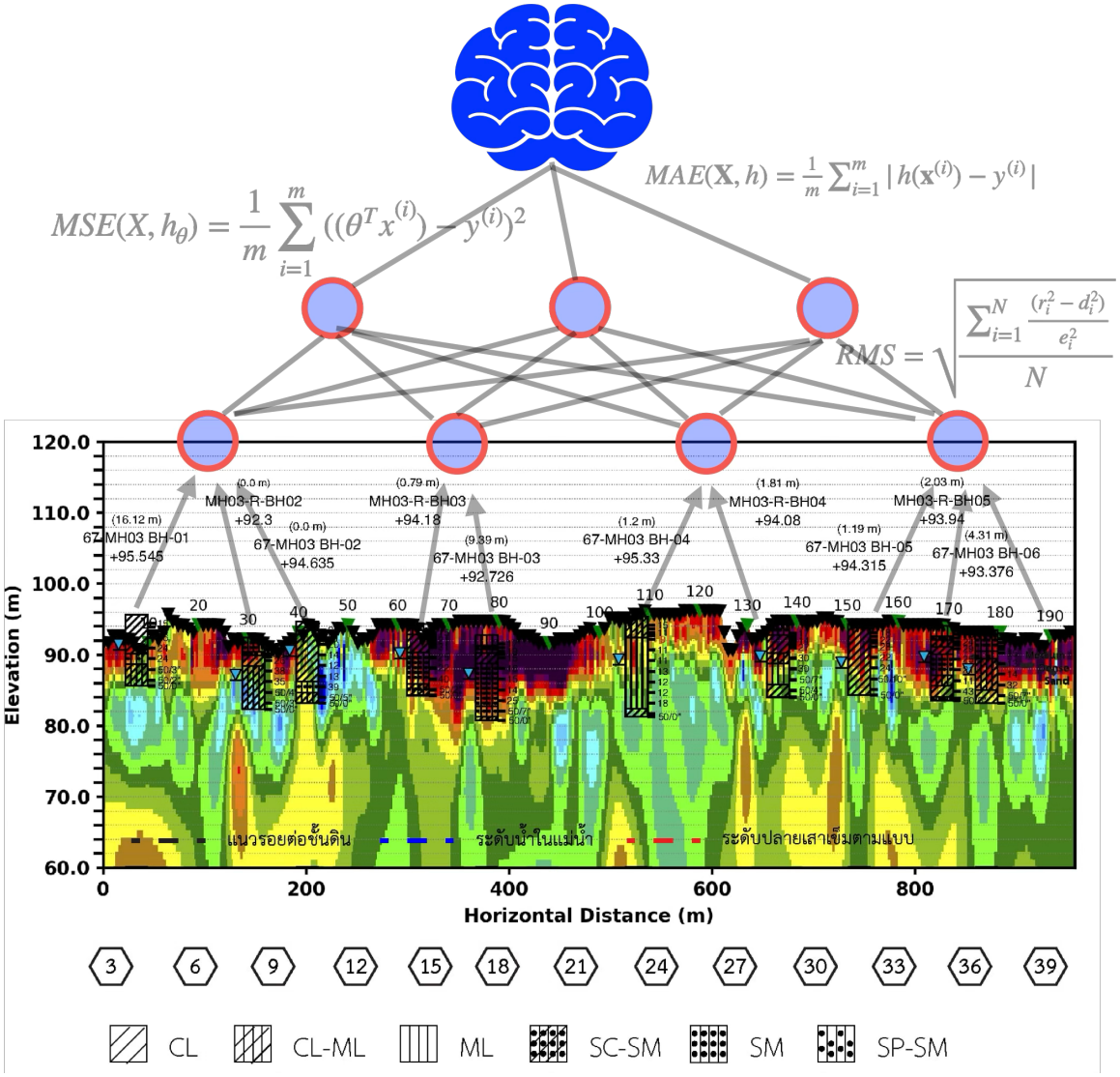
UNEARTHING SECRET

Machine Learning for soil classification
using geotechnical data and DC Resistivity Data



Songkhun Boonchaisuk

PREFACE

This report summarizes the culmination of a comprehensive 90-day **Scientist Invitation Program to Korea 2025** focused on machine learning and data analysis. The journey at Korea Institute of Geoscience and Mineral Resources (KIGAM) began with establishing a solid foundation in Python programming, essential mathematics, and SQL, which paved the way for a deeper exploration of core machine learning concepts.

Through a structured program, I progressed from fundamental algorithms like Regression Models to more advanced topics in deep learning, including Support Vector Machines (SVMs) and Reinforcement Learning. A significant portion of this training was dedicated to the practical application of these skills, from data preparation and feature engineering to the implementation of a final project using DC Resistivity Exploration datasets. This hands-on experience was crucial in bridging the gap between theory and real-world application.

This work would not have been possible without the guidance and support of many individuals. I would like to express my sincere gratitude to Dr. Seong Kon Lee for his invaluable help and advice throughout this period. My special thanks go to Dr. Minkyu Bang, whose insightful teaching and expert advice on machine learning. I am also deeply thankful to all lab members for their kind help, valuable suggestions, and for fostering a collaborative and supportive environment. All of the data was provided by Dr. ChatChai Vachiratienchai from Deep Geoscience.

I am grateful for this learning opportunity and look forward to applying the knowledge and skills acquired in teaching and researchs.

TABLE OF CONTENTS

PREFACE II

CHAPTER I INTRODUCTION 1

1.1. Background and Motivation 2
1.2. Objectives and Scope 3
1.3. Structure of the Report 4

CHAPTER II FUNDAMENTAL KNOWLEDGE AND TOOLS 5

2.1. Python for Data Science 6
2.2. Essential Mathematics for Machine Learning 9

CHAPTER III CORE MACHINE LEARNING ALGORITHMS: A THEORETICAL OVERVIEW 11

3.1. Regression Models 12
3.2. Support Vector Machines (SVMs) 13
3.3. Gradient Boosting 14

CHAPTER IV REGRESSION ANALYSIS OF GEOTECHNICAL DATA 16

4.1. The Geotechnical and Geophysical Dataset 17

4.2. Exploratory Data Analysis (EDA) 18
4.3. Feature Engineering 19
4.4. Application: Non-linear Regression on Borehole Data 20

CHAPTER V PRELIMINARY STUDY: FACIES CLASSIFICATION 22

5.1. Inspiration and ApproachThe 23
5.2. The Machine Learning Project Workflow 23

CHAPTER VI CAPSTONE PROJECT: FOCUSED SOIL CLASSIFICATION IN THE MUKDAHAN (MDH) AREA ..30

6.1. Problem Definition 31
6.2. Data Gathering and Preprocessing 31
6.3. Model Training and Evaluation 32

CHAPTER VII CONCLUSION 35

7.1. Summary of Learnings and Achievements 36
7.2. Challenges and Limitations 36
7.3. Future Directions 36

CHAPTER VIII ACTIVITIES 37

8.1. DC Resistivity Inversion for the Taebaek Dataset 38
8.2. Presentations and Other Activities 42
References 45

CHAPTER I
INTRODUCTION

The field of civil and geotechnical engineering is undergoing a significant transformation, driven by the integration of advanced data analysis and machine learning techniques. Traditionally reliant on established empirical methods and direct physical testing, the industry is now moving towards data-driven approaches to solve complex challenges, enhance predictive accuracy, and optimize project outcomes. This chapter introduces the context for this report, beginning with a specific, critical engineering problem—riverbank stability in Thailand—that serves as the primary case study. It will lay out the limitations of current practices and establish the motivation for applying a novel machine learning methodology. Subsequently, the chapter will define the precise objectives and scope of the project and provide an overview of the report's structure.

1.1. BACKGROUND AND MOTIVATION

Thailand, with its extensive network of rivers, faces a persistent and critical challenge: annual riverbank collapses. These events pose significant risks to infrastructure, property, and public safety. In response, government bodies such as the Royal Irrigation Department and the Department of Public Works and Town & Country Planning have initiated large-scale projects to construct protective embankment walls along vulnerable river sections.

The stability of these crucial structures, however, depends entirely on the integrity of their foundations. A key aspect of the current construction methodology involves:

- **Foundation Support:** Driving concrete piles into a stable, load-bearing soil layer known as the stiff soil basement.
- **Basement Identification:** The primary method for locating this basement layer is through drilling boreholes and conducting Standard Penetration Tests (SPT). A soil layer is typically considered a suitable basement if the SPT value exceeds 50 blows per foot, indicating very dense or hard soil.

This traditional approach, while reliable at the point of testing, suffers from a critical limitation that leads to structural failures:

- **The Problem of Interpolation:** Boreholes are expensive and time-consuming, meaning they are drilled at considerable distances from one another. To estimate the depth of the basement layer between these points, engineers often rely on simple linear interpolation.
- **Geological Complexity:** Subsurface geology is rarely simple or linear. The actual basement layer can vary significantly in depth and composition between boreholes. This discrepancy between the assumed and actual geology leads to piles that do not reach the stable basement, resulting in the eventual collapse of the embankment walls.
- **A New Approach:** To address this critical information gap, there is a move towards incorporating geophysical methods. DC Resistivity surveys, for example, can

provide a continuous 2D image of the subsurface electrical properties, offering a much more detailed view than sparse boreholes.

This leads to the central motivation for this project. While we have two valuable data sources—precise, "ground-truth" data from boreholes and comprehensive spatial data from resistivity surveys—interpreting them together is challenging. This project is motivated by the potential of machine learning to bridge this gap. By training a model on data where both sources are available (resistivity, water content, SPT, plastic limit, liquid limit), we aim to create a predictive tool that can accurately classify soil types and identify the basement layer over wide areas using resistivity data as a primary input. The ultimate goal is to enhance the efficiency, reduce the risks, and improve the overall reliability of geotechnical investigations for riverbank protection projects.

1.2. OBJECTIVES AND SCOPE

The primary objective of this project is to develop and validate a machine learning model for automated soil classification. The scope of the work encompasses the following key stages:

1. **Skill Acquisition:** A structured 90-day training program to build proficiency in Python, data analysis, and machine learning.
2. **Data Integration:** Collection, preprocessing, and integration of geotechnical borehole data and DC Resistivity survey data.
3. **Model Development:** Training and evaluation of various machine learning algorithms, including Support Vector Machines and Gradient Boosting.
4. **Analysis and Validation:** Rigorous assessment of model performance and interpretation of the results in a geotechnical context.

