Water Pumps in Tanzania

Matt Bonfante

Julian Ghadially

Vijai Kasthuri Rangan

Anchit Singh

Molly Wolfe

Abstract:

Tanzania suffers serious issues involving the lack of water it is able to provide for its people. For Tanzania, shedding the image of a "third-world" nation and striving to emerge as a thriving, middle-income one, starts with clean, easily accessible, water. Our goal was to help the people of Tanzania determine the functionality of every water pump so those not working properly could be fixed as quickly as possible. We approached this as a classification problem, in which we tried to classify any pumps that were functional but needed repair. Using the knowledge we have gained about machine learning techniques in our course, we were able to achieve a high level of accuracy in predicting the functionality of pumps. This will in turn help the Ministry of Water in Tanzania by identifying which pumps need to be repaired and therefore save time and money by allowing them to plan properly and optimize their resources.

Introduction/Background:

Water is a basic need for all human beings. In 2007, the World Bank committed to providing Tanzania with the technical and financial resources needed to combine rural and urban water resource management into one plan. When the project started, about 54% of Tanzanians had access to functioning water resources. According to metrics to date, only 53% of Tanzanians have access to functioning water [(1)]. Even with the $1.42 billion dollars that the water project raised (combination of money from the World Bank, various donors, and the Tanzanian government), water was still an issue for Tanzania [(1)]. Many of new newly built pumps were not maintained and are now in danger of failing across the communities.

Our motivations for choosing this project was to help The Ministry of Water in Tanzania identify which water pumps are functional but need repair. The Ministry of Water in Tanzania is an organization that strives to supply clean, safe, sustainable water at an affordable price for all citizens. The cost to repair a pump that has already been installed is much more efficient (with both money and time) than to replace the pump entirely. This knowledge drove our decision to focus on identifying the pumps that are functional but need repair. By identifying these pumps,

we hope for the quickest turnaround in order to provide the people in Tanzania with continuous access to an improved water source.

Data Acquisition/Description:

The dataset we used for this project was provided by Taarfia (an open sourced platform for the crowd sourced reporting and triaging of infrastructure related issues) and The Ministry of Water in Tanzania. We acquired the dataset through one of the competitions on Driven Data website [(2)]**.** The objective of the data is to predict whether or not a water pump is functional, non-functional, or functional needs repair, with a focus on identifying those pumps that are functional and needing repair. See below for definitions of each:

- **Functional** - A water pump yielded good quality water during the survey or it had no technical problems even if water was not present in a water point but it is available seasonally
- **Non-functional:** A water pump did not yield water for more than six months of the year. Reasons for non-functionality can be due to being in a dry region, producing poor water quality (for example too salty or too much fluoride), or due to management. Non-functional water pumps also include those that are under construction but not yet operational
- **Functional Needs Repair**: A water pump that is functional / operational but needs repair to operate at capacity

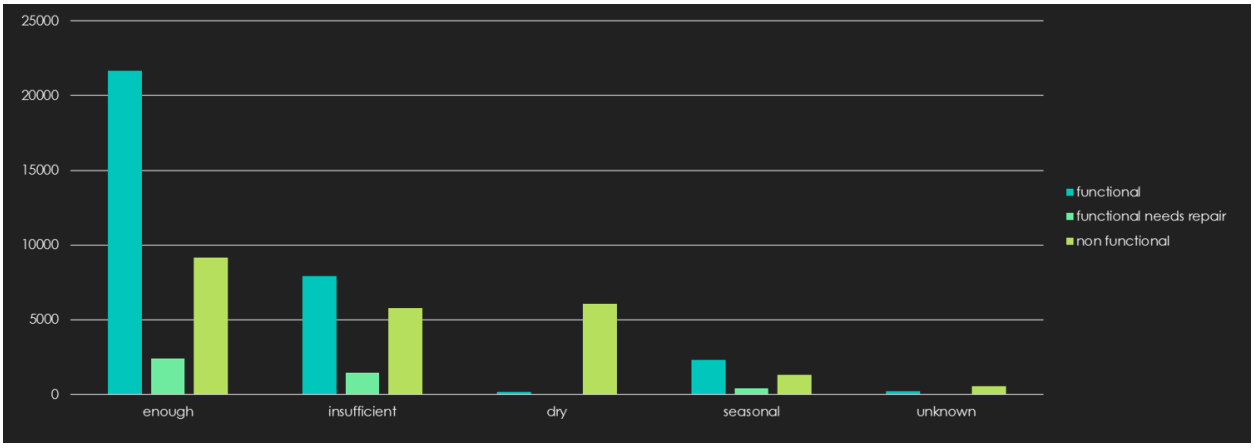
The data was collected using a combination of handheld sensors, paper reports, and user feedback reported via cell phones. There are 40 features total, including demographics (such as region, population, latitude, and longitude), water quality, construction year, etc. There are a total of about 75,000 rows, each row representing data at a different water pump.

Exploratory Data Analysis:

One way to visualize the data is by assessing the conditional distribution of our response variable on the variables that we initially considered as important. In our results, we will discuss which variables actually were important. These plots are shown below:

Water Quantity vs. Functionality

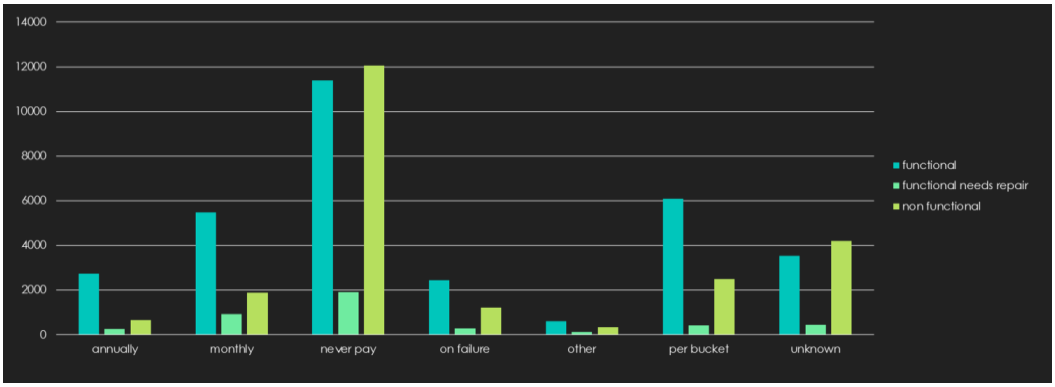
Figure 1



As seen in Figure 1 above, there are clear differences in the functionality proportions among the most significant water quantity levels. The 'enough' category has a high ratio of functional pumps compared to non-functional and FNR pumps. 'Insufficient' has a relatively balanced ratio, whereas 'dry' is almost completely composed of non-functional pumps. These results generally follow what we originally expected, i.e. it makes sense that as a pump has access to less water it is more likely to be non-functional. Dry water sources could possibly cause pipes to crack and become non-functional. Also, pumps with access to plenty of water are likely to be more utilized and therefore are made a priority to repair locally when there are minor malfunctions. There may not be this incentive for drier pumps which would not output water even if they were to be repaired.

Payment Type vs. Functionality

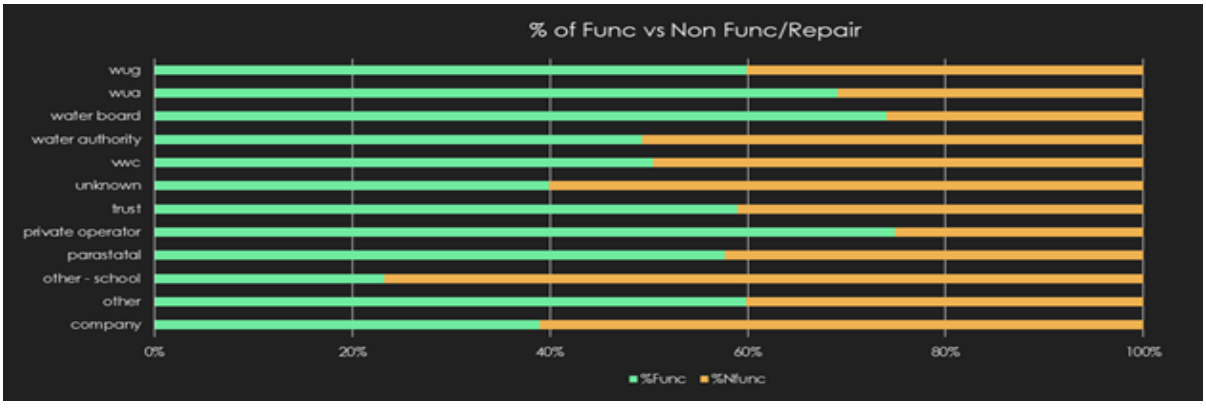
Figure 2



Above the three functionalities were plotted against their respective methods of payment. From this we can see that around half of the pumps are operated without requiring payment, while the other half is distributed among the following payment collection types; annual, monthly, per bucket, or on failure. What we found interesting was that all of the paid pump types had nearly identical distributions; very high proportions of functional pumps (nearly 80%). In contrast the 'never pay' pumps had the majority of pumps labeled as non-functional or FNR. From this we can draw the logical conclusion that pumps are far more likely to be functional if there is a source of revenue associated with the resource.

Management vs. Functionality

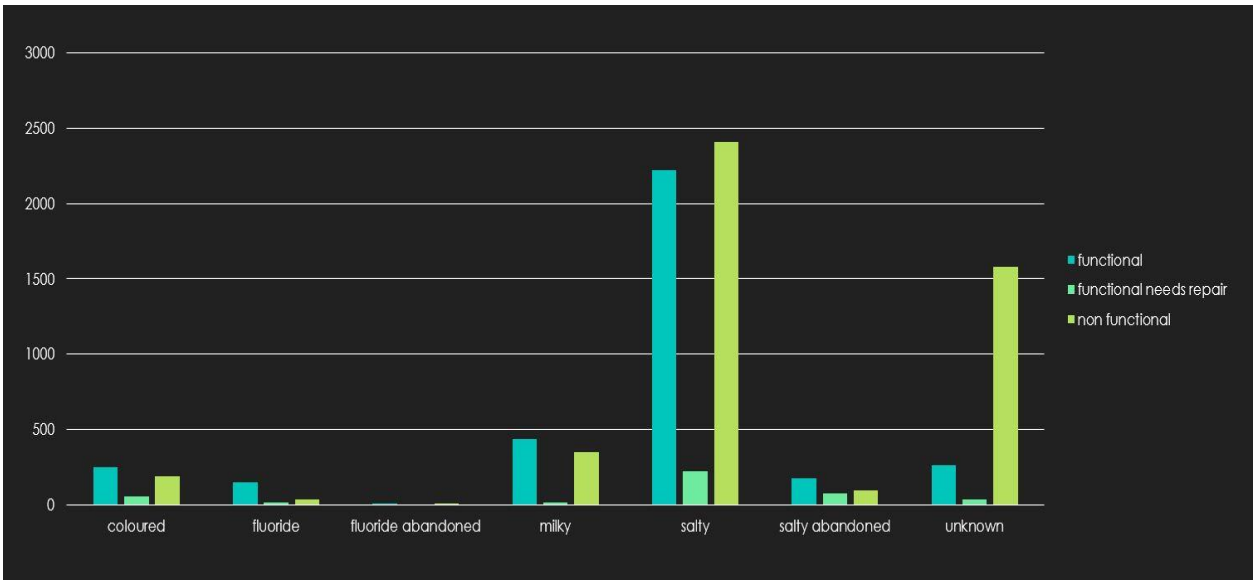
Figure 3



Next we wanted to evaluate whether functionality varied among different pump managers. The graph above shows each management company's proportion of Functional pumps (in green) compared to non-functional pumps (in orange). Our most significant finding related to the extreme categories, 'Private Operators' and 'School'. The pumps with the highest proportions of functional pumps were those managed by 'private operators'. These are pumps that are privately financed, and similar to the Payment Exhibit above this confirms that water typically flows to the money. The pumps that had the highest level of non-functional were the managed by schools. This shows the true severity of this water issue in Tanzania, and again provides evidence that functionality is tied to money as the Tanzanian schools are not properly financed.

