Abstract

In this work, we consider a dataset of 59,401 water pumps in Tanzania in which 7 continuous features and 31 categorical features are labeled with a target category of functional, functional needs repair, or non-functional, corresponding to the operational status of the pump. After appropriate data preprocessing, simple imputation of missing values, and a test-train split, Naive Bayes, K-nearest-neighbors, Logistic Regression, and Random Forest models are evaluated and hyperparameter-tuned in order to predict which performance category each pump in the test set will belong to. KNN, Random Forest, and Logistic Regression generally performed well across the metrics tested. Random forest achieved one of the best area-under-receiving-operating-curves (AUROC) with a score of 0.844 and a relatively high area-under-precision-recall-curve (AUPRC) score of 0.523, and was a useful example for feature importance extraction from the model. We used mean-decrease-in-impurity and permutation-based feature importance methods on the random forest model. Common important features among these methods include dryness of the pump reservoir, population the pump serves, and geographic location of the pump. These results agree with intuition and are useful predictors for the success of a new pump installation in Tanzania. We suggest future work such as dimensionality reduction and time-analysis of the pump construction years.

Introduction

Our team is participating in the "Pump it Up: Data Mining the Water Table" challenge hosted by DrivenData. In 2010, the United Nations recognized the right of every human being to have access to enough safe and affordable water for personal and domestic use. Furthermore, the water source must be within 1,000 meters of the home and collection time should not exceed 30 minutes. Water pumps are critical to securing the right to clean and drinkable water for the population of Tanzania, especially those who live in rural villages. The ability to foresee issues with water pumps and address them preemptively is important to ensure the continuous availability of clean, potable water.

Our approach to classifying the functionality of water pumps is using various machine learning models and then optimizing the models with the greatest AUPRC results from model assessment. The selected models are Naive Bayes, Logistic Regression, K-Nearest Neighbors, and Random Forest. Unlike previous approaches to the problem, we used KNN-imputation to fill in a lot of the missing values in the dataset, making the overall predictions more accurate. Finding a model that is highly accurate in predicting the functionality of a water pump would greatly aid in current humanitarian efforts in resource distribution.

Background

The dataset used in this project focuses on classifying the functionality of water pumps specifically in Tanzania. However, similar studies about water pumps in other regions of Africa have been conducted. This past August, a research paper titled "Contribution of physical factors to handpump borehole functionality in Africa" was published describing the use of machine

learning approaches to classifying the functionality of handpumps in Ethiopia, Uganda and Malawi. In contrast to our large dataset that contains information about more than 59,000 water pumps, they only examined 145 hand pump boreholes in the target region. The researchers initially used standardized functionality classification to classify each pump into one of six categories: fully functional (FF); unreliable (UR); low yield (LY); unreliable and low yield (UR-LY); non-functional (NF); and abandoned (AB). Then, they conducted multinomial regression to drop the abandoned and non-functional categories. The remaining four functionality categories were used as discrete categorical dependent variables. Regression models implemented a step-wise approach to optimize the set of independent variables in the model. However, they didn't go into depth about their model assessment methods. After their analysis, they determined hydrogeological conditions along with some other factors were very important to the functionality of the handpumps.

Since our dataset came from an online DrivenData challenge, there have also been other approaches (not published as research papers) more specific to classifying water pumps in Tanzania. One approach tested six different models, including random forest, XGBoost, decision trees, k-nearest neighbors, support vector machines, and logistic regression. For pre-processing, they implemented one-hot encoding, but it looks like they didn't account for the plethora of missing values in the dataset. They evaluated their models with F1 score and found that random forest performed the best with a score of around 0.77, followed by XGBoost and then decision tree, though XGBoost had the longest run time.

Methods

We decided to implement four different learning algorithms: naive bayes, logistic regression, k-nearest neighbors, and random forest. Naive bayes (NB) is a supervised machine learning algorithm that expands on the "naive" assumption that every pair of features, given the class variable value, is conditionally independent and makes an equal contribution to the predicted outcome. It uses the Bayes rules to calculate the posterior P(Y|X): $posterior = \frac{prior \times likelihood}{evidence} \Rightarrow P(Y|X) = \frac{P(Y) \cdot P(X|Y)}{P(X)}$. NB is useful for the problem of predicting the functionality of water pumps because from preprocessing work conducted, we know that not all the variables are independent. So, implementing this model would act as a sort of controlling model verifying our findings.

