FACULTY OF SCIENCE AND TECHNOLOGY

MASTER THESIS

Study programme / specialisation:

Petroleum Geosciences Engineering

The spring semester, 2022

Open / Confidential

Author: Daniel-Nicolae Dura

(signature author)

Course coordinator: Alejandro Escalona Varela

Supervisor(s): Raoof Gholami, Dora Luz Marin Restrepo, Øyvind Clausen

Thesis title: Machine learning for pay zone identification in the Smørbukk field using well logs and XRF data

Credits (ECTS): 30

Keywords: Pay zone, Machine Learning, Well logs, XRF chemical elements, Lithology.

Pages: 75

+ appendix: 11

Stavanger, 15th June, 2022

Machine learning for pay zone identification in the Smørbukk field using well logs and XRF data

By

Daniel-Nicolae Dura

MSc Thesis

Presented to the Faculty of Science and Technology
University of Stavanger

University of Stavanger

June 2022

Acknowledgments

I want to express my gratitude to my family for all the help and support offered throughout the years. Thank you for believing in me and for your constant support.

Similarly, I would like to thank my supervisors Raoof Gholami, Dora Marin, and Øyvind Clausen for their great comments, insightful ideas, and unconditional support throughout the thesis.

Abstract

As geosciences enter the age of big data, a faster and more sophisticated tool is needed to automate manual interpretation workflows, limiting industry professionals' ability to harness all available well-log data to reduce subsurface uncertainty and decision-making time. Moreover, new ways of improving the current state-of-the-art Machine Learning (ML) models' performance are needed.

Net Pay is critical in reservoir characterization, including estimating the original hydrocarbon in place, well test interpretations, calculations of ultimate recovery factors, and stimulation and completion designs (Egbele et al., 2005).

The motivation for the thesis is to create a more robust and consistent ML model for pay zone identification. For this purpose, the dataset for the study was constructed by performing conventional petrophysical analysis in the Smørbukk field, the Norwegian Sea, followed by identifying the pay zones and comparing the results with the available core data. In addition, XRF data was integrated with well logs to build four predictive classification models. This study demonstrates that ML can accurately identify pay zones with F1 scores ranging between 73 and 97%, and integrating XRF data can serve as an additional tool to improve reservoir characterization workflows.

The results indicate that XGBoost was the highest performing model regarding performance and validation time. The potential to integrate XRF chemical elements with well logs is promising as it can add up to a 4% improvement in identifying the pay zones. Finally, we compare all the models' performance and discuss possible reasons why vertical resolution and lateral and vertical variation in lithology impact the performance of the ML models as well as future approaches to have a more accurate assessment of the XRF data potential to enhance the overall classification performance and create a robust and consistent ML model for pay zone identification.

Table of Contents

Table of figures

List of tables

Nomenclature

Abbreviation	Description
XRF	Specification of the chemical composition of the minerals
ML	Machine Learning
GR	Gamma ray log
DENS	Bulk density log
RDEP	Deep resistivity log
NEUT	Neutron porosity log
DTC	Compressional slowness log
CALI	Caliper log
BS	Bit size log
MD	Measured depth
TVDSS	True vertical depth sub sea
PZ	Pay zone
NPZ	Non-pay zone
XGB	Xtreme gradient boosting algorithm
RF	Random forest algorithm
LR	Logistic regression algorithm
Vsh	Shale volume
PHIT	Total porosity
PHIE	Effective porosity
Sw	Water saturation
Si	Silicon concentration
Al	Aluminum concentration
K	Potassium concentration
Fe	Iron concentration
Ca	Calcium concentration
Fm	Formation
m	Meter
ppm	Parts per million
CPU	Central processing unit

Chapter 1

1. INTRODUCTION, OBJECTIVES, AND STRUCTURE OF THE THESIS

1.1. Introduction

Net Pay is a crucial parameter in reservoir evaluation and represents the interval of the reservoir containing a significant volume of potentially exploitable hydrocarbons (Worthington, 2010). The term derives from the fact that it can "pay" an income and is sometimes called the pay zone (Glossary, 2010). Selecting intervals based on their associated well log characteristics is the most typical way of identifying Net Pay. This is achieved using petrophysical cutoff values, which separate the intervals of interest from the non-contributing intervals (Worthington and Cosentino, 2005, Log Interpretation Charts 1984).

Performing good reservoir characterization and formation evaluation is critical in oil and gas exploration. Due to a lack of data, expertise, and the untamed nature of petroleum reservoirs, determining productive intervals has always been a challenge for petrophysicists. This process comes with uncertainty and potential interpretation bias. Moreover, well-logging and coring operations are time-consuming and expensive to carry out. Hence, an urgent task at present is to automate the process of processing versatile commercial information, the volume of which is constantly growing. For mature fields, the issue of maintaining production levels is especially acute, the solution of which, among other things, may be to re-analyze existing petrophysical information and identify previously undrained oil-saturated intervals. This problem of data reinterpretation is multivariate, laborious, and non-trivial. By learning from uncounted experiences from already explored and developed reservoirs, Artificial Intelligence (AI) and Machine Learning (ML) have made this process considerably faster,

easier, and more cost-effective. When using ML, detecting productive intervals can be largely automated.

ML in geosciences has sparked much research interest due to its enormous potential in recent years. As a result, there has been extensive research regarding the application of AI and ML in the field of petrophysics for automated pay zone identification.

Recent examples of data-driven applications for pay zone identification were conducted at basin and field scales (e.g., Stoddart et al., 2020; Arce and Thongsang, 2021). In addition, Guo et al. (2014) discussed the determination of productive intervals in parallel execution of the following tasks: lithology classification, regression of porosity, permeability, and volume of clays. Another data-driven example for net pay determination comes from Masoudi et al. (2014), where artificial neural networks were used in a carbonate reservoir to classify pay zones. Moreover, Tsandra et al. (2018) applied two ML algorithms: a gradient boosting for interval interpretation and a convolutional neural network. Besides, classifying lithology using ML methods is considered by (Gafurove et al., 2014; Mohamed et al., 2019; Hall, 2016), to mention a few.

The performance of current ML models for identifying pay zones is not very satisfactory and consistent. Hence, a better model or ways to increase the overall performance are still required. Current work is following previous data-driven approaches to identify pay zones. However, the novel approach considered in this thesis is the integration of XRF chemical elements with well logs to create a more robust and consistent ML model that can reduce the overall well costs, decision-making time, and potential interpretation bias when performing petrophysical analysis.

As a result of the Released Well Initiative (RWI) project (Figure 1), cuttings data such as XRF chemical elements for more than 1933 exploration wells, offshore Norway, are available.

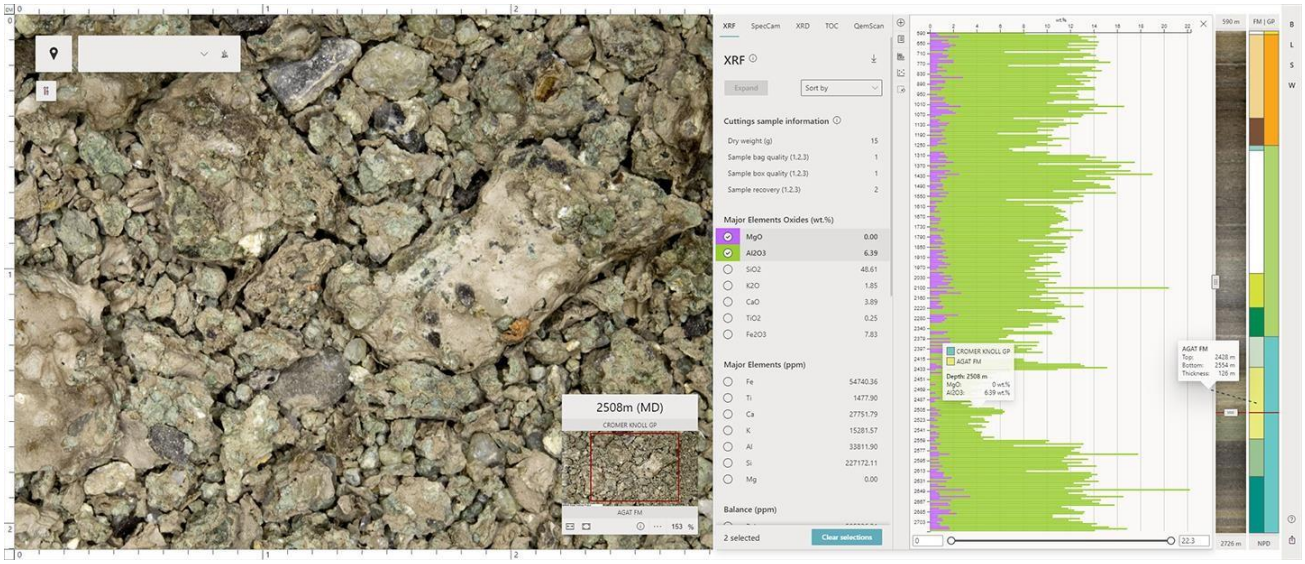


Figure 1:Released Well Initiative (RWI) project. Information such as major elements oxides and major elements ppm are provided with a visual representation of the cuttings for the specific depth interval (After pandionenergy.no).

According to Halvor Jahre, former head of exploration at Lundin Energy, "cuttings do represent the real geology. The information hidden in these samples may well uncover the clues leading towards new plays or play extensions". Thereby, an extensive database of XRF chemical elements exists, which leads to the following research question:

How much can XRF chemical elements add value to reservoir characterization workflows for pay zone identification?

This problem statement is empirically addressed using well logs and XRF data from the Smørbukk field.

1.2. Objectives

The main goal of this thesis is to create a robust and consistent ML model for pay zone identification by integrating well logs with XRF data.

The specific objectives to be covered in this study analysis are:

- Perform classical petrophysical calculations of Vsh, Porosity, and Water saturation.
- Create the label pay zone used in the ML process by applying cutoffs on the reservoir properties results.
- Build, train, test, and compare ML predictive models, including XGboost, Random Forest, and Logistic Regression.
- Study the impact of XRF chemical elements combined with well logs on the ML model performance for automated pay zone identification.

1.3. Structure of the thesis

There are six chapters in this thesis. Chapter 2 provides a brief geological description of the Smørbukk field, including structural setting and stratigraphy and essential background information regarding the research area. Chapter 3 explains the petrophysical analysis and ML techniques used to identify pay zones and how they are applied and assessed. Chapter 4 describes how the database was constructed and preprocessed before being used in the models. In addition, Chapter 4 concentrates on the presentation of results. Chapter 5 discusses the results' limitations and validity and recommendations for future research.

This thesis is finally summarized and concluded in Chapter 6.

Chapter 2

2. GEOLOGICAL SETTING, STRUCTURAL SETTING, AND STRATIGRAPHY

Chapter 2 covers a brief geological description of the study area. Information such as the location of the Smørbukk field in the Norwegian Sea, structural setting, and stratigraphy are provided. This is followed by the methodology used in the thesis.

2.1. Geological setting

This study was conducted in the Middle Jurassic reservoir formations Åre, Tilje, Ile, Tofte, and Garn of the Smørbukk field in the Norwegian Sea, Halten Terrace, Norway (Figure 2).

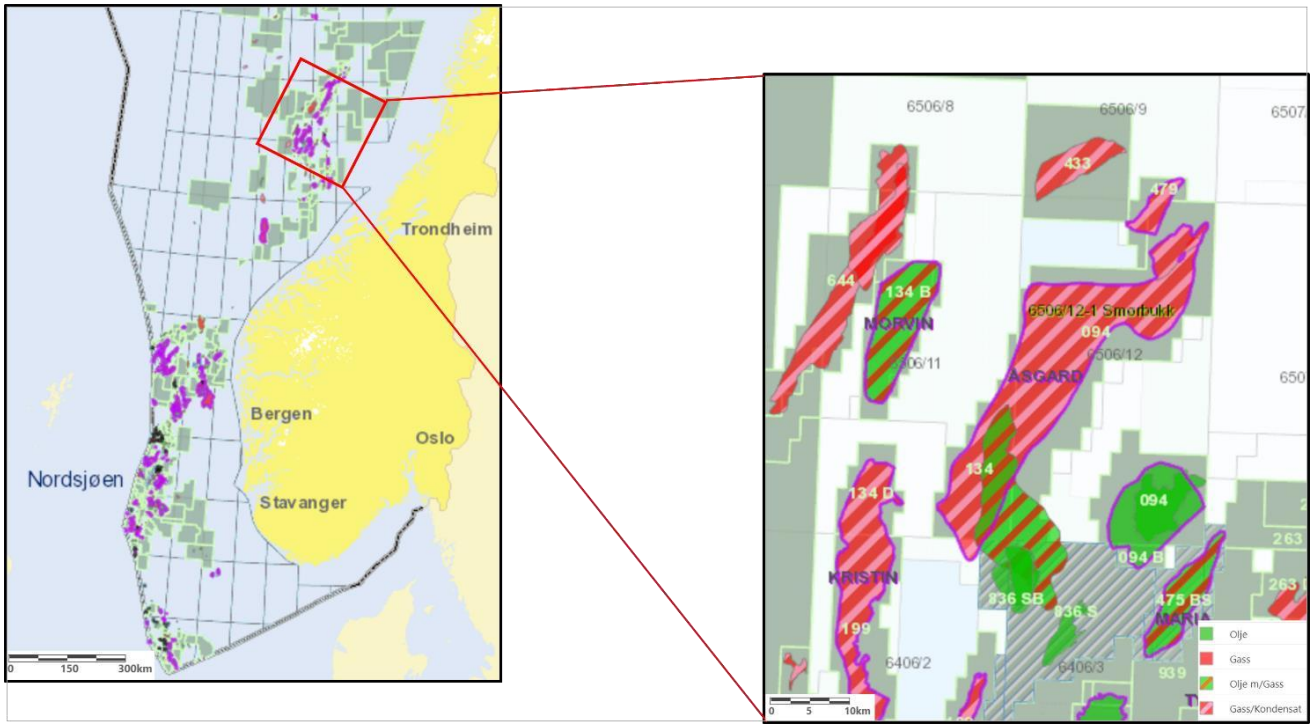


Figure 2:(A) Dataset location in the Norwegian Sea. (B) The license area is shown together with blocks and hydrocarbon presence of the study area (Modified from npd.no).

2.2. Structural setting

The Halten Terrace is located between the Trøndelag Platform in the east, and the Rås Basin in the west is an approximately 80 km heavily block-faulted rhomboidal structure formed as a result of Middle East Jurassic to Early Cretaceous rifting (Blystad et al., 1995) (Figure 3).

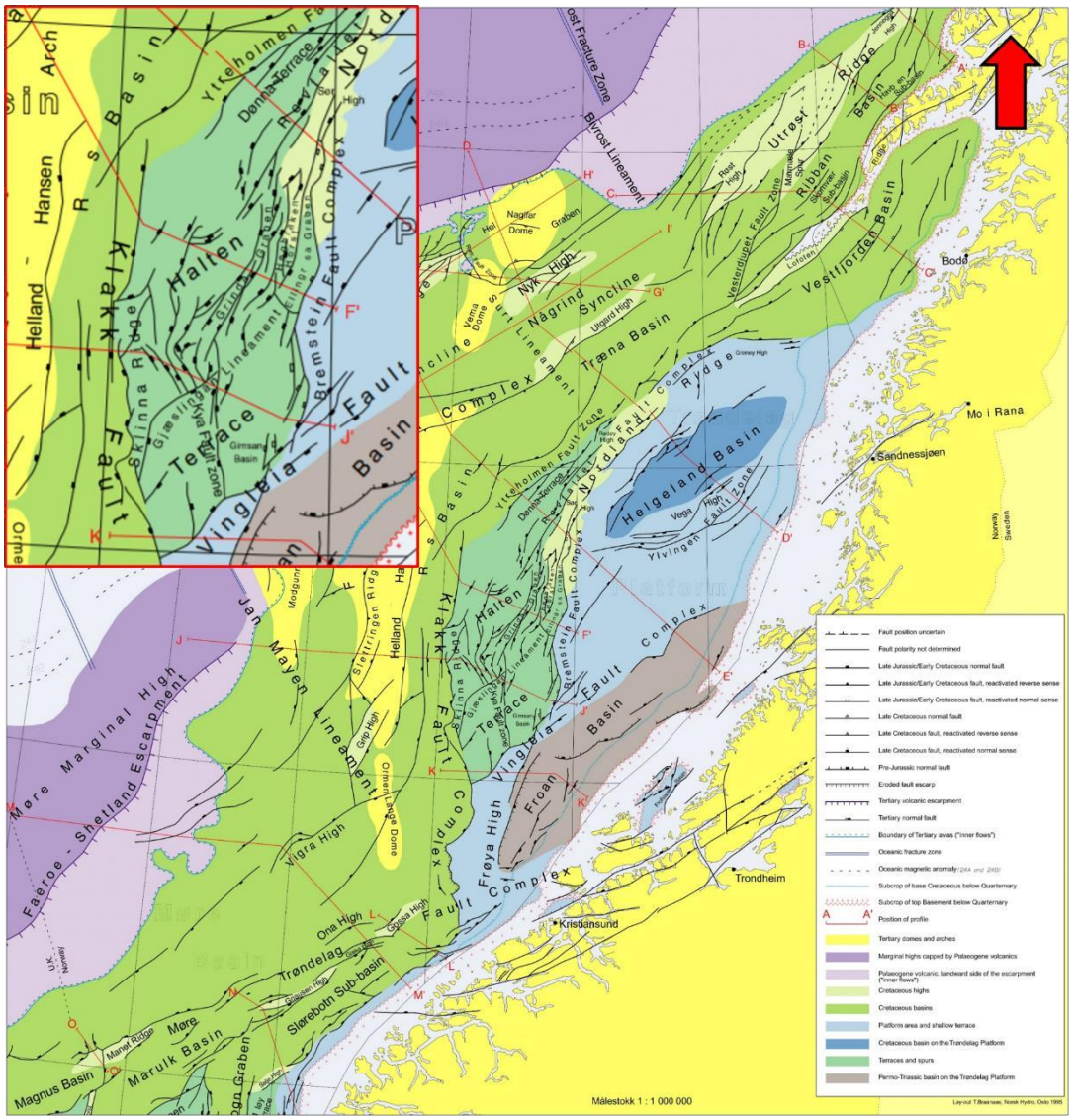


Figure 3:Structural map of the Norwegian Sea (Modified from Blystad et al., 1995).

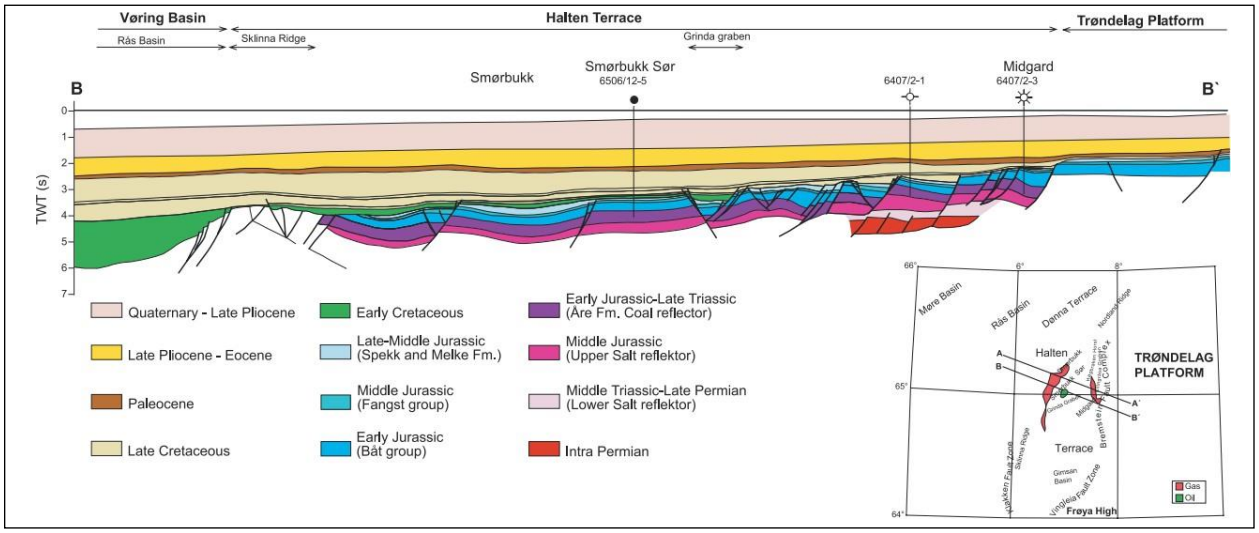


Figure 4:Regional seismic cross-section through Smørbukk, Smørbukk Sør and Midgard fields. (After Klefstad et. al 2014).

During the early stages of this rifting, a major movement occurred along the Klakk fault complex, whereas in later stages, the movement took place along the Bremstein Fault Complex (Figure 3). Jurassic normal faulting dominates the terrace (Figure 4). Large, tilted fault blocks dominate the eastern portion of the terrace, while smaller and more disturbed fault blocks dominate the western part (Koch & Heum, 1995). The Dønna Terrace is the continuation of the Halten Terrace to the north and is separated from the Trøndelag Platform by the Nordland Ridge (Figure 3).

The structural setting of the Smørbukk Field is shown on the WNW–ESE-orientated seismic section (Figure 4). The Smørbukk and Smørbukk South hydrocarbon fields are located in the northwest part of the Halten Terrace. The Smørbukk field lies at the crest of a southeast-dipping fault block, with a major normal fault to the west and an east-west trending graben to the north (Ehrenberg et al., 1992). The Smørbukk South field is located southeast of the Smørbukk field, and it is separated from the main field by an NNE-SSW trending syncline at the base Cretaceous level (Corfield & Sharp, 2000). The Smørbukk field is an anticlinal structure formed due to movement of the underlying Triassic salt and bounded to the east by a major, complex fault. This structure is interpreted to have developed due to a combination of salt doming and extensional faulting, primarily during the Late Jurassic

(Martinius et al., 2005).

2.3. Stratigraphy

This study focuses on the Båt Group (Åre, Tilje, Tofte formations) and the Fangst Group (Ile and Garn formations) reservoirs and the identification of the pay zones in these units. Production on Halten Terrace fields is largely from the Båt and Fangst groups. Sand-prone units of the Båt and Fangst groups were associated with major transgressions (Færseth, 2020).

The Lower and Middle Jurassic Åre, Tilje, Ile, and Garn formations are characterized by heterolithic succession, consisting of a series of extensively developed regressive–transgressive, fluvial, tide- and wave-influenced, tide-dominated, and marginal marine sandstone wedges. These formations are separated by marine mudstones of the Lower and Upper Ror, the Not, and the Melke formations, respectively (Figure 5) (Dalland et al., 1988).

The Ile-and Garn formations constitute some of the most important reservoir units in the prolific hydrocarbon province of the Norwegian Sea. The oldest unit penetrated at Smørbukk field is the Upper Triassic to Lower Jurassic Åre Formation. The most important source rock for the Smørbukk field petroleum system is the Upper Triassic to Lower Jurassic Spekk Formation (Ehrenberg et al., 1992). Sedimentary structures in the Tilje, Tofte, Ile, Garn, and Åre formations contain gas, condensate, and oil, and the reservoir lies between 2 500-4 850 meters down.

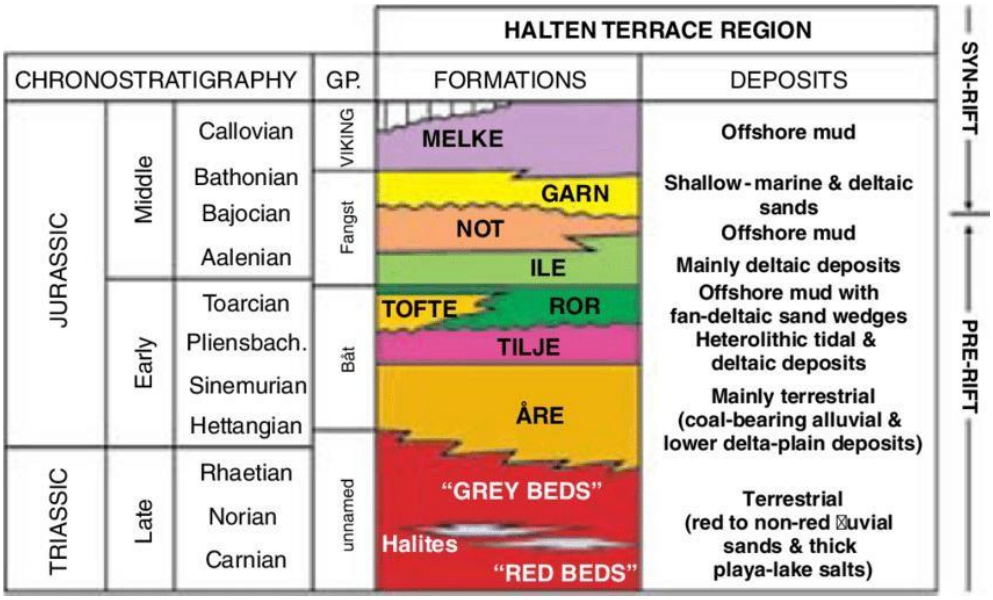


Figure 5:The Upper Triassic to Middle Jurassic stratigraphy of the Halten Terrace (Modified from Martinius et al., 2005; based on the original proposal of Dalland et al., 1988).

Chapter 3

3. DATASET DESCRIPTION, WORKFLOW, AND METHODOLOGY

Chapter 3 starts with an overview of the dataset given and continues with the information about the methodology used. The methodology is divided into two parts and consists of a thorough explanation of the petrophysical analysis used to compute the reservoir properties for the intervals of interest, followed by the ML process. The latter covers the concept of supervised ML, the main algorithms used, and some key aspects involved in making predictions. Python 3.9.6 version is applied for all methods conducted in this research study.