1.3. STRUCTURE OF THE REPORT

This report is organized into nine chapters. Chapter 2 covers the foundational knowledge in programming and mathematics. Chapter 3 provides a theoretical overview of the core machine learning algorithms used. Chapter 4 and 5 delve into the practical application of regression and classification. Chapter 6 presents the main capstone project in detail. Finally, Chapters 7 provide the conclusion.

CHAPTER II
FUNDAMENTAL KNOWLEDGE AND TOOLS

Before driving into the complexities of machine learning algorithms and their applications, it is essential to establish the foundational knowledge upon which this project is built. This chapter provides a detailed overview of the primary tools and theoretical concepts that were acquired and utilized during the training period. It begins with an exploration of the Python programming language and its powerful ecosystem for data science, covering key libraries for numerical computation, data manipulation, and visualization. Following this, the chapter will review the essential mathematical principles—spanning linear algebra, probability, and calculus—that underpin modern machine learning techniques. Finally, it introduces Scikit-learn, the versatile toolkit that brings these elements together, enabling the practical implementation of the models discussed in later

2.1. PYTHON FOR DATA SCIENCE

2.1.1. The Python Ecosystem:

Python has become the propular language for data science due to its simplicity, extensive libraries, and strong community support. For this project, an environment was established using Jupyter Notebooks, which provide an interactive platform for code development, visualization, and documentation.

2.1.2. NumPy for Numerical Computation:

The NumPy library is the cornerstone of numerical computing in Python. It provides a high-performance multidimensional array object and tools for working with these arrays. Its importance lies in its ability to perform vectorized operations efficiently, which is critical when processing large datasets. All Geothenical data, from borehole and inverted resistivity values, were represented and manipulated using NumPy lobrary.

2.1.3. Pandas for Data Manipulation:

The Pandas library is built on top of NumPy and provides data structures and data analysis tools. The core data structure, the DataFrame, is a two-dimensional table that is ideal for handling the structured borehole and geophysical data used in this project. Pandas was used extensively for tasks such as data loading, cleaning (e.g., handling missing values), filtering, merging, and aggregation (Code Snippet 2.1).

```
try:
    df_sample = pd.read_csv('Sample.csv')
    df_borehole = pd.read_csv('Borehole_2.csv')
    df_merged = pd.merge(df_sample, df_borehole,
left_on='borehole_id', right_on='id', suffixes=('', '_bh'))
    print("Files loaded and merged successfully!")

except FileNotFoundError as e:
    print(f"Error loading files: {e}. Please ensure both
'Sample.csv' and 'Borehole_2.csv' are in the directory.")
    df_merged = pd.DataFrame()

if not df_merged.empty:
```

```
    # --- Clean and Prepare the Merged Data ---
    for col in ['rho', 'spt', 'wn', 'll', 'pl']:
        df_merged[col] = pd.to_numeric(df_merged[col],
errors='coerce')
    df_merged.dropna(subset=['rho', 'spt', 'wn', 'll',
'pl', 'soil_type', 'profile_id', 'borehole_id'],
inplace=True)
    df_merged['PI'] = df_merged['ll'] - df_merged['pl']

    cols_to_check_for_zeros = ['rho', 'spt', 'wn', 'll',
'pl']
    df_no_zeros =
df_merged[(df_merged[cols_to_check_for_zeros] !=
0).all(axis=1)].copy()

    p_99 = df_no_zeros[['rho', 'spt', 'wn',
'PI']].quantile(0.99)
    df_final = df_no_zeros[
        (df_no_zeros['rho'] < p_99['rho']) &
        (df_no_zeros['spt'] < p_99['spt']) &
        (df_no_zeros['wn'] < p_99['wn']) &
        (df_no_zeros['PI'] < p_99['PI'])
    ]
    print(f"Data is ready. Total samples for analysis:
{df_final.shape[0]}")
```

Code Snippet 2.1: Example of loading and cleaning data with Pandas.

2.1.4. Data Visualization with Matplotlib and Seaborn:

Visualizing data is crucial for understanding it. Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations. Seaborn is built on top of Matplotlib and provides a high-level interface for drawing attractive and informative statistical graphics. These tools were used to perform Exploratory Data Analysis (EDA), creating histograms (Figure 2.1), scatter plots (Figure 2.2), and correlation matrices (Figure 2.3) to uncover underlying patterns in the data.

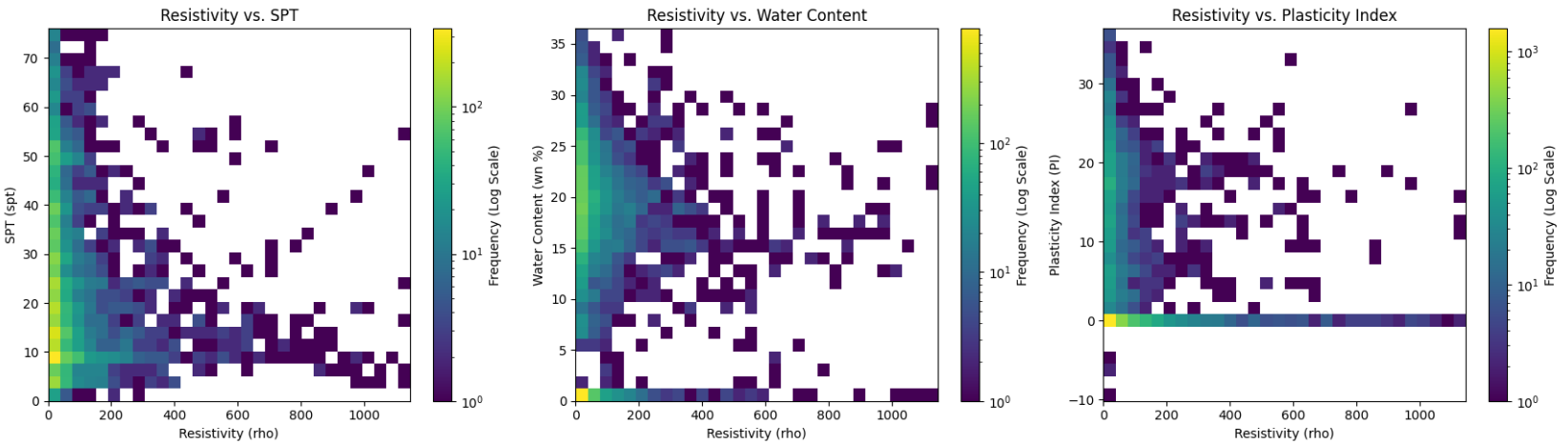


Figure 2.1: Example of a correlation matrix for geotechnical variables

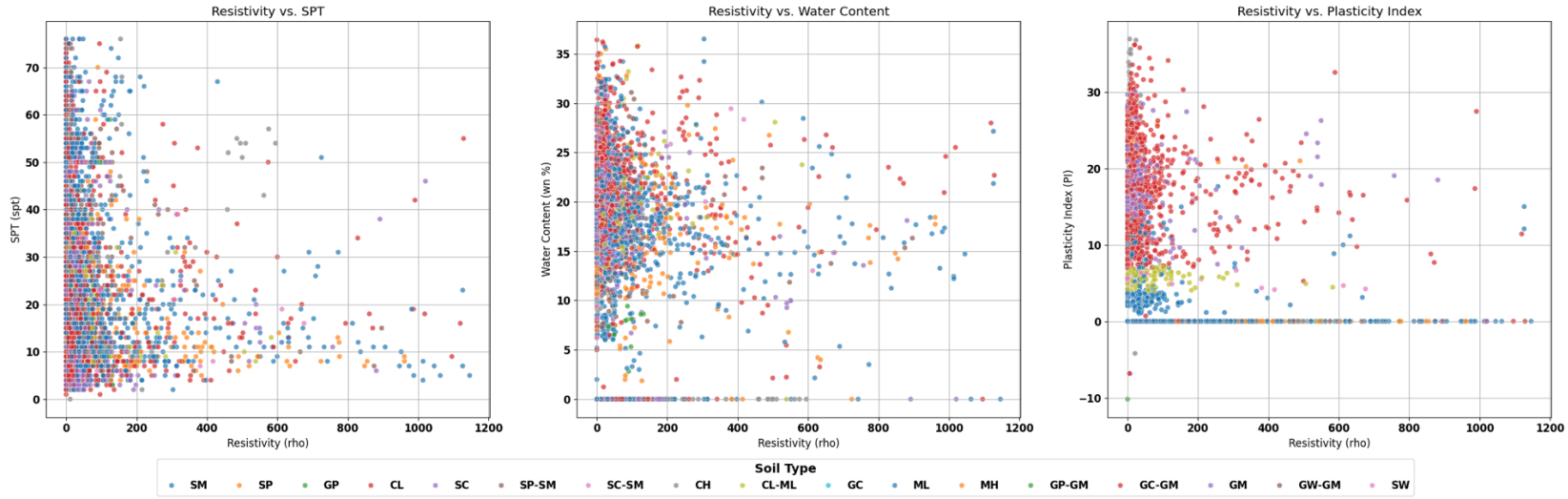


Figure 2.2: Example of a correlation matrix for geotechnical variables

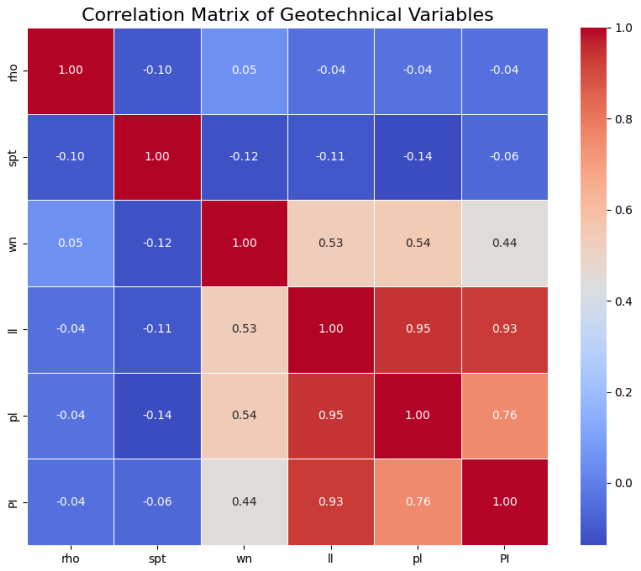


Figure 2.3: Example of a correlation matrix for geotechnical variables

2.2. ESSENTIAL MATHEMATICS FOR MACHINE LEARNING

2.2.1. Linear Algebra:

At its core, machine learning is applied linear algebra. Datasets are represented as matrices, and individual data points as vectors. Key concepts like matrix multiplication, transformations, and eigenvalues are fundamental to the operation of many algorithms, including Principal Component Analysis (PCA) and Support Vector Machines.

