Development of Machine Learning (ML) based application for the Detection of Geological Facies through analysis of Well-Log data

Thesis Report

Integrated Master of Technology

in

Applied Geophysics

by

Dhyey Dabhi

19JE0269

under the guidance of

Prof. Saurabh Dutta Gupta

DEPARTMENT OF APPLIED GEOPHYSICS

INDIAN INSTITUTE OF TECHNOLOGY

(INDIAN SCHOOL OF MINES)

DHANBAD

(INDIAN SCHOOL OF MINES)

DHANBAD

CONTENTS

CERTIFICATE

This is to certify that **Dabhi Dhyey Pranavbhai,** Admission No. **19JE0269**, has completed his dissertation work entitled "**DEVELOPMENT OF MACHINE LEARNING BASED APPLICATION FOR THE DETECTION OF GEOLOGICAL FACIES THROUGH ANALYSIS OF WELL-LOG DATA**" in partial fulfillment of the requirement for the award of Master of Technology in Applied Geophysics under the guidance of **Prof. Saurabh Dutta Gupta** at Indian Institute of Technology (Indian School of Mines), Dhanbad. This work has not been submitted elsewhere for any other degree or distinction.

SUPERVISED BY:

Prof. Saurabh Dutta Gupta

Associate Professor

Department of Applied Geophysics

Indian Institute of Technology

(Indian School of Mines), Dhanbad

FORWARDED BY:

Prof. S.K. Pal

Head of the Department

Department of Applied Geophysics

Indian Institute of Technology

(Indian School of Mines), Dhanbad

CLASSIFIED DATA

This is to certify that the thesis entitled "**Development Of Machine Learning Based Application For The Detection Of Geological Facies Through Analysis Of Well-Log Data**" being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad by Mr. Dabhi Dhyey Pranavbhai, Admission no. 19JE0269 for the award of Integrated M.Tech. degree in the Department of Applied Geophysics does not contains any classified information. This work is original and has yet not been submitted to any institution or university for the award of any degree.

Dabhi Dhyey Pranavbhai
19JE0269
Department of Applied Geophysics
Indian Institute of Technology
(Indian School of Mines), Dhanbad

DECLARATION

The Dissertation titled "**Development Of Machine Learning Based Application For The Detection Of Geological Facies Through Analysis Of Well-Log Data**" is a presentation of my original research work and is not copied or reproduced, or imitated from any other person's published or unpublished work. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgment of collaborative research and discussions, as may be applicable. Every effort is made to cite the published/unpublished work of others properly if it is referred to in the Dissertation. To eliminate the scope of academic misconduct and plagiarism, I declare that I have read and understood the UGC (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018. These Regulations were notified in the Official Gazette of India on 31st July 2018. I confirm that this Dissertation has been checked with the online plagiarism detector tool Turnitin (http:///www.turnitin.com) provided by IIT (ISM) Dhanbad and a copy of the summary report/report, showing similarities in content and its potential source (if any), generated online through Turnitin is enclosed at the end of the Dissertation. I hereby declare that the Dissertation shows less than 10% similarity as per the report generated by Turnitin and meets the standards as per MHRD/UGC Regulations and rules of the Institute regarding plagiarism. I further state that no part of the Dissertation and its data will be published without the consent of my guide. I also confirm that this Dissertation work, carried out under the guidance of Prof. Arun Singh, Assistant Professor, Department of Applied Geophysics, has not been previously submitted for assessment for the purpose of the award of a Degree either at IIT (ISM) Dhanbad or elsewhere to the best of my knowledge and belief

Dabhi Dhyey Pranavbhai

19JE0269

Department of Applied Geophysics

Indian Institute of Technology

(Indian School of Mines), Dhanbad

ACKNOWLEDGEMENT

I would like to express my special gratitude to my guide **Prof. Saurabh Dutta Gupta**, Associate Professor, IIT (ISM), Dhanbad, who has helped me successfully complete this thesis. With his knowledge and expertise in this field, he has guided me patiently and provided me with the necessary means and requirements to make this project flow swiftly.

I would also like to thank my parents and friends for all the support and encouragement during this whole semester, which helped me successfully complete this thesis unit. Lastly, I would like to thank the HOD of my department, **Prof. S.K. Pal**, for giving me the opportunity to do this thesis unit and also my batchmates, who have been patient with me and helped me during this dissertation work.

ABSTRACT

This research project focuses on the development of a robust data pipeline tailored for data cleansing and preparation. The pipeline encompasses several crucial steps, such as identifying patterns in missing data, imputing missing values in accordance with these patterns, performing feature engineering, and applying data scaling. An extensive evaluation of various interpolation techniques for handling missing data was conducted, with the selection of the most effective technique being guided by their performance metrics.

Following the data preprocessing phase, machine learning models, specifically the Support Vector Classifier, Random Forest Classifier, and XGBoost, were employed to analyze the data. The outcomes generated by these diverse models were systematically compared, and the model with the best performance metrics was ultimately chosen. This study demonstrates the importance of a comprehensive data pipeline and the selection of the optimal machine learning model for obtaining meaningful insights from the data.

Next semester, the research would be more focused on exploring Deep Learning techniques and 3D interpolation techniques for interpolating the facies existing between the well logs.

CHAPTER 1: INTRODUCTION

1.1 Introduction

Geological facies identification is a vital component in our quest to understand subsurface formations, playing a pivotal role in applications such as resource exploration, environmental management, and hazard assessment. Well log data, which originates from drilling operations, is a rich source of information for geological investigations. However, the utilization of well log data presents a unique set of challenges and limitations.