3.1. Dataset description

The wells included in this study are located on the western edge of the Halten Terrace, Norwegian Sea, approximately 200km off mid-Norway.

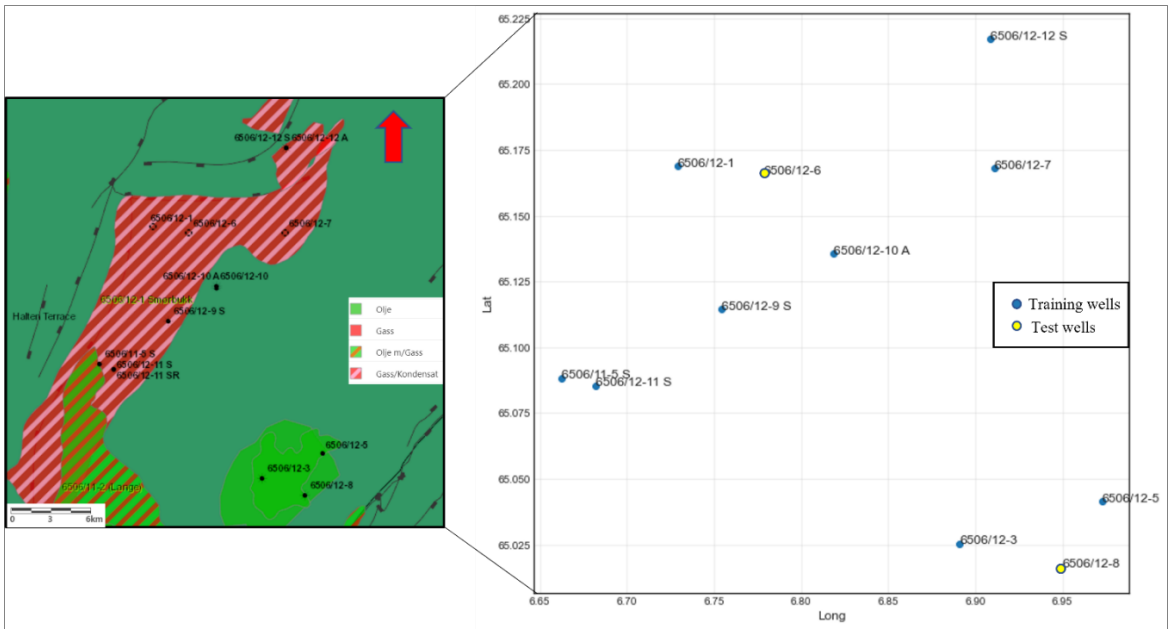


Figure 6:Outline of the wells' location within the Norwegian northern North Sea.

The data available for this thesis include different well data, core measurements, and XRF chemical elements for 11 wells in the Smørbukk field. Well logs include gamma-ray (GR), density (DENS), neutron (NEUT), resistivity (deep), and sonic (compressional), in addition to formation intervals. XRF data consists of 42 chemical elements, but only the major ones such as Al, K, Si, Fe, and Ca are included. The earliest well drilled was in 1984 instead of the latest well was drilled in 2009. Most of the wells were drilled in the '80s (Figure 7).

Table 1:Well data used in this study. A means is available.

Wells	TVDSS (Fm.)	TVDSS (m)	Well logs	Core	XRF
6506/11-5 S	ÅRE FM	4706	A	A	A
6506/12-1	ÅRE FM	4900	A	A	A
6506/12-3	TILJE FM	4336	A	A	A
6506/12-5	ÅRE FM	4557	A	A	A
6506/12-6	ÅRE FM	4708	A	A	A
6506/12-7	TILJE FM	4809	A	A	A
6506/12-8	TILJE FM	4305	A	A	A
6506/12-9 S	ÅRE FM	4879	A	A	A
6506/12-10 A	ÅRE FM	5337	A	A	A
6506/12-11 S	ÅRE FM	4820	A	A	A
6506/12-12 S	ÅRE FM	4880	A	A	A

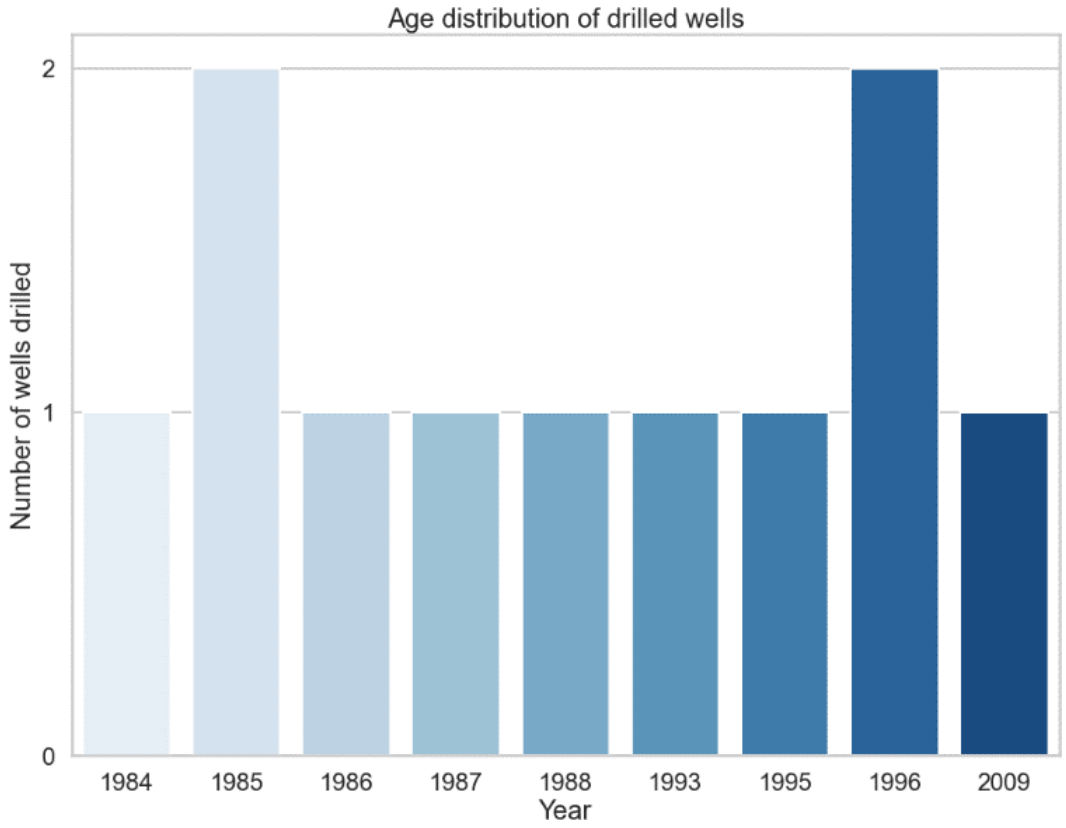


Figure 7:Age distribution of drilled wells used in the study.

3.2. Research workflow

The following research workflow was conducted in this thesis. The workflow is divided into conventional petrophysical analysis and ML pay zone identification.

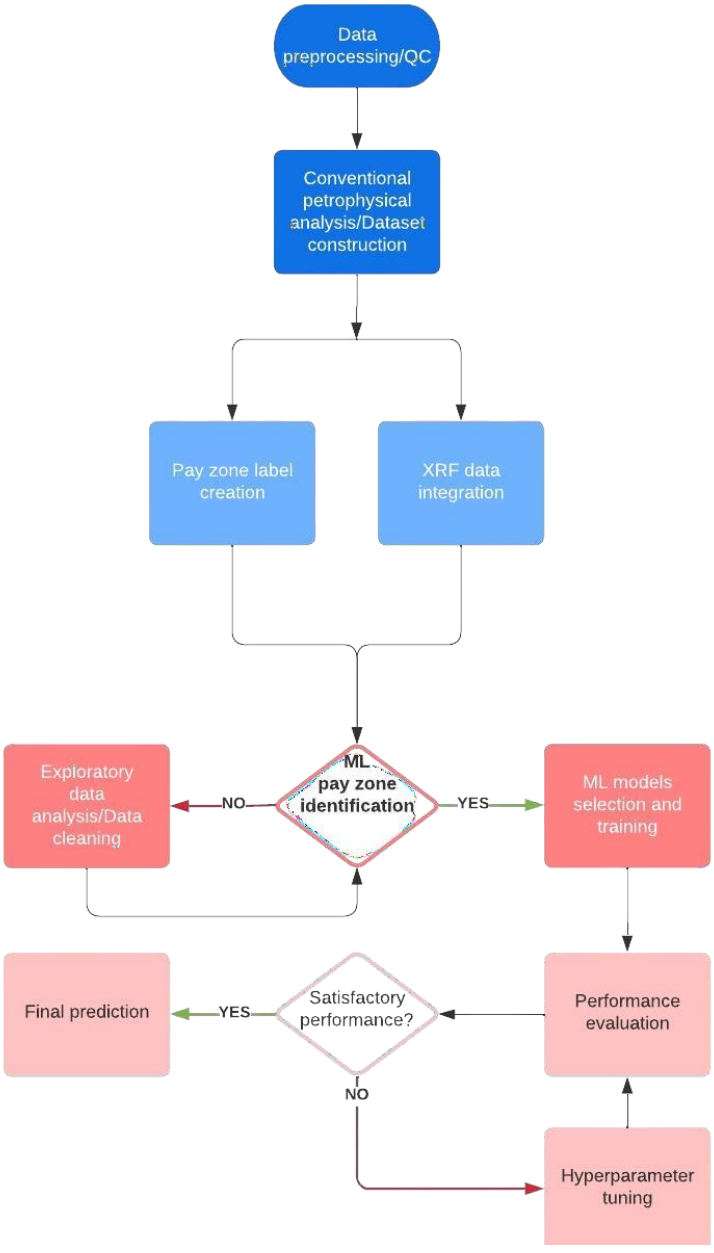


Figure 8:General workflow for this study.

3.3. Petrophysical parameters calculation

In petrophysical analysis, reservoir parameters like porosity (ϕ), shale volume (Vsh), and water saturation (Sw). These parameters have strong importance in determining the quantity and the ability to locate or assess whether or not the hydrocarbons are economically recoverable. The petrophysical evaluation was conducted for the interval between the Åre and Garn formations. This is followed by the main focus of this thesis which is to identify pay zones using ML.

3.4. Determination of Shale Volume (Vsh)

A knowledge of shale volume is necessary to quantitatively evaluate a formation, including determining the porosity and water saturation (Rodolfo et al., 2010). Log measurements such as GR or SP, which respond solely to shale, are the best ways to estimate the volume of shale (Vsh). In particular, the method using the GR log is generally the most straightforward, quick, and reliable. This calculation is essential because it shows how much shale can alter the effective porosity, fill the porous area, and reduce the space available for hydrocarbons. Other methods, such as using the resistivity log in extremely high resistivity formations, compensated neutron in very low porosity formations, and density versus neutron cross plots, can estimate shale volumes under certain circumstances.

This study calculated the shale volume from the GR log using the **linear method**.

Linear Method:

Vsh or Igr = (GR log – GR min)/(GR max - GR min)

Where,

Vsh or Igr = Shale volume, also known as Index of Gamma-ray (Fraction)

GR log = Gamma-Ray Reading (API Unit)

GR max = Gamma-Ray Maximum (API Unit)

GR min = Gamma-Ray Minimum (API Unit)

3.5. Determination of Porosity (Por)

Another important petrophysical parameter is total porosity, representing the proportion of pore volume or void space within a rock that may hold fluids (Bradley, 1987). The most powerful tools for determining porosity are stand-alone tools such as density, neutron, sonic, and nuclear magnetic resonance logs. In contrast, effective porosity is the interconnected pore spaces of the rock that allow fluids to flow easily (Bradley, 1987).

3.5.1. Total Porosity

Total porosity was determined in two ways using density and sonic log.

Density porosity

The density log was used to calculate the porosity, which can also determine lithology or hydrocarbon density. For good borehole conditions, the density log gives the best log porosity using the equation:

$$\emptyset_{\mathrm{D}}(\mathrm{DPHI}) = \mathrm{PHIT} = \frac{\rho_{\mathrm{matrix}} - \rho_{\mathrm{log}}}{\rho_{\mathrm{matrix}} - \rho_{\mathrm{fluid}}}$$

Where,

ρ_{matrix} = Density of the matrix material constant 2.67

ρ_{fluid} = Pore fluid density 1

ρ_{log} = Density log reading

Sonic porosity

For the bad borehole condition, porosity can also be calculated from the sonic log using (Wyllie et al., 1956):

$$\varphi_{\mathrm{s}} = \frac{\Delta T_{\mathrm{log}} - \Delta T_{\mathrm{mat}}}{\Delta T_{\mathrm{F}} - \Delta T_{\mathrm{mat}}}$$

Where,

$T_{\log}$ = log reading

ΔT_{mat} = matrix travel time constant 52

ΔT_{F} = fluid travel time constant 205

3.5.2. Effective Porosity

Effective Porosity is determined by using the following formula:

$$\phi_{\text{e}}(\text{PHIE}) = \frac{\rho_{\text{matrix}} - \rho_{\log}}{\rho_{\text{matrix}} - \rho_{\text{fluid}}} - \frac{V_{\text{sh}} * (\rho_{\text{matrix}} - \rho_{\text{sh}})}{(\rho_{\text{matrix}} - \rho_{\text{fluid}})}$$

Where,

V_{sh} = Volume of shale

ρ_{sh} = Bulk density of shale

3.6. Determination of Water Saturation

The last petrophysical parameter is the water saturation which is one of the most important inputs for evaluating the volume of hydrocarbon in reservoirs and, at the same time, one of the most uncertain. The water saturation represents the fraction of water in a given pore space (Schlumberger, 2022). Many models have been developed to calculate the water saturation, including the Archie, Simandoux, Waxman-Smits-Thomas (WST), and Indonesian model. Archie's equation (Archie, 1942) was used to calculate water saturation (S_w) for this study.

$$S_w = n\sqrt{\left(\frac{a}{\varphi^m}\right) \times \left(\frac{R_w}{R_t}\right)}$$

Where,

S_w= Water saturation

φ = Effective porosity

m = Cementation exponent

a = Tortuosity factor

n = Saturation exponent

R_w = Formation water resistivity

R_t = Formation true resistivity

In this study, for the calculation of water saturation, the deep resistivity log (RDEP) was taken as true formation resistivity (Rt), assuming that it is corrected for invasion, thin bed, and borehole effects. The constants a, m, and n were taken as 1, 1.75, and 2, while formation water resistivity was set to 0.031 Ωm in all water saturation calculations.

3.7. Supervised ML theoretical background

ML is a branch of artificial intelligence (AI) and computer science that signifies a paradigm shift from "normal programming" where all instructions must be explicitly given to the computer to "indirect programming" that takes place through providing data (Duval, 2019). ML is divided into four major categories based on how much and what supervision the algorithms receive during training, including supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. Algorithms that require external assistance are supervised ML algorithms (Figure 9). Supervised Learning is an ML paradigm for obtaining knowledge about a system's input-output relationship from paired input-output training samples (Liu & Wu, 2012).

$$Y = f(X)$$

Hence, According to Brownlee (2016), the goal of supervised learning is to build an artificial system that can estimate the mapping function to the point that you can forecast the output variables (Y) for new input data (X). The output can have either discrete or continuous values, which leads to classification or regression of the input dataset.

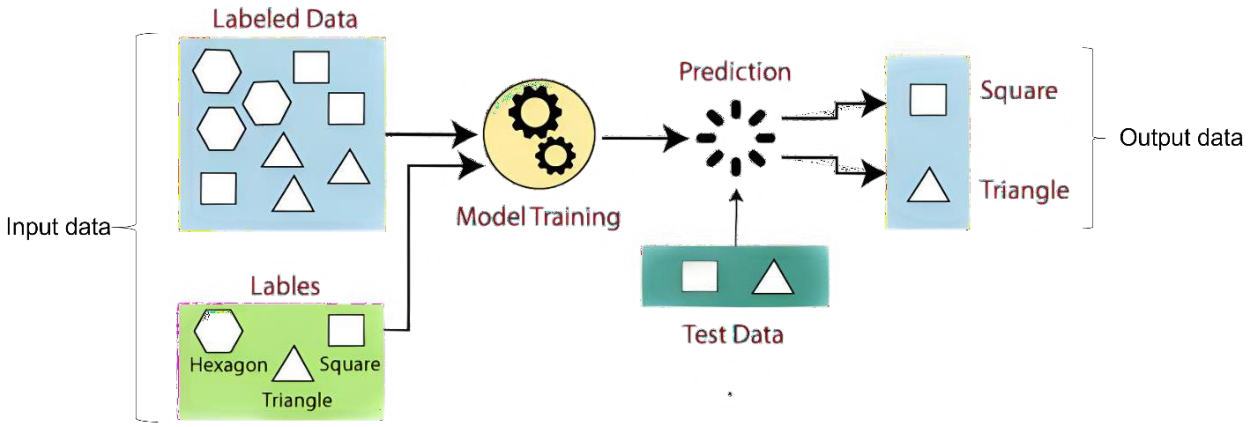


Figure 9:Supervised Machine Learning concept (Modified from javatpoint.com).

3.7.1. Logistic Regression

Logistic regression is the go-to method in statistics to classify an observation into one of two classes. However, it can also classify an observation into one of many classes.

In this study, apart from Logistic Regression, more advanced and computationally expensive models will be used to increase the overall performance.

Logistic regression is the name for the function used at the core of the method, the logistic function, which is also called the sigmoid function and can be written as:

$$\frac{1}{1 + e^{-value}}$$

Where e represents the base of the natural logarithms and value is the numerical value to be transformed. Hence, the predictions are transformed using the logistic function, and the model can be stated as:

$$P(Y \mid X) = \frac{e^{\beta_o + \beta_1 X}}{1 + e^{\beta_o + \beta_1 X}}$$

Using the inverse of the logistic function—called logit or log-odds— the logistic regression is transformed to generate the βo and $\beta 1$ coefficients, allowing logistic regression to fit a regression curve (Awad & Khanna, 2015). The coefficients or input values can be between negative infinity and positive infinity, and the output (P(Y |X) is constrained to values between 0 and 1 (Awad & Khanna, 2015).

$$\text{logit}\,(P(Y \mid X)) = \ln\,(\frac{P(Y \mid X)}{1 - P(Y \mid X)}) = \beta_0 + \beta_1 X$$

Because the logistic function curve is nonlinear, linear regression is performed using the logit transform, where P(Y|X) is the probability of success (Y) for a given value of X (Awad & Khanna, 2015).

Finally, to calculate the coefficients in the logistic function, the maximum likelihood method is used such that the predicted probability $p(X)$ corresponds to the observed probability in the data set.

3.7.2. Random Forest

Random Forest is an ensemble learning approach for classification and regression problems, in which the output of multiple decision trees is combined to reach a single result. It has proved to provide good classification performance and scalability during the past decade.

The term forest signifies using several decision trees necessary to make a classification decision. Random Forest is based on the process called bootstrapping and aggregation, which is widely known as "bagging". This is the first step in understanding the RF algorithm. Bootstrap Aggregation or bagging is a general procedure that can reduce the variance for algorithms with high variance, such as decision trees (Brownlee, 2016). The bootstrapping process represents taking random samples from the training data with replacement (Figure 10).

The second step involves creating decision trees using the bootstrapped dataset with a random subset of features at each step. Thus, decision trees consist of several nodes which aim to find the best possible split involving these features.

As seen in Figure 10, for the final prediction, Random Forest aggregates the output (in case of regression) or takes the majority vote (in case of classification) from the previously created decision trees. This ensures that the prediction from all decision trees has less correlation. By doing so, the variance is reduced, rather than developing a solution based on the output of a single deep tree.

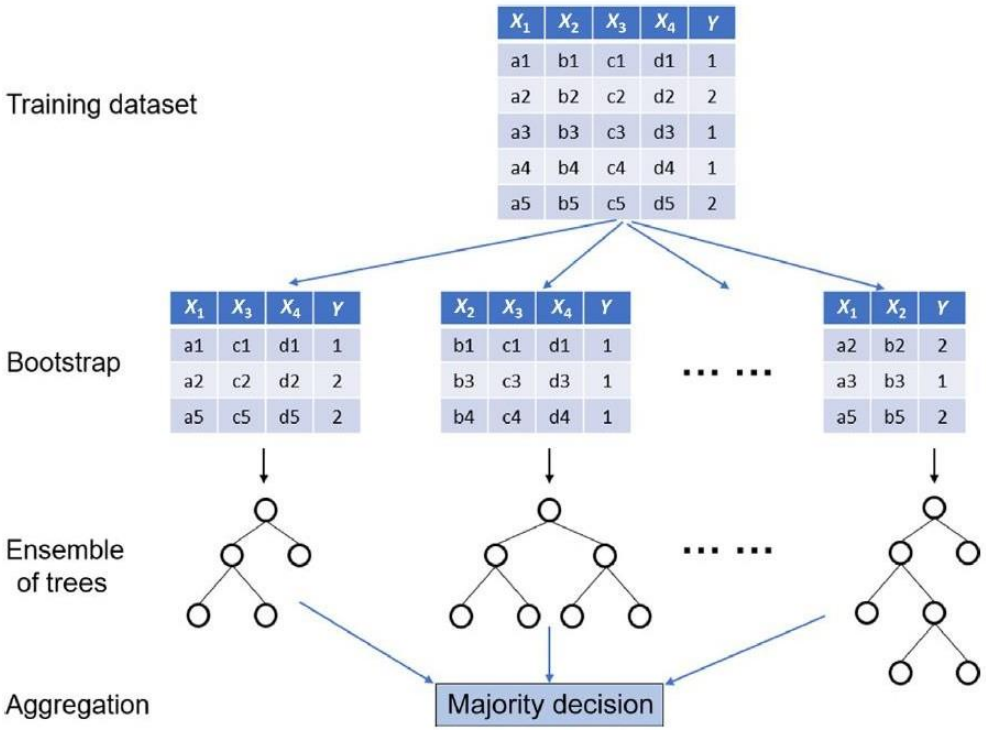


Figure 10:Implementation of RF classifier on a dataset that has four features (X1, X2, X3, and X4) and two classes (Y = 1 and 2) in which each tree is trained on different subsets of training sample and features (After Misra and Li, 2020).

Although individual classifiers are poor learners with significant volatility and bias for single decision trees, when all of them are combined, they constitute powerful learners (Awad & Khanna, 2015).

Random forest has a high level of accuracy and can handle large datasets quickly. Even if a considerable amount of the data is missing, it is an effective strategy for predicting missing data and

maintaining accuracy.

3.7.3. Xtreme gradient boosting (XGboost)

XGBoost is a relatively new algorithm introduced by Chen and Guestrin in 2016. The algorithm is one of the most known and powerful gradient boosting techniques (ensemble). Similar to Random Forest, XGboost is based on decision trees. However, one of the main differences between Random Forest and gradient boosting algorithms is that decision trees in gradient boosting are created in sequential form instead of being in parallel. Moreover, the output represents the sum of all results (Figure 11) rather than majority voting or averaging present in Random Forest:

$$\hat{y} = \sum_{k=1}^{n} f_k(x_i), f_k \in F$$

Where F means the space of regression trees, f_k corresponds to a tree, so $f_k(x_i)$ is the result of tree k, and $\hat{y}$ is the predicted value of i th instance x_i.

The ensemble consists of very simple base classifiers, also often referred to as weak learners. In contrast to bagging, boosting algorithms use random subsets of training samples without replacement from the training dataset (Raschka, 2015).

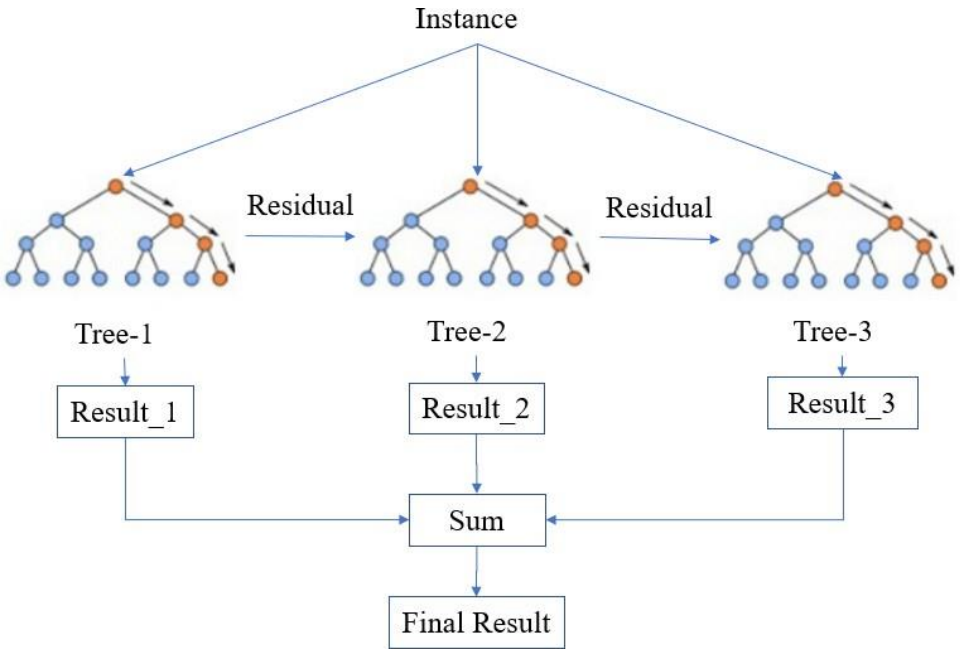


Figure 11:A general architecture of XGboost (After Wang et al., 2021).