2.2.2. Probability and Statistics:

Probability theory provides the framework for quantifying uncertainty. Statistical concepts are essential for data exploration, model evaluation, and understanding model outputs. Key concepts such as probability distributions, conditional probability, and hypothesis testing were central to analyzing the data and interpreting model performance.

2.2.3. Calculus:

Calculus is the mathematics of change and is essential for model optimization. The most common method for training machine learning models is called gradient descent, which uses derivatives (gradients) of a loss function to iteratively update the model's parameters to minimize error. A conceptual understanding of this process is vital for understanding how models "learn."

2.3. Scikit-learn: The Machine Learning Toolkit

Scikit-learn is an open-source and foundational machine learning library for Python. It is built upon NumPy, SciPy, and Matplotlib and provides a wide range of simple and efficient tools for data mining and analysis. Its consistent and user-friendly API makes it an indispensable tool for nearly every step of the machine learning workflow. The library's key capabilities can be summarized as follows:

- **Classification:** Identifying to which category an object belongs. Common algorithms include Support Vector Machines (SVM), K-Nearest Neighbors, and Random Forests.
- **Regression:** Predicting a continuous-valued attribute associated with an object. This includes algorithms like Linear Regression, Ridge Regression, and Lasso.
- **Clustering:** Automatically grouping similar objects into sets or clusters. Popular methods are K-Means, Spectral Clustering, and Mean-Shift.
- **Dimensionality Reduction:** Reducing the number of variables under consideration to simplify models, remove noise, and improve computational efficiency. Key techniques include Principal Component Analysis (PCA) and various feature selection methods.
- **Model Selection:** Comparing, validating, and choosing the best models and their parameters. Scikit-learn provides powerful tools for this, such as Grid Search, Cross-Validation, and a comprehensive suite of performance metrics.
- **Preprocessing:** Transforming raw data into a clean and suitable format for machine learning models. This includes essential tasks like feature scaling, standardization, handling of categorical variables, and imputation of missing values.

CHAPTER III
CORE MACHINE LEARNING ALGORITHMS: A THEORETICAL OVERVIEW

With the foundational tools and mathematical concepts established, this chapter transitions to the core of the project: the machine learning algorithms themselves. It provides a theoretical overview of the key supervised learning models that were studied and implemented. The chapter will begin with regression models, which are used to predict continuous values, and then move to classification models, which are used to assign categorical labels. For each algorithm, the discussion will cover its fundamental principles, the underlying mathematical equations, and a conceptual example to illustrate its application in a geotechnical context. This theoretical grounding is crucial for understanding the models' strengths, weaknesses, and the rationale behind their application in the later chapters of this report.

3.1. REGRESSION MODELS

Regression analysis is a set of statistical processes for estimating the relationships between a dependent variable (the outcome or target) and one or more independent variables (the predictors or features).

3.1.1. Linear Regression:

This is the simplest form of regression. It assumes a linear relationship between the input variables and the single output variable. The goal is to find the best-fitting straight line (or hyperplane in higher dimensions) that minimizes the sum of squared differences between the predicted and actual values.

Equation 3.1: For a single feature, the model is represented as:

$$Y = \beta_0+\beta_1X+\epsilon$$

Where:

Y is the target variable (e.g., SPT N-value).

X is the feature (e.g., depth).

β_0 is the intercept (the value of Y when X is 0).

β_1 is the coefficient or slope (the change in Y for a one-unit change in X).

ϵ is the error term.

Example: Imagine plotting SPT N-values against their corresponding depths. Linear regression would find the single straight line that passes as closely as possible to all the data points, providing a simple model to predict SPT at a depth not explicitly measured.

3.1.2. Non-linear Regression:

In many real-world scenarios, including geotechnics, relationships between variables are not linear. Non-linear regression models (such as polynomial regression) are capable of capturing these complex relationships by fitting a non-linear curve to the data.

Equation 3.2: A common approach is polynomial regression, which extends the linear model by adding polynomial terms:

$$Y=\beta_0+\beta_1X_1+\beta_2X_2+...+\beta_nX_n+\epsilon$$

Example: If the soil strength increases exponentially with depth, a straight line would be a poor fit. A polynomial model could fit a curve that better captures this accelerating trend, leading to more accurate predictions at greater depths.

3.2. SUPPORT VECTOR MACHINES (SVMS)

Support Vector Machines are a powerful and versatile class of supervised learning models used for both classification and regression. For classification, the core idea of an SVM is to find the optimal hyperplane that best separates the different classes in the feature space.

3.2.1. The Maximal Margin Classifier:

The "optimal" hyperplane is defined as the one that maximizes the margin, which is the distance between the hyperplane and the nearest data points from each class. These nearest points are called "support vectors" because they are the critical elements that support the hyperplane.

Equation 3.3: The decision boundary is defined by $w \cdot x - b = 0$.

The goal is to find the weight vector w and bias b that maximize the margin $2/||w||$.

Example: Consider classifying soil samples as either 'Sand' or 'Clay' based on two features like resistivity and water content. An SVM would find the line that not only

separates the two groups but also maintains the largest possible gap or "street" between them. The data points that lie on the edge of this street are the support vectors.

3.2.2. The Kernel Trick:

For data that is not linearly separable, SVMs can use a technique called the kernel trick. This involves mapping the data into a higher-dimensional space where a linear separator can be found. Common kernels include the Polynomial kernel and the Radial Basis Function (RBF) kernel. This ability to handle non-linear boundaries makes SVMs particularly suitable for complex classification tasks like identifying soil and facies types.

Equation 3.4: The kernel trick works by replacing the dot product of the input features with a kernel function:

$$K(x_i,x_j)=\phi(x_i)\cdot\phi(x_j).$$

This allows the algorithm to operate in a high-dimensional feature space without ever having to compute the coordinates of the data in that space.

Example: If the 'Sand' and 'Clay' data points are arranged in concentric circles, no straight line can separate them in 2D. The kernel trick can project this data into 3D, transforming the circles into two parallel planes that can be easily separated by a simple flat plane (the hyperplane).

3.3. GRADIENT BOOSTING

Gradient Boosting is an ensemble learning technique that builds a strong predictive model by combining a series of "weak" learner models, typically decision trees.

3.3.1. Ensemble Learning and Boosting:

The core idea of ensemble learning is that by combining multiple simple models, one can create a more powerful and robust model. Boosting is a specific type of ensemble method where models are built sequentially.

3.3.2. The Boosting Process:

Gradient Boosting builds the first model on the data. Then, a second model is built to correct the errors made by the first model. This process is repeated, with each subsequent model focusing on the most difficult-to-predict cases. The final prediction is a weighted sum of the predictions from all the individual models. This iterative approach makes Gradient Boosting one of the highest-performing algorithms for tabular data.

Equation 3.5: The model is built additively:

$$F_m(x)=F_m^{-1}(x)+h_m(x)$$

Where $F_m(x)$ is the model at stage m, and $h_m(x)$ is the new weak learner trained to predict the negative gradient (residuals) of the loss function from the previous stage.

Example: To predict SPT values:

1. The first model might simply predict the average SPT for all samples. This will have large errors.
2. A second decision tree is trained, but its goal is to predict the errors (residuals) of the first model.
3. The prediction is now updated: Prediction = (prediction from model 1) + (prediction from model 2).
4. This process is repeated hundreds of times, with each new tree correcting the remaining errors of the combined ensemble, gradually improving the overall accuracy.

CHAPTER IV
REGRESSION ANALYSIS OF GEOTECHNICAL DATA

Having established the theoretical foundations of several key machine learning algorithms, we now move to their practical application. This chapter marks the transition from theory to hands-on data analysis, focusing on regression techniques to predict continuous geotechnical parameters. We will begin by introducing the specific dataset used for this study, a rich collection of borehole measurements and geophysical data. Subsequently, we will conduct a thorough Exploratory Data Analysis (EDA) to understand the characteristics of the data and the relationships between different variables. This is followed by a discussion on feature engineering, where domain-specific knowledge is used to create more informative inputs for our models. The chapter culminates in a practical regression exercise, applying a Gradient Boosting model to predict a critical soil property, thereby demonstrating the entire workflow from raw data to a predictive model.

4.1. THE GEOTECHNICAL AND GEOPHYSICAL DATASET

The primary dataset for this study is an aggregation of information from multiple boreholes, combined with corresponding geophysical data. Each row in this dataset represents a distinct soil sample from a specific depth interval. The key features are:

4.1.1 Identification and Location:

- `id`: A unique identifier for each soil sample.
- `borehole_id`: Identifies the borehole from which the sample was taken.
- `start`, `end`: The top and bottom depths of the sample interval in meters.

4.1.2 Geotechnical Properties (Direct Measurements):

- `spt`: The Standard Penetration Test N-value, a measure of soil strength and density, recorded in blows per foot.
- `wn`: The natural water content, expressed as a percentage.
- `pl`, `ll`: The Atterberg Limits—Plastic Limit and Liquid Limit, respectively. These are water content thresholds that define the boundaries between a soil's semi-solid, plastic, and liquid states. They are crucial for classifying fine-grained soils.
- `sievexx`: The percentage of the soil sample by weight remaining on a sieve of a certain size (e.g., `sieve40` is the percent retained on a No. 40 sieve). These are used for grain size distribution analysis to determine the proportions of gravel, sand, and fines.

4.1.3 Geophysical Property (Indirect Measurement):

- `rho`: The average electrical resistivity value in Ohm-meters, extracted from the inverted 2D DC Resistivity model corresponding to the sample's depth (`start_model` to `end_model`).

4.1.3 Classification:

- `soil_type`: The soil classification code (e.g., SM, ML) determined according to the Unified Soil Classification System (USCS). This serves as the ground-truth label for classification tasks.

This USCS, based on ASTM D-2487, is the standard for describing soils for engineering purposes. The classification is determined through a two-stage process based on grain size and plasticity:

4.1.5 Coarse-Grained vs. Fine-Grained:

Soils are first divided based on the percentage of material passing the No. 200 sieve (0.075 mm). If more than 50% is retained, it is coarse-grained (Gravel or Sand); otherwise, it is fine-grained (Silt or Clay).

4.1.6 Detailed Classification:

- **Coarse-Grained Soils (e.g., GW, GP, SM, SC):** These are further subdivided based on the percentage of fines (material passing the No. 200 sieve). "Clean" sands and gravels (SW, SP, GW, GP) have less than 5% fines and are classified by their grain size distribution (well-graded vs. poorly-graded). Soils with more than 12% fines (SM, SC, GM, GC) are classified based on the plasticity of those fines (silty 'M' or clayey 'C').
- **Fine-Grained Soils (e.g., ML, CL, MH, CH):** These are classified using the Atterberg Limits. The Liquid Limit (LL) and the calculated Plasticity Index (PI = LL - PL) are plotted on the USCS Plasticity Chart. The position of the point relative to the "A-line" (PI=0.73(LL−20)) and the vertical line at LL=50 separates the soil into categories such as Lean Clay (CL), Silt (ML), Fat Clay (CH), and High-Plasticity Silt (MH).