Logistic regression (LR) is also a supervised learning algorithm for when the predicted outcome is discrete. It's typically used for predicting the probability of a binary event occurring. However, multinomial logistic regression is used when there are multiple possible outcomes, which is applicable for our problem of classifying whether a water pump is functional, non-functional, or function but in need of repair.

K-nearest neighbors (KNN) is a supervised machine learning algorithm that uses proximity to make classifications and predications for a point. It essentially examines a k number of data points that are closest to the current data point and then classifies based on the majority. KNN is useful for predicting the functionality of water pumps because it is a multiclass

classification problem. For hyperparameter tuning, we used grid search. GridSearchCV is an exhaustive search for the best parameters values using cross validation.

Random forest (RF) is an ensemble classification approach that utilizes many decision trees and their leaf nodes to make predictions. It also implements bagging and feature randomness. Each split in a decision tree considers a random group of features, and the tree is grown to its maximum size without pruning. The final predictions obtained are then aggregated over the B number of trees. An advantage of the random forest algorithm is that in comparison to regular decision trees, it reduces variance, though at the cost of a slight increase in bias, and it has an improved performance. It's also robust to errors and outliers. Similar to KNN, for hyperparameter tuning, we used grid search.

Experiments/Results

In this challenge, we are tasked with predicting which water pumps are faulty from a dataset collected by Taarifa and the Tanzanian Ministry of Water. There are 59,400 samples and 38 features. The features cover the type and quality of pump, the date of installation and location, and the management of the pump. There are a majority of categorical variables. The target variable has three possible values: functional, functional but in need of repair, and non-functional.

Variable Name	Variable Description	Variable Type	# of NA's	# of Unique Values
amount_tsh	Total static head (amount water available to waterpoint)	Categorical	0	98
date_recorded	The date the row was entered	Continuous, datetime	0	356
funder	Who funded the well	Categorical	3,635	1,897
gps_height	Altitude of the well	Continuous	0	2,428
installer	Organization that installed the well	Categorical	3,655	2,145
longitude	GPS coordinate	Continuous	0	57,516
latitude	GPS coordinate	Continuous	0	57,517
wpt_name	Name of the waterpoint if there is one	Categorical	0	37,400
num_private	No description given	Continuous	0	65

basin	Geographic water basin	Categorical	0	9
subvillage	Geographic location	Categorical	371	19,287
region	Geographic location	Categorical	0	21
region_code	Geographic location (coded)	Continuous	0	27
district_code	Geographic location (coded)	Continuous	0	20
lga	Geographic location	Categorical	0	125
ward	Geographic location	Categorical	0	2,092
population	Population around the well	Continuous	0	1,049
public_meeting	‘True/False’ is only description given	Binary	3,334	2
recorded_by	Group entering this row of data	Categorical	0	1
scheme_management	Who operates the waterpoint	Categorical	3,877	12
scheme_name	Who operates the waterpoint	Categorical	28,166	2,696
permit	If the waterpoint is permitted	Binary	3,056	2
construction_year	Year the waterpoint was constructed	Continuous	0	55
extraction_type	The kind of extraction the waterpoint uses	Categorical	0	18
extraction_type_group		Categorical	0	13
extraction_type_class	The kind of extraction the waterpoint uses	Categorical	0	7
management	How the waterpoint is managed	Categorical	0	12
management_grou	How the waterpoint is	Categorical	0	5

p	managed			
payment	What the water costs	Categorical	0	7
payment_type	What the water costs	Categorical	0	7
water_quality	The quality of the water	Categorical	0	8
quality_group	The quality of the water	Categorical	0	6
quantity	The quantity of water	Categorical	0	5
quanity_group	The quantity of water	Categorical	0	5
source	The source of the water	Categorical	0	10
source_type	The source of the water	Categorical	0	7
source_class	The source of the water	Categorical	0	3
waterpoint_type	The kind of waterpoint	Categorical	0	7
waterpoint_type_group	The kind of waterpoint	Categorical	0	6

Table 1: Variable names, descriptions, null values, and unique values from original dataframe

Since scheme_management encodes similar information as scheme_name, but has fewer missing values and fewer unique values, we will drop scheme_name. The variable recorded_by has only one unique value, thus it does not carry any additional information and is dropped. The variable num_private has no description, so we dropped it. We extract year_recorded and month_recorded from the datetime variable date_recorded, as raw datetime variables are not able to be used as features in models. Next, we performed analysis to find erroneous values.