The objective of the XGboost is to minimize the loss function whose value increases with how bad the classifier/regressor is:

$$\text{Obj}\ (\theta) = L(\theta) + \Omega(\theta)$$

Where $L(\theta) = \sum_{i=1}^{n}\ l(y_i, \hat{y})$ is loss function, $\hat{y}$ is the prediction and y_i is the target, and $\Omega(\theta) = \sum_{k=1}^{K}\ \Omega(f_k)$ regulates or penalizes the complexity of the model (Wang et al., 2019).

Instead of assigning a higher weightage to the previous, incorrectly classified samples like in boosting, in gradient boosting, the weak learner trains on the remaining errors (so-called pseudo-residuals) of the strong learner (Figure 11). As a result, more importance to misclassified observations is given.

At each iteration, the pseudo-residuals are computed, and a weak learner is fitted to the pseudo-residuals. Thus, the contribution of each weak learner to the final prediction is based on a gradient descent optimization process to minimize the overall error of the strong learner.

3.8. Hyperparameter tuning

A machine learning model usually has two parameters: training parameters and hyperparameters. Model parameters refer to the weights and coefficients derived from the algorithm's data, and every algorithm has a defined set of hyperparameters (Elgeldawi et al., 2021). In the training phase, training parameters are learned, but hyperparameters must be specified before learning begins. Hence, several hyperparameters must be set and carefully optimized to achieve maximal performance.

The most common hyperparameter optimization algorithms are grid Search, Random Search, and Bayesian Optimization. In this study, Grid Search was used as the tunning method. Grid search is one of the most widely used strategies for hyper-parameter optimization, which trains the machine ML algorithm for all combinations of hyperparameters given. In cases where the dataset is extremely large, and hyperparameter value ranges are large, it becomes computationally expensive to implement for most algorithms. In such cases, another standard hyperparameter optimization algorithm, such as Random Search, is preferred. In brief, Random Search involves combining random hyperparameters to find the most optimal solution for a model.

A standard extension of Grid Search is to use stratified cross-validation (Figure 14). Cross-validation (CV) is a helpful statistical method that involves training the model on several folds (5 to 10) with different hyperparameter combinations to reduce over-fitting or under-fitting. The parameter K in K-Fold cross-validation indicates how many folds a given dataset is divided into. One of the folds is kept as a validation set, while the remaining K-1 folds are used to train the machine learning model. Each fold of the K-Folds is utilized as a validating set, with K scores (performance). Finally, as illustrated in Figure 12, we average the model against each of the folds to get a final performance score for the model.

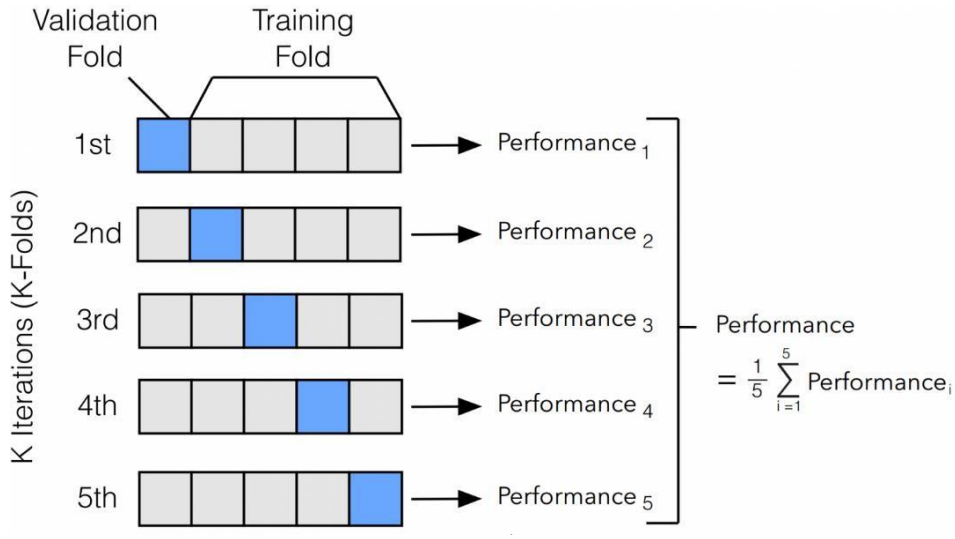


Figure 12:Example of cross-validation technique (After Raschka, 2020).

A summary of the hyperparameters used in this thesis can be seen in Table 2.

Table 2:Main Hyperparameters in XGBoost.

Parameter	Description
eta	Learning rate
max_depth	Maximum depth of a tree
min_child_weight	The minimum sum of instance weight in the child
subsample	The ratio of training set sampled for each tree.
colsample_bytree	The ratio of columns sampled for each tree

Where,

Eta (default = 0.3) is the model's learning rate representing the step size shrinkage to reduce overfitting and make the model more robust, with typical values between 0.01 and 0.2.

Max_depth (default = 6) and min_child_weight (default = 0) relates to the structure of each tree and are used prevent over-fitting. Max depth represents the maximum depth of the tree as opposed to min_child_weight, which sets a criteria minimum sum of weights of all observations required in a child. Higher depth will make the model more complex and will allow the model to learn relations

very specific to a particular sample. Moreover, too high values for the min_child_weight parameter can lead to under-fitting; An appropriate value for both of the parameters can be found through cross-validation.

The last two hyperparameters, subsample (default = 1) and colsample_bytree (default = 1) occur once in every boosting iteration. Subsample denotes the fraction of the training samples randomly sampled for each tree. Lower values can make the algorithm more conservative, but extremely low values can lead to under-fitting. Similarly, colsample_bytee is the fraction of features (randomly selected) used to train each tree.

3.9. Model evaluation

3.9.1. Bias-variance tradeoff

Understanding bias and variance is critical for understanding the behavior of the predictive models. The term refers to the fact that when trying to make a statistical prediction, there is a tradeoff between the accuracy of the prediction and its precision (Doroudi, 2020). This is of significant importance because understanding how different error types contribute to bias and variance might help improve data fitting and ultimately develop more accurate models. The prediction of any ML model can be broken down into three main parts:

- Bias error
- Variance error
- Irreducible error

Bias represents the error between the actual value and the average model prediction. It also describes how well the model matches the training dataset. Variance refers to how much the target function estimate will change if different training data is used (Brownlee, 2016). Irreducible error is also known as "noise," and as the name suggests, it cannot be minimized by the ML algorithms. It usually comes

as a consequence of incomplete features or inherent randomness in the data. We can determine whether a model is underfitting, overfitting, or well-generalized based on its performance on unseen data (testing data). The problem of overfitting and underfitting is shown in the following figure:

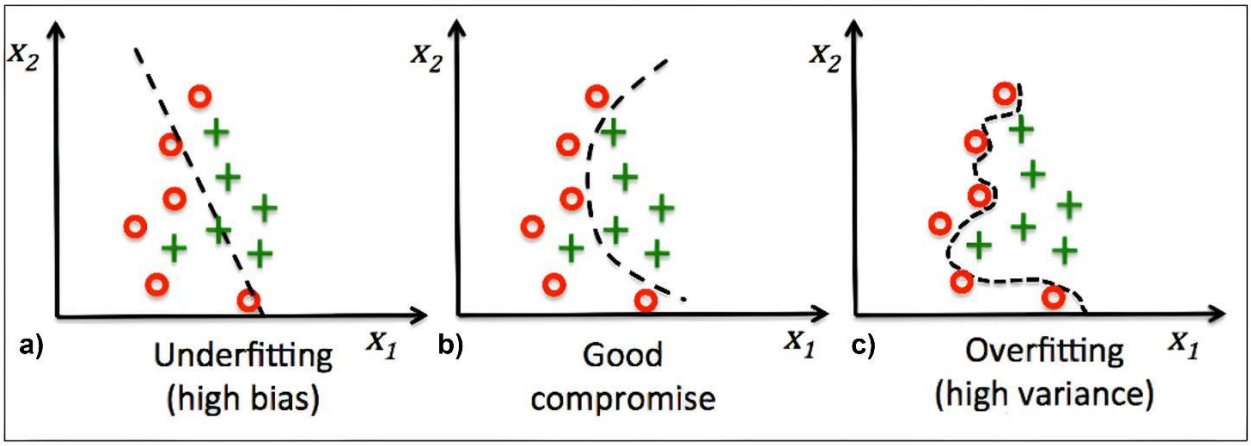


Figure 13:Example of underfitting, overfitting, and the sweet spot in supervised ML (Modified from Raschka, 2015).

Low variance (high bias) algorithms such as regression incur less complexity with simple or rigid underlying structures. This means that the model is not complex enough to capture the pattern in the training data well, leading to a high error in both training and testing data. This is known as underfitting (Figure 13 a).

On the other hand, high variance (low bias) algorithms such as decision trees involve more complexity with a flexible underlying structure. In this case, the model memorizes the training data, but it struggles to generalize the testing data, resulting in overfitting (Figure 13 c).

The tradeoff in complexity comes because an algorithm cannot be more complex and less complex at the same time. During the model training, the best overall accuracy is achieved with low bias and low variance (Figure 13 b), also known as the sweet spot between underfitting and overfitting. This implies that the ideal ML algorithm can accurately model the true relationship between the input and output and produces consistent predictions across different datasets.

This can be expressed as a total error:

$$Total\ error = Bias^2 + Variance + Irreducible\ Error$$

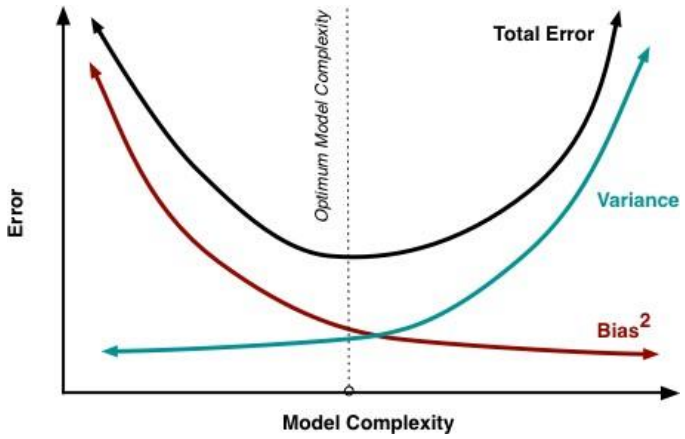


Figure 14:Bias and variance contributing to the total error (After Fortmann-Roe, 2022).

Ensemble methods such as boosting and bagging are two standard methods for finding the sweet spot or minimizing the total error. Bagging or bootstrap aggregating attempts to reduce the chance of overfitting complex models and decreases the variance by generating additional data while boosting attempts to improve the predictive flexibility of simple models by reducing the bias and variance. Consequently, other techniques, such as regularization, can also help reduce overfitting and help the model generalize better.

3.9.2. Stratified train-test-split

Moving forward, data splitting directly impacts the accuracy achieved by the model. Some classification problems do not have an equal class distribution in the dataset. In our case, we have two classes, NPZ and PZ. Using the conventional train-test-split, we might have only one target class in the training data. In this case, the model will not be able to learn what distinguishes the NPZ class from PZ class. As a result, we will not get a realistic estimate of how the model will perform.

Hence, a way to split the dataset into train and test sets so that the examples in each class account for the same proportion as per the original dataset. In other words, we want to make sure the class

proportions are preserved when splitting, especially important when dealing with class imbalances. In this thesis, data splitting was divided into three parts, including training, validation, and testing using a simple function in a python programming language. More specifically, 70% was limited to the training set and 30% to the validation set, while two unseen wells represent the testing set. It is a common rule to talk about validation sets when dealing with decisions that can affect the final model, such as selecting the classification algorithm or adjustment of its parameters. Whenever the performance of the final model is evaluated (final evaluation) on unseen data, one would rather speak of a test set. The need to separate the testing set from the validation set is because, in real life, we want to assess the model's true performance on unseen data that represents the population.

3.9.3. Performance evaluation

Classifier performance measures are calculated by comparing the predictions generated by the model with the true class labels present in the dataset. Common evaluation metrics for classification problems are Accuracy, Precision, Recall, and F1 score. Since the F1 score gives high weight to low values and there is no equal class distribution in the dataset, it was decided to use the F1 score as the primary evaluation metric. To compute this evaluation metric, it's first necessary to calculate the precision and recall, which are the foundations of the F1 score.

Precision is defined as the ratio of instances correctly classified as positive to all instances classified as positive, where TP is the number of True positives and FP is the number of False positives (also known as type I errors):

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

On the other hand, recall is defined as the ratio of instances correctly classified as positive to all positive instances, where FN is the number of False negatives (also known as type II errors):

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

Finally, F1 represents the harmonic mean of the precision and recall score and can be expressed as follows:

$$F_1 = 2 * \frac{(\text{ precision } * \text{ recall })}{(\text{ precision } + \text{ recall })}$$

Chapter 4

4. ANALYSIS AND RESULTS

This chapter includes the results of calculating the petrophysical parameters, including Vsh, porosity, and water saturation necessary to create the pay intervals. This stage is followed by the ML process with a strong emphasis on data analysis. The two main reservoirs, Ile and Garn, from two key wells are showcased.

4.1. Petrophysical analysis

For this study, core data for all the wells covering the hydrocarbon-bearing zones was provided.
The provided core porosity was considered total porosity, and it was compared with the calculated log derived total density porosity and sonic porosity. The water saturation was calculated using Archie's formula and the linear GR method was used to compute the shale volume. Water saturation results from well logs are compared with the core data for better calibration and evaluation. Figures 15 and 17 show the Vsh, porosity, and water saturation estimate from well logs in the Garn and Ile reservoirs for two key wells in the dataset.

Well 6506/12-8

6506-12/8 Garn Formation is filled with hydrocarbons (gas). As Figure 15, Gamma-ray log values vary between 22 and 139 API and increase relatively from minimum values, corresponding to a gradual upward change in the clay-mineral component, forming fining upward trend. This indicates a lithology change from sand to shale or an upward thinning of sand beds in a thinly interbedded sand-shale unit due to a decrease in the depositional energy. Strong hydrocarbon effects can be seen in the

density/neutron log. NEUT (neutron porosity) and DENS (density log) have higher separation (3911m-3928m, 3931m-3948m), which is an indication of gas. In addition, the resistivity log response is high while Sw (water saturation) shows lower readings (<0.5) and increases as we go up in the Formation.

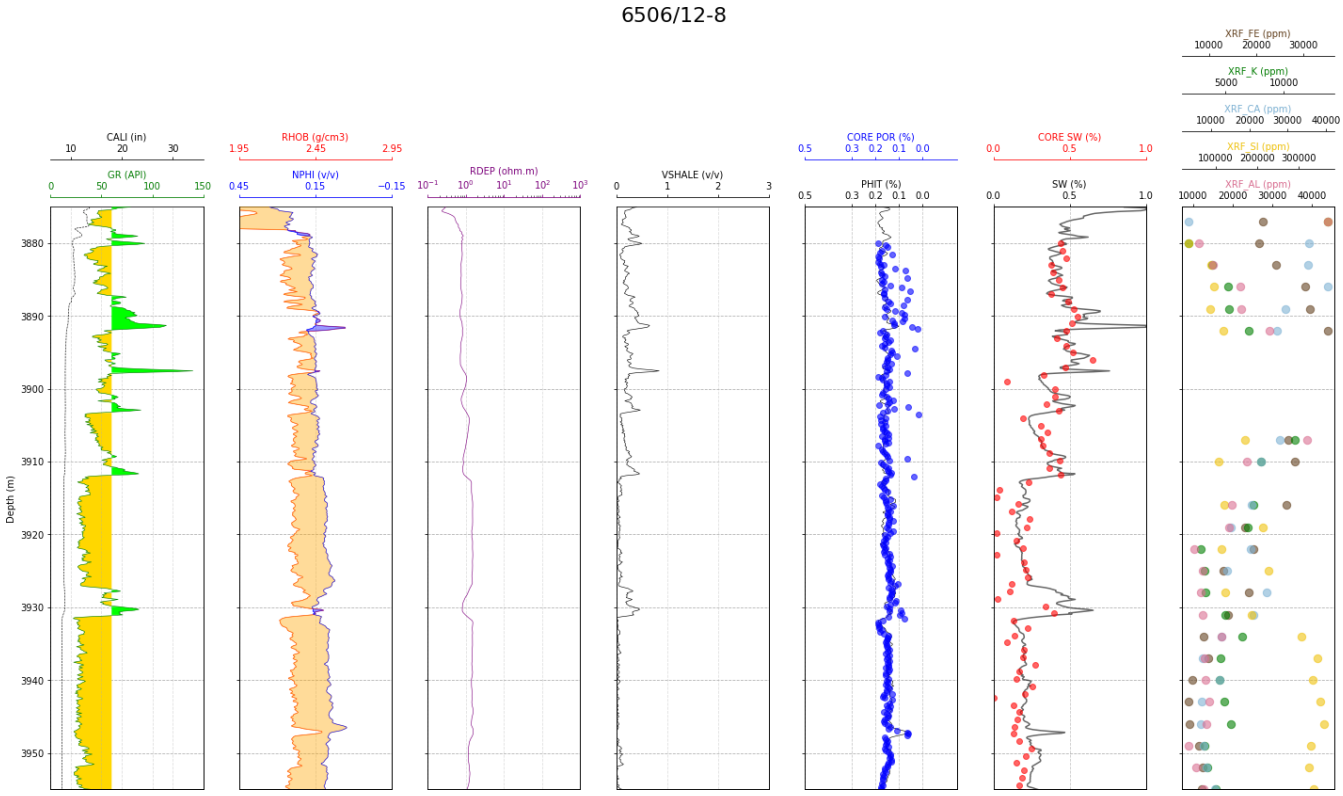


Figure 15:CPI (computer-processed interpretation) of Garn formation in well 6506/12-8.

To overcome the effects of bad hole conditions and variable hydrocarbon effects, both density porosity and sonic porosity methods were used to compute the porosity. Porosities ranging from 5 - to 19% were calculated, with strong hydrocarbon effects noted. The average porosity is 15% which makes Garn a good reservoir. The shale volume in the reservoir section of this well is given in Figure 15. The average concentration of the shale volume in the top part of the reservoir is less than 30%. With increasing depth in the reservoir section, the thickness of the shale zone is decreasing. In addition, in the lower part of the Garn Formation, in the interval between 3932m-3954m, XRF data confirms a high concentration of Silicon and a low concentration of Potassium, which indicates sandstone.

Further, histograms of shale volume, porosity, and water saturation can be seen in Figure 16. The histograms show that the reservoir intervals have clean sandstone of very good quality with extremely low shale content (0.15). In the reservoir interval, water saturation is 33%, indicating that this zone's hydrocarbon saturation is 67%.

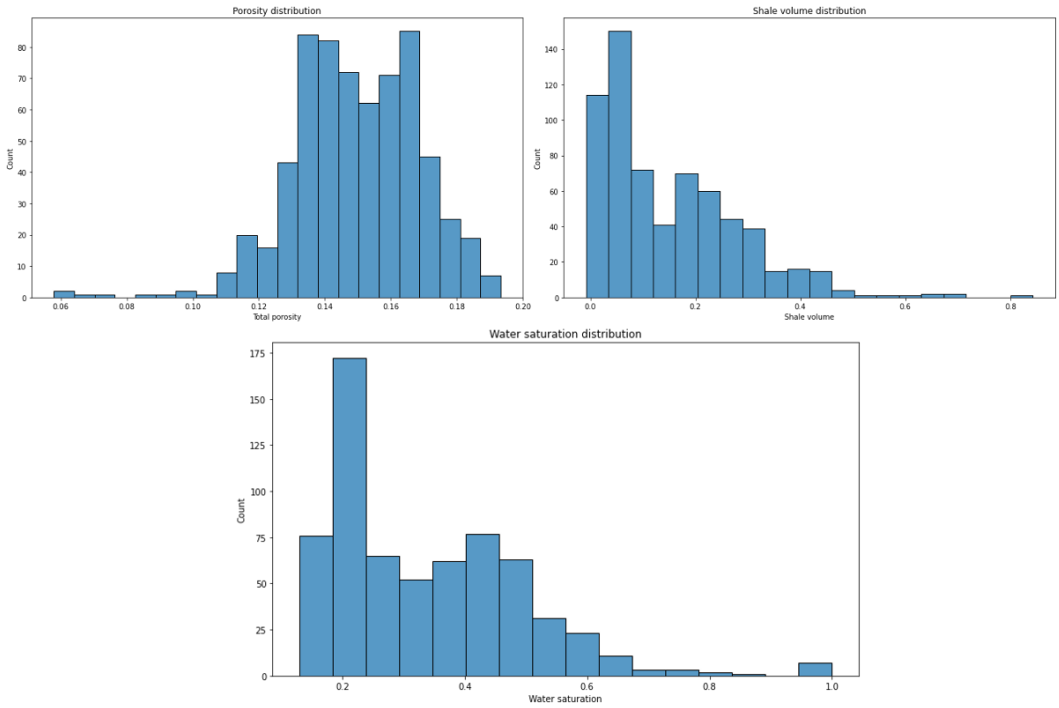


Figure 16:Histograms show the distribution of porosity, shale volume, and water saturation in the Garn Formation.

Table 3:Statistics of the calculated parameters for the Garn Formation in well 6506/12-8.

Zone	Top Fm. (m MD)	Bot Fm. (m MD)	VSH (frac.)	PHIT (frac.)	SW (frac.)
Garn	3875	3955	0.15	0.15	0.33

Well 6506/12-9 S

Compared to well 6506/12-8, the density log in well 6506/12-9 S is adversely affected by the poor hole conditions. The sonic log was used for porosity determination in the well section with the wash-out log, where the density log was adversely affected by the poor hole conditions.

The reservoir quality of the Ile Formation is generally fair based on porosity, shale volume, and water saturation calculations. The shale volume and water saturation in the reservoir interval are higher than the Garn Formation of well 6506/12-8, while the total porosity is lower. In well 6506-12/9 S, sand separation (low DENS and low NEUT) can be observed on density and neutron log along with high resistivity values on deep resistivity (RDEP), which further suggests that the Formation is gas/condensate filled as a large separation between density and neutron is observed. A similar fining upward trend of the GR log can be seen in Figure 17, indicating increasing shale content in the upper part of the reservoir. XRF data also confirms a relatively high concentration of Potassium and Silicon in the interval between 4464m and 4477m, which could indicate clay minerals presence. Moreover, Calcium concentration is low throughout the reservoir, which is an indication that the reservoir is sandstone.

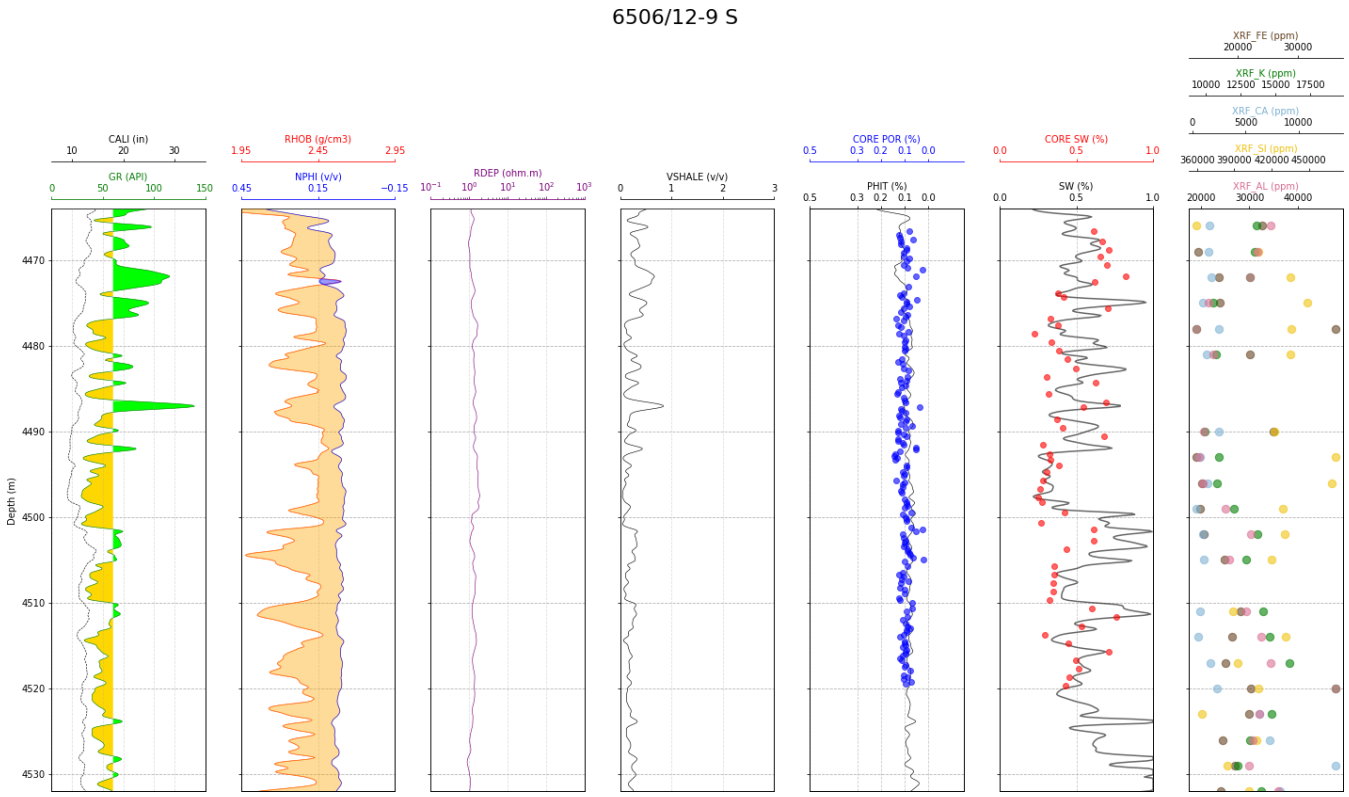


Figure 17:CPI (computer-processed interpretation) of Ile formation in well 6506/12-9 S

As shown in the log plot and histograms (Figures 17 and 18), the average percentage of the shale volume distribution is 22%, and the maximum concentration lies in the depth interval 4470-4477m. A very low shale volume concentration characterizes the lower part of the reservoir. Porosities ranging from 3 - 21% were calculated with strong hydrocarbon effects being noted, which makes the Ile a fair reservoir, especially in the lower part where the sandstones are more homogenous. Water saturation increases as we go up in the Formation. The summarized average shale volumes, porosities, and water saturations of the reservoir section are given in Table 4.