4.2. EXPLORATORY DATA ANALYSIS (EDA)

Before any modeling, a thorough EDA was performed on the dataset. The goals were to understand the distribution of each variable, identify potential outliers or errors, and examine the relationships between them. This involved creating histograms (Figure

2.1) to visualize the distribution of key continuous variables such as SPT (`spt`), water content (`wn`), and resistivity (`rho`). Scatter plots (Figure 2.2) and correlation matrices (Figure 2.3) were generated to visualize the relationships between these variables. For instance, plotting `spt` against `rho` helps investigate if a quantifiable correlation exists between soil strength and its electrical properties, which is a key hypothesis for this work (Figure 4.1).

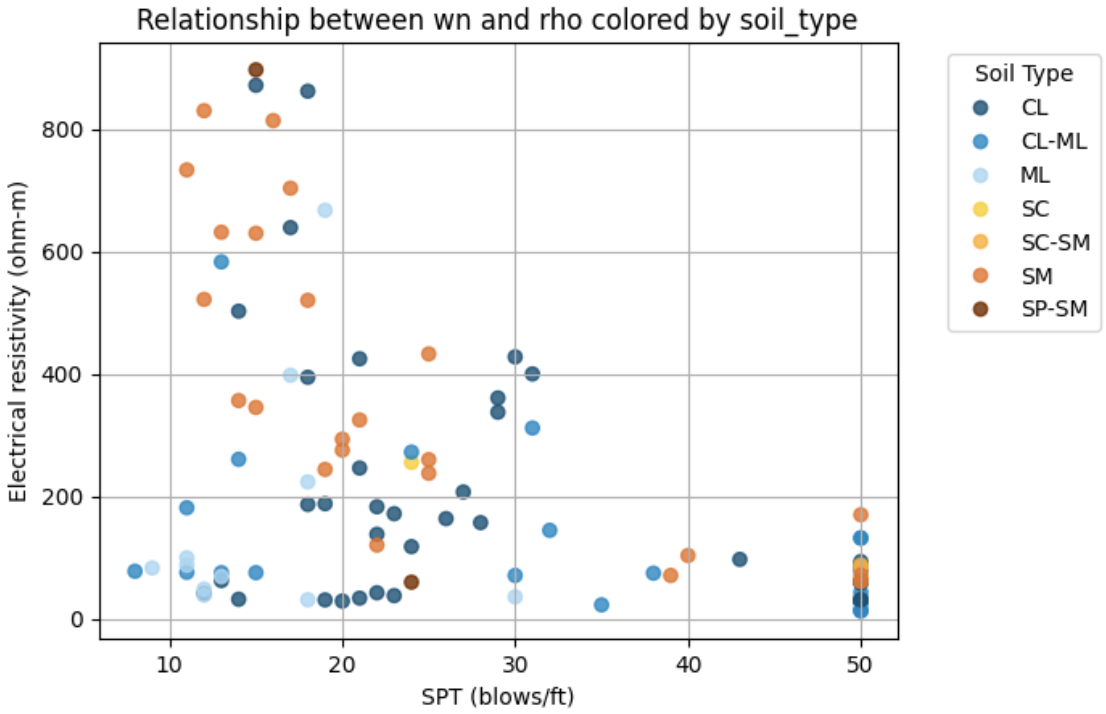


Figure 4.1: Scatter plot of SPT vs. Resistivity

4.3. FEATURE ENGINEERING

Feature engineering is the process of using domain knowledge to create new input features from the raw data to improve model performance. For the geotechnical data, several techniques were applied:

- **Plasticity Index (PI):** A crucial geotechnical parameter, the Plasticity Index, was calculated as the difference between the Liquid Limit and the Plastic Limit (PI=ll−pl). This new feature provides a quantitative measure of a soil's plasticity and is fundamental to the USCS classification, as seen in the Plasticity Chart.

- **Scaling:** Since variables like SPT and resistivity exist on vastly different numerical scales, feature scaling (e.g., Standardization) was applied. This process transforms the data to have a mean of zero and a standard deviation of one, ensuring that all variables contribute equally during the model's training phase.
- **Interaction Terms:** New features were created by combining existing ones (e.g., multiplying water content and plastic limit) to help the model capture potential non-linear interaction effects between soil properties.

4.4. APPLICATION: NON-LINEAR REGRESSION ON BOREHOLE DATA

A practical exercise was conducted to predict the SPT N-value (`spt`), a critical parameter for foundation design, using other available measurements. A Polyunomial Regressor was trained using resistivity (`rho`), water content (`wn`), plastic limit (`pl`), liquid limit (`ll`), and grain size information (`sievexx` columns) as input features. The model's performance was evaluated using metrics such as Mean Absolute Error (MAE) and R-squared. This exercise served to confirm the complex, non-linear relationships within the data and highlighted the predictive power of ensemble methods for geotechnical applications.

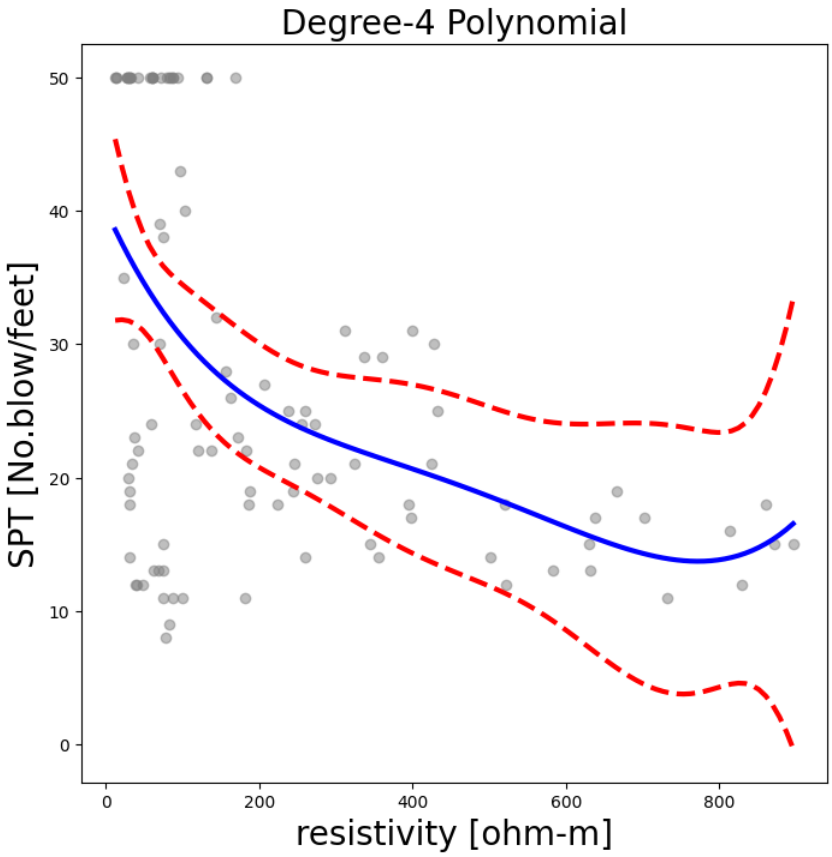


Figure 4.2: Scatter plot of SPT vs. Resistivity in grey dots, 4th order polynomial regrssion model in blue line with 95% confidential in red dashed-lines.

CHAPTER V
PRELIMINARY STUDY: FACIES CLASSIFICATION

This chapter shows a foundational exercise that was undertaken before commencing the main capstone project. The goal was to build and validate a complete machine learning workflow for classification, drawing inspiration from established methodologies in the geoscience community. By applying these techniques to the entire Sample.csv dataset, we aimed to gain practical experience in data conditioning, model training, and evaluation. This preliminary study was instrumental in developing the skills and insights necessary for the more focused and refined analysis presented in Chapter 6.

5.1. INSPIRATION AND APPROACHTHE

This initial phase of the practical work was inspired by a well-known machine learning tutorial from the Society of Exploration Geophysicists (SEG), which demonstrates the classification of sedimentary facies from well log data using a Support Vector Machine (SVM) by Brendon Hall (2016). Source codes and datasets canbe download via website: https://github.com/seg/tutorials-2016/tree/master/1610_Facies_classification. Facies are bodies of rock with specified characteristics. The goal of the SEG tutorial is to predict these facies categories in intervals of a well where no direct core samples are available, using only the continuous log data. This is a powerful application because it allows geoscientists to extrapolate detailed, point-source "ground-truth" information (from cores) along the entire length of a borehole. This served as a practical exercise to build a complete classification pipeline before proceeding to the specific challenges of the Mukdahan (MDH) case study.

After Download source codes, There is two errors about sklearn and NumPy libraries. In Facies Classification – SVM.ipynb, after compile the code if there is error find

```
    from sklearn.cross_validation import train_test_split
```

This command have to change to

```
    from sklearn.model_selection import train_test_split
```

The second error is NumPy value error

```
    adjacent_facies = np.array([[1], [0,2], [1], [4],
[3,5], [4,6,7], [5,7], [5,6,8], [6,7]], dtype=object)
```

Just add “,dtype=objet” at the end of np.array command.

5.2. THE MACHINE LEARNING PROJECT WORKFLOW

Following the tutorial's methodology, a complete workflow was implemented to classify Facies based on their physical and electrical properties.

5.2.1 Data Exploration and Conditioning:

The first step was to load the full dataset and perform an exploratory analysis. The data set we will use comes from a University of Kansas class exercise on the Hugoton and Panoma gas fields. For more on the origin of the data, see Bohling and Dubois (2003) and Dubois et al. (2007) and the Jupyter notebook that accompanies this tutorial at http://github.com/seg. A critical conditioning step was handling missing data points and then scaling all numerical features. Feature scaling is essential for SVMs, as the algorithm can be sensitive to features with vastly different ranges.

