

School of Mining Engineering



UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

**IMPROVING GRADE ESTIMATION USING MACHINE
LEARNING: A COMPARATIVE STUDY OF ORDINARY
KRIGING AGAINST MACHINE LEARNING ALGORITHMS**

Aniekan Akpabio

A research report submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Master of Science.

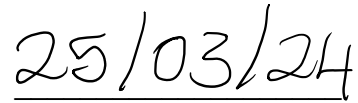
Johannesburg, 2024

DECLARATION

I declare that this report is my own, unaided work. I have read the University Policy on Plagiarism and hereby confirm that no plagiarism exists in this report. I also confirm that there is no copying nor is there any copyright infringement. I willingly submit to any investigation in this regard by the School of Mining Engineering and I undertake to abide by the decision of any such investigation.



Signature of Candidate



Date

“Success is not final; failure is not fatal: it is the courage to continue that counts”

Winston Churchill

ABSTRACT

This study investigated the efficiency of machine learning (ML) methods in the accurate prediction of ore grades, placing them in direct comparison with the established Ordinary Kriging (OK) methodology, a mainstay in geostatistical analysis. Utilising a dataset from a complex platinum group elements (PGE) deposit, the research assessed a suite of ML algorithms—namely, Random Forest (RF), Decision Trees (DT), Support Vector Regression (SVR), and particularly k -Nearest Neighbours (k NN). The latter is highlighted for its adeptness in assimilating spatial data correlations intrinsically, echoing the insights from Nwalia's analytical explorations.

The research engages with detailed swath plot analyses, comparative metric evaluations, and a nuanced understanding of spatial continuity, to illustrate the distinct advantages and operational competencies of the models. k NN, with its reliance on local data proximities and non-parametric nature, alongside RF, with its ensemble-based approach, emerged as capable in point estimate predictions. These models adeptly delineated local grade variations, demonstrating a high degree of reliability to the observed data and outperforming the OK model in both precision and accuracy.

Further, the study examined block estimate predictions, a cornerstone in practical mining and resource estimation, where both k NN and RF demonstrated a commendable ability to generalise predictions over larger spatial extents. This translates into significant potential for enhancing mineral resource estimation processes, tailoring them to the granular specifics of a given ore body, and refining block model accuracy to inform more strategic mining operations.

While the results endorse the ML methodologies as robust alternatives to traditional geostatistical techniques, the research also highlights the nuanced nature of these predictions. Factors such as the ore body's heterogeneity, the appropriateness of the variogram model, and the interplay between prediction scale and algorithmic performance are examined, offering a critical lens through which the suitability of each method is assessed.

The research suggests that while some models like LR and SVR are bounded by linear assumptions and hyperparameter sensitivities, non-linear models such as DT and RF can innately navigate the complex, multifaceted layers of geological data. The comprehensive evaluation extends to propose a novel set of performance metrics designed to capture the intricacies of grade prediction, thereby aligning closely with the operational demands and decision-making processes in the mining industry.

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my Creator and Heavenly Father, through whom all things are made possible. The unwavering love, support, and constant well-wishes from my father, mother, brothers, and sister have been indispensable in the completion of this project.

I also extend my sincere thanks to my colleagues at Impala Bafokeng: Palesa Nkhato, Ernest Mapukula, and Karin Barnard, for their invaluable assistance in putting geological and geostatistical concepts into clearer context for this research.

To all open-source enthusiasts and developers who contribute tirelessly to the Python community, your collective efforts in creating and maintaining libraries have been instrumental in advancing the field of data science and have provided invaluable tools for this research.

Finally, and most importantly, I express my profound gratitude to Professor R.C.A. Minnitt for his unwavering patience, integrity, depth of knowledge, and readiness to provide support at every stage of this research.

Table of Contents

DECLARATION	i
ABSTRACT	iii
ACKNOWLEDGMENTS	v
LIST OF FIGURES	ix
LIST OF TABLES	xi
LIST OF EQUATIONS	xii
LIST OF ABBREVIATIONS AND SYMBOLS	xiii
1 INTRODUCTION	1
1.1 Research Background and Context	1
1.2 Problem Statement	2
1.3 Justification for Research.....	2
1.4 Research Methods.....	3
1.5 Sources of Data	3
1.6 Research Validation.....	4
1.7 Structure of the Research Report.....	4
2 LITERATURE REVIEW.....	6
2.1 Introduction.....	6
2.2 Key Concepts, Theories and Studies	6
2.2.1 Geostatistics	6
2.2.1.1 Kriging	9
2.2.2 Machine Learning	10
2.3 Key Debates and Controversies.....	27
2.4 Chapter Summary.....	29

3	RESEARCH METHODOLOGY	30
3.1	Introduction	30
3.2	Research Design	30
3.2.1	Ordinary Kriging implementation.....	30
3.2.2	Machine Learning implementation	31
3.3	Data Collection Process.....	32
3.4	Data Validation Process.....	32
3.5	Data Analysis Approach: Estimation Criteria	32
3.6	Chapter Summary.....	34
4	DATA FOR THE STUDY.....	36
4.1	Introduction.....	36
4.2	Geological Setting.....	36
4.2.1	Regional Geology	36
4.2.2	Local geology	38
4.3	Project Background.....	38
4.4	Exploratory Data Analysis	39
4.4.1	Project X deposit	39
4.5	Variography	45
4.6	Machine Learning	56
4.6.1	Feature engineering: scaling, standardisation, and normalisation	57
4.6.2	Sci-kit Learn.....	59
4.6.3	Hyperparameters and Hyperparameter tuning	60
4.7	Chapter Summary.....	63
5	ANALYSIS AND FINDINGS.....	65
5.1	Introduction.....	65
5.2	GeostatsPy : Point and Block kriging functions.....	65

5.2.1	Point Prediction	65
5.2.2	Block predictions	66
5.3	ML and OK Model Validation and comparison: At point Location	69
5.4	Optimum Prediction parameters.....	76
5.4.1	Kriging Block Prediction.....	80
5.4.2	ML Block Prediction.....	83
5.5	ML and OK Model Validation and comparison: Block Predictions	86
5.6	Chapter Summary.....	98
6	CONCLUSIONS AND RECOMMENDATIONS	99
6.1	Introduction.....	99
6.2	Research Observations and Conclusion	99
6.3	Research Contributions	100
6.4	Research Limitations	101
6.5	Recommendations for Future Research Work	101
7	REFERENCES	103
	APPENDIX A: Figures	110

LIST OF FIGURES

Figure 2-1: Interpretation of a semi-variogram (Minnitt, 2020).....	7
Figure 2-2: Data pairing search template (Deutsch & Journel, 1997)	8
Figure 2-3: Euler diagram of Artificial intelligence (AI) (Gavrilova, 2020).....	11
Figure 2-4: Taxonomy of Machine Learning (ML) (Bhattacharjee, 2017).....	12
Figure 2-5: (a) Scatter plot of X and Y data (b) Straight line relationship between X and Y	14
Figure 2-6: An example of the NN search Problem	17
Figure 2-7: kNN classification example when $k = 3$ and $k = 7$ (Gokte, 2020)	19
Figure 2-8: Illustration of the SVM algorithm (Saini, 2023)	21
Figure 2-9: The Decision Tree process (The click Reader, 2021)	23
Figure 2-10: Illustration of a Random Forrest Tree algorithm (Sharma, 2023)	25
Figure 2-11: Academic Papers written on ML from 1993-2021 (Dumakor- Dupey & Arya, 2021)	27
Figure 2-12: Most popular ML algorithm used in grade estimation research (Dumakor-Dupey & Arya, 2021).....	28
Figure 4-1: Geological map of the BIC (Deseta, 2020).....	36
Figure 4-2: Stratigraphic column of the RLS (Deseta, 2020)	37
Figure 4-3: Lease area of Project X	38
Figure 4-4: Borehole Location map of the Project X deposit	39
Figure 4-5: Duplicate pairs in the deposit area.....	41
Figure 4-6: Histogram (left) and Boxplot (right) of PGE grade values.....	42
Figure 4-7: Probability plot of PGE grade.....	43
Figure 4-8: Project X location map according to PGE grades.....	44
Figure 4-9: Polygonal (left) and cell declustering (right)	46
Figure 4-10: Map of declustered weights over area of interest.....	47
Figure 4-11: Adjacent comparison of the original (Left) vs declustered histogram (Right)	48
Figure 4-12: Variogram map of the PGE in the Project X lease area.....	49
Figure 4-13: Experimental variograms with 9999.9m bandwidth tolerance.....	52
Figure 4-14: Experimental variogram with a reduced bandwidth	52
Figure 5-1: loocv_kb2d source code	69

Figure 5-2: Histogram of OK residuals (top), Scatterplot of predicted vs actual PGE grades.....	70
Figure 5-3: Histogram of LR residuals (top), Scatterplot of predicted vs actual PGE grades.....	72
Figure 5-4: Histogram of SVR residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)	72
Figure 5-5: Histogram of DT residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)	73
Figure 5-6: Histogram of RF residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)	73
Figure 5-7: Histogram of KNN residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)	74
Figure 5-8: custom_scoring function source code	79
Figure 5-9: PGE borehole location with estimation boundary	80
Figure 5-10: OK Grid Search parameter space	80
Figure 5-11: Grid dimension setup for ML block predictions.....	83
Figure 5-12: Python code for block aggregation for SVR predictions	84
Figure 5-13: OK swath plot at 200m intervals	88
Figure 5-14: DT swath plot at 200m intervals.....	88
Figure 5-15: RF swath plot at 200m intervals.....	89
Figure 5-16: SVR swath plot at 200m intervals	89
Figure 5-17: LR swath plot at 200m intervals	90
Figure 5-18: kNN swath plot at 200m intervals.....	90

LIST OF TABLES

Table 2-1: Example of coal data with predictor and response features	16
Table 4-1: Boreholes containing duplicate X, Y coordinates	40
Table 4-2: Summary Statistics of PGE grades	41
Table 4-3: Mean grade of PGE data capped at various thresholds	44
Table 4-4: Input parameters of the <i>geostatspy.geostat.declus</i> function.....	46
Table 4-5: Example of experimental variograms showcasing the iterative process.....	51
Table 4-6: Preferred experimental variogram parameters	53
Table 4-7: Table containing target and response variables of the Project X data	56
Table 4-8: A comparison between normalisation and standardisation.....	58
Table 4-9: Hyperparameter search space with descriptions.....	61
Table 5-1: <i>kb2d_locations</i> Input parameters.....	65
Table 5-2: Summary of the <i>kb2d</i> function input parameters.....	67
Table 5-3: Summary Result of estimated grade and error metrics	75
Table 5-4: <i>grid_search_kriging</i> Parameters and description.....	77
Table 5-5: Grid search Parameters and descriptions	77
Table 5-6: <i>calculate_performance_metric</i> function Parameters.....	78
Table 5-7: OK optimised block model parameters and Best OK results	82
Table 5-8: ML model blocks and block estimate	85
Table 5-9: Summary results of OK and ML block model error metrics.....	86

LIST OF EQUATIONS

(2.1) Ordinary Kriging Estimation Formula	10
(2.2) Ordinary Kriging variance Formula	10
(2.3) Linear regression equation with one predictor variable	14
(2.4) Modified linear regression equation accounting for statistical error	14
(2.5) Linear regression with multiple predictor variables.....	15
(2.6) k NN estimate of the regression function	20
(2.7) SVM model equation representing the hyperplane	21
(2.8) Calculation of the sign function to determine the class and hyperplane	22
(2.9) Constraint ensuring correct classification of data points	22
(2.10) Introduction of slack variable for handling miscalculation in soft margins.....	22
(2.11) Objective function for SVM optimisation with penalty term for misclassification.....	22
(2.12) Constraint for SVM optimisation problem incorporating slack variable and margin	22
(2.13) Representation of the decision tree for regression.....	24
(2.14) Objective function for constructing a regression tree, minimising the square error	24
(2.15) Definition of the first region partitioned by the decision node	24
(2.16) Definition of the second region partitioned by the decision node.....	24
(2.17) (2.14) Calculation for determining the optimal segmentation feature and threshold at a decision node	25
(2.18) Determination of output value for each sub-region in the regression tree	25
(3.1) R-squared formula.....	33
(3.2) MSE Formula	33
(3.3) RMSE formula	34
(3.4) MSE formula.....	34
(4.1) Data spacing equation for two dimensional volume	50
(5.1) SLOR calculation.....	78
(5.2) KE calculation.....	78

LIST OF ABBREVIATIONS AND SYMBOLS

Abbreviation	Description
2d	Two Dimensional
AI	Artificial Intelligence
BIC	Bushveld Igneous Complex
CoV	Coefficient of Variation
DT	Decision Tree
EDA	Exploratory Data Analysis
ERBFN	Elliptical
KE	Kriging Efficiency
<i>k</i> NN	<i>k</i> Nearest Neighbour
LOOCV	Leave-one-out Cross Validation
LR	Linear Regression
NN	Nearest Neighbour
MSE	Mean Squared Error
MAE	Mean Absolute Error
OK	Ordinary Kriging
PGE	Platinum Group Element
RLS	Rustenburg Layered Suite
RMSE	Root Mean Squared Error
RF	Random Forrest
SK	Simple Kriging
SLOR	Slope of Regression
SVM	Support Vector Machine
SVR	Support Vector Regression

1 INTRODUCTION

The life of a mining project is not determined when production begins, but, several years prior, during the exploration of a mineral deposit (Sinclair & Blackwell, 2002). As mining operations are known to be capital intensive, it is, therefore, of absolute importance during the evaluation process (of the mineral deposit) that the grade of the ore is accurately estimated. What culminates into a final decision on the feasibility of the project; grade estimation also lends itself to other aspects such as mine planning, production scheduling and processing capacity (Deutsch & Rossi, 2014).

Traditionally, grade estimation has relied heavily on geostatistical techniques which include a wide variety of kriging methods. Ordinary kriging (OK) is one of the most widely used kriging methods (Dumakor-Dupey & Arya, 2021).

1.1 Research Background and Context

Georges Matheron pioneered the geostatistical concepts of random function models and kriging in the 1960s (Matheron, 1963). Since then, Kriging has become synonymous with the acronym BLUE, meaning "best linear unbiased estimator" (Matheron & Klein geld, 1987). As mentioned, there exist various types of kriging in addition to OK, such as simple kriging (SK), indicator kriging, and factorial kriging analysis (Dumakor-Dupey & Arya, 2021). Each kriging method represents a stage in the evolution of geostatistics, from linear to nonlinear to multivariate.

OK has been championed by several scholarly articles; Afeni et al. (2021) highlighted this as OK was found to be superior to geometric methods since it involves the use of spatial modelling, and cross-validation tests can be done on estimates. Assumptions made within the family of kriging methods as well as the nature of the deposit impact its suitability for selection. Work carried out by Mpanza (2015) on a comparative study between OK and SK on a platinum group elements (PGE) deposit yielded mixed results, particularly in areas of high and low grades. This is further highlighted by Hekmatnejad et al. (2019) when comparing linear (OK) with nonlinear (disjunctive kriging) geostatistics. For gold grades with a heavy-tailed distribution, disjunctive kriging performed better than OK, however, copper grades that were much more moderately skewed yielded similar accuracy

between OK and disjunctive kriging. It is important to note that these conclusions depend on the implementation of the respective methods. OK was applied using a constant mean value and spatial correlation modelled through variograms, while disjunctive kriging involved transforming the data into normally distributed variables and using Hermite polynomials for prediction (Hekmatnejad, et al., 2019). Without detailed information on potential biases or inaccuracies in the implementation, it is difficult to definitively conclude that one method is superior to the other.

1.2 Problem Statement

OK and other kriging methods require certain assumptions to apply them effectively to produce accurate results. Kriging methods may not perform well in complex geological environments additionally, they require expert knowledge regarding variogram modelling and parameter selection. Subsequently, this has led to the search for alternative grade estimation methods (Dumakor-Dupey & Arya, 2021).

The search for alternative methods has resulted in an increase in the use of Machine Learning (ML) models. Such models have risen in popularity due to three main reasons:

1. Their ability to capture complex features,
2. They do not require assumptions about the spatial distribution of the grade,
3. They require less geoscientific knowledge (Erten *et al.*, 2021).

1.3 Justification for Research

The accurate estimation of mineral grades is critical for the sustainable and economically viable exploitation of mineral resources. Traditional geostatistical methods, such as Ordinary Kriging, have been extensively utilised for this purpose, benefiting from their ability to incorporate spatial autocorrelation among sample points. However, the advent of ML algorithms offers a promising alternative, potentially capable of capturing complex, non-linear relationships between mineral grades and various geological features that Ordinary Kriging might not fully exploit. Despite the growing interest in leveraging ML for grade estimation, a comprehensive comparison between these advanced algorithms and traditional

geostatistical methods is needed to understand their relative strengths and limitations.

This research aims to fill this gap by conducting a comparative study of Ordinary Kriging against various machine learning algorithms, specifically: Random Forests (RF), Support Vector Machines (SVM), Decision Trees (DT), Linear Regression (LR) and k Nearest Neighbour in the context of mineral grade estimation.

This research project aims to:

1. Evaluate the performance of OK against above mentioned ML algorithms.
2. Identify the best-performing ML algorithm.
3. Compare and contrast the results of different techniques and approaches used in the research.
4. To provide insight on how to use ML techniques to improve grades.
5. Provide recommendations for future research and implementation of ML.

1.4 Research Methods

This study will make use of quantitative and qualitative research methods. Borehole data will be collected and prepared; for OK, a traditional geostatistical framework will be applied. Regarding ML, after collection and preparation, the data will be split into training and test sets. From there onwards, the model will be trained with parameters being adjusted based on input features to obtain the best estimate. Lastly, the model(s) will be evaluated where accuracy will be measured. The Python programming language will be used to execute ML and OK workflows.

1.5 Sources of Data

Assay data of a mineral deposit containing all variables required for the study and stored within an excel file was the sole source of data used in the completion of this research project.

1.6 Research Validation

The validation criteria for the application of an ML model for grade estimation considered the following:

1. Accuracy: The most critical validation criterion is the accuracy of the model. The model must be able to estimate the grade with high accuracy.
2. Robustness: The model must be robust to outliers and noise in the input data. It should not produce unreliable estimates in the presence of outliers or noise.
3. Interpretability: The model should be interpretable, meaning that the relationship between the input data and the output should be easy to understand.
4. Generalisability: The model should be able to generalise to new data that it has not seen before. This means that it should be able to make accurate estimates for data that is not included in the training data.

1.7 Structure of the Research Report

This thesis consists of 6 chapters. Chapter 1 serves as the introduction which covers the background of the research topic, the problem statement, the justification for the research, and the research objectives. The research methodologies followed by this study and the sources of the information are introduced. The research validation criteria, as well as the contribution to knowledge, are discussed.

Chapter 2 reviews key concepts, theories and studies associated with geostatistics and machine learning. With geostatistics it details the critical methods and models used to analyse and predict spatially distributed geological data, emphasising the importance of understanding spatial relationships and continuity for accurate estimation and decision-making in geological studies. Specific to ML, an informative guide on the fundamentals, providing insights into its various sub-fields, methodologies, and algorithms, along with their practical applications and theoretical foundations.

Chapter 3 looks at the research approach used in this project. This is unpacked through the explanation of the research design, data collection process, data validation process and data analysis approach.

Chapter 4 investigates the data used for this project. The geological setting and context relating to the data used is provided. The quantitative and qualitative assessment of the mineral deposit is carried out through exploratory data analysis (EDA). From this, the geostatistical and ML workflows are completed with the establishment of grade estimation models for both.

Chapter 5 evaluates and validates the selection of the respective models through, findings of the models which are validated separately before direct comparison through numeric and visual methods.

Chapter 6 concludes this study reviewing key findings and observations. It outlines the research limitations before providing recommendations for future work.

2 LITERATURE REVIEW

2.1 Introduction

ML and geostatistical methods are both used to make predictions/estimates based on data, however, they differ in some keyways. Such discrepancies can only be fully appreciated by taking cognisance of the fundamental principles at work.

2.2 Key Concepts, Theories and Studies

2.2.1 Geostatistics

Geostatistics is a branch of statistics that deals with the study and analysis of spatial data. It aims to model and understand the spatial patterns and relationships of geological phenomena to make predictions and decisions based on available information.

A similar description of geostatistics is given by Journel & Huijbregts (1978), however, they further go on to state that the geological phenomena which can also be referred to as natural phenomena are characterised by the distribution of one or more variables in space which are called regionalised variables. It can therefore be said that variable (Z) which is distributed in space (x_i) is referred to as the regionalised variable $Z(x_i)$. An example of this would be variables contained within an assay data file, namely, grade, thickness, and lithology to list a few. Two aspects characterise a regionalised variable and that is the local random and the structured aspect (Journel & Huijbregts, 1978). At point x_i , $Z(x_i)$ can be referred to as a random variable; defined as a variable at a location which can assume a range of values (having multiple possible outcomes) represented by a probability density function.

The distribution of regionalised variables can be further explained through the concept of a random function which is a probabilistic description of the spatial distribution of a variable (Sinclair & Blackwell, 2002). The necessity of the random function in geostatistics is that it incorporates both the random and structured components of the spatial variability of a variable.

The random function $[Z(x)]$ which by definition is a set of random variables having a spatial location that when paired $[z(x) - z(x + h)]$, the outcome of each term is dependent on their pdfs. Graphically this can be represented with a semi-variogram (Isaaks & Srivastava, 1989).

Spatial continuity refers to the correlation between values over a given distance (h). At the head and tail of the vector for this distance lie values of a specific variable, $z(x)$ and $z(x + h)$, from which the difference between the two represents how similar they are (Minnitt, 2020). This comparison is done throughout a target area and can be visualised using experimental and model-fitted variograms. A semi-variogram is a function of distance and is given by the following formula, $\gamma = \frac{[z(x) - z(x+h)]^2}{2N}$. A modelled variogram is a best-fit line that best represents the general variability in the target area. Variograms are interpreted in terms of, the sill, range, and nugget effect, as shown in Figure 2-1.

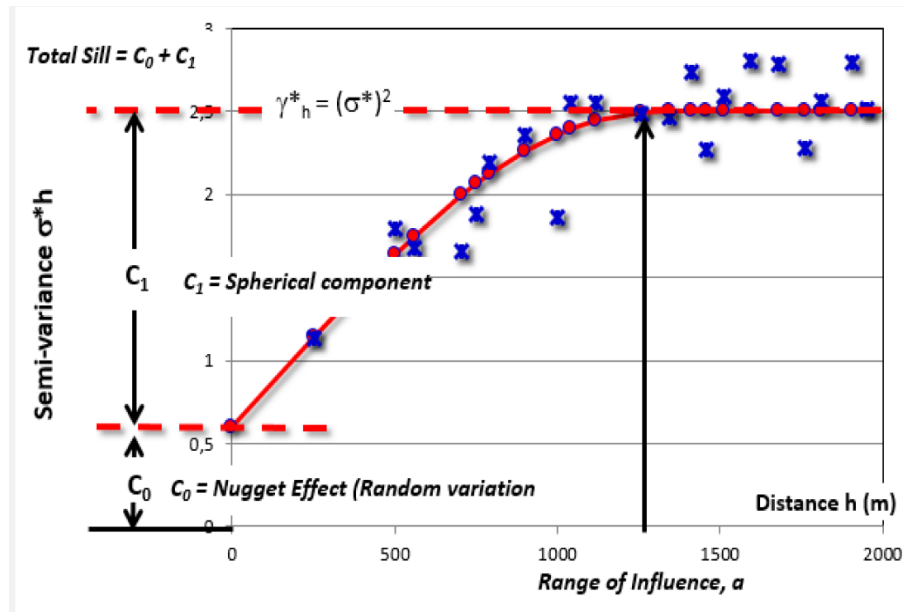


Figure 2-1: Interpretation of a variogram (Minnitt, 2020)

The sill (C) represents the total variance of the target area and is a summation of the random (C_0) and structured (C_1) component. The sill can be identified along the line where the variogram flattens out. At the distance which the variogram reaches the sill is called the range (a). The extent of the range can be interpreted as the range of influence, where the data is spatially correlated. Beyond this range,

the data is uncorrelated. The discontinuity at the origin is called the nugget effect which refers to the discontinuity in the target area at short distances (that is, distances less than the minimum spacing).

Spacing of the data can prove to be problematic when attempting to get pairs of data separated by a lag distance, particularly with scattered (irregularly spaced) data. With regularly spaced data there are fewer issues when searching for pairs along and across the data. There may be, however, problems in finding pairs in the diagonal direction of regularly spaced data. A search template as seen in Figure 2-2 makes use of tolerances to account for the would-be difficulties in obtaining data pairs.

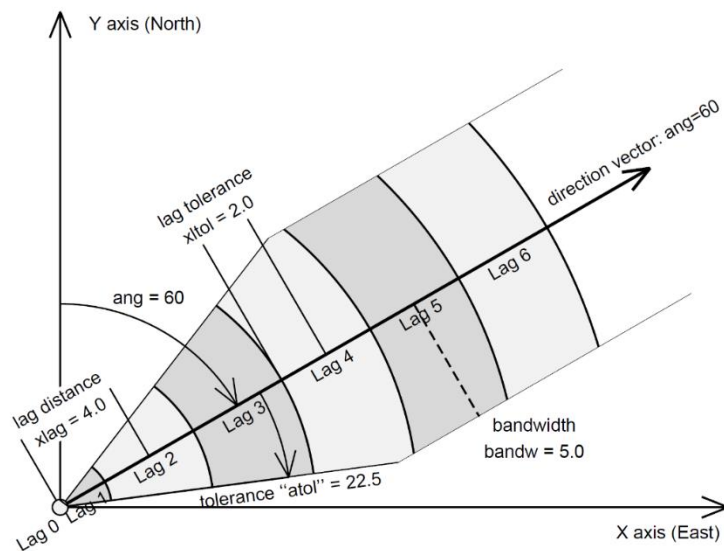


Figure 2-2: Data pairing search template (Deutsch & Journel, 1998)

This search template makes use of the following:

- Distance tolerance
- Azimuth direction
- Azimuth tolerance
- Dip tolerance

- Vertical bandwidth, and
- Horizontal bandwidth (Deutsch & Journel, 1998)

2.2.1.1 Kriging

Kriging is an estimation method for determining values at an unknown location. It is referred to as the best linear and unbiased estimator because it minimises the estimation variance ($\varepsilon = Z_{true} - Z_{kriging}$) and ensures that the expectation of the estimation error is equal to zero [$E(\varepsilon) = E(Z_{true} - Z_{kriging}) = 0$].

Kriging outperforms conventional estimations such as classic polygonal and inverse distance weighting methods because it considers the following:

- Position of the samples in a deposit.
- Distance between samples and point or block to be estimated.
- Spatial continuity of the variable under consideration.
- Number of samples and quality of data at each point (Armstrong, 1998).

A kriging estimate is obtained by assigning optimal weights to data in a given search neighborhood (Vann et al., 2003). Optimal weights are assigned according to distance, sample configuration and spatial continuity captured in the variogram. The estimation variance is an indication of how good the estimate is and is independent of the data values.

OK (Matheron, 1963), is not the only kriging method available but is the most relevant for this Project. Here, the kriging estimate is obtained by assigning optimal weights to data in a given search neighbourhood. Optimal weights in each method are assigned according to distance, sample configuration and spatial continuity captured in the variogram. The estimation variance is an indication of how good the estimate is and is independent of the data values.

The OK estimate and variance are given by the following formula:

$$Z_{OK} = \sum_i \lambda_i Z_i \quad (2.1)$$

$$\sigma_{OK}^2 = \sigma^2 - \left(\sum_{i=1}^n \lambda_i C_{iUL} + \mu \right) \quad (2.2)$$

From equation (2.1), the OK estimate (Z_{OK}) is the weighted sum of the observed values (Z_i), where weights (λ_i) are calculated to minimise the estimation variance and ensure unbiasedness.

The OK variance (σ_{OK}^2) in equation (2.2) is the estimation variance that reflects the uncertainty of the prediction. It dependant on the overall variance of the data (σ^2), the spatial covariance between the known and unknown locations (C_{iUL}) and the lagrange multiplier (μ).

With OK, to ensure that the minimum estimation variance is obtained, the sum of the weights is constrained to 1. This is done through the introduction of a Lagrangian multiplier. A global (prior mean) is not required, therefore, Z_{OK} is estimated locally using the available data. The assumption of stationarity, to the local neighborhood of the location being estimated is relaxed with OK.

2.2.2 Machine Learning

To the layman, the terms: ML, artificial intelligence (AI), deep learning and neural networks are all used interchangeably without any knowledge of the subtle distinctions between them (Janiesch et al., 2021). Such distinctions are captured perfectly by the Euler diagram in Figure 2-3 which shows that ML, neural networks, and deep learning are all sub-fields of AI.

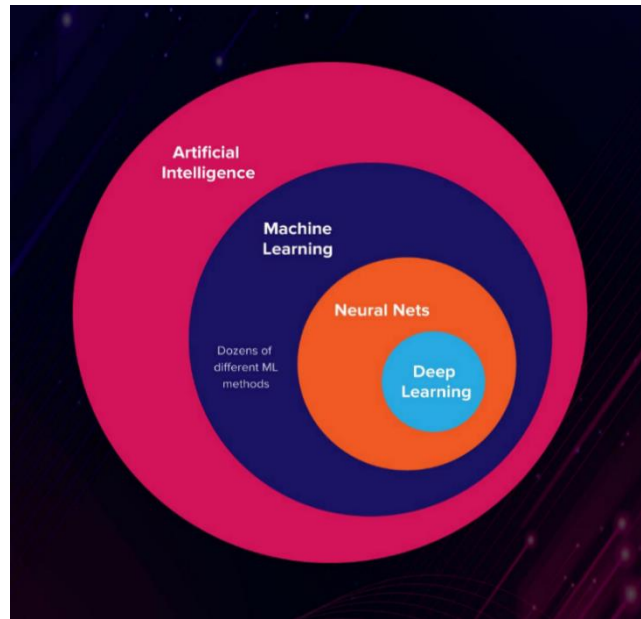


Figure 2-3: Euler diagram of Artificial intelligence (AI) (Gavrilova, 2020)

AI refers to any technique that enables computers to mimic human behaviour and replicate or outperform human decision-making to solve intricate tasks autonomously or with minimal human intervention (Janiesch et al., 2021). Simply put, ML is one such technique, which can be distinguished between three types, namely:

- supervised
- unsupervised and,
- reinforcement learning.

Supervised learning is a type of ML in which an algorithm is trained on labelled data. The goal of supervised learning is to learn the general rule or function that can accurately map data to the correct output based on the labelled example (Samson, 2020). Supervised learning algorithms fall into two categories: classification and regression. With classification, the algorithm must conclude from observed values and determine what category new observations belong to.

In regression, the algorithm must estimate and understand the relationship among variables. Forecasting algorithms make predictions based on past and present data, which in most instances are required in studying trends.

With unsupervised learning, ML algorithms study data to identify patterns without any human intervention. Through the algorithm, the machine determines correlations and relationships by analysing the available data. The data goes through a process of either clustering or dimension reduction (Badillo et al., 2020).

At the bottom of the taxonomy in Figure 2-4, the various algorithms associated with the sub-group falling under supervised and unsupervised learning are shown. As grade estimation is a regression/classification problem the focus will be on algorithms within the supervised learning sub-group.

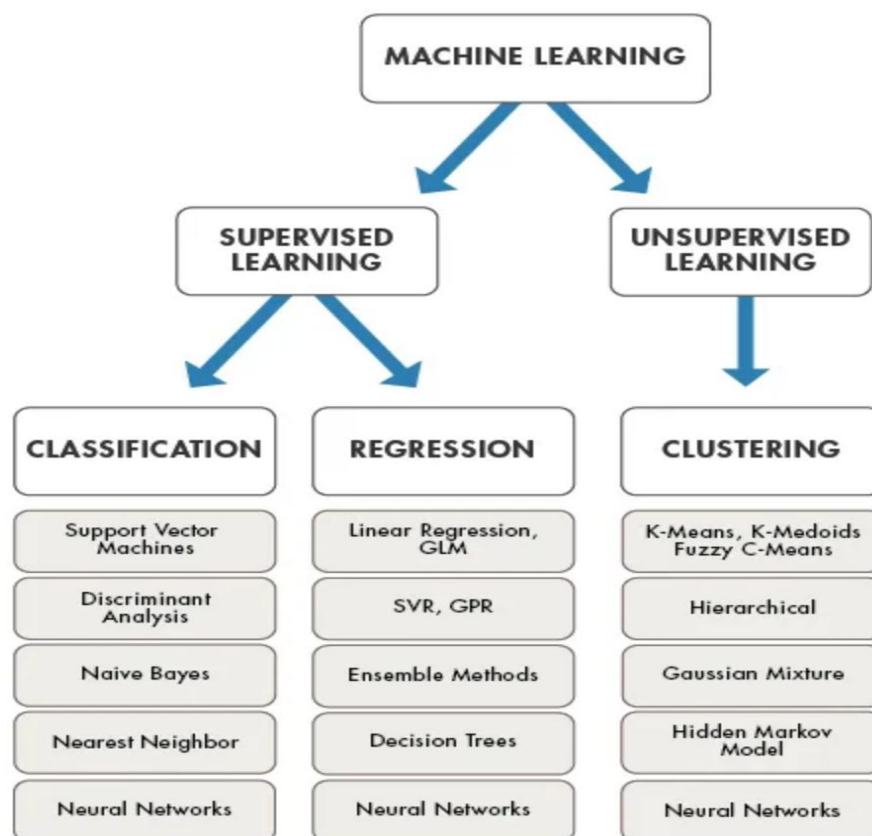


Figure 2-4: Taxonomy of Machine Learning (ML) (Bhattacharjee, 2017)

However, it is important to recognise ML is perceived differently depending on the academic or professional lens through which it is viewed. There are two primary schools of thought regarding the origin and classification of machine learning:

I. Machine Learning as a Branch of Statistics:

From a statistical perspective, ML is often considered an extension of traditional statistical methods, where the emphasis is on understanding data-generating processes, model assumptions, and inference. This view aligns ML closely with "statistical learning," a field grounded in probability theory and linear algebra. Statisticians focus on model interpretability, robustness, and the theoretical underpinnings of algorithms, viewing ML as a natural evolution of statistical techniques (Hastie et al., 2009).