A primary challenge in dealing with well log data is the prevalence of missing values and erroneous data. These issues stem from various sources, such as sensor malfunctions, data transmission errors, and incomplete data collection. Such imperfections can significantly compromise the accuracy and reliability of geological facies identification models, leading to potentially misleading interpretations and undermining the integrity of geological studies.

To address these challenges, this thesis report focuses on the development of a specialized data pipeline designed for well log data. It places a strong emphasis on comprehensive data cleaning and preprocessing techniques. The pipeline includes strategies for identifying patterns in missing data, imputing missing values based on these patterns, and enhancing data quality through feature engineering and scaling. Additionally, the evaluation of interpolation techniques aims to rectify missing data issues and elevate the overall quality of well log datasets, facilitating more precise geological facies identification.

This study also delves into the application of machine learning models on the preprocessed well log data. Through systematic comparisons of these models, we aim to identify the most effective approach for geological facies identification.

1.2 Research Objectives

The following are the main objectives of the study

- Create a robust well log data preprocessing pipeline to enhance data quality and reliability.
- Systematically evaluate interpolation methods to rectify missing data in well logs, improving data integrity.
- Develop and Optimize machine learning models to achieve precise geological facies identification, selecting the most effective approach.

1.3 Framework of Thesis

In Chapter 1, we present the introduction and objectives of the current study, along with a fundamental understanding of the framework guiding this thesis.

Chapter 2 is dedicated to conducting a literature survey on previous work in the field, in addition to exploring the theories and technologies applied in this research.

Chapter 3, we thoroughly explain the methodology employed for advancing this thesis. This chapter is further divided into sub-chapters covering aspects like the identification and management of missing data, data preprocessing methods, and the application of machine learning models for facies identification.

Chapter 4 contains the results of the ML models as well as visual performance metrics used for model evaluation.

Chapter 5 contains a basic Summary and Conclusion of the work and present study that had been done until now.

CHAPTER 2: THEORY & LITERATURE SURVEY

Well log data and it's importance in Geological Facies Identification

Well log data are subsurface measurements obtained during drilling operations and are invaluable for geological facies identification. The theory behind well log data emphasizes the role of these measurements in understanding subsurface formations. Well logs record various geophysical properties such as gamma ray, resistivity, and porosity, offering insights into lithological variations and geological structures.

Well log data's importance for facies identification lies in its ability to characterize different rock types, sedimentary environments, and geological features. By analyzing these data, geoscientists can distinguish between facies, which represent distinct lithological or depositional units. This understanding is essential for various applications, including hydrocarbon exploration, reservoir characterization, and environmental assessments.

Facies identification using well log data involves the analysis of log signatures and patterns. These logs provide critical information for constructing facies models, mapping reservoir properties, and making informed decisions in drilling and resource management. Accurate facies identification aids in optimizing well placement, reservoir modeling, and production forecasting, contributing to efficient subsurface exploration and resource utilization

Missing Values in Well Log Data

Missing values in well log data refer to data points that are absent within the dataset, impacting the completeness and quality of geological information. These missing values occur due to various reasons:

Discrete Missing Values: These are specific, isolated data points that are missing. They often result from sensor malfunctions, human errors during data collection, or quality control measures where certain data points are intentionally excluded. The aim is to remove unreliable measurements to maintain data quality.

Continuous Missing Values: Continuous missing values represent sections within the well log data where data is absent. This may occur due to drilling interruptions, sensor failures, or geological anomalies where certain measurements cannot be taken. These gaps disrupt the continuity of data collection along the borehole.

Understanding these types of missing values is essential for accurate geological interpretation, as the reasons behind their occurrence influence the choice of imputation or interpolation methods to address them effectively.

Cubic Interpolation: Cubic interpolation, a cornerstone of this endeavor, operates on the principle of estimating values between discrete data points through cubic polynomials. This method plays a crucial role in addressing the presence of missing values, identified through the 'missingno' library visualization, by seamlessly constructing a continuous curve throughout the well-log dataset. By implementing cubic interpolation, the project aims to create a more refined and continuous representation of geological features. This, in turn, contributes to elevating the overall completeness and precision of the geological facies identification model. Ultimately, the application of cubic interpolation enhances the capacity to discern intricate subsurface formations, facilitating more insightful decision-making in the realms of resource exploration and environmental assessments.

Support Vector Classifier

The Support Vector Classifier (SVC) is a powerful machine learning algorithm that aims to find an optimal hyperplane within a high-dimensional feature space, which best separates data points into distinct classes. Its theory is rooted in the concept of maximizing the margin between classes while minimizing the classification error. SVC achieves this by identifying support vectors, which are the data points closest to the hyperplane and crucial for defining the margin.

The primary goal of SVC is to find a decision boundary that not only separates the classes but is also resilient to new, unseen data. It uses a kernel trick to transform the data into a higher-dimensional space, allowing it to handle nonlinear relationships. By optimizing a margin-based objective function, SVC provides robust classification and is particularly effective in scenarios with complex decision boundaries, making it a valuable tool in various fields, including pattern recognition, image classification, and anomaly detection.

Random Forest Classifier

The Random Forest Classifier is a machine learning algorithm based on an ensemble of decision trees. Its theoretical foundation centers on the wisdom of crowds and the power of aggregating multiple decision trees for robust classification.