Uninformative plots

Figure 4

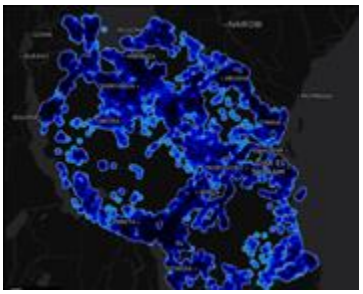


Above is an example of a plot which we tried that showed very little information. As you can see every category of water quality has very even and similar proportions of functionality groups. Graphs with results like these provided little insight as there was no way to distinguish differences among each individual categorical variable.

Density Maps

After initial exploratory analysis we wanted to utilize our location data (longitude & latitude) to get additional insights with respect to our country's geography. To do this we used cartodb.com which could plot each variable using our datasets location columns. The plots we found useful are described below:

Figure 5



Functional

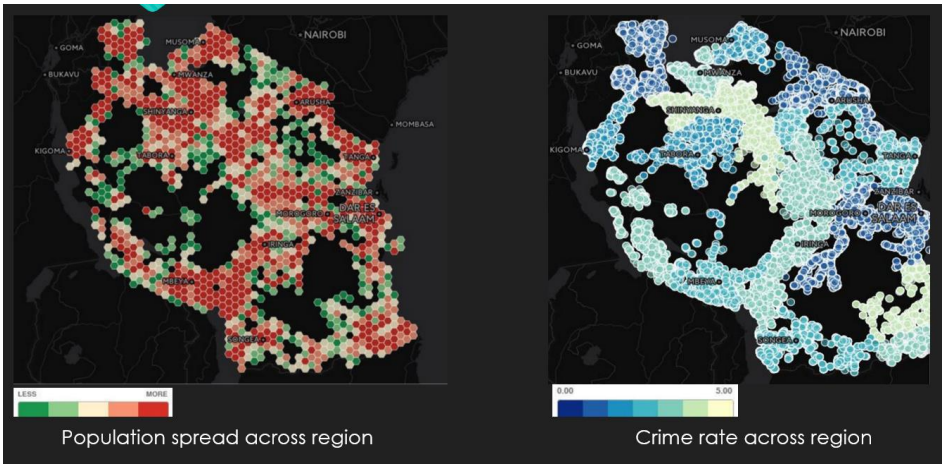
Functional Needs Repair

Non-Functional

We started by taking a look at the distributions of each pump functionality across Tanzania. From this we can see that Functional and Non-Functional pumps are similarly distributed across the country. One difference that can be noted is that the functional map has a noticeably higher density in the regions around major cities. This is likely due to the fact that these pumps serve a lot of the population and are therefore more regularly maintained. It is also worth noting that "Functional-Needs-Repair" pumps cover a smaller overall area and they are almost unrepresented in the two main unpopulated "holes" in the maps, unlike Functional and Non-Functional which cover a lot more of this area.

Population & Crime Density

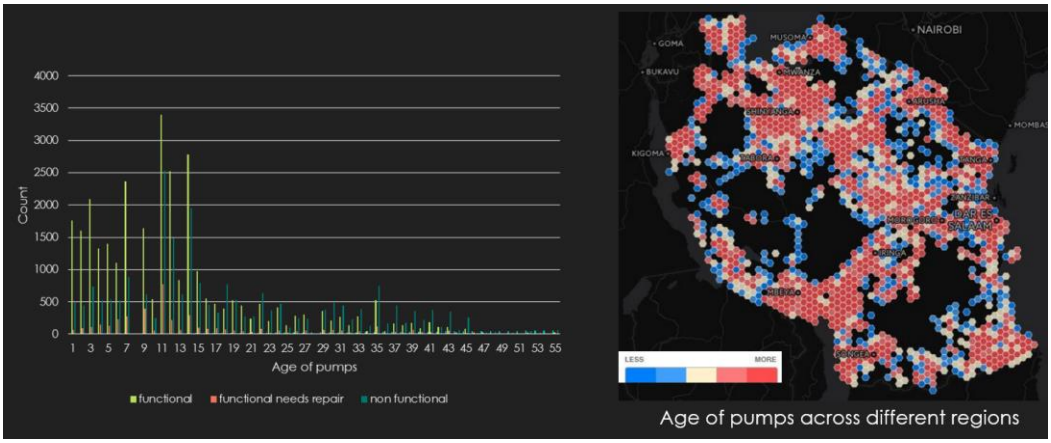
Figure 6



Next we evaluated the crime density across Tanzania. We thought that areas with higher crime rate may result in higher levels of Non-Functional pumps due to vandalism. Although at this point there is no way to prove causation, we did notice that areas of high crime density matched the areas of high Non-Functional density from the previous plots above.