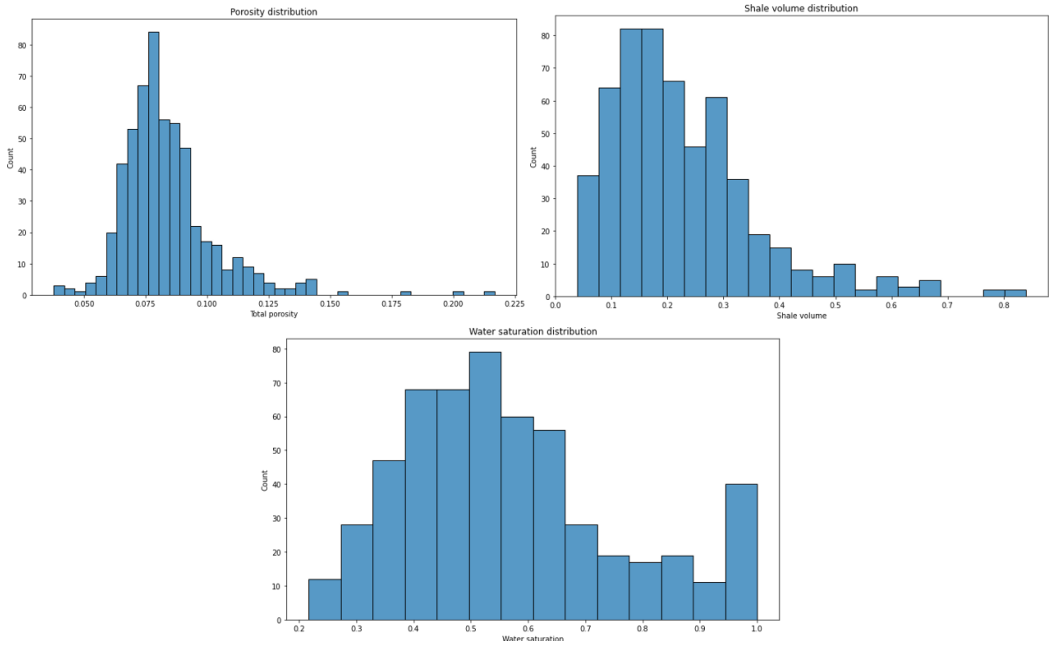


Figure 18:Histograms show the distribution of porosity, shale volume, and water saturation in the Ile Formation.

Table 4:Statistics of the calculated parameters for the Ile Formation in well 6506/12-9 S.

Zone	Top Fm. (m MD)	Bot Fm. (m MD)	VSH (frac.)	PHIT (frac.)	SW (frac.)
Ile	4464	4532	0.22	0.09	0.56

Based on the Vsh, total porosity, and water saturation, reservoir net pay zones are marked from the gross formation intervals. After calculating the reservoir properties such as porosity, the volume of shale, and water saturation, the pay zone intervals are estimated within the reservoir zones (Figure 19). In this study, for defining the reservoir interval, a volume of shale less than 50%, total porosity higher than 10%, and water saturation less than 50% are used (Table 5).

Table 5:Cutoff values considered for the field under study.

VSHALE	< 50 %
PHIT	> 10 %
SW	< 50 %

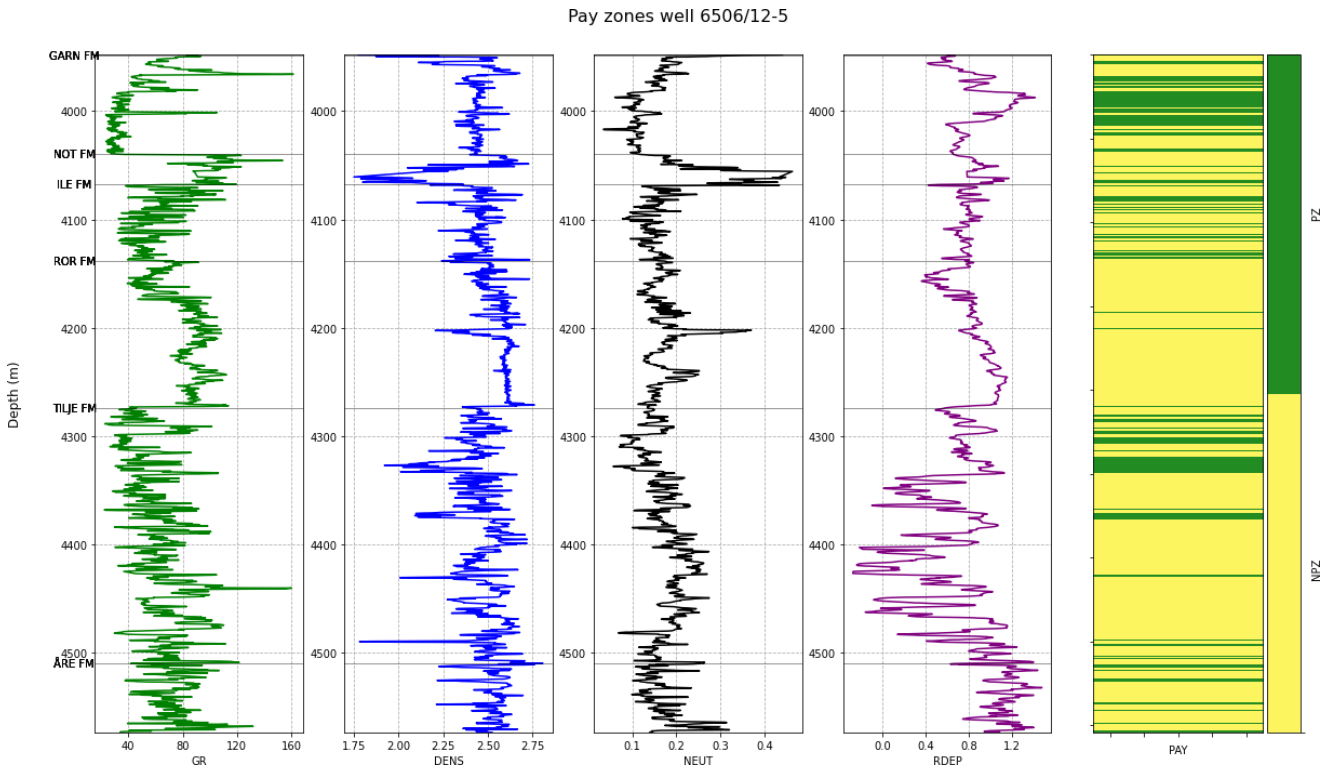
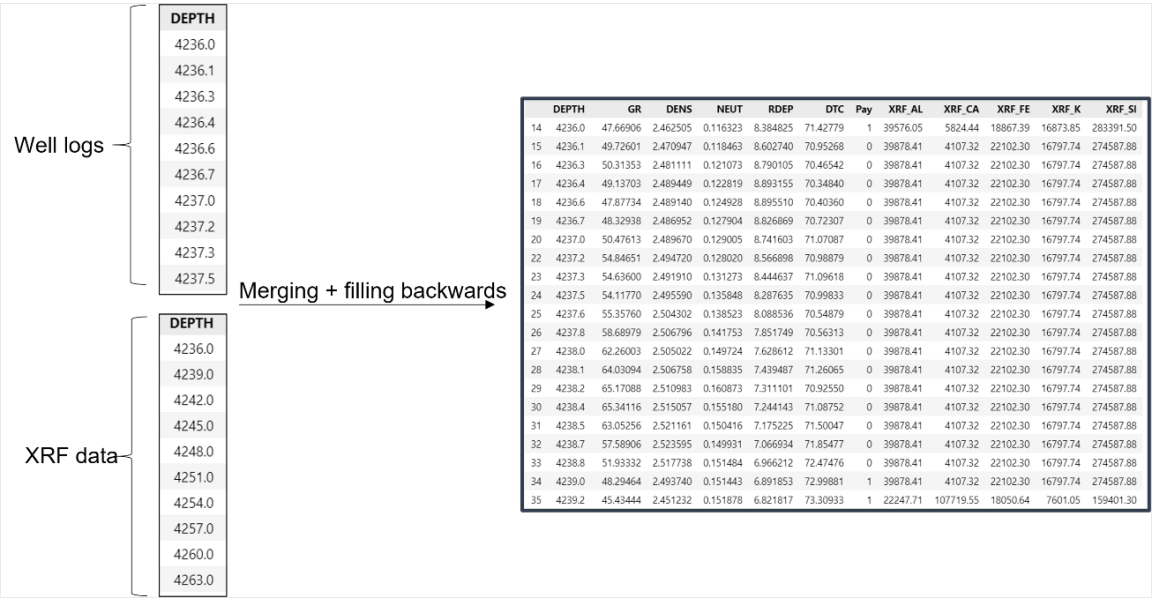


Figure 19:Pay zone intervals in well 6506/12-5. The location of the well is indicated in Figure 1. The green color represents the pay zone class (PZ), and the yellow color represents the non-pay class (NPZ).

4.2. Machine learning

4.2.1. Database

Before creating the final database, the well logs and XRF data were merged based on depth, as illustrated in Figure 20. The main challenge is the vertical resolution of the cuttings.



	DEPTH	GR	DENS	NEUT	RDEP	DTC	Pay	XRF_AL	XRF_CA	XRF_FE	XRF_K	XRF_SI
14	4236.0	47.66906	2.462505	0.116323	8.384825	71.42779	1	39576.05	5824.44	18867.39	16873.85	283391.50
15	4236.1	49.72601	2.470947	0.118463	8.602740	70.95268	0	39878.41	4107.32	22102.30	16797.74	274587.88
16	4236.3	50.31353	2.481111	0.121073	8.790105	70.46542	0	39878.41	4107.32	22102.30	16797.74	274587.88
17	4236.4	49.13703	2.489449	0.122819	8.893155	70.34840	0	39878.41	4107.32	22102.30	16797.74	274587.88
18	4236.6	47.87734	2.489140	0.124928	8.895510	70.40360	0	39878.41	4107.32	22102.30	16797.74	274587.88
19	4236.7	48.32938	2.486952	0.127904	8.826869	70.72307	0	39878.41	4107.32	22102.30	16797.74	274587.88
20	4237.0	50.47613	2.489670	0.129005	8.741603	71.07087	0	39878.41	4107.32	22102.30	16797.74	274587.88
22	4237.2	54.84651	2.494720	0.128020	8.566898	70.98879	0	39878.41	4107.32	22102.30	16797.74	274587.88
23	4237.3	54.63600	2.491910	0.131273	8.444637	71.09618	0	39878.41	4107.32	22102.30	16797.74	274587.88
24	4237.5	54.11770	2.495590	0.135848	8.287635	70.99833	0	39878.41	4107.32	22102.30	16797.74	274587.88
25	4237.6	55.35760	2.504302	0.138523	8.088536	70.54879	0	39878.41	4107.32	22102.30	16797.74	274587.88
26	4237.8	58.68979	2.506796	0.141753	7.851749	70.56313	0	39878.41	4107.32	22102.30	16797.74	274587.88
27	4238.0	62.26003	2.505022	0.149724	7.628612	71.13301	0	39878.41	4107.32	22102.30	16797.74	274587.88
28	4238.1	64.03094	2.506758	0.158835	7.439487	71.26065	0	39878.41	4107.32	22102.30	16797.74	274587.88
29	4238.2	65.17088	2.510983	0.160873	7.311101	70.92550	0	39878.41	4107.32	22102.30	16797.74	274587.88
30	4238.4	65.34116	2.515057	0.155180	7.244143	71.08752	0	39878.41	4107.32	22102.30	16797.74	274587.88
31	4238.5	63.05256	2.521161	0.150416	7.175225	71.50047	0	39878.41	4107.32	22102.30	16797.74	274587.88
32	4238.7	57.58906	2.523595	0.149931	7.066934	71.85477	0	39878.41	4107.32	22102.30	16797.74	274587.88
33	4238.8	51.93332	2.517738	0.151484	6.966212	72.47476	0	39878.41	4107.32	22102.30	16797.74	274587.88
34	4239.0	48.29464	2.493740	0.151443	6.891853	72.99881	1	39878.41	4107.32	22102.30	16797.74	274587.88
35	4239.2	45.43444	2.451232	0.151878	6.821817	73.30933	1	22247.71	107719.55	18050.64	7601.05	159401.30

Figure 20:The process of merging the XRF chemical elements with the wireline logs.

Cutting's data are sampled through the entire borehole approximately every 10m instead of well cores, typically taken for limited reservoir intervals. The cutting samples are first washed, dried, and photographed and then subjected to analyses to determine the chemical composition (XRF) and high-resolution white light and UV light photography. The final database consists of five well logs (GR, RDEP, DENS, NEUT, RDEP) and five XRF chemical elements (Al, Ca, Fe, K, Si), which represent the features of the ML process while the pay represents the label (Figure 21).

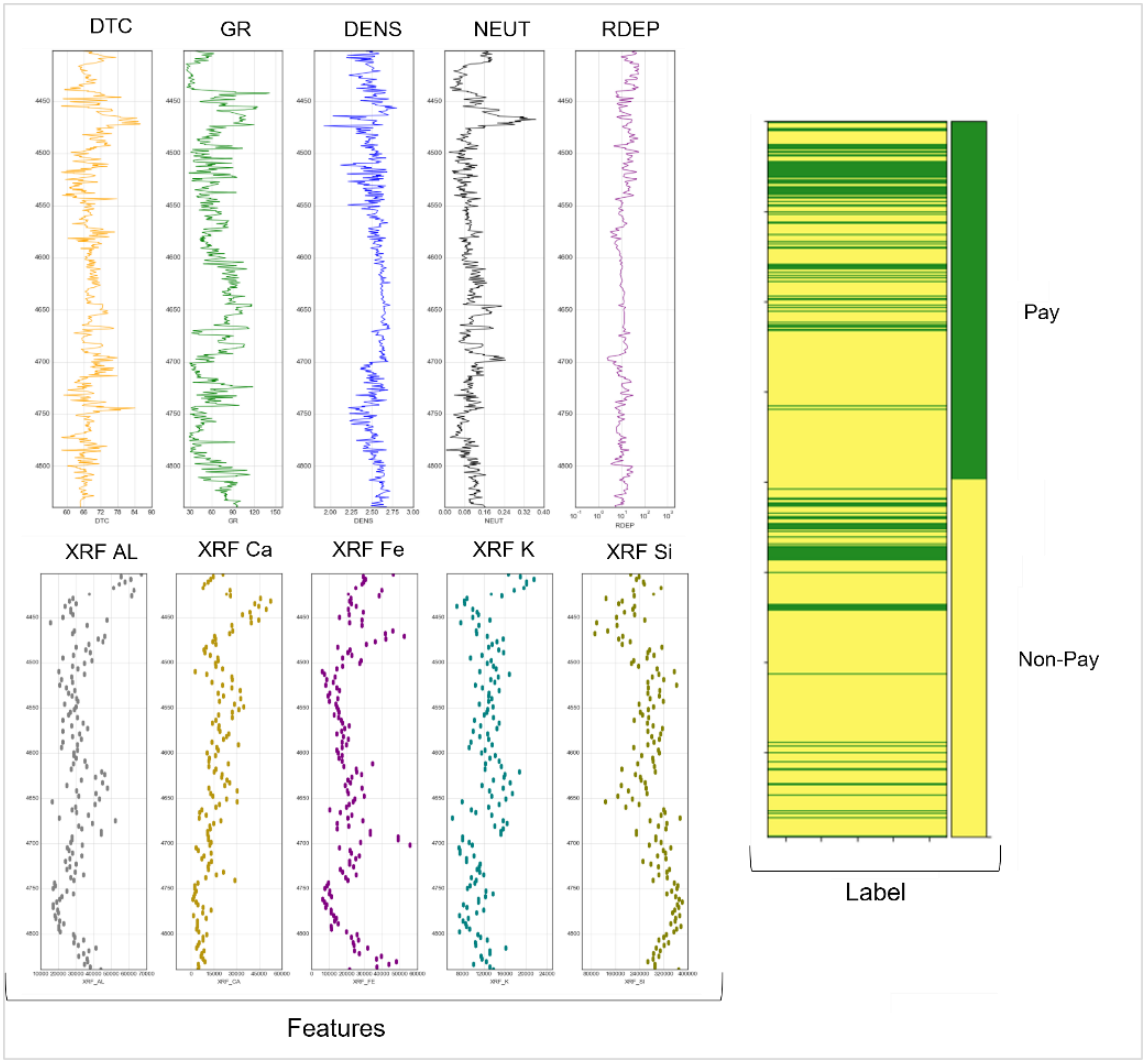


Figure 21:The features and the label considered in the dataset.

4.2.2. Statistical analysis

After assembling the dataset, the first step is to analyze and clean the dataset. The quality of the training data highly influences the performance of the ML models. Hence, data cleaning and processing are critical steps before any ML algorithms can be implemented, with a strong influence on the success of any data-driven project. Models trained on raw datasets are forced to take in noise as information, leading to accurate predictions when the noise is uniform within the training and validation set. However, it is prone not to generalize well when unseen data is shown to it. The dataset included in this thesis consists of 46738 rows × 18 columns (Figure 22).

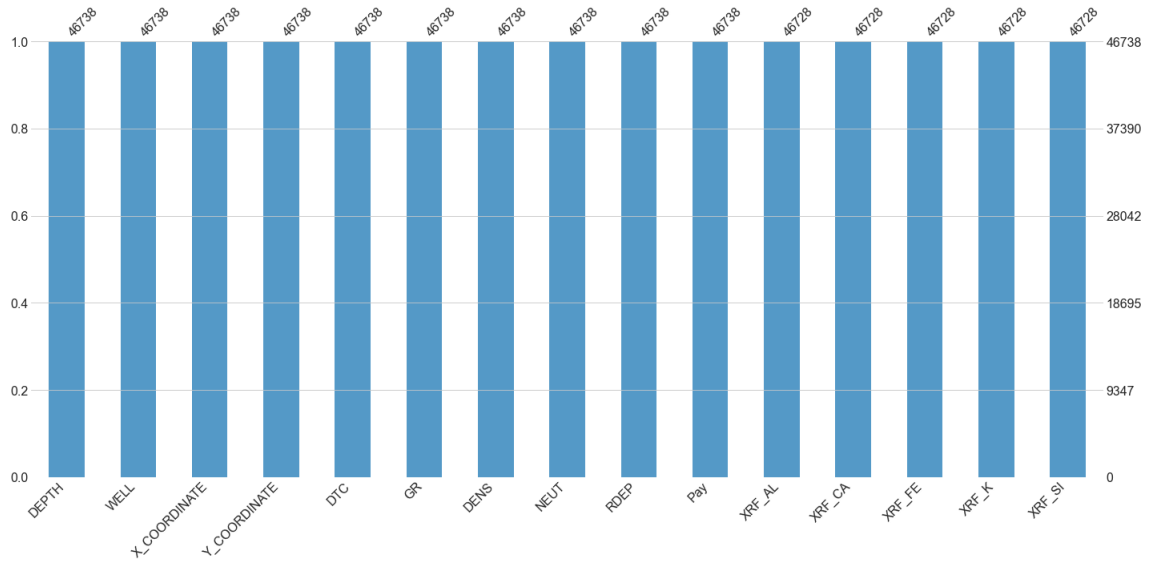


Figure 22:Data coverage.

Table 6 shows the data coverage is almost 100%, with only ten missing values present for XRF chemical elements, while the well logs do not present any missing values.

Table 6:Missing values percentage for the columns present in dataset.

Column	Missing number	Missing percent
XRF_Si	10	0.000214
XRF_K	10	0.000214
XRF_Fe	10	0.000214
XRF_Ca	10	0.000214
XRF_Al	10	0.000214
GR	0	0
NEUT	0	0
DENS	0	0
DTC	0	0
RDEP	0	0

An essential fact about the dataset is that it is unbalanced. In this study, the positive class is represented by the NPZ class, while the PZ class represents the minority or negative class. The majority class accounts for 81% of the total values with data points close to 38000, as depicted in Figure 23. The minority class accounts for only 19% of the dataset, with almost 9000 data points. Consequently, the ratio of Class-1 to Class-2 instances is almost 80:20 or more concisely 4:1. Working with unbalanced datasets is a fundamental problem in ML and very common when doing classification problems. This makes the model biased toward the majority class since it contains the most amount of instances/data points and performs poorer on the minority class.

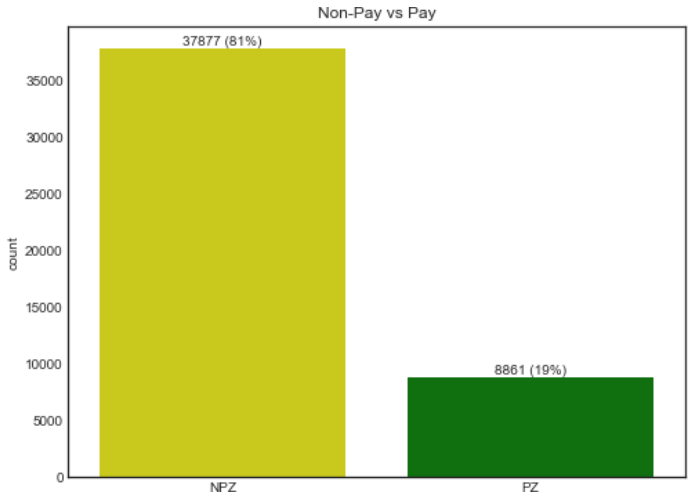


Figure 23:Pay distribution of the dataset

To address the class imbalance problem, different data manipulation techniques such as oversampling or under-sampling can be applied to enhance the performance of the minority class. However, in this study, the scale_pos_weight parameter from XGboost was used. Setting scale_pos_weight gives greater weight to the minority class. The value for the scale_pos_weight parameter comes because of dividing the majority class (NPZ) by the minority class (PZ). This affects scaling errors made by the model during training on the minority class and encourages the model to over-correct them (Brownlee, 2020).

As shown in Figure 24, all the wells in this study have variable pay distribution. In particular, the wells with the highest pay/non-pay ratio are wells 6506/12-3 and 6506/12-8. The imbalance ratio for well 6506/12-3 is 1.6, and 0.9 more for well 6506/12-8. On the other hand, the least amount of pay/non-pay ratio can be seen in wells 6506/12-10 A, 6506/11-5 S, and 6506/12-11 S. The imbalance ratio for these wells is 17, 7.2 and 7.4 respectively.

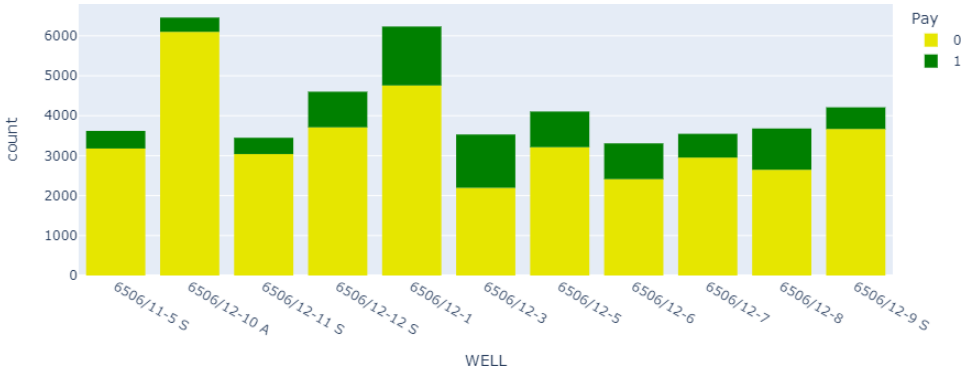


Figure 24:Pay distribution in each well 0-Non-Pay;1-Pay.

The main reservoirs in the Smørbukk field are the Middle Jurassic Garn, Ile, and Tofte Formations, with average pay close to 50% and 20%. On the other hand, the pay present in the Lower Jurassic Ror, Tilje, and Åre Formations is significantly less than in Garn and Ile Formations, as depicted in Figure 25.

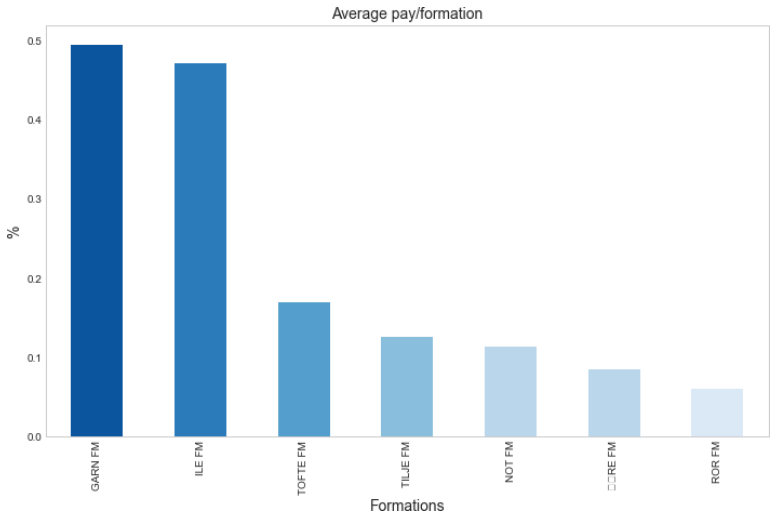


Figure 25:Average pay in each FM for the wells in the Smørbukk field.