Random Forest operates by constructing numerous decision trees with random subsets of the training data and features, then combining their predictions through a majority voting mechanism (classification) or averaging (regression). This ensemble approach mitigates overfitting and enhances generalization.

The algorithm's success stems from its ability to capture complex relationships in data while reducing variance and increasing accuracy. It is resilient to noisy data and missing values, making it a versatile tool for various classification tasks, including image recognition, medical diagnosis, and geological facies identification. Furthermore, Random Forest provides feature importance rankings, aiding in feature selection and interpretation, making it valuable for both predictive accuracy and model explainability.

XG-Boost Algorithm

XGBoost (Extreme Gradient Boosting) is a state-of-the-art machine learning algorithm designed to enhance gradient boosting techniques. Its theoretical foundation centers on the concept of boosting, a method that combines multiple weak learners (typically decision trees) to create a robust, highly accurate model.

XGBoost operates by iteratively training new decision trees that correct the errors made by the previous ones. It focuses on optimizing a loss function, effectively minimizing prediction errors. The algorithm introduces regularization terms to prevent overfitting, and it handles missing values efficiently.

XGBoost's distinguishing features include a gradient-based optimization method, parallel processing, and the ability to handle both classification and regression tasks. It has gained prominence in various applications such as Kaggle competitions, fraud detection, and recommendation systems due to its impressive predictive power, speed, and flexibility. Its theoretical underpinnings make it a valuable tool for tackling complex, real-world problems with high-dimensional data.

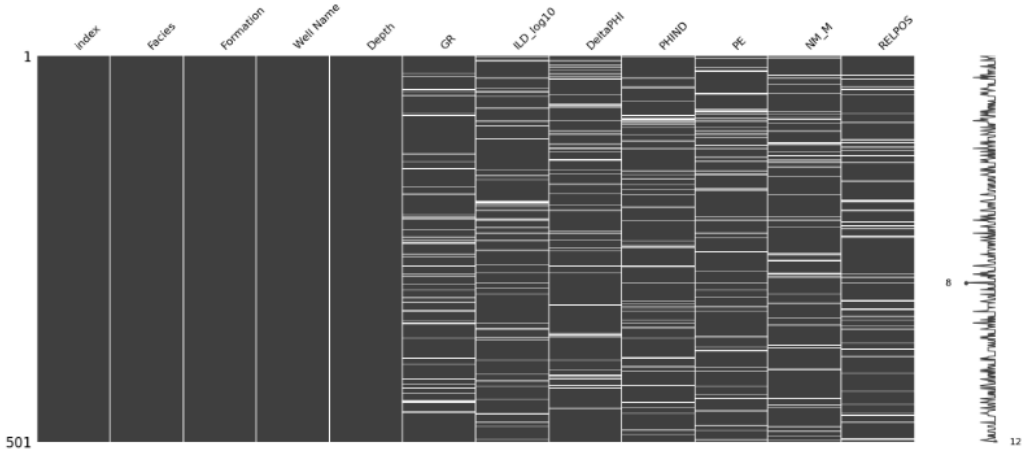
CHAPTER 3: METHODOLOGY

3.1 Identification and Dealing with Missing Values:

The first step in our methodology involves addressing missing values within the well log dataset. Identifying these missing values is crucial as it allows us to understand the extent of data gaps and their nature. Majorly, we focus on identifying if the missing values are discrete or continuous in nature. The missing values can be visualized using the *'missingno'* library available in python programming language. On creating missing value matrices and visualizing it, we can identify the nature of missing values.

Discrete missing values are isolated points of missing data within the dataset, and they can be addressed by interpolating the data to fill these gaps effectively. On the other hand, continuous missing values represent sections where data is absent. For these, we rely on well log correlations to impute the missing data accurately. Addressing missing values is vital as it ensures data completeness and reliability, which are essential for robust geological facies identification.

On visualizing the missing value matrix for the current dataset using *'missingno'* library, the following result was obtained:



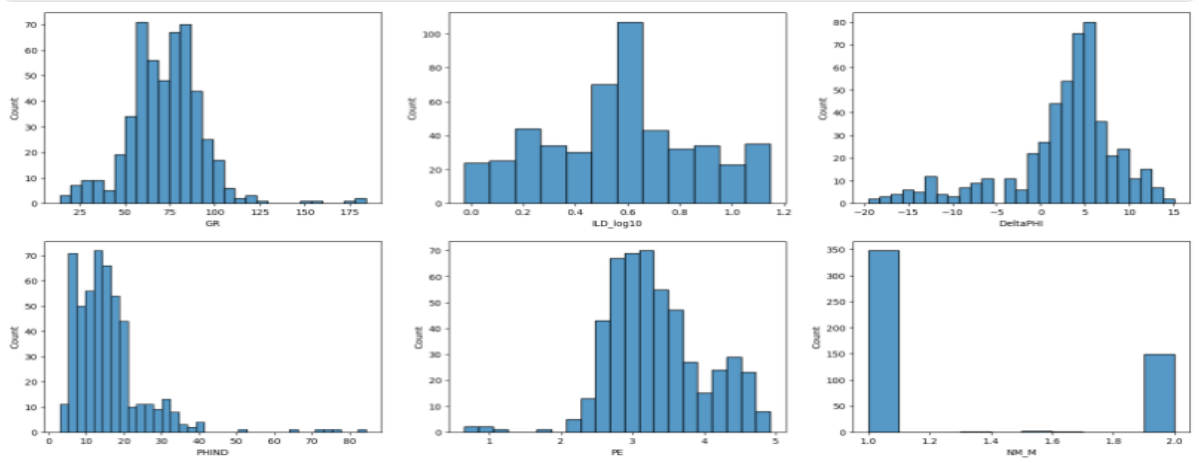
The gray patches in the image represent non-null values while the white stripes in the image represent missing values in the dataset. Hence, it is clear from the plot the missing values are discrete in nature and hence using 1D interpolation techniques would be beneficial in this case.