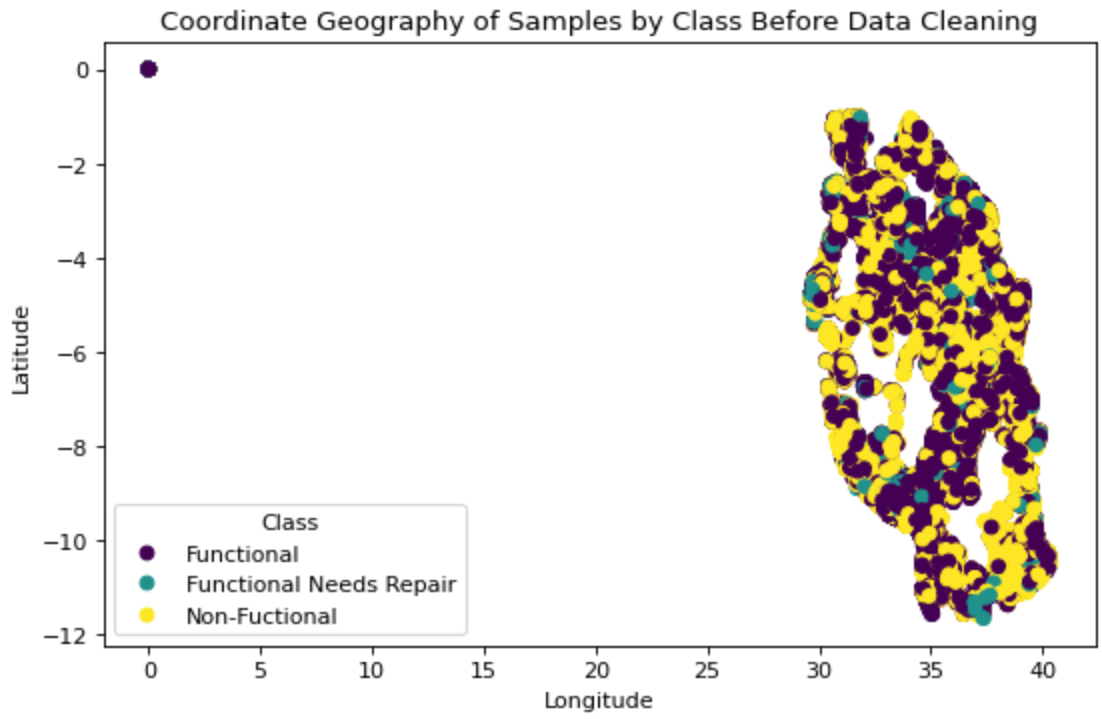

Figure 1: Longitude and latitude values of samples before data cleaning

There are a group of samples with erroneous values of longitude coordinates. This is confirmed by the fact that Tanzania has a longitude range between 29 and 40 degrees. 1,812 samples in the dataframe were found with longitude equal to 0 and latitude equal to -2 x e^-8. These samples were changed to null values in those features.

Similarly, samples with construction_year equal to 0 were changed to null values in that feature.

