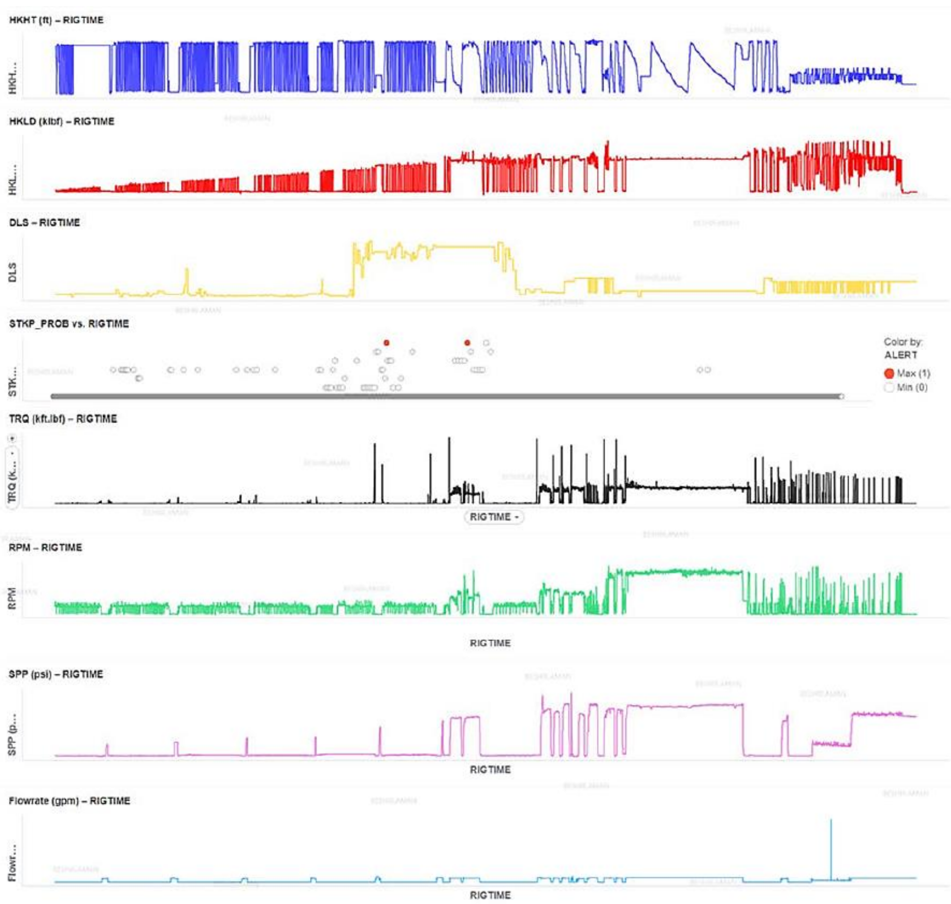

Figure 8: Two Stuck pipe Alerts with high probability before an Incident in Well-A (Ahmed et al.)

A methodology was presented using linear regression with a moving window to record and monitor the classic warning signs of pipe sticking. It is believed that while drilling a well, it informs the driller about its condition. The model presented in the study demonstrated that stuck pipe alerting can be automated, and early warning signs of stuck pipe can be identified from drilling parameters. For operational reasons, one or two of the alerts recorded in this study were false. However, the model shows overall promising results (accuracy of 0.67, the sensitivity of 0.8) for the ten validation wells.

This approach can be further improved by refining existing symptoms, including additional symptoms and scenarios, and optimizing model performance for real-time deployment, ultimately helping to reduce the uncertainty in the likelihood of pipe stuck.

The approach using logging-while-drilling (LWD) measurements in real-time to predict pipe sticking was proposed by Hess. The paper explains how to interpret LWD measurements to help determine if mud weight, which is about to be increased, is helpful if the penetration rate should be slowed. There are three main groups of variables that the model uses to predict possible sticking:

- Variables available pre-run (hole size, tool size, number, and spacing stabilizers)
- Variables available during running (uncontrolled) (measured depth, total vertical depth, equivalent circulation density, dogleg severity, Inclination, Temperature)
- Variables available while running (controlled) (rate of penetration (ROP), rotary speed, flow rate, mud weight, plastic viscosity of mud, yield point of mud, Gel strength of mud, Oil/Water ratio of mud, Chlorides in mud)

When the required data is collected, and extra variables are created, the information is fed into linear regression software. All items entered receive their coefficients, which are summed up further. If the sum is 0, the drill-string is assumed to be free; if the sum is 1, the stuck pipe accident occurs. Finally, the model sets every coefficient based on the previously entered wells used to train the model. An example of coefficients is shown in Table 1.

-16.457	Y Intercept	-0.029	2°/100ft-3°/100ft
-1.532	Insurance (0-No/1-Yes)	-0.011	3.01°/100ft-4°/100ft
0.078	Hole Size	-0.222	4.01°/100ft-5°/100ft
0.093	Tool Size	-0.043	5.01°/100ft-7.5°/100ft
-0.015	Hole Size-Tool Size	0.552	7.5°/100ft-10°/100ft
0.401	# Of Stabilizer In BHA	0.518	10.01°/100ft-15°/100ft
-0.033	Min Stabilizer Spacing	0.007	ROP
0.048	Max Stabilizer Spacing	-0.181	Times Back reamed between stands
-0.032	Distance From First to Last Stabilizer	-0.002	RPM
-1.259	Reamer In BHA (0-No/1-Yes)	0.119	Mud Weight
0.187	Surge ECD	0.051	Plastic Viscosity
-0.029	Mw Static	-1.503	Ln (Plastic Viscosity)
0.216	Surge-Mw Static	-0.078	Yield Point
-0.284	ACTECDX-Mud Weight	2.679	Ln (Yield Point)
-0.165	ACTECDX	-0.027	Plastic Viscosity+Yield Point
-0.011	Temperature	-0.04	Gel 10 Sec
3.078	Ln (Temperature)	-0.009	Gel 10 Min
0.026	Inclination	0.031	10 Min-10 Sec Gel
0.012	Dogleg Severity At TD	-0.011	Oil
-0.049	Highest Dogleg Severity	0.325	Oil/Water Ratio

Table 1: Model Equation Coefficients and Associated Variables (Hess 2016)

As a case study, the model was first trained using data from 42 wells of the Gulf of Mexico. Seven wells were used to test the model. The model predicted 3 out of 7 stuck pipe incidents correctly. The prediction was "BHA would get stuck permanently" for the other four wells while it was stuck permanently only in one case.

The shortcomings of this approach are the following. The model cannot be used for fields located out of the Gulf of Mexico because the model is constructed and trained for the particular wells typical for that area. Moreover, the model cannot effectively predict the incident if there are one or zero stabilizers in the BHA.

2.6 Hybrid Method to Detect Stuck Pipe Incident

The new method proposed by Zhang et al. integrates first-principles physics-based models, including a transient solid transport model, a drill string model (torque and drag model), and data-driven models. The proposed models are developed based on the analysis of field and experimental data. The physics-based model considers the basic rules of fluid mechanics, drill string mechanics, and multiphase flow during drilling operations.

Stuck pipe events in this paper are classified into two categories: stuck pipe incidents associated with hole cleaning, which are indexed by the HCSPI (Hole Cleaning Stuck Probability index), and incidents caused by other reasons, which are indexed by the GSPI (General Stuck Probability index). Real-time stuck pipe analysis and risk prediction generate two separate risk indices by analyzing torque data from three different sources: predicted torque with cuttings, T_c, predicted torque without cuttings, T_{nc}, and the measured out-of-hole free rotation torque, T_m. HCSPI is obtained by comparing the predicted torque with cuttings exposure and the predicted torque without cuttings. The GSPI is obtained by comparing the difference between the predicted torque adjusted for drill cuttings and the measured rotational torque at the bottom hole. The general workflow to predict the stuck pipe indexes can be summarized in Figure 9.

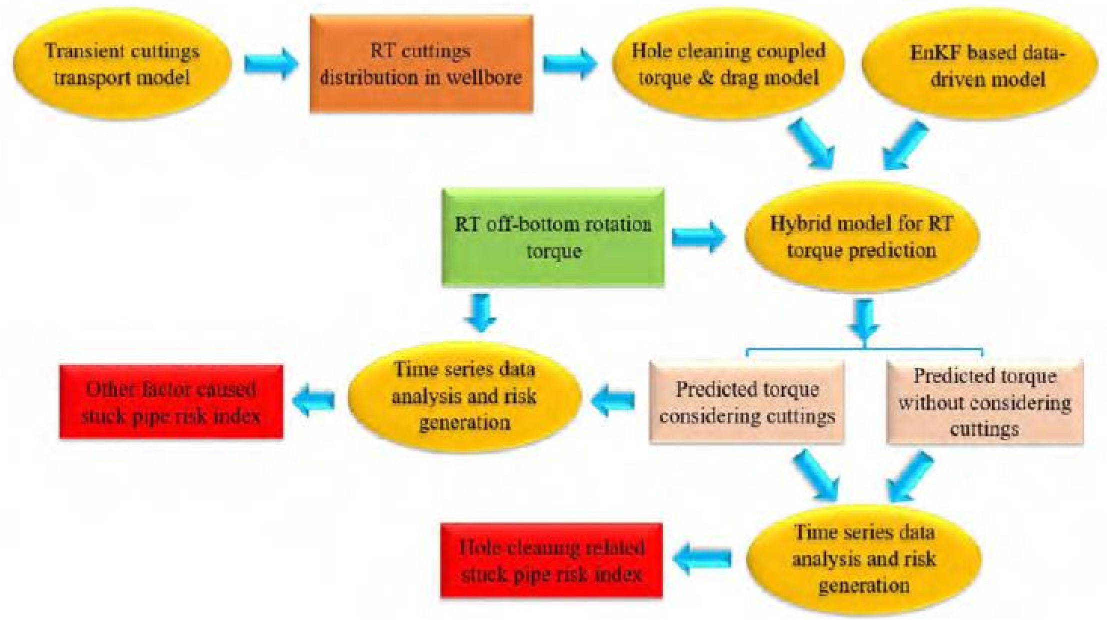

Figure 9: Workflow for the RT-risk Generation Model (Zhang et al.)

Case study. The measured torque gradually increases as drilling progresses from 1450 m to 1850 m. The difference between the predicted torque with cuttings and those without cut effect gradually increases, indicating that cuttings accumulate in the wellbore. At a depth of about 1850 m (22:30), the torque rapidly increased from 15 kNm to 20 kNm, indicating an emergency in the well. The drilling crew did not pay attention to the sharp increase in torque, and the drilling process continued. Another 150 m were drilled until the pipe was completely stuck. While drilling this 150m section, the rate of change of the measured torque also increases significantly, as shown in Figure 10.

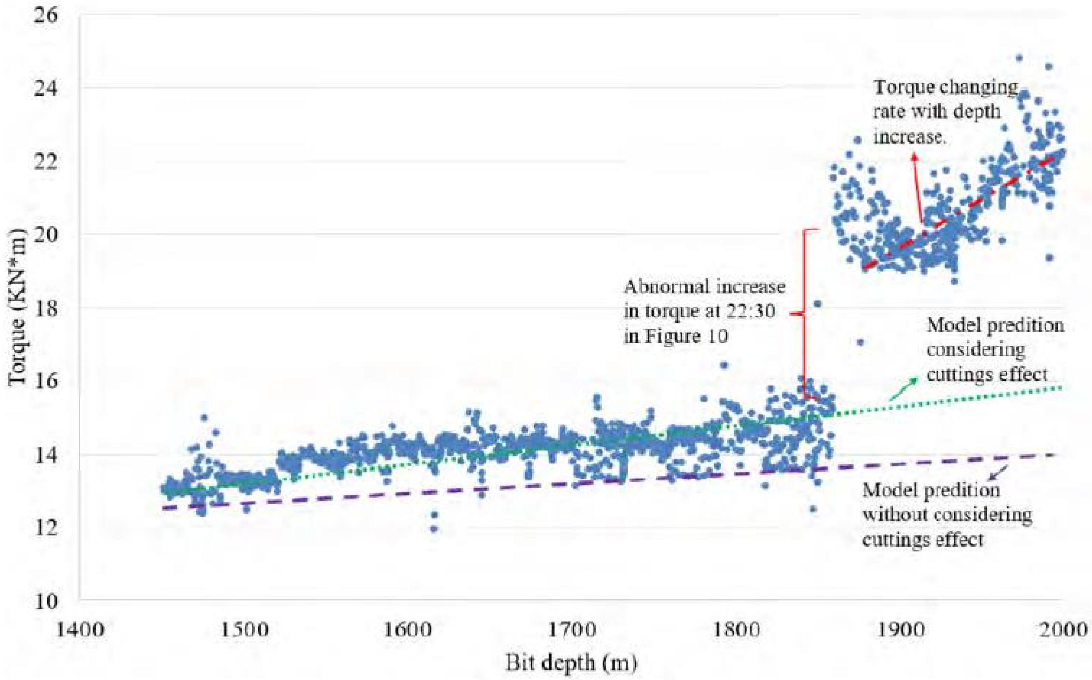

Figure 10: Analysis of torque changes with bit depth (Zhang et al.)