Age Distribution & Density

Figure 7



Finally we wanted to see the distribution of pump ages split by functionality and then see these overall pump ages distributed on the map. From observing the graph on the left we saw that the majority of the pumps were built in the past 20 years and that the majority of pumps are

"functional" across this time period. On the other hand, the remaining pumps over the age of 20 had much higher distributions of "Non-Functional" pumps. This makes sense as older pumps are more likely to break down due to older pumps. From the map we see that older pumps are clustered around major cities while newer pumps are being built in the less populated areas. This would follow a construction pattern moving from the most populated areas, and then expanding outward over time to the rural locations.

Data Pre-Processing:

Before performing any models on our dataset, there were many decisions to be made around cleansing. Below are the issues we identified with data cleansing and how we chose to fix them in our dataset.

1. **Numerical Variables:** There are three numerical features that we manipulated as part of the data cleansing process.
    - **Construction Year** - One feature we thought might be important is the age of the water pump, since older pumps are less likely to be functional and this was visible when we performed the exploratory data analysis. We determined that we could use the Construction Year and Date Recorded features to compute age for each pump. However, more than 1/3rd of the training data was missing the Construction Year or was 0. From our initial analysis we found that the water point type, which contain information regarding the type of pump, to be an useful indicator of the construction year as different types of pumps were common across different periods. We also found the region would be a good indicator for the construction year as less developed regions had newer pumps in the exploratory analysis. Thus, we were able to condition the Construction Year on Water Point Type and Region and impute the median value for each segment. The overall median Construction Year was 2000.
    - **Age** - We subtracted Construction Year after imputing as above from Date Recorded to compute the Age. Upon adding age to the training data we were

able to remove the construction year and the date recorded from the training data set used for the models.

- **Population -** The Population field contained over 21,000 empty cells. Using the publicly available census data that provided population statistics[3] at the ward level, we were able to fill in the empty cells. This data was scraped from the city population website[3]. However, as we discuss in the results section, the lasso logistic regression performed worse in the hold-out set with the population and age variable included in the model.
- **Latitude & Longitude -** The latitude and longitude features had missing values. In order to create accurate spatial maps, we wanted to populate these fields. We used District and Region features to identify the missing latitude and longitude observations.

2. **Categorical Variables:** Out of the 40 features in our dataset, 31 were categorical variables.
    - Within each categorical feature, many of the data points were empty or N/A. The first step we took to clean this was to replace any blank or N/A cells to read "unknown" which would create a new level to capture any missing information.
    - Many of these features contained data in Swahili, a language native to Tanzania. We converted these into English and used the nltk package in Python to find the most frequent words among these translated words. We created levels for the most frequent words and excluded words that appeared infrequently.
    - Finally, we wanted to reduce the number of levels across the categorical variables that contained more than 32 levels (funder, installer and scheme_name). We reduced the number of levels for the scheme management, funder, and installer categories by selecting the top 10 most frequent categories amongst the functional needs repair pumps. We then selected these top 10 levels within each category from the larger training data set.
3. **Class Imbalance:** Among the three groups of pumps in our test dataset, the class we wanted to primarily predict (functional needs repair) only accounted for 7% (See **Figure**

8 for details). There were two different ways we approached this issue. The first was down-sampling. In order to choose an optimal level of down-sampling, different positive class proportions were tested against the validation set, using f1 score as our metric (see **Figure 9**). This tuning was validated against both the random forest and lasso logistic regression. The lasso logistic regression selected a down-sampling proportion of 20% positive class. The down-sampled dataset was used to train our f1-optimized models. Models that sought to compete on driven data for overall accuracy did not use the down-sampled dataset. The second approach we used to deal with the imbalanced dataset was to include a cost function in the model with the ROC curves. We used a 10:1 cost function for false positive as we assumed that the cost involved in making a mistake in the identifying a class as "functional needs repair" when it was another class would be 10 times more than identifying it right. We made this assumption as it would not be economically viable for the team to send out resources to repair to water points that were functional or non-functional. This allowed us to find the threshold from the ROC curves. We identified the best threshold that had the maximum sensitivity for a false positive rate that was less than 10%.

Figure 8

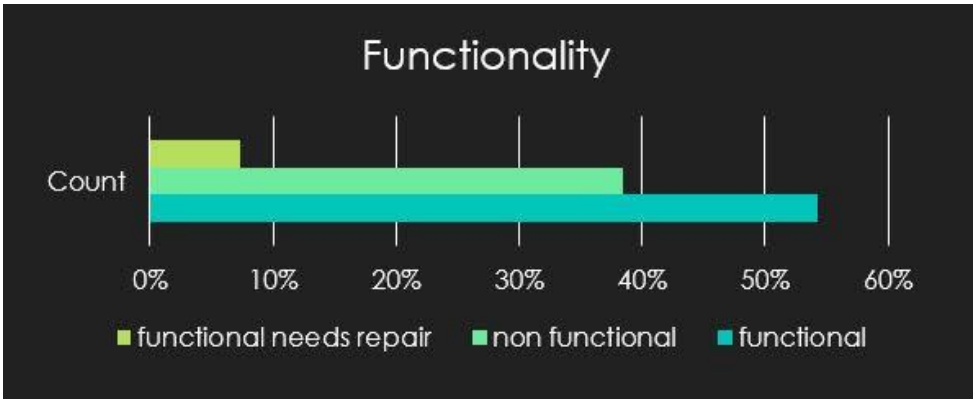
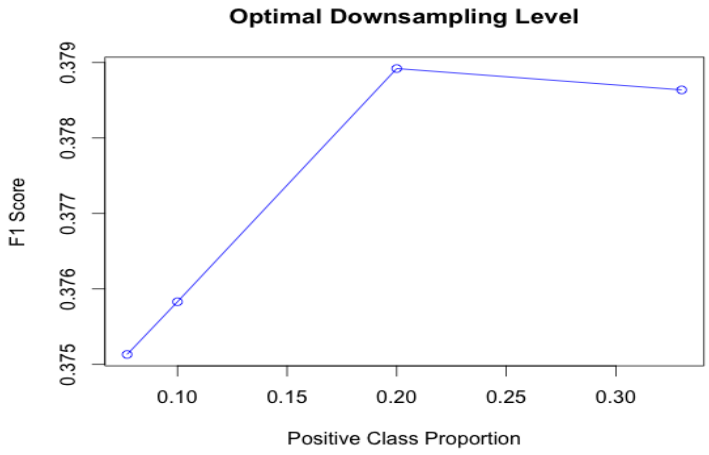