Understanding the underlying data distribution before applying any statistical modeling approach is critical in finding an optimal solution for any ML problem. The distribution of the features that will be used to eliminate some of the outliers can be seen in Figure 26. Most of the features present a skew

distribution. An example of a feature that shows a Gaussian distribution is XRF Si. The features with a skew distribution give uneven range, which is an issue, especially if their range is huge (e.g., Neutron Log). This might be a problem when finding an optimal classification solution, especially when implementing distance-based and gradient descent-based ML algorithms. Adjusted (engineered) features will have a smaller and more even range.

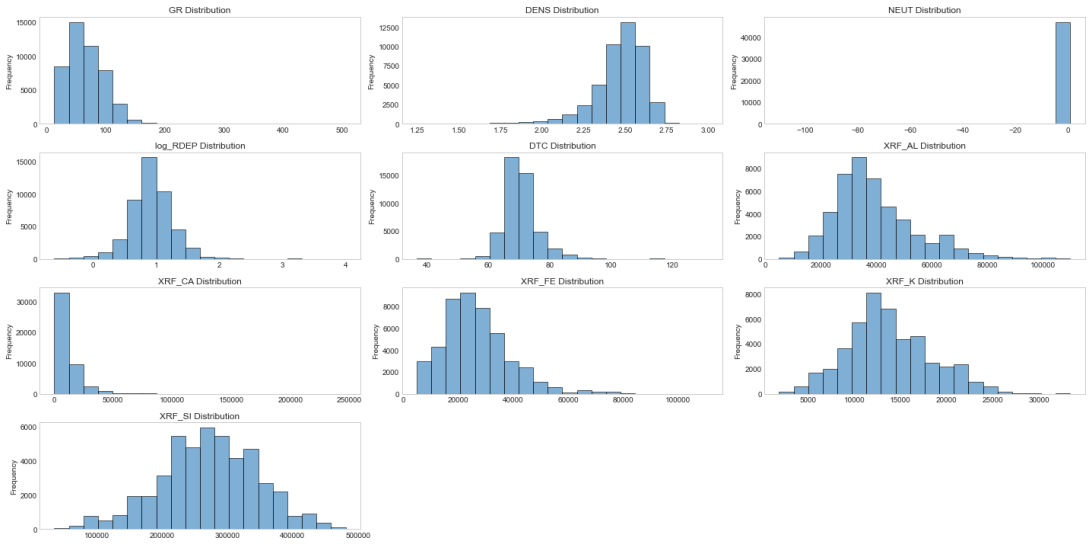


Figure 26:Features' (well logs and XRF chemical elements) distribution in the dataset.

Moreover, the boxplot's statistical summaries can help identify possible abnormal values that might be outside of the physical boundaries. This will have a positive impact on the accuracy of the ML models. Focusing on removing the "extreme outliers" rather than just outliers mitigates the risk of information loss which will cause the models to have lower accuracy. From the boxplot (Figure 27), extreme outliers (>300 API) are present in the GR log. This could be indicated by the enrichment of radioactive minerals such as k-felspar, zircon, or mica. XRF data also confirms the presence of high values of Uranium and Thorium. Extreme negative outliers can be seen in the Neutron Log. One data point has an extremely high negative value (-109 g/cm^3) in well 6506/12-1 and corresponds to Åre Formation. Moreover, a significant difference between the 75th quantile and the maximum DTC value can be noticed for both pay classes. High DTC values (>100 µs/ft) correspond

with the Åre Formation and are present in three wells (6506/12-10A, 6506-12-1, 6506/12-6).

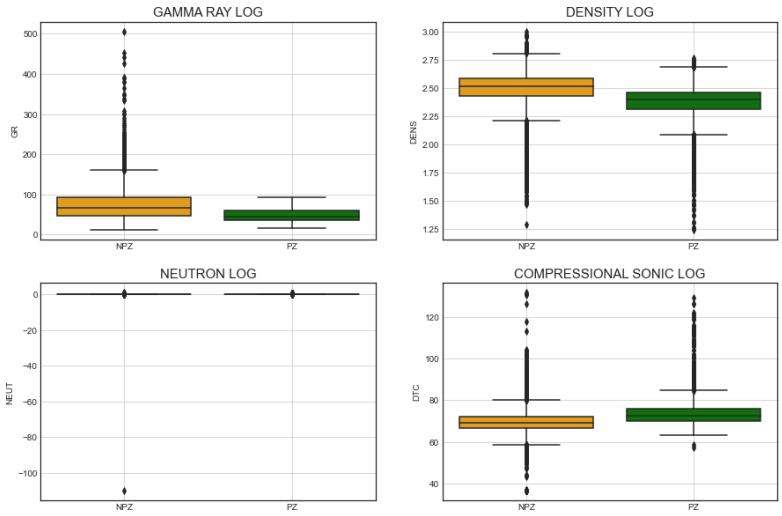


Figure 27:Statistics of the wireline logs labeled by pay classes

This could be explained by the presence of coal seams at the base of the Formation that are up to 8m thick (NPD, 2022).

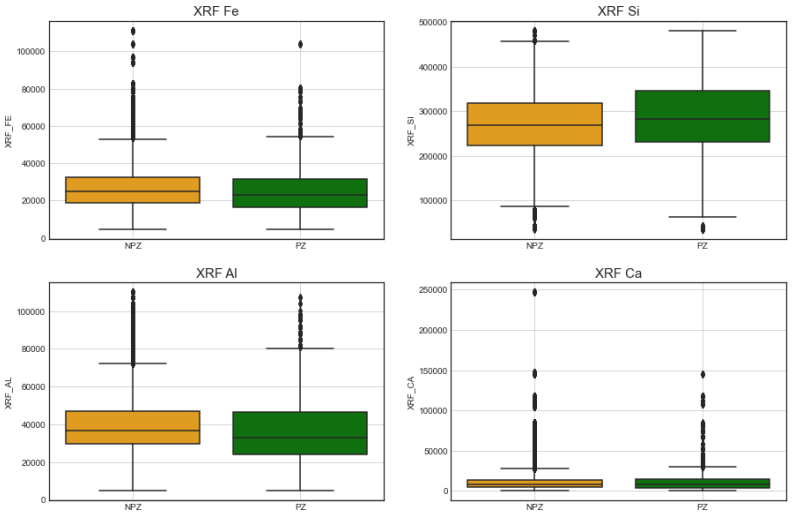


Figure 28:Statistics of the XRF chemical elements labeled by pay classes

Regarding the XRF data, approximately 1.6 % of the XRF Ca values are larger than 50 000 ppm. This chemical element significantly differs between the 75th quantile and maximum value. Similarly, XRF Al has 1.63 % of the values larger than 80 000 ppm. In addition, the extreme outliers for XRF Fe (>80

000 ppm) account for only 0.27% of the dataset as opposed to XRF Si, which has more negative outliers present for both classes.

Consequently, the upcoming subsection focuses on preserving data quality and magnitudes using different feature engineering techniques.

4.2.3. Feature engineering

It is widely known that ML algorithms require features with specific characteristics to work correctly. Logarithm transformation (or log transform) is one of the most used mathematical transformations in feature engineering. This method was used to log transform the deep resistivity log (RDEP). As a result, the effect of outliers due to the normalization of magnitude differences decreases, and the model becomes more robust.

Previously identified outliers for both well logs and XRF data were removed using Isolation Forest (Figures 29 and 30). Based on the previous analysis, the outlier fraction in the algorithm was set to 0.002 %.

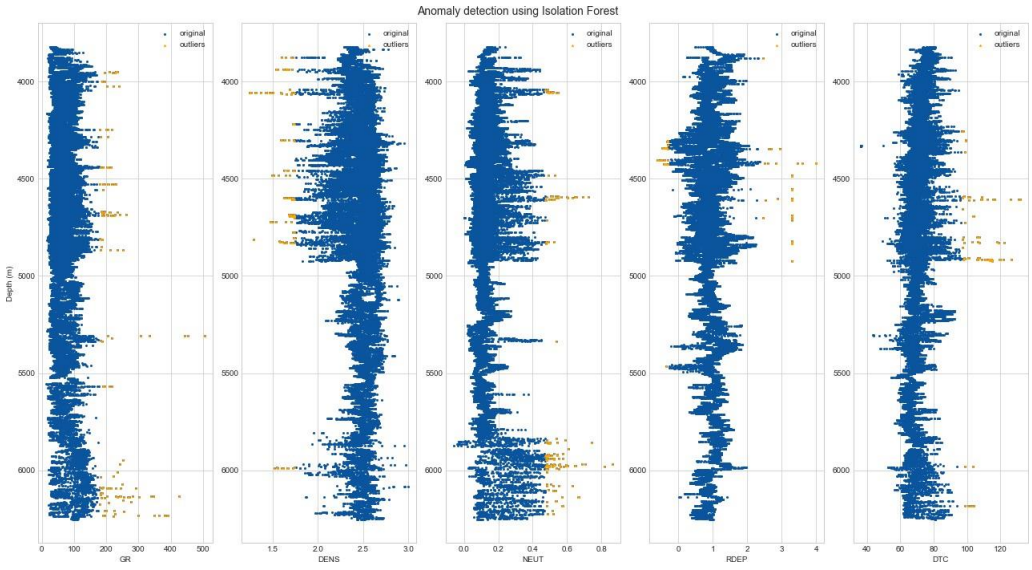


Figure 29:Outlier removal results using Isolation Forest for the Wireline logs.

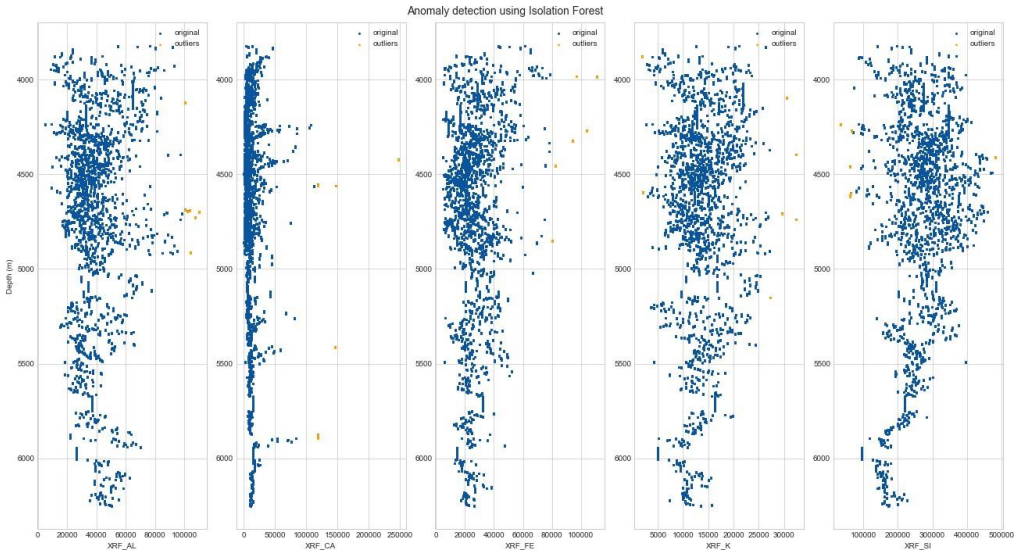


Figure 30:Outlier removal results using Isolation Forest for the XRF chemical elements.

As shown in both figures, the largest number of outliers were removed from the NEUT, GR, and DENS log instead of the least number of outliers removed from DTC and XRF K. In total, less than 0.3 % of outliers were removed for each feature (Figure 31). Finally, the mean outliers removed for all the columns present in the dataset is 159.6.

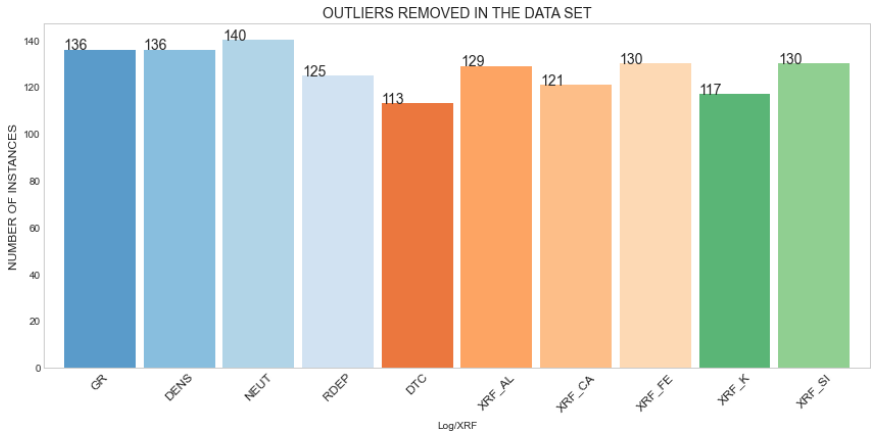


Figure 31:Outlier removal results using Isolation Forest for the XRF chemical elements.

On the other hand, feature scaling was performed using Standard Scaler, and the missing values were replaced by the mean value of the affected columns. Screening the features involved in the ML process could reduce information redundancy and help with model interpretability. Moreover, before building

any statistical models, ML, or deep learning models, it is recommended to check Spearman's Correlations between the variables.

Figure 32 shows Spearman's Correlations heatmap between the variables present in the dataset.

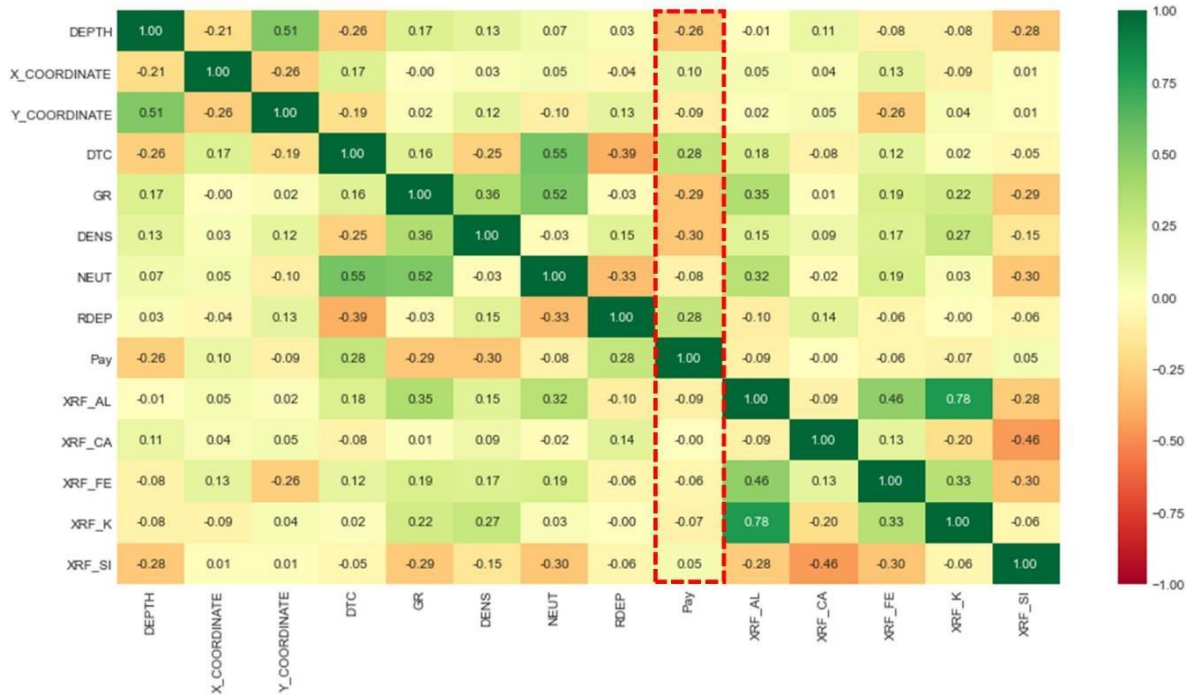


Figure 32:Spearman correlation between the features and the label present in the dataset.

Instead of gut feeling and domain expertise, the Spearman correlation heatmap can reveal the feature importance based on the correlation between the variables. From the correlation summary (Figure 32) for all the features present, it can be noticed that DTC and RDEP have the highest positive correlation (0.28) with the label. This relationship is expected because higher values of compressional slowness indicate an increase in porosity and less mineral content presence. Similarly, high resistivity values may indicate a hydrocarbon-bearing formation, while GR is a very good indicator for lithology. The other two important features are DENS (-0.30) and depth (-0.26), which negatively correlate with the label. The XRF chemical elements are not as important as expected. XRF Al, K, and Si, which could be related to clay minerals or sandstone presence (in the case of Si), have a low negative and positive correlation with the label. Besides the existing linear correlation between the previously mentioned features, XRF Ca, did not show any apparent relationship with the label.

4.2.4. ML model selection

To estimate the accuracy of the ML models, the average stratified 10k-fold cross-validation technique was considered. The performance of the ML models was evaluated on the validation set using the F1 score.

Figure 33 summarizes the performance summary for each of the models. As can be seen from the plot, the XGBoost and Random Forest algorithms which can handle high dimensional and complex data structures, are the highest performing algorithms. In comparison to XGboost, Random Forest performed 2.2 % lower. At the same time, Logistic Regression showed the least favorable performance.

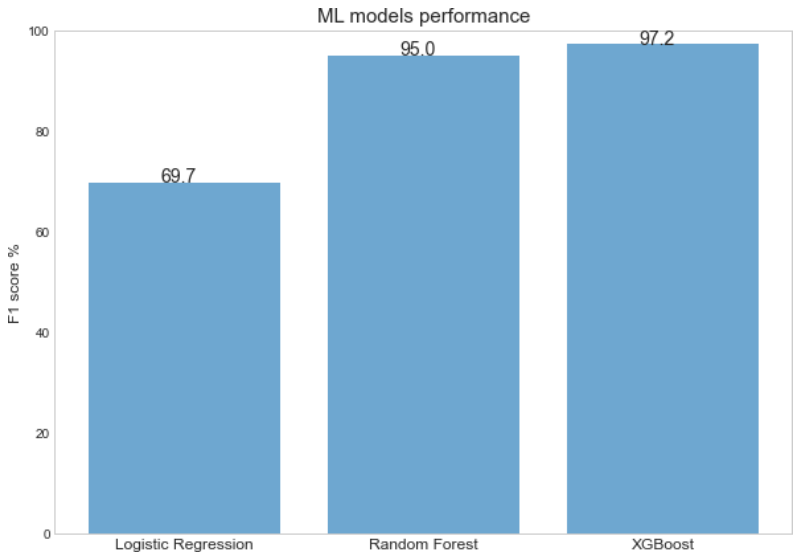


Figure 33:ML models performance using stratified ten k-fold cross-validation.

Logistic regression was the fastest ML model to validate but the lowest-performing in comparison with Random Forest and XGboost (Figure 34). It is interesting to note that Random Forest was far and away the slowest to validate; however, it was the second-best performing. Generally, a trade-off in performance and validation time can be seen. As a result, the XGBoost model was selected to create further different models, which will be showcased in the next chapter.

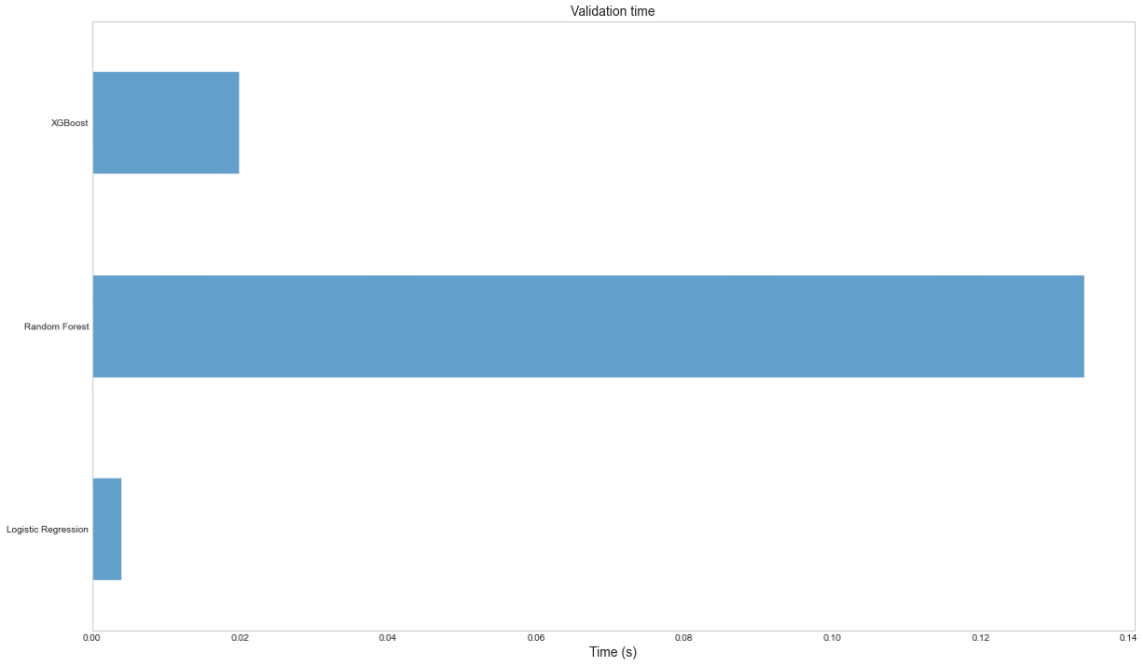


Figure 34:Validation time by model

4.2.5. XGBoost models

After selecting the best-performing model, it was decided to build four models to identify the label "pay" for two test wells using different features as input. Wells 6506/12-8 and 6506/12-6 are used to evaluate the true performance of the models. The location of the wells is indicated in Figure 6.

In model 1, only DEPTH, GR, and RDEP logs were used as input features to identify pay zones. This was decided because GR and RDEP logs are more commonly acquired in wells than the rest of the logs. In addition to DEPTH, GR, and RDEP logs, model 2 consists of XRF chemical elements. Finally, model 3 consists of all the available logs, including GR, RDEP, DENS, NEUT, DTC, and DEPTH, while Model 4 consists of the same logs + XRF chemical elements.

The scale_pos_weight parameter from XGBoost was used to combat the class imbalance problem. The scale_pos_weight parameter was set to 2.5 for well 6506/12-8 and 2.7 for well 6506/12-6. The performances of the ML models for the pay zone identification are summarized in Tables 7 and 8 using the F1 score for the training, validation, and testing set.

Table 7:Models' evaluation using F1 score for well 6506/12-8.

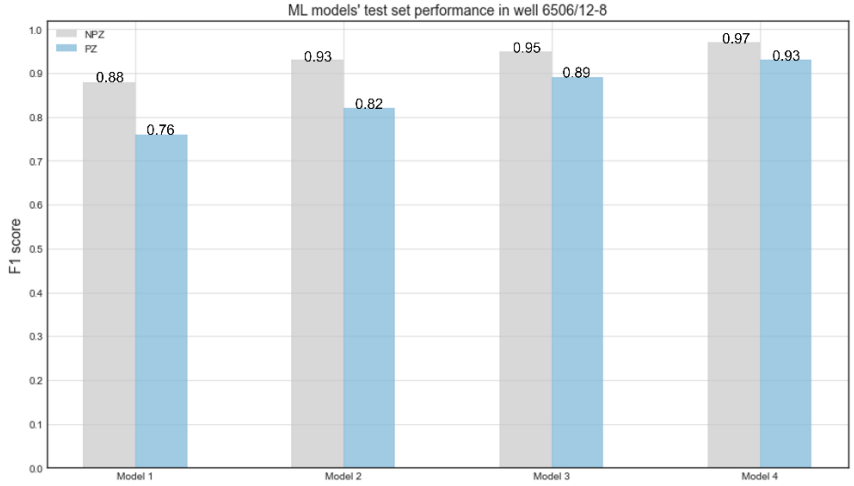
Class	Model 1			Model 2			Model 3			Model 4		
	Tra.	Val.	Tes.	Tra.	Val.	Tes.	Tra.	Val.	Tes.	Tra.	Val.	Tes.
NPZ	0.96	0.94	0.88	0.98	0.96	0.93	1	0.99	0.95	1	0.99	0.97
PZ	0.84	0.77	0.76	0.89	0.83	0.82	0.99	0.96	0.89	1	0.97	0.93

Table 8:Models' evaluation using F1 score for well 6506/12-6.

Class	Model 1			Model 2			Model 3			Model 4		
	Tra.	Val.	Tes.	Tra.	Val.	Tes.	Tra.	Val.	Tes.	Tra.	Val.	Tes.
NPZ	0.96	0.95	0.91	0.98	0.96	0.93	1	0.99	0.98	1	0.99	0.99
PZ	0.82	0.77	0.73	0.90	0.83	0.78	1	0.97	0.95	1	0.97	0.96

The results illustrate that input features such as GR and RDEP, combined with XRF data, can perform satisfactorily in both wells. However, when including all the available logs, the overall performance of ML models is increasing notably (Figure 35).

Based on metric analysis (Tables 7 and 8), model 4 has the lowest error among both wells on training, validation, and testing sets. With a limited number of logs (GR and RDEP) used as input, XRF data adds 5 and 6 % improvement on the testing set when identifying the PZ class and 2 and 5% improvement when identifying the NPZ class.