The proposed model for predicting pipe stuck was run on the data for post-drill analysis and algorithm validation—the risk increases in 4 hours before the incident, as shown in Figure 11. The HCSP stuck index associated with wellbore cleanup was already 0.5 at 20:00, and before the incident, it rises over 0.7. Since the ROP was about 200 m / h, which is relatively fast, and the well inclination was about 55 degrees (which is the most challenging place to clean the hole), the well was not cleaned sufficiently during the drilling process. Before 10:30 pm, the risk index for other causes of GeSP was shallow (0.2 to 0.3), indicating that no other significant factors could cause pipe sticking apart from cleaning the wellbore. However, with a sudden increase in torque, GeSP suddenly increased to more than 0.9, representing a high probability of pipe stuck occurrence.

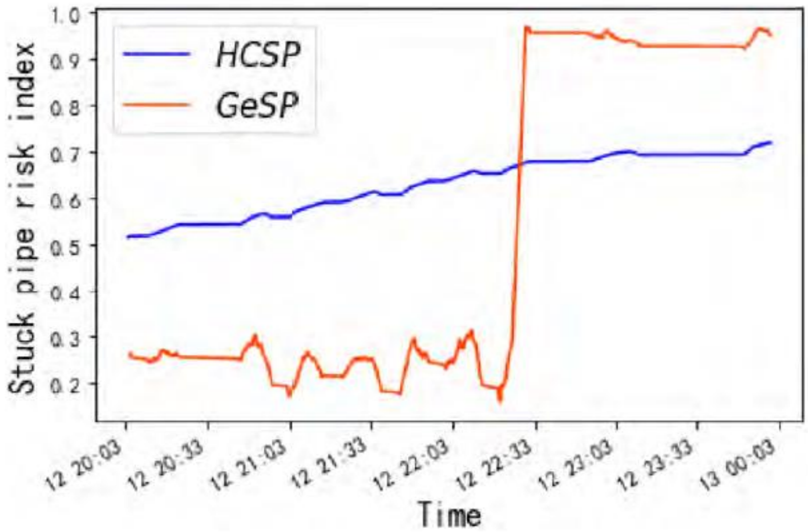

Figure 11: Risk indexes as They were Predicted by the Model (Zhang et al.)

The method to detect various downhole failures was presented by Skalle et al. The researchers used real-time drilling data and end of well reports to analyze the entire situation. All the available data, such as equipment characteristics, geology, challenges characteristics, actions used to tackle these challenges, were used. The whole ontology to recognize the problem and its early signs was developed. The model implies thirty-five dynamic agents extracting symptoms from the real-time data. The central methodology of developing such agents is the following:

- Find symptoms manually in historical real-time data and tag their depth
- Develop data agents in Matlab
- Evaluate agent performance

After doing so, the structure of the ontology was developed. To detect causes accurately, the model sets up the relations between each part of ontology to be bi-directional. Then, when the failure case occurred, the model estimates possible failure using detected symptoms. Several metrics such as path strength, explanation strength, and others were used to estimate failure-probability relations. As an example of ontology structure, the part of it is presented in Figure 12.

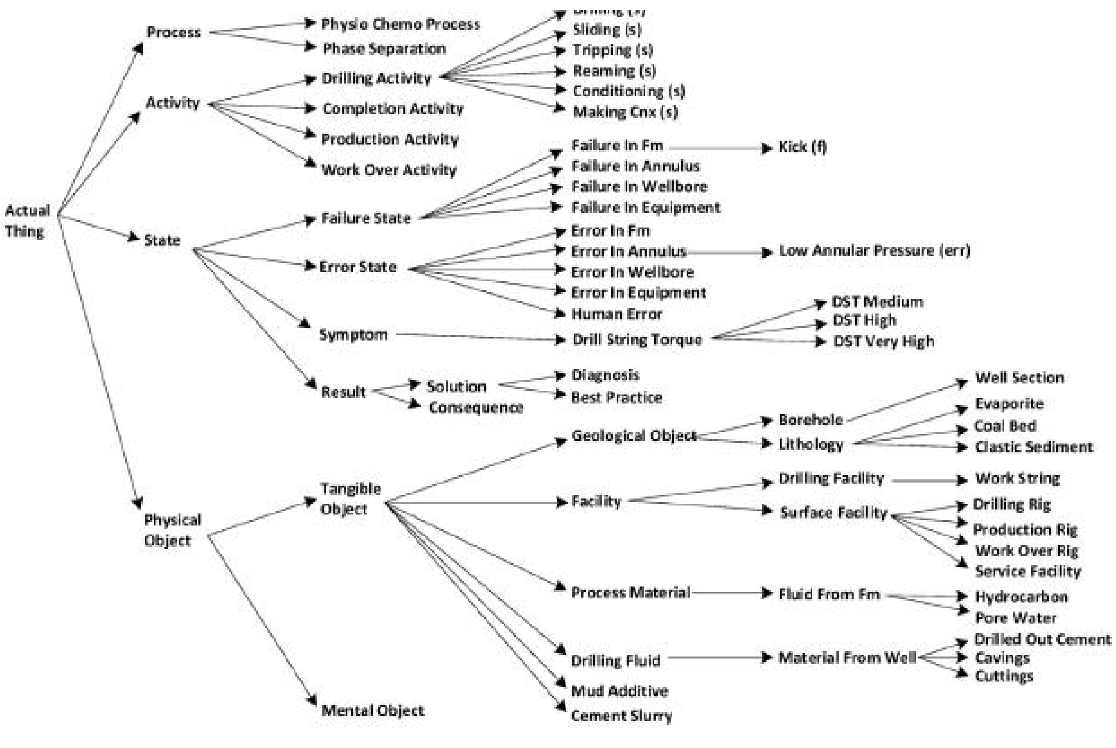

Figure 12: Part of Ontology Structure (Skalle et al)

The model is trained to recognize almost all drilling problems, but the example was provided only for motor stall detection. However, the same approach is used to detect other incidents. Based on analysis of all the factors which were monitored, the model initiated the case of a motor stall 12 hours before it happened.

Both models presented here have several shortcomings. As for the first example, such a model cannot accurately predict the stuck pipe events, which are not related to differential sticking. In addition, various downhole problems detection model requires enormous amounts of data and its preparation work. Moreover, when the model is completely set up, it is applicable only for offset wells if they have similar well construction, used equipment, lithology, and others. Otherwise, there is a high risk of false alarms.

2.7 ANN Method to Detect Stuck Pipe Incident

Tsuchihashi et al. explore real-time pipe stuck prediction using deep learning. The researchers propose a 3-D Convolutional Neural Network (CNN) approach with constrained depth data. The clip illustrates depth domain data in 2D histogram images with a unique time domain abstraction. This study uses 30 field well data prepared in multivariate time series - 20 for training and 10 for validation. These checks include six stuck cases, and the 3D-CNN model successfully detected early signs of stuck three times before it occurred. The portion of the data clip that aids in detecting the anomaly is indicated by a Gradient Weighted Class Activation Map (grad-CAM), providing a physical explanation of the black-box model. The binary labeled data from stuck pipe events was fed to the CNN. This binary labeled data helps the model to define crucial early signs and changes in its patterns.

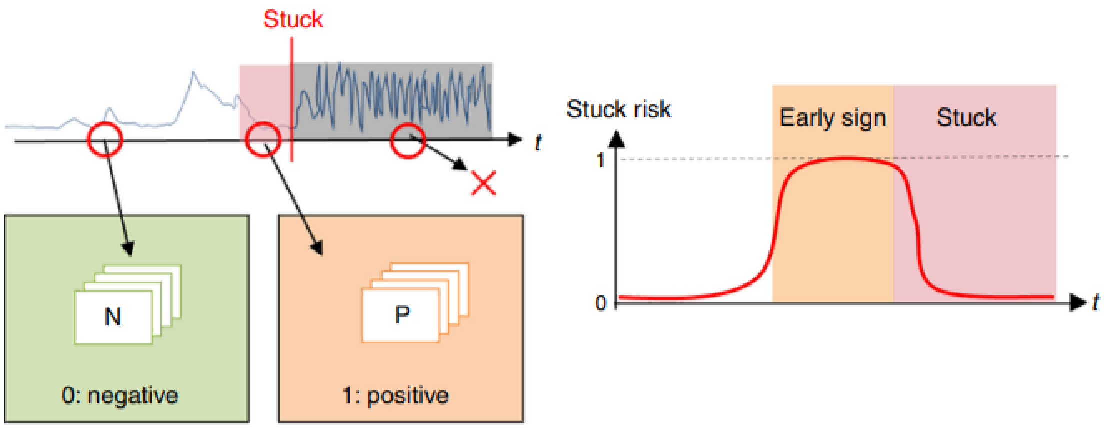

Figure 13: Convolutional Neural Network Application on Detecting Stuck Pipe Incident (Tsuchihashi et al.)

For the case study, real-time prediction of 1D-CNN and 3D-CNN is shown in Figure 14. In this case, 1D-CNN cannot detect features, whereas 3D-CNN does. On 3D-CNN output, the risk of stuck pipe was high 30 minutes before the incident occurred. It is visible in the figure that the first signs of sticking appeared earlier than an hour before the occurrence.

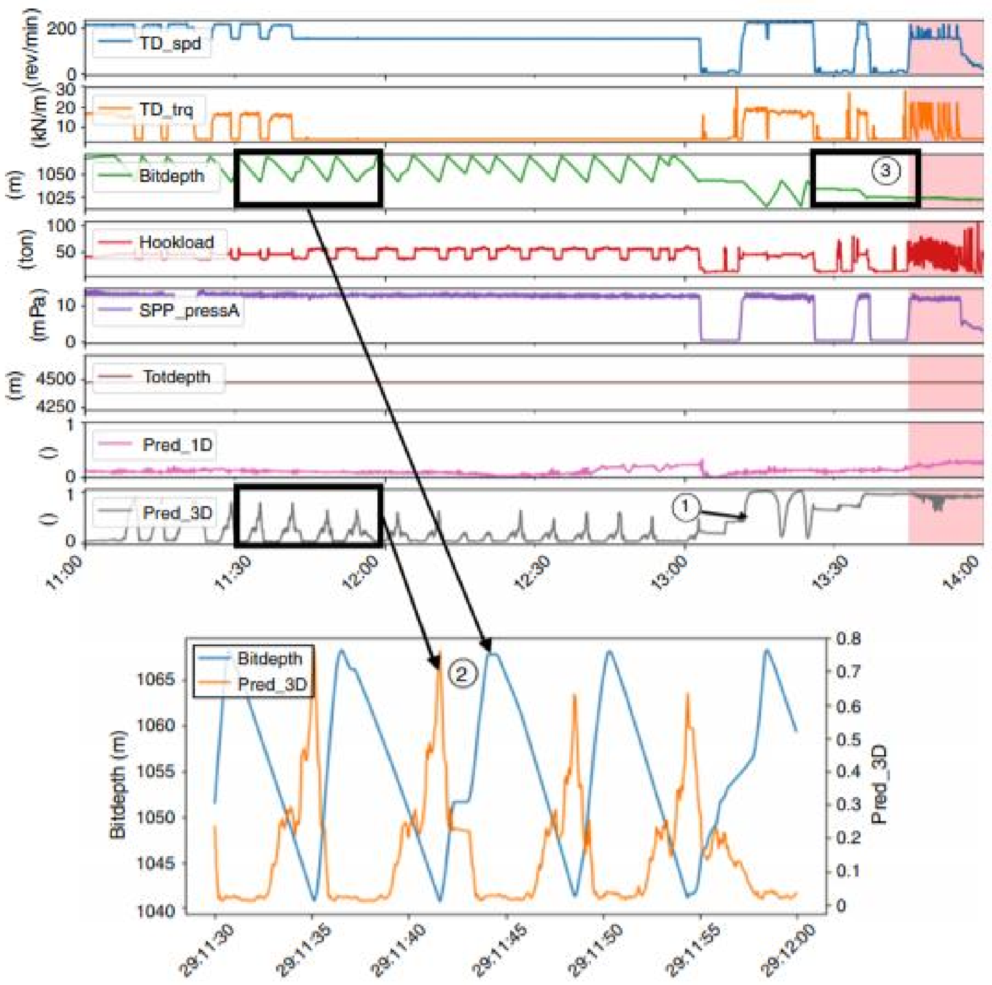

Figure 14: Real-time Stuck Risk Prediction and Validation (Tsuchihashi et al.)