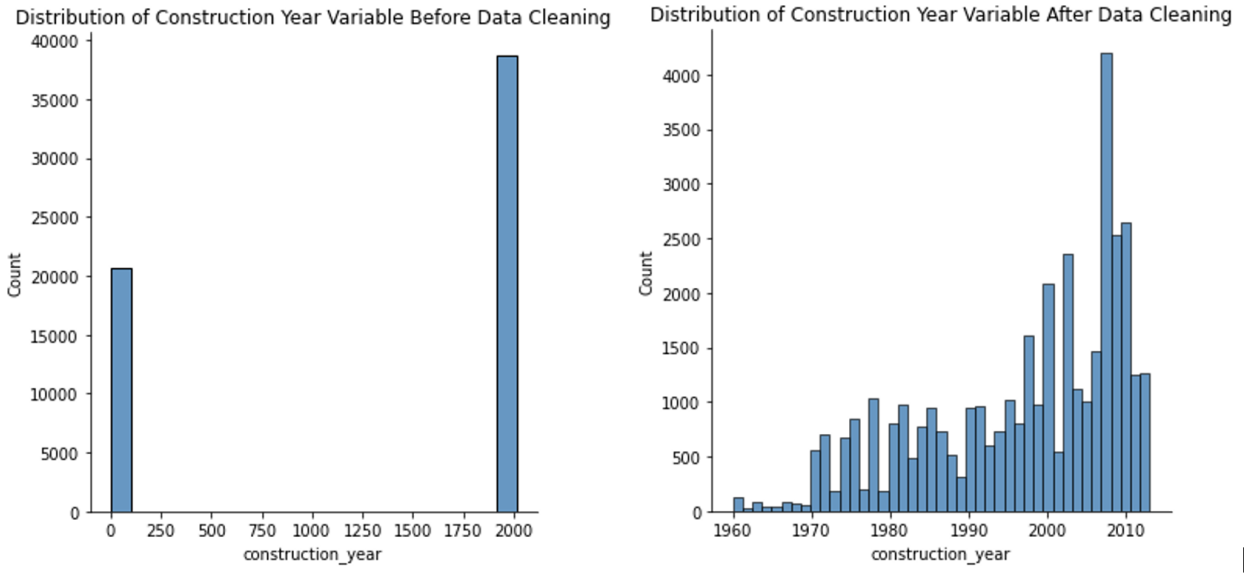

Figure 2: Histograms of construction_year variable before and after data cleaning

We also noticed 21,381 samples with population equal to 0, which we changed to null values in that feature. We found it very unlikely that so many wells would be constructed without people to use them. However, we do recognize that there may have been legitimate zero-population estimates that we changed (population around a well may have left due to climate change, urbanization, etc.).

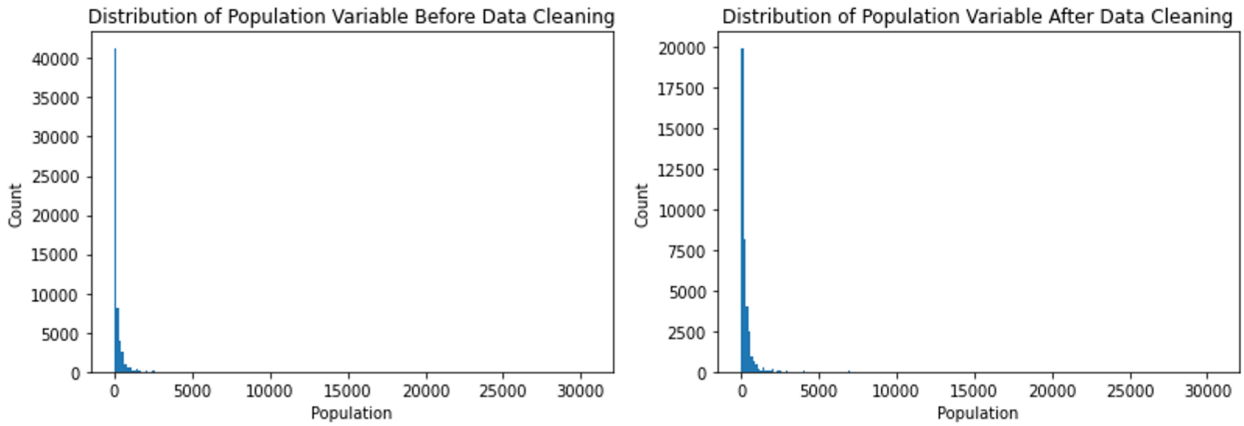

Figure 3: Histograms of population variable before and after data cleaning

We found additional erroneous values in the variable amount_tsh (water available to the waterpoint). In this case where amount_tsh was coded as zero but the well was labeled as functional or functional in need of repair, we changed the variable to null because it is impossible for a well to be functioning without access to water.

Next, we decided to reduce the number of factors in some of the categorical variables. For example, the variable "subvillage" has 19,287 unique values. Lumping the rare subvillages into an "Other" group makes the data more general, which will help our algorithms detect broader patterns and will also reduce the amount of computational power needed. One-hot encoding just "subvillage" would have resulted in an additional 19,287 sparse columns to the dataframe. Thus, we aimed to collapse each categorical variable to the maximum number of factors where each factor had at least 100 samples. "Subvillage" was reduced to 20 factors, "installer" to 80 factors, "ward" to 50 factors, "wpt_name" to 20 factors, and "funder" to 90 factors.

We then examined the null values in the dataset by variable and class to determine if there was a pattern to the missing values.

Class	amount_tsh	funder	installer	longitude	subvillage	population	public_meeting	scheme_management	permit
F	19706	1981	2000	870	205	11274	1678	1873	1673
FR	0	1217	1215	556	165	8332	1500	1781	1083
NF	3048	437	440	386	1	1775	156	223	300

Table 2: Missing values by class, where F = Functional, FR = Functional in need of repair, and NF = Not functional.

Since the dataset is imbalanced, we did not expect an equal number of missing values for each class. As shown in Figure 3, the majority class is "functional" and the minority class is "functional in need of repair," with "non functional" in between. However, it is surprising that we see a pattern of the "functional in need of repair" class having more null values than the "non functional" class, when the former has far fewer samples than the latter. The exception is the variable "amount_tsh." We believe that the high proportion of missing values in the class "functional in need of repair" may be due to the fact that the types of wells that can function while damaged may have an unrecorded variable, like accessibility, that prevents full data collection. However, future work ought to further analyze this pattern.