Figure 9



4. **External Data Additions**: There were several additional features that were not included in the original dataset that we thought could be helpful in predicting the status of a water pump. These features included crime rates [(4)], health status [(5)], and employment rates [(3)] among the different regions in Tanzania. We believed that regions higher water borne diseases related death rate could be an indicator for unavailability of sufficient water/water pumps. Similarly regions of high crime could indicate vandalism and hence more non-functional or pumps that would need repair. We were able to find these statistics online and add them as ordinals in our dataset.
5. **Feature Reduction**
    - Extraction: PCA was used for extraction. After dummy coding the categorical variables, we had many binary columns. Some of the categories were correlated (rolled up version for the same column). We applied PCA to reduce dimensions and extract uncorrelated dimensions. Also, with so many categorical variables we were not sure whether linear PCA would work. Hence we tried no linear PCA using Kernels.
    - Selection: lasso logistic regression model was used for selection. Using normalized data, variable importance was ranked by the magnitude of the coefficients. Because dummy variables that occur more frequently are theoretically more important, this ranking only reflects the importance of

variables with respect to those pumps that fall within that category. The results of this feature reduction was used in the multilevel model and as an exploratory data analysis tool. Out of the sparse, 1100+ dummy categories, Lasso, at a cross-validated lambda of 0.0028, selected 243 dummy variables. Categories within region, extraction type, scheme name, and local government authority ranked among the highest.

Predictive Models:

Regularized Logistic Regression

Using the optimal 20% downsampled data, as specified in the class imbalance section, we trained a multinomial logistic lasso regression. Using a hold-out set within the training sample, the lasso regularization parameter lambda was optimized to 0.0028 using f1 score as a metric. The inputs, which are covered in the preprocessing section, were normalized and converted to dummy variables. In order to deal with class imbalance, we had to adjust the threshold for classification. Using the same hold out set as we did to tune lambda, we first generated an ROC curve and selected the threshold that optimized f1 score, given its direct impact on our model criteria. This threshold was 0.40 (See Figure 10). Our overall f1 score on the hold-out set was .382 and the f1 score on the test set was .361.

Figure 10

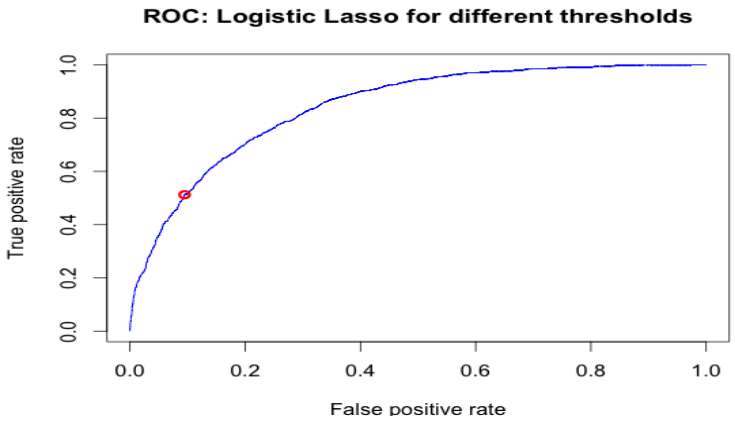


Figure 1: ROC curve for different thresholds on the Logistic Lasso. The Red circle represents the f1-optimized threshold selected from the hold-out set.

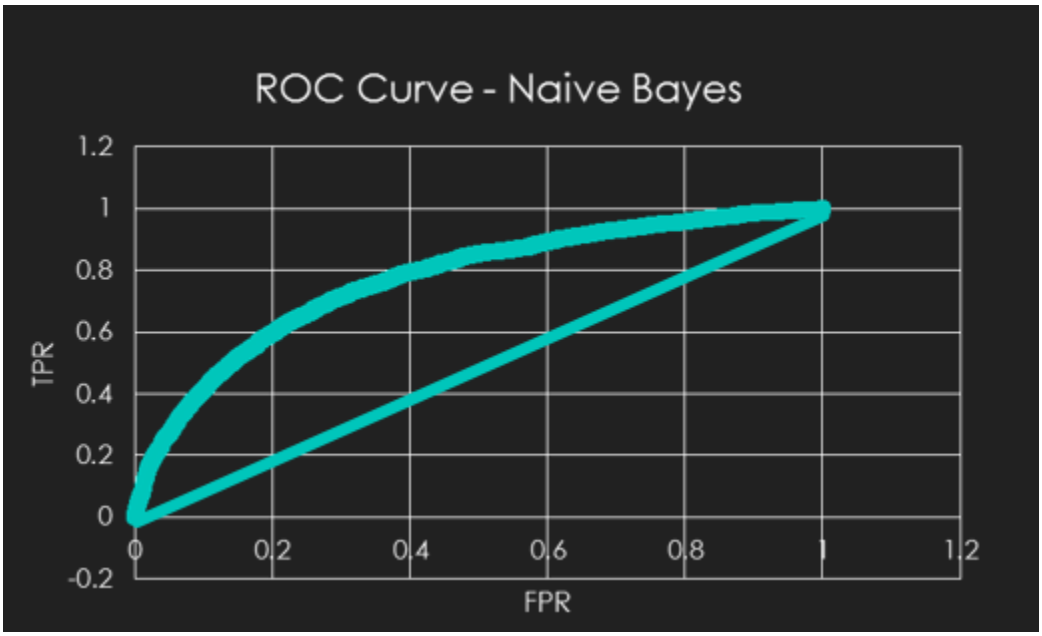
Naive Bayes

The multiclass problem was modeled using Naive Bayes with all the features in training as the input for the model. The model was trained using a training set that contained 50% of the total training set and validated against a 50% hold out set. To tackle class imbalance down sampling of the "functional needs repair" class was done at 50%. The F1 scores was obtained by choosing a threshold by filtering the false positive rate to be <0.1. This was done in order to minimize the cost associated with error in predictions associated with the positive class – functional needs repair. The Naive Bayes gave us a low f1 score within the holdout set, as reported below. We have used other models to improve the f1 score.

