MACHINE LEARNING APPROACH FOR THE PREDICTION OF THE STATUS OF TANZANIAN WELLS [COMP4030 CW2 - Data Science and Machine Learning]

*Thomas Cotter
Computer Science
University of Nottingham
Nottingham, England
psytc8@nottingham.ac.uk

*Loo Yang Shen Jason
Computer Science
University of Nottingham
Nottingham, England
hfyyl5@nottingham.ac.uk

***Abstract*—This paper details our approaches and results for the COMP4030 CW2. We have used a number of machine learning algorithms in order to predict the status ('Functional', 'Functional - Needs Repair' & 'Non-Functional') of water wells in Tanzania. This is important as not only does knowing the status of the well help keep the total percentage of functioning wells higher, but it also allows for more effective spending from the government as they no longer would have to send workers out to check the status of the well. Our results suggested that the most important features were ADD RESULTS, and the best classification model as ADD RESULTS**

***Index Terms*—Machine Learning, Data Science, Classification**

I. Introduction

This section will provide an introduction to the dataset and the research questions we have attempted to solve.

A. Research Questions

We have decided that our research question will be:

What factors are most important for determining the status of a well, and how accurately can we classify wells based on these features?.

We choose this question because we are interested in the factors that determine the status of a well, and using ML to try to classify these wells into 1 of 3 classes: Functional, Non-Functional & Functional Needs Repair. From this question, we can think about some follow-up questions. These could include:

- How does the accuracy of the classification model vary with different feature sets and classification algorithms?
- Could we use our results to ensure that wells are built and repaired so that fewer wells are non-functional?

B. Dataset

We will be using the dataset from the Tanzanian Ministry of Water, which contains information on the status of wells in Tanzania to answer our research question. This dataset has 59400 rows, with 40 different features. These 40 features could be broken down into three subgroups which hold information regarding: a) Geographic Location of the Wells. b) Management of the wells. and c) Water Condition of the wells. The dataset is originally split into 2 different files, one for labels and one for the actual data. These can be merged easily with pandas through left join on the "ID" column.

Fig. 1 shows the distribution of the target variable, which is the status of the well. We can see from this that we might need to oversample the 'functional needs repair' class.

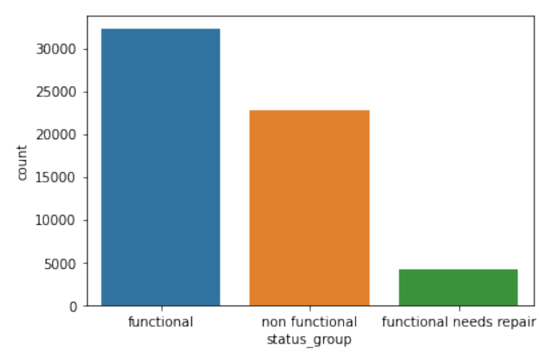


Fig. 1: Distribution of the target variable

C. Management Structure

One of the requirements of this project was to work separately within our pairs and try different approaches to the problem. This idea only works if we communicate the results found within our approaches to each other periodically so that we don't lose important insights that could combine well together. We took the three stages, 'Data Analysis', 'Data Preprocessing' & 'Data Classification' and followed a 'Chistmas Tree'-like management approach. This means for section, we performed our own analysis, before combining our results before moving onto the next section. This was also done cyclically, for example when we re-examined the

Preprocessing step in order to improve classification results. See Fig. 2 for more details

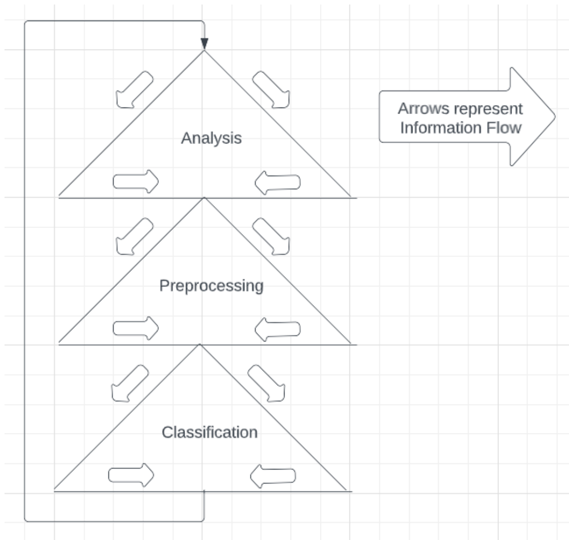


Fig. 2: Management Structure

II. Literature Review

In this section, we review the relevant literature on predicting machine failure, with a focus on studies that have used similar datasets to Pump It Up. Prediciting machine failure is an important area of reasearch, as it has the potential to reduce downtime and maintenance costs. Some machines may be part of a critical infrastructure, so preventing them from failure is of upmost importance.