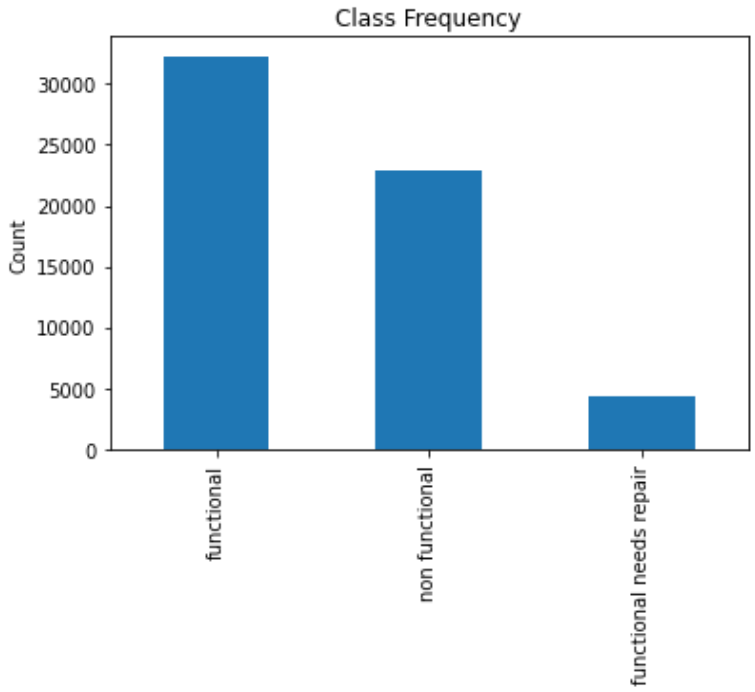

Figure 3: Bar plot of data labels

Before we began addressing the missing values in the dataset, we split the data into training and testing sets to prevent data leakage. We chose a test size of 30% of the total set. We then one-hot-encoded our categorical columns, which resulted in a total of 574 columns. We then standard scaled all columns except the binary and one-hot-encoded columns. Finally, we applied knn-imputation to address the missing values.

Next, we addressed the imbalanced nature of our data by upsampling the two minority classes. We used sampling with replacement on the classes “non functional” and “functional in need of repair” to achieve three equal classes of 22,627 samples each.

The models we selected were Naive Bayes, Logistic Regression, K-Nearest Neighbors, and Random Forest. We chose Naive Bayes because it is relatively fast and simple to implement, which makes it a good choice for applications where speed is a concern. In our dataset, we have a lot of features and rows, so speed is definitely a concern. Naive Bayes also functions on the assumption that all features are completely independent from each other. From the pre-processing steps, we know that is not true for these features. So, implementing Naive Bayes acts similar to a control variable as we know it should not perform the best considering not all of our features are independent.

Logistic regression was chosen because it can be regularized to prevent overfitting, which makes it a good choice for data sets with a large number of features. Multinomial logistic

regression can also be used for multiclass classification problems, which is exactly what our prediction goal is. Logistic regression does not really have any critical hyperparameters to tune.

K-Nearest Neighbors was chosen because it is a simple and intuitive method for classification and regression tasks, which makes it easy to understand and implement. KNN is a non-parametric method, which means that it does not make any assumptions about the underlying data distribution. This makes it more flexible and better suited for data sets with complex or non-linear relationships. It's a lazy learning method, which means that the training process is relatively fast, and the model is only updated when a prediction is requested. This makes it a good choice for applications where the training data is large or the model needs to be updated frequently. It can easily be used for multiclass classification problems. In terms of hyperparameter tuning to select an optimal k value, we used a manual implementation of a grid search.

Random forest was selected for our ensemble method, which means that it combines the predictions of multiple decision trees to improve the overall performance of the model. This makes it less susceptible to overfitting and more robust to noise and outliers in the data. It can be trained in parallel, which makes them faster and more efficient than other tree-based methods. It also provides a good trade-off between accuracy, interpretability, and computational efficiency. Based on background research conducted, previous work classifying water pump functionality also implemented random forest, confirming our decision to use the model. A manual grid search was used for hyperparameter tuning to determine the most optimal ensemble size, criterion, max depth, and min-leaf samples.