Figure 11

Model	f1 Score	Overall Accuracy
Naïve Bayes	**0.301**	**62%**

Figure 12



SVM

We chose to run Support Vector Machines on the training set to see how they performed. We used R to build the SVR model using the e0171 library. We first used the linear kernel to check the performance of the model and then used the RBF (with modified slack penalty) to check if a non-linear kernel would perform better on the dataset. Given that the

dataset has more than 40 features, which are almost all categorical, we expected that the non-linear model would performed better, but the RBF SVM did a better job.

Furthermore, while building the model, it was necessary to identify the right set of features that could be inputed to train the model. We found that some of the features, including lga, extraction_group, date_recorded, management_group, extraction_type_group, quality_group, and quantity_group were left out, as these were fields were representative of the observations at a higher level. The more granular variables, such as quantity, quality, management, extraction_type, etc. were more informative.

We created two models with the different kernels (linear and RBF). A hold out set of 50% was created from the given training data to validate the accuracy and f1 score of the models that were built. In order to tackle class imbalance, a downsampled dataset containing 50% “functional needs repair” was used to build the models.

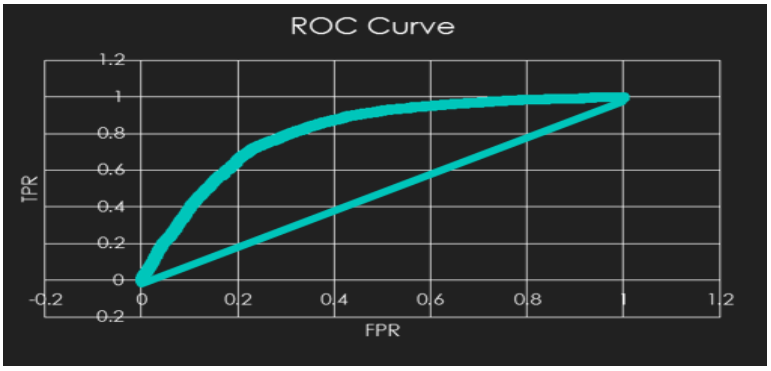
For the linear SVM**,** we used all the features selected from the above feature selection criterion including the external variables that were added to the training data. The model was run by setting the probability parameter in the model to be TRUE. This is required because we would need to calculate the probability of predictions of the test data against each of the three classes so that we could classify them as either “functional”, “functional needs repair” or “non-functional”.

To run the radial basis function as the kernel for SVM all the features selected from the above feature selection criterion including external variables were added to the training data. The model was iterated to find the different values for the slack penalty that provided the highest F1 score, as the primary goal was to maximize this score on the models. The best model was identified to have a cost (slack penalty) of 10 for a gamma of 1/39. The results for the F1 score in the holdout set are presented as below (Figure 13) for the different slack penalties. Figure 14 shows the ROC curve for RBF kernel with c=10. We did not validate these scores in a separate test set because our other models out-competed the SVM.

Figure 13

Model	f1 Score	Overall Accuracy
Linear Model	**0.2939**	**64%**
RBF (c=1)	0.3036	65%
RBF (c=10)	**0.3124**	**68%**
RBF (c=100)	0.3117	68%
RBF (c=1000)	0.3122	67%

Figure 14



Random Forest

As we had a lot of categorical variables, we used the ensemble method, random forest, to predict pump functionality. We had two goals for our project. From a business point of view, we wanted to have high precision on functional needs repair to make sure we were sending repairmen to pumps that actually needed repair, and we wanted to have high recall so that a higher proportion of pumps that need repair actually get repaired. Hence we used the F1 score to combine our need for precision and recall on functional needs repair. Secondly, because we were participating in a live data science competition in driven data which used accuracy as metric, we wanted to predict each class accurately. Hence, we trained the random forest and GBM models based on accuracy. Then we used the model with optimal parameters to find the threshold for optimal f1 score for the positive class (functional needs repair) from ROC curve. Different down sample sizes were used to train the model as well.

Below are the steps we took to train the model.

- **Feature selection/Processing:**

i. Removed group version of various columns (like 'quality_group' for quality, 'quantity group' for quantity, 'management_group' for management), as discussed in the SVM section

ii. Converted all the categorical variables into dummy variables

iii. Scaled the whole dataset

iv. Tried PCA(normal & Kernel(to capture non linearity) to reduce the number of columns in dummy data set

- **Parameter tuning:**

i. Number of trees: Varied trees from 100 to 500 to find best hold-out F1 score and accuracy (See table below).