There may be times when the early symptom did not exist due to a drastic change in the situation. Hence, spotting the early symptom would be impossible. However, differential pressure sticking is different from others. The process of embedding the drill pipe into the mud cake does not occur immediately. Nevertheless, this method still requires specialists to prepare the data accurately before training; it takes too much time. Moreover, this approach requires large amounts of data from various stuck pipe events.

Another attempt to use an artificial neural network to predict stuck pipe events was taken by Meor et al. The proposed Wells Augmented Stuck Pipe Indicator system is a data-driven approach, which uses historical drilling data and engineering knowledge to monitor and detect impending stuck pipe incidents. The model is based on real-time data, stuck pipe mechanism recognition, and operation time. The main objective of the model is to provide an alarming system that engineers will utilize for real-time drilling implementation.

The main principle of the model is the following. The cloud has three preliminary prepared models representing different stuck pipe mechanisms: differential sticking, sticking due to poor hole cleaning, and mechanical sticking because of hole geometry. However, the real-time implementation experience of this model is still unknown and cannot be evaluated.

The model is evaluated using qualitative assessments to obtain a holistic result. First, real-time drilling data is quantified using mean absolute percentage error and normalized root mean square error. It was then further analyzed qualitatively using the confusion matrix. Analyzed data includes proxy parameters, historical forecast data, warnings, and alarms generated for the well evaluation. These evaluations inform about the limitations and opportunities for further improvement and development of machine learning models.

The discussed methods based on ANNs have one significant disadvantage. The building and tuning of such models are challenging tasks. Furthermore, the ANN is extremely sensitive to input data, so it should be adequately prepared. Finally, the building and tuning of such models vary from well to well, so such solutions are not ready now to be utilized for real-time drilling.

2.8 Gaps in the Discussed Methods

As mentioned in the introduction chapter, current torque and drag monitoring approaches in stuck pipe detection have different shortcomings.

Statistical learning methods can predict torque and drag during drilling. Mostly it is a predictive model that does not need the new BHA or bit data but only surface parameters. However, this model requires large amounts of input data. Such kind of models may be improved by shortening the amount of learning data to a specified length in real-time for different formations to have a more efficient and stricter prognosis for the current situation in the well. Also, the combination of parameters used by statistical learning methods may lead the system to recognize other actions as possible signs of stuck pipe, leading to false alarms.

A moving window-based linear regression machine learning approach to detect early signs of pipe stuck during drilling was shown in one of the previous paragraphs. The proposed method uses unsupervised learning to detect trend spikes and calculates the stuck probability by weighting different parameters. However, such a system is also prone to false alarms because the contribution of different parameters to stuck pipe probability varies due to the situation. Moreover, using unsupervised learning does not allow recognition behavior of the spontaneous model, which can lead to unexpected results.

A hybrid approach integrates physics-based models, including a transient solid transport model, a drill string model (torque and drag model), and data-driven models. The model works well to recognize sticking caused by hole cleaning problems, but it cannot recognize other kinds of sticking mechanics.

ANN approaches have several shortcomings. First, ANN limits the number of input parameters. Second, neural networks are most effective in tasks for which a large set of disparate initial data is inherent. Thus, setting up a neural network for a specific job is a difficult task that deserves special attention. Moreover, neural networks require special data preparation, filtering. This fact makes these approaches unsuitable for real-time monitoring.

Chapter 3 Methodology

3.1 Overview

This chapter mainly focuses on describing the process followed to develop a model to detect the early signs of impending stuck pipe by real-time monitoring the three relevant surface drilling parameters. The model is built in a MATLAB R2020A environment. The basic workflow of the proposed model is shown in Figure 15.

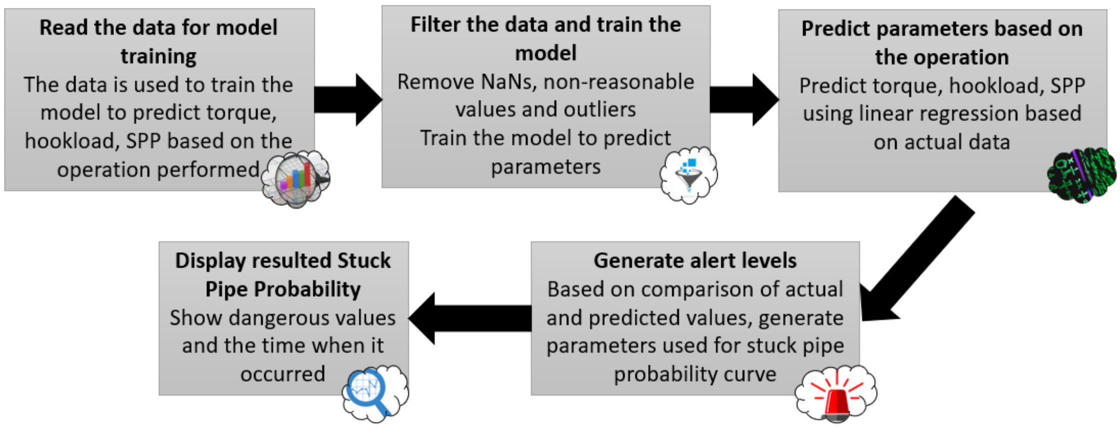

Figure 15: Basic Workflow of Developed Methodology

Generally, the model compares the actual data with predicted data to trace the deviations. Then, based on the severity of the detected deviation, an alert level is generated, which is later converted to a probability curve; eventually, based on generated stuck probability results, the risk of stuck pipes can be assessed.

The working principle of the developed model can be classified into three stages

1. Data Preparation
2. Predictive Models Construction
3. Real-time Implementation

Figure 16 shows the detailed workflow of the proposed model.

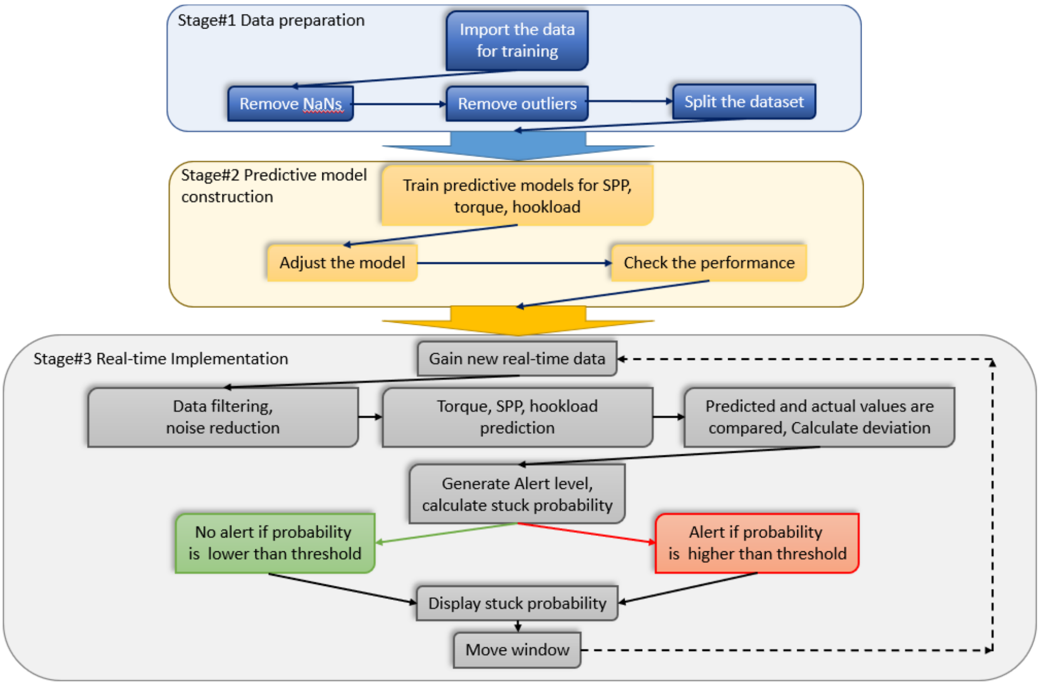

Figure 16: Proposed Model Detailed Workflow

3.2 Input Data Description

Various parameters are monitored during drillings, such as rotary speed, mudflow in, block position, hook load, and so on. Some parameters are predictable because they typically should follow the linear regression trend. So, the large amounts of data coming from the well may be used to build and train the models to predict such parameters' behavior, which may be helpful for further analysis.

Most sticking mechanisms are related to the buildup of cuttings bed in the annulus, increasing friction between the wellbore wall and drill string. Several researchers (Ahmed et al., 2019; Carpenter, 2016) report that the most proper way to monitor such dangerous wellbore conditions is to analyze trends of standpipe pressure, surface torque, and hook load in real-time. For the current application, these three main parameters will be used to be monitored and analyzed.

As a result, the following Table 2 summarizes the target parameters predicted by the model and the required inputs for each target (Hedge et al., Noshi et al.).

Input Data	Target
Weight On Bit, Average Rotary Speed, Rate Of Penetration	Surface Torque
Weight On Bit, Mud Flow In, Average Rotary Speed, Average Surface Torque, Block Position, Rate of Penetration	Hook load
Weight On Bit, Flow Rate, Rate of Penetration	Standpipe pressure (SPP)

Table 2: Summary of Proposed Targets and Their Relevant Inputs

3.3 Data Preparation

To improve the model's performance and accuracy, it is strictly recommended to review the data, detect outliers and prepare the data for further analysis. This part will explain the workflow of data preparation to predict required parameters correctly. The general workflow is presented in Figure 17.

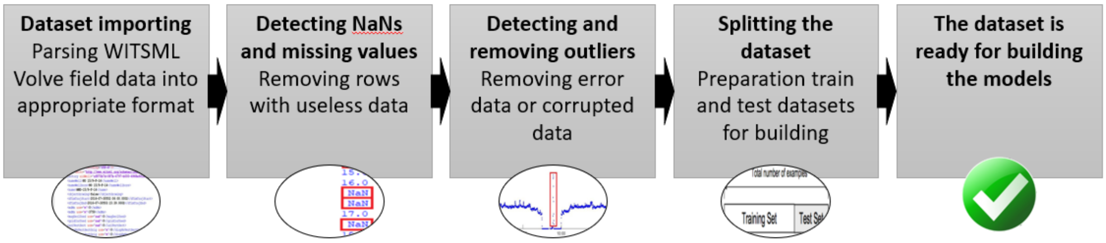

Figure 17: General Workflow of Dataset Preparation Stage

3.3.1 Input Data Uploading

Generally, the data coming from the well has the WITSML (Wellsite information transfer standard markup language) format. This format is a standard format of transmission data from the rig to the oilfield company office. Such data format cannot be used to train the model via machine learning algorithms. Therefore, it should be converted to .csv or .las etc. To convert the data to the appropriate formats, a process of «parsing» should be initiated. Parsing is an automated process of collecting data from the database. The parsing algorithm gathers required content from a large amount of information and converts it to the required format. For the given master thesis, the data from the Volve oilfield was parsed to .csv format and processed afterward.

3.3.2 Missing Values Identification and Removing

Machine learning algorithms do not support data with missing values. The missing values in real-time drilling data are usually having a unique number or stated as NaNs. In most cases, this unique number is -999.25. Thus, it is not a complicated task to detect missing values in the dataset. In the proposed model, the system deletes the whole row with missing values. The MATLAB command *rmmissing* is applied for this action. It removes missing entries from an array or table A. If A is a vector, then the function removes any entry that contains missing data. If A is a matrix or table, then *rmmissing* removes any row that contains missing data.