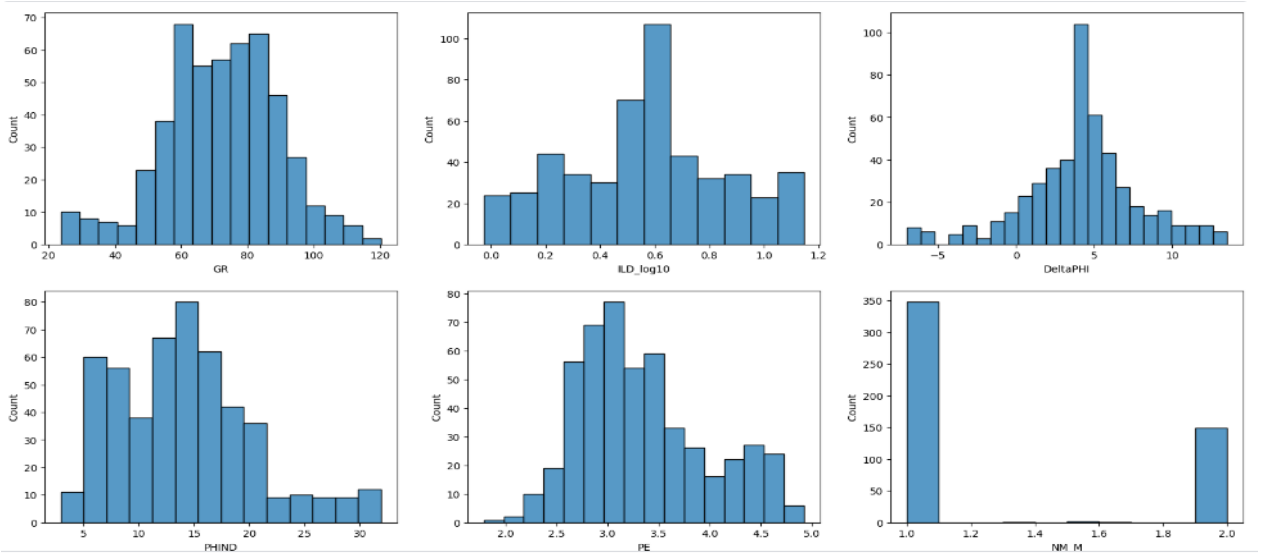
Different 1D interpolation techniques were implemented including ‘linear’, ‘quadratic’ and ‘cubic’ interpolation, out of which cubic interpolation outperformed the other two.

3.2 Data Preprocessing:

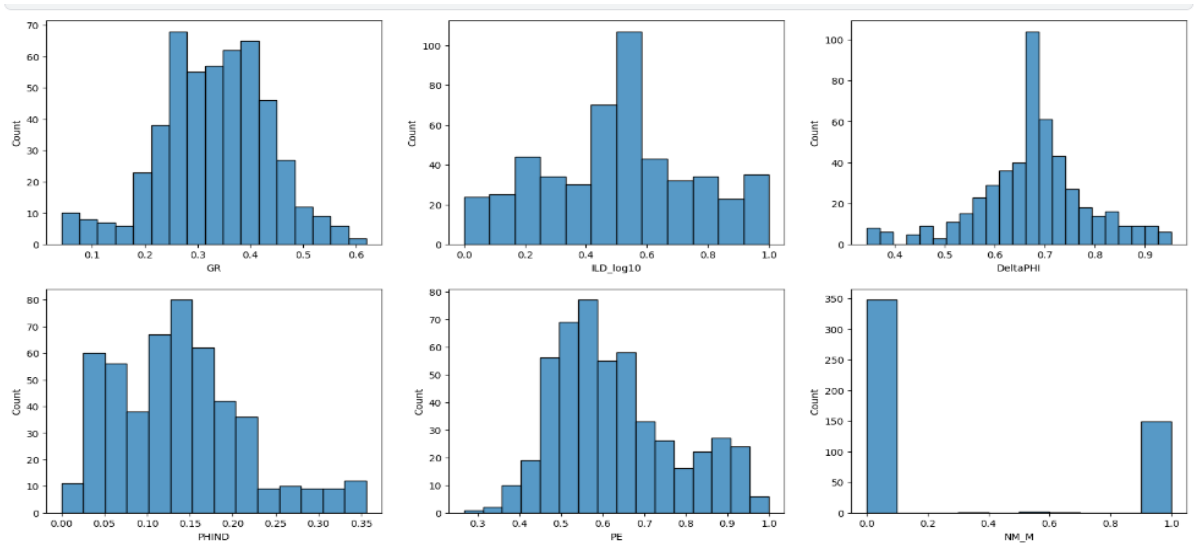
After handling missing values, we proceed to data preprocessing. This phase is essential for enhancing the quality and usability of the dataset. Three critical aspects are considered:

a. Outlier Removal : To remove outliers, the Interquartile Range (IQR) method is applied to each column in the 'logs' dataset. Values outside the range defined by 1.5 times the IQR above the third quartile (Q3) or below the first quartile (Q1) are considered outliers. Identified outliers are replaced with the median of the respective column. This methodology ensures a robust approach to outlier removal, promoting data integrity and mitigating the impact of extreme values on subsequent analyses. Histograms for all the well logs before and after the application of outlier removal process is shown below