II. Machine Learning as a Branch of Computer Science:

In contrast, computer scientists see ML as a subset of AI, concerned with building systems that can autonomously learn from and make data-driven decisions (Bishop, 2007). The focus here is on algorithms, computational efficiency, and the scalability of models. ML in this context is about creating practical systems capable of handling large datasets and complex tasks, often emphasising performance rather than interpretability.

These differing perspectives highlight the versatility of machine learning, as it serves both as a tool for data-driven inference in statistics and as a foundational component of intelligent systems in computer science.

2.2.2.1 Linear Regression

Regression is arguably one of the most widely used statistical technique; often used for investigating and modelling relationships between variables (Montgomery, et al., 2012).

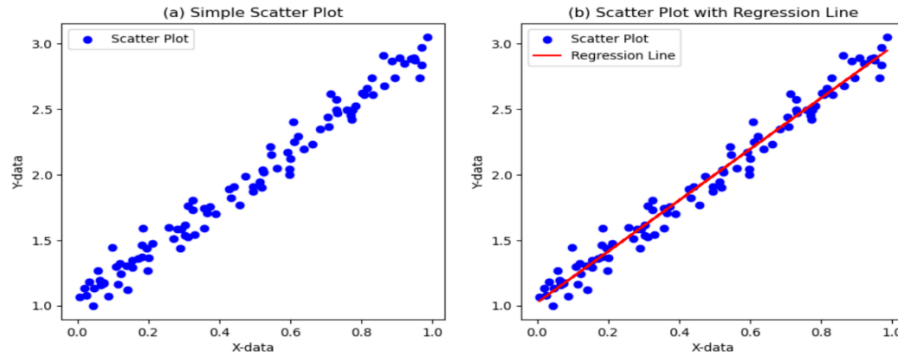


Figure 2-5: (a) Scatter plot of X and Y data (b) Straight line relationship between X and Y

The scatter plot in figure 2-5a shows the relationship between X-data and Y-data. LR, aims to predict data through a function that minimises the distance between itself, and all the data points shown by the regression line in figure 2-5b.

The function explaining the relationship between the x and y data points can be written as:

$$y = \beta_0 + \beta_1 x \quad (2.3)$$

Where β_0 is the intercept and β_1 is the slope. From figure 2-5b it can be observed that the data points do not fall exactly on the regression line. This therefore means that equation 2.3 should be modified to account for this. Let ε be an error representing the difference between the observed value y and the regression line ($\beta_0 + \beta_1 x$). The linear regression model is given by the equation:

$$y = \beta_0 + \beta_1 x + \varepsilon \quad (2.4)$$

The error ε is a statistical error that accounts for the failure of the model to fit the data exactly. The x and y variable is termed the predictor and response variable (Montgomery et al., 2012). Equation 2.4 only involves one predictor and is referred to as a simple linear regression model. However, it may happen that the response variable y is related to more than one predictor, $x_1, x_2, \dots, x_k$, so that:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_k x_k + \varepsilon \quad (2.5)$$

Equation 2.5 is referred to as the multiple linear regression model. The term 'linear' is used to indicate that the model is linear in parameters $\beta_0, \beta_1, \dots, \beta_k$ and not because y is a linear function of the x 's (Montgomery et al., 2012).

2.2.2.2 Train-test Split

As LR is the easiest ML algorithm to conceptualise, it will be used as a reference to understand fundamental ML concepts before proceeding with the breakdown of the remaining four algorithms, SVM, DT, RF and k NN.

Table 2-1 contains random coal data with predictor and response features which can also be referred to as variables. CV will be the response variable (y) which is also the variable we are trying to predict. X, Y, Ash, and Sulphur will be the predictor variables (x) which are the variables used to predict the response variable.

Table 2-1: Example of coal data with predictor and response features

	X	Y	Thickness	CV	Ash	Sulphur
0	9500	12600	1.84	22.26	19.29	0.83
1	9650	12600	1.73	21.36	19.97	0.78
2	9950	12600	1.65	22.34	19.09	1.03
3	10100	12600	2.22	22.19	21.96	1.11
4	10250	12600	2.54	22.43	18.55	1.03
5	10550	12600	2.65	24.07	17.63	1.24
6	10850	12600	2.84	22.11	18.65	0.86
7	9500	12750	2.18	20.90	20.59	0.99
8	9650	12750	2.08	21.70	20.49	0.89
9	9800	12750	1.47	20.83	20.74	1.07

Each column in table 2-1 representing x and y variables can be considered as vectors which can be put in a ML notation. Given N items of data x_i along with response outcomes y_i , where i is the index from 1 to N . The model which is represented by the function $f(x, \theta)$ where θ is the model parameters. The goal in ML is to iterate towards the parameters θ that gives the predicted outcomes $\hat{y}_i = f(x_i, \theta)$ are as close to y_i as possible. The model can now be used to make predictions with new and unseen data (Nichols et al., 2018).

Before the model is developed the data is split into two groups, namely the training and the test dataset. The training dataset is used to find the parameters that produce the model results $\hat{y}_i$ closest to y_i . The test data is used to assess the performance of the model and is not used to influence the model parameters (Nichols et al., 2018).

2.2.2.3 k Nearest Neighbour

The k NN algorithm does not have a single inventor or a definitive "first" publication, as it evolved over time from the broader concept of nearest neighbour (NN) method. The work of Cover & Hart (1967) on NN was highly influential in

formalising and analysing its performance as well as properties in the context of pattern classification.

The NN decision rule classifies an unclassified sample point based on the classification of its closest neighbour from a set of previously classified points. This rule looks at only the single nearest neighbour to the query point (point of interest), regardless of how many other neighbours are close by or how the distribution of neighbours might influence the classification. The decision is entirely based on the most proximal point (Cover & Hart, 1967).

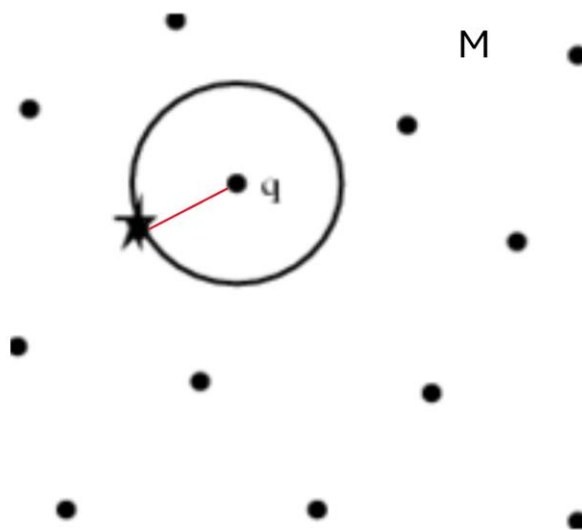


Figure 2-6: An example of the NN search problem

Figure 2-6 demonstrates the concept of the NN algorithm in a 2-dimensional (2d) space. In a collection of data points M for a query point q , the nearest neighbor is the data point M_i within M that is closest to q according to a defined distance $dist(q, M_i)$. This distance measure can be any metric that defines the "closeness" of data points, such as the Euclidean distance, Manhattan distance, Minkowski distance, or any other appropriate measure for the data in question (Sammut & Webb, 2010).

Within the ML discipline, the k NN algorithm is often referred to as an instance-based or lazy learning algorithm. Instance-based because it operates directly on the instances (data points) themselves during the classification or regression

process. Unlike other algorithms that build a generalisable model from the training data, k NN compares new instances (query points) to the specific examples in the training dataset to make predictions. This approach requires the algorithm to retain the entire dataset, and each instance contributes to the decision-making when querying the model (Venkatesh et al., 2017).

Also termed "lazy" because it does not learn a discriminative function from the training data that can be used to make predictions. Instead, it defers computation until classification is performed. This means that k NN does not build an internal model during the training phase; rather, it stores the dataset and waits until a prediction is requested (Venkatesh et al., 2017). The learning process happens at the time of prediction by searching for the k nearest neighbours among the training instances, calculating the distances between the instances, and performing a majority vote (for classification) or averaging (for regression) among the k nearest neighbours.

The number of neighbours necessary for consideration while predicting the class of a new point is denoted by the letter k in k NN. Considering the k nearest neighbours to an unclassified sample point rather of just one, the k NN algorithm expands on the concept of the NN rule.

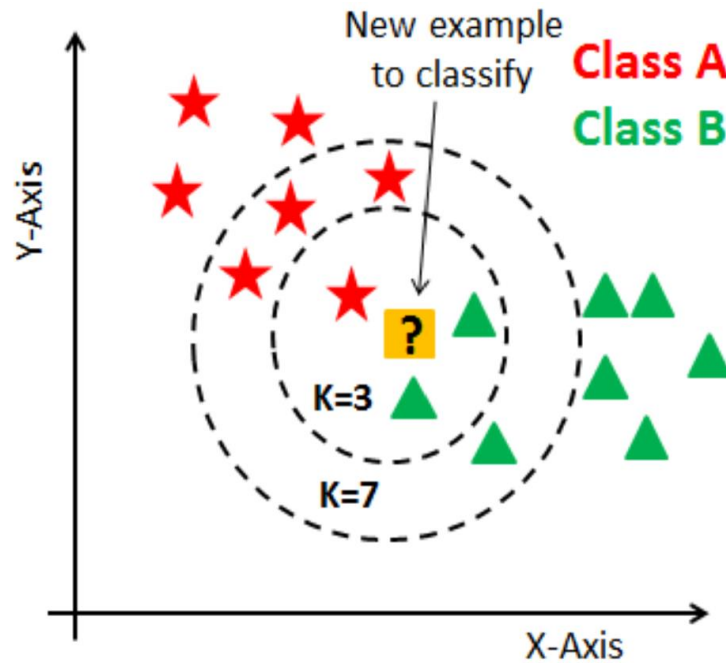


Figure 2-7: k NN classification example when $k = 3$ and $k = 7$ (Gokte, 2020)

The scenario in figure 2-7 explains the k NN algorithm for a classification problem. It shows a 2d scatter plot with two different classes of data points: Class A (stars) and Class B (triangles). The plot includes a new example (unclassified sample point) that needs to be classified. The algorithm operates by finding the k closest data points (neighbours) to the new example. The distance between points is usually calculated using Euclidean distance in a two-dimensional space. For $k=3$, the three closest neighbours to the new example all belong to Class A. Therefore, by majority vote, the new example would be classified as Class A. When $k=7$, there are four stars and three triangles within the circle. With most stars, the new example would be classified as Class A.

Figure 2-7 also shows how the choice of k influences classification. A smaller k represents more local information, which can be sensitive to noise and outliers, whereas a bigger k includes more neighbours, which can smooth out anomalies but may obscure decision boundaries if k is too large. The decision of which class the new example belongs shows that with different values of k , the classification boundary can change, further demonstrating the importance of choosing the right " k " for the given data.

While the process of finding the nearest neighbours is the same in both classification and regression, the method of aggregating the neighbours' information differs to suit the nature of the prediction being made, categorical labels for classification and continuous outcomes for regression.

To represent k NN mathematically, of a given vector x , its nearest neighbour (closest vector to x) x_i , is identified. To index neighbours $NN(x, j)$ is used to denote the index of the j^{th} nearest neighbour to the vector x . Keeping track of indices is important because it helps reference the specific data points in the dataset that are the closest to x . The k NN estimate of the regression function is then:

$$u(x) = \frac{1}{k} \sum_{j=1}^k y_{NN}(x, j) \quad (2.6)$$

Where $u(x)$ is the average of the response values y of the k nearest neighbours (Yunsheng et al., 2017).

2.2.2.4 Support Vector Machines

A hyperplane can be thought of as a flat sheet or line in space. In ML it is used to separate features (Summit & Webb, 2011). For example, grade control; distinguishing between two classes; low grade and high grade. A hyperplane resembles a boundary that helps decide which class a feature belongs to. It is also a crucial concept in an algorithm like SVM.

SVM was developed in three main steps. First, in 1963 Vapnik and Lerner proposed that the best hyperplane to separate training examples is the one with the widest margin. Secondly, from the 1960s to the 1990s, Vapnik and Chervonenkis developed a theory justifying the maximum margin principle statistically. Thirdly, Boser, Guyon and Vapnik in 1992 incorporated kernel functions into the maximum margin models. This resulted in a formulation close to current SV. A kernel is a function that transforms the data into a higher-dimensional space, where it becomes easier to find a hyperplane that separates the classes. The key idea behind kernels is that they allow SVMs to perform complex

classification tasks without explicitly transforming the dataset into a higher-dimensional space. (Summit & Webb, 2011).

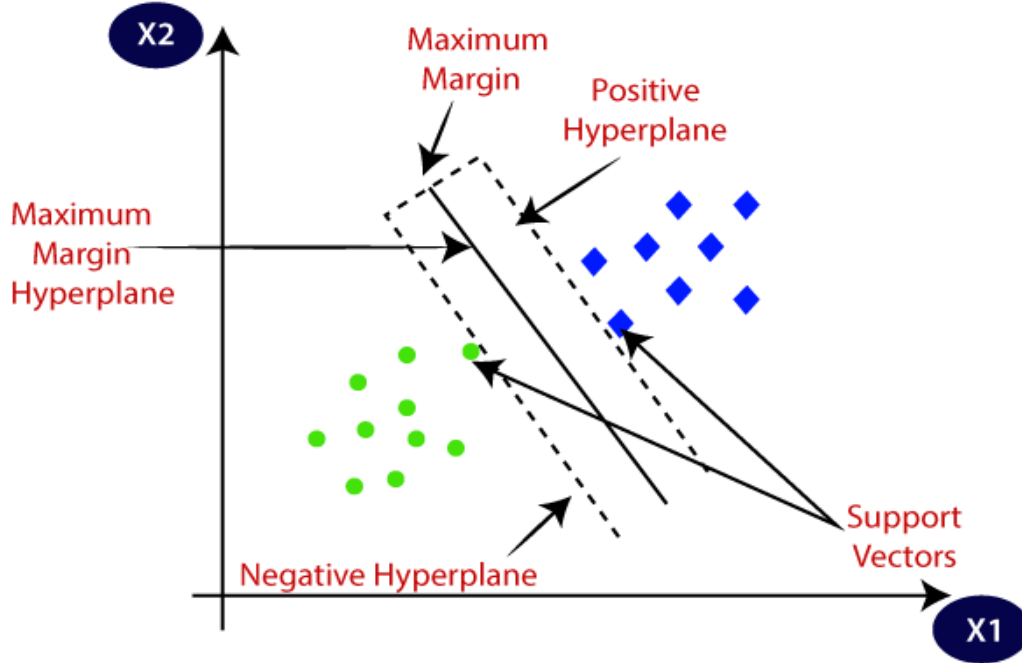


Figure 2-8: Illustration of the SVM algorithm (Saini, 2023)

Put simply, SVM is a supervised machine learning algorithm with the objective of finding a hyperplane that best separates classes (Caraka et al., 2020). The green and blue shapes seen in figure 2-8 represent two different classes. The solid line dividing the two classes is the maximum margin hyperplane, which best separates the data. “Best” because the distance between the support vectors and the hyperplane are maximal. Support vectors are the data points that are the closest to the hyperplane, subsequently, affecting its position as well.

The SVM model aims to find the hyperplane which can be represented mathematically by the equation:

$$f(x) = x^T \beta + \beta_0 \quad (2.7)$$

Where x is a feature vector, β , a weight coefficient and β_0 is a bias term. The hyperplane ($f(x)$) is based on the sign of the distance from the support vectors. The sign function will be positive on the one side of the hyperplane.

and negative on the other, 0 when falling on the hyperplane, given by the equation:

$$G(x) = \text{sign}(f(x)) \quad (2.8)$$

SVM is a constrained optimisation problem in that the goal is not only to find the hyperplane but also to minimise the margin between classes. This constraint is represented by:

$$y_i(x_i^T \beta + \beta_0) \geq 0 \quad (2.9)$$

Where y_i is the response output with classes -1 and 1. Equation 2.9 ensures that all data of each class must fall on the correct side of the hyperplane. In real-world scenarios, data may not be perfectly separable and, may result in misclassification. Referring to figure 2-8 this means green data points classified as blue and vice versa.

SVM introduces the concept of soft margins, by allowing for some misclassification. The optimisation problem now includes a slack variable (ξ_i) to represent the amount by which a data point is allowed to violate the margin or be on the wrong side of the hyperplane. Equation 2.9 now becomes:

$$y_i(x_i^T \beta + \beta_0) \geq M - \xi_i \quad (2.10)$$

Where M is the margin between the hyperplane. Solving for SVMs using quadratic programming with linear inequality constraints, we can express equation 2.10 as:

$$\text{minimise}_{\beta, \beta_0} \left(\frac{1}{2M^2} + C \sum_{i=1}^N \xi_i \right) \quad (2.11)$$

$$\text{Subject to } \xi_i \geq 0, \quad y_i(x_i^T \beta + \beta_0) \geq M - \xi_i \quad (2.12)$$

In equation 2.12, the parameters of the plane maximise the margin while minimising the error. C is a constant hyperparameter that controls the trade-off between maximising the margin and minimizing the classification error (Caraka et al., 2020).

It is important to note that the margin optimisation problem is transformed into an inner product problem using the Lagrange multipliers method. This transformation is crucial because it enables the introduction of kernel functions, allowing the SVM algorithm to perform efficiently in higher-dimensional spaces without explicitly mapping the data into those dimensions. This kernel trick effectively makes SVM a "kernelizable" method, which is fundamental to its ability to handle complex, non-linear decision boundaries (Cortes & Vapnik, 1995).

2.2.2.4 Decision Trees

DT is a supervised ML algorithm used for both regression and classification. It is a predictive modelling tool that visually represents a decision-making process (Jafrasteh et al., 2018).

From figure 2-9, the base of the DT is the root node. From the root node, a series of decision nodes flow. These decision nodes represent decisions to be made and the leaf or terminal nodes are the consequences of these decisions. The branch represents the subsection(s) of the tree.

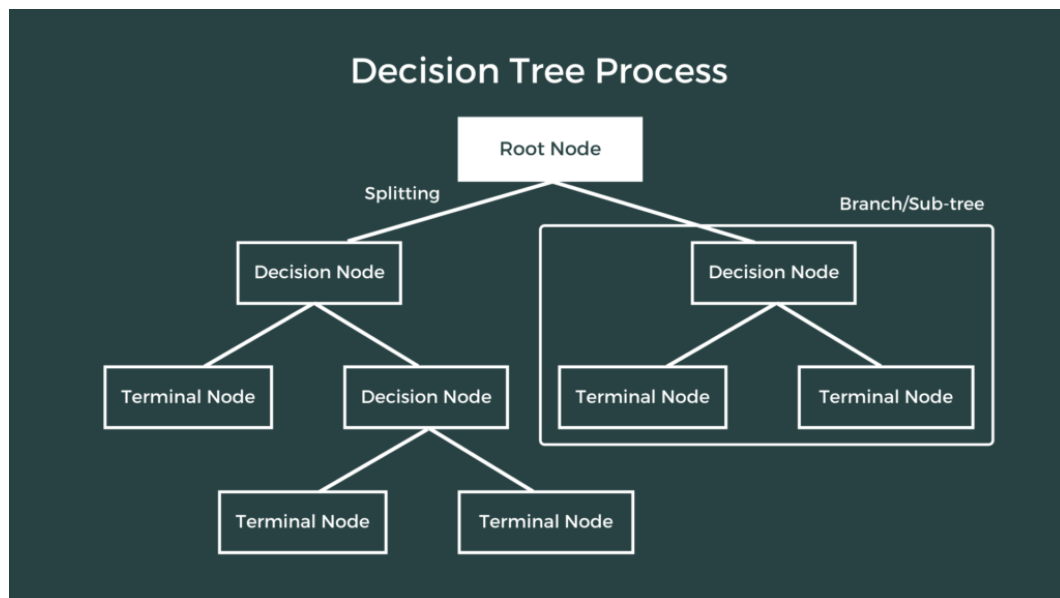


Figure 2-9: The Decision Tree process (The click Reader, 2021)

A regression tree divides the predictor space into M units $\{R_1, R_2, \dots, R_m\}$. Each leaf node of the regression tree corresponds to a unit which has a fixed value c_m .

This value is the predicted outcome for all data points that fall within region R_m . The decision tree model can be expressed as:

$$T(x) = \sum_{m=1}^M c_m I(x \in R_m) \quad (2.13)$$

The index function $I(x \in R_m)$ evaluates to 1 when the regression tree identifies x as belonging to R_m , and 0 otherwise. The primary objective in constructing a regression tree is to minimise the square error of the dataset D , a collection of all data points $\{(x_i, y_i)\}$, expressed as:

$$\sum_{x_i \in D} (y_i - T(x_i))^2 \quad (2.14)$$

The squared difference between the actual target value y_i and the predicted value $T(x_i)$ measures the accuracy of the prediction.

To begin, the suitable division is chosen by selecting the j^{th} dimension of the variable x and its corresponding threshold s . At each decision node, the node partitions into two regions:

$$R_1(j, s) = \{x | x[j] \leq s\} \quad (2.15)$$

And

$$R_2(j, s) = \{x | x[j] > s\} \quad (2.16)$$

Regions R_1 and R_2 given by equations 2.15 and 2.16 are the two subsets of data points resulting from the split. R_1 includes points where the j^{th} feature is less than or equal to s , and R_2 includes points where the j^{th} feature is greater than s .

The optimal segmentation feature j and the segmentation threshold s at the node are given by:

$$\min_{j,s} \left[\min_{c_1} \sum_{x_i \in R_{1(j,s)}} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_{2(j,s)}} (y_i - c_2)^2 \right] \quad (2.17)$$

The optimal j and s for a node are determined by minimising the variance of the two sub-sample sets resulting from the partitioning. Equation 2.17 formalises the process where c_1 and c_2 are the mean value of the samples (Sun et al., 2021).

Subsequently, the output value for regression tree is determined. For each sub-region R_m (representing a leaf node), the corresponding c_m value is directly chosen as the category mean values, i.e.,

$$c_m = \operatorname{argmin}_c \sum_{x_i \in R_m} (y_i - c)^2 = \operatorname{average}(y_i | x_i \in R_m) \quad (2.18)$$

2.2.2.5 Random Forrest Tree

RF is an ensemble learning method that builds upon the DT algorithm to create a more robust and accurate predictive model. While a single DT is a powerful tool for classification and regression tasks, it is prone to overfitting. RFs mitigate this issue by constructing a multitude of decision trees during training and aggregating their outputs to improve prediction accuracy and generalisation (Hastie, et al., 2009) (See figure 2-10).

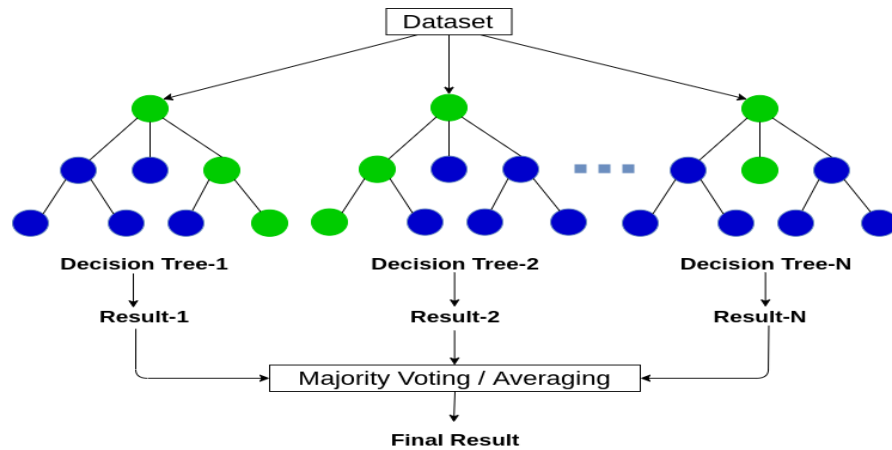


Figure 2-10: Illustration of a Random Forrest Tree algorithm (Sharma, 2023)

The key processes that differentiate Random Forests from single Decision Trees are:

I. Bootstrap Sampling:

During the training phase, RFs employ a technique known as bootstrap sampling (or bagging). This involves randomly sampling the training data with replacement to create multiple subsets, each of which will be used to train a separate DT. This means that some data points may appear multiple times in a subset, while others may not appear at all. This diversity in training subsets helps to reduce the variance of the model (Hastie, et al., 2009).

II. Random Feature Selection:

In addition to varying the training data, RFs introduce another layer of randomness by selecting a random subset of features at each split in the DTs. Typically, when splitting a node, only a random subset of the total features is considered, rather than evaluating all possible features. This randomness ensures that the trees are less correlated with each other, which enhances the overall predictive performance when the trees' outputs are combined (Liaw & Wiener, 2002).

III. Aggregation of Predictions:

Once all the DTs have been trained, the RF makes predictions by aggregating the results of each tree. For classification tasks, this aggregation is typically done through a majority vote—each tree "votes" for a class label, and the class with the most votes is selected as the final prediction. For regression tasks, the predictions of all trees are averaged to produce the final output (Zhang & Ma, 2012).

By averaging the predictions of many DTs, RF reduces the risk of overfitting that is commonly associated with individual DTs. Overfitting occurs when a model is too closely fitted to the training data, capturing noise rather than the underlying data patterns. The diversity among the trees in RF improves the model's ability to accurately detect the real patterns and relationships within the data.

2.3 Key Debates and Controversies

ML is by no means new in the mining industry, with papers relating to grade estimations being published in the early 1990s (Dumakor-Dupey & Arya, 2021). ML makes use of computer algorithms to analyse, learn and predict outcomes. Its application to geostatistical data without the need for established assumptions was exhibited in the work done by Samson (2020) through the use of an elliptical radial basis function network (ERBFN). ERBFN is an ML method that does not require the assumption of stationarity and only requires spatial coordinates as an input feature for known data points.

Grade estimation is an aspect of mineral estimation which over the years has seen the popularisation in the use of ML. Figure 2-11 shows the number of academic research conducted with regard to mineral estimation over 28 years. Figure 2-6 further indicates two cycles of increase and decreases in ML research between 2007-2014 and 2015-2021.

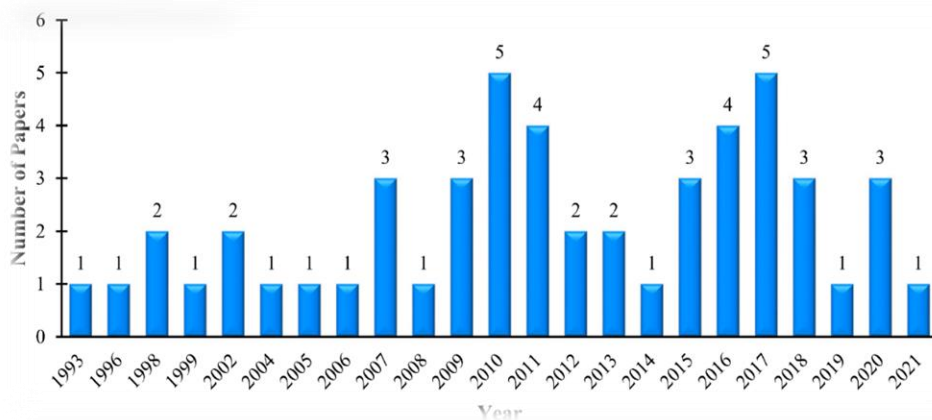


Figure 2-11: Academic Papers written on ML in mineral resource estimation from 1993-2021 (Dumakor-Dupey & Arya, 2021)

Additionally, the most popular algorithms over these 28 years are shown in Figure 2-12. Artificial neural network accounts for 57% of the preferred machine learning algorithms used in the research papers.

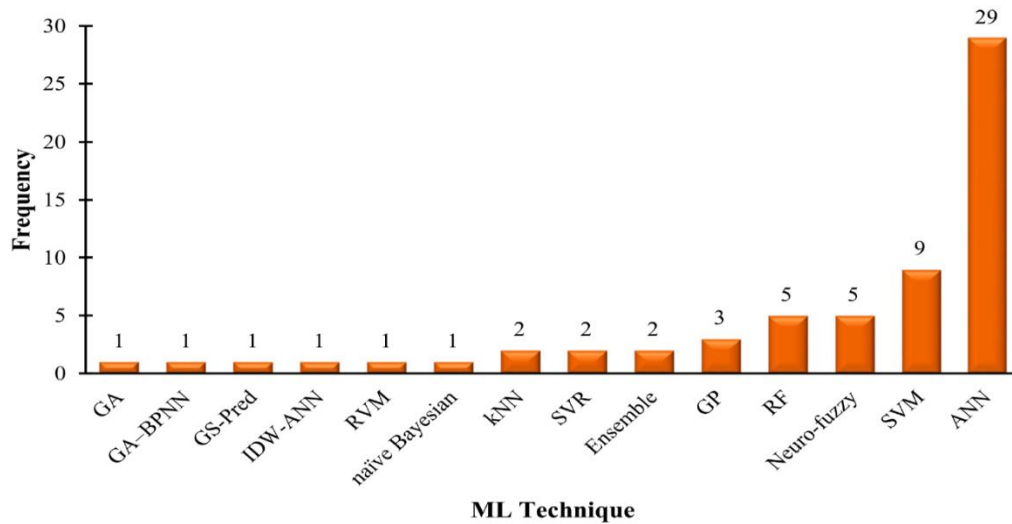


Figure 2-12: Most popular ML algorithm used in grade estimation research
(Dumakor-Dupey & Arya, 2021)

According to Dumakor-Dupey & Arya (2021), ML algorithms can produce superior results for classification and regression problems than geostatistical techniques. This was proven by Da Silva *et al.*, 2020 when comparing various grade control methods. In this study, misclassification was reduced by 5.7% relative to a simulation-based approach when using ML. As grade control is mainly a classification problem, the use of ML would be fitting. Another reason for the increase in the use of ML is that ML algorithms can accommodate a combination of different geospatial parameters for grade prediction and even ore classification (Dumakor-Dupey & Arya, 2021). ML methods are not without their shortcomings, as a single ML algorithm may not suffice in producing the best estimates. For instance, consider the case where coordinates are used as a predictor and the fact that ML methods do not rely on spatial dependence between values can lead to artefacts along the coordinate's direction (Erten *et al.*, 2021). A solution to this problem is the use of an ensemble method, which is simply the use of multiple machine learning algorithms to obtain a better predictive performance (Erten *et al.*, 2021).

2.4 Chapter Summary

The literature review provides a comprehensive examination of the application of ML and geostatistical methods in making predictions or estimates based on data, highlighting their fundamental principles, applications, key debates, and controversies within the field of geostatistics, particularly in mineral and grade estimation.

Geostatistics is defined as a branch of statistics focusing on the analysis and modelling of spatial data. The concept of regionalized variables is central to geostatistics, where a variable's spatial distribution is characterized both by its randomness and structured aspects. This distribution is analytically represented through semi-variograms, which graphically display the spatial continuity of variables over distance. Key elements of semi-variograms include the sill, range, and nugget effect, which describe the variance and correlation of data in a spatial context. Kriging, a prominent geostatistical method, is highlighted for its precision in estimating values at unknown locations by considering the spatial continuity and sample distribution, making it superior to other conventional estimation methods.

ML, on the other hand, is presented as a subset of artificial AI that includes neural networks and deep learning. ML encompasses various algorithms categorized into supervised, unsupervised, and reinforcement learning. The literature review emphasizes supervised learning, particularly its use in classification and regression problems relevant to grade estimation. Key ML algorithms such as LR, DT, SVM, RF and k NN are discussed, with explanations on their operational mechanisms, advantages, and applications in predicting outcomes based on input features.

In conclusion, the literature review explains the distinct characteristics, advantages, and limitations of ML and geostatistical methods in spatial data analysis and prediction. While geostatistics offers a robust framework for understanding spatial variability and continuity, ML provides versatile and often more accurate predictive models classification and regression problems. The ongoing development and comparison of these methodologies signify a rich area of research and application, particularly in the field of mineral estimation and mining.