There were also several points with values less than zero, which does not make any sense regarding the used parameters. To remove such values, the loop built detects the value less than zero in every column besides the *Time* column and deletes the entire row with such value. In most cases, such outliers affect other values in the row too. Therefore, the complete row of the data should be removed in this case.

The model considers outliers detection during drilling operations. However, the dataset also contains values of several parameters during connections. The rows with a value of Weight on bit equal to zero are also deleted. It helps to get the pure drilling phase from the dataset.

3.3.3 Outliers Detection and Replacement

Commonly, the outliers in the data may appear for different reasons, such as:

- Occurrence while experimenting,
- Wrong calibration,
- Sensor damage/error etc.

The MATLAB function *rmoutliers* helps to remove such outliers from the real-time data.

By default, this function detects a value that is more than three scaled median absolute deviations (MAD) as an outlier. However, adding '*movmedian*' allows defining outliers as points more than three local scaled MAD away from the local median within a sliding window. Next, the algorithm finds the outliers' locations in *data* relative to the points in t with a window size of 10 minutes (for the case) and removes them. As the window moves, it does not change the size but re-calculates MAD and then detects and removes the values that are more than 3 MAD. The example of the moving window approach is presented in Figure 18.

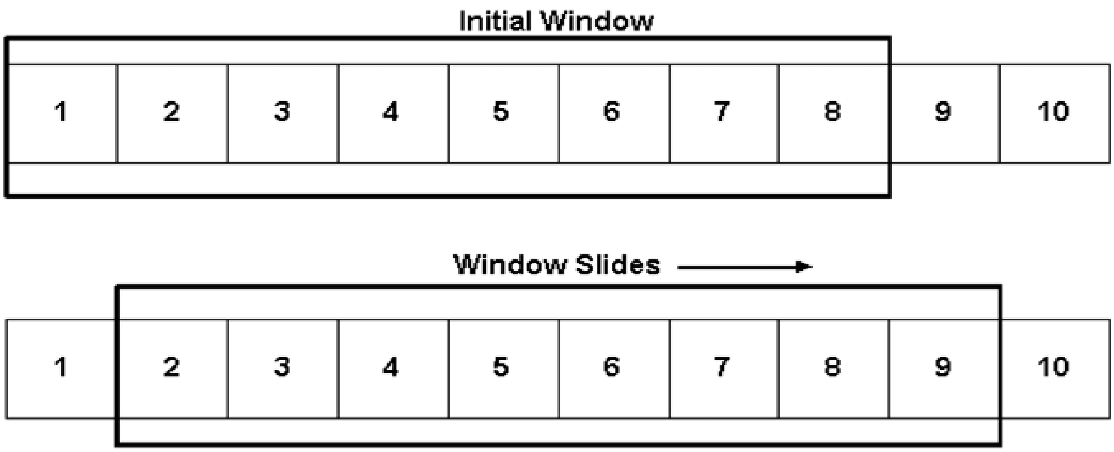

Figure 18: Sliding Window Approach (Source: researchgate.net)

After using this approach, the resulting dataset becomes free of outliers. An example of raw data and filtered data for torque after using *rmoutliers* is shown in Figure 19 (red squares highlight outliers).

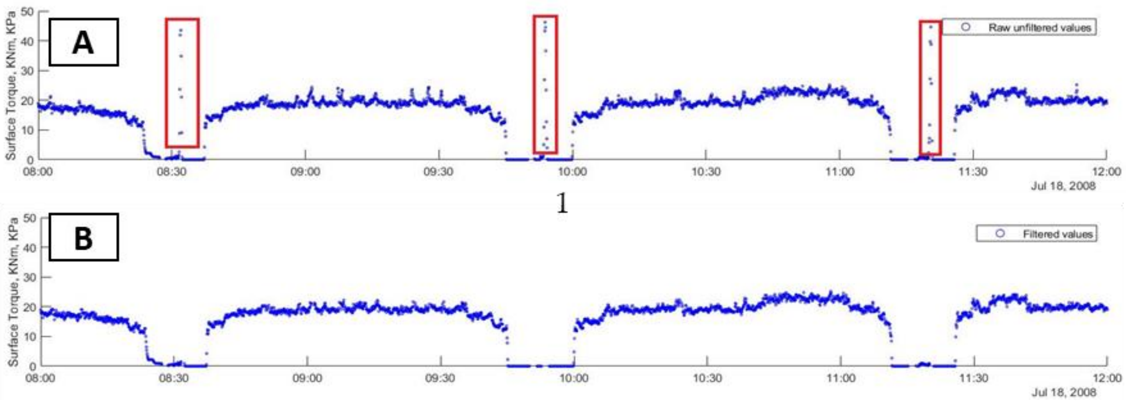

Figure 19: Unprocessed Torques Data with Highlighted Outliers are shown in upper Figure19-A. The lower Figure19-B shows the same data after removing the outliers

3.3.4 Data-set Dividing

It is recommended to split the dataset into train and test sets. The training set is fed into the model to allow the model to find the behavioral patterns of various parameters and train itself. The test set is used to analyze the model's performance on the novel data, which were not engaged in the training process. Hence, two partitions are a more efficient approach when each set has its functions:
• Training set for training a machine learning model.
• Test set for accessing the model performance or evaluating it.

The MATLAB software allows the creation of two sets based on one dataset randomly using function *cvpartition*. The datasets for training and testing are chosen randomly via this function. As a result of this function usage, two datasets to train and test the model are created.

3.4 Predictive Models Building and Tuning

In this section, the process of building three predictive models for torque, standpipe pressure, and hook load will be explained. The section also presents the issues of model tuning and the testing of the predicting model.
The basic workflow of the model training and tuning is illustrated in Figure 20.

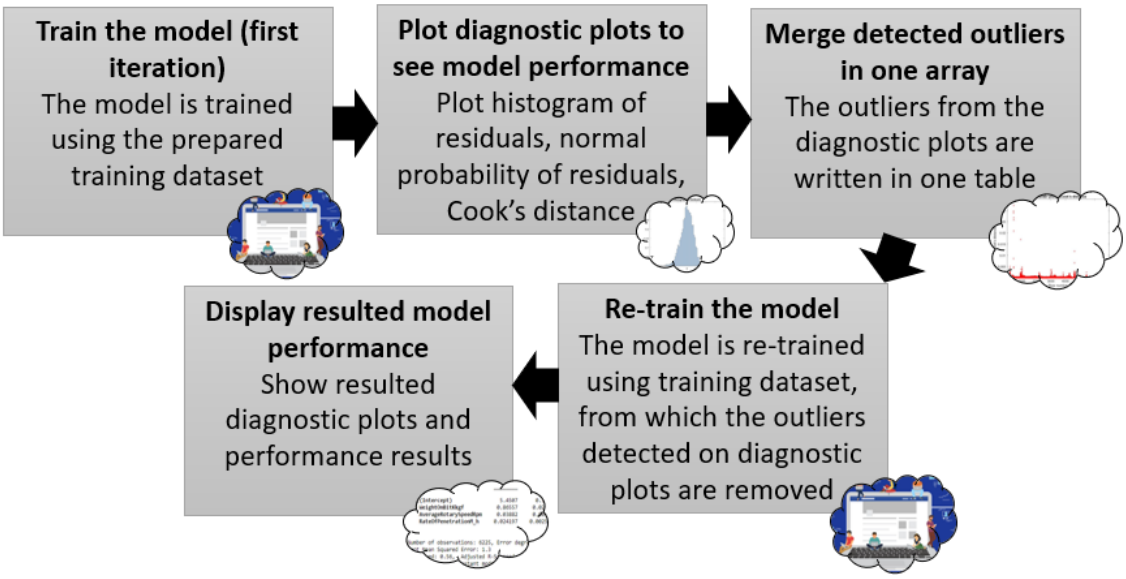

Figure 20: Predictive Model Construction and Optimization Workflow

Stepwise regression is a systematic method for adding and removing terms from a linear or generalized linear model. The method is based on the statistical significance of the terms in explaining the response variable. In the current application, the function *stepwiselm* is used to train the model. *Stepwiselm* creates a linear model and automatically adds to or trims the variables. The workflow explaining the algorithm is shown in Figure 21.

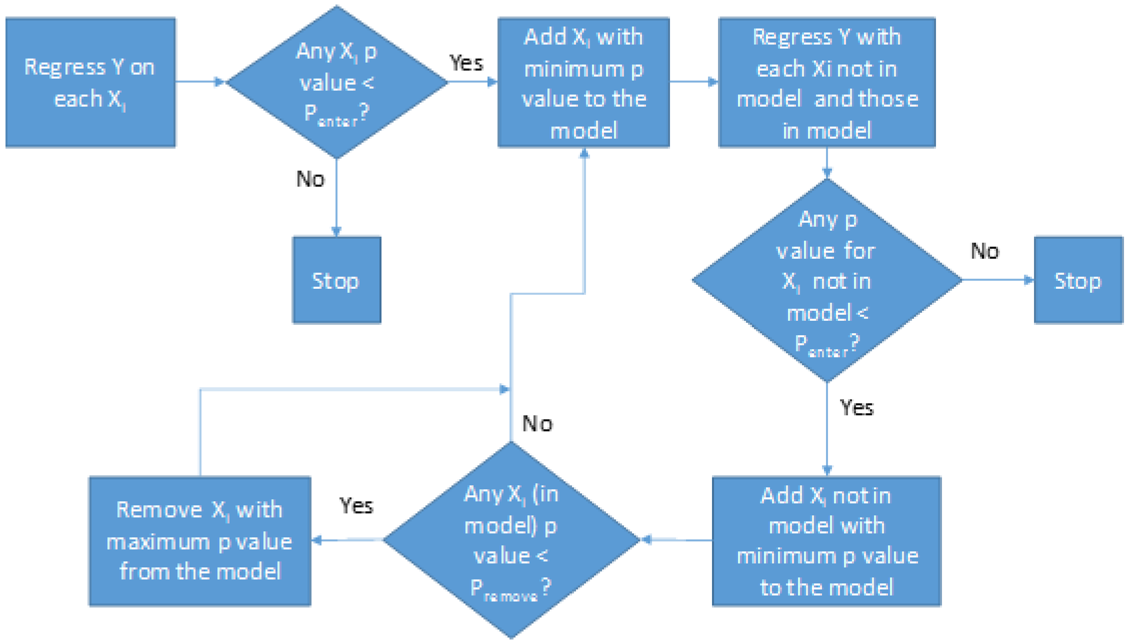

Figure 21: Stepwise Regression Algorithm Working Principle

The pValue is the crucial idea of the algorithm workflow. The pValue concept is the evaluation of the probability of the model's coefficients to affect its performance. It is testing whether the coefficient is equal to zero or not. A low value of p (usually < 0.05) suggests that the coefficient most likely can be added to the model because it impacts the Y response relatively strongly. Conversely, a high value of p suggests that the coefficient is zero, which means that the predictor variable does not impact the Y response and should not be included in the model.

3.4.1 Initial Model Training

Further steps will cover the model training. First, the initial version of the model is built using the prepared training dataset. Then, the software shows basic information about the created model, such as the resulted regression model, root mean square error, r-squared, pValue, and other information. This information is shown in Figure 22.

```
mdltorque =
Linear regression model:
    AverageSurfaceOrqueKN_m ~ 1 + WeightOnBitKkgf + AverageRotarySpeedRpm + RateOfPenetrationM_h

Number of observations: 6229, Error degrees of freedom: 6225
Root Mean Squared Error: 1.31
R-squared: 0.556,  Adjusted R-Squared: 0.555
F-statistic vs. constant model: 2.59e+03, p-value = 0
```

Figure 22: software response after building the model

3.4.2 Diagnostic Plots

The next step is plotting diagnostic plots and their analysis. The resulted residuals plots are represented in figures 23-25. The data used to train the regression model should be normally distributed (Frees, 2009). Therefore, the histogram of the residuals can be beneficial to check whether the variance of parameters is usually distributed. A symmetric bell-shaped histogram evenly spaced around zero indicates that the normality assumption is expected to be correct (Figure 23).

A normal probability plot of residuals can also prove this assumption. If the resulting plot is approximately linear, it proves that the error terms are normally distributed (Figure 24).

Cook's distance or Cook's *D* is a commonly used estimate of the influence of a data point when performing a regression analysis. Cook's distance can be used in several ways: to indicate influential data points that are particularly worth checking for validity, to indicate regions of the design space where it would be good to obtain more data points (Figure 25).