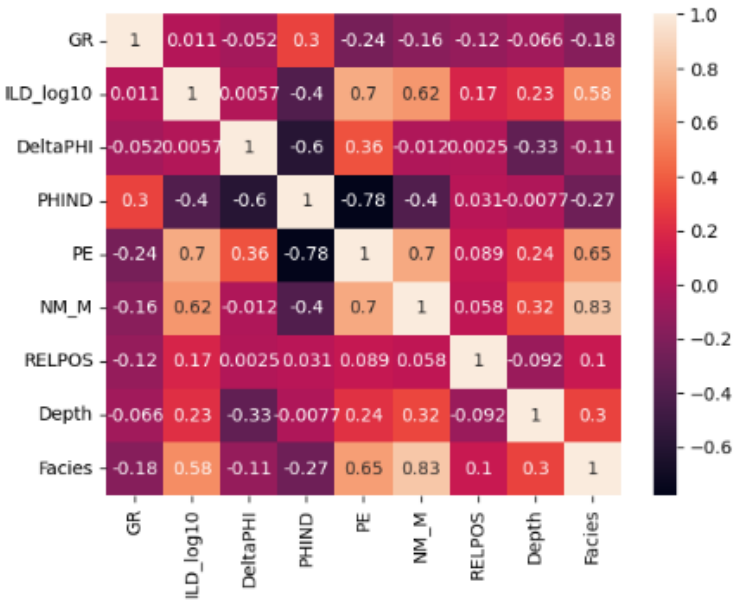


b. Data Scaling: Proper data scaling ensures that all features are on a consistent scale, preventing certain variables from having undue influence during the modeling process. In this step, we apply techniques like Min-Max scaling and Standardization to standardize feature scales, ensuring a balanced and well-behaved dataset. This scaling is particularly crucial for models that rely on distance measures or gradient-based optimization.



c. Feature Selection: Feature selection plays a significant role in improving model performance and reducing computational complexity. Redundant or irrelevant features can hinder the model's effectiveness and lead to increased computational overhead. Using feature

selection methods, we identify and retain the most informative attributes while eliminating those that do not significantly contribute to the task at hand. This streamlined dataset accelerates model training and simplifies result interpretation, making it easier to extract meaningful insights from the data.



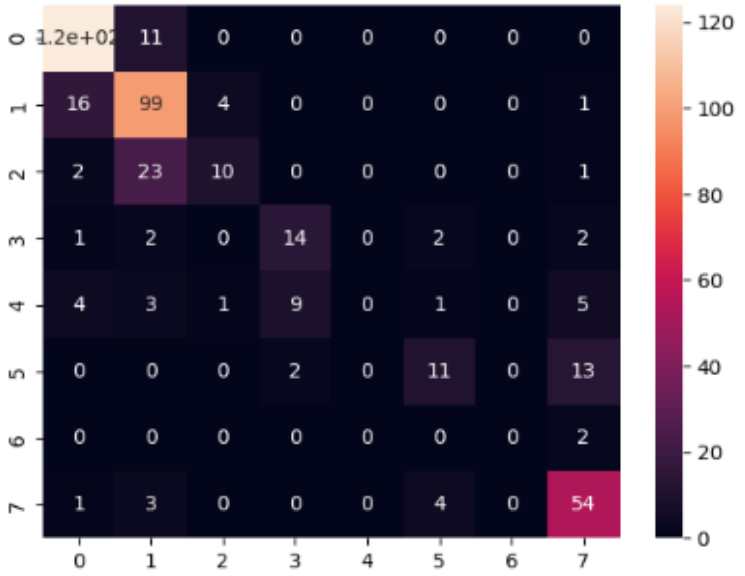
3.3 Developing ML Models:

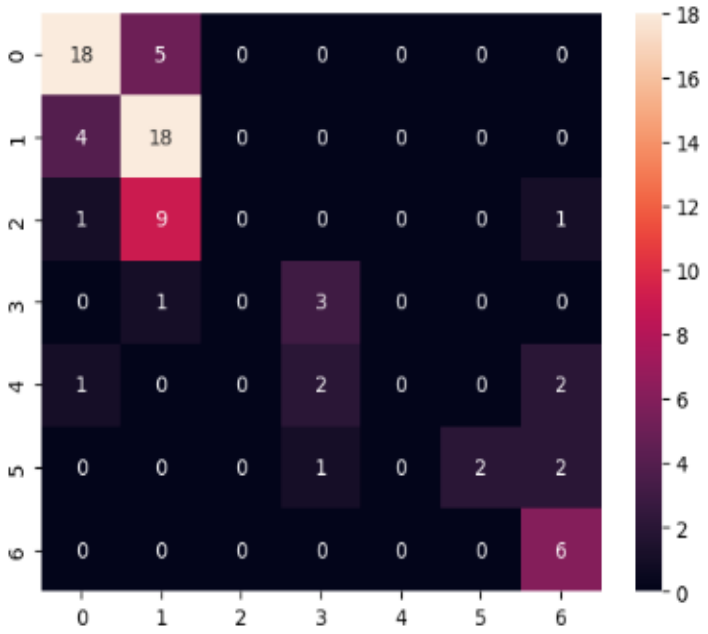
In the final phase, we focus on building machine learning models for geological facies identification. We employ three distinct models: Support Vector Classifier (SVC), Random Forest, and XG Boost. Each of these models brings its unique strengths to the task, and by using a variety of models, we aim to assess their performance in identifying geological facies. The dataset, which has been well-preprocessed in the earlier phases, serves as the input for these models. We will evaluate the effectiveness of each model by analyzing their results and comparing their performance metrics. This step is crucial for selecting the best model or combination of models that can accurately and reliably identify geological facies from well log data, contributing to more informed decision-making in subsurface exploration and geological studies.