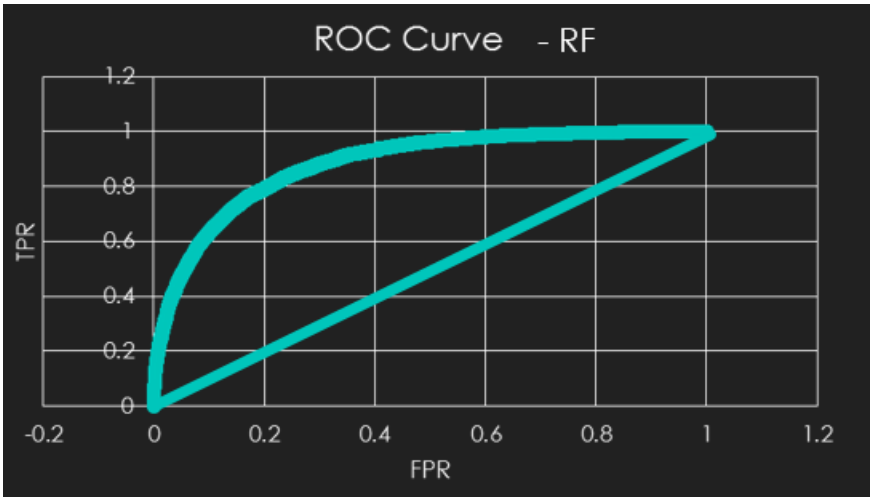
Number of trees	F1 Score	Accuracy (%)
100	0.385	79.3
200	**0.395**	**79.5**
300	0.388	79.3
400	0.398	79.4
500	0.394	79.3

ii. Number of random parameters: Tried using sqrt(max_parameters), log(max_parameters), and max_parameters as the number of random parameters in each tree. The square root number of parameters gave us the highest f1 score in the hold-out set.

Parameters	F1 Score	Accuracy (%)
Auto	**0.387**	**79.6**
Log	0.389	79.1
None	0.397	79.2

iii. Imbalanced data set: Created a train and holdout data set from the driven data train data. Tried different down-sampling percentages and 10% functional needs repair resulted in the highest f1 score in the hold-out set.

iv. ROC curve threshold: We tried to find the threshold that maximizes the F1 score based on graph between true positive rate and false positive rate



v. Scalibility: R and Python were not very scalable when we tried to fit model on the full data set. To counter that we tried two things. Firstly we used PCA to do feature extraction to reduce the number of dummy variables by selecting the most important principal components. Secondly we tried H2o–random forest. It's is an in-memory prediction engine which allows large data set to be used in real time without the need of sampling. We came to know about this through the forum discussions in the driven data community. It's highly scalable and we built random forest model on the entire training data using h2o

15

Based on the above parameters, we tried three different version of random forest on the hold-out data set and test data set (Driven Data). For hold-out data set, we have reported both F1 score and Accuracy. However, for test data set(Driven Data),we were not able to report F1 score as the competition was evaluating only accuracy

	Holdout Data Set		**Test(Driven data)(Trained on the full training set)**
Model	**F1 score**	**Accuracy (%)**	**Accuracy (%)**
Random Forest	0.39	79.2	81.15
Random Forest PCA	0.38	75.8	-
H2o Random forest	**0.392**	**81.7**	**81.45**

Gradient Boosting Machines

- **Feature selection/Processing:**
  - **i.** Removed group version of various columns(like 'quality_group' for quality, 'quantity group' for quantity, 'management_group' for management
  - **ii.** Converted all the categorical variables into dummy variables
- **Parameters tuning:**
  - **i.** Imbalanced data set: Created a train and holdout data set from the driven data train data. Tried different down sampling percentages to account for class imbalance
  - **ii.** Number of trees: Varied trees from 100 to 500 to find best F1 score/Accuracy

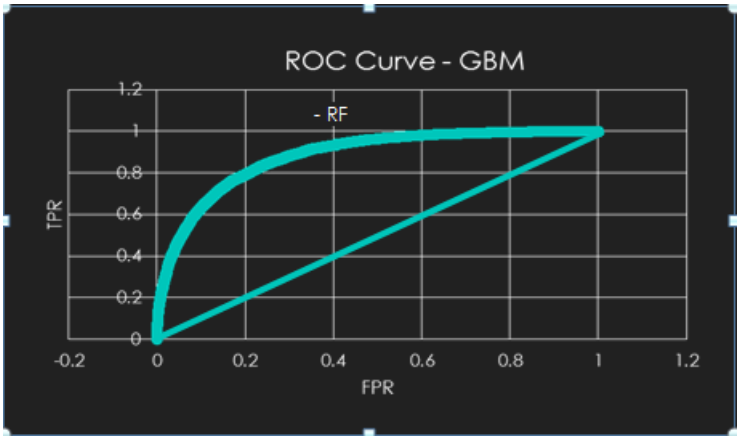
Number of trees	F1 Score	Accuracy
100	0.359	77.4
200	**0.364**	**77.79**
300	0.361	77.5
400	0.3608	77.49
500	0.3605	77.38

  - **iii.** **Depth of trees:** Varied the depth of trees from 5 to 25 to find the best score

Max depth	F1 Score	Accuracy
5	0.361	77.8
10	0.379	77.3
15	**0.368**	**77.89**
20	0.4006	78.8
25	0.3875	78.4

  - **iv.** **ROC Curve Threshold:** For the positive class tried to find the threshold that maximizes the F1 score based on graph between true positive rate and false positive rate

Figure 16



Deep Learning- Multi layer Perceptron

We trained the multi-layer perceptron to find the optimal number of nodes and epochs

- **Feature selection/Processing:**
  - **i.** Removed group version of various columns(like 'quality_group' for quality, 'quantity group' for quantity, 'management_group' for management
  - **ii.** Converted all the categorical variables into dummy variables
- **Parameters tuning:**
  - **i.** Imbalanced data set: Created a train and holdout data set from the driven data train data. Tried different down sampling percentages to account for class imbalance
  - **ii.** Number of nodes in hidden layer- Varied trees from 10 to 30 to find best F1 score /Accuracy

Nodes	Accuracy
10	73.5
20	75.9
30	76.1

  - **iii.** Epochs: Varied the depth of trees from 20 to 80 to find the best F1 score/Accuracy

Epochs	Accuracy
20	76.4
30	76.6
40	76.3

50	75.9
60	76.2
70	76.1
80	75.8

Multi-Level Model

Given that the Tanzanian water pump data included hierarchical geographic information in the region, district, divisions, and wards variables, we hypothesized that a multi-level model would glean information that other models could not. With a multi-level model, variables that differ from region to region can be modeled using individual level data. The individual-level data, or "random effect" is balanced out by a global "fixed effect." In this way, the model tries to prevent overfitting.