3 RESEARCH METHODOLOGY

3.1 Introduction

The research approach used is directly related to the objectives of this project. This process provided the framework for how ML models compared with OK regarding grade prediction accuracy. Additionally, it aided determining which ML algorithm performed the best.

3.2 Research Design

Both the ML and OK workflow implementation was facilitated by opensource Python libraries.

3.2.1 Ordinary Kriging implementation

For this research, the *GeostatsPy* package was utilised to carry out the OK estimation process. This package is an invaluable resource for geoscientists and data analysts working with spatial data, as it incorporates functions from the *GSLIB* (Geostatistical Library) into the Python environment. *GSLIB* itself is a robust and practical set of codes designed for constructing spatial modeling workflows. Developed by Michael J. Pyrcz, a professor at the University of Texas at Austin, *GeostatsPy* aims to facilitate the application of geostatistical methods directly within Python, simplifying the process for researchers and students engaged in these techniques. (Pyrcz et al., 2021).

This research utilised the two primary components of the *GeostatsPy* package. The first component, *geostatspy.geostats*, includes Python re-implementations of *GSLIB* functions, covering variogram analysis, distribution transformations, and spatial estimation and simulation methods. The second component, *geostatspy.GSLIB*, provides low-tech wrappers for *GSLIB*'s visualizations and numerical methods (Pyrcz, et al., 2021).

Using the *GeostatsPy* package, OK is applied to mineral grades using the spatial coordinates of sample points as the primary input to obtain point and block estimates.

3.2.2 Machine Learning implementation

The library used to execute the ML spatial prediction workflow is Sci-kit learn. More details about this package will be mentioned in chapter 4. Through Sci-kit learn, ML models referred to in chapter 2 are selected. To make predictions at point locations, the data used for the research was split into distinct training and testing sets from which the Sci-kit learn prediction convention was used to make predictions.

Producing block estimates required the use of a contemporary approach used by Nwalia et al., 2024. The authors highlighted the use of a block discretisation technique (microblocking), which involves predicting values at a much finer scale (microblocks) and then aggregating these predictions to form larger blocks (macroblocks). This approach is designed to circumvent the challenges associated with the change of support problem inherent in traditional geostatistics.

The approach used in this project generally aligns with the concept of spatial interpolation using machine learning. It effectively demonstrates how to implement a process for generating predictions at a finer scale (discretisation points within blocks) and then aggregating these predictions to a larger scale for comparison or integration with geostatistical models like Kriging (Nwalia et al., 2024).

Here is how the approach used in this research project corresponds to key aspects of the methodology discussed in the paper:

1. **Discretisation of Space:** By defining a grid of discretisation points within each block, the method captures the essence of predicting at a finer scale within the predefined grid dimensions, which is crucial for detailed spatial analysis.
2. **Microblock to Macroblock Aggregation:** The approach calculates mean predictions for each block based on the finer-scale predictions, which corresponds to the aggregation process from microblocks to macroblocks. This step is essential for aligning machine learning predictions with the spatial resolution used in the OK model.
3. **Spatial Extent and Block Size Considerations:** The approach considers the spatial extent of the study area and the size of the blocks, ensuring that the predictions cover the entire area of interest. This is consistent with the need to

match the spatial coverage of ML models with that of traditional geostatistical models.

4. **Implementation of Machine Learning Predictions:** The ML model training or prediction process ensures availability of predictions for the discretisation points. This implementation step is crucial for applying machine learning to spatial prediction tasks.
5. **Adaptation to Spatial Data:** By adjusting the origin of the grid based on an estimation boundary and considering the total number of blocks and discretisation per block, the approach was able to show adaptability to the specific spatial configuration of the dataset.

3.3 Data Collection Process

This process involved assembling a dataset from a mining project that includes mineral grade data along with other relevant geological and geospatial variables. It was necessary to ensure that the dataset is sufficiently large and varied to facilitate a robust analysis. This involved performing preprocessing to handle missing values and ensuring compatibility with both OK and ML models.

3.4 Data Validation Process

The validation process serves to ensure that the data used in this project is reliable, accurate and appropriate. The focus points under this process are missing data, outliers, data integrity, data quality, variable types, and data completeness.

3.5 Data Analysis Approach: Estimation Criteria

Leave-One-Out Cross Validation (LOOCV) for OK was the method used to validate the accuracy and performance of the kriging model. In LOOCV, for a dataset with N the validation process is repeated N times. In each iteration, one observation is removed from the dataset and serves as the validation data, while the remaining $N - 1$ observations are used as the training data.

With ML cross-validation is often used in conjunction with the initial split to ensure that the model's performance is robust and not dependent on a particular way of

splitting the data. The k -fold cross-validation method was used. Here, the training set is split into k smaller sets (or folds). The model is trained on $k - 1$ of these folds and validated on the remaining part. This process is repeated k times, with each of the k folds used exactly once as the validation data. The k results can then be averaged to produce a single estimation of model performance. In making point estimation, after evaluating and selecting the best model through cross-validation, the entire dataset is used to train the model before making predictions. This approach assumes that the more data the model sees, the better it will perform.

Validation error metrics like R-squared, RMSE (Root Mean Square Error), MSE (Mean Square Error), and MAE (Mean Absolute Error) are crucial for assessing the performance of regression models. Each metric offers a different perspective on the error between the predicted values by a model and the actual values observed in the data. The formulations and explanations for each are as follows:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (3.1)$$

R^2 measures the proportion of the variance in the dependent variable that is predictable from the independent variable(s). SS_{res} is the sum of squares of residuals (the differences between observed and predicted values), and SS_{tot} is the total sum of squares (the differences between observed values and the mean of observed values). R-squared values range from 0 to 1, where a higher value indicates a better fit to the data.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3.2)$$

MSE calculates the average of the squares of the errors, that is, the average squared difference between the estimated values $\hat{y}_i$ and actual values y_i . MSE gives a higher weight to larger errors due to the squaring of each term, which means it is useful when large errors are particularly undesirable.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (3.3)$$

RMSE is the square root of *MSE*. It measures the standard deviation of the residuals or predictions errors. By taking the square root of *MSE*, *RMSE* converts the error metric back to the same unit as the target variable, making interpretation easier. Like *MSE*, *RMSE* penalises larger errors more heavily.

$$MAE = \frac{1}{n} \sum_i |y_i - \hat{y}_i| \quad (3.4)$$

MAE calculates the average of the absolute errors between the predicted values and the actual values. Unlike *MSE* and *RMSE*, *MAE* gives a linear weight to all errors, which means it is less influenced by the occasional, but very large errors. It is a straightforward measure of average error magnitude.

Visualisations such as scatterplots, histograms of residuals and swath plots were made to gain additional understanding and perspective of results obtained from the data.

3.6 Chapter Summary

Chapter 3 outlines the approach, design, and analytical methods employed to compare ML models and OK for predicting mineral grades, aiming to identify the superior ML algorithm in terms of prediction accuracy. It begins by establishing the research's foundation; directly linking its methodology to the project's objectives. It sets the stage for a detailed examination of how ML models stack up against OK in grade prediction accuracy and which specific ML algorithm outperforms the rest. A commitment to open-source Python libraries is outlined for implementing both ML and OK workflows, emphasising the accessibility and applicability of the tools used.

It further outlines the implementation of OK and ML for predicting mineral grades. The *GeostatsPy* package is showcased for OK, highlighting its utility in integrating *GSLIB* functions into Python for geoscientific analysis, including variogram analysis

and spatial estimations. It emphasizes the package's role in facilitating research in geostatistics. For ML, Scikit-learn is introduced as the tool adapted for spatial prediction. A microblocking technique for block estimates is introduced, marking a significant advancement in combining ML with spatial interpolation to address geostatistical challenges, thereby enhancing precision in spatial analysis.

This chapter introduces Leave-One-Out Cross Validation (LOOCV) for OK and k -fold cross-validation for ML, explaining their importance in assessing model performance and robustness. It then describes the validation error metrics (R , RMSE, MSE, and MAE) used to evaluate the regression models, providing formulas for each, and explaining their significance in measuring prediction errors. It concludes by acknowledging the role of visualisations in enhancing the understanding of data and model performance, setting the stage for a comprehensive evaluation of ML and OK methodologies in spatial data analysis.

4 DATA FOR THE STUDY

4.1 Introduction

The role of geology in mineral estimation cannot be understated. It provides insights about the characteristics of a mineral deposit (Sinclair & Blackwell, 2002). This implies that lack of geological knowledge would result in inaccurate grade estimations. This chapter will provide an overview of the geological setting of the study area. Exploratory data analysis (EDA) will be performed to provide information on the distribution of the deposit grades. Variography and ML will also be discussed. To preserve confidentiality of the site location, the data has been rotated and translated and, will be referred to as Project X.

4.2 Geological Setting

4.2.1 Regional Geology

The BIC is the largest mafic igneous layered intrusion and is known for hosting the highest concentration of platinum group elements (PGE) in the world (See figure 4-1).

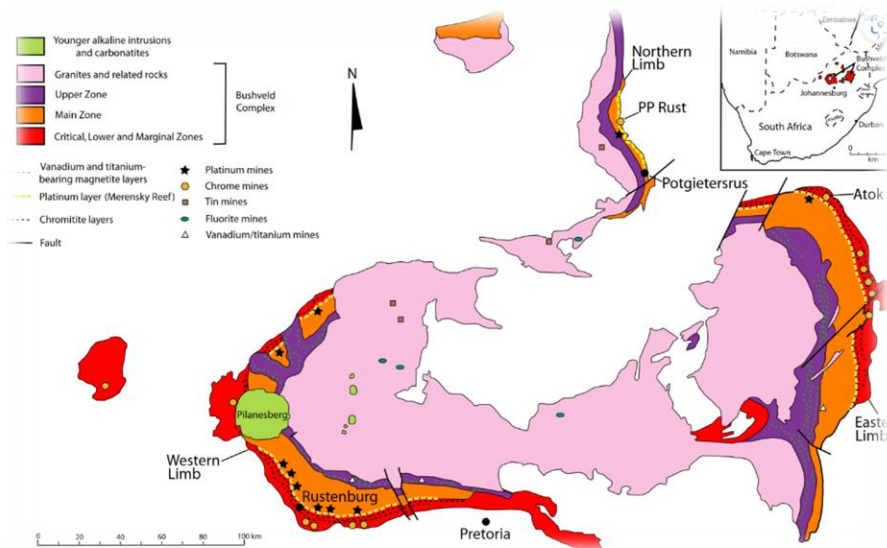


Figure 4-1: Geological map of the BIC (Deseta, 2020)

The outcrops of the mafic rocks of the BIC are exposed in a manner that divides it into three distinct parts known as the eastern, western, and northern limb (Shouwstra & Lee, 2000). The BIC comprises of three main suites namely: Lebowa Granite Suite, Rustenburg Layered Suite (RLS), and Rashoop Granophyre Suite. The RLS is subdivided into five zones named the: marginal, lower, critical, main, and upper zone (See figure 4-2) (Cawthorn & McKenna, 2006). The mineralisation of the PGE deposit occurs in the critical zone with the principal reef type deposit being Merensky Reef and UG2.

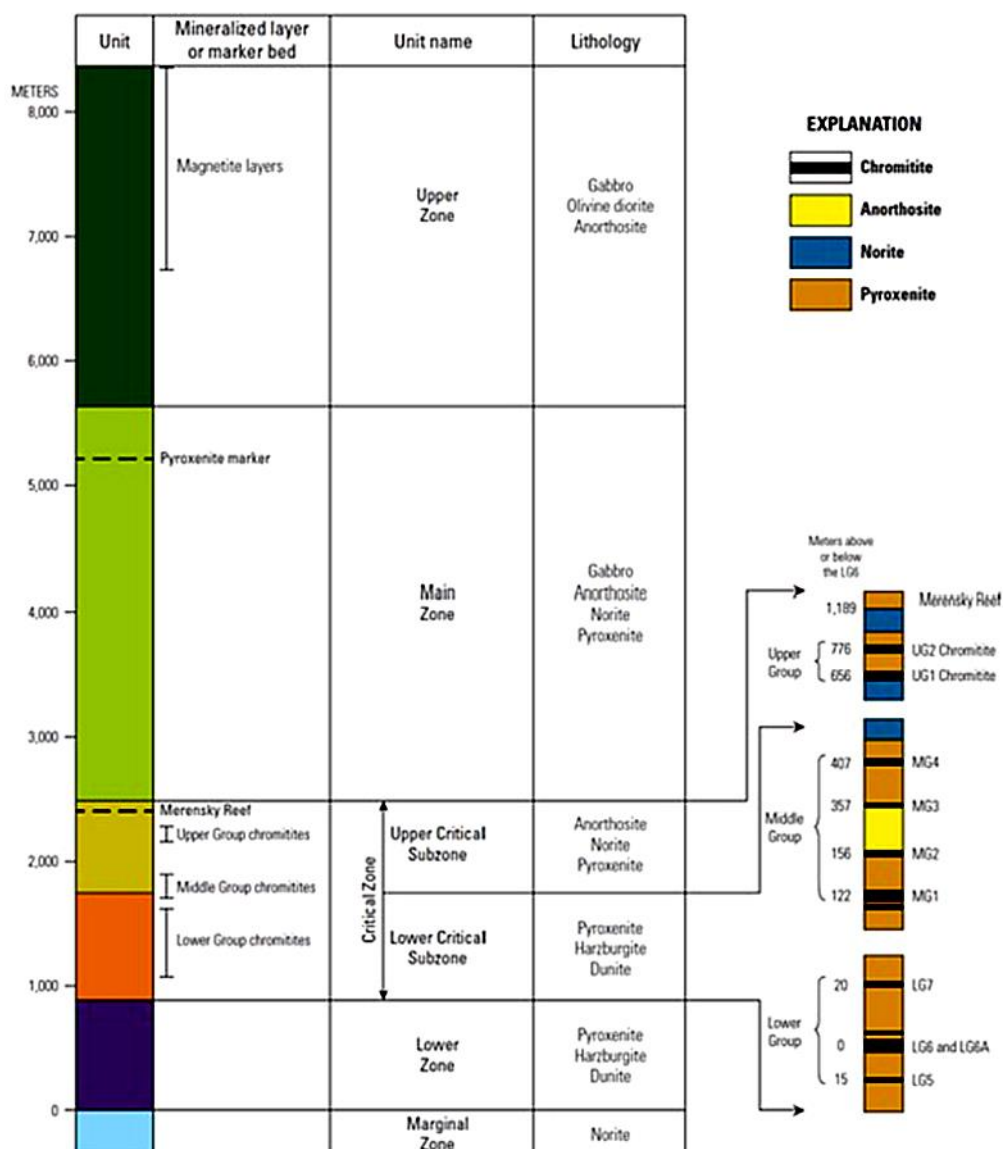


Figure 4-2: Stratigraphic column of the RLS (Deseta, 2020)

4.2.2 Local geology

The UG2 chromitite (Cr), located in the upper critical zone represents the most significant source of PGEs globally. Measuring 0.5-1m in thickness, this reef is found at depths ranging from 15-400m beneath the Merensky reef. The distance between these two reefs expands from east to west. Comprising of two to four smaller Cr reefs, the UG2 has an average Cr composition of 43.5%.

4.3 Project Background

Project X can be seen in figure 4-3; in the drilling programme both the Merensky and UG2 reef were intersected. The project, however, focuses solely on the UG2 reef. The geologic features include, faults, shears, dykes and iron-rich ultramafic pegmatoids (IRUP's) indicated in figure 4-3. Said features were inferred from detailed logging of the borehole core, while others from their surface expression.

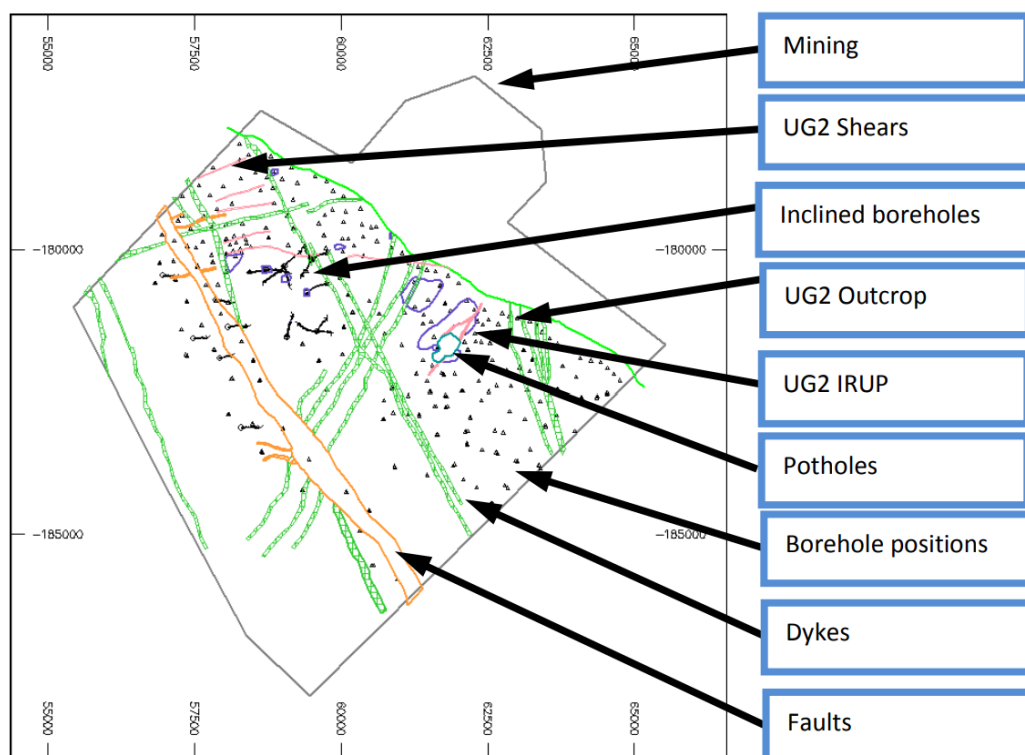


Figure 4-3: Lease area of Project X

4.4 Exploratory Data Analysis

EDA helps to summarise the main characteristics of a given dataset through visual and numerical techniques. According to Sinclair (1998), EDA accounts for more than 50% of the estimation process. Sinclair (1998) further goes on to state the benefits of EDA which are namely:

1. To recognise and eliminate errors
2. Provide enough knowledge about the important sample statistics
3. Understand the relationship between sample characteristics
4. Identify outliers

4.4.1 Project X deposit

The project X deposit consists of 570 boreholes. The estimation domain in the X direction ranges from 56837-64759m, in the Y direction ranges from -185780- -178136m, in the Z direction ranges from 289-1102 meters.

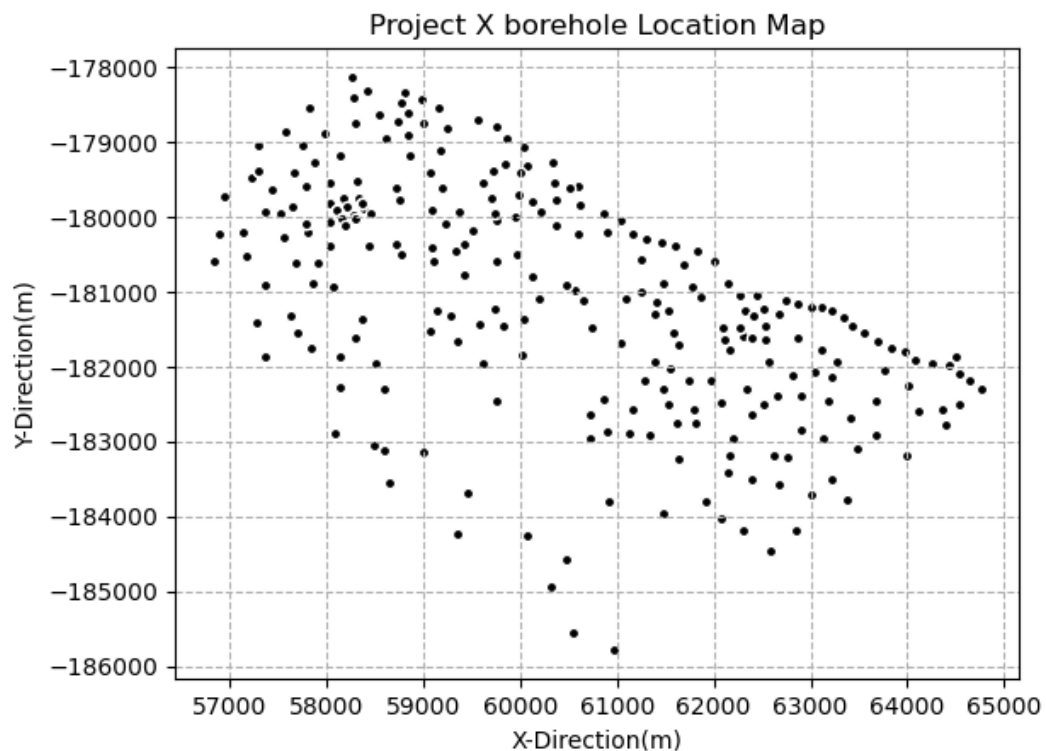


Figure 4-4: Borehole location map of the Project X deposit

Upon inspecting the data for duplicates, six boreholes were found to share the same X and Y coordinate with at least one other borehole (see table 4-1). The boreholes, however, differ in other attributes such as grade (PGE) and elevation (Z). The occurrence of the duplicates could be due to multiple reasons; hence, their handling should take into consideration the context and objective of the analysis.

Table 4-1: Boreholes containing duplicate X, Y coordinates

BHID	X	Y	Z	PGE
WA36D1	62864.15	-181625.45	937.03	7.5552
WA36D2	62864.15	-181625.45	937.01	8.3516
WW20D0	57693.09	-180625.31	627.44	6.1055
WW20D1	57693.09	-180625.31	627.39	7.2038
WW59D0	57295.25	-179397.25	746.56	2.2010
WW59D1	57295.25	-179397.25	746.31	2.4470

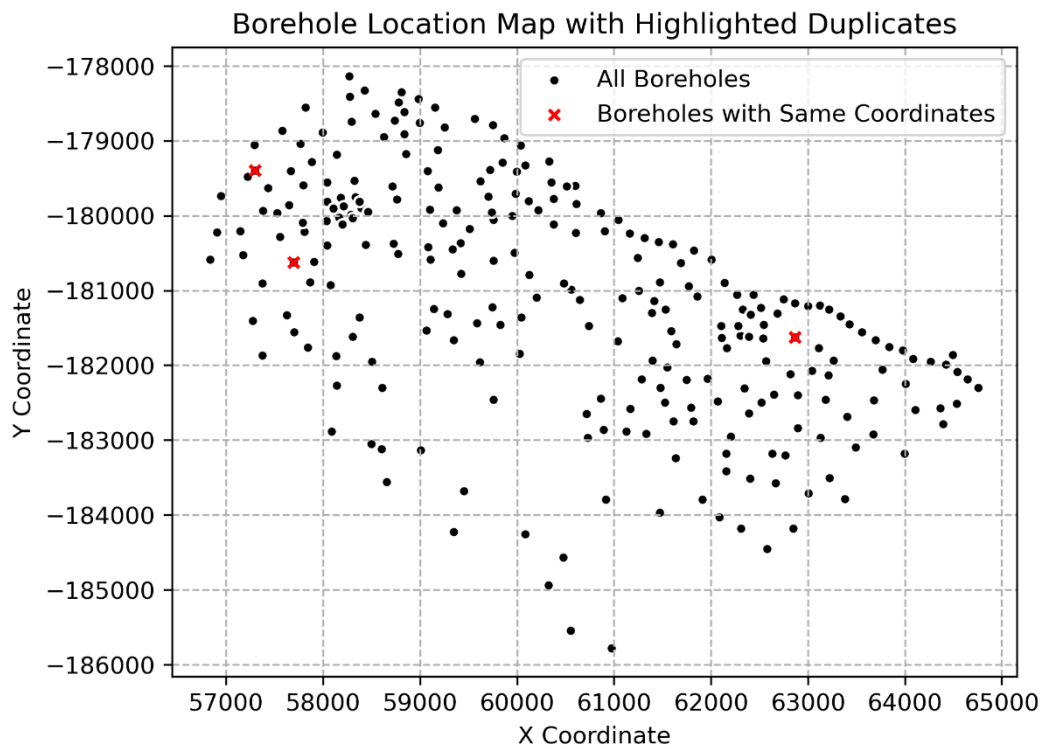


Figure 4-5: Duplicate pairs in the deposit area

Figure 4-5 shows the duplicate pairs within the area of interest. With the research being geared towards the predictive capabilities of ML, it was decided that attributes be differentiated by depth. This was done by increasing the coordinates (X and Y) of each borehole by 0.001m. This shift represents a small distance relative to the scale of the study area; the impact can be considered negligible; therefore, all 570 borehole data points were used.

Figure 4-4 clearly shows that the boreholes are spread across the entire project area. There are however areas where boreholes appear to be clustered together, further coding the location maps according to PGE grade values allows for better interpretation. From table 4-2 the deposit has a mean value of 5.76 *g/t* and the median grade is 5.90 *g/t*. The median is approximate to the mean, hinting at a relatively symmetric distribution around the centre. This is in stark contrast to the mode, suggesting that the mean could possibly be influenced by high-value outliers.

Table 4-2: Summary Statistics of PGE grades

	PGE
Mean	5.76
Median	5.90
Mode	2.20
Standard Deviation	2.20
Sample Variance	4.84
Kurtosis	0.44
Skewness	0.20
Range	13.75
Minimum	1.12
Maximum	14.88
CoV	0.38
Count	570.00

The lowest grade value is 1.12 g/t and the highest is 14.88 g/t , this gives us a range of 13.68 g/t . The inter quartile range is 2.9 g/t , the range between the lower quartile and the median is 1.66 g/t which is comparable to the range between the upper quartile and the median at 1.27 g/t . This suggests moderate variability throughout the deposit. The variance and standard deviation of the deposit is 4.84 $(g/t)^2$ and 2.20 g/t respectively. The moderate value in the variance and standard deviation justifies assertions made over the deposit from the ranges.

Figure 4-6 provides the needed visual context to complement the summary statistics. The histogram clearly shows two peaks, first, around 2-4 g/t and second, around 6-8 g/t . This explains the mode vs mean/median discrepancy and suggests there are distinct populations of varying grades in the dataset. The boxplot indicates potential outliers with points beyond the upper “whisker” of the plot. This was verified by assessing its impact on the mean.

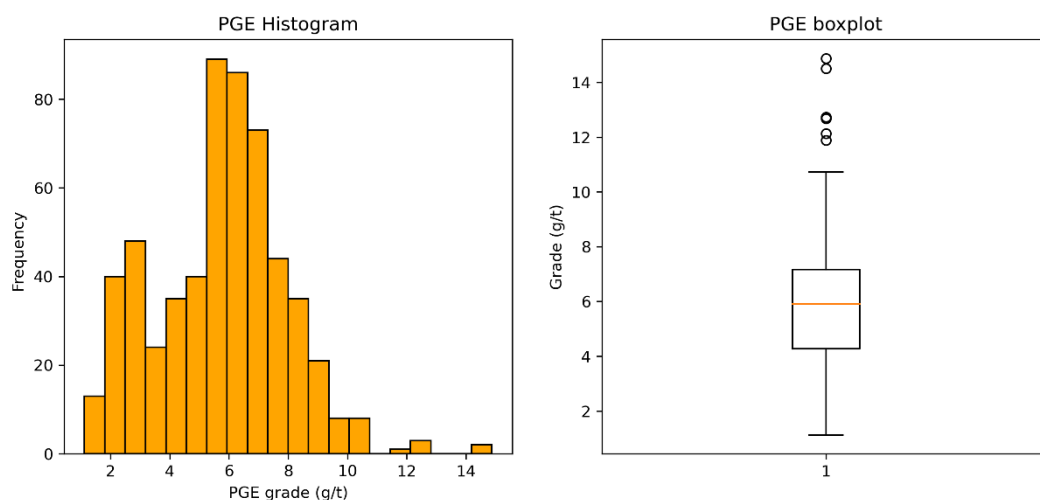


Figure 4-6: Histogram (left) and Boxplot (right) of PGE grade values

A slightly positive kurtosis of 0.44 suggests the distribution of the PGE grades has a heavier tail than the normal distribution. At first glance the skewness (0.20) and kurtosis indicate a slight bias of the PGE grades towards higher grade values even though the distribution may appear to be relatively symmetrical.

Figure 4-7 shows that the deposit does not follow a normal distribution. Furthermore, “relative symmetry” is confirmed to apply solely to grade values near

zero on the x-axis. From the probability plot, distinct populations of different grade ranges can be observed. This may challenge the notion of bimodality as these ranges indicate the multi-modal nature of the deposit, with up to four population observed with ranges of 0-3g/t, 3-5g/t, 5-11g/t and, 11-15g/t. For simplicity's sake these grades can be grouped in low, medium, and high grades (see figure 4-8). These grade categories are mixed throughout the area of interest and is more prevalent with the medium and high grades. Low grades make up considerably less portion of the grade values and can be seen to be denser in the eastern portion of the map before spreading out sparsely in a north-easterly direction.

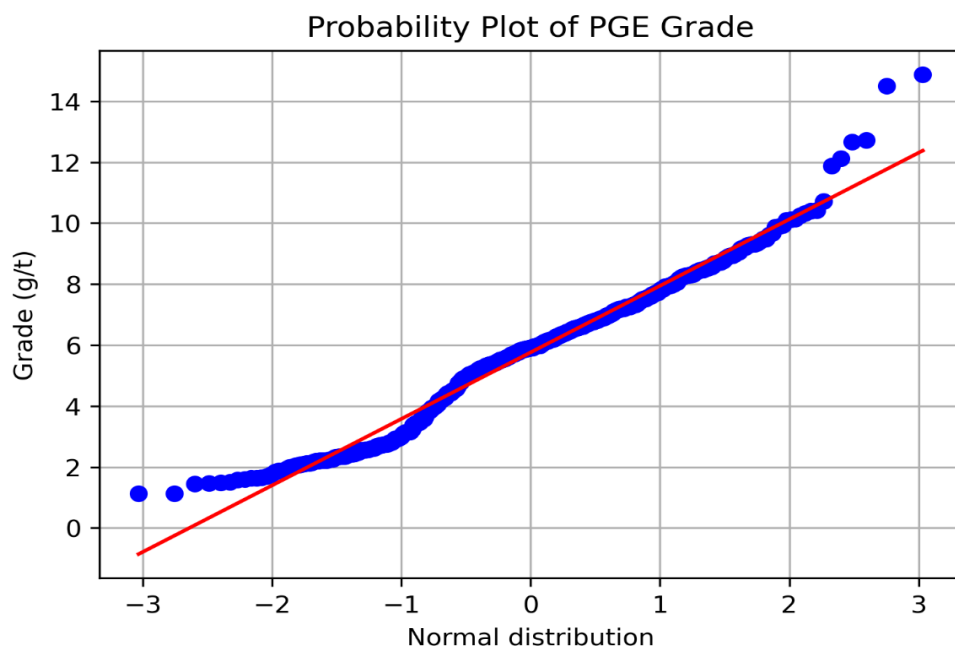


Figure 4-7: Probability plot of PGE grade

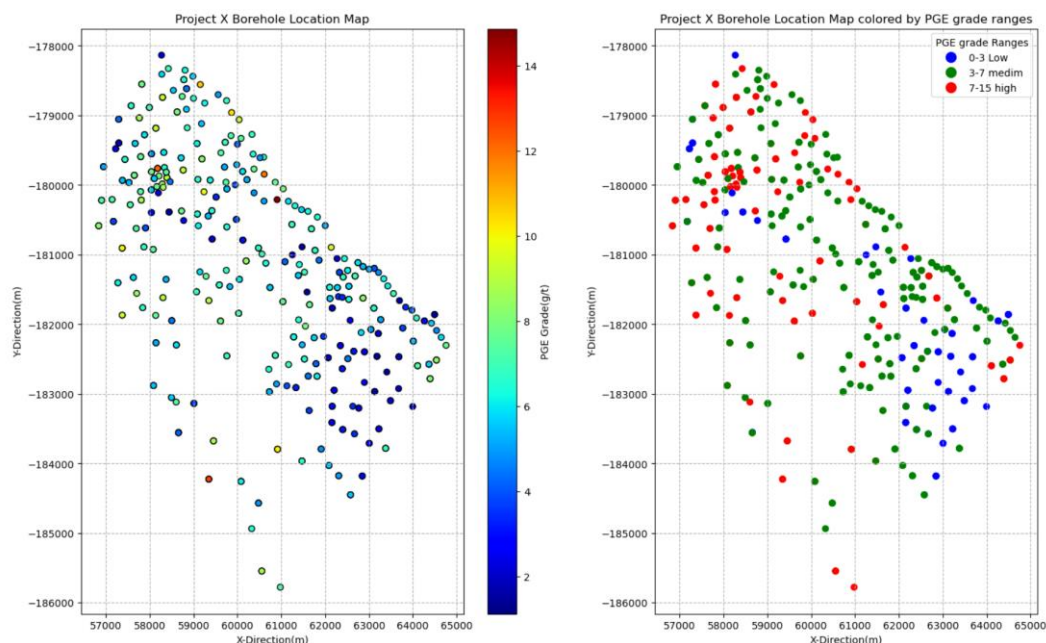


Figure 4-8: Project X location map according to PGE grades

Greater separation can be observed above $11g/t$ in figure 4-7 which poses a question as to whether such values have undue influence over the deposits' statistics. From left to right, table in appendix A1 contains an ordered array of PGE grade values from lowest to highest. The top six grade values (potential outliers) are $11.883g/t$, $12.130g/t$, $12.677g/t$, $12.728g/t$, $14.499g/t$ and $14.875g/t$. Determining the influence of these extreme values on the deposit was done through a process called capping.