Before any model evaluations were performed, the baseline accuracy from randomly guessing the majority class every time was conducted. Its accuracy was 54.1%, meaning that the target accuracy for the selected models was greater than 54.1%. The model evaluation metrics used were Area Under Receiving Operating Curve (AUROC), Area Under Precision-Recall Curve (AUPRC), and F1-Score. For AUROC, the x-axis is the false positive rate (FPR), and the y-axis is the true positive rate (TPR). Typically, an AUROC score of 0.8 indicates a good prediction. For AUPRC, the x-axis is recall, and the y-axis is precision. Typically, an AUPRC score of 0.6 - 0.7 is good, but scores under 0.5 are pretty common. For AUPRC, since our problem is a multi-label problem, the overall AUPRC was calculated based on the micro-average, the quantifying score on all classes jointly. To analyze the results, AUPRC was primarily examined. F1-score is preferred when we have a class imbalance. To calculate F1-Score, the formula is $\frac{2 \times precision \times recall}{precision + recall}$. Below are the empirical results and comparison for all the models.

Model	AUROC	AUPRC (Micro-Averaged)	F1-Score
Naive Bayes	0.799	0.494	0.546

Logistic Regression	0.844	0.544	0.571
K-Nearest Neighbors	0.836	0.596	0.626
Random Forest	0.844	0.523	0.491

Table 3: Empirical results for each model.

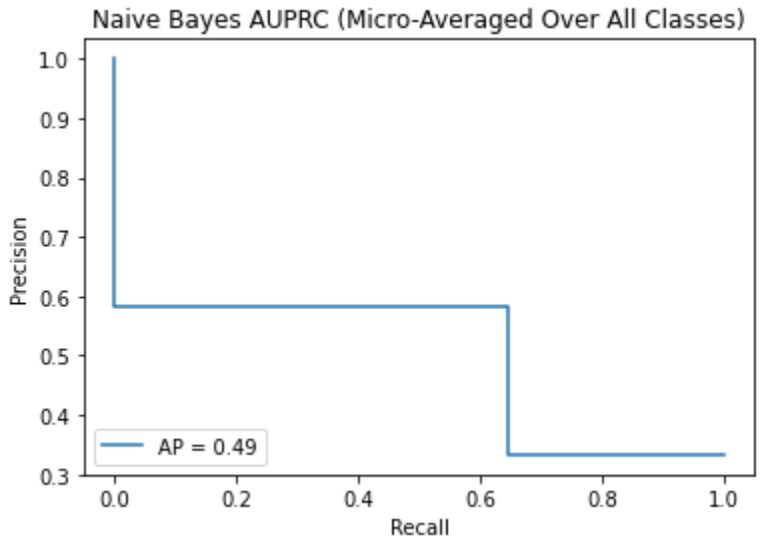

Figure 4: Naive Bayes AUPRC Graph.

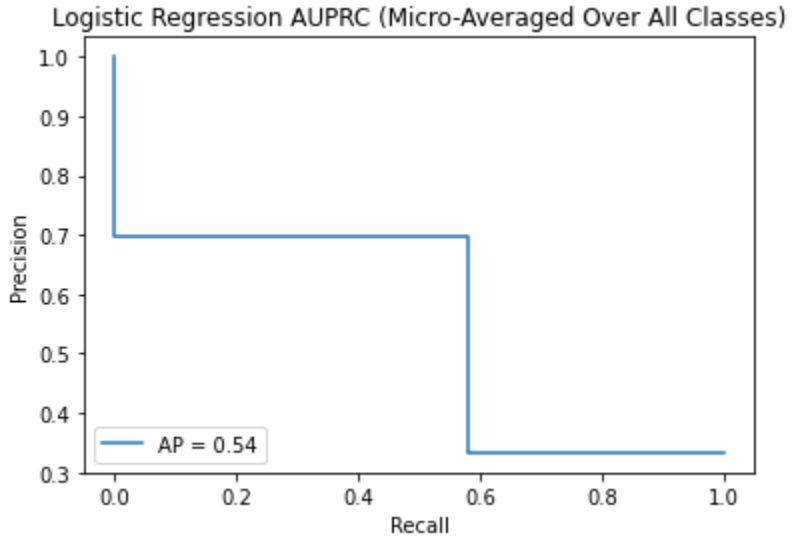

Figure 5: Logistic Regression AUPRC Graph.