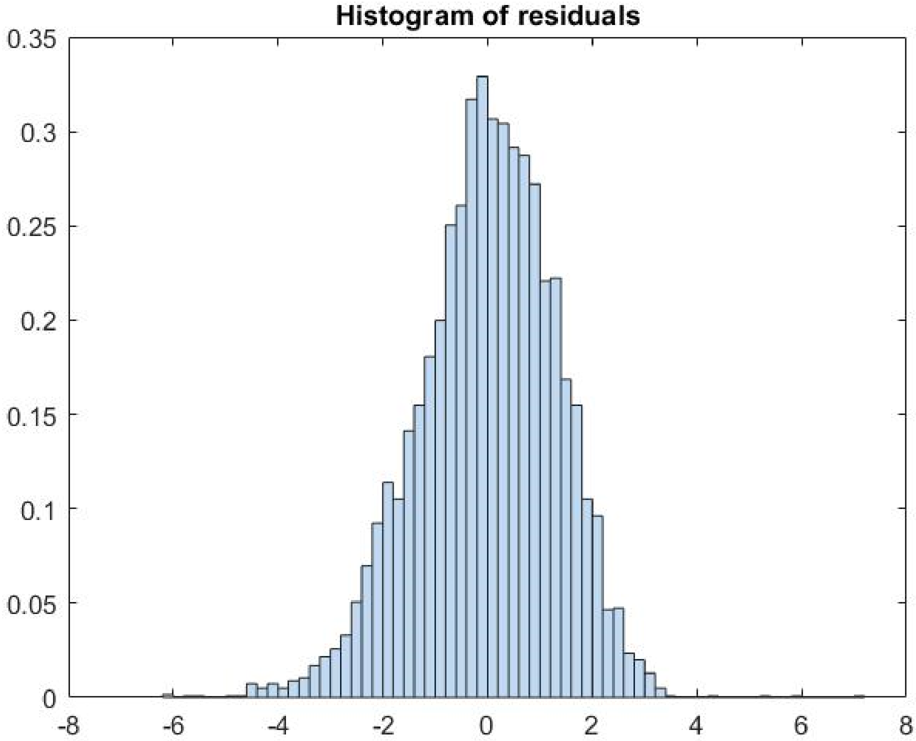

Figure 23: Residuals Errors Histogram

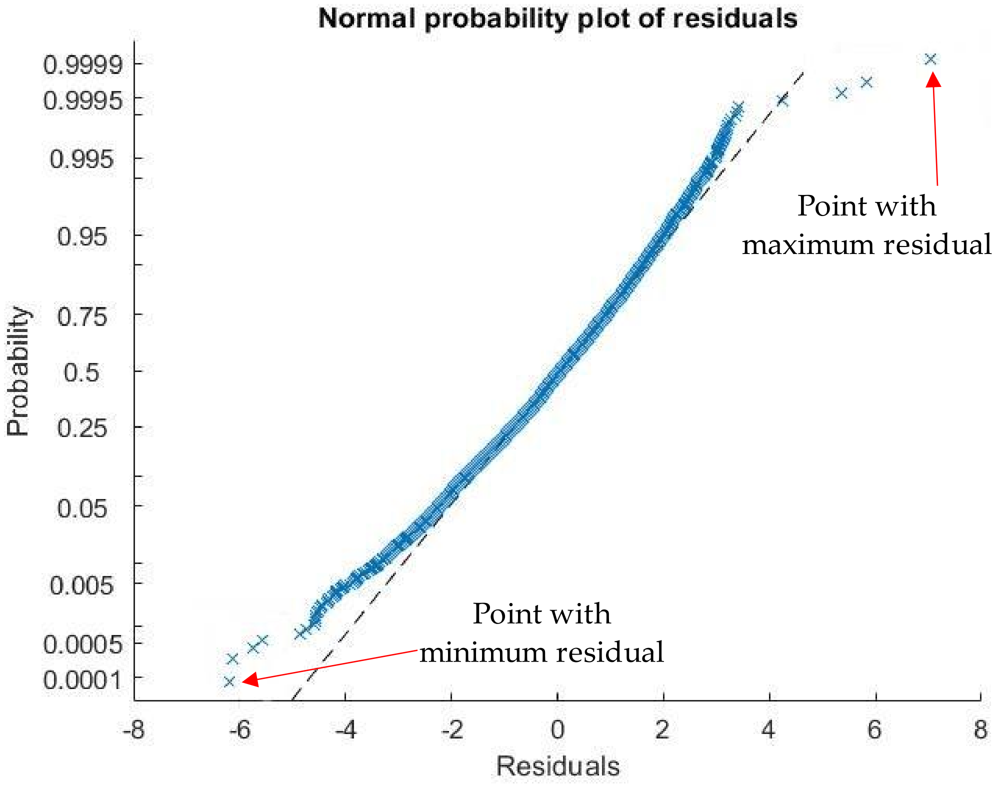

Figure 24: Residuals Errors Normal Probability Plot

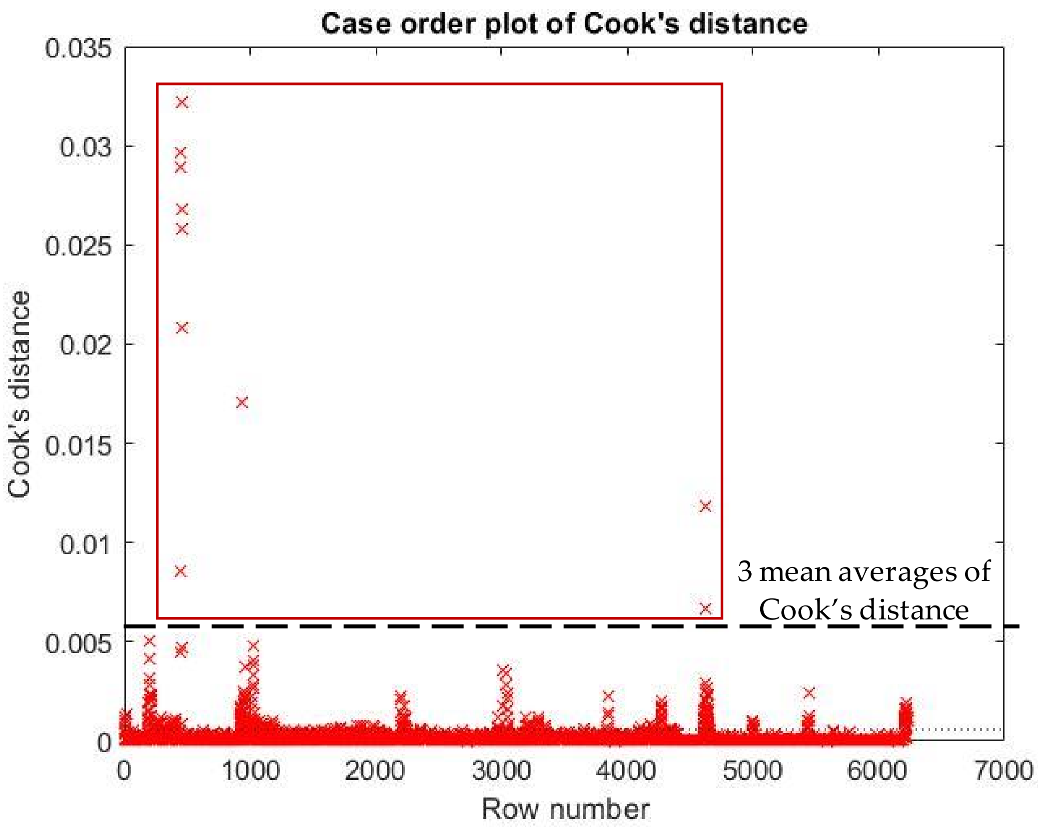

Figure 25: Cook's Distance Plot.

3.4.3 Detecting Outliers in Trained Model

Red marks highlight several data points in Figures 24 and 25. It is visible that these values are out of the dominating trend, so such values are considered to be outliers and can be removed. The model can be re-trained by a filtered dataset to increase performance. In the proposed approach, the model performs such filtering autonomously. Based on the provided plots, the model detects maximum and minimum values from Figure 24 via *max* and *min* functions in Matlab, respectively.

Moreover, to increase the model's performance, the program also uses Figure 25 to detect the outliers, which cannot be detected by the first method. The next step is to detect the values in the training dataset, which has a value greater than three mean average values of Cook's distance (Recommended by Matlab user guide). These maximum, minimum values and values with large Cook's distance are collected in one table to be used in the next step.

3.4.4 Model Re-training

After the outliers have been detected , these outliers are removed from the training dataset. The model is re-trained when the training dataset is free of outliers. This action helps to increase the performance and made the relations between variables in resulted linear regression stronger. The resulted RMSE and R^2 shown in **Error! Reference source n ot found.** prove that the new model is more accurate than the previous version. It demonstrates that after removing the values that are not in the primary trend, the root mean square error goes down, and r-squared increases.

Evaluator	With Outliers	After Removing Outliers
RMSE	1.31	1.3
R2	0.756	0.86
Adjusted R2	0.755	0.866

Table 3: torque predictive model performance comparison before removing outliers and after removing outliers

3.4.5 Model Performance Evaluation

To evaluate the performance of the trained models, the Pearson correlation coefficient R^2 and RMSE were used as performance indicators.

Pearson correlation coefficient or Pearson's r is defined in statistics as measuring the strength of the relationship between two variables. Pearson's correlation coefficient calculates the effect of change in one variable when the other variable changes. For example, the Pearson coefficient is calculated as follows:

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i^n (y_i - \ddot{y}_i)^2}{\sum_i^n (y_i - y_i')^2},$$

Where; SS_{RES} is the residual sum of squared errors of the regression model, SS_{TOT} is the total sum of squared errors.

The issues concerning the range of values of the Pearson correlation coefficient between X and Y are represented in Table 4.

R^2 range	Correlation strength between X and Y
≈1	Perfect correlation
0.8-1	Strong positive correlation
0.3-0.6	Moderate positive correlation
0-0.3	Weak positive correlation
≈0	No linear correlation

Table 4: Pearson coefficients and correlation strength between X and Y (Profillids et al.)

Adjusted R^2 also indicates how well terms fit a curve or line but adjust for the number of terms in a model. If one adds more useful variables, adjusted r-squared will increase.

The value of Root Mean Square Error may help to evaluate the model correctly. RMSE is the rooted sum of distances between two points:

$$RMSE = \sqrt{\frac{1}{N}\sum\nolimits_{i=1}^{N}(y_i - \ddot{y_i})}$$

This formula heuristically depicts that RMSE can be viewed as some normalized distance between predicted values and the vector of observed values. Figure 26 explains this concept. First, for every particular point, the residual (red arrow) is calculated. The residual is the distance between the regression line and the data point. Then, the sum of the residuals is divided by the number of observations. Finally, the root of the resulted sum gives the RMSE.