Table 4-3: Mean grade of PGE data capped at various thresholds

	Threshold (g/t)				
	11.883	12.130	12.677	12.728	14.499
Mean (g/t)	5.749	5.751	5.755	5.755	5.761

It is a process whereby values higher than a defined outlier threshold is reset to the outlier threshold itself (Deutsch & Rossi, 2013). Iterating this process over a range of outlier threshold grades allows for comparison between the actual grade and the capped grade. Using the highest five grade values as thresholds, a general reduction in the mean value was observed. This reduction, however, is marginal. The difference between the lowest threshold and the actual grade is $0.013 g/t$

which equates to 0.23% of the deposits' mean grade, thus rendering the effects of outliers negligible.

According to Coombes (2008), precious metals are predominately log-normally distributed. The distribution of the Project X deposit may suggest that this is not universal for all PGE deposits. Sinclair and Blackwell (2002) describe the coefficient of variation (CoV) as an indicator as to whether a distribution is normal or log normal. CoV is the *standard deviation/mean*, above 0.5 the distribution is better described by a log normal distribution, below 0.5, it is likely to approach a normal distribution. Applying this rule of thumb to the deposit, a CoV of 0.38 implies the deposit is not log normal. The apparent presence of multi-modality in the data may hint at the underlying distribution associated with the other attributes within the data file such as the contributing and associated elements to the composition of the PGE as well as density. This can be further inspected by plotting and examining the correlation of associated elements in the dataset (Moon, et al., 2006).

4.5 Variography

The analysis of grade continuity (Variography) is concerned with comparing samples to the distance and orientation between them (Journel & Huijbregts, 1978). This can be misleading because in a real-world setting drill holes are mostly drilled in areas of greatest interest. The EDA suggested the effects of the large grade values are negligible, however, a second opinion would help to dispel any uncertainty. Declustering is a technique used to address the challenges associated with uneven borehole spacing.

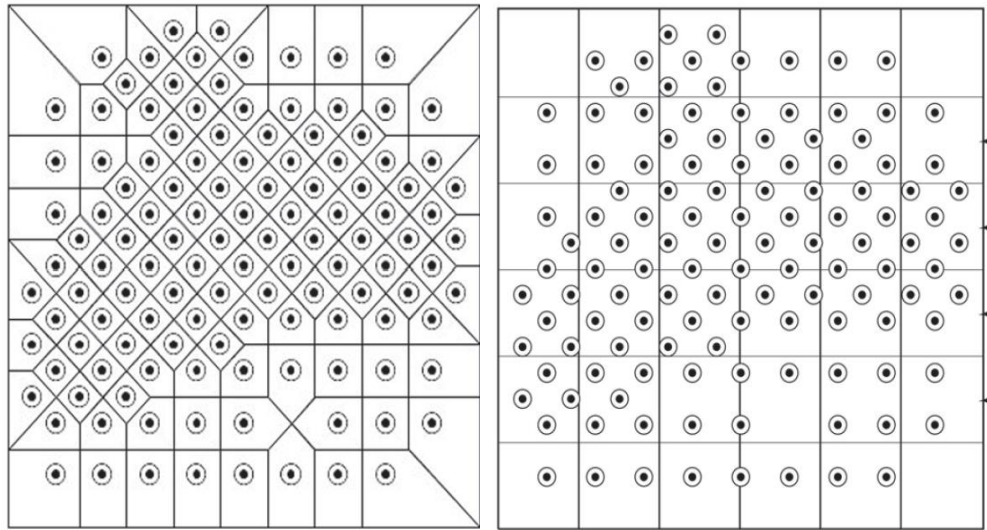


Figure 4-9: Polygonal (left) and cell declustering (right)

The premise of declustering techniques is to allocate weight to individual data points according to their proximity to neighbouring data $w_i = i, \dots, n$ (Deutsch & Rossi, 2013). Figure 4-9 shows an example of polygonal declustering to the right-hand side and cell declustering to the left. They each have best use cases, for instance, polygonal declustering provides poor declustering weights except in relatively simple 2-D cases with very well understood lease or geological units (Deutsch, 2015). Such reasons make the use of cell declustering more favourable as it is more robust.

The *geostatpy* package makes use of cell declustering. Through the function *geostatpy.geostats.declus* it makes use of the *declus* program reimplemented from *GSLIB*. Table 4-4 contains the input parameters required by the *geostatpy.geostat.declus* function. The outputs of this function are the weights, cell sizes and the declustered means.

Table 4-4: Input parameters of the *geostatpy.geostat.declus* function

Parameter	Description
<i>df</i>	Data Frame containing the data
<i>xcol</i>	X coordinates
<i>ycol</i>	Y coordinates
<i>vcol</i>	grade values

<i>iminmax</i>	1; minimise the declustering mean,0 maximise the declustering mean
<i>noff</i>	Number of origin offsets
<i>ncell</i>	Number of cells with a minimum and maximum
<i>cmin</i>	Based on the minimum size of the dataset
<i>cmax</i>	Based on the maximum size of the dataset

To correct sampling bias against high grade areas *iminmax* will be set to 1 to minimise the declustered mean. The weights were visualised over the area of interest as seen in figure 4-10. It was observed that the weights are either slightly above or below 1, with a few exceptions around the edges of the map. This suggests that the points in this area are under sampled. The distribution of the weights of the project area suggests that the grade values are neither dense nor sparse and represent the average sample density across the area.

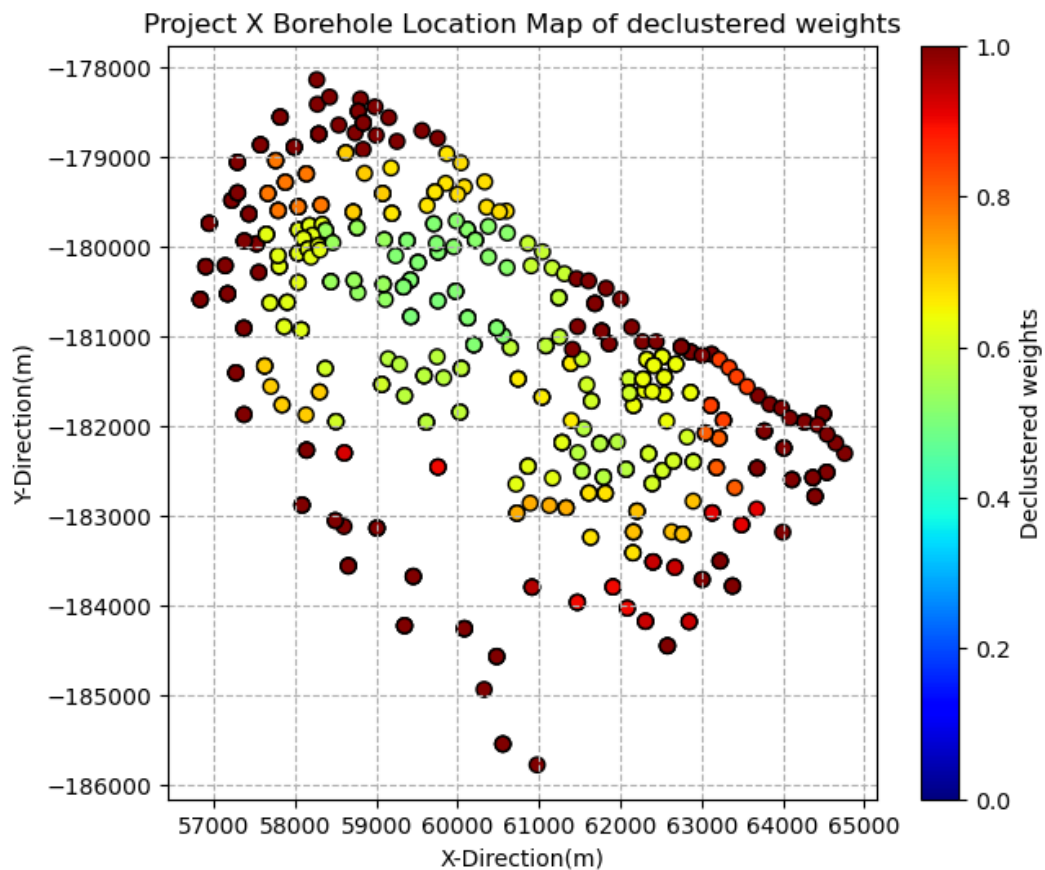


Figure 4-10: Map of declustered weights over area of interest

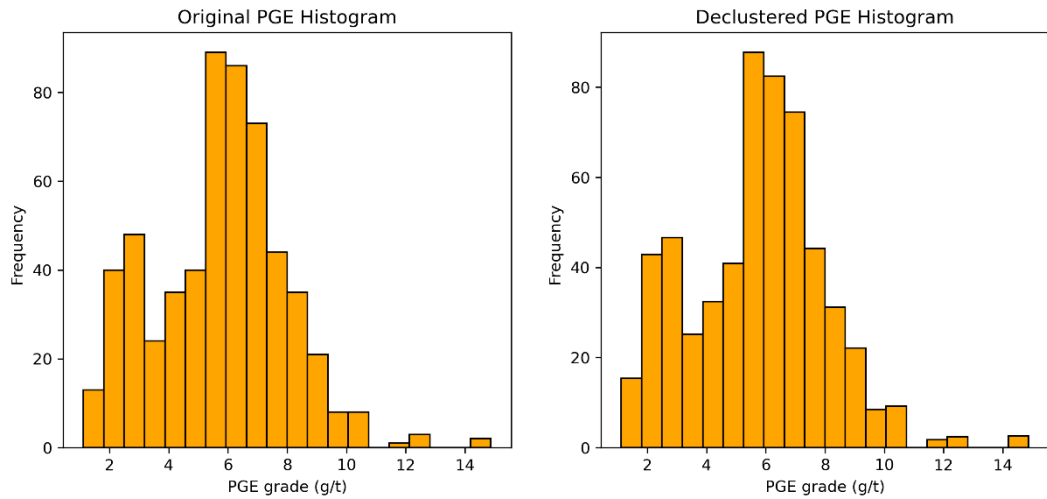


Figure 4-11: Adjacent comparison of the original (Left) vs declustered histogram (Right)

This is further cemented by the marginal differences in the distribution of the grades between the original data and the declustered data observed in figure 4-11. The declustered mean is 5.678 g/t which gives a negligible correction of 0.0146 g/t .

As a precursor to experimental variograms, a variogram map serves as a useful visualisation tool that aids in exploring the spatial continuity of a given dataset (Abzalov, 2016). The *geostatspy* package allows for this through the *geostats.varmapv* function. According to Deutsch & Rossi (2013), standardising data into normal scores helps to deal with any outliers; this is a requirement before a variogram map can be produced with *varmapv*.

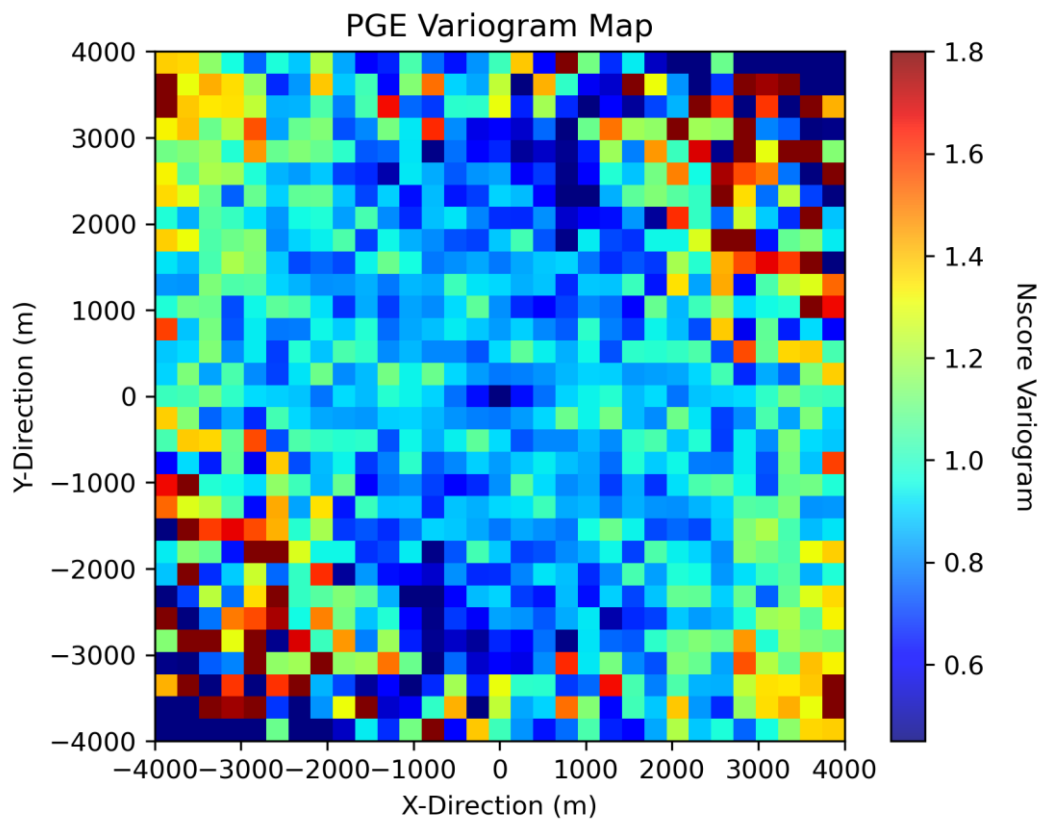


Figure 4-12: Variogram map of the PGE in the Project X lease area

The variogram map of the PGE deposit in figure 4-12 shows no clear elongated patterns or directional trends that would suggest anisotropy. An expected anisotropy would be indicated by consistent colour changes in a particular direction, which would have implied that the spatial correlation of the PGE grade is directionally dependent. The lack of any preferred directional trend in figure 4-12 rendered any inference on anisotropy at this point inconclusive. The range of influence was not directly observable in figure 4-12 as it is not a traditional plot with distance on one axis and semi-variance on the other. At the origin (0,0 distance), the lower semi-variance values imply a low-moderate nugget effect within the PGE deposit. Definitive statements could not be made about the anisotropy and range of influence over the project area from figure 4-12. A series of directional experimental variograms were constructed to overcome this dilemma. Since the borehole configuration of the deposit is not regular, more emphasis was placed on using suitable tolerances.

At this point it is vital that the difference between the average borehole and data spacing is discussed. The key difference lies between what is being measured and for what purpose. Borehole spacing specifically refers to the arrangement of boreholes and is more about the physical process of data collection. Data spacing as explained by Wilde & Deutsch (2010) refers to the distance between adjacent points within a representative area. The data spacing at a location, denoted as $s(u)$, depends on the number of nearby samples, $n_v(u)$, within a certain volume $V(u)$. For two-dimensional volume, data spacing is calculated using the equation:

$$s(u) = \sqrt{\frac{V(u)}{n_v(u)}} \quad (4.1)$$

The *DataSpacing* program is not available on *geostatspy*, however, coding the logic of equation 4.1 in python yielded a data spacing of 324m. Meaningful average data spacing is obtained by adjusting the radius according to the scale and distribution of the data.

Selection of suitable tolerances like much of Variography is undoubtedly an iterative process. There remains a criterion for reasonable tolerance selection. According to Deutsch & Rossi (2013) if tolerances are too small the variogram will be noisy, conversely, when too large, the spatial continuity will be averaged out and imprecise. A variogram is only valid for one half of the area of interest. The objective is to balance detail and stability. If data is erratic or varies significantly across the study area, a larger lag tolerance can help ensure enough data pairs per lag especially in sparsely sampled regions.

The selection of the variogram calculation direction is influenced by expected geological anisotropy, the availability of data samples and the software in use (Deutsch & Rossi, 2013). Bandwidths are used to limit the number of unreasonable associations and maximise reasonable pairs. Its application will vary across geostatistical software packages. As *geostatspy* is a reimplementation of *GSLIB*, it makes use of a rectangular pyramid for clipping the experimental variogram (Deutsch, 2015). Experimental variograms were plotted with the initial estimates of lag distance and tolerance, then, adjusted based on spatial structure.

Table 4-5: Example of experimental variograms showcasing the iterative process

	Trial Variogram Parameters				
	Lag distance(m)	Lag tolerance(m)	No. lags	Azimuth (°)	Azimuth Tolerance(°)
Variogram 1	70	35	30	0	10
	162	81	30	0	10
	500	250	30	0	10
Variogram 2	70	35	30	45	15
	162	81	30	45	15
	500	250	30	45	15
Variogram 3	70	35	30	90	22.5
	162	81	30	90	22.5
	500	250	30	90	22.5
Variogram 4	70	35	30	135	45
	162	81	30	135	45
	500	250	30	135	45

Table 4-5 contains a series of directional variograms to provide a basis for interpretation of the preferred direction of continuity. The lag distances of 70,162 and 500m represent distances over short, intermediate, and long intervals. The number of lags were kept constant at 30 for practicality and efficiency. To accurately determine the direction of anisotropy, directional experimental variograms were constructed in the N-S, E-W, NE-SW, NW-SE directions represented by azimuths of 0,90,45,135°. The azimuth and azimuth tolerances were changed from one series to the next. Variograms 1-4 are not the only series of variograms constructed for the project but rather serves as a template to demonstrate the iterative process involved in the model selection. With the aim of negating the effects of misleading anisotropy, potential bias and loss of resolution, the initial lag tolerances were kept at exactly half the lag distance. This produced highly erratic and noisy data over short and intermediate distances and unsuitable for use in producing kriging estimates. Figure 4-13 represents the importance of selecting an appropriate bandwidth tolerance.

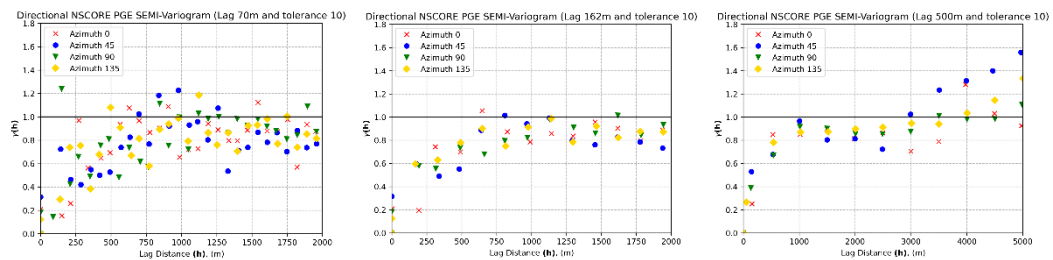


Figure 4-13: Experimental variograms with 9999.9m bandwidth tolerance

The plots in figure 4-13 do not show a consistent pattern across azimuths for the different lag distances, neither do they diverge significantly to suggest anisotropy. This is owed to the fact that the default bandwidth was set too high at 9999.9m. It is also evident that the overly large bandwidth led to loss of spatial information, produced misleading results, and compromised computational efficiency.

To ensure that there were enough data pairs within each lag to make the experimental variogram more reliable, the default bandwidth was changed. An initial bandwidth of half the lag distance was chosen to evaluate the discrepancies in the experimental variograms at short, intermediate, and large lag distances.

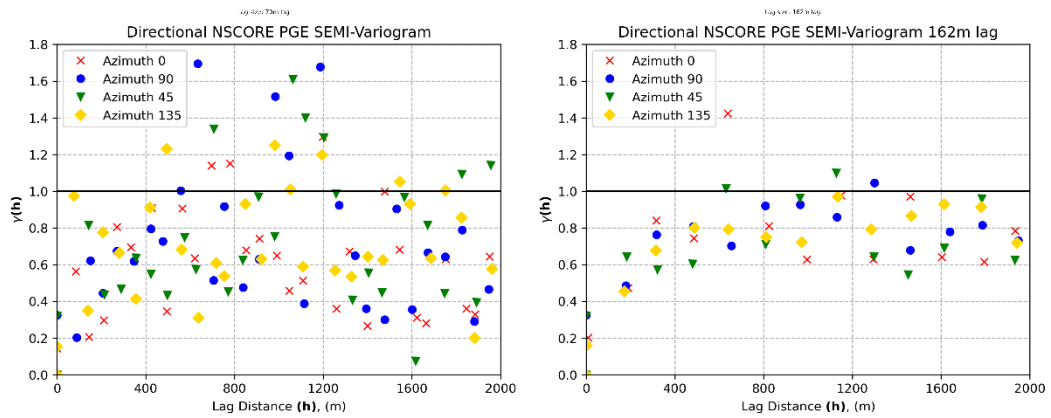


Figure 4-14: Experimental variogram with a reduced bandwidth

In contrast to figure 4-13, figure 4-14 indicates anisotropy, as semi variance values vary with direction. 70m lag distance provides the most information on small scale spatial variability and anisotropy. Due to the higher density of points, it is more influenced by local variation and noise. The 162m lag distance provides more of a balance between resolution and smoothing, appearing to be less sensitive to small scale variations. 500m lag distance is less detailed than both short and intermediate lag distances, capturing the overall continuity over large distances. For general trend identification and model fitting the 162m lag distance was found to provide the best compromise.

The parameters used to produce the 162m experimental variogram were altered to produce a more suitable variogram for model fitting. This is represented by the parameters seen in table 4-6. Increasing the lag distance to 200m allows for capturing the spatial structure between data points. It is close enough to the average data spacing to provide detailed information while avoiding noise that occurs at shorter distances. The lag tolerance of 100m follows the rule of thumb convention. Reducing the number of lags from 30 to 25 helped to exclude distances where there is no meaningful spatial correlation. A 100m bandwidth is appropriate as it ensures that enough data pairs are included in the calculation of the semi variance for each lag. It further allows for the inclusion of data pairs that are spaced reasonably close to each other while maintaining integrity of directional information. The azimuth tolerance of 22.5° was found to provide a good balance between capturing directional trends and including enough data pairs for robust statistics.

Table 4-6: Preferred experimental variogram parameters

Experimental variogram parameters				
Lag distance (m)	Lag tolerance (m)	nlag	Bandwidth (m)	Azimuth tolerance (°)
200	100	25	100	22.5

Examining the resulting experimental variogram allowed for inferences to be made on the spatial structure and the direction of preferred continuity. The direction of preferred continuity is typically indicated by the azimuth with the largest range before reaching the sill. Based on the experimental variograms provided in figure 4-15, the NW-SE (Azimuth 135°) experimental variogram was selected as the direction of preferred continuity as it extends the furthest before plateauing at the sill.

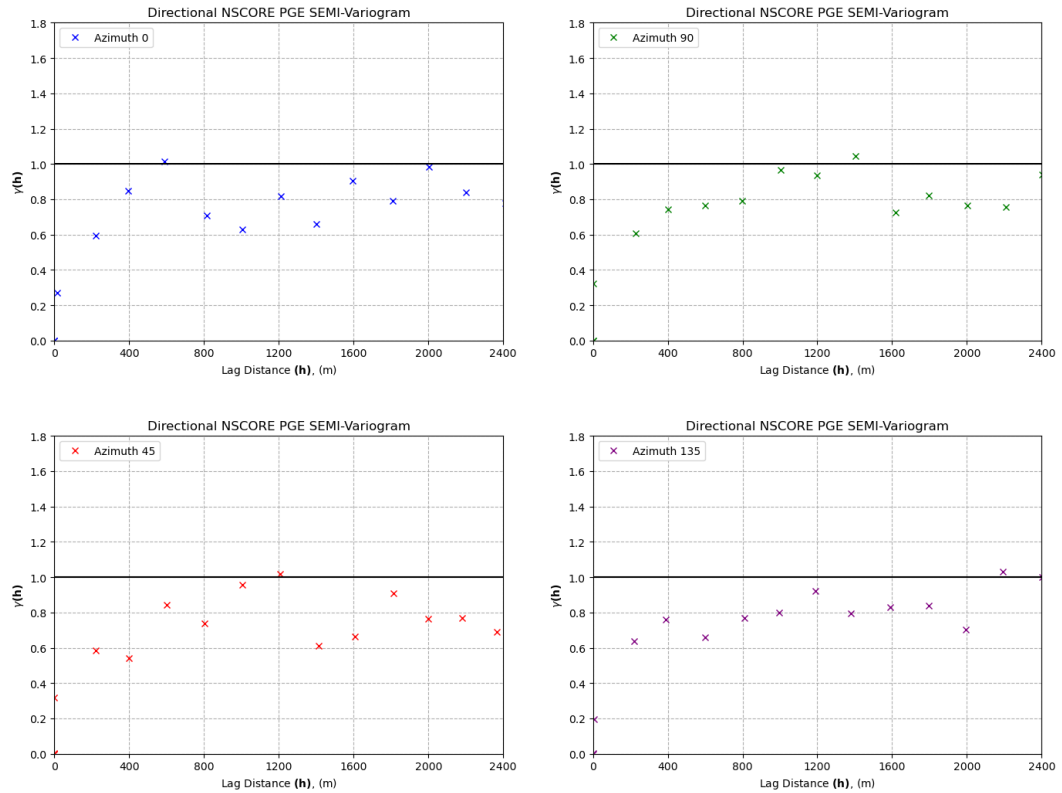


Figure 4-15: Suitable experimental variogram parameters across 0,90,45 and 135° azimuths

Anisotropy is indicated by the difference in the shape and range of the experimental variograms in different directions. Further examining figure 4-16, there appears to be geometric anisotropy present, as the variograms for azimuths 45 and 135° level off at the different lag distances. The nugget effect, which is the value at the origin appears consistent across all directions suggesting that the microscale variability is not directionally dependent.

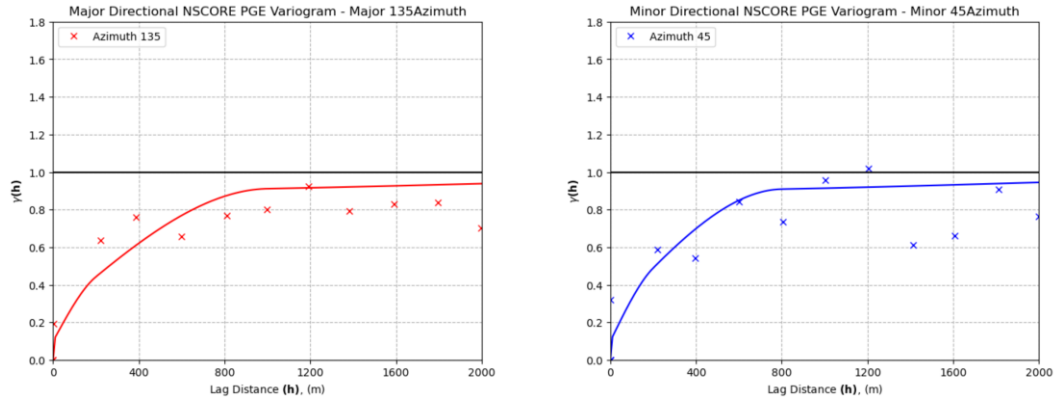


Figure 4-16: Experimental and model variogram for the 135°(major) and 45°(minor) direction of anisotropy for the PGE data

Three structures were found to be most suitable to match the experimental variogram as closely as possible while maintaining a model that is justified by the data and the underlying physical processes (see figure 4-16). The default *make_variogram* and *vmodel* functions within the *geostatspy* package only accommodate two structures. The open-source nature of python allows for the altering of source code to meet user specific needs. The *make_variogram3* and the *vmodel3* functions were created to facilitate the use of three structures.

```
nug = 0.1; nst = 3
it1 = 1; cc1 = 0.15; azi1 = 135; hmaj1 = 200; hmin1 = 200
it2 = 1; cc2 = 0.65; azi2 = 135; hmaj2 = 1000; hmin2 = 800
it3 = 1; cc3 = 0.1; azi3 = 135; hmaj3 = 5000; hmin3 = 4500
```

Figure 4-17: Model variogram parameters

Figure 4-17 represents the parameters used to create the nested variogram model, with a nugget effect and three structures (*it1*, *it2*, *it3*) that explain different proportions and scales of spatial variability. All structures are represented by the spherical model, with first structure (*it1*) accounting for 15% of the partial sill ($cc1 = 0.15$), the second (*it2*) and third (*it3*) structures contribute 65 and 1% respectively to the partial sill ($cc2 = 0.65, cc3 = 0.1$). Combined with the nugget effect of 10% ($nug = 0.1$) which amounts to the sill /total variability of 1.

The suitability of the models is inferred by three points, namely, data coverage, nugget, sill, and range. Firstly, the model curves pass near or through the bulk of the experimental data points seen in figure 4-16. Secondly, the models capture the low nugget effect, indicating negligible micro-scale variability or measurement error. Lastly, the models reach the sill and range that capture both short and long-range dependencies in the data.

Confirmation in the suitability of the variogram models necessitated the use of validation metrics to assess the predictive performance of the variogram models and will be considered in the next chapter.

4.6 Machine Learning

Keeping with the ML convention discussed in the literature review, at this point it should be clear that PGE is the response variable. Table 4-7, excluding BHID, contains the predictor variables associated with the PGE data.

Table 4-7: Table containing target and response variables of the Project X data

	BHID	X	Y	Z	DENSITY	PGE	CU	NI	PT	PD	RH	AU
0	WV124D0	64537.971	-182515.439	940.59	4.1604	9.6785	0.0132	0.1229	3.9519	4.9729	0.5590	0.1946
1	WV124D1	64537.871	-182515.549	940.65	4.3021	6.9002	0.0072	0.1192	3.8498	2.3939	0.6306	0.0259
2	WV124D2	64537.761	-182515.659	940.67	4.1150	8.4249	0.0286	0.1578	4.1353	3.5229	0.5775	0.1891
3	WV93ID0	64758.171	-182303.839	1014.90	4.1652	7.5486	0.0112	0.1225	4.3245	2.4174	0.6668	0.1400
4	WV94ID0	64647.281	-182189.059	1010.84	4.3862	5.7957	0.0056	0.1169	3.7544	1.4438	0.5633	0.0341

How these predictor variables relate to the response variable can be visualised with a correlation matrix (see figure 4-18). The relationship and distribution of the predictor variables provided more context on the multimodal nature of the PGE data which was inferred during EDA. From figure 4-18 however, it was discovered that platinum (Pt) and palladium (Pd) contribute the most to the PGE deposit with a correlation of 0.95.

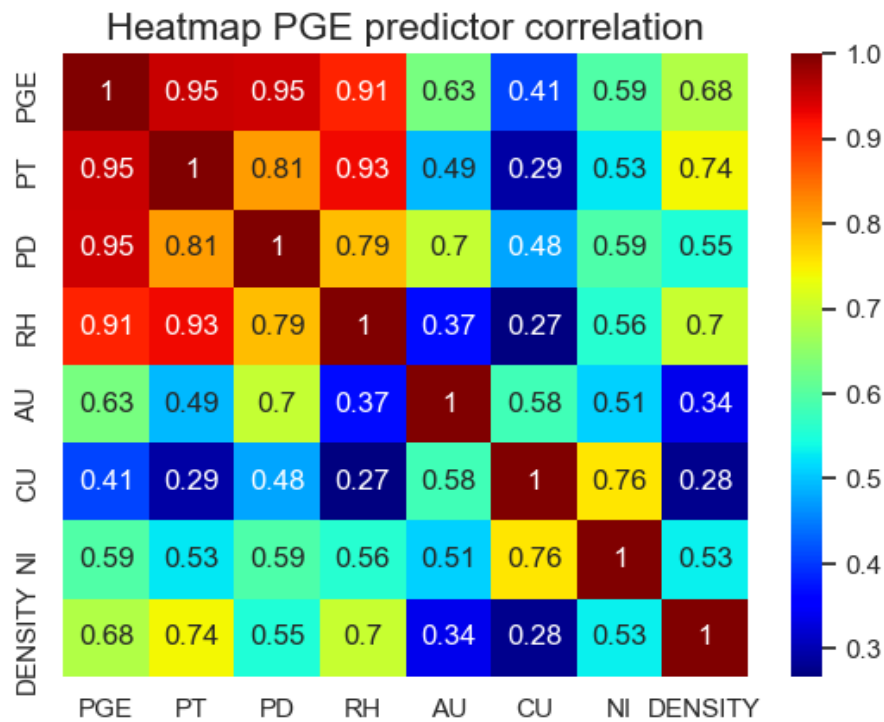


Figure 4-18: Correlation heatmap of predictor variables

Pairplots (see appendix A2) show the distribution of the individual predictor variables as well as the correlation scatterplots between the variables. Paying attention to the CoVs of Pt, Pd, rhodium (Rh) and Gold (Au), it is evident that the PGE deposit consists of a mixture of distributions namely normal and log normal.