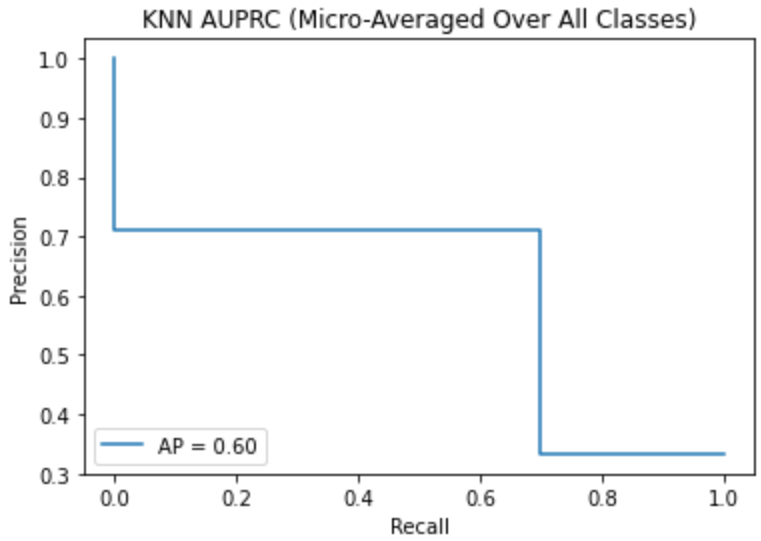

Figure 6: KNN AUPRC Graph.

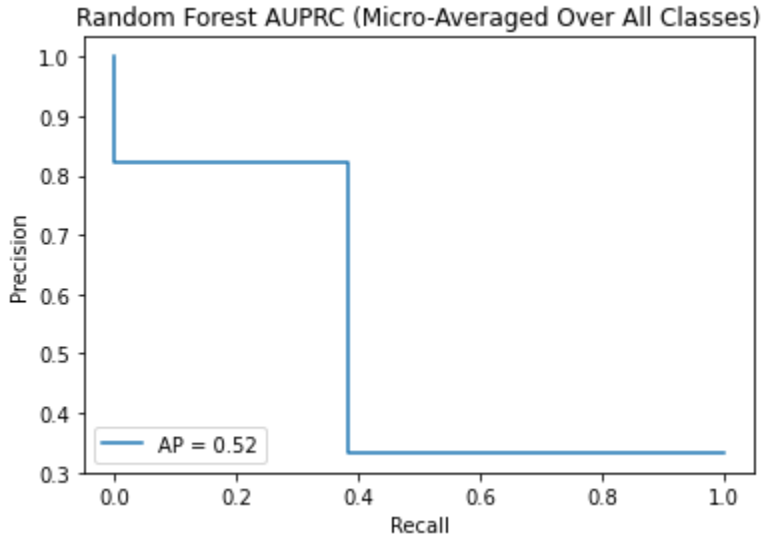

Figure 7: Random Forest AUPRC Graph.

The optimal hyperparameters are k = 4 for the number of nearest neighbors in KNN. For the random forest, the optimal forest size is 41, criterion is entropy, max depth is 25, and min-leaf-samples is 1. However, it is worth remembering that only a small set of hyperparameter values could be searched, since fitting the model repeatedly becomes very computationally expensive very quickly as there are four hyperparameters to tune and thus a quadruply-nested for loop. Note that we are most interested in the AUPRC metric because we want the highest sensitivity with respect to the performance classifying the positive class. However, the AUROC metric is also of high interest. In general, random forest performs the best of the considered models, and has an AUROC of 0.844 and an AUPRC of 0.523. KNN and logistic regression also

perform well, but do not provide much room for analysis of feature importance as compared to the random forest model.

We are ultimately interested in the features that are most useful for predicting pump performance in Tanzania. For the random forest model, it is possible to think of these as features with the greatest information gain, which usually might appear near the roots of the trees in the ensemble. Even though it was outperformed on some metrics, we choose to use the random forest as a model for feature importance and evaluate it based on AUROC, the random forest's strongest metric. Using sklearn's built-in random forest feature importance, it is possible to efficiently extract the most important features of the random forest. This can be done with two methods - mean decrease in impurity (MDI) and permutation-based methods (PM). In MDI, the mean (across all trees in the forest) decrease in node impurity is computed for each feature. In PM, the score assigned is proportional to the decrease in accuracy caused by shuffling a particular feature in the out-of-bag (OOB) samples over many permutations (see permutation importance article). In other words, if the model performance suffers greatly by "shuffling" the information of a particular feature, then that feature must be quite important. The top 5 and bottom 5 (by direct sorting) most important features, in terms of each metric, are shown below.