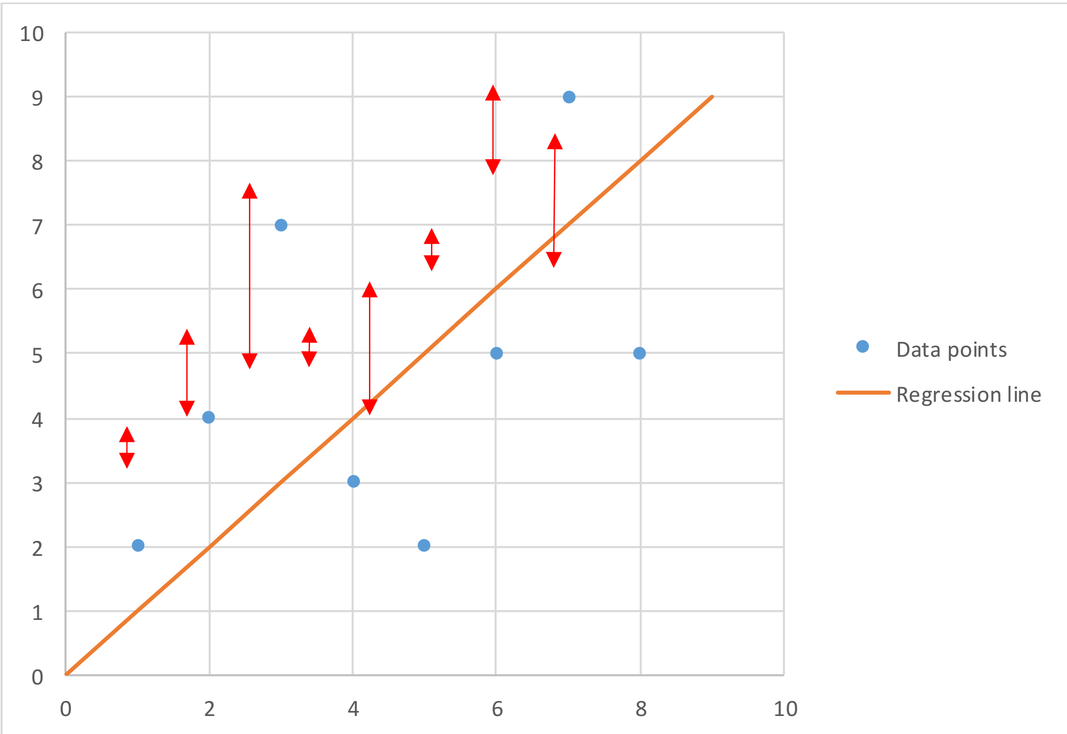

Figure 26: The Visualization of the RMSE Concept

3.5 Stuck Pipe Probability Monitoring and Evaluation

In this part, the process developed for generating the stuck pipe probability monitoring curve and alerting system is provided; in general, the process can be divided into six steps as follows:

1. Feeding real-time data to the model
2. Computation of the predicted indicators (Torque, SPP, hook load)
3. The predicted and real-time indicator averages calculation
4. Deviation calculation
5. Alert table generation
6. Stuck pipe probability calculation

3.5.1 Feeding Real-time Data into the Model

Once the models are trained, they are ready to predict the required parameters. As the drilling process is continuous, the newly generated real-time data is fed into the model (this data is highlighted in blue in Figure 27). The model divides the newly generated data into small time-based windows. Each window has about 2-3 minutes of drilling data. The model provides a possibility to change the window size. Figure 27 shows a visualization of the window approach using torque indicator as an example. However, the same approach is applied to other indicators (SPP, hook load, torque).

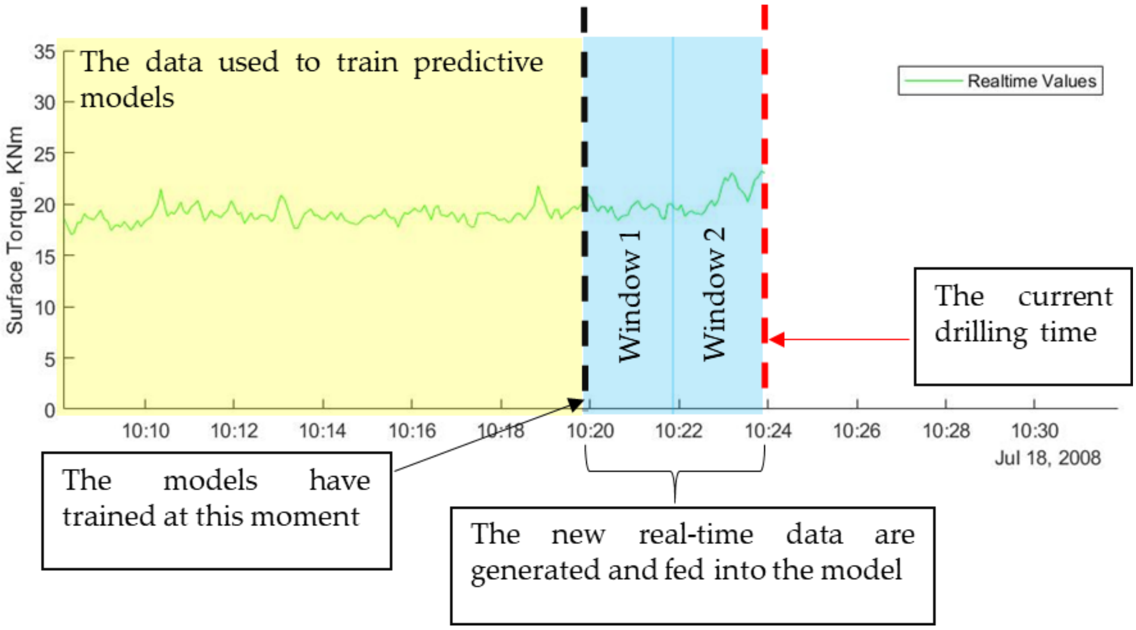

Figure 27: Moving Window Concept Example

Once the data for the windows are imported into the model, data preparation takes place. First, the NaNs are detected via the *rmmissing* function, then the outliers are deleted using the *rmoutliers* function with a moving window.

3.5.2 Predictive Indicator Values Generation

Further steps are related to the torque, SPP, and hook load values prediction by trained predictive models. For every real-time data point within the predefined window, the model predicts the value of the relevant indicator. Figure 28 illustrates the predicted and real-time values within the small windows for the torque. It is essential to mention that the model predicts the standpipe pressure and hook load values in the same manner.

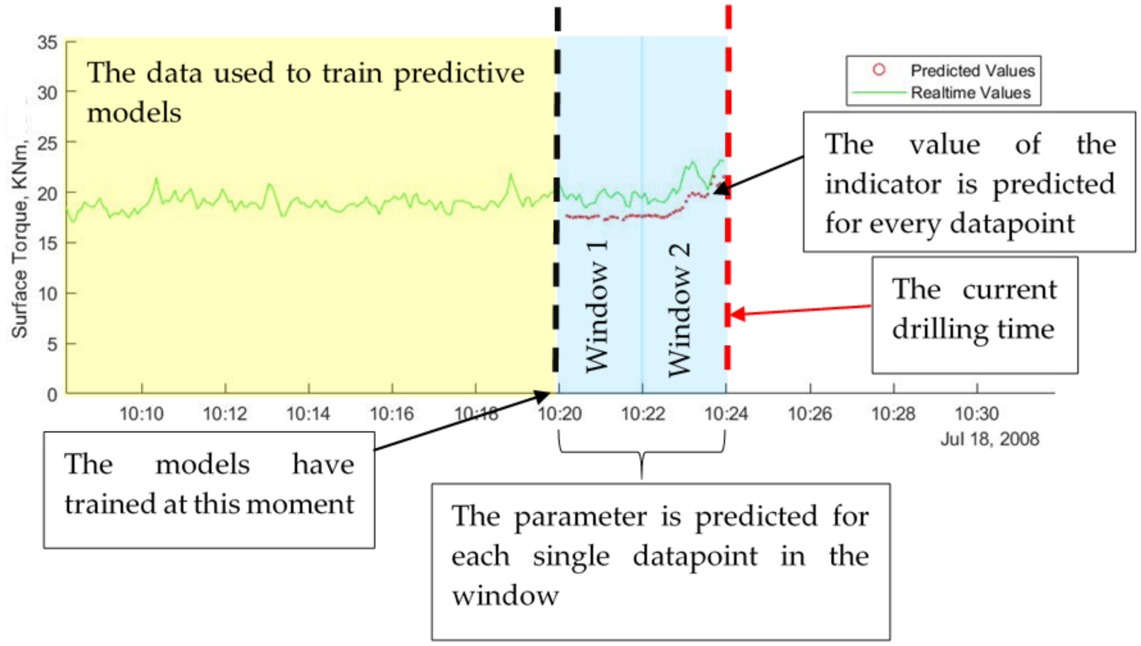

Figure 28: Actual versus Predicted Values within The Specific Window Frame

3.5.3 Sliding Windows Averages Calculation

After successful prediction, the model has two kinds of data points for every indicator divided in windows: predicted values and real-time gained values. These values are relevant for every particular point in the windows. To evaluate the accuracy of the predicted parameters, the model calculates the average of the predicted indicator within the window and the real-time parameter's value average within the same window for further comparison. The model calculates these averages for all three indicators in each window. The resulting averages are used to estimate the difference between the real-time indicator values and the predicted ones. The concept of deviation helps to compare these averages and evaluate the difference.

3.5.4 Deviation Detection and Calculation

As the next step, the model needs to compare the predicted average and the real-time average of every indicator within the same window. To estimate the difference between these two, the parameter which the model applies to evaluate data behavior should be introduced. The proposed model utilizes the deviation concept as a part of the indication of impending stuck pipe.

The deviation formula compares the average of the actual value in the window and the average of the predicted one. The deviation for the indicator x can be expressed mathematically as:

$$Deviation(\%) = \left|\frac{x\_average_{actual} - x\_average_{predicted}}{x\_average_{predicted}}\right| \cdot 100 \quad (1)$$

where $x\_average_{actual}$ – the average of real-time values for an indicator in the specialized window, $x\_average_{predicted}$ - the average of predicted values for the indicator in the specialized window. This value is calculated separately for every window.

Because the model has three indicators, the following deviations are calculated:

1. Torque deviation
2. Hook load deviation
3. Standpipe pressure deviation

After this step, the model has the following data, which were calculated separately for every predefined window:

- Real-time datapoints of every required parameter
- Predicted values of torque, SPP, hook load
- Average real-time value of torque, SPP, hook load
- Average predicted value of torque, SPP, hook load
- Calculated deviation of torque, SPP, hook load in %

3.5.5 Alert Generation

After the previous step, the model has computed the deviation of every indicator. Further, it needs to appraise the calculated deviations. Mainly, the deviation threshold varies from well to well. It means that there is not always a single value beyond which a deviation from the model should trigger a warning. The possible solution is to combine the indicators' deviations in a single stuck pipe risk probability, which distributes from 1% to 100%. To convert the deviations combination into the stuck pipe probability, the model uses the additional calculation step - the alert generation technique.

Using Equation 1, the deviation for every indicator in the predefined window has determined. Onwards, the model compares the deviation value for the next window (in this example, window 2) and the previous one (window 1). The procedure of comparison is the following. If the deviation for the next window is more significant than for the previous one, it means that the difference between real-time data points and predicted ones is even greater for the next window. Such behavior of parameters implies that the real-time value tends to grow. Nevertheless, the predicted indicator shows that there should be no increase in the indicator's real-time value.

In this case, the system recognizes such an event as a sign of growing stuck pipe risk. It means that the system should check how significant the difference is in the deviation and decide whether it can be recognized as an alert. The proposed model uses the relation between the deviations of the indicator for the next window and the previous one. This relation is calculated as follows:

$$Deviation\ relation = \frac{Deviation\ of\ the\ indicator\ for\ the\ next\ window}{Deviation\ of\ the\ indicator\ for\ the\ previous\ window} \quad (2)$$

The deviation relation is calculated for every indicator. Then, the model analyzes the resulted indicator's deviation relations to transform it into a stuck pipe probability. However, the model should not generate an alarm if the deviation relation is more than 1, which means the increase of the indicator exists. Otherwise, any minor deviation will be recognized as a possible sign of stuck pipe with high risk. The weighting method is introduced to avoid the "all-or-nothing" proposition while a relatively small deviation relation is detected (Figure 29).

Check the deviation relation	If the deviation relation is less than 1	If the deviation relation is between 1 and 3	If the deviation relation is greater than 3
Alert level	0	Take calculated deviation relation value as alert level	3

Figure 29: Alert Classification

For instance, if the deviation for the indicator in the next window (window 2) is 12% and the deviation for the previous window (window 1) is 6%, the deviation relation for the case is 2 (by applying Equation 2). Therefore, the alert level for this indicator equals two, according to Figure 29. If the deviation relation is three or more, the alert level is set to 3. It is crucial to set the maximum possible alert level so extra-large data points do not affect the overall alert level. Therefore, the alert level is calculated for every indicator which is presented in the model.

After calculating the indicator deviation alert level, the model has three alert level values for every indicator, representing its deviation trends. The resulting alert level values should be transformed into a single value, indicating the current situation in the well from the stuck pipe perspective. The stuck pipe probability estimation based on the alert levels helps to obtain the impending stuck pipe risk in real-time.

3.5.6 Stuck-Pipe Incident Probability Estimation

As it was explained before, after alert level calculation, every indicator has its alert level value for the next window. The sum of the alert levels represents the current risk of stuck pipe. The final step is to compare the actual alert level with the maximum possible alert level to find the Stuck Pipe Risk in percent. The stuck pipe risk in percent can be mathematically expressed as:

$$Stuck\ pipe\ risk\ (\%) = \left(\frac{Sum\ of\ all\ Alert\ Levels}{Maximum\ Alert\ Level}\right) \cdot 100\ (3)$$

The maximum alert level is predefined. In general, the maximum Alert Level is three times the number of parameters monitored (because there are three levels of alert for each parameter in the system, Figure 29). Thus, since three indicators are monitored in the current model, a maximum alert level is 9.