4.6.1 Feature engineering: scaling, standardisation, and normalisation

Feature engineering is essential in producing precise and effective machine learning models. It necessitates for the use of scaling, normalisation, and standardisation to make the data more suitable for modelling (Wu et al., 2020). The utilisation of these techniques, enhance model accuracy, mitigate the effects of outliers, and guarantee uniform data scaling.

Feature scaling is a method of preprocessing data in machine learning which involves adjusting the values of features or variables within a dataset to a common uniform scale. Common methods for feature scaling include standardisation, and normalisation. In datasets where predictor variables vary widely in range, units, or magnitude, without feature scaling such variations would result in biased model performance or hinder the ML process.

Table 4-8: A comparison between normalisation and standardisation (Bhandari, 2024)

Normalization	Standardization
Rescales values to a range between 0 and 1	Centers data around the mean and scales to a standard deviation of 1
Useful when the distribution of the data is unknown or not Gaussian	Useful when the distribution of the data is Gaussian or unknown
Sensitive to outliers	Less sensitive to outliers
Retains the shape of the original distribution	Changes the shape of the original distribution
May not preserve the relationships between the data points	Preserves the relationships between the data points
Equation: $(x - \min)/(\max - \min)$	Equation: $(x - \text{mean})/\text{standard deviation}$

Standardisation within ML context where data is centred around the mean and standard deviation. Also referred to as Min-Max scaling, normalisation is a scaling technique where data values are shifted and rescaled so that they end-up ranging between 0 and 1 (Wu et al., 2020). Table 4-8 compares the differences between normalisation and standardisation. A preference for one over the other is solely down to the problem at hand and the algorithms used. Some ML algorithms are sensitive to feature scaling while others are invariant. For this project, the best result was obtained by fitting the model to the original data.

From the available predictor features available, the general approach is to select the fewest predictor variables that provide the best results. For this project the selection of prediction variables was greatly affected by comparison considerations. The primary goal of spatial interpolation is to predict values for unsampled locations based on their spatial relationships to sampled locations. Using only X and Y coordinates focuses the model on understanding and capturing these spatial relationships, without introducing potential confounding factors from

other variables. For this reason, The Z variable was excluded from the list of predictor variables.

4.6.2 Sci-kit Learn

Sci-kit learn serves as a python library that aids in the development of conventional ML model. It consists of a suite of python-based tools, functions and procedures that facilitate the deployment of a wide range of both supervised and unsupervised ML algorithms. It is built from already existing technology like NumPy, pandas, iPython, Sympy and Matplotlib (Pedregosa et al., 2011).

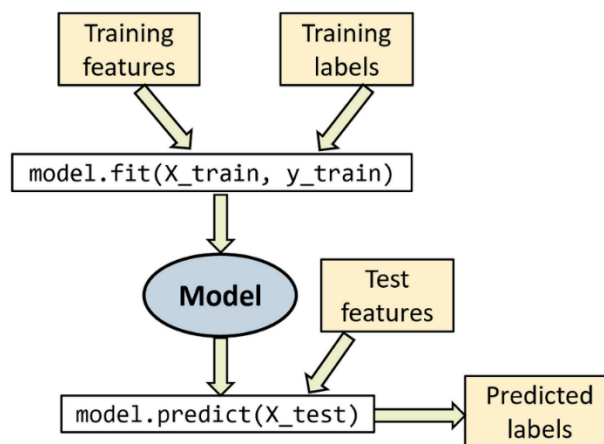


Figure 4-19: Supervised ML workflow

The process of formulating a prediction can be seen in figure 4-19. This method can be described in roughly five steps. Namely: (i) Selection of a suitable model, (ii) preparation of data (iii) splitting the data, (iv) training the model and (v) making predictions.

```
models = {  
    'Linear Regression': LinearRegression(),  
    'SVM': SVR(),  
    'SVM2': make_pipeline(StandardScaler(), SVR()),  
    'Decision Tree': DecisionTreeRegressor(),  
    'Random Forest': RandomForestRegressor(),  
    'KNN': make_pipeline(StandardScaler(), KNeighborsRegressor())  
}
```

Figure 4-20: Initialisation of ML models

Figure 4-20 shows the models selected for the project being initialised from the Sci-kit learn library. Further describing the Scikit learn prediction process in figure 4-19; an 80/20 approach was used to split the data, where 456 data were used for training and 114 data were used for testing the model.

StandardScaler is a preprocessing function provided by the Sci-kit learn library in Python. Such functions are necessary for algorithms such as *k*NN and SVR, as *k*NN calculates the distance between different points by distance metrics like Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this feature. Therefore, the scale of the variables impacts the *k*NN algorithm. *StandardScaler* helps to normalise the feature set to ensure that each feature contributes approximately proportionately to the final distance. Like *k*NN, SVR also depends on the distance between the data points. SVR tries to find a decision boundary that is maximally far away from any data point. If one feature has a very large range, it will dominate the other features, leading to a biased decision boundary. Scaling ensures that the SVR model is equally sensitive to all features. The *make_pipeline* function in the code is used to streamline the process, creating a pipeline that first scales the data and then applies the respective algorithm.

The training data was then fed into the selected models by calling the function *model.fit()*. Once the model was trained, new predictions were made by passing new data into *model.predict()* function. Here, the different models will use what it has “learned” to make predictions on new data. This was essential for evaluating the models.

4.6.3 Hyperparameters and Hyperparameter tuning

Hyperparameters are parameters whose values are set before the learning process begins. Unlike model parameters which are learned during training, hyperparameters are externally set and govern the training process itself. They can significantly influence the performance of a model, affecting its ability to learn from the data (Probst et al., 2019).

Table 4-9 contains the hyperparameters tuned for this project. Relevant hyperparameters for regression are only used in the case of ridge and lasso

regression, which meant that no hyperparameters were applied to Linear Regression. Decision trees share the same hyperparameters to Random Forest excluding *n_estimators*, because only a single tree is considered with Decision Trees. With SVR, the default kernel was used to reduce computational time when tuning for optimal parameters.

Table 4-9: Hyperparameter search space with descriptions

ML Model	Hyperparameter	Default	Project Range	Description
Random Forest	<i>n_estimators</i>	100	[50,100, 150,200]	The number of trees in the forest. Many trees can improve performance but increases computational cost
	<i>max_depth</i>	None	[3, 10, 20]	Controls overfitting. Higher depths can allow the model to learn highly specific patterns
	<i>min_sample_split</i>	2	[2,3,4,5,6,7,8,9,10]	The minimum number of samples required to split an internal node
	<i>friedman_mse</i>		None	a modification of the mean squared error (MSE) that is used for improving the model's ability to deal with certain types of datasets, particularly for boosting models.
SVR	<i>C</i>	1	[0.1, 1, 10]	Trades off correct fitting of the training data against maximisation of the decision function's margin

	<i>kernel</i>	rbf	[rbf]	Specifies the type of hyperplane used to separate the data
	<i>epsilon</i>	0.1	[0.1, 1, 10]	Defines the margin of tolerance where no penalty is given to errors
kNN	<i>n_neighbors</i>	5	[3, 5, 7, 9]	Number of neighbors to use for k-neighbours queries
	<i>weights</i>	uniform	['uniform', 'distance']	Weight function used in prediction. Possible values
	<i>metric</i>	minkowski	['euclidean', 'manhattan']	The distance metric to use for the tree

Hyperparameter tuning, which can also be referred to as hyperparameter optimisation, is the process of finding the optimal set of hyperparameters given for a ML model to maximise its performance. This usually involves iteratively searching through a predefined space of hyperparameter values and assessing the model performance with each set of values. Cross-validation is often used in this process and the best model is selected based on metrics such as accuracy, precision, recall or any other similar metric. Hyperparameter tuning strategies include, grid search, random search, Bayesian optimisation, gradient-based optimisation, and lastly evolutionary algorithms.

For the ease of application, grid search was the preferred hyperparameter tuning strategy used for this project. Sci-kit learn has the *GridSearchCV* function that like any grid search strategy exhaustively considers all parameter combination provided in a grid.

```

# best_params = {}

# Loop through each model and perform GridSearchCV
for name, model in models.items():
    tqdm.write(f"Starting grid search for {name}...")

    # Use GridSearchCV
    grid_search = GridSearchCV(model, param_grids[name], cv=5, scoring='r2', n_jobs=-1)
    grid_search.fit(X_train, y_train)

    # Get the best parameters
    best_params[name] = grid_search.best_params_
    tqdm.write(f"Best parameters for {name}: {best_params[name]}")
    tqdm.write(f"Completed grid search for {name}.\n")

# best_params now contains the best parameter set for each model

```

Figure 4-21: Source Code of the grid search technique used on Project X data

The code snippet in figure 4-21 performs hyperparameter tuning based on the project ranges outlined in table 4-9 using the *GridSearchCV* from sci-kit learn. Focusing on the line *grid_search = GridSearchCV(...)* a 5-fold cross-validation was used for this project, indicated by *cv = 5*. Here, the PGE data was split into five smaller sets where the model was trained and validated five times. The parameter R^2 was the metric used to evaluate the performance of each hyperparameter combination given by *scoring = 'r2'*.

At the end of the loop in figure 4-21, *best_params* will hold the optimal parameters as seen in figure 4-21 for each model as determined by the grid search, using R^2 as the performance metric. This allows for the use of the best-found model configuration to train the final model on the entire training set or further evaluate them on the test (see Appendix A3).

4.7 Chapter Summary

Chapter 4 examines the geological background and data preparation essential for conducting the comparative study between OK and machine ML models for PGE grade prediction. This chapter lays the foundation by detailing the regional and local geological setting of a PGE deposit, referred to as Project X. It is situated within the Bushveld Igneous Complex, renowned for its significant PGE concentration, with a focus on the UG2 reef.

The chapter transitions into an EDA of the Project X deposit, encompassing 570 boreholes. Statistical and visual analyses, such as histograms, box plots, and probability plots, revealing a complex, multi-modal distribution of PGE grades, indicating distinct populations of varying grades within the dataset. Duplicate borehole coordinates are addressed, ensuring data integrity for further modelling.

Variography is extensively discussed, with an emphasis on analysing spatial continuity and determining suitable variogram models. Techniques such as declustering and the construction of experimental variograms are employed to correct sampling biases and to accurately capture the spatial relationships of PGE grades.

The chapter concludes with a detailed examination of machine learning methodologies, focusing on feature engineering and the preprocessing of data for ML modelling. It introduces the use of Sci-kit Learn for ML implementation, highlighting the process of hyperparameter tuning and model validation strategies. The chapter sets a comprehensive stage for the application of both OK and ML models in predicting ore grades, underscoring the meticulous data preparation and analysis carried out to ensure the reliability and accuracy of subsequent predictive modelling.

5 ANALYSIS AND FINDINGS

5.1 Introduction

In this chapter, the application and description of *GeostatsPy* functions as well custom python functions that extends their application will be discussed. These functions form an integral part of the geostatistical analysis and model validation process. Validation and comparison of the ML and OK models for point and block estimates.

5.2 *GeostatsPy*: Point and Block kriging functions

5.2.1 Point Prediction

The *kb2d_locations* function is an implementation of 2d kriging, specifically designed to perform spatial prediction at a set of locations based on observed data. It follows the principles outlined in *GSLIB*'s *KB2D* algorithm (Deutsch & Journel, 1997), adapted for Python.

Table 5-1: *kb2d_locations* Input parameters

Inputs	
<i>df</i>	A pandas DataFrame containing the spatial data
<i>xcol,ycol</i>	Column names for the x and y coordinates in <i>df</i>
<i>vcol</i>	Column name for the variable of interest in <i>df</i>
<i>tmin,tmax</i>	Trimming limits for the property; values outside this range are excluded from the analysis.
<i>df_loc</i>	A pandas DataFrame containing the locations where predictions are to be made.
<i>xcol_loc,ycol_loc</i>	Column names for the x and y coordinates in <i>df_loc</i>
<i>ndmin,ndmax</i>	Minimum and maximum number of data points to use for kriging at a location.
<i>radius</i>	Maximum search radius to find neighbouring points for the kriging calculation
<i>ktype</i>	Type of kriging; 0 for simple kriging, 1 for ordinary kriging.
<i>skmean</i>	The mean of the variable of interest, used in simple kriging.
<i>vario</i>	A dictionary containing variogram model parameters

This function allows for both simple and ordinary kriging, depending on the input parameters specified in Table 5-1.

The following steps describe the execution process within the *kb2d_locations* function:

1. **Initialisation:** Sets up constants and loads variogram parameters.
2. **Data Trimming:** Filters the input DataFrame based on the specified trimming limits
3. **KD-Tree Construction:** Creates a KD-Tree for efficient spatial search of nearest neighbours within the specified radius
4. **Loop Over Locations:** For each location in *df_loc*, the function performs the following steps:
 - **Neighbour Search:** Uses the KD-Tree to find the nearest data points within the search radius
 - **Covariance Calculation:** Computes spatial covariance values based on the variogram model for the data points and the prediction location
 - **Kriging System Setup:** Assembles the kriging matrix based on the covariances and solves it to obtain weights for the neighbouring points.
 - **Prediction and Variance Estimation:** Calculates the predicted value and estimation variance using the obtained weights.

The function is highly customisable with respect to the spatial data, the locations for prediction, and the variogram model. Error handling includes checks for insufficient data within the search radius and singular kriging matrices, which can occur in certain spatial configurations.

5.2.2 Block predictions

The *kb2d* function is an advanced version of the 2d kriging algorithm, tailored for gridding spatial data. It extends the basic principles of kriging to estimate values over a specified grid rather than at individual locations. This function is particularly useful for creating spatial predictions across an entire area of interest, providing not only estimates at each grid node but also the estimation variance and kriging weights.

Table 5-2: Summary of the *kb2d* function input parameters

Input Parameters	Summary
Spatial data	Includes the DataFrame <i>df</i> containing the spatial data, columns names <i>xcol</i> , <i>ycol</i> , <i>vcoll</i> for coordinates and variable of interest, and trimming limits <i>tmin</i> , <i>tmax</i>
Grid	Defines the grid on which to perform kriging, including dimensions (<i>nx</i> , <i>ny</i>), origin (<i>xmn</i> , <i>ymn</i>), and cell size (<i>xsiz</i> , <i>ysiz</i>)
Discretisation	<i>nxdis</i> , <i>nydis</i> specify the number of discretisation points per grid cell, used to approximate the block kriging.
Search	Including <i>ndmin</i> , <i>ndmax</i> for minimum and maximum data points used in kriging and radius for the search radius
Kriging Type	<i>ktype</i> and <i>skmean</i> specify the kriging type (simple or ordinary) and the global mean (if simple kriging).
Variogram Model	<i>vario</i> provides the variogram model parameters.

Table 5-2 summarises the input parameters necessary for carrying out block kriging with the *kb2d* function. Unlike point kriging, block kriging estimates average values over spatial blocks (grid cells in this case), which can provide a smoother and more generalised spatial prediction. This approach approximates the spatial variability within each grid cell by distributing discretisation points and averaging their covariances, improving the estimation of block values.

Like *kb2d_locations*, here is how the *kb2d* function carries out the block kriging process:

1. **Initialisation:** Sets up necessary constants, loads the variogram model, and prepares arrays for internal calculations
2. **Data Preparation:** Trims the input data based on specified limits and constructs a KD-Tree for efficient spatial searching
3. **Grid and Discretization Setup:** Initialises the estimation grid and calculates discretisation points within each grid cell to approximate block kriging.
4. **Kriging Loop:**
 - For each grid cell, it identifies nearby data points within the search radius using the KD-Tree

- Depending on the number of nearby points, it either proceeds with kriging calculations or assigns a default 'unestimated' value
 - For valid points, it constructs the kriging system (matrix of covariances and vector of covariances between the data points and the grid cell) and solves it to obtain kriging weights
 - Calculates the estimate and variance for the grid cell based on the solved weights
5. **Estimation and Variance Mapping:** Fills in the output matrices *kmap* and *vmap* with the estimated values and variances, respectively, for each grid cell.

5.3 ML and OK Model Validation and comparison: At point Location

With LOOCV using the *kb2d_locations* function, each of the 570 data points were iterated through; one data point was used as the validation data and the remaining 569 data points as the training data for each iteration. This was facilitated through the *loocv_kb2d* function seen in figure 5-1.

```
def loocv_kb2d(df, xcol, ycol, vcol, vario_params, tmin, tmax, ndmin, ndmax, radius, ktype, skmean):
    """
    Perform LOOCV with kriging and return RMSE, prediction errors, and a DataFrame of predictions.

    :param df: DataFrame with the spatial data (including 'X', 'Y', and 'PGE' columns)
    :param xcol: Column name for x coordinates in df
    :param ycol: Column name for y coordinates in df
    :param vcol: Column name for the variable of interest in df
    :param vario_params: Variogram model parameters for kb2d_locations
    :param tmin: Property trimming limit (lower)
    :param tmax: Property trimming limit (upper)
    :param ndmin: Minimum number of data points for kriging
    :param ndmax: Maximum number of data points for kriging
    :param radius: Search radius for kriging
    :param ktype: Kriging type
    :param skmean: Simple kriging mean value
    :return: Array of prediction errors, DataFrame of predictions
    """
    prediction_errors = [] # To store the prediction errors
    predictions = [] # To store predictions for each Left-out point

    for i in range(len(df)):
        # Separate one Location for validation
        validation_point = df.iloc[i]
        df_train = df.drop(i)
        # Prepare df_loc with just the Location information for the Left-out point
        df_loc = pd.DataFrame([validation_point[xcol], validation_point[ycol]])

        # Perform kriging for the Left-out point
        predicted, _ = geostats.kb2d_locations(
            df=df_train,
            xcol=xcol,
            ycol=ycol,
            vcol=vcol,
            tmin=tmin,
            tmax=tmax,
            df_loc=df_loc,
            xcol_loc=xcol,
            ycol_loc=ycol,
            ndmin=ndmin,
            ndmax=ndmax,
            radius=radius,
            ktype=ktype,
            skmean=skmean,
            vario=vario_params
        )

        # Compute prediction error
        actual_value = validation_point[vcol]
        prediction_error = predicted[0] - actual_value
        prediction_errors.append(prediction_error)
        predictions.append({xcol: validation_point[xcol], ycol: validation_point[ycol], 'Predicted_PGE': predicted[0]})

    # Convert predictions List to DataFrame
    predictions_df = pd.DataFrame(predictions)

    return prediction_errors, predictions_df
```

Figure 5-1: *loocv_kb2d* source code

After obtaining the kriging prediction, the function calculates the predicted value with the actual value of the left-out point. The function returns an array of prediction errors for each validation point and a DataFrame containing predictions. Each entry in the prediction DataFrame includes *x* and *y* co-ordinates of the validation point and its predicted value.

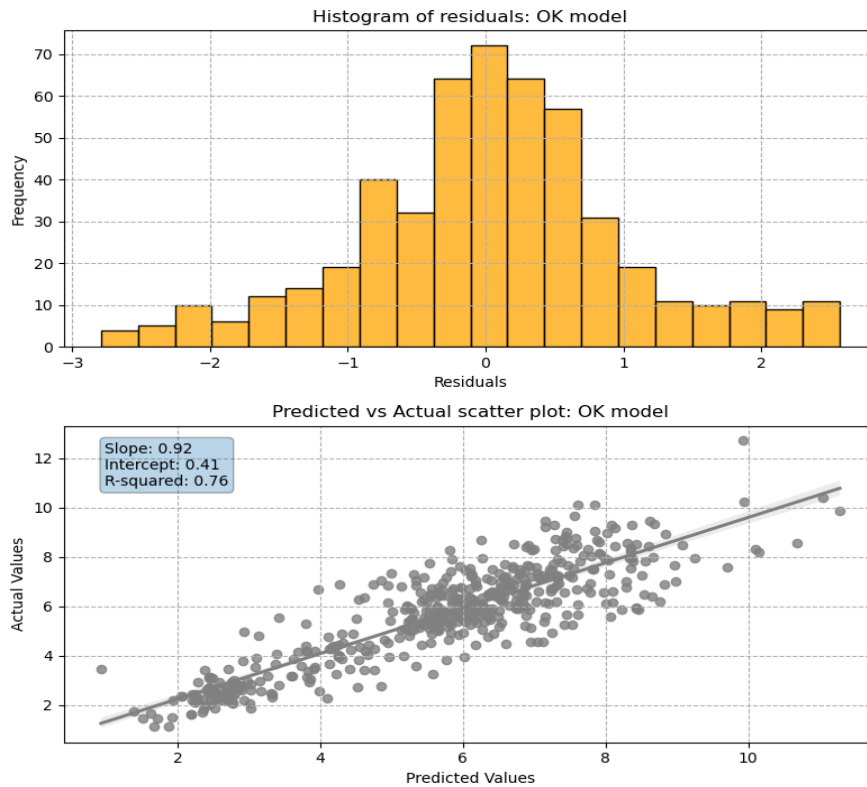


Figure 5-2: Histogram of OK residuals (top), Scatterplot of predicted vs actual PGE grades

Observing Figure 5-2, the histogram's central alignment around zero signals that the OK model neither systematically overpredicts nor underpredicts the PGE values, indicating a lack of bias in the model. The symmetrical distribution of residuals alludes to the model's prediction errors being normally distributed—a favourable attribute suggesting that the errors are stochastic rather than systematic.

The span of the histogram reveals the range of the residuals, with the majority clustered between -2 and 2, signifying that most predictions by the model are proximate to the actual values albeit with some variability. This dispersion reflects the model's precision while the tails, particularly the modestly elongated right tail, denote occasional underprediction.

The scatter plot in Figure 5-2 would ideally show all points aligned along a diagonal if the predictions were flawless. The linear pattern of the point spread demonstrates a linear congruence between the model's forecasts and the actual values, indicative of the model's accuracy. The regression line, characterized by a slope of 0.92 and an intercept of 0.41, implies that a unit rise in the predicted value corresponds to a 0.92 g/t escalation in the actual value—slightly diverging from the optimal slope of 1, suggesting a conservative estimation of higher values.

An R-squared value of 0.76 implies that the model explains 76% of the variability in the dataset, a robust figure reflecting the model's reliable predictive performance. The concentration and layout of the data points around the regression line provide further insight into the consistent predictive quality of the model.

Figure 5-3 suggests that the LR model demonstrates modest predictive capacity, with a broad scatter of data points indicative of an inadequate capture of variance, corroborated by a minimal R-squared value of 0.12. The histogram of residuals hints at normality but reveals a minor skew, with a higher concentration of negative residuals. Figure 5-4 reveals a more centred distribution of residuals for the SVR model relative to the LR model, suggesting enhanced model performance. Although the scatter of data points remains, they exhibit greater density around the line of best fit, reflecting a superior R-squared value of 0.4. Figure 5-5 shows that the DT model's residuals are tighter and more centrally aligned around zero, signalling heightened predictive precision. The points adhere more closely to the line of best fit, illustrated by an increased R-squared of 0.72.

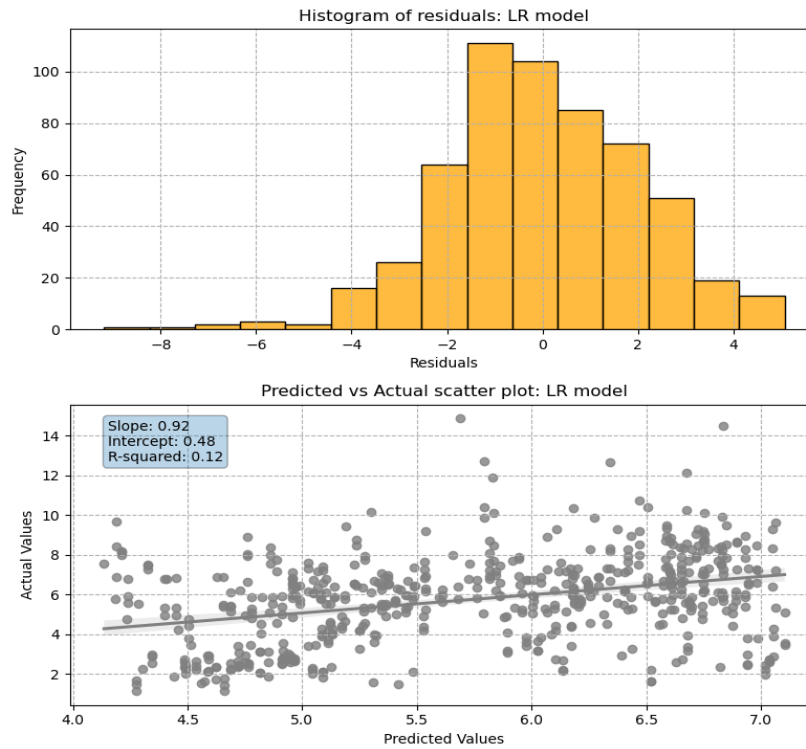


Figure 5-3: Histogram of LR residuals (top), Scatterplot of predicted vs actual PGE grades

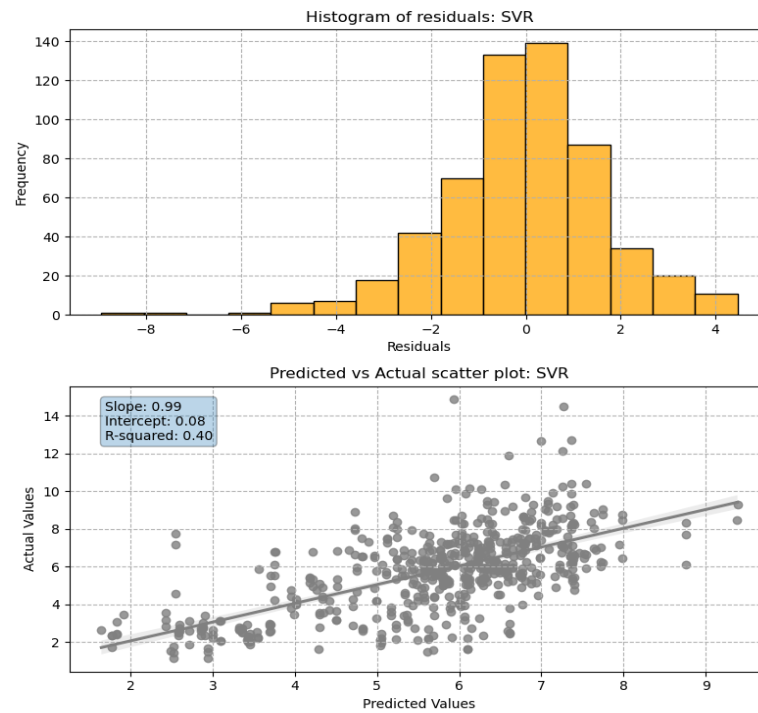


Figure 5-4: Histogram of SVR residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)

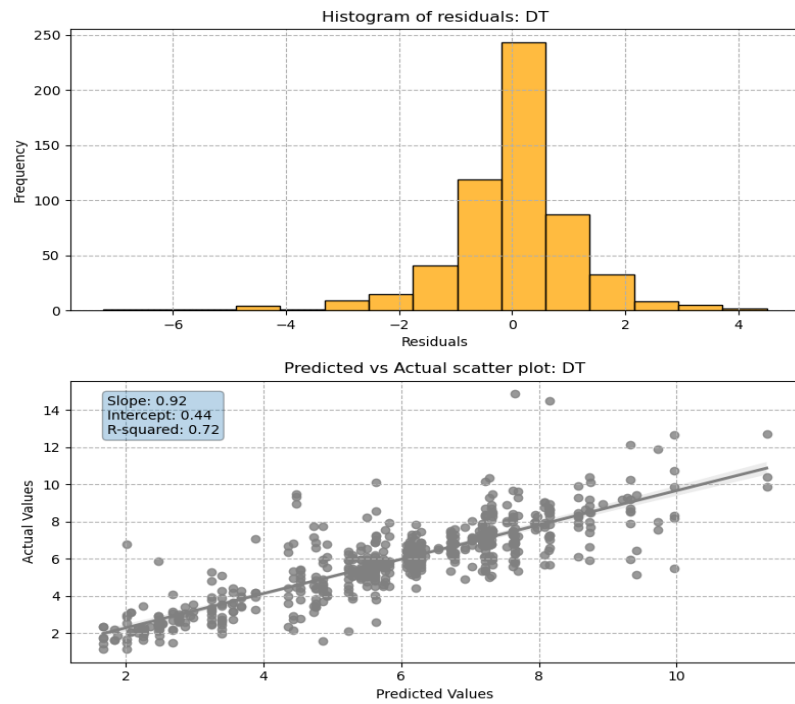


Figure 5-5: Histogram of DT residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)

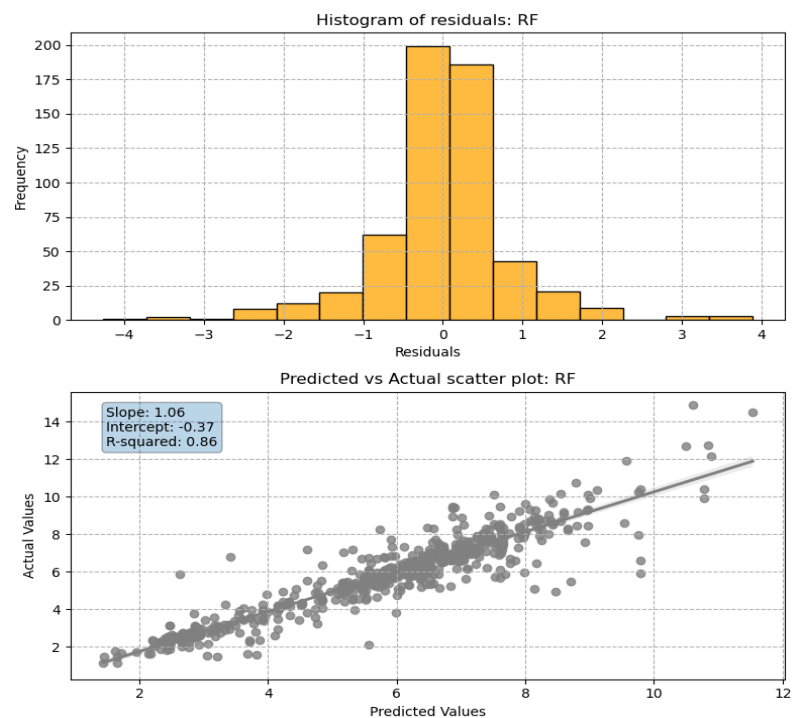


Figure 5-6: Histogram of RF residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)

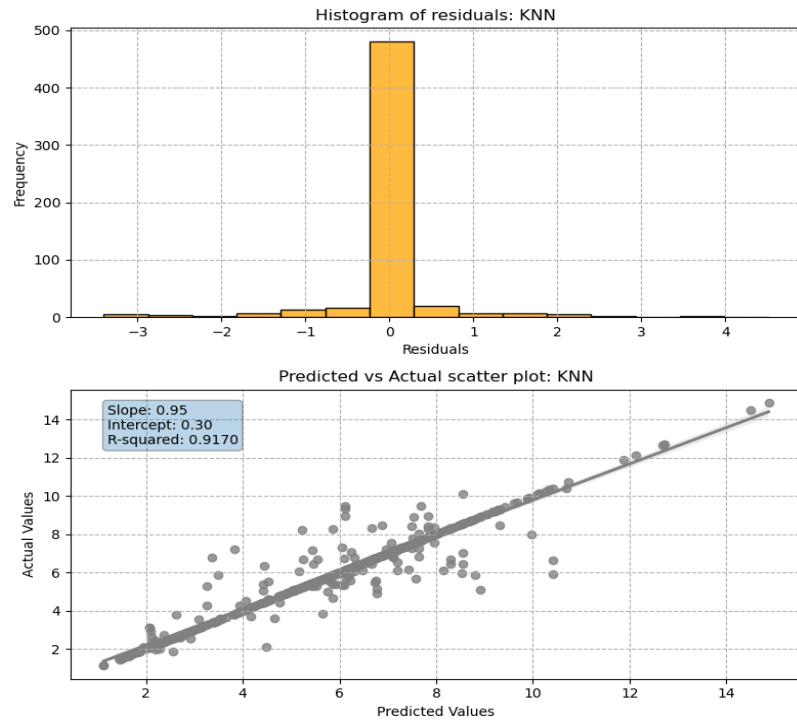


Figure 5-7: Histogram of KNN residuals (top), Scatterplot of predicted vs actual PGE grades (bottom)

Figure 5-3 suggests that the LR model demonstrates modest predictive capacity, with a broad scatter of data points indicative of an inadequate capture of variance, corroborated by a minimal R-squared value of 0.12. The histogram of residuals hints at normality but reveals a minor skew, with a higher concentration of negative residuals. Figure 5-4 reveals a more centred distribution of residuals for the SVR model relative to the LR model, suggesting enhanced model performance. Although the scatter of data points remains, they exhibit greater density around the line of best fit, reflecting a superior R-squared value of 0.4. Figure 5-5 shows that the DT model's residuals are tighter and more centrally aligned around zero, signalling heightened predictive precision. The points adhere more closely to the line of best fit, illustrated by an increased R-squared of 0.72.

Transitioning to Figure 5-6, the RF model reveals the most focused and centralised residual distribution among the four models, indicating the most precise predictions. Despite a slope marginally exceeding unity at 1.06, suggesting minor over-predictions, the high R-squared value of 0.86 demonstrates the RF model's robustness in explaining a substantial portion of the variance.