This dataset is from nine wells (with 4149 examples), consisting of a set of seven predictor variables and a rock facies (class) for each example vector and validation (test) data (830 examples from two wells) having the same seven predictor variables in the feature vector. The data set consists of seven features (five wireline log measurements and two indicator variables) and a facies label at half-foot depth intervals. Five wire line log curves include gamma ray (GR), resisitivity logging (ILD_log10), photoelectric effect (PE), neutron-density porosity difference (DeltaPHI) and average neutron-density porosity (PHIND). Two geologic constraingin variables are nonmarine-marine indicator (NM_M) and relative position (RELPOS). In machine learning terminology, the set of measurements at each depth interval comprises a *feature vector*, each of which is associated with a *class* (the facies type). We will use the pandas library to load the data into a dataframe, which provides a convenient data structure to work with well-log data. A critical conditioning step was handling missing data points and then scaling all numerical features using Scikit-learn's StandardScaler. This transforms each feature to have zero mean and unit variance, which is essential for SVMs, as the algorithm can be sensitive to features with vastly different ranges.

Table 5.1: Facies labels with their descriptions.

FACIES	LABEL	ADJACENT FACIES	CLASS OF ROCKS
1	SS	2	Nonmarine sandstone
2	CSiS	1,3	Nonmarine coarse siltstone
3	FSiS	2	Nonmarine fine siltstone
4	SiSh	5	Marine siltstone and shale
5	MS	4,6	Mudstone (limestone)
6	WS	5,7	Wackestone (limestone)
7	D	6,8	Dolomite
8	PS	6,7,9	Packstone-grainstone (limestone)
9	BS	7,8	Phylloid-algal bafflestone (limestone)

Scikit also includes a handy function to randomly split the training data into training and test sets. The test set contains a small subset of feature vectors that are not used to train the network. Because we know the true facies labels for these examples, we can compare the results of the classifier to the actual facies and determine the accuracy of the model. Let's use 20% of the data for the test set.

5.2.2 Model Training:

An SVM classifier from the scikit-learn library was trained on the conditioned dataset. The input features included all available numerical measurements: Gr, ILD_log10, DeltaPHI, PHIND, PE, NM_M and RELPOS columns. The model learned to find the optimal boundaries, or hyperplanes, in the multi-dimensional feature space to separate the different facies. The SVM is a map of the feature vectors as points in a multi dimensional space, mapped so that examples from different facies are divided by a clear gap that is as wide as possible.

5.2.3 Evaluation:

The performance of the trained classifier was evaluated using a confusion matrix (Figure 5.1). This allowed for a detailed visualization of the model's accuracy, showing which soil types were correctly predicted and where misclassifications occurred. The experience gained from interpreting this output was crucial for understanding the model's strengths and weaknesses and directly informed the refined approach taken in the subsequent capstone project.

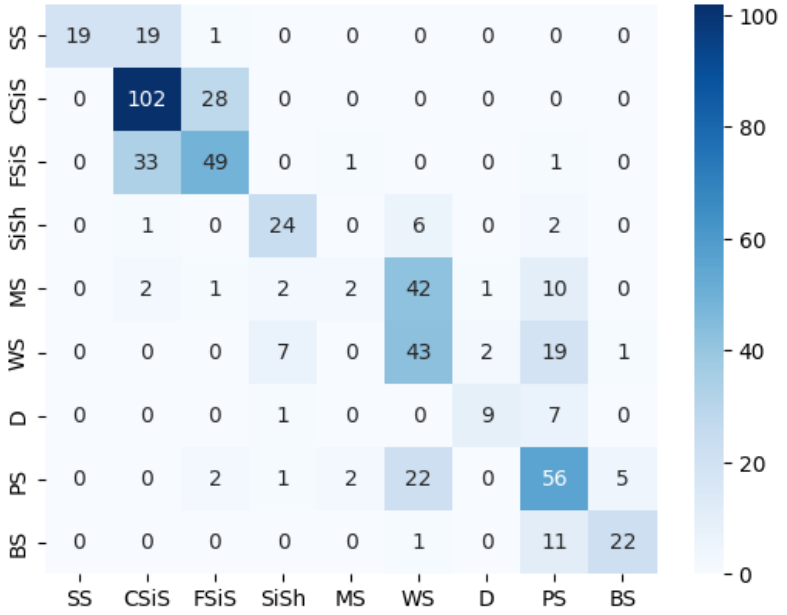


Figure 5.1: Confusion matrix from the preliminary SVM model on the full dataset.

The classifier so far has been built with the default parameters. However, we may be able to get improved classification results with optimal parameter choices. We will train a series of classifiers with different values for C and gamma. Two nested loops are used to train a classifier for every possible combination of values in the ranges specified. The classification accuracy is recorded for each combination of parameter values. The results are shown in a series of plots, so the parameter values that give the best classification accuracy on the test set can be selected (Figure 5.2).

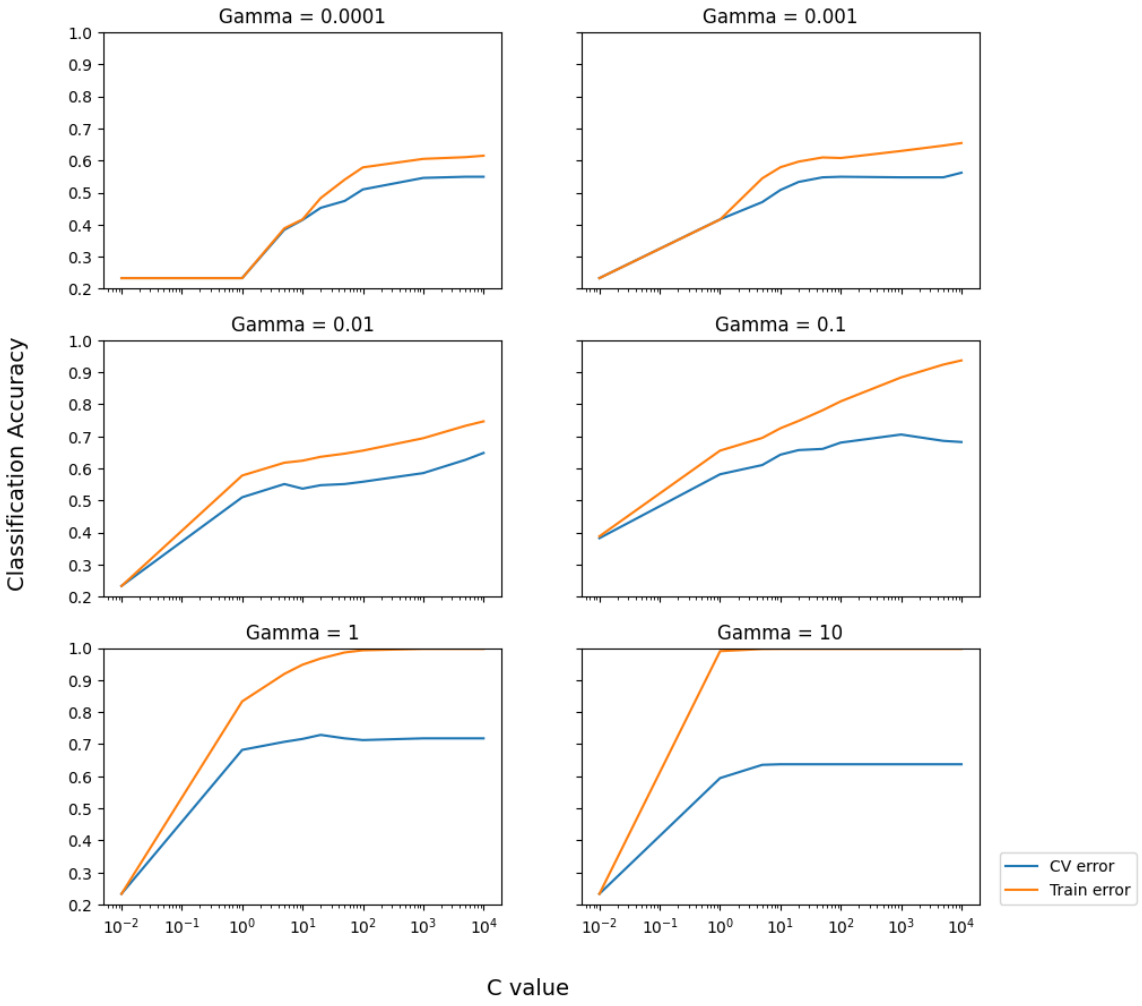


Figure 5.2: Cross validation error curve show best accuracy for C=10 and gamma = 1.

Precision and recall are metrics that give more insight into how the classifier performs for individual facies. Precision is the probability that given a classification result for a sample, the sample actually belongs to that class. Recall is the probability that a sample will be correctly classified for a given class. Precision and recall can be computed easily using the confusion matrix as shown in code snippet 5.1.

```
clf = svm.SVC(C=10, gamma=1)
clf.fit(X_train, y_train)

cv_conf = confusion_matrix(y_test, clf.predict(X_test))

print('Optimized facies classification accuracy = %.2f' %
accuracy(cv_conf))
print('Optimized adjacent facies classification accuracy =
%.2f' % accuracy_adjacent(cv_conf, adjacent_facies))
display_cm(cv_conf, facies_labels,
           display_metrics=True, hide_zeros=True)
```

Code Snippet 5.1: Example of precision and recall calculation.

Consider facies SS (nonmarine Sandstone), In the test set, if a sample was lebeled SS the probability the sample was correct is 0.88 (precision). If we know a sample has facies SS, then the probability it will be correctly labeled by the classifier is 0.77 (recall). It is desirable to have high values for both precision and recall, but often when an algorithm is tuned to increase one, the other decreases. The F1 score combines both to give a single measure of relevancy of the classifier results. F1 score for SS is 0.82.

5.2.4 Applying the classification model to the blind data:

Now we have trained facies classification model we can use it to identify facies to the blind dataset. The blind dataset was separated from training and testing dataset to test the model (Figure 5.3).