One study by Pathak et al. (2023) [1] compared the performance of TabNet, a sequential attentive classification architecutre designed for tabular data, and tree-based approaches such as XBBoost. They found that TabNet outpeformed XGBoost, boasting an 83% accuracy compared to XGBoosts 78%. TabNet makes use of Transformers, a machine learning algorithm which uses self-attention to differentially weight the significance of each part of the input. A point of note is that TabNet does not require feature engineering to perform at these standards. While TabNet is an interesting solution, we have not used it in our report, as our primary goal was to showcase an end-to-end machine learning solution, and this includes feature enginering.

Pham et al studied fully connected neural networks on the Pump it Up dataset [2], settling on a model with 7 hidden layers and cross-entropy loss as the loss function. Their trained model achieved a 78.6% accuracy rate on the test dataset. Although our project has prioritized tree-based methods for their speed, it would be interesting to compare our results with those of a neural network.

Jithin Paul also undertook a study on the Pump it Up dataset, experimenting with different models to test for the best reulsts [3]. He proposed four methods, RandomForest, DeepLearning, LogisticRegression & AdaBoost. The results showed that RandomForest, with 120 trees, was the most accurate model, achieving a mean accuracy of 81.18% on the test dataset. This finding demonstrates that tree-based models perform exceptionally well on this dataset, which motivated us to use more sophisticated tree-based methods in our own work.

Mahabub [4] investigated the effectiveness of ensemble methods to detect fake news. They found that combining different classification methods into a Ensemble Voting Classifier produced the best results. We took this information and applied it to our problem, implementing a Voting Classifier to determine it's suitability for our problem.

III. Methodology

A. Data Analysis

When conducting data analysis, the python library Seaborn [5] was our primary data visualization tool. We found that Seaborn produces more visually appealing graphs than those produced by MatPlotLib and the library itself is simpler to use. We used this alongside Pandas [6], a python library for advanced data handling. The results obtained from our initial data analysis guided our pre-processing decisions. (See IV for detailed results).

1) Thomas: Thomas produced visualizations of the waterpoint & management type distributions for each decade as well as a line chart plotting count against construction-year. The objective behind these graphs was to determine the possibility of using other values in the dataset to fill in the missing construction year data. This be discussed further in IV

2) Jason: Jason checked the number of missing and unique values across 38 features provided. The main reasoning behind it was to identify features that would require imputations due to missing or invalid values. In addition, this step allowed him to distinguish features which would have to be recategorised due to the large number of unique values present. He also generated Count Plots for each of the previously mentioned features based on water wells' status. These plots allowed him to visualise the distribution of the wells' status based on features. Thus, making it easier to identify features that could possibly influence the status of a water well.

B. Data Preprocessing

During the pre-processing phase, we followed our Christmas tree planning approach, working separately before combining our respective results. We each tried different techniques for pre-processing to determine the optimal ones. We both used pandas techniques to format the data in the format we wanted. Pandas is fast data manipulation tool, which makes it perfect for this problem. In this phase, we also implemented feature engineering and data cleaning approaches.

1) Thomas: Thomas initially performed the imputation of the latitude & longitude missing values by applying mean imputation with a specific approach. Firstly, the data was grouped by 'ward', followed by calculating the mean for the missing lat/lon values. If any NaNs remained (e.g. due to missing 'ward' information), then the data was grouped by 'LGA' and the process repeated. Finally, the data was grouped by 'region', ensuring no missing values were left. The will be discussed further in IV.

He also looked at the Funder & Installer columns, which contained 1000s of unique values. In his approach, he decided to group the column into Seven categories: 'Charity', 'Government', 'Local Government', 'Private', 'Religious', 'Foreign' & 'School'. This was done to simplify the dataset, making it easier for the model to learn the distribution. Finally, Thomas also used the SMOTE-ENN algorithm from imblearn [7], to balanced the imbalance in the 'Functional - Needs Repair' class. Imbalanced datasets could potentially affect model performance, especially in tree-based models.