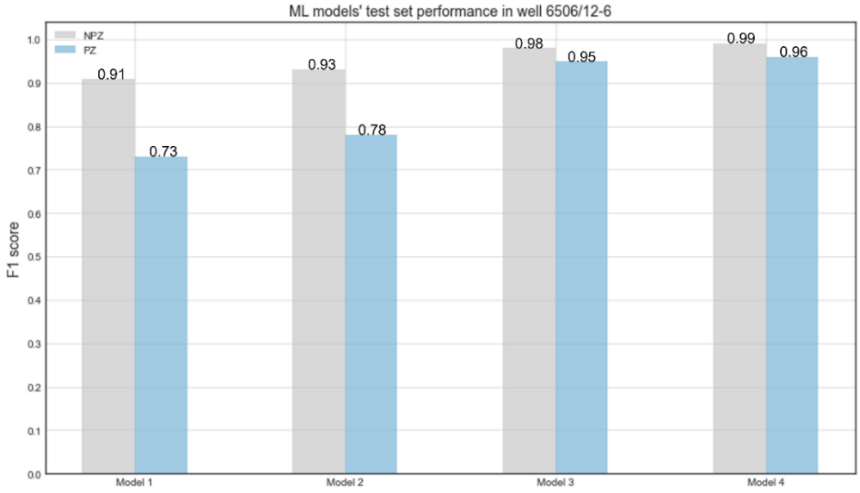


Figure 35:ML models performance on PZ and NPZ class in well 6506/12-8 and well 6506/12-6.

Based on Figures 37 and 38, which represent the actual value vs. prediction outcomes, it is observed that there is a very good correlation between these two values by using model 3 and model 4. All the models correctly predicted the non-reservoir Ror Formation in both wells. Model 2 has a satisfactory performance on identifying the thicker pays but struggles to identify the thin pays corresponding to Tilje and Ile Formations in well 6506/12-8 (Figure 37). However, when adding XRF chemical elements to GR and RDEP (model 2), a clear improvement can be seen in the Ror Formation, were previously wrong identified pays in model 1 were corrected and matched with the actual pay. This could be explained by the fact that mudstones are the dominant lithology in the Ror Formation which sometimes contain interbedded silty and sandy sequences (NPD, 2022). XRF data also confirms high Fe, K, Al, and Si values, which could be related to the lithology, hence the significant improvement (Figure 36 a).

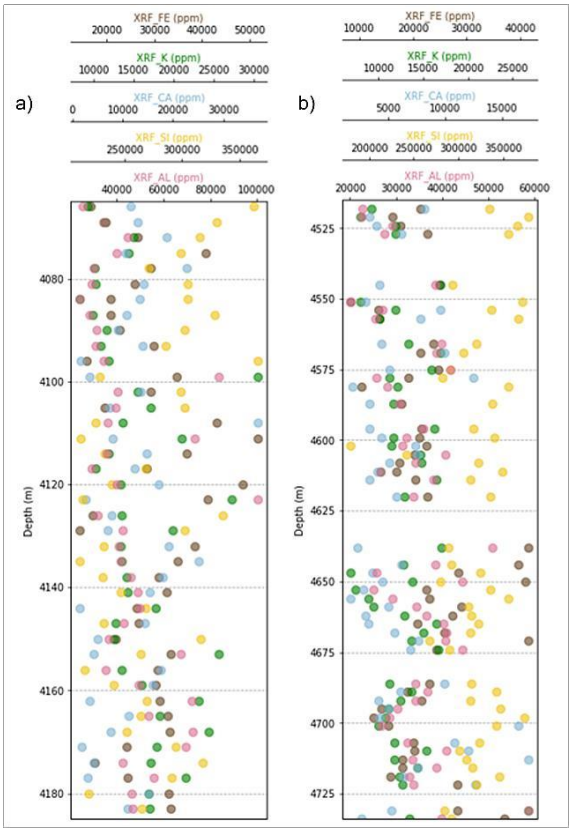


Figure 36: XRF chemical elements for a) well 6506/12-8 Ror FM; b) well 6506/12-6 Tilje FM.

In addition, when incorporating all the available logs, the model can capture the underlying data trends much better, leading to a perfect match with the actual pay. In particular, adding XRF data to the preexisting suits of logs (model 4) leads to 4% and 1% improvement in PZ class in the testing set in well 6506/12-8 and well 6506/12-6, respectively. Moreover, there is a 2% and 1% improvement in the NPZ class. It can be seen that the pays present in the clean sands of the Garn Formation were identified by all models (Figure 37).

In the lower part of the Ile Formation, in the interval between 4032m-4040m, where the sandstones become more heterogenous in well 6506/12-8, there is a significant improvement in identifying the NPZ class by adding XRF chemical elements to model 3. Moreover, a slightly better representation of the pay corresponding to Tilje Formation in the interval between 4205m-4215m can be seen. In both cases, the lithology within the reservoir intervals varies from clean sand, shaly sand, and shale.

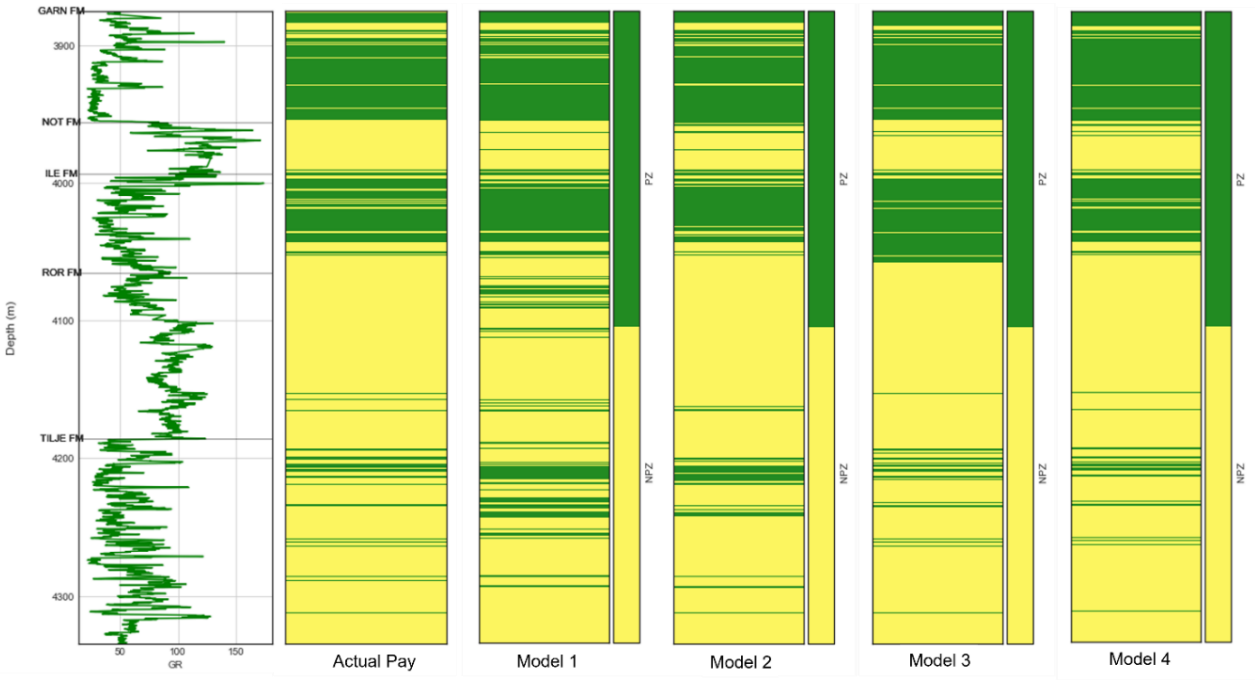


Figure 37: Pay zone prediction results by different models in well 6506/12-8. The green color represents the pay zone class (PZ), and the yellow color represents the non-pay class (NPZ).

On the other hand, when comparing the predictions and experimentally measured values in well 6506/12-6, the improvement is only 1% in both classes. An improvement can be noticed in the lower part of the Ile Formation in the interval between 4375m-4380m, where the sandstones in the reservoir are more heterogeneous. The previously wrong predicted NPZ class was corrected and matches much better with the actual pay (Figure 38, model 4). On the other hand, previously unidentified pays corresponding to the clean sands of the Garn Formation using model 1 were partially identified using model 2. Besides, the comparison between model 3 and model 4 in this specific interval is relatively the same. In addition, previously wrong predicted thin pays corresponding to the Tilje Formation in the interval between 4190m-4210m were corrected using model 4. The better match with the actual values could be attributed to the increasing shale content in the upper part of the reservoir, where XRF data also confirms high values of K, Al, and Si (Figure 36 b). As a result of these findings, model 4 was selected for further optimization to increase the overall performance of the ML model to accurately predict the PZ and NPZ classes.

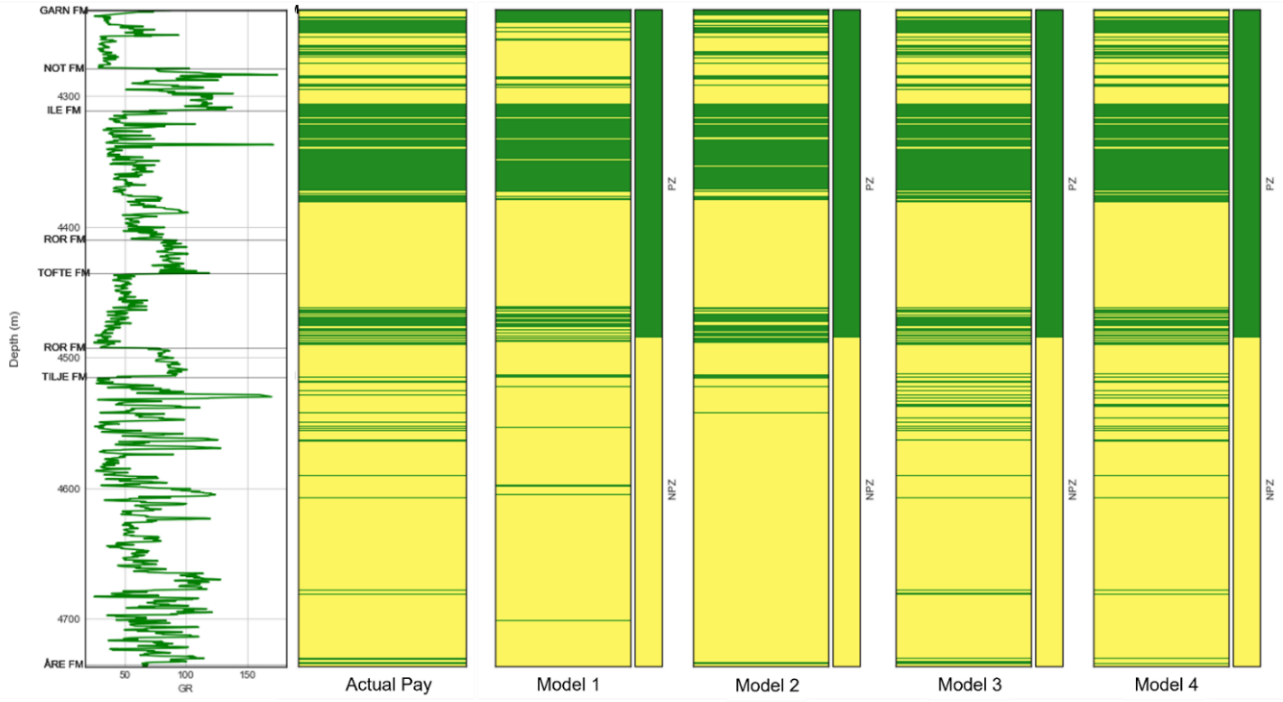


Figure 38:Pay zone prediction results by different models in well 6506/12-6. The green color represents the pay zone class (PZ), and the yellow color represents the non-pay class (NPZ).

4.2.6. Model optimization

The next step is to increase the model performance using GridSearchCV with cross-validation. The grid search and tenfold cross-validation are used. One drawback experienced when using this method was the runtime since it is computationally expensive to run. As the dataset is relatively large and especially the hyperparameter range, in addition to GridSearchCV, RandomizedSearchCV was tested. Since the latter is not an exhaustive method like GridSearchCV, it reduces the chances of overfitting the training data. The lists of the tuned hyperparameter search interval and the optimal hyperparameter values applied can be seen in Table 9.

Table 9: Main hyperparameters used.

Algorithm	Hyperparameters Tuned	Search Interval	Optimal Values
XGBoost	learning_rate	0.01 – 0.2	0.05
	subsample	0.5 – 1	0.8
	colsample_bytree	0.5 – 1	0.8
	max_depth	3 – 10	9
	n_estimators	100-900	900
	min_child_weight	1-7	3

After performing both methods, the optimal values chosen for the models are learning_rate of 0.05, a subsample of 0.8, colsample_bytree of 0.8, max_depth of 9, n_estimators of 900 and min_child_weight of 3.

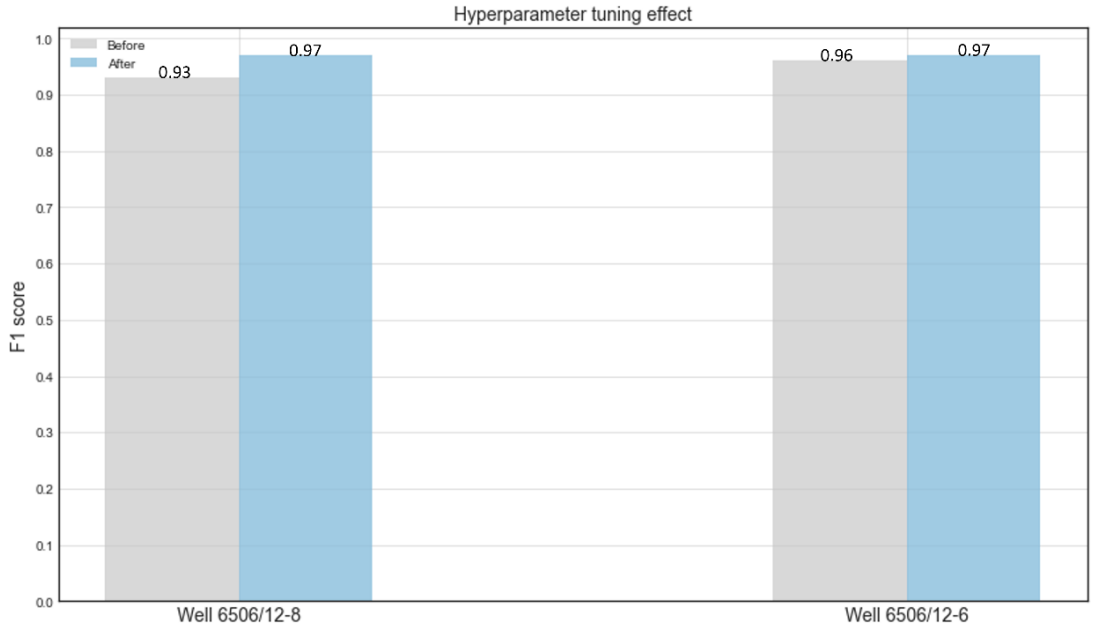


Figure 39:Hyperparameter tuning effect on PZ class in well 6506/12-8 and well 6506/12-6.

For both wells, hyperparameter tuning added 1 and 4% improvement on the testing set PZ class from an F1 score of 0.93 and 0.96 to 0.97, as seen in Figure 39. On the other hand, there is a 2% improvement on NPZ class in well 6506/12-8 and a minimal improvement (1%) for well 6506/12-6. Similarly, the training and validation set performance remained the same for both wells. The final

performance of model 4 in both wells can be seen in Figures 40 and 41, where in addition to the available logs, XRF chemical elements are plotted.

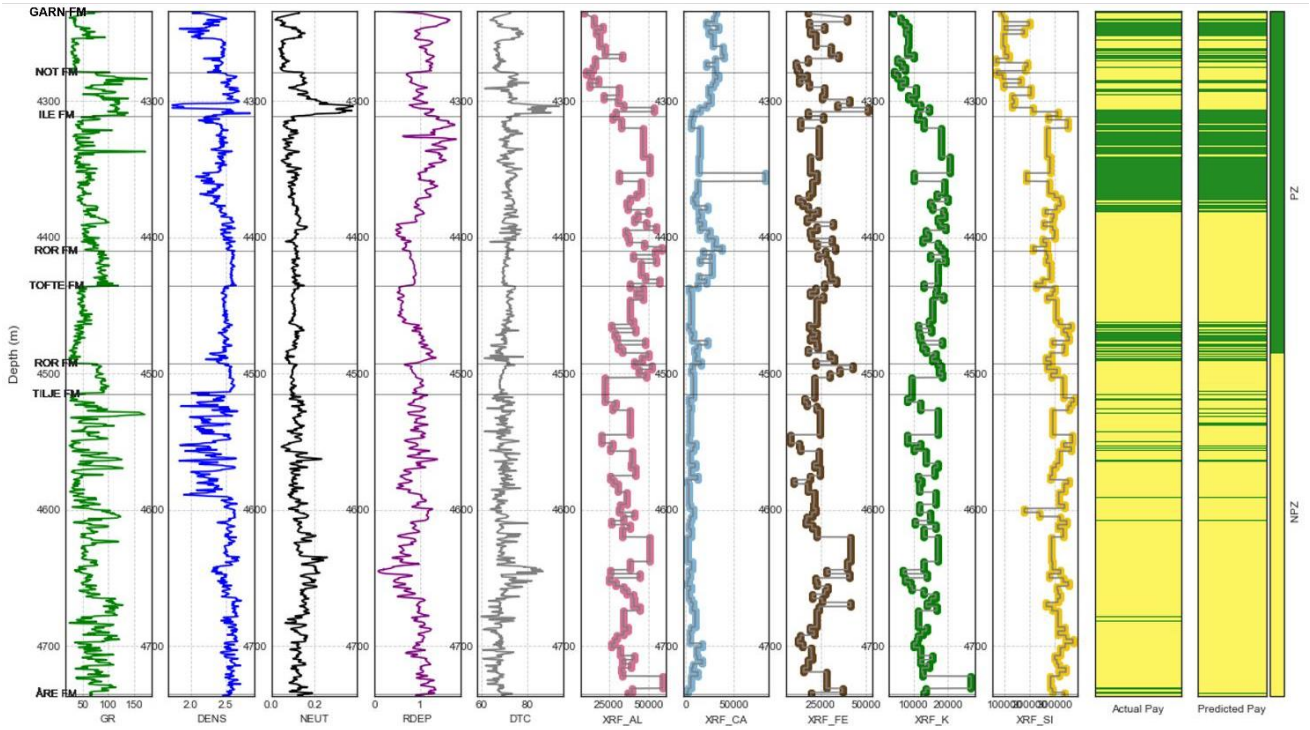


Figure 40:Model 4 performance after optimization in well 6506/12-6

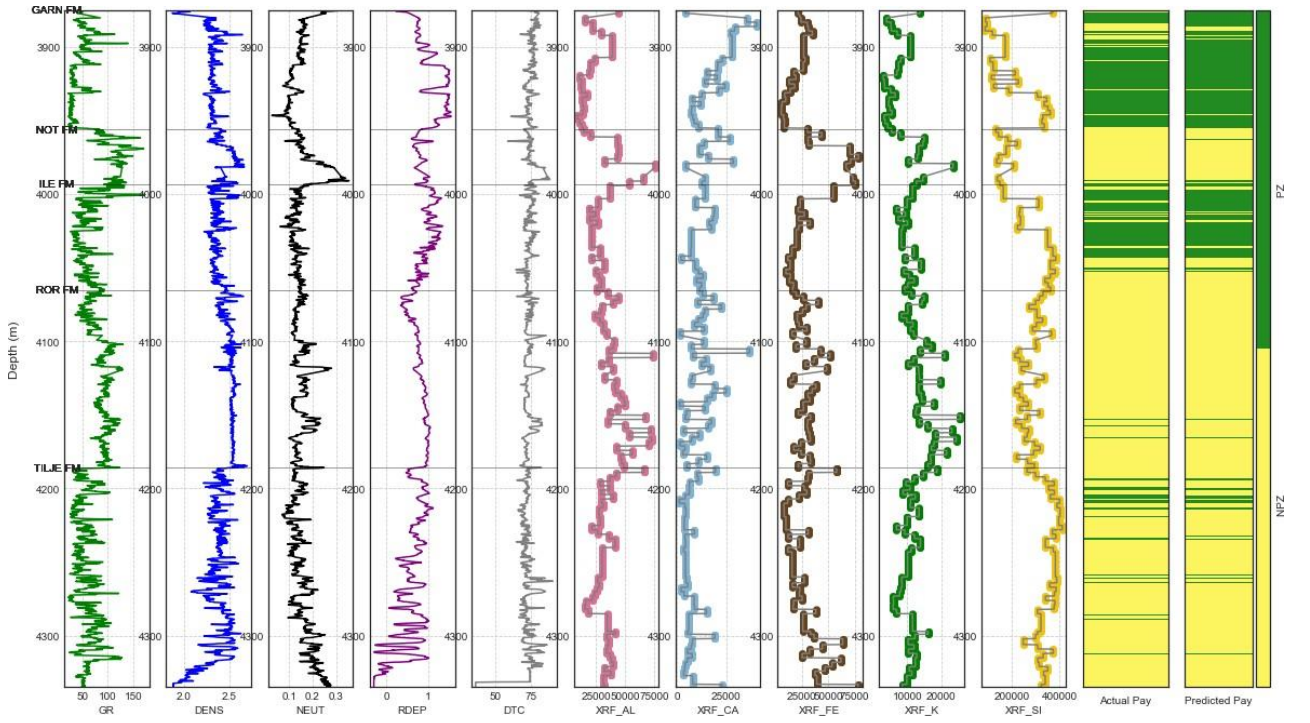


Figure 41:Model 4 performance after optimization in well 6506/12-8

4.2.7. XGBoost interpretability

Moving forward, it is also interesting to understand the theoretical approach to predicting the output of the ML model. For this, SHAP (Shapley Additive exPlanations) was used. In the summary plot (Figure 42), the collection of dots in the figure represents individual data points. The horizontal dispersion on the x-axis depicts the feature impact. The point coloration reflects whether the original feature value caused a higher (red) or lower (blue) prediction. Features' impact on the model is listed on the y-axis, depending on the rank order. The top one is the most contributor to the predictions, and the bottom is the least or zero-contributor.

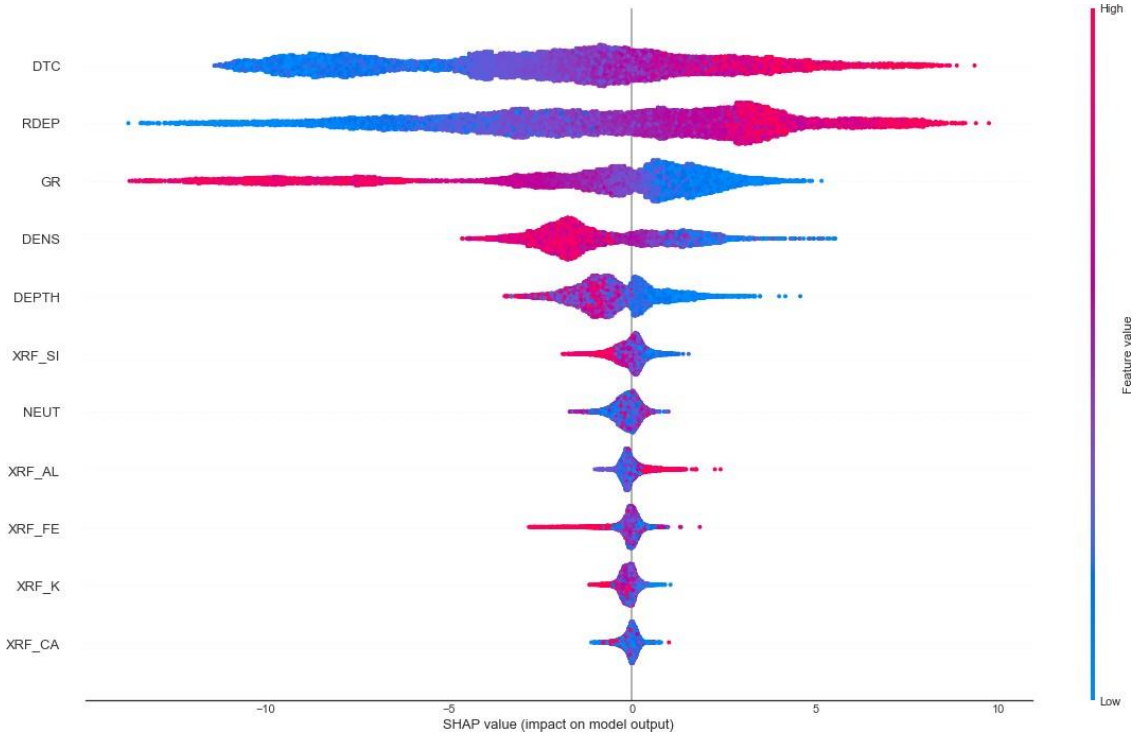


Figure 42:The SHAP summary plot illustrates the relationship between the features and the impact on the predicted class PZ.

From the summary plot, it can be noticed that the wireline logs represent the most important features of the model. At the same time, XRF chemical elements have the least importance, which is probably due to the vertical resolution. Of the eight features present, the highest contribution is associated with DTC, followed by RDEP, GR, and DENS, while the features with the lowest impact can be associated

with XRF Si, NEUT, XRF Al, XRF Fe, XRF K, and XRF Ca.