When examining the k NN model as shown in figure 5-7, it presents a striking improvement in predictive performance compared to the previous models. The residuals are almost perfectly centred, with an exceptional narrowing of the distribution, implying superior accuracy in the point predictions of the PGE grades. The data points align almost seamlessly with the line of best fit, underscored by an impressive R-squared value of 0.92, indicating that the k NN model captures much of the variance in the data.

Table 5-3: Summary Result of estimated grade and error metrics

Model	Comparative Metrics				
	Estimated Grade (g/t)	RMSE	MSE	MAE	R-squared
OK	5.69	0.996	0.992	0.748	0.76
DT	5.77	1.180	1.393	0.775	0.72
SVR	5.72	1.709	2.921	1.293	0.40
RF	5.77	0.846	0.715	0.551	0.86
LR	5.75	2.061	4.247	1.629	0.12
k NN	5.76	0.644	0.415	0.211	0.92

From table 5-3 it is evident that all models estimate the mean grade within a narrow range around the observed mean of 5.76 g/t determined in chapter 4. OK model estimates the grade at 5.69 g/t , demonstrating competent predictive capabilities with a RMSE of 0.996, MSE of 0.992, MAE of 0.748, and an R-squared value of 0.76, which indicates a decent fit to the data.

The DT model slightly overestimates the grade at 5.77 g/t , with higher error metrics (RMSE of 1.180, MSE of 1.393) and a slightly reduced R-squared value of 0.72 compared to OK, implying a reliable model with a higher prediction error. The SVR provides an estimated grade of 5.72 g/t but accompanies higher errors (RMSE of 1.709, MSE of 2.921, MAE of 1.293) and a lower R-squared value of 0.40, indicating a weaker predictive model relative OK and DT.

The RF model also slightly overestimates the grade, estimating a grade equivalent to the observed value at 5.77 g/t and showcasing the lowest error values among its counterparts (RMSE of 0.846, MSE of 0.715, MAE of 0.551) alongside a high

R-squared of 0.86. This demonstrates that the RF model captures a significant amount of variance in the data, making it a robust predictor of the PGE grade.

LR, while offering an estimated grade close to the observed value at 5.75 g/t, registers the highest prediction errors (RMSE of 2.061, MSE of 4.247, MAE of 1.629) and the least R-squared value of 0.12, signifying a less accurate fit to the observed data.

Remarkably, the k NN model predicts the exact observed grade with 5.76 g/t and stands out with the most favourable error measurements (RMSE of 0.644, MSE of 0.415, MAE of 0.211) and the highest R-squared value of 0.92, suggesting an exceptional fit and the highest predictive precision among the assessed models. This points to k NN's superior effectiveness in accurately forecasting point ore grades, outperforming the other models in terms of accuracy and reliability.

5.4 Optimum Prediction parameters

The notion that kriging is a 'minimum variance estimator' holds true only when the neighbourhood is properly defined (Vann et al., 2003). This has attributed to the importance of a quantitative kriging neighbourhood analysis (QKNA). QKNA is tasked with determining the optimum combination of estimation parameters that result in unconditional biasness during kriging (Chanderman et al., 2017). The estimation parameters namely, block size, number of informing samples, search range and discretisation points are evaluated based on two conditional bias statistics: kriging efficiency (KE) and slope of regression (SLOR). KE quantifies how well the kriging estimate replicates the local block grade whilst SLOR summarises the extent to which high and low grades have been overly smoothed.

Like the grid search process described in chapter 4. A custom function named *grid_search_kriging* (see appendix A4) was built to find optimal OK parameters of best interpolation performance from the *geostatspy.geostats.kb2d* function (see table 5-4). It should be noted that to accommodate the three structures used in the model variogram the *kb2d* function was adapted and renamed *kb2d_3*.

Table 5-4: *grid_search_kriging* Parameters and description

Parameter	Description
<i>df</i>	The input dataframe containing the spatial data
<i>col, ycol, vcol</i>	Column names in the dataframe representing the spatial coordinates (x, y) and the variable of interest
<i>tmin, tmax</i>	The minimum and maximum trimming limits
<i>nx, ny</i>	Number of grid cells along x and y directions
<i>xmn, ymn</i>	Minimum values of x and y coordinates
<i>xsiz, ysiz</i>	Cell size in x and y directions
<i>ktype, skmean, vario</i>	Parameters defining the variogram model

The search parameters in the *grid_search_kriging* function is systematically varied to explore different combinations and find the optimal configuration for the kriging process.

Table 5-5: Grid search Parameters and descriptions

Search Parameter	Description
<i>search_radii</i>	A list of search radii for the search neighbourhoods in the kriging process.
<i>ndmin_values, ndmax_values</i>	Lists of values representing the minimum and maximum number of data points required for estimation
<i>nxdis_values, nydis_values</i>	Lists of values representing the maximum number of discretisation points used in the horizontal and vertical directions, respectively, for calculating the kriging weights

The *grid_search_kriging* function iterates through all combinations of the search parameters using nested loops. For each combination, it performs the following steps:

- I. It conducts kriging interpolation (with *geostats.kb2d_3*) using the parameter values described in table 5-4 and 5-5
- II. Calculates performance metrics; SLOR and KE using the *calculate_performace_metric* function
- III. Evaluates the overall performance using the *custom_scoring* function
- IV. Updates the best performance and parameters if the current combination yields better results than the previous best

- V. The function returns the best combination of parameters along with the corresponding best performance score

Table 5-6: *calculate_performance_metric* function Parameters

Parameter	Description
<i>predicted_results</i>	Predicted values obtained from OK
<i>cbb</i>	The block variance
<i>kriging_variance</i>	Variance associated with the kriging predictions
<i>s</i>	The Lagrange multiplier, often derived during the kriging process
<i>unest</i>	Value used to represent undefined or missing kriging variances

The *calculate_performance_metric* function, whose parameters are detailed in table 5-6 calculates the two-performance metrics SLOR and KE. KE is given by the equation:

$$\frac{\sigma_B^2 - \sigma_K^2}{\sigma_B^2} \quad (5.1)$$

Where σ_B^2 represents the block variance and σ_K^2 the kriging variance. A KE of 1 indicates a perfect match between the estimated grade and true grade distribution. Clustering or sparse data can see a drop in KE often resulting in negative KE signalling extremely poor estimates (Coombes, 2008).

The implication of SLOR which was alluded to briefly earlier on is given by the equation:

$$\frac{\sigma_B^2 - \sigma_K^2 - \lambda}{\sigma_B^2 - \sigma_K^2 - 2\lambda} \quad (5.2)$$

Where λ represents the Lagrange multiplier, which ensures that OK weights are optimal and add up to 1. A SLOR of 1 implies that the estimated low and high grades correspond to those of the observed data (Coombes, 2008).

The *custom_scoring* function is responsible for quantifying the overall performance of the kriging predictions by combining the KE and SLOR metrics with appropriate weighting and penalties (see figure 5-8). KE and SLOR are evaluated within the ideal range of [0, 1], a penalty of -1 was applied to values outside [0,1] with the desired value retained within the ideal range. For the Project X data, a weighting of 0.85 for KE and 0.15 for SLOR was used, as accuracy was considered the primary concern as it relates to decisions based on predicted values. The combined score was normalized to the [0, 1] range to ensure consistency and comparability. Adjustments are made to account for penalties so that the resulting overall score reflects the quality of the kriging predictions.

```
def custom_scoring(KE, SLOR):
    # Initialize score components
    ke_score = 0
    slor_score = 0

    # KE Scoring: Apply stronger penalties for values outside the [0, 1] range
    if 0 < KE <= 1:
        ke_score = KE # Within the ideal range
    else:
        ke_score = -1 if KE <= 0 or KE > 1 else KE # Apply a stronger penalty for values outside [0, 1]

    # SLOR Scoring: Apply stronger penalties for values outside the [0, 1] range
    if 0 < SLOR <= 1:
        slor_score = SLOR # Within the ideal range
    else:
        slor_score = -1 if SLOR <= 0 or SLOR > 1 else SLOR # Apply a stronger penalty for values outside [0, 1]

    # Combine scores with equal weighting
    overall_score = 0.85 * ke_score + 0.15 * slor_score

    # Normalize to 0 to 1 range
    overall_score = (overall_score + 1) / 2 # Adjust scores from [-2, 1] to [0, 1]
    overall_score = max(0, min(1, overall_score)) # Ensure within [0, 1]

    return overall_score
```

Figure 5-8: custom_scoring function source code

5.4.1 Kriging Block Prediction

The boundary seen in figure 5-9 represents the extent of the measured PGE resource within the Project X lease area.

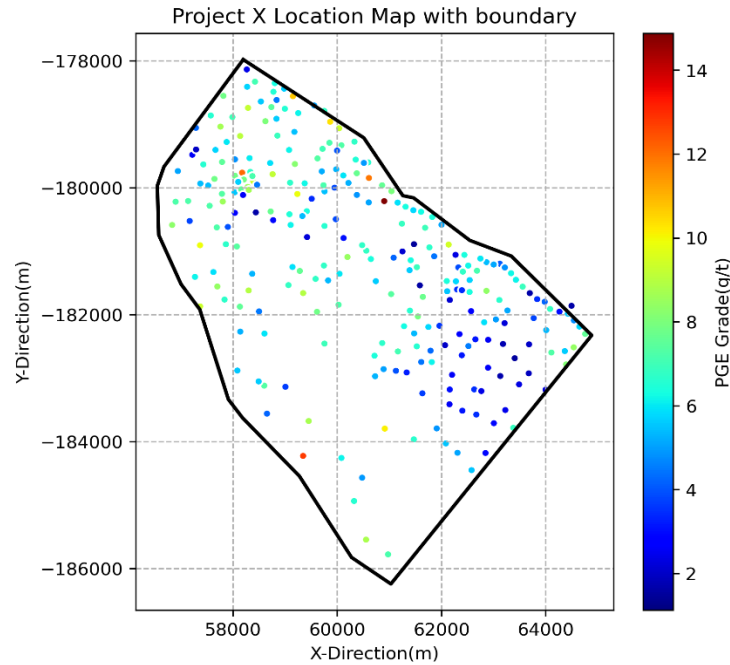


Figure 5-9: PGE borehole location with estimation boundary

The density of information required to identify the mineral resource for stratabound PGE is specified in technical reports published by publicly traded mining and exploration companies (Zientek et al., 2014). For deposits in South Africa, measured mineral resources are defined with holes spaced 250-300m apart (Zientek, et al., 2014). Based on this information a $250 \times 250m$ block model was chosen to for the PGE resource with the model origin being the minimum X and Y values of the boundary points.

The computational power available for an expansive coverage of the search parameter space was limited.

```
search_radii = [3400,4000]
ndmin_values = [3,5,8,12]
ndmax_values = [10,15,20]
nxdis_values = nydis_values = [2,4]
```

Figure 5-10: OK Grid Search parameter space

The chosen parameter space in figure 5-10 reflects a balanced approach aimed at exploring a range of model configurations from more conservative (smaller search radius and fewer samples) to more inclusive (larger search radius and more samples). The inclusion of a relatively small number of discretisation points represents a focus on computational efficiency while still attempting to capture necessary spatial variability.

Trials conducted with discretisation points below two and above four were found to produce suboptimal results at an added cost of increased computational time. The search space used was well-suited to identifying a kriging model configuration that balanced the need for computational efficiency with the desire for predictive accuracy and reliability. The total number of parameter combinations is the product of the number of options for each parameter. The total of 96 combinations was significant and manageable with modern computing resources, which allowed for a thorough exploration of the parameter space.

Table 5-7 provides information regarding the optimized block model parameters and the best OK results obtained used the *grid_search_kriging* function. The high number of blocks suggests that the kriging model was able to produce estimates for a vast majority of the grid cells of the project area. From chapter 4, through EDA, the average grade of the PGE data was 5.76 *g/t*. The estimated grade was slightly higher at 6.72 *g/t* and suggests that the kriging process, which leverages spatial correlation among PGE data points, predicts a higher concentration of the PGE within the measured resource boundary of the lease area, than the simple average of observed grades. This further indicates that the spatial distribution of higher-grade PGE has a significant influence on the surrounding blocks, leading to an overall increase in the estimated grade.

The block variance is the difference total sill and the variance contained within a block. A low block variance of 0.109 (*g/t*)² indicates less variability and more homogeneity within blocks. An OK variance of 0.007 (*g/t*)² is significantly lower than the block variance, illustrating the reduction in uncertainty or variability in the grade estimates because of the kriging process. The decrease in variance is a direct outcome of the spatial interpolation and the use of neighbouring sample

data, showcasing the effectiveness of the kriging method in reducing estimation errors.

Table 5-7: OK optimised block model parameters and Best OK results

Origin	X = 5655.155m	Y = -186244.915m
Block size	250 × 250m	
Cell size	20 × 20m	
Search Radius	3400m	
Min/Max Samples	Min = 12	Max = 15
Discretisation points	X = 2	Y = 4
Best OK results		
Number of valid blocks	62127 blocks	
Estimated grade	6.79 g/t	
Block Variance	0.109 (g/t) ²	
OK Variance	0.007 (g/t) ²	
Lagrange Multiplier	0.00015	
Overall Score	0.973	

When taking the sign and magnitude of the Lagrange multiplier into consideration, a negative Lagrange multiplier, suggests that the constraint (the unbiasedness condition) is being actively enforced to correct for a tendency in the kriging system. It might also indicate that without this adjustment, the system could potentially overestimate the values (Srivastava & Isaaks, 1989). The small magnitude (0.00015) indicates that the adjustment needed to enforce unbiasedness is minor, A small adjustment is sufficient to ensure that the kriging predictions are, on average, unbiased.

5.4.2 ML Block Prediction

The approach outlined in the code snippet generally aligns with the concept of spatial interpolation using machine learning, as discussed in the context of spatial prediction and block modelling in chapter 3.

```
# Grid dimensions
total_blocks_x, total_blocks_y = 250, 250 # Total number of blocks in x and y directions
block_size = 33 # Size of each block (assuming square blocks)
discretization_per_block = 2 # Discretization points per block dimension

# Calculate total grid size
grid_size_x = total_blocks_x * block_size
grid_size_y = total_blocks_y * block_size

# Calculate discretization point spacing within a block
dis_spacing = block_size / discretization_per_block

# Generate discretization points
# Note: Adjust xmin, xmax, ymin, ymax based on the actual area you want to cover
xmin, xmax = df_measured['X'].min(), df_measured['X'].max()
ymin, ymax = df_measured['Y'].min(), df_measured['Y'].max()

# Adjusting to cover the full extent of the grid if df_measured is Limited
# Here, you might need to manually set xmin, xmax, ymin, ymax to cover the desired area fully

dis_points_x, dis_points_y = [], []

for x_block in range(total_blocks_x):
    for y_block in range(total_blocks_y):
        for dx in range(discretization_per_block):
            for dy in range(discretization_per_block):
                # Calculate the coordinates of each discretization point
                x_coord = xmin + x_block * block_size + dx * dis_spacing + dis_spacing / 2
                y_coord = ymin + y_block * block_size + dy * dis_spacing + dis_spacing / 2
                dis_points_x.append(x_coord)
                dis_points_y.append(y_coord)

# DataFrame for ML predictions
df_dis_points = pd.DataFrame({'X': dis_points_x, 'Y': dis_points_y})
```

Figure 5-11: Grid dimension setup for ML block predictions

Setting up the grid dimensions involved creating a structured spatial framework on which to perform ML predictions. This process can be observed in figure 5-11. The total number of blocks in both the x (*total_blocks_x*) and y (*total_blocks_y*) directions is set to 250m. The number of discretisation points within each block dimension was set to two (*discretization_per_block*) because finer microblocks (more discretisation points) do not necessarily increase macroblock accuracy, making the ML estimation process less computationally expensive. Discretisation Point Spacing (*dis_spacing*) was determined by dividing the block size by the number of discretisation points per block.

The code in figure 5-11 iterates through each block and each discretisation point within that block to calculate the coordinates of each point. The calculation accounts for the position of the block within the grid and the position of the point within the block, ensuring that each point is accurately placed. Discretisation points

are calculated such that there is a half-step offset ($dis_spacing / 2$) from the block's edge to the first point, ensuring that points are centered within their respective discretisation intervals.

After generating the x and y coordinates for all discretisation points, these were compiled into a DataFrame (*df_dis_points*). *df_dis_points* serves as the basis for subsequent machine learning predictions, providing a structured way to associate predictions with specific spatial locations.

5.4.2.1 Adding Machine Learning predictions

For each ML model, an array of predictions was made and aligned in sequence with the discretisation points. This means each prediction corresponds directly to the spatial location with the same index of a DataFrame.

```
# Assuming dis_point_predictions are in the same order as the discretization points were generated
# And assuming df_dis_points has the same order and length as dis_point_predictions

# Add predictions to the DataFrame
df_dis_points['PredictionsSVR'] = dis_point_predictionssvr

df_dis_points['block_x_index'] = ((df_dis_points['X'] - xmin) // block_size).astype(int)
df_dis_points['block_y_index'] = ((df_dis_points['Y'] - ymin) // block_size).astype(int)

# Aggregate predictions by block
block_predictions = df_dis_points.groupby(['block_x_index', 'block_y_index'])['PredictionsSVR'].mean().reset_index()
block_predictions.rename(columns={'PredictionsSVR': 'BlockPredictionsSVR'}, inplace=True)
```

Figure 5-12: Python code for block aggregation for SVR predictions

For demonstrative purposes, figure 5-12 shows the process of adding ML predictions to the *df_dis_points* DataFrame for SVR. *df_dis_points['PredictionsSVR'] = dis_point_predictionssvr* adds the predictions as a new column to *df_dis_points*, associating each discretisation point with a predictive value.

Aggregating predictions at block level required each discretisation point to be associated with a block. The expressions $((df_dis_points['X'] - xmin) // block_size).astype(int)$ and $((df_dis_points['Y'] - ymin) // block_size).astype(int)$ calculate the x and y block indices, respectively. Subtracting $xmin$ or $ymin$ from the coordinates aligns the grid to a 0-based index system. Dividing by *block_size* determined the block's position in the grid and applying floor division (*//*) converts this to an integer block index. These indices

were then added to *df_dis_points* as new columns (*block_x_index* and *block_y_index*).

The final step is to aggregate (average) the predictions within each block. which groups the DataFrame by block indices and calculates the mean prediction for all points within each block. The grouped data was transformed into a new DataFrame (*block_predictions*) with columns indicating the block index and the aggregated prediction (*BlockPredictionsSVR*) for that block.

Table 5-8: ML model blocks and block estimate

Model	No of blocks	Block Estimates (<i>g/t</i>)
LR	62 500	5.56
SVR	62 500	7.88
DT	62 500	6.61
RF	62 500	6.39
KNN	62 500	6.25

The block estimates from the ML models in table 5-8 when compared with the OK block model and the observed mean, show variations that reflect differences in each model's prediction capabilities and possibly their sensitivity to the spatial structure of the data. The LR model estimates the grade at 5.56 *g/t*, which is noticeably below the mean grade of 6.97 *g/t* reported by the OK model with its 62127 blocks. This lower estimate by the LR model may suggest a conservative approach, potentially underestimating the grade when compared with the comprehensive OK model.

The SVR model, on the other hand, has the highest block estimate at 7.88 *g/t*. This figure significantly exceeds both the observed mean grade and the OK model's estimate, indicating that the SVR model may be overly sensitive to certain aspects of the data, leading to an overestimation that could stem from model-specific hyperparameters or a susceptibility to outliers.

The DT model estimate stands at 6.61 *g/t*, closer to the OK model's estimation but still below the OK model's grade, suggesting a moderate overestimation when compared to the observed mean. This might be reflective of the DT model's ability

to capture complex patterns in the data, although it may also be incorporating some noise as part of its predictive process.

The RF model offers an estimate of 6.39 *g/t*, which situates it nearer to the observed mean and offers a balance among the provided estimates. This suggests that the RF model, with its ensemble approach, provides a stable estimate that appreciably captures the underlying data structure without majorly overfitting or underfitting, though it is still below the OK model's grade.

Finally, the *k*NN model gives an estimate of 6.25 *g/t*, which is conservative relative to the SVR and DT estimates and slightly under the OK model's grade. This positioning indicates that the KNN model, with its emphasis on local data structure, potentially offers a cautious estimation that closely mirrors the central tendency of the observed grades without reaching the higher grade estimation presented by the OK model. The KNN's performance, with the closest estimate to the OK model's grade among the ML models, suggests a robust predictive capability that may strike an optimal balance between the granularity of local variations and the global trend captured by the OK model.

5.5 ML and OK model validation and comparison: Block Predictions

Grade estimates from all models should be evaluated (see table 5-9) not just on the mean grade estimate but also on how well it reflects the true distribution and spatial variability of the grades. The observed mean provides a baseline for comparison. By comparing estimates to actual values on a block-by-block basis, the local accuracy of the models can be assessed. Block-wise comparison enables statistical analysis of the estimation accuracy. For each block, error metrics were calculated to quantify the performance of each model

Table 5-9: Summary results of OK and ML block model error metrics

Model	RMSE	MSE	MAE	R-squared
OK	2.176	4.437	1.594	0.156
LR	2.060	4.245	1.628	0.123
SVR	1.712	2.930	1.294	0.395
DT	1.342	1.803	0.945	0.636
RF	0.085	1.177	0.756	0.759
<i>k</i> NN	0.884	0.782	0.574	0.838

The OK model, with 62,127 blocks and an average grade of 6.97 g/t, establishes a reference with a RMSE of 2.176, MSE of 4.437, MAE of 1.594, and an R-squared of 0.156. These figures suggest that while the OK model has provided a robust baseline, its predictive accuracy is surpassed by ML models.

The LR model reports slightly improved error metrics compared to OK, including an RMSE of 2.060 and MSE of 4.245, though its R-squared value at 0.123 is lower, indicating less variance explained by the model.

SVR presents better error metrics RMSE of 1.712, MSE of 2.930, MAE of 1.294) and a higher R-squared value of 0.395 compared to both OK and LR, suggesting a more accurate prediction capability.

DT model shows further improvement, with an RMSE of 1.342, MSE of 1.803, and MAE of 0.945, alongside an R-squared value of 0.636, indicating a substantial grasp of the data's variance, and better performance in grade estimation.

RF model displays a marked superiority in error reduction, with the lowest RMSE of 0.085, MSE of 1.177, and MAE of 0.756 among the compared models, along with an R-squared of 0.759, suggesting a high level of accuracy in the grade predictions.

Once again, the KNN model stands out with the lowest error metrics across the board, presenting an RMSE of 0.884, an MSE of 0.782, and an MAE of 0.574, combined with the highest R-squared value of 0.838. This indicates an exceptional predictive accuracy, reflecting the KNN model's capacity to closely mirror the actual grade distribution. *k*NN's robust performance demonstrates its efficiency in capturing the complex spatial relationships within the data, suggesting it could be a potent tool for block grade estimation in mining operations.

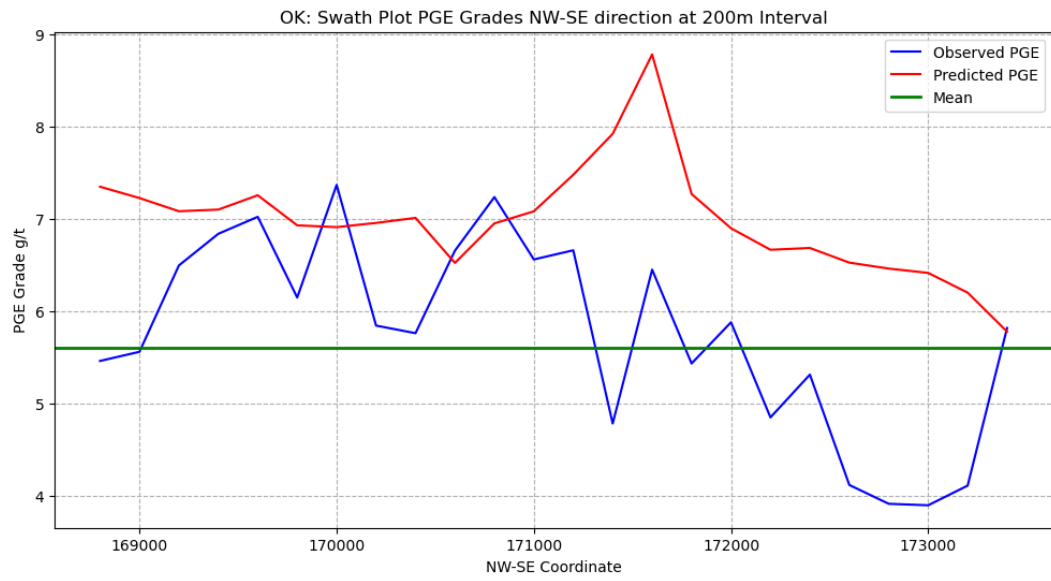


Figure 5-13: OK swath plot at 200m intervals

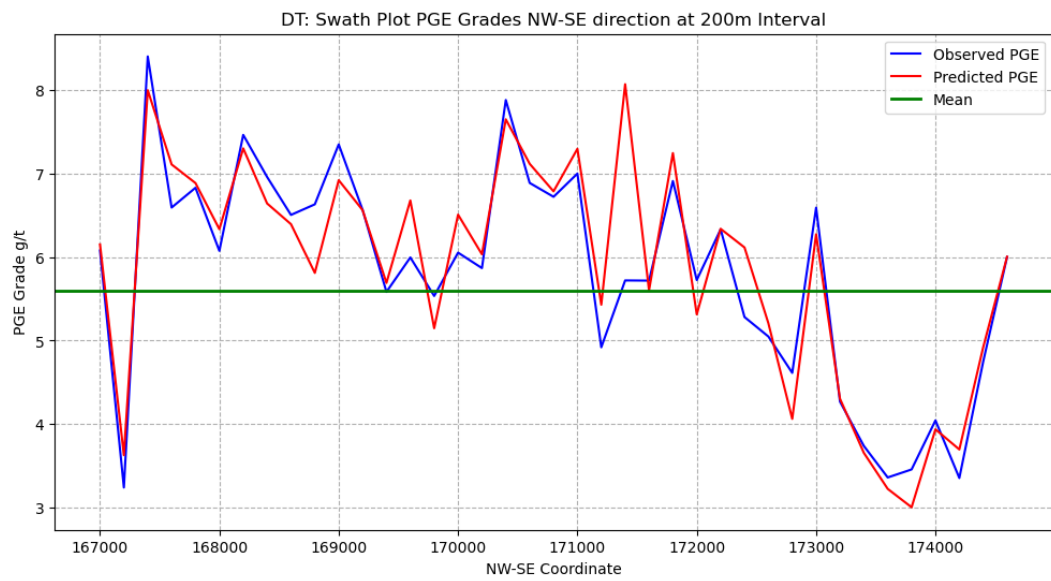


Figure 5-14: DT swath plot at 200m intervals

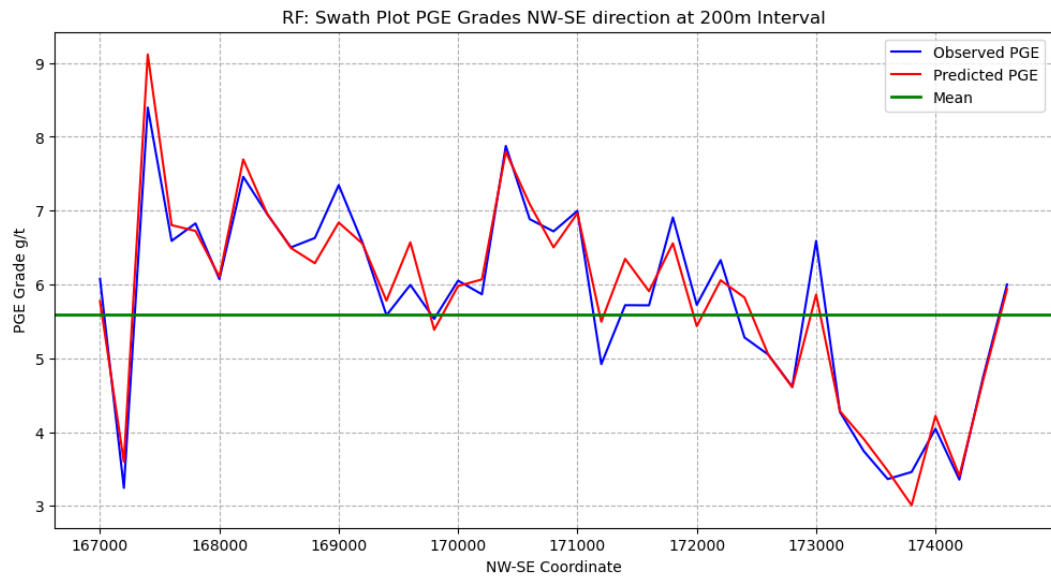


Figure 5-15: RF swath plot at 200m intervals

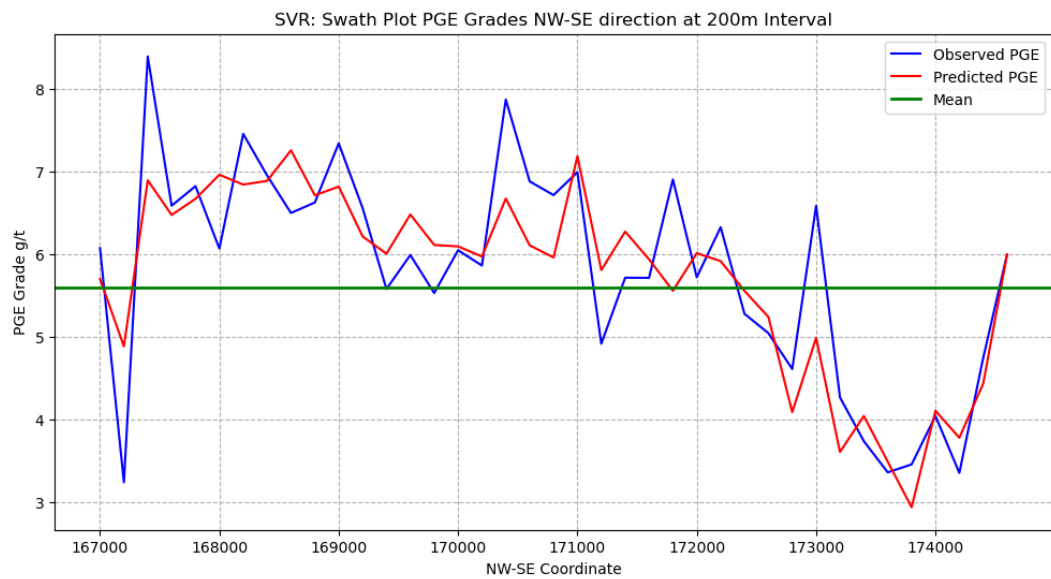


Figure 5-16: SVR swath plot at 200m intervals

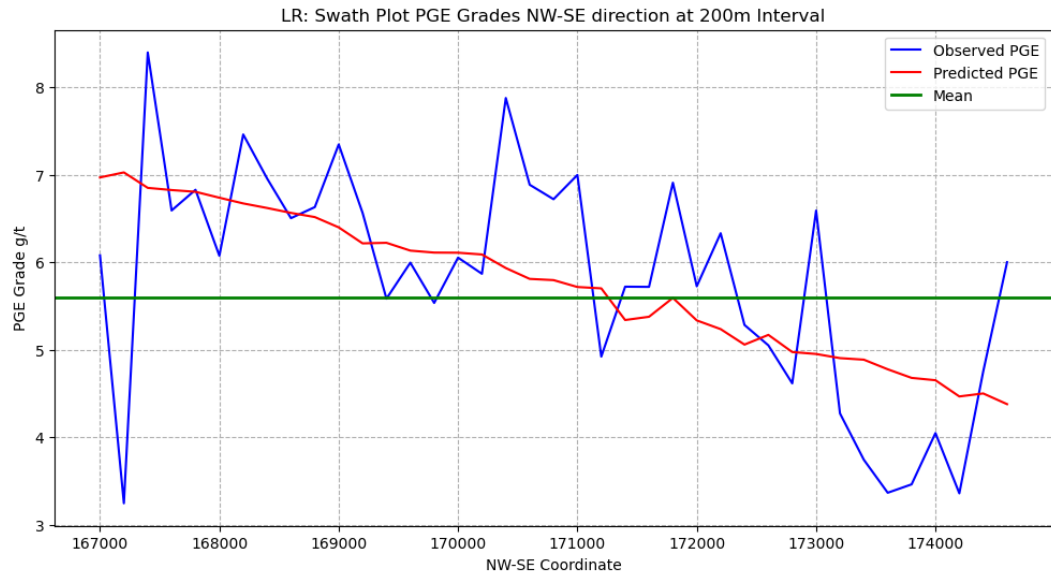


Figure 5-17: LR swath plot at 200m intervals

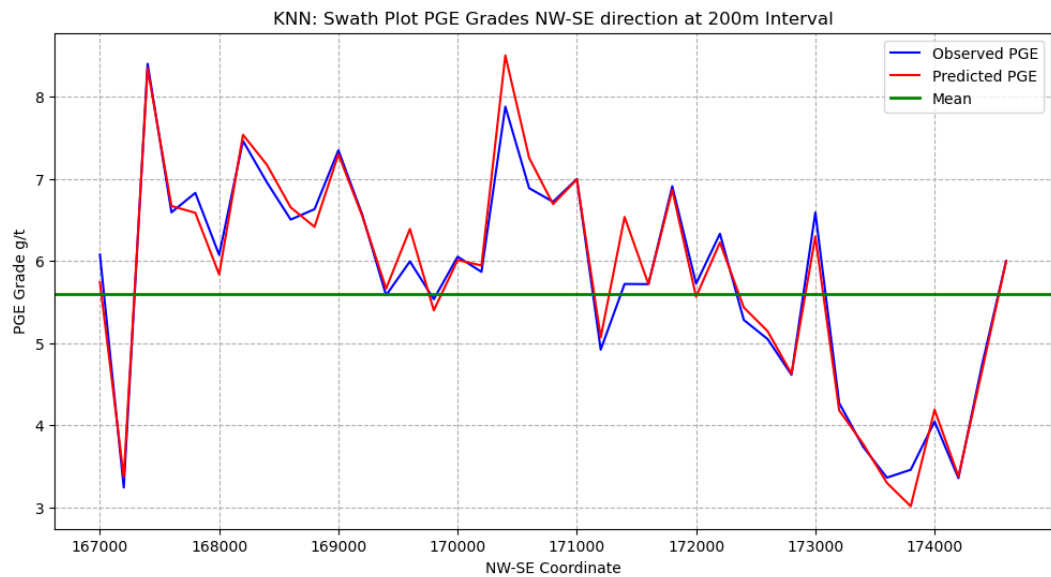


Figure 5-18: *k*NN swath plot at 200m intervals

The swath plots in Figures 5-13 to 5-18 provide a visual representation of the predicted and observed PGE grades along the NW-SE axis, which was acknowledged as the principal direction of continuity, sampled at 200m intervals. The plots incorporate a red line that delineates the observed PGE grades, a blue line for the predicted PGE grades by various models, and a green line that marks the mean PGE grade across all observations. The proximity of the blue to the red

line at any given point on the x-axis serves as an indicator of the model's predictive precision in that specific locale.