2) Jason: Due to the presence of missing data in the Latitude & Longitude columns, Jason decided to apply mean imputation based on the Region column. The values of the Region column corresponds with the name of cities present in Tanzania. By calculating the mean latitude and longitude value of each region, a rough estimation of the geographical location of wells with missing value could be known. Therefore, allowing valuable insights to be learnt by ML models during training.

During the Data Analysis Phase, he had noticed that two columns Permit and Public Meeting had missing values. Upon further inspection, these two columns only contain the values of either 'True' or 'False'. Data Imputation was not done at this stage due to the lack of valuable data present in the data set. To preserve the distribution of these two columns, a unique values known as 'Unknown' was given to rows with missing data.

Jason had performed data recategorisation on both funder and installer columns which contained 1000 - 2000 unique values. He had first performed data cleaning to fix spelling mistakes that were present in each column. This was followed by identifying unique values with count above the threshold of 1000. Each unique value above the previously mentioned threshold would be classified as its own individual class while those below it would be classified under the 'Other' value. Data recategorisation would be able to reduce the number of unique values present in these two columns leading to better ML model training performance.

Although the number of unique values in the Region Column are minimal, Jason decided to reclassify them according to their geographical zones based on data that was provided by the Tanzania Water and Sanitation Network [8]. This reduced the number of unique values of the Region Column to 7 from 21. New insights could potentially be drawn upon based on the zone that the water wells is currently located in.

The data set provided had feature and target variables in the form of categorical form. Jason had converted these features into numerical form that is required of by ML models prior to training. He had encoded the values of the feature and target variables in the form of alphabetical order whereby a = 0, b = 1 etc.

C. Data Classification

To classify the data, we utilized various classification algorithms. Initially, our focus was on testing the performance of different algorithms without any feature selection. This would allows us to focus on the best performing ones. The results from this test can be seen in Table. I. From these results, we can see that XGBoostClassifier, CatBoostClassifier, BaggingClassifier & HistGradientBoostingClassifier were the most effective for our problem. We decided to fully explore these models using feature selection & hyper-parameter tuning. Our feature selection process employed a Chi-Squared test and feature importance obtained from a RandomForest to generate a subset of potential features.

TABLE I: Initial Classification Results

Algorithm	Accuracy	Precision	Recall
XGB	0.797811	0.751084	0.632675
CatBoost	0.794865	0.745754	0.633800
Bagging	0.793434	0.704477	0.658841
HistGradientBoosting	0.790909	0.743381	0.623847
DT	0.756902	0.643165	0.644585
KNN	0.708754	0.623125	0.563458

1) Thomas: Initially, Thomas followed on from the decision to balance the dataset. He trained the 4 top models (XGBoost, CatBoost, Bagging & HGBoost) on the balanced dataset and compared the results to models trained on the unbalanced dataset. The results were also compared when optimizing for 'recall_macro' or 'accuracy'. The results can be seen in IV. Furthermore, he used Weights and Biases (WandB) [9] to further analyse the subset of features produced by the Chi-Squared Test and RandomForest. WandB provided an easy-to-use MLOps platform for tracking our experiments and helped us understand which features worked best for each classifier. He trained multiple classifiers on different subsets of the feature set (ranging from sizes 0.75N to N, where N is the size of the original feature set), and tracked this information in WandB. WandB allows us to quickly filter for the best performing 'runs', and find out which feature set was used in this run. He also used WandB to further analyse the results of our cross-validation hyper-parameter tuning, in much a similar process to feature selection.

2) Jason: From the Random Forest feature importance and Chi-Squared Test, Jason had shortlisted 16 features that influenced the classification of Tanzanian Water Wells. Prior to model training, the dataset was subjected to a 80%:20% train test split. This was followed by normalising the testing and training set features with z-score normalisation.

For this study, he considered three different machine learning models (Random Forrest, XGBoost & CatBoost). Hyper-parameter tuning was conducted via GridSearchCV [10] from the sklearn library. Upon completion of the hyperparameter tuning process, the most optimal parameters of each model were recorded before being passed for model training.