It is found that high DTC, RDEP, and low gamma-ray have a positive impact on the model, hence are, the characteristics of pay zones which are consistent with common knowledge. Depth values are correlated with their SHAP values (colors seem to advance in order from left to right). It is interesting to see that XRF Si is more important than NEUT log and has the highest importance among the XRF chemical elements included in the study. It is found that a high concentration of Silicon has a negative impact on the ML model, while low values of Silicon have a positive impact on the model or leads to pay zone identification. This could be related to the lateral and vertical variation of the sandstones in the reservoir, as seen previously. Moreover, silicate minerals are the largest and most important class of minerals and makeup approximately 90 percent of Earth's crust (Deer et al., 1992). Because of Silicon's high chemical affinity for oxygen, it becomes a major component of clay minerals classes such as Kaolinite Group, Montmorillonite/Smectite, Illite group, or Chlorite group. This might explain its behavior in the predictive model. In addition, high Al and low K values are associated with positive SHAP values and, therefore, the label. On the other hand, the addition of Iron reduces the chance of finding potential pay since it is negatively correlated with the label. This could be explained by the presence of iron minerals in reservoir rocks which play a negative role in the reservoir quality by decreasing the porosity and permeability. Besides, XRF Ca variable is not as dispersed in its SHAP values, but most values are not at zero (which implies no influence on the prediction).

4.2.8. DTC, RDEP, and GR model

After seeing the highest contributing features in the model, it was interesting to see how the model performs by only including the most impactful features (GR, RDEP, and DTC). The model's performance on training, validation, and test set for both wells can be seen in Figures 43 and 44. Surprisingly, compared to the test set performance of model 4, including only these logs is very

similar, with an F1 score of 0.93 in well 6506/12-8 and 0.95 in well 6506/12-6.

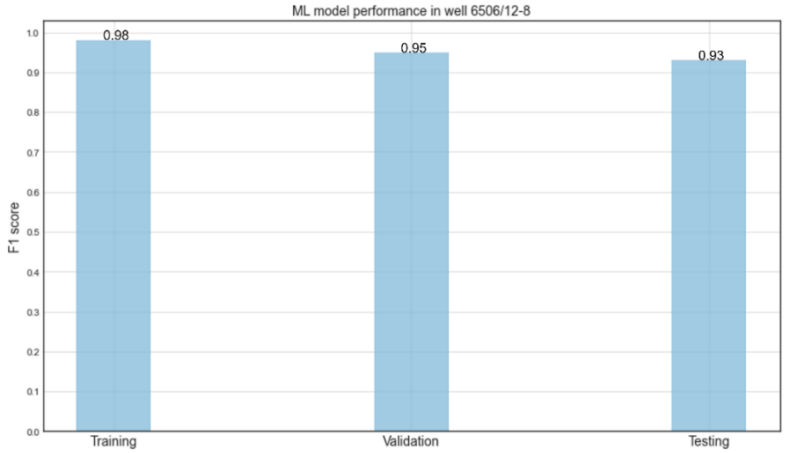


Figure 43:Performance evaluation in the training, validation, and testing set in well 6506/12-8.

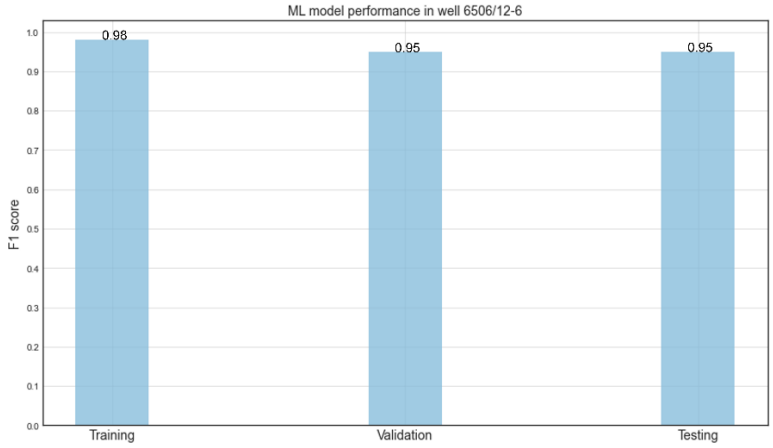


Figure 44:Performance evaluation in the training, validation, and testing set in well 6506/12-6.

Moreover, the performance on the training set is 2% lower than the performance of model 4, while performance on the validation set is also 2 % lower.

Chapter 5

5. DISCUSSION AND FUTURE WORK

According to the current study, log variables and XRF data were used to estimate pay zones that were not attempted previously. Although the analysis showed that the integration of XRF data could be helpful when identifying pay zones in the Smørbukk field, there are some non-negligible limitations in the presented solution and some interesting areas for further research. In the thesis, we used a supervised learning approach, and the discussed models were able to predict the pay zones in the Smørbukk field with high accuracy. In the first part of this chapter, we will discuss the validity of the results before we elaborate further on the main aim of the thesis and the importance of several factors in making predictions. Lastly, some interesting suggestions for further research are presented.

5.1. Discussion

5.1.1. Petrophysical Analysis

Based on the petrophysical calculations, the results indicate a good match with the core-derived porosity and water saturation. Choosing to calculate the petrophysical parameters using a combination of density and sonic led to the most optimal results. Density-derived porosity, in general, has a good vertical resolution, while using compressional sonic to calculate the porosity is much less sensitive to borehole conditions and often only moderately sensitive to hydrocarbon effects. The logs showed good reservoir properties, especially in the Garn Formation. The Garn and the Ile Formations were analyzed and found hydrocarbon-bearing. The reservoirs analyzed are composed of heterogenous sandstones, and integrating XRF data with wireline logs generated additional information to have a better analysis and understanding of the targeted reservoir section.

5.1.2. ML models

Starting with the data analysis stage, a critical factor was revealed in this thesis: data set imbalance (Figure 23).

This condition has been extensively studied, searching for an optimum technique to handle the class imbalance classification problems (Daskalaki et al., 2006; Gu et al., 2008; Fernandez et al., 2018). Compared with various evaluation metrics, the F1 score metric aims to find an equal balance between precision and recall. This supports the choice of the F1 score method as the evaluation metric for the ML phase.

Three ML algorithms were compared in this study in terms of CPU runtime and accuracy (F1 score). The XGboost model was the best performing model in this study and the fastest to validate, followed by Random Forest. Similarly, Logistic Regression was tested with moderate success. According to the feature importance correlation, well logs have the highest linear correlation with the label, while XRF chemical elements have the lowest correlation (Figure 32). In particular, XRF Ca has zero correlation. Its lowest impact, although its importance was found in the results but also confirmed by sensitivity analysis (Figure 42).

With a more detailed analysis of these predictive models collectively (Figure 45), it can be concluded that the performance of the models on the testing set increases with the addition of new features.

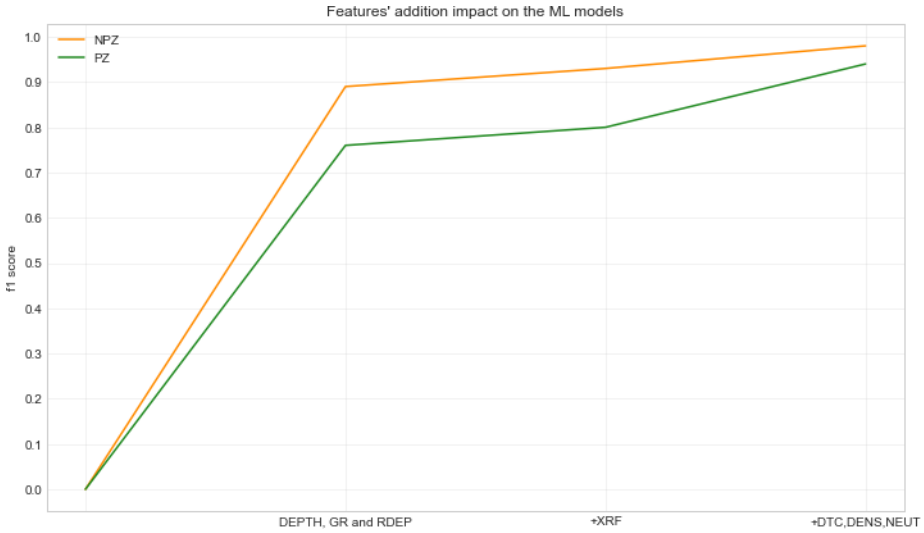


Figure 45: F1 score variation on ML models based on feature addition.

This could be explained by the Spearman correlation matrix between the features and the label present in the dataset (Figure 23). In model 3, DTC and DENS are included, which have a positive and negative correlation with the label (Pay) of 0.28 and -0.30, indicating that they are very important when making predictions. On the other hand, NEUT is another extra feature in model 3 which has a very low negative correlation (-0.08) with the label, indicating that NEUT is not a very important feature.

Model 1 and 2 faced more difficulties in accurately classifying the thin pays. The predictive model 4, with the addition of XRF chemical elements, leads to the highest performance in both NPZ and PZ classes. XRF improves the testing set by 5 and 6% when added to GR and RDEP while using all the available logs; the overall improvement is 1% and 4% in identifying the PZ class (Figure 46).

This high performance could be related to clay minerals' presence and contribution since the XRF Si, XRF K, and XRF Al have the highest importance among the XRF chemical elements, as shown in Figure 42. Moreover, XRF data contributed to a better representation of the NPZ class in both wells.

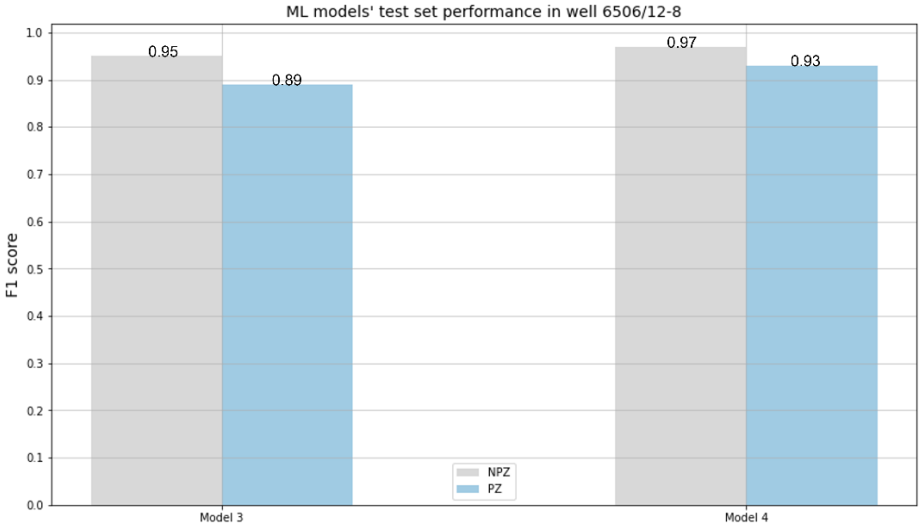


Figure 46:Test set performance comparison in well 6506/12-8 to highlight the clear contribution of XRF chemical elements.

It can be noticed that the vertical and lateral variation of the lithology has a strong correlation with the XRF chemical elements' impact. In particular, in parts of the reservoir with heterogenous sands, the impact of adding XRF chemical elements is higher as opposed to the reservoir intervals that contain clean or homogenous sands. Hence, there is a tendency for performance to improve when there is an increase in the shale content depending on the reservoir interval when integrating XRF data with well logs. This might be a possible explanation for the performance difference in model 4 for the wells included in the study. Another possible reason for the performance difference in model 4 could be attributed to the high vertical resolution of the XRF data for the wells included in the training and testing set. For instance, well 6506/12-6 has a higher vertical resolution (especially in the Ile Formation, which is 21m, and Tilje Formation, which is 18m, Figure 47) as opposed to the vertical resolution present in well 6506/12-8 which is lower (12m for both Ile and Tilje Formations).

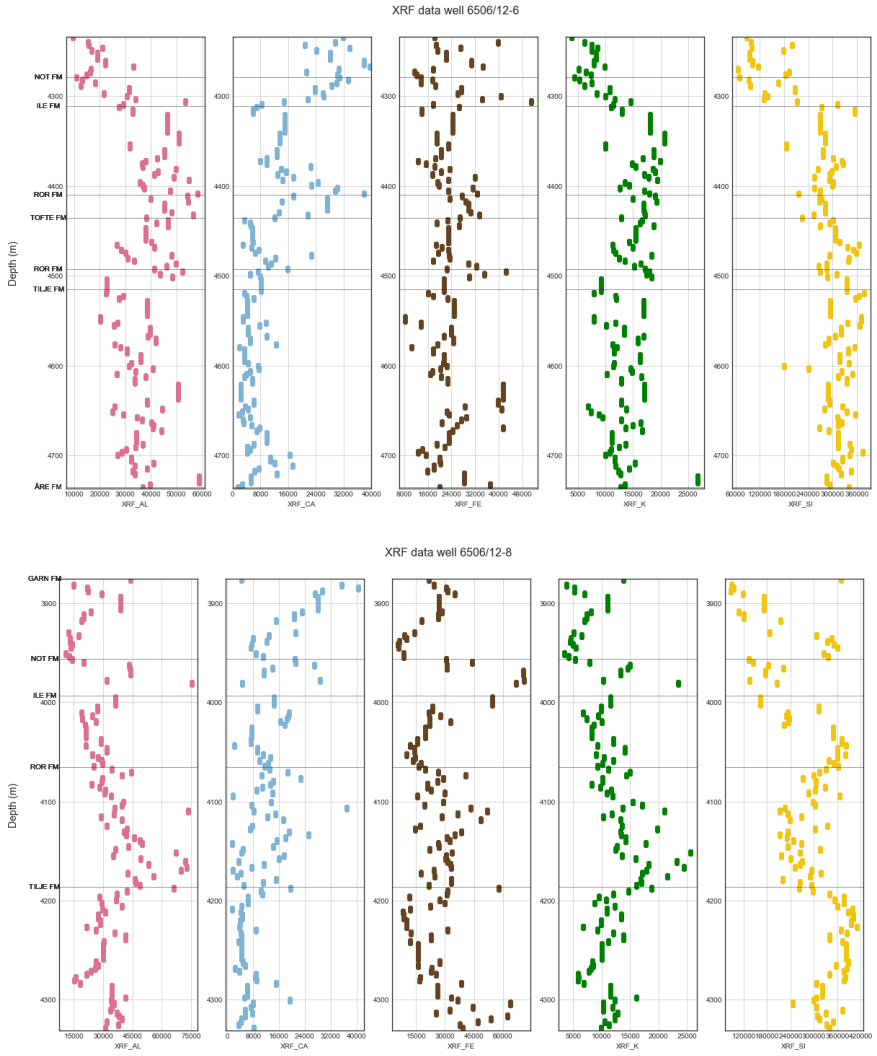


Figure 47:Vertical resolution for the XRF chemical elements. The top image corresponds to well 6506/12-6, while the bottom image corresponds to well 6506/12-8.

This makes the model difficult to capture the underlying data trends by introducing noise in the dataset, resulting in a higher chance of wrong predictions. It is well-known that the performance of ML models is highly influenced by the quality of the training data and has been extensively studied throughout the years (Sessions and Valtorta, 2006; Gudivada et al., 2017; Jain et al.,2020). Nothing helps ML projects more than improving the quality of the data they run on. In our case, the vertical resolution of the XRF data for each Formation varies for each well included in the study as a result of different sampling

rates for the cuttings.

This can affect the quality of the training data and lead to a model that is unable to capture the underlying data trends properly. However, there was still a 1% improvement in the PZ class for the well under consideration. Moreover, in Spearman's correlation, it can be seen that XRF chemical elements have a low correlation with the label pay (Figure 32). In particular, XRF Ca has zero correlation. Its lowest impact, although its importance was found in the results but also confirmed by sensitivity analysis (Figure 42).

Consequently, it might be unnecessary to include an additional correlating parameter as an input feature to predict pay zones. According to the statistical analysis of feature importance (Figure 42), the decreasing order of importance of input variables for predicting pay zones would be as follows: DTC, GR, RDEP, and bulk density. Including DEPTH also has a relatively high importance in the ML models, which could be attributed to learning from nearby lithology or considering geological setting when making predictions.

On the other hand, the GR, RDEP, and DTC models showed higher accuracy than model 1, model 2, and model 3 and comparable performance with model 4. A possible explanation for the model's high performance is that porosity was calculated using density porosity and sonic porosity depending on the difference between nominal (CALI) and measured hole size (BS). This might lead to inaccurate results and make the model biased, leading to an extremely high performance considering the input features. This means that combining DTC with only GR and RDEP can lead to biased results. To address this, it is recommended to include, in addition to GR and RDEP, only logs that have not been used to calculate porosity. As a result, the model will be less biased and more reliable.

Finally, the iterative grid search procedure to tune the hyperparameter values of model 4 was a critical step in enhancing the classification performance and control the ML model's behavior. Tuning the

model hyperparameter led to a significant performance improvement on the testing set (up to 4%) compared to using the default values.

5.2. Future work

Vertical resolution is the main challenge in incorporating XRF chemical elements with well logs. The vertical resolution varies depending on the reservoir presence from 3m, 10m, or sometimes 20m. This makes the model difficult to capture the underlying data trends, especially when the variation is high, leading to inaccurate or wrong predictions.

The following ideas can further expand the potential of this study:

1. Since the Well Released Initiative project, there have been almost 2000 wells with XRF data, creating a database with up to a 3m coring interval depending on the formations/reservoirs present would lead to a more accurate assessment of the overall impact of the XRF chemical elements. Thus, this will create a more robust and consistent ML model for pay zone identification.
2. Similarly, increasing the number of wells and extending the study across fields, in other words, creating a large-scale dataset, could potentially create a powerful tool for an upcoming exploration project.
3. Another interesting idea would be to create minerals from XRF chemical elements based on the chemical formula to better understand the targeted reservoir section. However, this will require more information about the standards of the XRF measurements.
4. Finally, testing different ML algorithms such as Artificial Neural Networks (ANN) as the size of the dataset increases.

Chapter 6

6. CONCLUSIONS

This research study aimed to create a robust and consistent ML model to automatically identify pay zones by integrating well logs with XRF chemical elements. This study proves that XRF chemical elements can be an additional tool to enhance the current ML models' performance. This will reduce decision-making time and interpretation bias during the initial phases of well evaluation and petrophysical analysis, allowing industry professionals to focus more on value-creating tasks that can reduce subsurface uncertainty.

The following conclusions can be drawn from this study:

1. Experimental core analysis reports (POR and SW) are essential for a good and reliable estimate of porosity and water saturation for pay zone determination.
2. Data quality is a prerequisite for achieving good results using ML experiments. Hence, data processing and analysis are critical steps in any ML problem. Outlier removal can reduce the dataset's noise, decreasing the chance of overfitting the training data.
3. Lateral and vertical lithology variation strongly correlates with the impact of XRF chemical elements. The higher contribution of the XRF data is associated with heterogeneous sands and increasing shale content, while in the case of clean or homogenous sands, the overall contribution is lower, especially when incorporating more logs.
4. For the pay zone identification of the Smørbukk field, the performance of the ML with XRF chemical elements is promising. It could potentially enhance/highlight mineralogy changes, resulting in better predictions. High DTC, low gamma-ray, and high resistivity are strong

indicators of pay zones, while DEPTH is another important feature.

5. Experiments using ensemble ML algorithms (RF and XGboost) have high performance (95 and 97%) in terms of F1 scores. Based on the performance (F1 score) and computational efficiency, XGboost outperformed RF and LR. XGboost can lead to balanced or unbalanced performance when adjusting the scale-pos-weight parameter.
6. The models tested with a limited number of logs have the highest error, while the performance increases significantly when adding new features, especially on the testing set.

References

Arce, N. D. C., & Thongsang, P. (2021). Pay Zone Determination by Applying Automatic Machine Learning (AutoML) in the Mckee Field, Taranaki Basin, New Zealand. *Bulletin of Earth Sciences of Thailand*, *13*(2), 32–41.

Awad, M., & Khanna, R. (2015). *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Springer Nature. https://doi.org/10.1007/978-1-4302-5990-9

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*(10), 281–305.

Blystad, P., Brekke, H., Færseth, R.B., Larsen, B.T., Skogseid, J., & Tørudbakken, B. (1995). *Structural elements of the Norwegian continental shelf. Part II: The Norwegian Sea region* (No. 8; 8, p. 44). Norwegian Petroleum Directorate.

Bradley, H. B. (1987). *Petroleum engineering handbook*. Society of Petroleum Engineers,Richardson, TX. https://www.osti.gov/biblio/5929149-petroleum-engineering-handbook

Brownlee, J. (2020, February 4). How to Configure XGBoost for Imbalanced Classification. *Machine Learning Mastery*. https://machinelearningmastery.com/xgboost-for-imbalanced-classification/

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Corfield, S., & Sharp, I. (2000). Structural style and stratigraphic architecture of fault propagation folding in extensional settings: A seismic example from the Smørbukk area, Halten Terrace, Mid-Norway. *Basin Research*, *12*, 329–341. https://doi.org/10.1111/j.1365-

2117.2000.00133.x

Dalland, A., Worsley, D., & Ofstad, K. (1988). *A lithostratigraphic scheme for the Mesozoic and Cenozoic and succession offshore mid- and northern Norway*. Norwegian Petroleum Directorate Bulletin, 4.

Daskalaki, S., Kopanas, I., & Avouris, N. (2006). Evaluation of Classifiers for an Uneven Class Distribution Problem. *Applied Artificial Intelligence*, *20*(5), 381–417. https://doi.org/10.1080/08839510500313653

Deer, W. A., Howie, R. A., & Zussman, J. (1992). *An introduction to the rock-forming minerals* (2nd ed). Longman Scientific & Technical ; Wiley.

Dinesh Yadav. (n.d.). *Weighted Logistic Regression for Imbalanced Dataset | by Dinesh Yadav | Towards Data Science*. Retrieved April 26, 2022, from https://towardsdatascience.com/weighted-logistic-regression-for-imbalanced-dataset-9a5cd88e68b

Doroudi, S. (2020). The Bias-Variance Tradeoff: How Data Science Can Inform Educational Debates. *AERA Open*, *6*(4), 2332858420977208. https://doi.org/10.1177/2332858420977208

Duval, A. (2019). *Explainable Artificial Intelligence (XAI)*. https://doi.org/10.13140/RG.2.2.24722.09929

Dwivedi, R., Singh, C., Yu, B., & Wainwright, M. J. (2021). Revisiting minimum description length complexity in overparameterized models. *ArXiv:2006.10189 [Cs, Math, Stat]*. https://doi.org/10.48550/arXiv.2006.10189

Egbele, E., Ezuka, I., & Onyekonwu, M. (2005, August 1). *Net-To-Gross Ratios: Implications in Integrated Reservoir Management Studies*. Nigeria Annual International Conference and Exhibition. https://doi.org/10.2118/98808-MS

Ehrenberg, S. N., Gjerstad, H. M., & Hadler-Jacobsen, F. (1992). *Smorbukk Field: A Gas*

Condensate Fault Trap in the Haltenbanken Province, Offshore Mid-Norway: Chapter 21. *14*, 323–348.

Elgeldawi, E., Sayed, A., Galal, A. R., & Zaki, A. M. (2021). Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis. *Informatics*, *8*(4), 79. https://doi.org/10.3390/informatics8040079

Færseth, R. (2020). Structural geology and basin development of the Norwegian Sea. *Norwegian Journal of Geology*, *100*. https://doi.org/10.17850/njg100-4-1

Fernández, A., García, S., Galar, M., Prati, R., Krawczyk, B., & Herrera, F. (2018). *Learning from Imbalanced Data Sets*. https://doi.org/10.1007/978-3-319-98074-4

Gafurove, D. O., Gafurove, O.M., & Kontorovich V.A. (n.d.). Interpretation of data from geophysical studies of theTalakan oil and gas condensate field by trained neural networks, forecast of the structure of the Osinsky horizon. *Seismic Exploration Technologies*, *4*, 85–92.

Galdi, P., & Tagliaferri, R. (2019). Data Mining: Accuracy and Error Measures for Classification and Prediction. In *Encyclopedia of Bioinformatics and Computational Biology* (pp. 431–436). Academic Press. https://doi.org/10.1016/B978-0-12-809633-8.20474-3

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, *4*(1), 1–58. https://doi.org/10.1162/neco.1992.4.1.1

Gu, Q., Cai, Z., Zhu, L., & Huang, B. (2008). *Data Mining on Imbalanced Data Sets*. 1020–1024. https://doi.org/10.1109/ICACTE.2008.26

Gudivada, V., Apon, A., & Ding, J. (2017). Data Quality Considerations for Big Data and Machine Learning: Going Beyond Data Cleaning and Transformations. *International Journal On Advances in Software*, *10*(1 and 2), 1–20.

Guo, D., Zhu, K., Wang, L., Li, J., & Xu, J. (2014). A new methodology for identification of potential pay zones from well logs: Intelligent system establishment and application in the

Eastern Junggar Basin, China. *Petroleum Science*, *11*(2), 258–264.

Hall, B. (2016). Facies classification using machine learning. *The Leading Edge*, *35*(10), 906–909. https://doi.org/10.1190/tle35100906.1

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Jain, A., Patel, H., Nagalapatti, L., Gupta, N., Mehta, S., Guttula, S., Mujumdar, S., Afzal, S., Sharma Mittal, R., & Munigala, V. (2020). Overview and Importance of Data Quality for Machine Learning Tasks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3561–3562. https://doi.org/10.1145/3394486.3406477

Jason Brownlee. (2016). *Machine Learning Mastery with Python: Understand Your Data, Create Accurate Models and Work Projects End-To-End* (v1.4). Machine Learning Mastery.