CHAPTER 4: RESULT & DISCUSSION

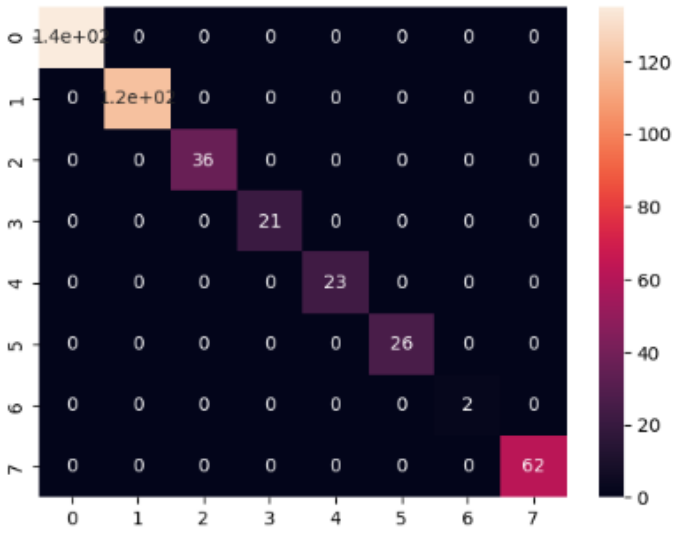
This section outlines the performance metrics derived from the machine learning models discussed earlier. The training data had dimensions of (425,8) for input and (425,1) for output. Model evaluation took place on the validation dataset, featuring an input shape of (76,8) and an output shape of (76,1). Given the dataset's class distribution non-uniformity, we employed performance metrics like the confusion matrix and F1 score. These metrics offer a comprehensive assessment of the models' effectiveness in handling varying class frequencies, providing valuable insights into their overall performance and suitability for the given task.

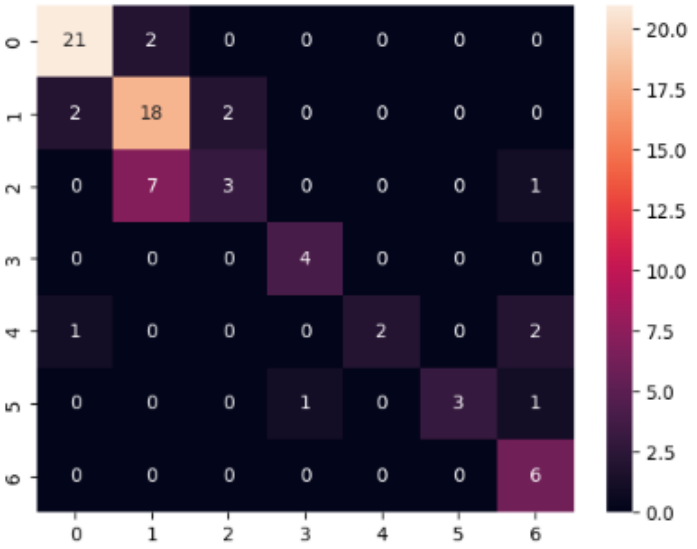
Support Vector Classifier (SVC) - The F1 scores for the training and validation data sets are 0.734 and 0.618, respectively. These results strongly indicate that the model is underfitting the provided dataset, implying that augmenting the model's complexity would likely enhance predictive performance. The ensuing sections present the calculated confusion matrices for both the training and validation datasets, offering a detailed insight into the model's performance on these respective sets.