To evaluate the performance of each model, Jason used three different evaluation methods. K-fold cross validation with the value of K being 5 , confusion matrix and receiver operating characteristic (ROC) curve. K-fold cross validations was used to evaluate the performance of the model on unseen data. Meanwhile, confusion matrix showcases the predictions which are labelled correctly and incorrectly by class. Although ROC Curve is usually meant for binary classification, it could still be extended to multiclass classification through pairwise comparison i.e., one class vs all other classes.

IV. Results

This section discusses the results obtain from each of the three stages, namely Data Analysis, Data Preprocessing and Data Classification. We will detail the results from each of our different approaches.

A. Data Analysis

1) Thomas: As mentioned in the methodology section. Thomas produced a number of visualisations to help understand the construction year feature. One example of these can be seen in Fig. 3. From these, we can see that there is a slight variation in Waterpoint Type usage across decades, but nothing of significance. For this reason, the missing values in the Construction year were kept as 'Unknown'. The distribution of the status group values within instances with 'Unknown' or 'Not Unknown' can be seen in Fig 3

B. Data Preprocessing

1) Thomas: Initially, the imputation of missing values was required. Thomas looked at imputing the missing values in the Latitude and Longitude feature. The results can be seen in Fig. 4.

Secondly, as mentioned the Installer and Funder columns had 1000s of unique values. There was also crossover between the values in the two columns. For this reason, Thomas decided to group the values together. This was performed manually using his own knowledge on the subject and the internet. The result was 8 categories, the counts for each of those categories in both the Funder & Installer column can be seen in Table II. These categorized versions of the features proved to provide some benefit to the model when testing the subset of features to use in WandB.

TABLE II: Funder & Installer Categories

Column	Category	Count
Funder	Government	20199
	Charity	11066
	Unknown	8216
	Foreign Aid	8131
	Religious	4087
	Private	3889
	Local Government	3774
	School	38
Installer	Local Government	22515
	Government	10327
	Unknown	8800
	Charity	7487
	Private	3853
	Foreign Aid	3346
	Religious	3046
	School	26

Finally, as metioned Thomas used the SMOTE-ENN algorithm to oversample the minority class. The results of this can be seen in Fig. 5.

C. Classification

1) Thomas: Post hyper-parameter tuning, metrics including accuracy, precision ('macro'), recall ('macro') and f1-score were calculated for each of the models. The models were tested on both the imbalanced and SMOTE-ENN balanced dataset. The results can be seen in Table III

Since the difference between different models and datasets is neglible (the imbalanced dataset actually perfoming better in some cases), Thomas decided to combine all the models to create a VotingClassifier to possibly squeeze out some extra performance. These results can be seen in Table IV

TABLE IV: Voting Classifier Results

Accuracy	Precision	Recall	F1-Score
0.810	0.759	0.650	0.679

Using both the results from model based feature selection, and feature selection after looking at the results from Weights & Biases, Thomas decided to use the following features in his final model: **'age'**, **'latitude_imputation'**, **'longitude_imputation'**, **'construction_decade'**, **'quality_group'**, **'basin'**, **'extraction_type'**, **'cat_installer'**, **'population'**, **'gps_height_zscorenormalise'**, **'cat_funder'**, **'quantity'**, **'consistent_water'**, **'source_class'**, **'zones'**, **'waterpoint_type'**, **'season'**, **'extraction_type_class'**, **'payment'**.

To relate this back to our research questions, we can say that these features are the most important when looking at whether wells are functional or not.We can also say we can classify whether a well is function or not with 81% accuracy.

V. Discussion

In this section, we discuss the results of our analysis. We compare each of our separate approaches to each other, as well as comparing them to the performance of models in the literature.

1) Tom:

VI. Conclusion

References

[1] K. Pathak and L. Shalini, "Pump it up: Predict water pump status using attentive tabular learning," 2023.

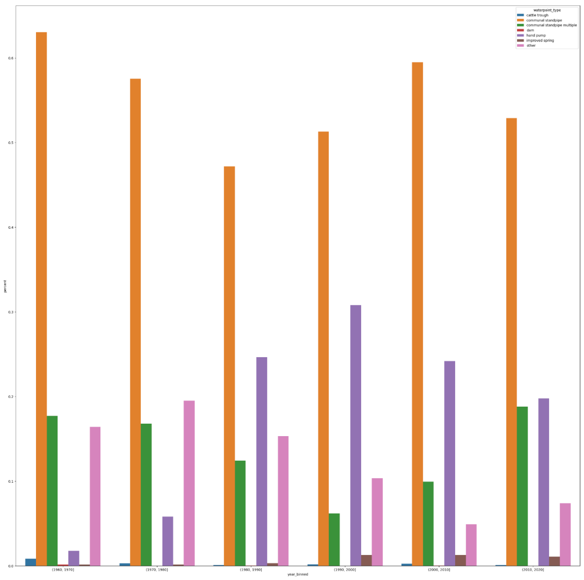
[2] A. Pham, B. Backus, and L. Zhu, "Pump it up: Mining the water table," *Stanford Machine Learning Projects 2018*, 2018.