In Figure 5-13, the OK model's predictions exhibit a notable deviation from the observed values, not fully capturing the observed data's innate fluctuations, aligning with the elevated error metrics associated with the OK model. DT model, depicted in Figure 5-14, presents an improved consistency between observed and predicted values, suggesting an enhanced capacity to adequately represent the data's variability. However, DT can manifest abrupt shifts in prediction, stemming from its inherent decision-rule-based structure which can cause sudden changes as data inputs transition across defined thresholds, resulting in the step-like pattern observed.

The RF model's swath plot in Figure 5-15 displays the most accurate alignment between predicted and observed values amongst the tree methods, correlating with its optimal error metrics. RF tends to average the decision boundaries of individual trees, which may overfit in isolation, hence yielding more accurate and generalisable predictions. Conversely, the SVR model, shown in Figure 5-16, grasps the overall trend within the data but not to the level of accuracy of DT and RF. The LR model's predictions in Figure 5-17 typically shadow the trend of the observed data; however, they do not capture all the fluctuations, possibly indicative of underfitting or overfitting, particularly in regions where the predictors and target variable exhibit a non-linear relationship.

The k NN swath plot, shown in Figure 5-18, reveals a highly accurate reproduction of the observed grade's profile, showcasing a blue line that often closely tracks the red line. The inherent nature of k NN, relying on the similarity of nearby data points for predictions, allows it to mirror local fluctuations effectively. The discrepancies that do occur may be due to k NN's sensitivity to the selection of k and the distance metric used, which can impact its ability to generalize between data points that are not closely clustered.

These swath plots provide a graphical understanding of each model's predictive behaviours, with full scatter plots and histograms of the actual vs. predicted

estimates to be included in appendix A (A5-A10), offering a similar layer of analytical depth into the model's performances.

5.6 Spatial analysis

Estimation maps provide a visual representation of the spatial distribution of predicted values across the project area. This helps in identifying spatial patterns, trends, and anomalies.

In figure 2-19, the LR estimate map shows a smooth and continuous gradient across the study area. This gradient reflects the predictions made by the linear regression model, where higher values are concentrated in a specific region of the map, gradually transitioning to lower values as you move across the area. The smoothness of the gradient indicates that the linear regression model is imposing a linear relationship on the data, where the change in the predicted variable is assumed to be a simple, constant rate across the spatial domain.

The actual data points exhibit much more variability, with localised pockets of high and low values scattered throughout the study area. These localised variations are indicative of complex spatial processes that are not well-represented by a linear model.

In figure 2-20 kNN estimate map displays a high degree of localised variation, characterised by distinct zones of high and low estimates. This pattern is a direct result of how the kNN algorithm operates: it estimates the value at a given point based on the average (or weighted average) of the values of its nearest neighbours. The influence of these neighbouring points leads to predictions that reflect the local data distribution closely, making kNN particularly effective at capturing spatial heterogeneity.

The reliance on the nearest neighbours can lead to sharp, abrupt transitions between zones of different predicted values, especially when there is a stark contrast between neighbouring data points. These sharp transitions may not always represent the true underlying spatial continuity of the data.

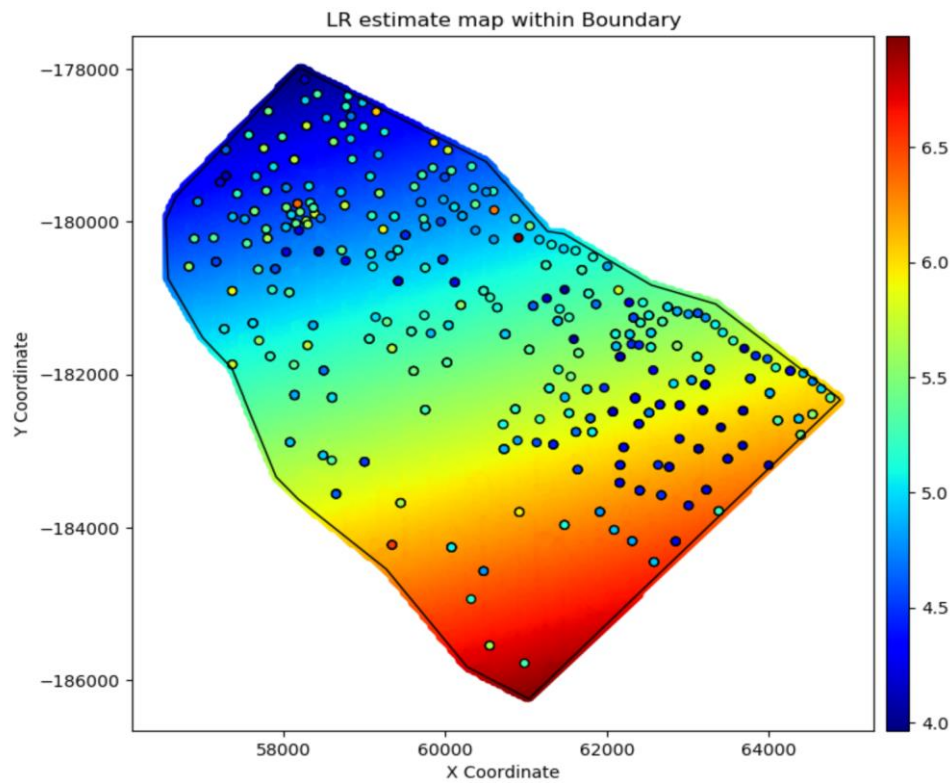


Figure 5-19: LR estimate map

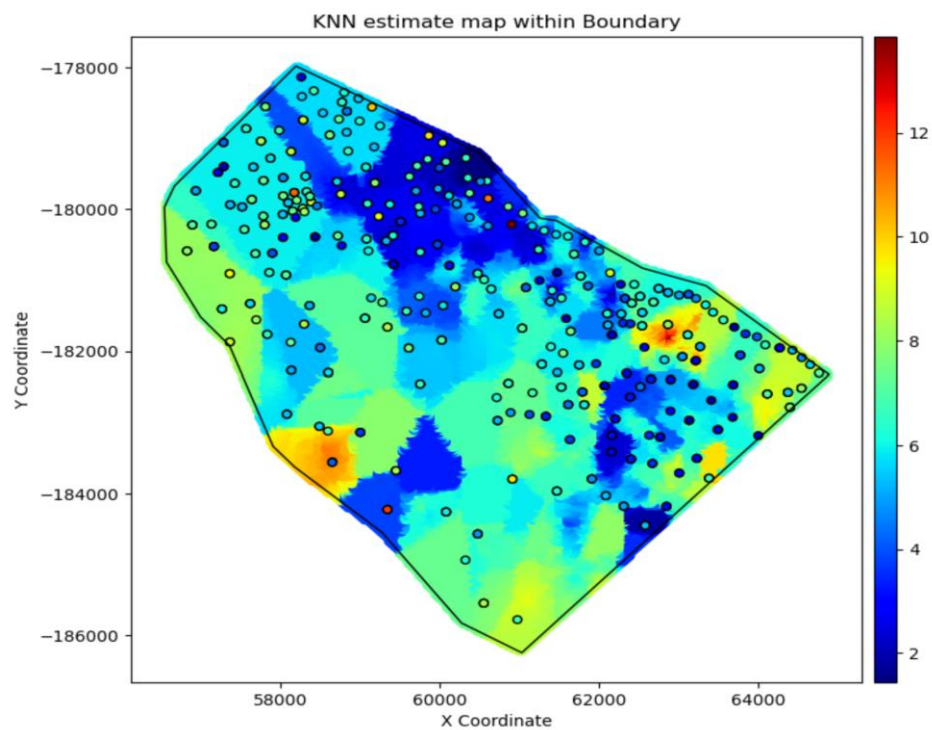


Figure 5-20: kNN estimate map

From figure 5-21, the DT estimate map is characterised by its cross-hatched appearance, where the predicted values are grouped into distinct regions with sharp, well-defined boundaries. This arises from the nature of the decision tree algorithm, which partitions the data space into a series of non-overlapping rectangles (or regions) based on the values of the predictor variables (in this case, the spatial coordinates X and Y). Each of these regions is assigned a single predicted value, which is typically the mean or median of the data points within that region.

The DT model captures some of the local variations in the data, it also introduces several artifacts—specifically, the artificial boundaries between regions of different predicted values. These boundaries are a direct result of the way the decision tree partitions the data space, and they may not correspond to any real features or trends in the actual data.

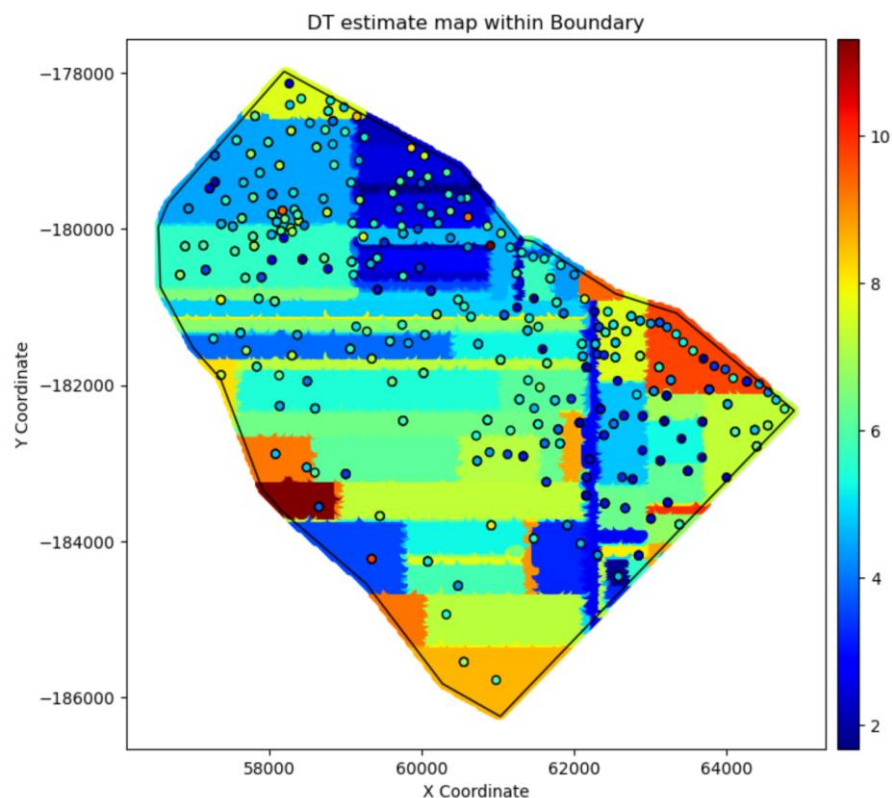


Figure 5-21: DT estimate map

Similarly in figure 5-22, the RF map also exhibits a crossed-hatch appearance, however, it shows a smoother and more continuous spatial variation compared to the DT map.

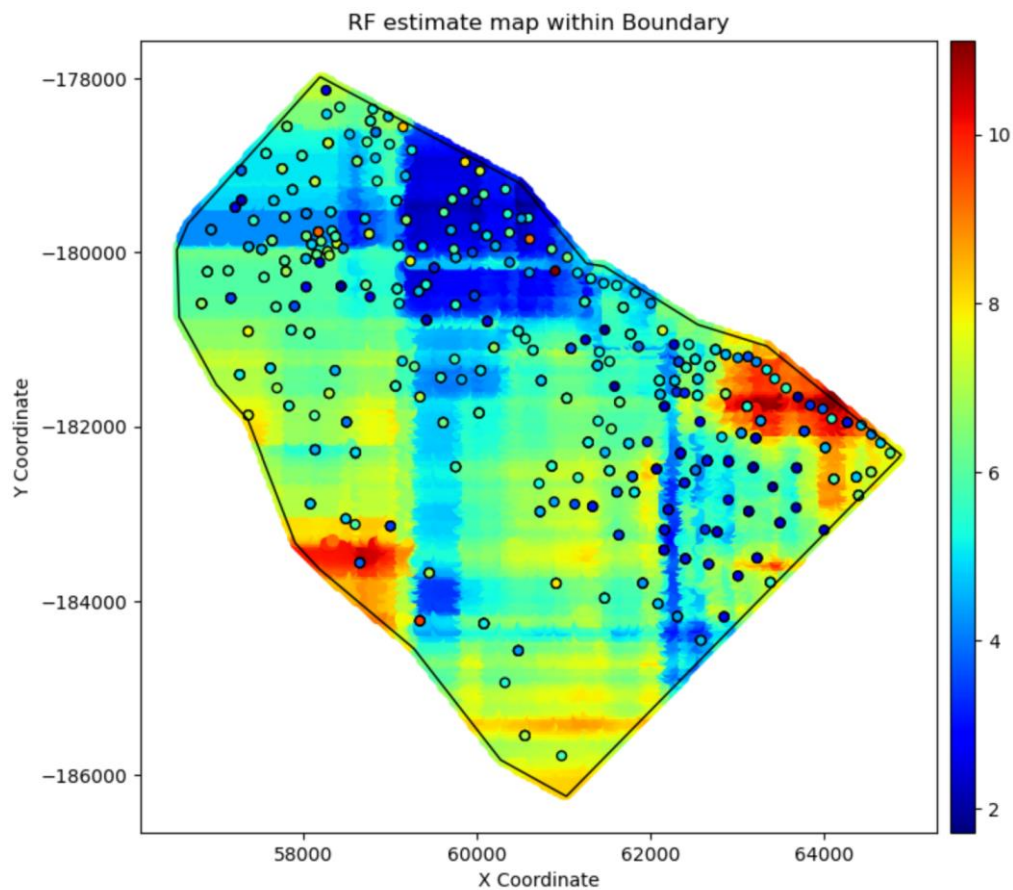


Figure 5-22: RF estimate map

The smoothness of the SVR map in figure 5-23 reflects the algorithm's ability to generalise the data, creating a continuous prediction surface that avoids overfitting to noise or minor fluctuations in the dataset.

When comparing the SVR map to the actual values at sample locations, it becomes clear that while SVR does an excellent job of capturing the broader trends in the data, it might miss some of the finer, localised variations—especially in areas where the data exhibits sharp changes over short distances. This limitation is inherent to the SVR approach, which prioritises smoothness and generalisation over capturing every small variation in the data.

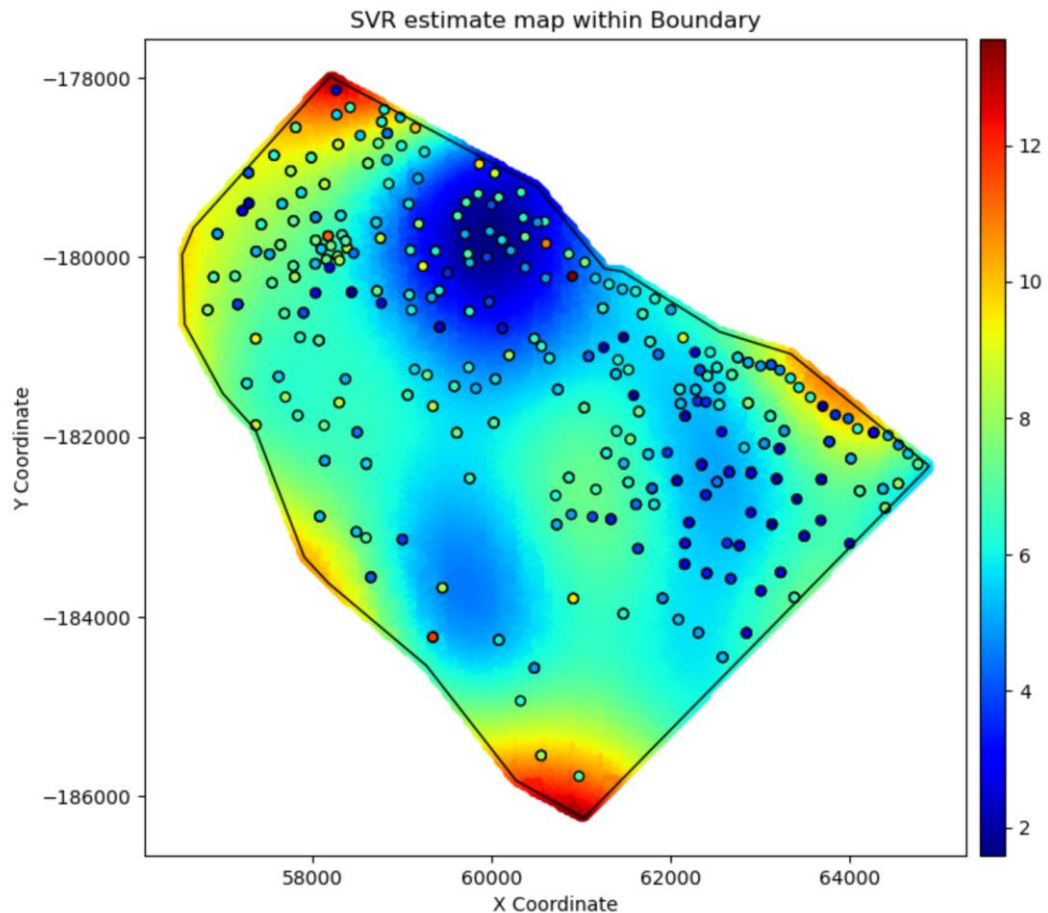


Figure 5-23: SVR estimate map

OK provides a smooth yet detailed surface, capturing both large-scale trends and local variations as seen in figure 5-24. OK explicitly models spatial autocorrelation, leading to a nuanced and statistically optimal prediction surface. OK offers the best balance between capturing global trends and local details, outperforming other methods in preserving spatial continuity.

Compared to other models, LR fails to capture local variations, resulting in a surface that is too smooth and doesn't reflect the actual complexity of the data. LR is most effective in scenarios with clear, global trends but is inadequate for capturing intricate spatial patterns.

SVR produces a smooth map with subtle non-linear trends, effectively capturing global patterns. The effectiveness of SVR heavily depends on the choice of kernel and hyperparameters.

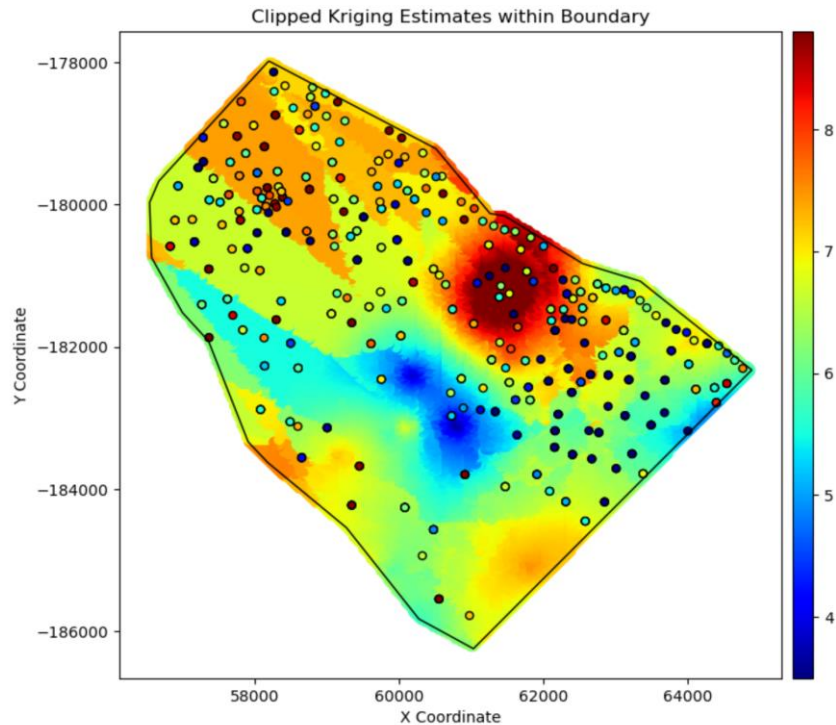


Figure 5-24: OK estimate map

kNN captures local detail effectively, showing distinct zones of high and low estimates based on the nearest neighbours' values. The kNN map exhibits abrupt changes between different zones, reflecting the algorithm's sensitivity to local data points. Unlike LR, kNN captures localised variations well but introduces sharp, unnatural boundaries, making the map appear less smooth compared to methods like SVR or OK.

RF balances the local detail of kNN with the smoother transitions of SVR and Kriging, making it more versatile. Despite being an improvement over DT, RF can still introduce some cross-hatched patterns or grid-like artifacts, particularly when using coordinates as features.

Compared to RF, DT, and kNN, OK is less prone to artifacts, making it the most reliable for spatial interpolation, especially in geosciences. Each method has its strengths and weaknesses, and the choice of which to use depends on the specific characteristics of the spatial data and the goals of the analysis. For a detailed and accurate representation of spatial data, OK is generally the most robust method, followed by RF and SVR for balanced or non-linear trends.

5.7 Chapter Summary

Chapter 5 presents the heart of the research findings, focusing on the application and evaluation of OK and ML models in predicting PGE grades. Initially, it provides an overview of the methodology employed for point and block kriging using the *GeostatsPy* package, followed by a detailed comparative analysis of the prediction capabilities of ML algorithms (RF, DT, SVR, kNN) against OK.

The chapter evaluates the models' performance at point prediction levels, using error metrics such as RMSE, MSE, and MAE alongside visual tools like histograms and scatter plots of residuals to assess model accuracy. It highlights the superior performance of the *k*NN and RF models in capturing local variations in ore grades, outperforming OK in terms of precision and accuracy.

Subsequently, block estimate predictions were examined, emphasising the models' ability to generalise from detailed point data to broader spatial extents. This section demonstrates the potential of ML models, particularly *k*NN and RF, to enhance mineral resource estimation, offering significant insights into their application in practical mining and resource estimation scenarios.

Chapter 5 consolidates the research's core findings, affirming the efficiency of ML models, especially *k*NN, in geological prediction tasks. It underscores the potential of integrating ML with traditional geostatistical methods to revolutionize mineral resource estimation, providing a robust framework for future advancements in mining engineering and geostatistics.

6 CONCLUSIONS AND RECOMMENDATIONS

6.1 Introduction

Accurate grade prediction a critical component at any stage along the mining value chain. Traditional geostatistical approaches really on the assumption of stationarity. The use of ML in grade estimation does not require such assumptions; limiting the extent of human interaction making the process more streamlined in the continuous pursuit of unbiased and minimized errors. The potential for ML to improve grade prediction was attempted through the comparison of four machine learning models against OK.

6.2 Research Observations and Conclusion

In the realm of geological prediction problems, the ML models have demonstrated their suitability, by offering versatility and adaptability to various data patterns and structures, which are common in geoscience datasets. Similarly, the k NN model has proven effective due to its capacity to capture the intricacies of geological formations through its proximity-based predictive approach, making it highly relevant for geological datasets that inherently display non-linear and complex spatial relationships.

The analysis of the five ML models used for predicting PGE grades—both at the point and block levels—highlights the nuanced applications and advantages of different algorithms in the geostatistical field. The RF model exhibits exceptional performance owing to its ensemble learning technique, which combines the outcomes of numerous decision trees to produce a consensus prediction. This method effectively captures complex patterns within the data and naturally counters the overfitting that individual trees are prone to.

DT, although simpler than RF, can adeptly identify non-linear interactions within the dataset. Yet, their application can lead to varied results, particularly when scaling from more localised point predictions, to block predictions, which aggregate over larger volumes. The step from point-to-block estimates is critical, as it involves

not just an extrapolation of the model's ability to predict but also an integration over larger, potentially more heterogeneous spatial volumes.

LR and SVR, while being foundational algorithms in ML, encounter inherent constraints due to their linearity and sensitivity to specific configurations. In geological data contexts, characterized by their complexity and non-linear behaviours, these models can underperform compared to their non-linear counterparts.

Notably, the k NN model emerges as a highly suitable tool for geological predictions, as evidenced by the swath plots and comparative metrics provided.

The applicability of these models to geological prediction is not absolute; rather, it is contingent upon the characteristics of the ore body itself. The transition from accurate point predictions to robust block predictions with k NN, and to a lesser extent with RF and DT, demonstrates adaptability and a capacity to accommodate the averaging necessary for block predictions. This proficiency is particularly valuable in mining operations where decisions are made on a block-by-block basis, and where the ability to generalize from training data to unseen spatial extents is paramount.

In conclusion, while the RF model excels due to its ensemble approach and the DT model for its simplicity and ability to capture non-linearity, it is the k NN model that stood out for its direct applicability to the spatial domain of geosciences. The integration of these ML models suggests a promising direction for enhancing the predictive accuracy and efficiency of mineral resource estimation, which is vital for optimising resource extraction and management.

6.3 Research Contributions

This study has fulfilled the objectives set out in Chapter 1, contributing significantly to the understanding of ore grade prediction. It has illuminated the crucial role of spatial correlation in predicting accuracy, underscoring the potential for developing ML techniques that more effectively incorporate these correlations, particularly across varying scales. By comparing traditional geostatistical methods like OK with

various ML algorithms, the research has provided empirical evidence of the advantages and adaptability of ML in geological predictions, especially highlighting the efficiency of k NN. Drawing on the Nwalia, et al., 2024 article, this research supports the suitability of k NN in geological settings due to its ability to process local data variations without pre-assumptions on data distribution. Furthermore, this research has laid the groundwork for potentially new performance metrics tailored to ore grade prediction that align with practical mining outcomes.

6.4 Research Limitations

There were inherent limitations within the study. The research has primarily focused on the prediction accuracy of block and point estimates, which may not encapsulate the full scope of ore body variability. Additionally, the success in the application of ML algorithms like k NN is dependent on the ore body's characteristics, and thus the results may not be universally applicable. Another limitation lies in the choice of performance metrics; while metrics like KE and Slope of SLOR are robust, they might not fully capture the complexities involved in ore grade prediction. A further constraint is the reliance on the variogram model within the kriging process, where the study assumes that the model is the most appropriate representation of spatial relationships, which may not always hold true across different ore bodies.

6.5 Recommendations for Future Research Work

Recommendations for future work would be:

- **Hybrid modelling:** Investigating hybrid models that combine the strengths of ML and geostatistical methods to leverage spatial correlations effectively at both point and block scales.
- **Model Tuning and Validation:** Develop robust methodologies for tuning hyperparameters and validating models, possibly using cross-validation techniques that are sensitive to spatial dependencies.
- **Feature Engineering:** Further research into incorporating geologically meaningful features and spatial descriptors could enhance the models' predictive capabilities.

- **Data Augmentation and Simulation:** Utilise data augmentation and synthetic data generation to enhance the training of ML models, especially where data is scarce.
- **Practical Implementation:** Perform case studies to demonstrate the practical implementation of ML models in operational mining scenarios, addressing industry-specific challenges.

future research can advance the application of ML in ore grade prediction, leading to more accurate, reliable, and efficient resource estimation and mining operation.

7 REFERENCES

The click Reader, 2021. *Medium*. [Online]

Available at: <https://medium.com/@theclickreader/decision-tree-regression-explained-with-implementation-in-python-1e6e48aa7a47>

[Accessed 4 December 2023].

Abzalov, M., 2016. *Applied Mining Geology*. s.l.:Springer.

Armstrong, M., 1998. *Basic linear geostatistics*. 1 ed. s.l.:Springer Berlin, Heidelberg.

Badillo, S. Banfai, B., Birzele, F., Davykov, I., Hutchinson, L., Kam-Thong, T., Siebourg-Polter, J., Bernhard, S., & Zhang, J. D., 2020. An introduction to machine learning. *Clinical pharmacology & therapeutics*, 107(4), pp. 871-885.

Bekin, M., Hsu, D. & Mandal, S., 2018. Reconciling modern machine learning practice and the bias-variance trade-off. *arXiv: Machine Learning*.

Bhandari, A., 2024. *Analytics Vidhya*. [Online]

Available at: <https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/>

[Accessed 12 January 2024].

Bhattacharjee, J., 2017. *Medium*. [Online]

Available at: <https://medium.com/technology-nineleaps/popular-machine-learning-algorithms-a574e3835ebb>

[Accessed 11 November 2023].

Bishop, C. M., 2007. *Pattern Recognition and Machine Learning*. 2007 ed. New York: Springer.

Caraka, E. R., Lee, Y., Chen, R-C. & Toharudin, T., 2020. Using hierarchical likelihood toward support vector machines: Theory and its application. *IEEE Access*, Volume 8, pp. 194797-194807.

- Cawthorn, R. & McKenna, N., 2006. Extension of the western limb, Bushveld Complex (South Africa), at Cullinan diamond mine. *Mineralogical Magazine*, 70(3), pp. 241-256.
- Chanderman, L., Dohm, C. & Minnitt, R., 2017. 3D geological modelling and resource estimation for a gold deposit in Mali. *The Journal of the Southern African Institute of Mining and Metallurgy*, Volume 117, pp. 189-197.
- Coombes, J., 2008. *The art and science of resource estimation: A practical guide for geologists and engineers*. Perth: Coombes Capability.
- Cortes, C. & Vapnik, V., 1995. Support-vector networks. *Machine Learning*, 20(3), pp. 273-297.
- Cover, T. M. & Hart, P. E., 1967. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1), pp. 21-27.
- Deseta, N., 2020. *Geology for Investors*. [Online]
Available at: <https://www.geologyforinvestors.com/geology-of-the-pge-deposits-in-the-bushveld-complex-south-africa/>
[Accessed 11 October 2023].
- Deutsch, C., 2015. *geostatisticslessons*. [Online]
Available at: <https://geostatisticslessons.com/lessons/celldeclustering>
[Accessed 5 December 2023].
- Deutsch, C. & Journel, A., 1998. *GSLIB: Geostatistical Software Library and User's Guide*. 2nd ed. s.l.:Oxford University Press.
- Deutsch, C. & Rossi, M., 2013. *Mineral Resource Estimation*. Berlin: Springer.
- Deutsch, J., 2015. *Geostatistics Lesson*. [Online]
Available at: <https://geostatisticslessons.com/lessons/variogramparameters>
[Accessed 10 December 2023].

- Dumakor-Dupey, N. K. & Arya, S., 2021. Machine learning—A review of applications in mineral resource estimation. *Energies*, 14(14), pp. 1-29.
- Erten, E. G., Yavuz, M. & Deutsch, C. V., 2021. Grade estimation by a machine learning model. *Applied Earth Science*, 130(1), pp. 57-66.
- Gavrilova, Y., 2020. *Serokell*. [Online]
Available at: <https://serokell.io/blog/ai-ml-dl-difference>
[Accessed 20 April 2023].
- Gokte, S. A., 2020. *KDnuggets*. [Online]
Available at: <https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html>
[Accessed 28 January 2024].
- Hastie, T., Tibshirani, R., Friedman, J. & Friedman, J. H., 2009. *Elements of Statistical Learning: Data Mining, Inference and Prediction*. 2nd ed. New York: Springer.
- Hekmatnejad, A., Emery, X. & Alipour-Shahsavari, M., 2019. Comparing linear and non-linear kriging for prediction and ore/waste classification in mineral deposits. *International Journal of Mining, Reclamation and Environment*, 33(4), pp. 247-264.
- Irizarry, R. A., 2019. *Introduction to data science: Data analysis and prediction algorithms with R*. 1 ed. s.l.:CRC Press.
- Isaaks, E. & Srivastava, M., 1989. *Applied Geostatistics*. New York: Oxford University Press.
- Jafrasteh, B., Fathianpour, N. & Suarez, A., 2018. Comparison of machine learning method for copper grade estimation. *Computation Geosciences*, 22(5), pp. 1371-1388.
- Janiesch, C., Zschech, P. & Heinrich, K., 2021. Machine learning and deep learning. *Electronic Markets*, 31(3), pp. 685-695.