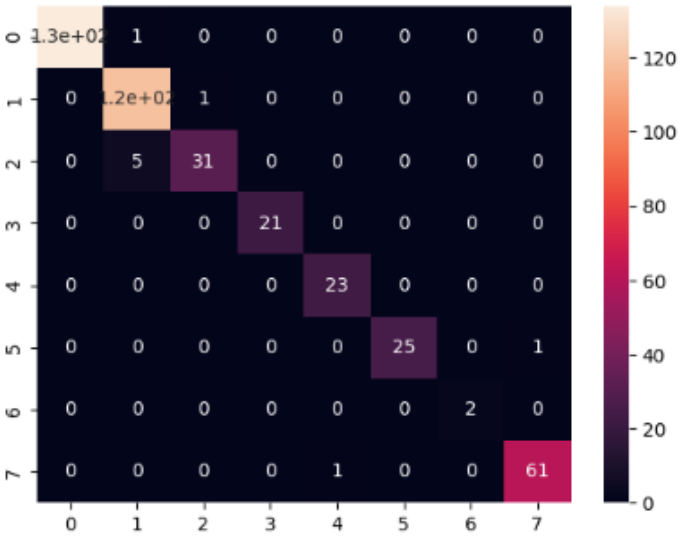


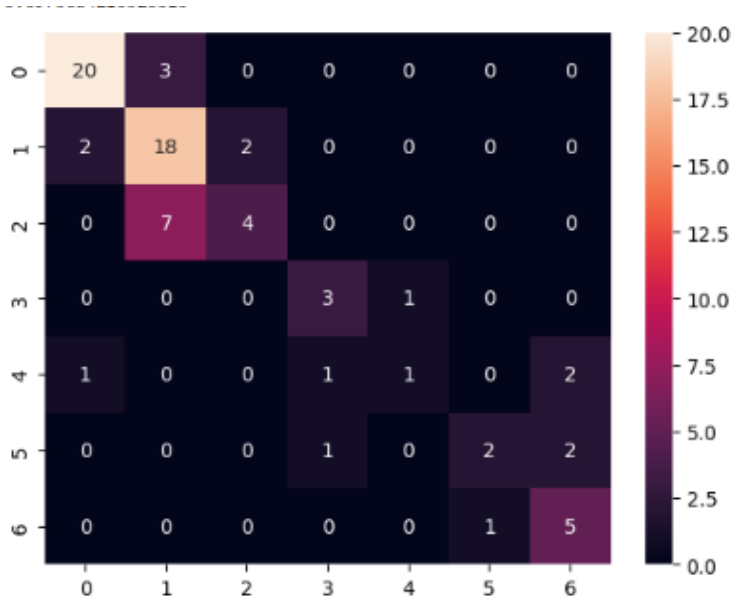
Random Forest Classifier - The F1 score for training data and validation data is 0.98 & 0.75 respectively. Hence, it is clearly visible that the model is overfitting on the given dataset. Providing more data to the model might improve its performance on the data. Below is the confusion matrix calculated on the training and validation data.





XG-Boost Classifier - The F1 score for training data and validation data is 0.98 & 0.75 respectively. Hence, it is clearly visible that the model is overfitting on the given dataset. Providing more data to the model might improve its performance on the data. Below is the confusion matrix calculated on the training and validation data.




20 3 0 0 0 0 0
2 18 2 0 0 0 0
0 7 4 0 0 0 0
0 0 0 3 1 0 0
1 0 0 1 1 0 2
0 0 0 1 0 2 2
0 0 0 0 0 1 5
0 1 2 3 4 5 6
20.0 17.5 15.0 12.5 10.0 7.5 5.0 2.5 0.0

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

The culmination of this research project marks a significant stride in the realm of applied geophysics, where the development of a machine learning-based application for the detection of geological facies through well-log data analysis has been meticulously explored. The journey began with the establishment of a robust data preprocessing pipeline, addressing critical challenges such as missing values to ensure the quality and reliability of the dataset. Subsequently, the application of three distinct machine learning models—Support Vector Classifier (SVC), Random Forest, and XG Boost—unfolded diverse insights into their respective strengths and limitations in the task of identifying geological facies.

The performance metrics, notably the F1 scores, confusion matrices, and model evaluations, have cast a spotlight on the effectiveness of the models. The SVC, characterized by an F1 score of 0.734 for training data and 0.618 for validation data, suggests an underfitting scenario. This signals an opportunity for improvement through heightened model complexity. Conversely, the Random Forest and XG Boost models exhibit signs of overfitting, with F1 scores of 0.98 for training data and 0.75 for validation data. Addressing this overfitting could potentially be achieved by providing more data to enhance the models' generalization capabilities.

5.2 Future Work

Looking ahead, the future trajectory of this research will focus on delving into advanced methodologies to further elevate the accuracy and robustness of geological facies identification. Deep learning techniques will take center stage, leveraging their capacity to

autonomously extract intricate hierarchical features, particularly adept at handling complex geological patterns. This exploration aligns with the overarching goal of enhancing the models' ability to discern subtle nuances within the well-log data.

Moreover, 3D interpolation methods will be a key avenue of investigation, aiming to interpolate geological facies existing between well logs. This three-dimensional perspective holds the promise of providing a more comprehensive understanding of subsurface formations, enriching the insights gleaned from traditional two-dimensional analyses.

In essence, this research not only contributes valuable insights to the field of applied geophysics but also lays the groundwork for future endeavors that seek to push the boundaries of accuracy and comprehensiveness in geological facies identification through the synergy of advanced machine learning techniques and innovative data analysis methodologies.

REFERENCES

- Image-guided 3D interpolation of borehole data Dave Hale, Center for Wave Phenomena, Colorado School of Mines
- Missing log data interpolation and semiautomatic seismic well ties using data matching techniques Sean Bader1 , Xinming Wu1 , and Sergey Fomel1
- Geological structure guided well log interpolation for high-fidelity full waveform inversion Yangkang Chen,1,∗ Hanming Chen,2 Kui Xiang2 and Xiaohong Chen
- Imputation of missing well log data by random forest and its uncertainty analysis panelRunhai Feng a, Dario Grana b, Niels Balling

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import matplotlib.pyplot as plt
from tqdm import tqdm

df =
pd.read_csv('/kaggle/input/well-log-facies-dataset/facies_data.csv')
print(df.shape)
df.head()

df.describe()

df['Formation'].value_counts()

df['Well Name'].value_counts()

final_df = df[df['Well Name']=='CROSS H
CATTLE'].reset_index().sort_values(by=['Depth'])
final_df.head()

logs = ['GR','ILD_log10','DeltaPHI','PHIND','PE','NM_M','RELPOS']
plt.figure(figsize=(20,50))
for i,log in enumerate(logs):
    plt.subplot(4,2,i+1)
    plt.scatter(final_df[log],final_df['Depth'])
    plt.xlabel(log)
    plt.ylabel('Depth')

import random

def drop_values(df,cols=['GR'],drop_perc=0.1):
    temp_df = df.copy()
    for col in tqdm(cols):
        print(col)
        drop_data = np.round(temp_df.shape[0]*drop_perc)
        while drop_data:
            index = random.randint(1,temp_df.shape[0]-2)
            if index==0:
                continue
            if type(temp_df[col][index])!=None:
                temp_df[col][index] = None
                drop_data-=1
            else:
                continue
    return temp_df
```