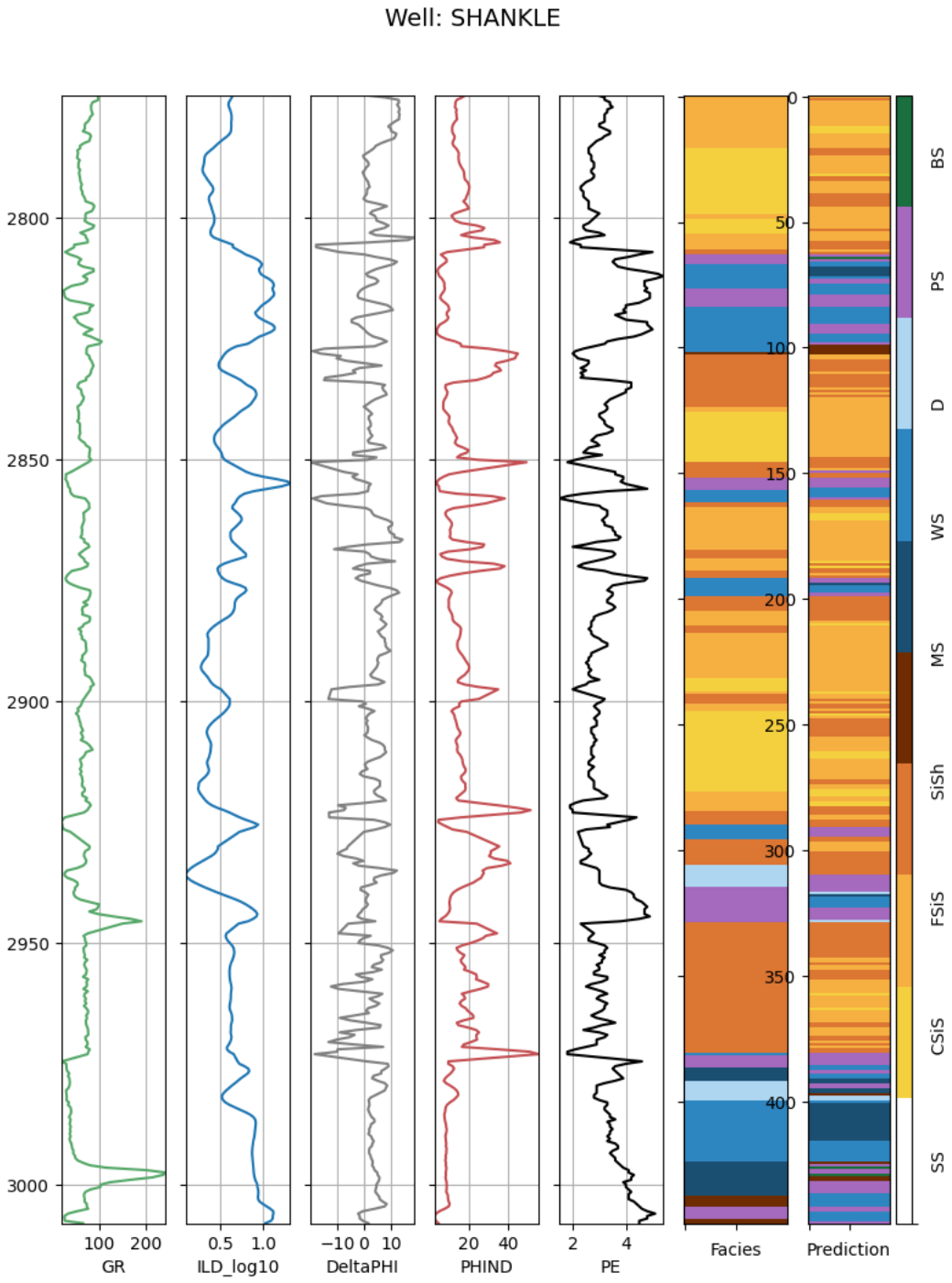


Figure 5.3: Cross validation error curve show best accuracy for C=10 and gamma = 1.

CHAPTER VI

CAPSTONE PROJECT: FOCUSED SOIL CLASSIFICATION IN THE MUKDAHAN (MDH) AREA

Building upon the foundational skills and preliminary investigations of the preceding chapters, this chapter presents the central contribution of this report: the capstone project. Here, the focus shifts from a general application of machine learning to a targeted, practical problem. The objective is to develop a specialized soil classification model for a specific region, Mukdahan (MDH), Thailand. This chapter details the entire process, from refining the dataset to address real-world challenges like class imbalance, to training and rigorously evaluating a final, optimized model. The work presented here represents the culmination of the 90-day training program, integrating all the acquired knowledge into a single, cohesive project.

6.1. PROBLEM DEFINITION

The capstone project narrows the focus to a specific geographical area of interest: Mukdahan (MDH), Thailand. The objective is to build a highly optimized machine learning model for soil classification tailored to the local geology of this region. This approach acknowledges that a non-uniqueness, inverted resisitivity model may not perform as well as a model trained on a more geographically and geologically consistent dataset by using machine learning.

6.2. DATA GATHERING AND PREPROCESSING

6.2.1. Dataset Overview:

The project utilizes a filtered subset of the original data, 01_MDH_Sample.csv. This file contains only the borehole data relevant to the Mukdahan area. As describe in chapter 2, five datasets including: SPT (`spt`), water content (`wn`), Plastic limit (`pl`), Liquid limid (`ll`), and resistivity (`rho`) were used as input for the input datasets.

6.2.2. Addressing Class Imbalance:

A critical step in the preprocessing phase was to address the issue of class imbalance. An initial analysis of the MDH dataset revealed that several soil classes, specifically SC (Clayey Sand), SC-SM (Silty, Clayey Sand), and SP-SM (Poorly-graded Sand with Silt), had very few samples compared to the other classes. Training a model with such imbalanced data can lead to a classifier that is biased towards the majority classes and performs poorly on the minority classes.

6.2.3. Data Grouping Strategy:

To mitigate this issue, a domain-informed grouping strategy was implemented, as detailed in the Soil_Classification - SVM_groupSM.ipynb notebook. The minority classes SC, SC-SM, and SP-SM were all re-labeled and grouped into the more general SM (Silty Sand) class. This is a justifiable simplification as all these soils are fundamentally coarse-

grained (sands) with varying amounts of fines. This process resulted in a more balanced and robust dataset for training the classifier.

6.3. MODEL TRAINING AND EVALUATION

6.3.1. Model Selection:

A Support Vector Machine (SVM) classifier was selected for the final model, building upon the experience from the preliminary study as describe in Chapter 5.

6.3.2. Evaluation Strategy:

A cross-validation strategy from confusion_matrix from Scikit-learn library was used to ensure that the model's performance was evaluated robustly. The next step, we will try a stratified k-fold cross-validation strategy to check the performance. This is particularly important for smaller datasets to ensure the results are not skewed by a single random train-test split.

6.3.3. Performance Metrics:

Model performance was assessed using a confusion matrix and the standard classification metrics: Accuracy, Precision, Recall, and F1-Score as show in Table 6.1 and Figure 6.1.

6.4. Results and Interpretation

The SVM model trained on the grouped Mukdahan dataset demonstrated strong performance in classifying the major soil types of the region. The results validate the data-driven approach, showing that a combination of geotechnical and geophysical parameters can be used to build an effective automated soil classifier. The confusion matrix provides detailed insights into which soil types are most easily distinguished and where potential misclassifications occur, offering valuable information for refining the model in the future.

Table 6.1: Comparison of performance metrics for the final MDH SVM model.

PRED TRUE	CL	CL-ML	ML	SM	TOTAL
CL	8				8
CL-ML		3	1	1	5
ML			1		1
SM	1	1		3	5
Precision	0.89	0.75	0.50	0.75	0.80
Recall	1.00	0.60	1.00	0.60	0.79
F1	0.94	0.67	0.67	0.67	0.78

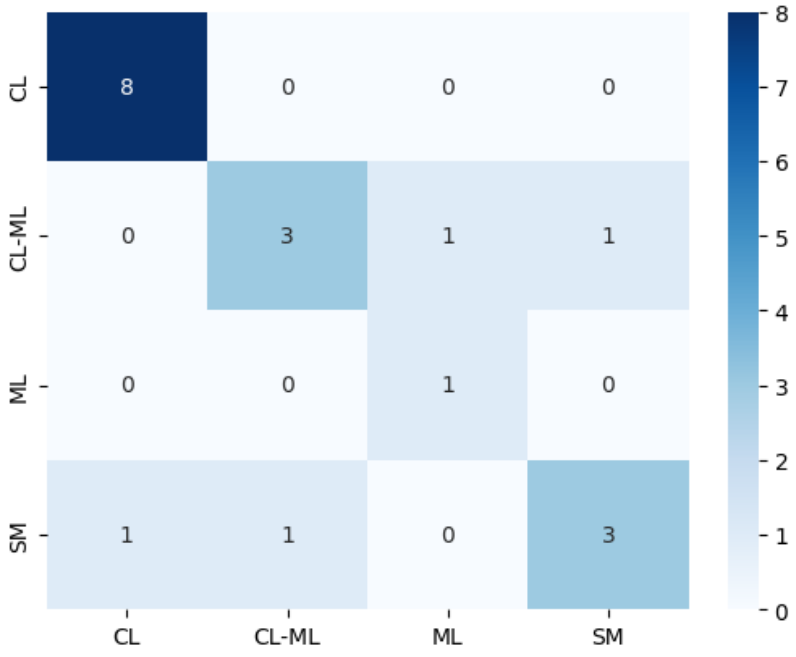


Figure 6.1: Confusion matrix for the final MDH SVM model.