- Journel, A. & Huijbregts, C., 1978. Mining Geostatistics. In: *Mining Geostatistics*. London: Academic Press, pp. 26-95.
- Liaw, A. & Wiener, M., 2002. Classification and Regression by randomForest. *R news*, 2(3), pp. 18-22.
- Matheron, G., 1963. Principles of geostatistics. *Economic geology*, 58(8), pp. 1246-1266.
- Minnitt, R., 2020. *MINN 7054A: Simulation Techniques*. Johannesburg: University of the Witwatersrand.
- Montgomery, D. C., Peck, E. A. & Vinning, G. G., 2012. *Introduction to Linear Regression Analysis*. 5th ed. s.l.:Wiley.
- Moon, C. J., Whateley, M. K. & Evans, A. M., 2006. *Introduction to mineral exploration*. 2nd ed. Carlton: Blackwell Publishing .
- Nichols, J. A., Chan, H. W. H. & Barker, M. A., 2018. Machine learning: Application of artificial intelligence to image diagnoses. *Biophysical Reviews*, 11(1), pp. 111-118.
- Nwalia, G., Zhang, S., Bourdeau, J. & Frimmel, H., 2024. Spatial Interpolation Using Machine Learning: From Patterns. *Natural Resources Research*, Volume 33, pp. 129-161.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blonde, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., & Cournapeau, D., 2011. Sci-kit learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85), pp. 2825-2830.
- Posa, D. & Marcotte, D., 1992. Robustness of kriging weights to non-bias conditions geology. *Mathematical Geology*, 24(8), pp. 759-773.

- Probst, P., Boulesteix, A. & Bisch, B., 2019. Tunability: Importance of hyperparameters of machine learning algorithms. *Journal of Machine Learning Research*, Volume 20, pp. 1-32.
- Pyrz, M.J., Jo, H., Kuppenko, A., Liu, W., Gigliotti, A. E., Sulomaki, T., & Santos, J., 2021. *PyPI*. [Online]
Available at: <https://pypi.org/project/geostatspy/>
[Accessed 3 December 2023].
- Saini, A., 2023. *Analytics Vidhya*. [Online]
Available at: <https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/>
[Accessed 4 December 2023].
- Sammut, C. & Webb, G. I., 2010. *Encyclopedia of Machine Learning*. 1 ed. New York: Springer New York, NY.
- Samson, M., 2020. *Mineral resource estimates with machine learning and geostatistics*, Alberta: University of Alberta.
- Sharma, A., 2023. *Analytics Vidhya*. [Online]
Available at: <https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/>
[Accessed 4 December 2023].
- Shouwstra, R. & Lee, C. K. E., 2000. A short geological review of the bushveld complex. *Platinum Metals Review*, 44(1), pp. 33-39.
- Sinclair, A., 1998. Exploratory data analysis: A precursor to resource/reserve estimation. *Canadian Institute of Mining, Metallurgy and Petroleum*, Volume 7, pp. 77-90.
- Sinclair, A. J. & Blackwell, G. H., 2002. *Applied mineral inventory estimation*. s.l.:Cambridge University Press.

- Sinclair, A. J. & Blackwell, G. H., 2002. *Applied Mineral Inventory Estimation*. Cambridge: Cambridge University Press.
- Srivastava, R. & Isaaks, E. H., 1989. *Applied Geostatistics*. New York: Oxford university press.
- Summit, C. & Webb, G. I., 2011. *Encyclopedia of Machine Learning*. Boston, MA: Springer US.
- Sun, J., Zhang, R., Cheng, M., Chen, B., Wang, X., Li, Q., & Ren, L., 2021. Identification of porosity and permeability while drilling based on machine learning. *Arabian Journal for Science and Engineering*, 46(7), pp. 7031-7045.
- Vann, J., Jackson, S. & Bertoli, O., 2003. *Quantitative Kriging Neighbourhood Analysis for Mining Geologist - A Description of the Method With Worked Case Examples*. Bendigo, Australasian Institute of Mining and Metallurgy, pp. 1-10.
- Venkatesh, B., Singh, D. & Leavline, E., 2017. *Pragmatic Investigation on Performance of Instance-Based Learning Algorithms*. Singapore, Springer.
- Wilde, B. J. & Deutsch, C. V., 2010. *Centre for computational geostatistics*. [Online]
Available at: https://www.ccgaberta.com/ccgresources/report12/2010-403_data_spacing_and_classification_programs.pdf
[Accessed 10 December 2023].
- Wu, J., Chen, S. & Liu, X., 2020. Efficient hyperparameter optimization through model-based reinforcement learning. *Neurocomputing*, Volume 409, pp. 381-393.
- Yunsheng, S., Jiye, L., Jing, L., & Xingwang., 2017. An efficient instance selection algorithm for k nearest neighbour regression. *Neurocomputing*, Volume 251, pp. 26-34.

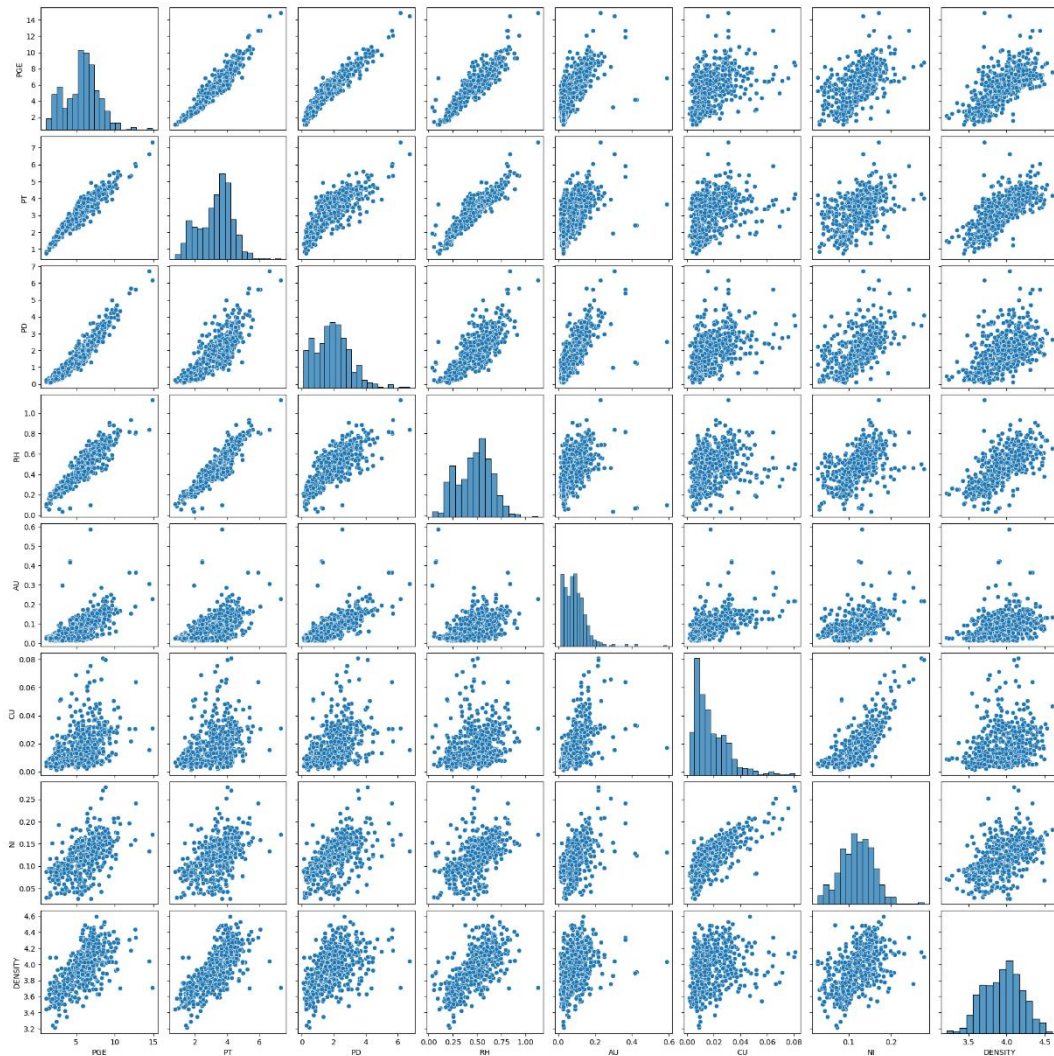
Zhang, H. & Ma, Y., 2012. *Ensemble Machine Learning: Methods and Applications*. New York: Springer.

Zientek, M. L., Causey, J. D., Parks, H. L. & Miller, R. J., 2014. *Platinum-group elements in southern africa: mineral inventory an assessment of undiscovered mineral resource*, Reston, VA: U.S. Geological Survey.

APPENDIX A: Figures

```
array([ 1.1238,  1.1239,  1.4565,  1.4679,  1.4885,  1.5085,  1.5746,
        1.6081,  1.6375,  1.6408,  1.6608,  1.7017,  1.7522,  1.8424,
        1.8807,  1.8853,  1.9458,  1.9901,  2.016 ,  2.0559,  2.0717,
        2.0893,  2.1124,  2.1208,  2.1262,  2.142 ,  2.1785,  2.201 ,
        2.201 ,  2.2095,  2.2095,  2.2189,  2.2223,  2.2399,  2.259 ,
        2.2609,  2.3031,  2.3291,  2.3323,  2.3412,  2.3502,  2.3518,
        2.3563,  2.3594,  2.4098,  2.4158,  2.4246,  2.4285,  2.4444,
        2.447 ,  2.4759,  2.487 ,  2.4953,  2.5465,  2.5476,  2.5519,
        2.5611,  2.569 ,  2.5735,  2.5822,  2.5965,  2.61 ,  2.6117,
        2.6181,  2.6369,  2.6505,  2.6811,  2.6952,  2.6979,  2.7163,
        2.7229,  2.7288,  2.7337,  2.7399,  2.7435,  2.7469,  2.7589,
        2.7919,  2.7961,  2.7983,  2.8126,  2.86 ,  2.8827,  2.9242,
        2.927 ,  2.9336,  2.9337,  2.9398,  2.9766,  2.9771,  3.0063,
        3.0553,  3.0913,  3.1113,  3.1288,  3.1356,  3.1458,  3.1468,
        3.1571,  3.1624,  3.1805,  3.2451,  3.247 ,  3.381 ,  3.3917,
        3.4181,  3.4277,  3.4342,  3.4408,  3.4667,  3.4701,  3.4741,
        3.5429,  3.5573,  3.5605,  3.5838,  3.5907,  3.652 ,  3.6803,
        3.7761,  3.8081,  3.8201,  3.8347,  3.8507,  3.872 ,  3.9041,
        3.9357,  3.9495,  3.9669,  3.981 ,  3.9947,  4.0054,  4.0328,
        4.0479,  4.0786,  4.1584,  4.1709,  4.1938,  4.1939,  4.2012,
        4.226 ,  4.2613,  4.2635,  4.2701,  4.271 ,  4.3559,  4.3628,
        4.4157,  4.4165,  4.4236,  4.4246,  4.431 ,  4.4369,  4.4791,
        4.4961,  4.5064,  4.5095,  4.534 ,  4.5453,  4.5609,  4.5731,
        4.6392,  4.655 ,  4.7413,  4.7486,  4.7803,  4.7882,  4.7899,
        4.8435,  4.876 ,  4.8991,  4.9001,  4.907 ,  4.9231,  4.9345,
        4.9376,  4.9574,  4.9964,  4.9996,  5.0056,  5.0471,  5.051 ,
        5.0526,  5.0578,  5.0669,  5.0759,  5.0786,  5.0789,  5.0953,
        5.0998,  5.1011,  5.1283,  5.1422,  5.163 ,  5.1865,  5.1867,
        5.1961,  5.2332,  5.2359,  5.2378,  5.2543,  5.2596,  5.2677,
        5.2723,  5.2742,  5.2949,  5.3041,  5.3185,  5.3201,  5.322 ,
        5.339 ,  5.3478,  5.3515,  5.3637,  5.3659,  5.3731,  5.3734,
        5.3869,  5.3872,  5.3915,  5.4009,  5.4055,  5.4188,  5.4193,
        5.4392,  5.448 ,  5.4596,  5.4648,  5.4783,  5.4882,  5.5125,
        5.5184,  5.5186,  5.5212,  5.5281,  5.5315,  5.5396,  5.5449,
        5.5524,  5.5554,  5.5701,  5.5795,  5.5824,  5.5996,  5.6005,
        5.6147,  5.627 ,  5.6392,  5.6695,  5.6841,  5.6847,  5.6916,
        5.6957,  5.697 ,  5.6975,  5.7136,  5.7142,  5.7215,  5.7474,
        5.7643,  5.7723,  5.7731,  5.7781,  5.7795,  5.7957,  5.7997,
        5.8314,  5.8371,  5.8413,  5.8471,  5.8492,  5.8522,  5.8553,
        5.8687,  5.87 ,  5.8701,  5.8752,  5.8796,  5.8871,  5.8887,
        5.8934,  5.8952,  5.9001,  5.9009,  5.9034,  5.9065,  5.9124,
        5.9166,  5.9267,  5.9411,  5.947 ,  5.9566,  5.9639,  5.9675,
        5.9697,  5.9741,  5.9788,  5.9788,  5.9842,  5.9855,  5.9879,
        6.0081,  6.0114,  6.0175,  6.0506,  6.0593,  6.0655,  6.0867,
        6.1039,  6.1055,  6.1078,  6.1229,  6.1273,  6.1299,  6.1349,
        6.1429,  6.1433,  6.1623,  6.1634,  6.1768,  6.1813,  6.1878,
        6.1908,  6.1962,  6.2021,  6.2123,  6.2341,  6.2395,  6.2612,
        6.2755,  6.2775,  6.2865,  6.2929,  6.2961,  6.3049,  6.3307,
        6.3391,  6.34 ,  6.3576,  6.3633,  6.3774,  6.3821,  6.3835,
        6.3865,  6.4025,  6.408 ,  6.4224,  6.433 ,  6.4366,  6.4373,
        6.4439,  6.4444,  6.4632,  6.4779,  6.4886,  6.5125,  6.5131,
        6.5139,  6.535 ,  6.5356,  6.5456,  6.5494,  6.5535,  6.5554,
        6.5608,  6.5728,  6.5758,  6.5776,  6.5916,  6.5962,  6.5999,
        6.6118,  6.6131,  6.6184,  6.6186,  6.6636,  6.666 ,  6.6664,
        6.6697,  6.6877,  6.6921,  6.7135,  6.7143,  6.717 ,  6.7404,
        6.7408,  6.7449,  6.7549,  6.7633,  6.7642,  6.7666,  6.7699,
        6.7838,  6.8014,  6.802 ,  6.8073,  6.8092,  6.8229,  6.8337,
        6.8444,  6.8616,  6.8675,  6.8763,  6.8766,  6.8853,  6.8942,
        6.8987,  6.9002,  6.9172,  6.9355,  6.9381,  6.9775,  6.9775,
        6.9815,  6.9995,  7.0024,  7.0029,  7.0359,  7.0409,  7.0424,
        7.0521,  7.1029,  7.1033,  7.118 ,  7.1349,  7.144 ,  7.1477,
        7.1708,  7.1735,  7.1826,  7.1898,  7.1928,  7.1974,  7.2019,
        7.2038,  7.2071,  7.2109,  7.2229,  7.2467,  7.2531,  7.2538,
        7.2586,  7.2632,  7.2646,  7.2837,  7.2906,  7.3012,  7.3027,
        7.3219,  7.3254,  7.3278,  7.337 ,  7.3573,  7.3851,  7.4129,
        7.4178,  7.4444,  7.4687,  7.4902,  7.5031,  7.5166,  7.5168,
        7.5224,  7.5251,  7.5486,  7.5552,  7.5786,  7.5972,  7.6061,
        7.6114,  7.6453,  7.6748,  7.6756,  7.6771,  7.713 ,  7.7174,
        7.7267,  7.7331,  7.7879,  7.8016,  7.8334,  7.8363,  7.8368,
        7.8737,  7.8923,  7.9268,  7.933 ,  7.9372,  7.9393,  7.9396,
        7.9693,  7.9721,  8.0091,  8.0098,  8.0384,  8.0524,  8.0656,
        8.1205,  8.1672,  8.2007,  8.2311,  8.2414,  8.2551,  8.2626,
        8.2801,  8.2868,  8.2898,  8.3044,  8.3126,  8.3244,  8.3324,
        8.3516,  8.3897,  8.4249,  8.4396,  8.4622,  8.4648,  8.4683,
        8.4746,  8.5127,  8.5332,  8.5371,  8.5597,  8.5721,  8.5923,
        8.6819,  8.6843,  8.6988,  8.7193,  8.7202,  8.7605,  8.7798,
        8.8405,  8.8842,  8.9197,  8.9325,  8.9401,  8.9718,  9.0296,
        9.0626,  9.1707,  9.202 ,  9.2097,  9.2768,  9.2892,  9.3102,
        9.3127,  9.3493,  9.4174,  9.4804,  9.4844,  9.6171,  9.6785,
        9.8757,  9.8889,  9.926 , 10.1037, 10.1227, 10.1408, 10.2581,
        10.3336, 10.396 , 10.4141, 10.7296, 11.883 , 12.1302, 12.6769,
        12.728 , 14.4988, 14.875 ])
```

Appendix A 1: Ordered array of PGE grade values from lowest to highest for data capping



Appendix A 2: Pairplot of distribution of the individual predictor variables as well as the correlation scatterplots between the variables

```
models = {
    'Linear Regression': LinearRegression(),
    'SVM': SVR(kernel = 'rbf', gamma = 'scale', epsilon = 0.8, C = 150),
    'SVM2': make_pipeline(StandardScaler(), SVR(kernel = 'rbf', gamma = 'scale', epsilon = 0.8, C = 150)),
    'Decision Tree': DecisionTreeRegressor(min_samples_split = 10, max_depth = 25, criterion = 'friedman_mse'),
    'Random Forest': RandomForestRegressor(n_estimators = 200, min_samples_split = 3, max_depth = 20, criterion = 'friedman_mse'),
    'KNN': make_pipeline(StandardScaler(), KNeighborsRegressor(n_neighbors=3, weights='distance', metric='euclidean'))
}
```

Appendix A 3: Initialised ML models with optimised parameters

```
In [ ]: def grid_search_kriging(df, xcol, ycol, vcol, tmin, tmax, nx, xm, xsiz, ny, ym, ysiz, ktype, skmean, vario):
    # Define search parameters
    search_radius = [3400, 4000]
    ndmin_values = [3, 5, 8, 12]
    ndmax_values = [10, 15, 20]
    nxdis_values = nxdis_values = [2, 4]

    # Initialize best performance and parameters
    best_performance = -float('inf')
    best_params = {}
    best_kriging_result = None
    best_weights_map = None
    best_cbb = None
    best_s = None
    best_kriging_variance = None

    # Iteration count and total iterations
    total_iterations = len(search_radius) * len(ndmin_values) * len(ndmax_values) * len(nxdis_values) * len(nydis_values)
    iteration_count = 0

    # Grid search loop
    for radius in search_radius:
        for ndmin in ndmin_values:
            print(f"Current ndmin: {ndmin}")
            for ndmax in ndmax_values:
                print(f"Current ndmax: {ndmax}")
                for nxdis in nxdis_values:
                    for nydis in nydis_values:
                        iteration_count += 1

                        # Skip invalid configurations
                        if ndmin > ndmax:
                            continue

                        # Assuming kb2d_3 is modified accordingly to return cbb, s, and kriging variance
                        kriged_results, kriging_variance, weights_map, cbb, s, vk = geostats.kb2d_3(df, xcol, ycol, vcol, tmin, tmax, nx, xm, xsiz, ny, ym, ysiz, nxdis, nydis, ndmin, ndmax, radius, ktype, skmean, vario)

                        # Calculate SLOr and KE
                        slor, ke = calculate_performance_metric(kriged_results, cbb, kriging_variance, s, unest=999.)

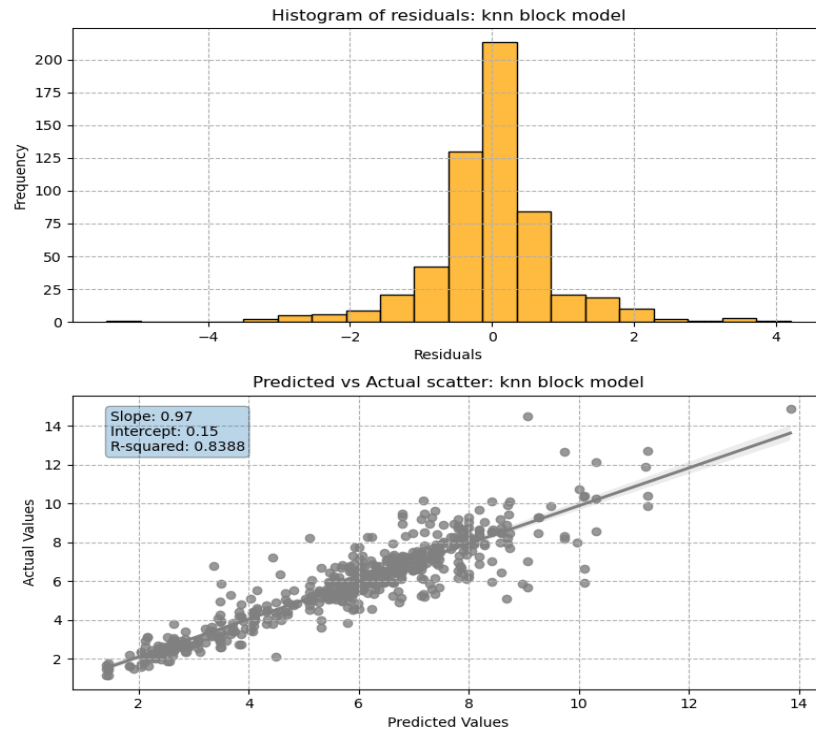
                        # Use custom scoring to evaluate the overall performance
                        overall_performance = custom_scoring(KE=ke, SLOr=slor)

                        # Update best performance and parameters based on overall performance
                        if overall_performance > best_performance:
                            best_performance = overall_performance
                            best_params = {
                                'radius': radius,
                                'ndmin': ndmin,
                                'ndmax': ndmax,
                                'nxdis': nxdis,
                                'nydis': nydis
                            }
                            best_ke = ke
                            best_slor = slor
                            #best_kriging_result = kriged_results
                            #best_weights_map = weights_map
                            #best_cbb = cbb
                            #best_s = s
                            #best_kriging_variance = kriging_variance

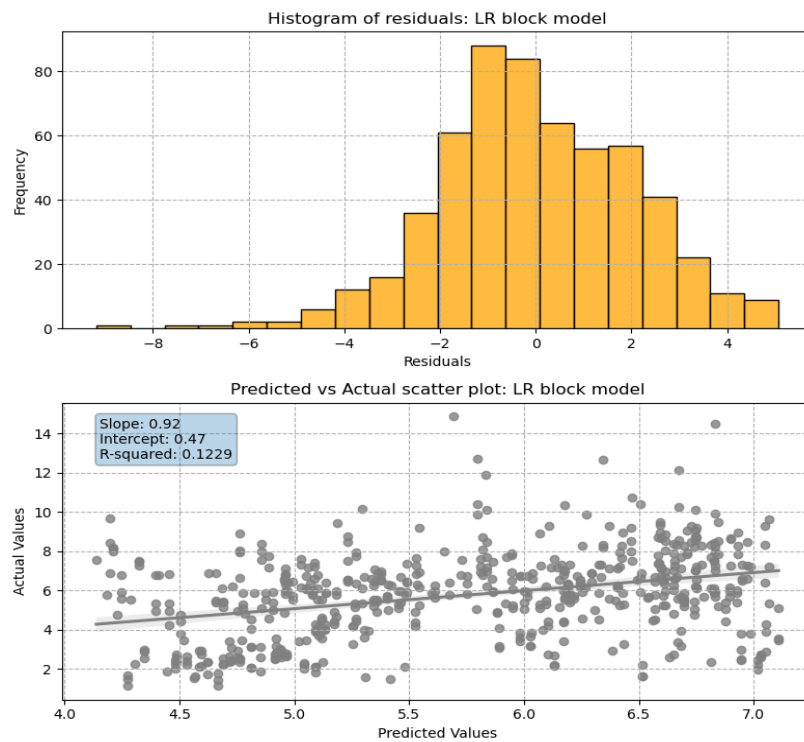
                        # Progress update
                        if iteration_count % 10 == 0 or iteration_count == total_iterations:
                            print(f"Iteration {iteration_count}/{total_iterations}, Current Best Performance: {best_performance}, Current Best Parameters: {best_params}, Best KE: {best_ke}, Best SLOr: {best_slor}")

    return best_params, best_performance
```

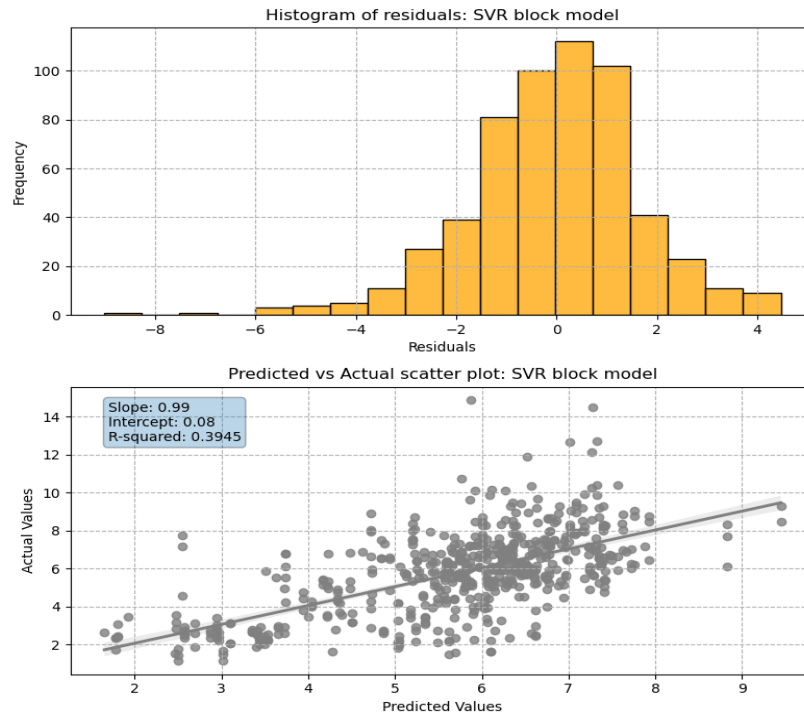
Appendix A 4: Source code of the *grid_search_kriging* function



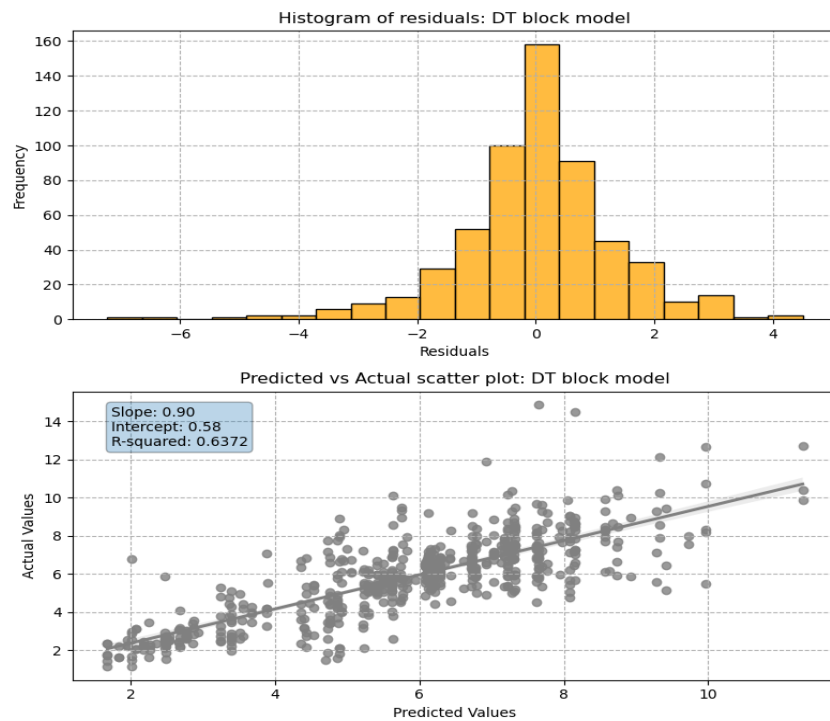
Appendix A 5: Histogram of KNN residuals (top), Scatterplot of predicted vs actual PGE block grades



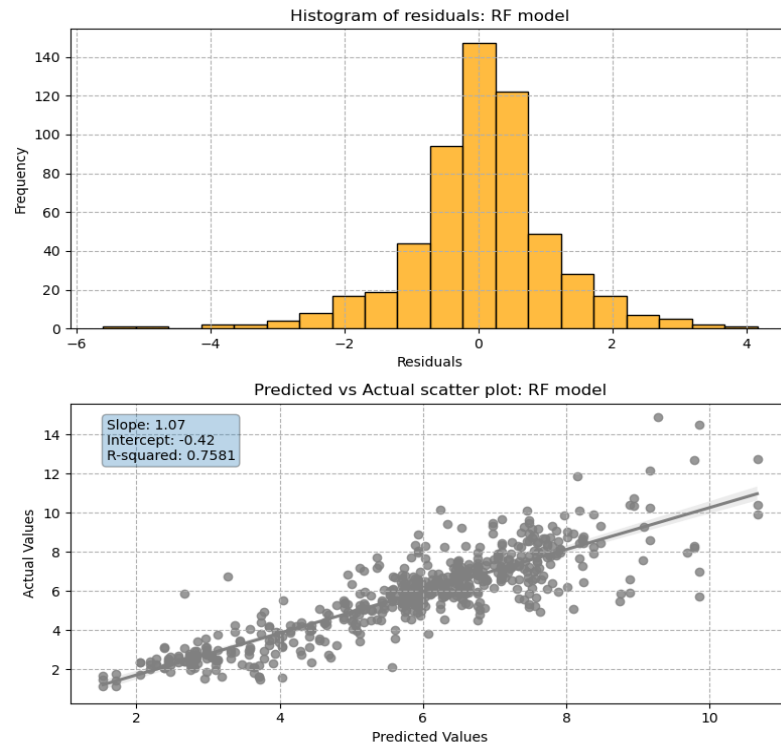
Appendix A 6: Histogram of LR residuals (top), Scatterplot of predicted vs actual PGE block grades



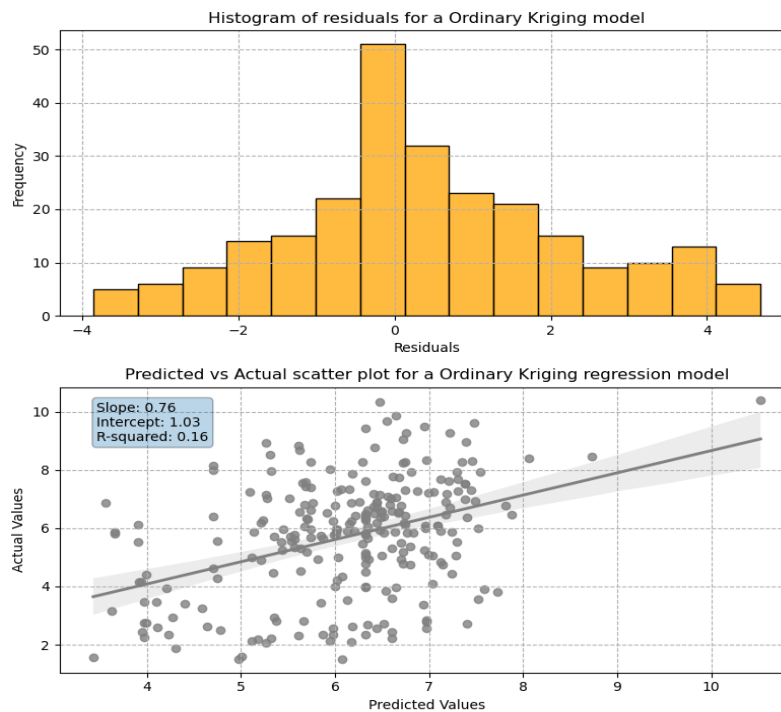
Appendix A 7: Histogram of SVR residuals (top), Scatterplot of predicted vs actual PGE block grades



Appendix A 8: Histogram of DT residuals (top), Scatterplot of predicted vs actual PGE block grades



Appendix A 9: Histogram of RF residuals (top), Scatterplot of predicted vs actual
PGE block grades



Appendix A 10: Histogram of OK residuals (top), Scatterplot of predicted vs
actual PGE block grades