Finally, the calculated stuck pipe risk is plotted versus time. The model's alarm threshold is set to 50%. If the value of the probability calculated exceeds this limit, the model recognizes it as a high risk of stuck pipe and triggers an alarm to notify the driller. Once the new real-time drilling data is generated, the windows move further. The prediction and risk calculation process continue for newly generated data.

3.6 Model Validation

To demonstrate the developed model's ability to detect the downhole conditions that lead to high stuck pipe risk, two experiments were performed based on historical data belonging to two different wells. The data used is time-based data. The highlighted features of the cases are presented in Table 5.

Case #	Downhole incident	The data frequency	The window size used	Indicators used
1	Stuck pipe	One data point per 5 seconds	2 minutes 30 seconds	SPP, Hookload, Surface torque
2	High stuck pipe risk with overpull while tripping out	One data point per 10 seconds	3 minutes 20 seconds	SPP, Hookload, Surface torque

Table 5: Case Studies Utilized Data Overview

3.6.1 Case Study#1

In this case, the data of the Volve oilfield from the open-source was used. Precisely, the drilling data for July 18 from the well F-5 was implied to validate the model. According to the daily drilling report (DDR), the pipe got stuck in the well between 15:00 and 15:15 during drilling.

The real-time set of the parameters consists of the following data:

- Time
- The flow rate in, L/min
- Average standpipe pressure, KPa
- Average Hookload, Kkgf
- Block position, m
- Rate of penetration, m/h
- Average surface torque, kNm
- Weight on Bit, Kkgf
- Average rotary speed, rpm

The dataset, which is thoroughly cleaned of NaN containing rows, has 4593 data points. The resulting dataset has the pure drilling data phase information. The frequency of the data used is 1 point per 5 seconds. The distribution of every dataset parameter is shown in Figure 30.

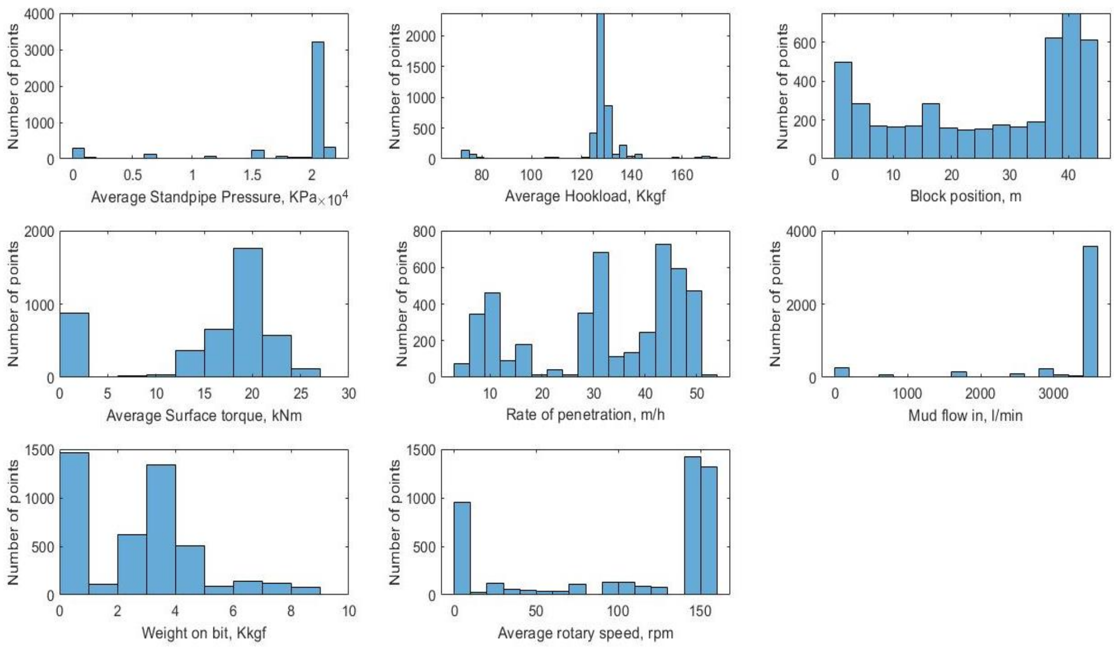

Figure 30: Statistic Distribution of Data Set#1 Prior to applying the Data Cleaning Process

The model was trained based on 36% of the dataset (1676 points). The only drilling phase data for training was used to train the model. The training dataset has 2 hours and 15 minutes of data (the dataset was taken in the interval 7:45-10:00). The predictive models for three indicators are based on the input data depicted in Table 6.

Input Data	Indicator
Weight on Bit, Kkgf; Average rotary speed, rpm; Rate of penetration, m/h	Surface Torque
Weight on Bit, Kkgf; The flow rate in, L/min; Average rotary speed, rpm; Average surface torque, kNm; Block position, m; Rate of penetration, m/h	Hook load
Weight on Bit, Kkgf; The flow rate in, L/min; Rate of penetration, m/h	SPP

Table 6: Input Data Utilized by Each Predictive Models as Implemented in Case Study#1

After the first training, the model automatically detected the following outliers in the training dataset: 114 SPP values, 28 Torque values, 153 Hookload values. According to this detection, the training dataset was corrected for every predictive model to optimize its performance. These values were removed from the training dataset. The predictive models were re-trained automatically after the dataset filtration. The result of the model's adjustment is shown in Table 7. The predictive models have increased the performance in every case.

The model	Performance parameter	Before Tuning	After Tuning
SPP predictive model	R^2	0.748	***0.986***
	Adjusted R^2	0.747	***0.986***
	RMSE	2580	***717***
Torque predictive model	R^2	0.792	***0.961***
	Adjusted R^2	0.792	***0.961***
	RMSE	3.33	***1.4***
Hookload predictive model	R^2	0.726	***0.868***
	Adjusted R^2	0.725	***0.867***
	RMSE	6.83	***2.46***

Table 7: Predictive Model Performances Before and After Optimization (Case Study#1)

The validation dataset has 2917 data points with the same frequency as the training dataset. The data used for the model validation are taken from 10:00 to 16:00. The window size used in this case study is 2.5 minutes (each parameter in the window has 36 real-time data points).

The model was trained based on the same day data from 7:45 to 10:00, which means the model was suitable for the BHA used, the bit used. When the process of training and tuning the model was done, re-playing to validate the model was initiated, so the validation data was fed to the model as in a real-time implementation case. The alarming level is set to 50%, so if the stuck probability is higher than 50%, the system generates an alarm.

The stuck pipe probability curve represented in Figure 31 shows several alarms generated by the model. The first alarms were more than 3,5 hours before the actual incident (two alarms 51% and 62% about 11:30). Further, the alarm was triggered seven times with the probability from 60 to 80%. Finally, the last and the highest stuck pipe probability was detected 5 minutes before the reported stuck pipe incident, with a value of 100%. Based on the current case study, it can be concluded that in real-time implementation, the model could notify the driller about dangerous conditions and high stuck pipe risk so that the correct actions can be taken in time.

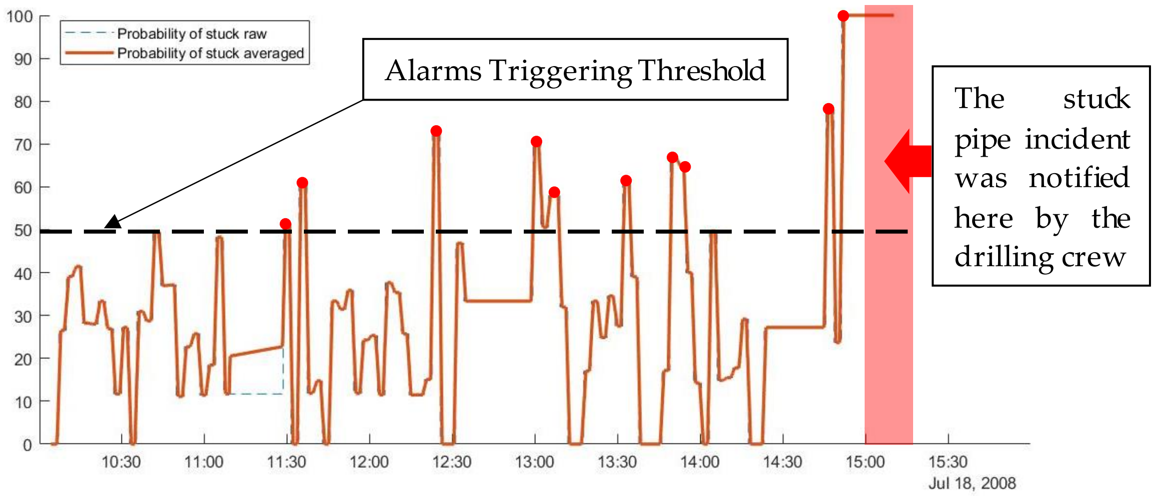

Figure 31: Well F-5 (Volve oilfield) with Stuck Pipe Incident occurred between 15:00 and 15:15; (as it can be seen, the model generated several alarms (red dots) before the actual incident encountered)

3.6.2 Case Study#2

The second validation case was done using the same re-play manner. The primary activity performed in the dataset was drilling. The set of real-time recorded parameters consists of the following data:

- Time
- Flow rate in, gpm
- Standpipe pressure, psi
- Hook load, klbs
- Block position, ft
- Rate of penetration, ft/hr
- Surface torque, lb*ft
- Weight on bit, klbs
- Rotary speed, rpm

The complete dataset has 24 hours of data (8640 data points with one data point per 10 seconds frequency). The distributions of the parameters used in the model as inputs are represented in Figure 32.

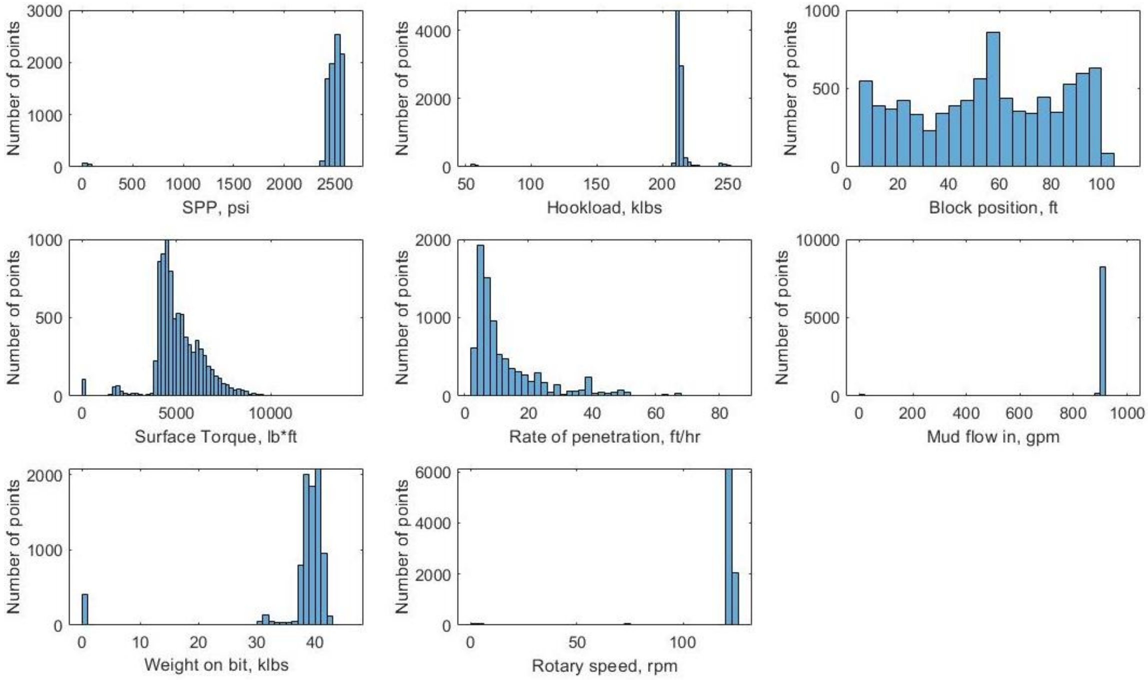

Figure 32: Statistic Distribution of Data Set#2 Before applying the Data Cleaning Process

In this case study, 50% of the dataset (4320 data points from 00:00 to 12:00) was used to train the predictive model. The rest of the dataset (4320 data points from 12:00 to 24:00) represents a validation dataset. The inputs used in predictive models for the current case study are depicted in Table 8.