```
print(logs)

final_df_na.describe()
```

---------- Data Preparation Complete ----------

Idnetifying and Dealing with missing values

```
import missingno as msno

msno.matrix(final_df_na)

from scipy import interpolate

x_GR = final_df_na[final_df_na['GR'].notnull()][['Depth','GR']]
x_ILD_log10 = final_df_na[final_df_na['ILD_log10'].notnull()]
[['Depth','ILD_log10']]
x_DeltaPHI = final_df_na[final_df_na['DeltaPHI'].notnull()]
[['Depth','DeltaPHI']]
x_PHIND = final_df_na[final_df_na['PHIND'].notnull()]
[['Depth','PHIND']]
x_PE = final_df_na[final_df_na['PE'].notnull()][['Depth','PE']]
x_NMM = final_df_na[final_df_na['NM_M'].notnull()][['Depth','NM_M']]
x_RELPOS = final_df_na[final_df_na['RELPOS'].notnull()]
[['Depth','RELPOS']]
interp_lst = [x_GR,x_ILD_log10,x_DeltaPHI,x_PHIND,x_PE,x_NMM,x_RELPOS]

result_df = final_df_na.copy()
for col,x in tqdm(zip(logs,interp_lst)):
    f = interpolate.interp1d(x.iloc[:,0],x.iloc[:,1])
    xnew = result_df[result_df[col].isnull()]['Depth']
    ynew = f(xnew)
    result_df[col][result_df[col].isnull()] = ynew

final_df_na.describe()[logs]

result_df.describe()[logs]

diff_df = result_df[logs] - final_df[logs]
diff_df.head(10)

print(logs)

diff_df.pow(2).mean().pow(0.5)
```

Outlier Detection and Handling

```
import seaborn as sns

plt.figure(figsize=(20,15))
for i,col in enumerate(logs):
```

```
    if col=='RELPOS':
        continue
    plt.subplot(3,3,i+1)
    sns.histplot(result_df[col])
    plt.xlabel(col)

for col in logs:
    q1 = result_df[col].quantile(0.25)
    q3 = result_df[col].quantile(0.75)
    iqr = q3 - q1
    upper = q3 + 1.5*iqr
    lower = q1 - 1.5*iqr
    upper_outliers = np.where(result_df[col]>upper)[0]
    lower_outliers = np.where(result_df[col]<lower)[0]
    outliers = np.concatenate([lower_outliers,upper_outliers])
    result_df.loc[outliers,col] = result_df[col].median()

plt.figure(figsize=(20,15))
for i,col in enumerate(logs):
    if col in ['RELPOS','Depth']:
        continue
    plt.subplot(3,3,i+1)
    sns.histplot(result_df[col])
    plt.xlabel(col)
```

Data Scaling & Quality Check

```
from sklearn import svm
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler

result_df.columns

logs.append('Depth')
logs.append('Facies')

sns.heatmap(result_df[logs].corr(),annot=True)

logs.pop(len(logs)-1)

x = result_df[logs]
y = final_df['Facies']

from sklearn.model_selection import train_test_split as tts

x_train,x_test,y_train,y_test = tts(x,y,test_size=0.15,shuffle=True)
print(x_train.shape,y_train.shape,x_test.shape,y_test.shape)

scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

```
scaled_df = scaler.transform(result_df[logs])
plt.figure(figsize=(20,15))
for i,col in enumerate(logs):
    if col in ['RELPOS','Depth']:
        continue
    plt.subplot(3,3,i+1)
    sns.histplot(scaled_df[:,i])
    plt.xlabel(col)
```

Model Preparation

```
clf = svm.SVC()
clf.fit(x_train,y_train)

yhat_test = clf.predict(x_test)
yhat_train = clf.predict(x_train)

from sklearn.metrics import confusion_matrix,f1_score

sns.heatmap(confusion_matrix(y_train,yhat_train),annot=True)
print(f1_score(y_train,yhat_train,average='micro'))

sns.heatmap(confusion_matrix(y_test,yhat_test),annot=True)
print(f1_score(y_test,yhat_test,average='micro'))

from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier()
clf.fit(x_train,y_train)

yhat_train = clf.predict(x_train)
yhat_test = clf.predict(x_test)

sns.heatmap(confusion_matrix(y_train,yhat_train),annot=True)
print(f1_score(y_train,yhat_train,average='micro'))

sns.heatmap(confusion_matrix(y_test,yhat_test),annot=True)
print(f1_score(y_test,yhat_test,average='micro'))

import xgboost as xgb
from xgboost import XGBClassifier

model = XGBClassifier(**params)
model.fit(x_train,y_train-1)

yhat_train = model.predict(x_train)
yhat_test = model.predict(x_test)

sns.heatmap(confusion_matrix(y_train-1,yhat_train),annot=True)
print(f1_score(y_train-1,yhat_train,average='micro'))

sns.heatmap(confusion_matrix(y_test-1,yhat_test),annot=True)
print(f1_score(y_test-1,yhat_test,average='micro'))
```