Koch, J.-O., & Heum, O. R. (1995). Exploration trends of the Halten Terrace. In *Norwegian Petroleum Society Special Publications* (Vol. 4, pp. 235–251). Elsevier. https://doi.org/10.1016/S0928-8937(06)80044-5

Kombrink, H. (2020, August 24). Uncovering new petroleum plays through analysing 600,000 cutting samples. *Expronews.Com*. https://expronews.com/technology/uncovering-new-petroleum-plays-through-analysing-600000-cutting-samples/

Liu, Q., & Wu, Y. (2012). Supervised Learning. In *Encyclopedia of the Sciences of Learning* (pp.

3243–3245). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_451

Martinius, A. W., Ringrose, P. S., Brostrøm, C., Elfenbein, C., Næss, A., & Ringås, J. E. (2005). Reservoir challenges of heterolithic tidal sandstone reservoirs in the Halten Terrace, mid-Norway. *Petroleum Geoscience*, *11*(1), 3–16. https://doi.org/10.1144/1354-079304-629

Masoudi, P., Arbab, B., & Mohammadrezaei, H. (2014). Net pay determination by artificial neural network: Case study on Iranian offshore oil fields. *Journal of Petroleum Science and Engineering*, *123*, 72–77. https://doi.org/10.1016/j.petrol.2014.07.007

Misra, S., & Li, H. (2020). Chapter 9—Noninvasive fracture characterization based on the classification of sonic wave travel times. In S. Misra, H. Li, & J. He (Eds.), *Machine Learning for Subsurface Characterization* (pp. 243–287). Gulf Professional Publishing. https://doi.org/10.1016/B978-0-12-817736-5.00009-0

Mohamed, I. M., Mohamed, S., Mazher, I., & Chester, P. (2019, September 23). *Formation Lithology Classification: Insights into Machine Learning Methods*. SPE Annual Technical Conference and Exhibition. https://doi.org/10.2118/196096-MS

Park, H. A. (2013). An introduction to logistic regression: From basic concepts to interpretation with particular attention to nursing domain. *Journal of Korean Academy of Nursing*, *43*(2), 154–164. https://doi.org/10.4040/jkan.2013.43.2.154

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830.

Raschka, S. (2020). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *ArXiv:1811.12808 [Cs, Stat]*. http://arxiv.org/abs/1811.12808

Rodolfo, S. B., Arteaga, D., Martin, C., & Rodriguez, F. (2010, December 1). *The Correct Shale-*

Volume Characterization Increases Hydrocarbon Reserves: Case Study of Cretaceous Formation, Lake Maracaibo, Venezuela. SPE Latin American and Caribbean Petroleum Engineering Conference. https://doi.org/10.2118/136811-MS

S. Fortmann-Roe. (n.d.). *Understanding the Bias-Variance Tradeoff*. Retrieved April 26, 2022, from https://scott.fortmann-roe.com/docs/BiasVariance.html

Sai Nikhilesh Kasturi. (n.d.). *Ensemble methods: Bagging & Boosting*. Retrieved April 26, 2022, from https://medium.com/swlh/difference-between-bagging-and-boosting-f996253acd22

Schlumberger Log Interpretation Charts. (1984). Schlumberger.

Sebastian Raschka. (2015). *Python Machine Learning*. Packt.

Sessions, V., & Valtorta, M. (2006). The Effects of Data Quality on Machine Learning Algorithms. *Proceedings of the 11th International Conference on Information Quality (ICIQ), Vol. 6 (Cambridge, MA: MIT)*, 485–498.

Stoddart, D., Alaei, B., Larsen, E., Oikonomou, D., & Naeini, E. Z. (2020). Missed pay prediction—The advent of a new era. *First Break*, *38*(10), 47–50. https://doi.org/10.3997/1365-2397.fb2020072

Supervised Machine learning—Javatpoint. (n.d.). Www.Javatpoint.Com. Retrieved May 2, 2022, from https://www.javatpoint.com/supervised-machine-learning

Tsanda, A., Bukharev, A., Budennyy, S., & Andrianova, A. (2018). *Well Logging Verification Using Machine Learning Algorithms*. 1–3. https://doi.org/10.1109/IC-AIAI.2018.8674435

Wang, W., Chakraborty, G., & Chakraborty, B. (2021). Predicting the Risk of Chronic Kidney Disease (CKD) Using Machine Learning Algorithm. *Applied Sciences*, *11*(1), 202. https://doi.org/10.3390/app11010202

Wang, Y., Pan, Z., Zheng, J., Qian, L., & Li, M. (2019). A hybrid ensemble method for pulsar candidate classification. *Astrophysics and Space Science*. https://doi.org/10.1007/s10509-

019-3602-4

Waskom, M. (2021). seaborn: Statistical data visualization. *The Journal of Open Source Software*, *6*, 3021. https://doi.org/10.21105/joss.03021

Wes McKinney. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, *445*. https://doi.org/10.25080/Majora-92bf1922-00a

Worthington, P. F. (2010). Net Pay — What Is It? What Does It Do? How Do We Quantify It? How Do We Use It? *SPE Reservoir Evaluation & Engineering*, *13*(05), 812–822. https://doi.org/10.2118/123561-PA

Worthington, P. F., & Cosentino, L. (2005). The Role of Cutoffs in Integrated Reservoir Studies. *SPE Reservoir Evaluation & Engineering*, *8*(04), 276–290. https://doi.org/10.2118/84387-PA

Appendix

Appendix 1. Data preprocessing

```python
def preprocess_data(path):

    """
    Description:
        Script used to preprocess the data including reading the las files and creating one dataframe
    Args:
        path: dataset location (str)

    Returns:
        p_data: preprocessed data (dataframe)
    """

    las_files = []
    path = path
    files = os.listdir(path)

    for file in files:
        if file.lower().endswith('.las'):
            las_files.append(path + file)

    #convert each las file to a dataframe
    df_list=[]
    for file in las_files:
        las = lasio.read(file)
        las_df = las.df()

        las_df['WELL'] = las.well.WELL.value
        las_df['DEPTH'] = las_df.index
        df_list.append(las_df)

    #concatenate the files
    df = pd.concat(df_list,sort=True)
    df.replace(-9999.00, np.nan, inplace=True)
    p_data = df[['DEPTH','WELL','X_COORDINATE','Y_COORDINATE','NPD_STRATIGRAPHY', 'CALI', 'BS','GR','DENS','NEUT','RDEP','DTC']]

    return p_data
```

Appendix 2. Petrophysical calculations

```
def petrophysical_calculations(data):

    """
    Description:
        Script used to calculate the petrophysical parameters for well 6506/12-8
    Args:
        data: well data (dataframe)

    Returns:
        well: final data with petrophysical parameters (dataframe)
    """

    well=data
    well['VSHALE'] = shale_volume(well['GR'], well['GR'].quantile(q=0.99),
                                  well['GR'].quantile(q=0.01))
    well['dif'] = well['CALI'] - well['BS']

    #bad hole flag
    well['bad_hole'] = np.where(((well['dif'] >= 0.02) | ((well['dif'] <= 0))),1,0)

    #total porosity
    well['PHIT'] = np.where(well['bad_hole'] == 1, dt_porosity(well['DTC'], 52, 205), density_porosity(well['DENS'], 2.67, 1))

    #effective porosity
    well['PHIE'] = well['PHIT']- (well['VSHALE']*(2.67-2.56)/(2.67-1))

    #formation water resistivity
    well['RW'] = 0.031

    #Calculate Archie SW
    well['SW'] = sw_archie(well['PHIE'], well['RDEP'], well['RW'], 1, 1.75, 2)

    #limit water saturation to 1
    well['SW'] = well['SW'].mask(well['SW']>1, 1)

    return well
```

Appendix 3. Petrophysical analysis plot

```
def analysis_plot(data, top_depth, bottom_depth):

    """
    Description:
        Plotting the petrophysical analysis results
    Args:
        data: well data (dataframe)
        top_depth: minimum depth (int)
        bottom_depth: maximum depth (int)

    Returns:
          plot: Different tracks with petrophysical analysis results, core data and XRF chemical elements.
    """
    well = data
    fig, ax = plt.subplots(nrows=1, ncols=7, figsize=(20,12), sharey=True)
    fig.subplots_adjust(top=0.75,wspace=0.1)
    fig.suptitle(f'{well.WELL.iloc[0]}', fontsize=22)
    fig.subplots_adjust(top=0.75,wspace=0.1)
    #General setting for all axis
    for axes in ax:
        axes.set_ylim(top_depth,bottom_depth)
        axes.invert_yaxis()
        axes.yaxis.grid(True, linestyle='--')
        axes.get_xaxis().set_visible(False)
    #Cali
    ax2=ax[0].twiny()
    ax2.set_xlim(6,36)
    ax2.set_ylabel('Depth (m)')
    ax2.plot(well["CALI"], well.index, color = "black",linestyle = '--', linewidth = 0.6)
    ax2.spines['top'].set_position(('outward',50))
    ax2.set_xlabel('CALI (in)',color='black')
    ax2.tick_params(axis='x', colors='black')
    ax2.spines["top"].set_edgecolor("black")
    #ax2.set_xticks([8, 10, 12, 14])
    ax2.set_ylabel("Depth (m)")
    ax2.grid(True, alpha=0.4, linestyle='--')
    ax[0].set_ylabel('Depth (m)')
```

```
    #Cali
    ax2=ax[0].twiny()
    ax2.set_xlim(6,36)
    ax2.set_ylabel('Depth (m)')
    ax2.plot(well["CALI"], well.index, color = "black",linestyle = '--', linewidth = 0.6)
    ax2.spines['top'].set_position(('outward',50))
    ax2.set_xlabel('CALI (in)',color='black')
    ax2.tick_params(axis='x', colors='black')
    ax2.spines["top"].set_edgecolor("black")
    #ax2.set_xticks([8, 10, 12, 14])
    ax2.set_ylabel("Depth (m)")
    ax2.grid(True, alpha=0.4, linestyle='--')
    ax[0].set_ylabel('Depth (m)')

    # Gamma Ray
    ax3=ax[0].twiny()
    ax3.set_xlim(0,150)
    ax3.plot(well["GR"], well.index, color = "green",linewidth = 0.6)
    ax3.spines['top'].set_position(('outward',10))
    ax3.set_xlabel('GR (API)',color='green')
    ax3.tick_params(axis='x', colors='green')
    ax3.spines["top"].set_edgecolor("green")
    ax3.set_xticks([0, 50, 100, 150])
    ax3.grid(True, alpha=0.4, linestyle='--')

    #fill GR log
    sand_GR_line=60
    ax3.fill_betweenx(well.index, sand_GR_line, well['GR'], where=(sand_GR_line>=well['GR']), color = 'gold', linewidth=0) # sand
    ax3.fill_betweenx(well.index, sand_GR_line, well['GR'], where=(sand_GR_line<well['GR']), color = 'lime', linewidth=0) # shale

    # Density
    ax4=ax[1].twiny()
    ax4.set_xlim(1.95, 2.45, 2.95)
    ax4.plot(well["DENS"], well.index, color = "red",linewidth = 0.6)
    ax4.spines['top'].set_position(('outward',50))
    ax4.set_xlabel('RHOB (g/cm3)',color='red')
    ax4.tick_params(axis='x', colors='red')
    ax4.set_xticks([1.95, 2.45, 2.95])
    ax4.spines["top"].set_edgecolor("red")
    ax4.grid(True, alpha=0.4, linestyle='--')

    # Neutron placed above density
    ax5=ax[1].twiny()
    ax5.set_xlim(-0.15,0.45)
    ax5.invert_xaxis()
    ax5.plot(well["NEUT"], well.index,color="blue",linewidth = 0.6)
    ax5.spines['top'].set_position(('outward',10))
    ax5.set_xlabel('NPHI (v/v)', color='blue')
    ax5.tick_params(axis='x', colors='blue')
    ax5.set_xticks([0.45, 0.15, -0.15])
    ax5.spines["top"].set_edgecolor("blue")

    # RDEP
    ax6=ax[2].twiny()
    ax6.set_xlim(0.1,100)
    ax6.set_xscale('log')
    ax6.spines['top'].set_position(('outward', 10))
    ax6.set_xlabel('RDEP (ohm.m)', color='purple')
    ax6.plot(np.log10(well["RDEP"]), well.index, color = "purple",linewidth = 0.6)
    ax6.tick_params(axis='x', colors='purple')
    ax6.spines["top"].set_edgecolor("purple")
    ax6.set_xticks([0.1, 1, 10, 100, 1000])
    ax6.grid(True,alpha=0.4, linestyle='--')
```

```
    # VSHALE
    ax7=ax[3].twiny()
    ax7.set_xlim(0, 3)
    ax7.spines['top'].set_position(('outward', 10))
    ax7.set_xlabel('VSHALE (v/v)', color='black')
    ax7.plot(well["VSHALE"], well.index, color = "black",linewidth = 0.6)
    ax7.tick_params(axis='x', colors='black')
    ax7.spines["top"].set_edgecolor("black")
    ax7.grid(True,alpha=0.4, linestyle='--')

    # Total porosity
    ax8 = ax[4].twiny()
    ax8.plot("PHIT", "DEPTH", data = well, color = "black",linewidth=0.6)
    ax8.set_xlabel("PHIT (%)")
    ax8.set_xlim(0.45, -0.15)
    ax8.xaxis.label.set_color("black")
    ax8.tick_params(axis='x', colors="black")
    ax8.spines["top"].set_edgecolor("black")
    ax8.set_xticks([0, 0.10, 0.20, 0.30, 0.5])
    ax8.grid(True,alpha=0.4, linestyle='--')

    #Core
    ax9 = ax[4].twiny()
    ax9.set_xlim(0.45, -0.15)
    ax9.scatter('CPOR', 'DEPTH' ,data = core, color = "blue",alpha=0.6)
    ax9.spines['top'].set_position(('outward',50))
    ax9.set_xlabel("CORE POR (%)")
    ax9.xaxis.label.set_color("blue")
    ax9.tick_params(axis='x', colors="blue")
    ax9.spines["top"].set_edgecolor("blue")
    ax9.set_xticks([0, 0.10, 0.20, 0.30, 0.5])
    ax9.grid(True,alpha=0.4, linestyle='--')

```

```
    # Core SW
    ax11 = ax[5].twiny()
    ax11.set_xlim(0, 1)
    ax11.scatter('CORE_SW', 'DEPTH' ,data = core, color = "red",alpha=0.6)
    ax11.spines['top'].set_position(('outward',50))
    ax11.set_xlabel("CORE SW (%)")
    ax11.xaxis.label.set_color("red")
    ax11.tick_params(axis='x', colors="red")
    ax11.spines["top"].set_edgecolor("red")
    ax11.set_xticks([0, 0.5, 1])
    ax11.grid(True,alpha=0.4, linestyle='--')

    ax12 = ax[6].twiny()
    #ax12 = ax.flatten()
    ax12.spines['top'].set_position(('outward',160))
    ax12.scatter('XRF_FE','DEPTH', data= df,color='#654321',label='XRF_FE',s=70,alpha=0.6)
    ax12.set_xlabel("XRF_FE (ppm)")
    ax12.xaxis.label.set_color("#654321")

    ax13 = ax[6].twiny()
    #ax12 = ax.flatten()
    ax13.spines['top'].set_position(('outward',120))
    ax13.scatter('XRF_K','DEPTH', data= df,color='green',label='XRF_K',s=70,alpha=0.6)
    ax13.set_xlabel("XRF_K (ppm)")
    ax13.xaxis.label.set_color("green")

    ax14 = ax[6].twiny()
    #ax12 = ax.flatten()
    ax14.spines['top'].set_position(('outward',80))
    ax14.scatter('XRF_CA','DEPTH', data= df,color='#7FB3D5',label='XRF_CA',s=70,alpha=0.6)
    ax14.set_xlabel("XRF_CA (ppm)")
    ax14.xaxis.label.set_color("#7FB3D5")
```

```
    ax15 = ax[6].twiny()
    #ax12 = ax.flatten()
    ax15.spines['top'].set_position(('outward',40))
    ax15.scatter('XRF_SI','DEPTH', data= df,color='#F1C40F',label='XRF_SI',s=70,alpha=0.6)
    ax15.set_xlabel("XRF_SI (ppm)")
    ax15.xaxis.label.set_color("#F1C40F")
    #ax15.xaxis.set_major_locator(ticker.MultipleLocator(30000))

    ax16 = ax[6].twiny()
    #ax12 = ax.flatten()
    ax16.scatter('XRF_AL','DEPTH', data= df,color='#DB7093',label='XRF_AL',s=70,alpha=0.6)
    ax16.set_xlabel('XRF_AL (ppm)')
    ax16.xaxis.label.set_color("#DB7093")


    #fill the neutron-porosity log
    x2p, _ = (ax4.transData + ax5.transData.inverted()).transform(np.c_[well["DENS"], well.index]).T
    ax5.autoscale(False)
    ax5.fill_betweenx(well.index, well['NEUT'], x2p, color="orange", alpha=0.4, where=(x2p > well['NEUT'])) # hydrocarbons
    ax5.fill_betweenx(well.index, well['NEUT'], x2p, color="blue", alpha=0.4, where=(x2p < well['NEUT'])) # water


    plt.tight_layout()
```

Appendix 4. Cutoff generation

```
def gen_cuttof(df):
    """

    Description:
        Function used to generate the pay zones for each well
    Args:
        data: df (dataframe)

    Returns:
            pay (1) and non-pay intervals (0)
    """

    if (df['SW']<0.5) & (df['VSHALE']<0.5) & (df['PHIT']>0.10):
        return 1
    else:
        return 0

df['Pay'] = df.apply(gen_cuttof, axis=1)
```

Appendix 5. Pay zones visualization

```
def pay_plot(logs, well_num, pay_colors):
    """

    Description:
        Logs and pay zones visualization
    Args:
        well_num: well number (int)
        pay_colors: pay colors (lst)

    Returns:
            plot: Different tracks with the pay zones flags.
    """

    wells = logs['WELL'].unique()
    logs = logs[logs['WELL'] == wells[well_num]]
    logs = logs.sort_values(by='DEPTH')
    cmap_facies = colors.ListedColormap(pay_colors[0:len(pay_colors)], 'indexed')
    top = logs.DEPTH.min()
    bot = logs.DEPTH.max()
    real_label = np.repeat(np.expand_dims(logs['Pay'].values, 1), 1, 1)
    f, ax = plt.subplots(nrows=1, ncols=5, figsize=(20,12))
    log_colors = ['brown', 'green', 'blue','green','orange','orange','green', 'blue','black','purple','purple','orange']

    f.subplots_adjust(top=0.85,wspace=0.2)
    for x in range(6,10):
        ax[x-6].plot(logs.iloc[:,x], logs.DEPTH, color=log_colors[x])
        ax[x-6].set_ylim(top, bot)
        #ax[i-7].set_xlim(logs.iloc[:,i].min(), logs.iloc[:,i].max())

        ax[x-6].set_xlabel(str(logs.columns[x]))
        ax[x-6].invert_yaxis()
        ax[x-6].grid(linestyle='--')
        ax[x-6].xaxis.set_major_locator(plt.MaxNLocator(5))
        ax=ax.flatten()
```

```
        for i in tops['DEPTH']:
            if ((i>=top) and (i<=bot)):
                ax[x-6].axhline(y=i, linewidth=0.5, color='black',alpha=0.8)
        for (i,j) in zip(tops['DEPTH'],tops['NPD_STRATIGRAPHY']):
            if ((i>=top) and (i<=bot)):
                ax[0].text(0.2, i ,j, horizontalalignment='center',verticalalignment='center')

    im = ax[-1].imshow(real_label, interpolation='none', aspect='auto', cmap=cmap_facies)
    ax[-1].set_xlabel('PAY')
    ax[-1].set_yticklabels([])
    ax[-1].set_xticklabels([])
    ax[0].set_ylabel('Depth (m)',fontsize=12,labelpad=35)


    divider = make_axes_locatable(ax[-1])
    cax = divider.append_axes("right", size="20%", pad=0.05)
    cbar=plt.colorbar(im, cax=cax)
    cbar.set_label((100*' ').join(['NPZ', 'PZ']))

    cbar.set_ticks(range(0,1)); cbar.set_ticklabels('')

    f.suptitle('Pay zones well '+str(wells[well_num]), fontsize=16,y=0.9)

```

Appendix 6. XRF data integration

```
wells = pd.read_csv('petrophysics.csv')
wells.dropna(inplace=True)
xrf = xrf.reset_index(drop=True)
wells = wells.reset_index(drop=True)
wells['DEPTH'] = np.round(wells['DEPTH'],1)
#round .9 depth values to neares integer
def depth_change(depth):
    if str(depth).split('.')[1] == '9':
        return(np.round(depth))
    else:
        return depth
wells['DEPTH']=wells.apply(lambda x: depth_change(x.loc['DEPTH']), axis=1)

xrf = xrf[['DEPTH','WELL',
       'XRF_AL',
       'XRF_CA',
       'XRF_FE',  'XRF_K', 'XRF_SI','XRF_U','XRF_TH']]
data  = pd.merge(wells,xrf, how='left', on=['DEPTH','WELL'])
fill_backward = data.fillna(method='bfill')
```

Appendix 7. ML model selection

```
from sklearn.model_selection import cross_val_score, KFold
from imblearn.pipeline import Pipeline
from time import time
from sklearn.preprocessing import StandardScaler

features_s = ['DEPTH', 'GR','DENS','NEUT','DTC','RDEP','XRF_AL', 'XRF_CA', 'XRF_FE', 'XRF_K', 'XRF_SI']
X = training_data[features_s]
y = training_data['Pay']

X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.3, random_state=101, shuffle=True)

classifier = [
        ("clf", LogisticRegression(max_iter=300, solver='lbfgs')),
        ('RF', RandomForestClassifier()),
        ('XGB', XGBClassifier(use_label_encoder=False))
        ]
split = 10
kf = StratifiedKFold(n_splits=split, shuffle=True,random_state=51)

imb = Pipeline([
  ('scaler', StandardScaler()),
  ("clf", LogisticRegression(max_iter=300, solver='lbfgs'))
])
results = []
i=1

for name, clf in classifier:
  t0 = time()
  target_names = ['NPZ', 'PZ']
  imb.set_params(clf = clf)
  cv_results = cross_val_score(imb, X_train, y_train, scoring='f1', cv=kf).mean()
  model_f = imb.fit(X_train, y_train)
  train_time = str(time() - t0)
  print(f"train time: {train_time}" )
  t0 = time()
  y_pred = model_f.predict(X_test)
  test_time = str(time() - t0)
  print(f"test time:  {test_time}")
  results.append(cv_results)
  print('---------------MODEL {} F1 score {}---------------'.format(i,results))

```

Appendix 8. XGBoost model

```
452  def xgb_model(train_set, val_set, test_set):
453
454      """
455      Description:
456          In this script XGBoost model is trained to predict the pay zones on validation and testing set.
457
458      Args:
459          train_set: training dataset (dataframe)
460          val_set: validation dataset (dataframe)
461          test_set: testing dataset (dataframe)
462
463      Returns:
464              train_pred_xgb: training set predictions (dataframe)
465              val_pred_xgb: validation set predictions (dataframe)
466              test_pred_xgb: test set predictions (dataframe)
467      """
468
469      from sklearn.model_selection import train_test_split
470      from xgboost import XGBClassifier
471
472      # selected features to be used while training
473      features_selected_xgb = ['DEPTH','GR','RDEP','DENS','NEUT','DTC','XRF_AL', 'XRF_CA', 'XRF_FE', 'XRF_K', 'XRF_SI']
474
475      x_train = train_set[features_selected_xgb]
476      y_train = train_set['Pay']
477
478      x_val = val_set[features_selected_xgb]
479      y_val = val_set['Pay']
480
481      x_test = test_set[features_selected_xgb]
482      y_test = test_set['Pay']
483      x_train_strat, X2, y_train_strat, Y2 = train_test_split(x_train,
484                                                              y_train,
485                                                              train_size=0.3,
486                                                              shuffle=True,
487                                                              stratify=y_train,
488                                                              random_state=0)
489
490          scale_pos_weight = y_hidden.value_counts()[0]/y_hidden.value_counts()[1]
491          model_xgb = XGBClassifier(tree_method='gpu_hist',
492                                    predictor='gpu_predictor',
493                                    scale_pos_weight=scale_pos_weight,
494                                    random_state=42,
495                                    )
496
497      # fitting the XGBoost model
498      model_xgb.fit(x_train_strat[features_selected_xgb], y_train_strat.values.ravel())
499
500      # predicting
501      train_pred_xgb = model_xgb.predict(x_train[features_selected_xgb])
502      val_pred_xgb = model_xgb.predict(x_val[features_selected_xgb])
503      test_pred_xgb = model_xgb.predict(x_test[features_selected_xgb])
504
505      return train_pred_xgb, val_pred_xgb, test_pred_xgb
```

Appendix 9. Hyperparameter tuning

```
from sklearn.model_selection import RandomizedSearchCV

params = {
 "learning_rate" : [0.01,0.02,0.05,0.1,0.2],
 "n_estimators":[100,300,500,700,900],
 "subsample": [0.5,0.6,0.7,0.8,0.9,1],
 "colsample_bytree": [0.5, 0.6, 0.7, 0.8,0.9,1],
 "max_depth" : [3, 4, 5, 6, 7, 8,9,10],
 "min_child_weight":[ 1, 3, 5, 7 ]
}

xg_model=RandomizedSearchCV(model_xgb,param_distributions=params,n_iter=5,scoring='f1',n_jobs=-1,cv=10,verbose=3)

#model fitting
xg_model.fit(x_train,y_train)

#best estimator
xg_model.best_estimator_
```