Input Data	Indicator
Weight on bit, klbs; Rotary speed, rpm; Rate of penetration, ft/hr	Surface Torque
Weight on bit, klbs; Flow rate in, gpm; Rotary speed, rpm; Average surface torque, kNm; Block position, m; Rate of penetration, ft/hr	Hook load
Weight on bit, klbs; Flow rate in, gpm; Rate of penetration, ft/hr	SPP

Table 8: Input Data Utilized by Each Predictive Models as Implemented in Case Study#2

The model detected and removed 36 SPP values, 102 torque values, and 46 hook load values for the training dataset applied in the case study. The results of the adjustment are demonstrated in Table 9.

The model	Performance parameter	Before adjustment	After adjustment
SPP predictive model	R^2	0.931	***0.973***
	Adjusted R^2	0.931	***0.973***
	RMSE	60.1	***32.2***
Torque predictive model	R^2	0.443	***0.452***
	Adjusted R^2	0.443	***0.452***
	RMSE	896	***826***
Hookload predictive model	R^2	0.737	***0.99***
	Adjusted R^2	0.736	***0.99***
	RMSE	6.71	***0.738***

Table 9: Predictive Model Performances Before and After Optimization (Case Study#2)

Figure 33 shows the results obtained when the validation dataset was fed into the model. Again, the alarming level was set to 50%, as in the first case study. In the current dataset, there was no report of a stuck pipe incident. However, the probability of a stuck curve shows several triggered alarms while drilling (at 13:45, 15:00, 17:20. 18:05, 20:00, 20:50). Furthermore, according to the daily drilling report, the drilling crew notified interment overpull at different depths while tripping out. This fact proves the mentioned theory regarding the hole conditions.

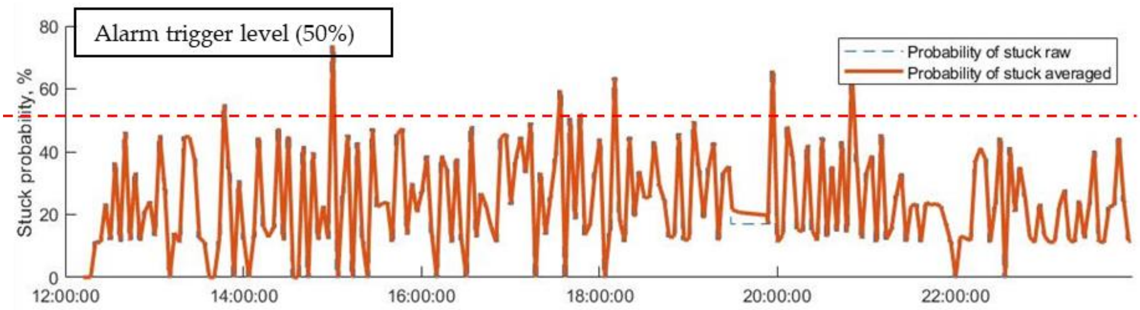

Figure 33: Graphical Presentation of the Generated Probability Curve for Second Case Study

3.7 Model Limitations

As every well is unique, the mechanism of the stuck pipe and its symptoms differ from well to well. In addition, these symptoms may be sensitive to the formation drilled, BHA used, drilling bit used, the diameter of tools in BHA, mud properties.

The proposed model does not need information about BHA, well profile, mud properties, and other parameters. However, the main limitation is that the model requires some drilling time to gain the real-time data from the current well and be trained first for the particular case or section. When drilling a new section, change bit, or BHA, the model should be re-trained to have accurate results. In addition, the data used to train the model must be incident-free data. If the stuck pipe earlier sign is presented at the input data set , then the model will be trained with improper data and as result , it will not be able to detect the stuck pipe

Moreover, the alerting system may be improved by enhancing the number of alarm levels and describing them. This action can make the model more sensitive to unique changes and give more accurate results. As for case study 2, there was no actual stuck pipe, but the hole conditions were poor, so the model can be possibly adjusted to inform the user not only about high stuck pipe risk but also about changes in the wellbore conditions, which may lead to possible stuck pipe development.

Chapter 4 Conclusions

The main conclusion of the thesis can be summarized in the following points:

1. Detecting the early signs of the impending stuck pipe has a benefit for operator companies and drilling contractors. Most stuck pipe incidents result in a significant increase in non-productive time, making this kind of incident more costly for the operators. In addition, from the drilling contractor's perspective, detecting the early signs of possible stuck pipe allow the drillers to be notified about the dangerous conditions in the wellbore from the stuck pipe point of view.
2. One method that can be used to identify the early warning signs of a stuck pipe incident is data-driven methods. Although these methods have shown promising results, they still have specific shortcomings. Generally, these approaches require large amounts of input data and proper adjustment. Thus, it takes time to prepare the model for implementation in real-time data monitoring. Moreover, in most cases, such data preparation cannot be automated; the fact makes these approaches inapplicable for real-time implementation.
3. Methodology for detecting possible signs of possible stuck pipe incidents while drilling by monitoring the main three relevant surfaces drilling parameters was presented.
4. The proposed model uses the deviation approach to detect the imminent stuck pipe incident. A unique alerting system was developed to evaluate the deviation severity and transform it into a stuck pipe probability curve. The curve is utilized to generate alarms in the case that the probability value exceeds the predefined threshold.
5. Based on the conducted case studies, the following facts can be denoted:
    - The example of implementing the methodology for the real-time data in case study #1 shows that the model can detect early signs of developing stuck pipe several hours before the actual incident occurred.
    - The model also reported the high stuck pipe risk in case study#2. Although there was no stuck pipe incident, the model generated several alarms, which can be interpreted as poor hole conditions. This hypothesis is proven by the daily drilling report for the next day, where the high overpull while tripping out was denoted.
6. The shortcomings of the proposed model are underlined. Despite numerous advantages, the model needs actual data gained during drilling to be trained. The proposed 3-level alarming system can be improved by introducing more levels. This improvement is assumed to make the model more precise and sensitive in particular cases.

Bibliography

1. Muqeem, Muhammad A., Weekse, Alexander E., and Ali A. Al-Hajji. *Stuck Pipe Best Practices – A Challenging Approach to Reducing Stuck Pipe Costs*. Paper presented at the SPE Saudi Arabia Section Technical Symposium and Exhibition, Al-Khobar, Saudi Arabia, April 2012. DOI: https://doi.org/10.2118/160845-MS
2. Robert F. Mitchell and Stefan Z. Miska. *Fundamentals of Drilling Engineering*. SPE Textbook Series No. 12, 2011, 696 pp.
3. David James. 2018. *Introduction to Machine Learning with Python: A Guide for Beginners in Data Science (1st. ed.)*. CreateSpace Independent Publishing Platform, North Charleston, SC, USA.
4. Hegde, Chiranth, Wallace, Scott, and Ken Gray. *Real-Time Prediction and Classification of Torque and Drag During Drilling Using Statistical Learning Methods*. Paper presented at the SPE Eastern Regional Meeting, Morgantown, West Virginia, USA, October 2015. DOI: https://doi.org/10.2118/177313-MS
5. Brankovic, Aida , Matteucci, Matteo , Restelli, Marcello , Ferrarini, Luca , Piroddi, Luigi , Spelta, Andrea , and Fabrizio Zausa. *A Data-Based Approach for the Prediction of Stuck-Pipe Events in Oil Drilling Operations*. Paper presented at the Abu Dhabi International Petroleum Exhibition & Conference, Abu Dhabi, UAE, November 2020. DOI: https://doi.org/10.2118/202625-MS
6. Ahmed, Omogbolahan S, Aman, Beshir M, Zahrani, Majed A., and Folorunsho I. Ajikobi. *Stuck Pipe Early Warning System Utilizing Moving Window Machine Learning Approach*. Paper presented at the Abu Dhabi International Petroleum Exhibition & Conference, Abu Dhabi, UAE, November 2019. DOI: https://doi.org/10.2118/197674-MS
7. Hess, Joshua *Pipe Sticking Prediction Using LWD Real-Time Measurements*. Paper presented at the IADC/SPE Drilling Conference and Exhibition, Fort Worth, Texas, USA, March 2016. DOI: https://doi.org/10.2118/178828-MS
8. Zhang, Feifei , Islam, Aminul , Zeng, Hao , Chen, Zengwei , Zeng, Yijin , Wang, Xi , and Siyang Li. *Real-Time Stuck Pipe Prediction by Using a Combination of Physics-Based Model and Data Analytics Approach*. Paper presented at the Abu Dhabi International Petroleum Exhibition & Conference, Abu Dhabi, UAE, November 2019. DOI: https://doi.org/10.2118/197167-MS
9. Pål Skalle, Agnar Aamodt, *Petrol 18 946: Downhole failures revealed through ontology engineering*, Journal of Petroleum Science and Engineering, Volume 191,2020,107188, ISSN 0920-4105, https://doi.org/10.1016/j.petrol.2020.107188.
10. Tsuchihashi, Naoki, Wada, Ryota , Ozaki, Masahiko , Inoue, Tomoya , Mopuri, Konda Reddy, Bilen, Hakan , Nishiyama, Tazuru , Fujita, Kazuhiro , and Kazuya Kusanagi. *Early Stuck Pipe Sign Detection with Depth-Domain 3D Convolutional Neural Network Using Actual Drilling Data*. *SPE J.* 26 (2021): 551–562. DOI: https://doi.org/10.2118/204462-PA
11. Meor Hashim, Meor M., Yusoff, M. Hazwan, Arriffin, M. Faris, Mohamad, Azlan , Tengku Bidin, Tengku Ezharuddin, and Dalila Gomes. *Performance Improvement of Wells Augmented Stuck Pipe Indicator via Model Evaluations.* Paper presented at the International Petroleum Technology Conference, Virtual, March 2021. DOI: https://doi.org/10.2523/IPTC-21455-MS

12. Noshi, C. I., and J. J. Schubert. "*Using Supervised Machine Learning Algorithms to Predict BHA Walk Tendencies.*" Paper presented at the SPE Middle East Oil and Gas Show and Conference, Manama, Bahrain, March 2019. DOI: https://doi.org/10.2118/195111-MS
13. Edward W. Frees 2009. "*Regression Modeling with Actuarial and Financial Applications*, Cambridge University Press, 2009. 584pp. ISBN: 978-0-521-13596-2), pages 3-7, September.
14. Mathworks. (2021). *Data Analysis: User's Guide* (R2021A). Retrieved March 10, 2021, from https://uk.mathworks.com/help/pdf_doc/matlab/
15. Carpenter, Chris. "*Stuck-Pipe Prediction With Automated Real-Time Modeling and Data Analysis.*" *J Pet Technol* 68 (2016): 72–73. DOI: https://doi.org/10.2118/0616-0072-JPT
16. V.A. Profillidis, G.N. Botzoris, in *Modeling of Transport Demand*, 2019 Retrieved May 12, 2021, from https://www.sciencedirect.com/topics/social-sciences/pearson-correlation-coefficient
17. Salminen, Kent, Cheatham, Curtis, Smith, Mark, and Khaydar Valiulin. *Stuck Pipe Prediction Using Automated Real-Time Modeling and Data Analysis.* Paper presented at the IADC/SPE Drilling Conference and Exhibition, Fort Worth, Texas, USA, March 2016. DOI: https://doi.org/10.2118/178888-MS

Acronyms

ANN	Artificial Neural Network
BHA	Bottomhole Assembly
CART	Classification And Regression Trees
CNN	Convolutional Neural Network
COLT	Computational Learning Theory
DDR	Daily Drilling Report
grad-CAM	Gradient Weighted Class Activation Map
GSPI	General Stuck Probability Index
HCSPI	Hole Cleaning Stuck Probability Index
LWD	Logging-While-Frilling
MAD	Median Absolute Deviation
ML	Machine Learning
MLR	Machine Learning Regression
MSER	Mean Squared Error
NaN	Not A Number
NPT	Non-Productive Time
RMSE	Root Mean Squared Error
RPM	Revolution Per Minute
SLM	Statistical Learning Methods
SPP	Standpipe Pressure
SVM	Support Vector Machines
T&D	Torque And Drag
WITSML	Wellsite Information Transfer Standard Markup Language

List of Figures

List of Tables