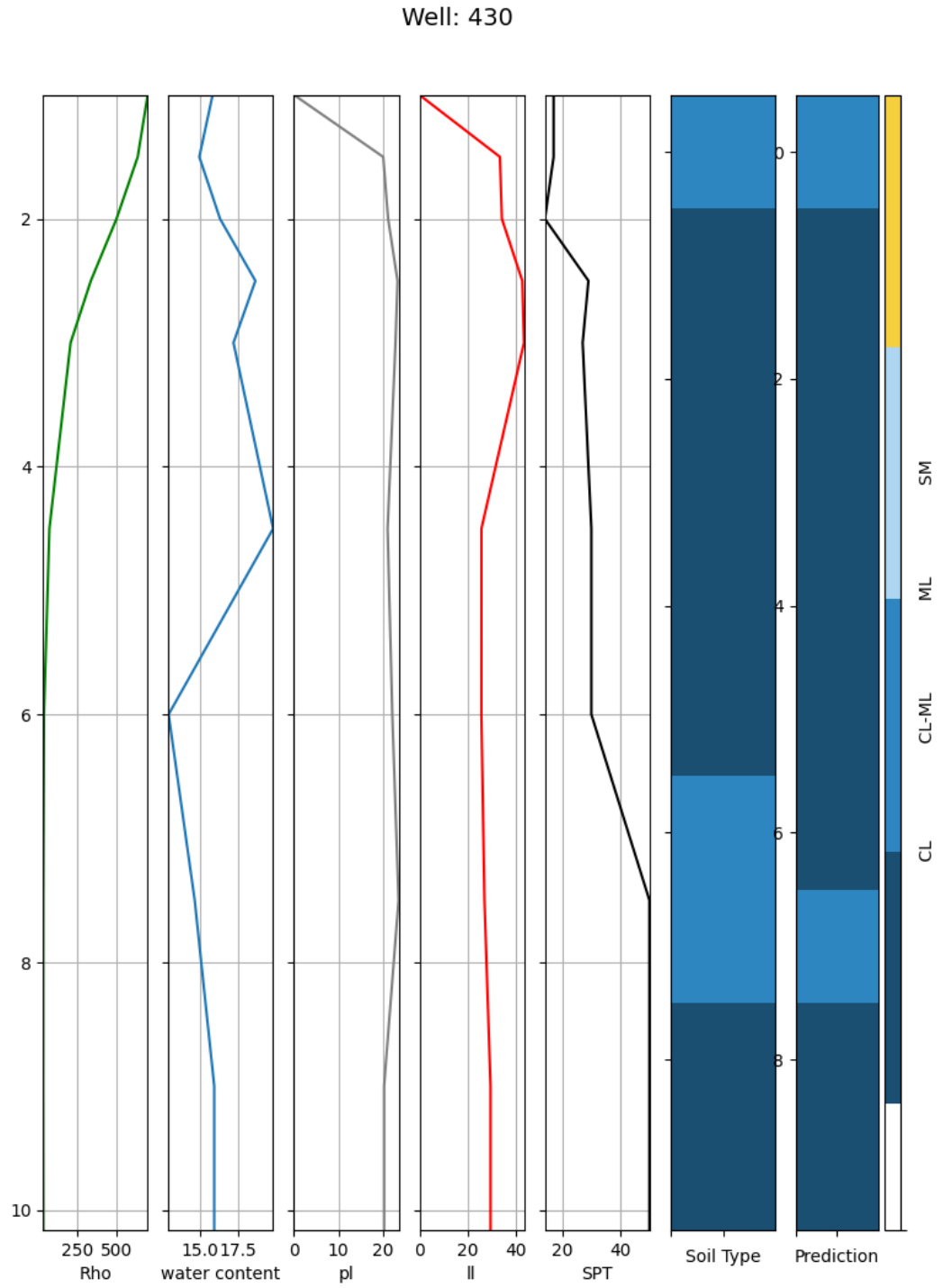


Figure 6.2: predicted soil classes for a survey line in the MDH area.

CHAPTER VII
CONCLUSION

This report has detailed a comprehensive journey through the fundamentals of machine learning and its application to a significant geotechnical engineering problem. From establishing foundational knowledge in Python and core algorithms to executing a focused capstone project, each chapter has built upon the last. This concluding chapter serves to synthesize the entire effort. It will summarize the key learnings and achievements of the 12-week program, candidly discuss the challenges and limitations encountered during the project, and finally, look ahead to propose promising directions for future research that can build upon the work presented here.

7.1. SUMMARY OF LEARNINGS AND ACHIEVEMENTS

This 90-day program provided a comprehensive education in the theory and practice of machine learning. The primary achievement was the successful development and application of a machine learning pipeline for a real-world geotechnical problem. This involved data exploration, preprocessing, feature engineering, and a crucial data-informed strategy to handle class imbalance. The final capstone project delivered a focused SVM model for soil classification in the Mukdahan area, demonstrating the feasibility of using this technology to enhance site characterization.

7.2. CHALLENGES AND LIMITATIONS

The main challenges encountered were related to data quality and class imbalance. The model's performance is inherently limited by the quality and quantity of the available training data for the MDH area. The grouping of minority classes, while a necessary practical step, means the model cannot distinguish between the finer subclasses of sandy soils. Furthermore, the model is specific to the Mukdahan geological environment and would require retraining to be applied to other sites.

7.3. FUTURE DIRECTIONS

Future work could expand on this project in several exciting directions:

- **Incorporate to Resistivity crossection:** Integrating other technique or regression for geotechnical data could further improve the interpreation or soil classification.
- **Explore Advanced Models:** While SVM performed well, testing more advanced ensemble models like Gradient Boosting or Random Forests on the refined MDH dataset could yield further performance improvements.
- **Develop a real-time tool:** The trained model could be deployed as part of an interactive software tool for geotechnical engineers, providing real-time predictions to aid in decision-making during site investigations in the Mukdahan region.

The capstone project narrows the focus to a specific geographical area of interest:

CHAPTER VIII
ACTIVITIES

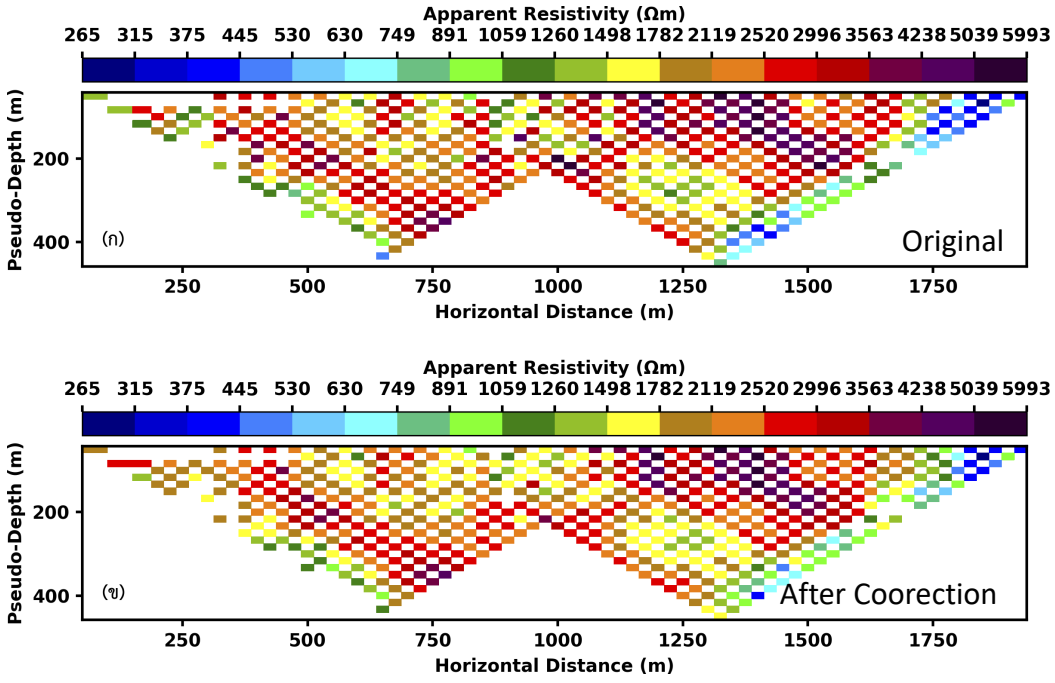
Beyond the primary focus on machine learning development, the training program encompassed a range of practical geophysical data processing tasks and other activities. This chapter documents these efforts, which provided valuable hands-on experience and opportunities for knowledge sharing. It details the work on DC resistivity data from the Taebaek dataset, including data processing and a comparative analysis of different inversion software. This chapter also summarizes other significant activities, such as presentations delivered to the KIGAM community and involvement in field equipment testing, which rounded out the comprehensive learning experience.

8.1. DC RESISTIVITY INVERSION FOR THE TAEBAEK DATASET

A significant practical exercise involved processing and inverting a DC resistivity dataset from Taebaek. This work was conducted using two industry-standard inversion software packages: RES2DINV and DGS2DINV, with a specific focus on incorporating topography into the models.

8.1.1 Data Processing:

The initial step was to process the raw observed data. This involved identifying and correcting noisy data points that were significantly different from their neighbors. These outliers were replaced by the average value of the surrounding data points (Figure 8.1). This processing step resulted in a 25.04% difference between the raw and the processed datasets and was crucial for achieving a stable inversion.



Data Difference: 25.04%

Figure 8.1: Comparison of a) observed and b) processed DC resistivity data]

8.1.2 Inversion with RES2DINV:

The processed data was inverted using both the new and legacy versions of RES2DINV, with the resulting resistivity models shown in Figure 8.2.

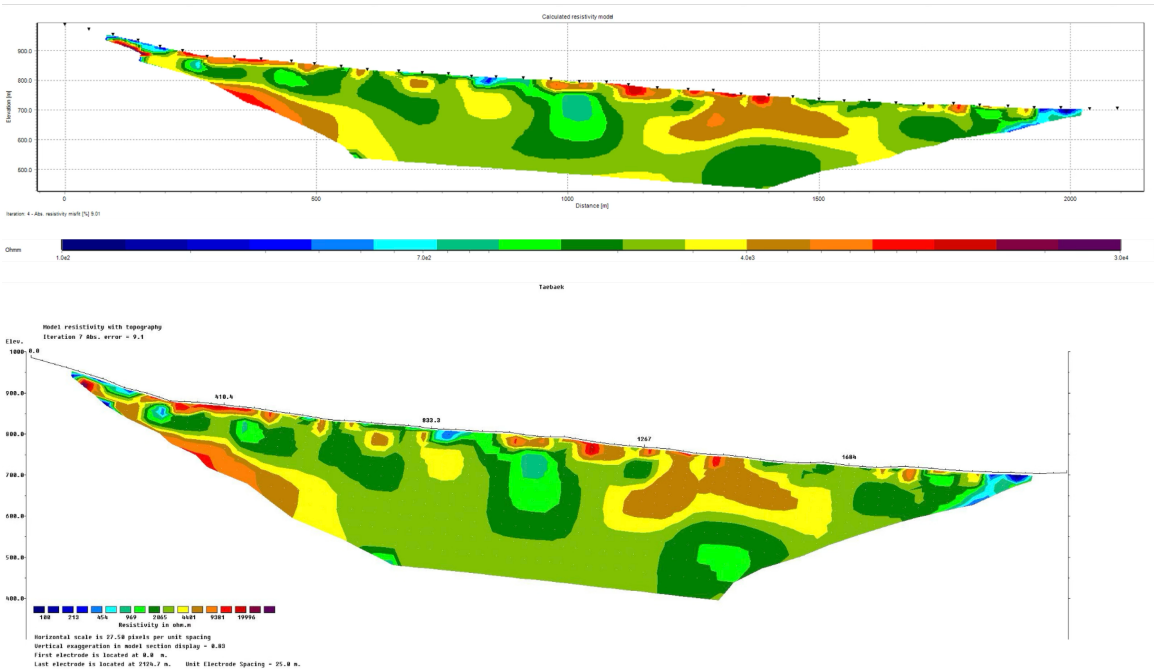


Figure 8.2: Inverted resistivity models from a) new and b) legacy versions of RES2DINV

8.1.3 Inversion with DGS2DINV:

A more extensive analysis was performed with DGS2DINV. The software was used to invert the original raw data, the processed data, and also to test a new "high noise inversion" algorithm. The results are compared in Figure 8.3.