[3] J. Paul, "Pump it up - data mining the water table," https://jitpaul.blog/2017/07/12/pump-it-up/, accessed: 2023-04-28.

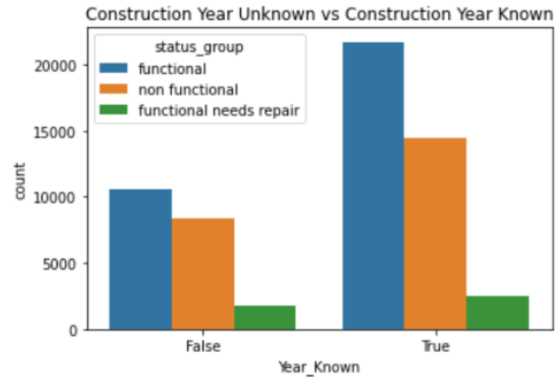
[4] A. Mahabub, "A robust technique of fake news detection using ensemble voting classifier and comparison with other classifiers," *SN Applied Sciences*, 2020. [Online]. Available: UR-https://doi.org/10.1007/s42452-020-2326-y

[5] M. Waskom, "Seaborn: statistical data visualization," https://seaborn.pydata.org/, accessed: 2023-04-24.

[6] "Pandas: Python data analysis library," https://pandas.pydata.org/, accessed: 2023-04-24.