MDI, Feature	MDI, Value	PM, Feature	PM, Value
longitude	7.877976e-02	quantity_group_dry	0.006510
latitude	7.788767e-02	population	0.003844
gps_height	4.365414e-02	latitude	0.003760
quantity_group_dry	3.691732e-02	quantity_dry	0.003507
population	3.166807e-02	waterpoint_type_other	0.001740
…	…	…	…
waterpoint_type_dam	6.194876e-06	source_class_groundwtaer	-0.002132
lga_Lindi Urban	4.818504e-06	extraction_type_class_gravity	-0.002160
lga_Nyamagana	3.860338e-06	district_code	-0.003002
scheme_management_none	1.578113e-07	amount_tsh	-0.003311
Extraction_type_other - mkulima/shinyanga	0.000000e+00	extraction_type_gravity	-0.003844

Table 4: Sample results of feature importance analysis of the random forest model described earlier. **MDI, Value** and **PM, Value** are sorted highest-to-lowest by numerical value, and only the top 5 and bottom 5 results are shown. Note that because of random processes used to generate these results, results may vary slightly when re-run, although it should be similar.

Discussion

The results of Table 4 yield insight as to the most relevant features for building the random forest. For example, latitude and longitude tend to be important (ie, geographic location is relevant), the population is important, and dryness of the well (quantity_dry and quantity_group_dry) is important. In other words, the features listed are features that are generally most relevant at predicting water pump functionality for our random forest model. It is reassuring that the relevant features are in agreement with real-life intuition – we can reasonably justify that location, dryness, and population using each pump should have an impact on the performance of the pump.

The results of this analysis show that several classical machine learning models are successful at predicting categorical (functional, functional needs repair, not functional) performance of water pumps in Tanzania based on available features from our dataset. The most successful and intuitive of these models was the random forest, which achieved a 0.891 AUROC score after hyperparameter tuning. The random forest model also guided us towards the most important features for prediction, some of which include the geographic location of the well, population that the well serves, and whether or not the well's reservoir is dry.

We hope that such analysis can serve as a guide for humanitarian efforts to install future pumps in Tanzania. For example, a prospective pump funder aiming to install a new pump in Tanzania may wish to consider our model to decide if a proposed installation will be successful, based on the features of the pump they wish to install. For example, running the proposed project (with as many features as possible that align with the features of our dataset) through the random forest model will output a prediction as to whether a pump with those features might be functional or not. Our analysis so far suggests that the most important factors for prospective installers to consider are geographic location, population that the well should serve, and the dryness quality of the well's reservoir, among others.

Some possible avenues for future work include dimensionality reduction and more comprehensive time-analysis. In terms of dimensionality reduction, notice as an example that the random forest's feature importance **PM, Feature** column includes two similar attributes, quantity_group_dry and quantity_dry. There is somewhat redundant information from these features, i.e, both describe the dryness of the reservoir in a similar way. Dimensionality reduction techniques such as principal component analysis (PCA) may help reduce these redundancies, although somewhat at the cost of interpretability.

In terms of the time-analysis, it might be useful to consider only the contributions from a particular feature (e.g, construction_year) and determine if there is a critical value where we

might see a significant response in the target category (for the same example, maybe pumps installed after 1997 tend to map to functional pumps much more strongly than pumps older than this date). For example, such analysis of the construction_year feature might yield a rough estimate of the lifetime of pumps installed in Tanzania (in this example, 25 years), and maybe it is even possible to trace which other features contribute most to this pump lifetime. The expected lifetime of certain pumps is valuable information for repair teams choosing how to spend limited resources, and it may be possible to extract that information from the models described in this report. However, we leave such analysis to future work.

Contributions

Leah Smith:

- Performed all the data preprocessing up until running the actual models
- Wrote the data description section and the preprocessing-related paragraphs in the experiments/results section

Mei Deng:

- Implemented and conducted model assessments for Naive Bayes and Logistic Regression models and AUPRC model assessments for all models
- Wrote background, methods, and model selection (under experiments), and AUPRC results (under experiments)

Josh Peacock:

- Conducted initial data preprocessing efforts that led to revisions of final preprocessing
- Implemented and conducted model assessments for KNN and Random Forest model
- Wrote about abstract, model assessment, and results/discussion

Code/Dataset

- Google Drive - note: please only refer to the following files
  - Folder: CS334_FP_preprocessing_version1
  - ML Final Project.ipynb
  - training_set_labels.csv
  - training_set_values.csv
  - xTest_s_knn.csv
  - xTest_s_mode.csv
  - xTrain_s_knn.csv
  - xTrain_s_mode.csv
  - yTest.csv
  - yTrain.csv
- Google CoLab