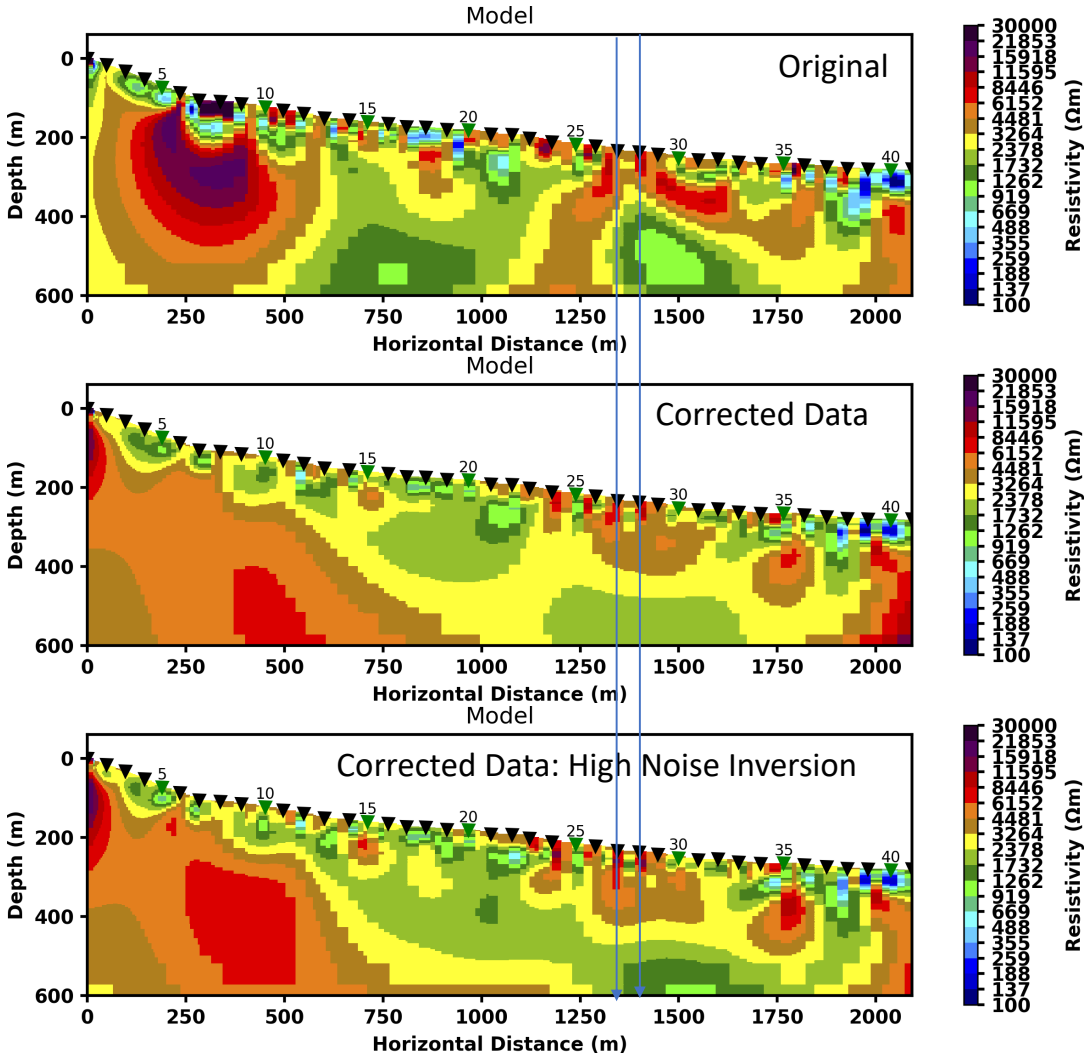


Figure 8.3: Inverted models from DGS2DINV for a) original data, b) processed data, and c) high noise inversion

8.1.4 Analysis of High-Resolution Data:

During the inversion process, unusually long calculation times and high RMS misfit values were observed. It was hypothesized that this was due to the high-resolution gradient dipole-dipole data included in the full dataset. To test this, a comparative study was conducted. The high noise inversion algorithm was run on the full processed dataset and then on a reduced dataset with the gradient dipole-dipole data removed. The results, shown in Figures 8.4 and 8.5, were compared to analyze the impact of the high-resolution data on inversion stability and quality.

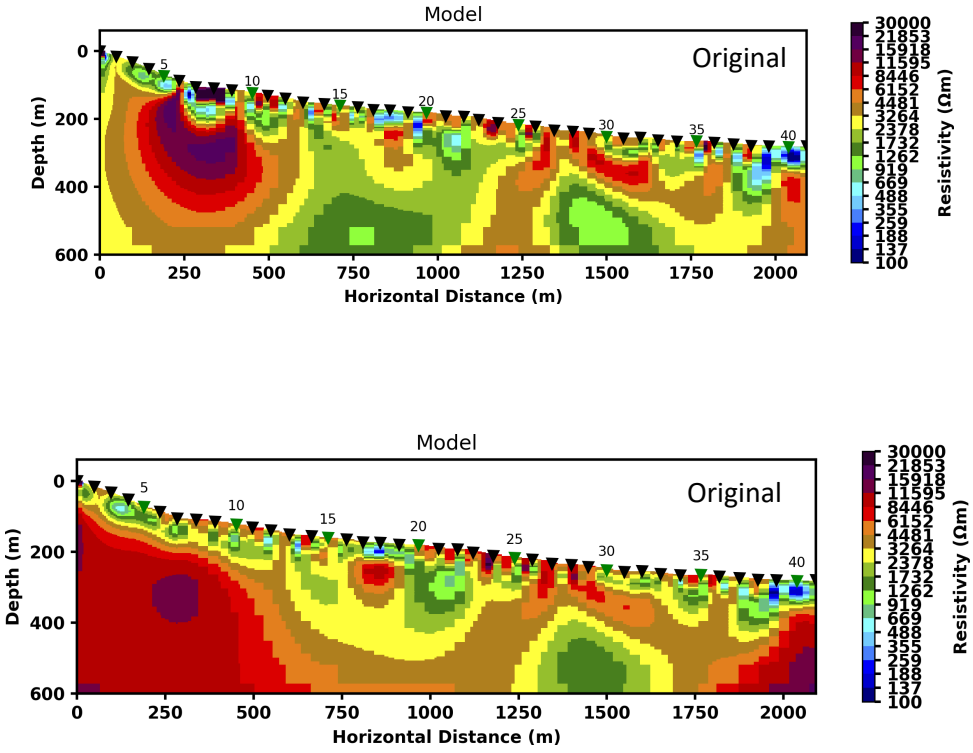


Figure 10.4: High noise inversion result for the full processed dataset

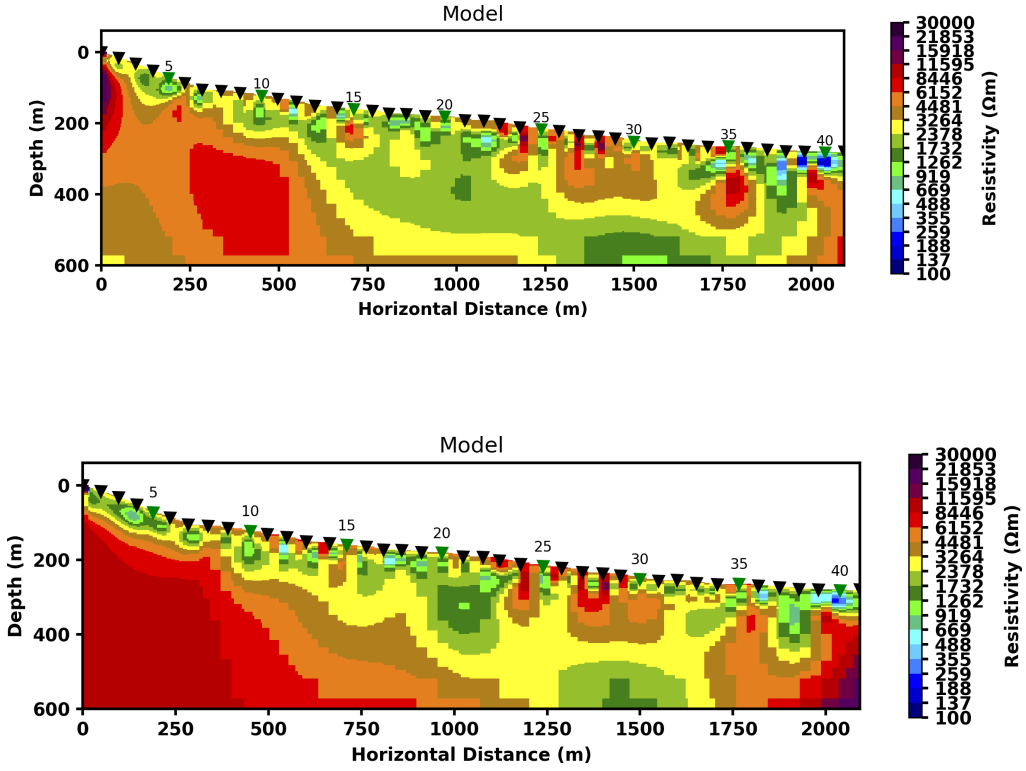


Figure 8.5: High noise inversion result for the dataset without gradient dipole-dipole data

8.2. PRESENTATIONS AND OTHER ACTIVITIES

8.2.1 Presentations:

At the beginning of the training program, on July 2, 2025, I represented the Thai participants by giving a presentation titled "An Overview of Thailand's Science and Technology Policy" (Figure 8.6). I also had the opportunity to present a talk on "The crustal study in Thailand" to KIGAM members (Figure 8.7).

Figure 8.6: Photo from the presentation on An Overview of Thailand's Science and Technology Policy

Figure 8.7: Photo from the presentation on the crustal study in Thailand

8.2.2 Field Work:

I participated in field testing of a magnetic drone, which provided practical experience with modern geophysical survey equipment (Figure 8.7).

Figure 8.7: Photo from the magnetic drone testing

Moreover, I have an opportunity to learnig and visit the air-borne electromagnetic survey at Taebaek as shown in Figure 8.8, 8.9 and 8.10

Figure 8.8: Electromagnetic, magnetic and radiometric sensor preparing

Figure 8.8: Helicopter used for air-borne electromagnetic survey

Figure 8.8: group photo

REFERENCES

1. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics.
2. McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. O-Reilly Media.
3. Hall, B. (2016). *Facies classification using machine learning*. SEG 2016 Annual Meeting. https://github.com/seg/tutorials-2016/tree/master/1610_Facies_classification
4. Hall, B. (2016). *Facies classification using machine learning.* The Leading Edge 35: 906–909. https://doi.org/10.1190/tle35100906.1
5. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.