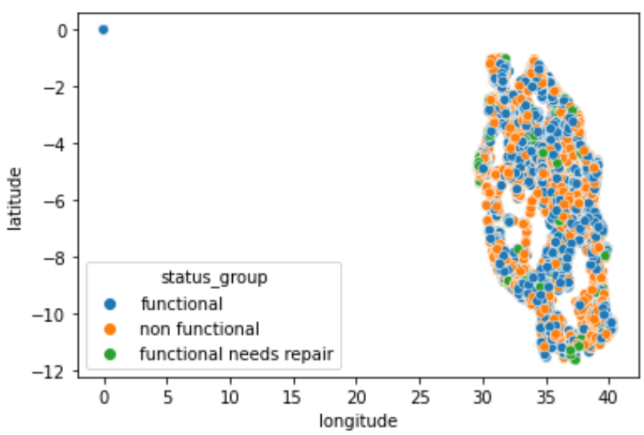


(a) Waterpoint Type vs Construction Year

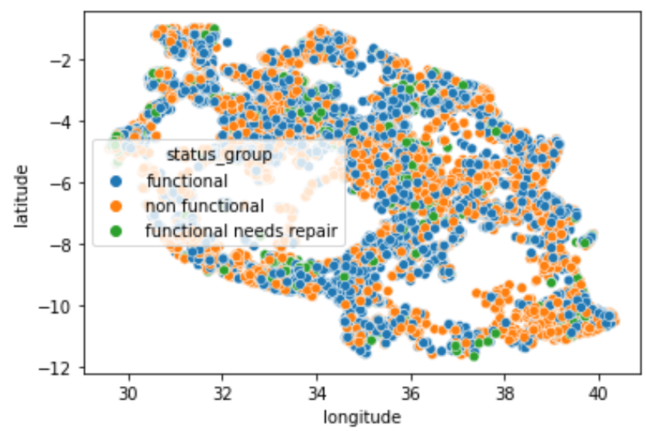


(b) Construction Year Unknown vs Construction Year Known

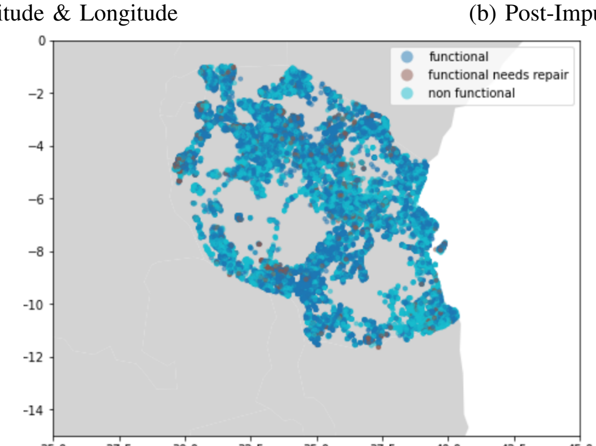
Fig. 3: Construction Year Data Analysis



(a) Pre-Imputation Latitude & Longitude

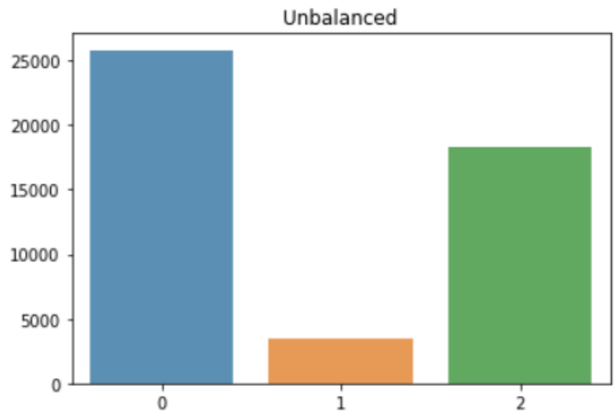


(b) Post-Imputation Latitude & Longitude

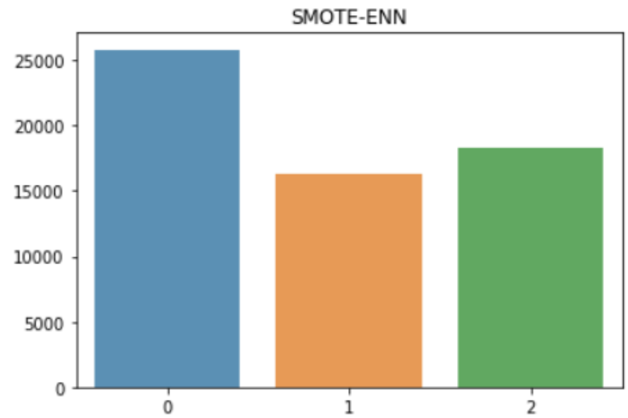


(c) Imputated Latitude & Longitude Overlayed onto a map of Tanzania

Fig. 4: Latitude & Longitude Imputation



(a) Pre-Oversampling Class Distribution



(b) Post-Oversampling Class Distribution

Fig. 5: SMOTE-ENN Oversampling

TABLE III: Classification Results

Scoring	Dataset	Model	Accuracy	Precision	Recall	F1-Score
Recall Macro	Balanced	XGBoost	0.802	0.701	0.668	0.683
		CatBoost	0.783	0.669	0.660	0.664
		HistGradientBoosting	0.796	0.693	0.665	0.677
		BaggingClassifier	0.803	0.703	0.676	0.687
	Imbalanced	XGBoost	0.804	0.720	0.662	0.683
		CatBoost	0.791	0.697	0.652	0.668
		HistGradientBoosting	0.806	0.730	0.658	0.681
		BaggingClassifier	0.803	0.722	0.660	0.681
Accuracy	Balanced	XGBoost	0.800	0.700	0.662	0.677
		CatBoost	0.787	0.678	0.665	0.670
		HistGradientBoosting	0.797	0.699	0.666	0.680
		BaggingClassifier	0.801	0.701	0.671	0.684
	Imbalanced	XGBoost	0.804	0.748	0.648	0.676
		CatBoost	0.797	0.753	0.635	0.665
		HistGradientBoosting	0.806	0.733	0.662	0.686
		BaggingClassifier	0.811	0.741	0.656	0.681

[7] G. Batista, R. C. Prati, and M. C. Monard, “a study of the behavior of several methods for balancing machine learning training data,” *ACM Sigkdd Explorations Newsletter 6 (1), 20-29*, 2004.

[8] “Tanzania water and sanitation network,” http://www.tawasanet.or.tz/index.php, accessed: 2023-04-25.

[9] “Weights and biases,” https://wandb.ai/, accessed: 2023-04-24.

[10] “Gridsearchcv,” https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, accessed: 2023-04-25.