The first step to incorporating our more granulated, geographical information is to include intercepts for a selection of variables conditioned on each region, so that the resulting intercepts reflect both the region-level random effect balanced by the global-level fixed effect. Using the selected ranked features from lasso as a guide, we started incorporating intercepts for different categories using the lmer function from the r package lme4. What ensued was a series of complications, which we were not able to fully resolve.

While correlated inputs and appropriate variable selection was a concern, the first task was to solve our convergence issue. Many of the levels from our important categories were causing the model to diverge, despite being free of NA values and despite containing both ones and zeros in the training data. In order to resolve this, we tried every individual dummy level for the extraction type variable and found the ones that allowed the model to run appropriately. However, many combinations of these "safe" variables would still cause the model to diverge when ran collectively in the model. We were unable to resolve this issue, but we were able to use some of the variables that worked to achieve a decent f1 score of 0.36 in the test set. This was trained using downsampling to obtain 33% of the positive class. We presume that this model still has a lot of potential given that such a small percentage of our information was able to obtain this fairly decent f1 score.

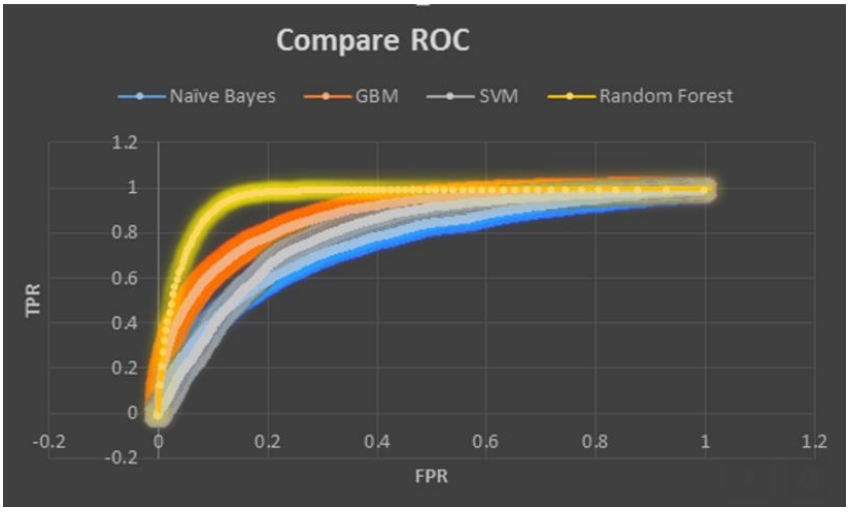
Our response to this failed covergence was to use pca as inputs as well as the linear mixed effects model function from the statsmodels package in python. We reasoned that if there was a problem with either the package or certain dummy variables, a new package with pca as input would likely solve either issue. Using pca would also help reduce the feature set through feature extraction while simultaneously providing uncorrelated inputs. We trained the model on the first principal component conditioned on region, and the model ran successfully, but it produced a warning that the maximum likelihood estimation was on the border of the convergence boundary. Given that we developped this most recent solution late in the process, we did not have time to work out this issue. Had this model worked, we would then consider the effectiveness of using pca for categorical variables. Principal Component Analysis will generate distances between different dummy variables in large dimensional spaces, but whether these distances reflect reality depends on how different categories relate to one another. Principal component analysis is usually not used for datasets that have a large number of categorical variables.

Final Results:

We found the best model to be random forest with respect to accuracy (81.7%) and hold-out F1 Score (0.392). The model gave an overall accuracy of 81.45% on the test set provided by Driven Data (using the H20 Random Forest function in R). The competition is currently live on the Driven Data website, and the accuracy we achieved ranks in the top 50 (out of 1,100 teams competing).

In general, we found that the models using trees outperformed the other models that we tested. Figure 17 shows the ROC Curve comparisons among several of the models that we tested. The model with the highest AUC is the random forest.

Figure 17



Model	**Accuracy**	**F1 Score**
Logistic Regression	71%	0.36
MLM	-	0.36
Naïve Bayes	62%	0.3
SVM	68%	0.31
GBM	78%	0.38
Random Forest	**81.45%**	**0.39**

Lessons Learned:

·Below summarizes some of the lessons we learned throughout the project. We discovered new tools in R and Python, and identified challenges with certain models we tested.

- **Spatial Maps:** We tried to use the leaflet package in R to make spatial maps using D3.js functionality. We found the package to be very complex and cumbersome for implementation. Doing further research, we discovered a tool via an online web application that uses D3.js and produces spatial maps based on data provided. This tool provided extra functionality that we were not able to produce via the leaflet package

due to our limited knowledge of spatial maps. Using the online web tool, we were successfully able to create effective spatial maps with our data.

- **SQLDF:** Our data required a heavy amount of preprocessing and data munching. This was taking us a lot of time to implement using the functions we were familiar with in R. We discovered the library, sqldf, which provides a SQL interface in R. Since we were familiar with writing SQL code, this made it much easier and more efficient for us to clean and manipulate the dataset.
- **Imbalanced Data:** We first implemented downsampling on the data, and then used business knowledge as cost functions (to reduce false positive rates) to find the optimal threshold from the ROC curve. We found that implementing cost functions using the performance function available in the ROCR library did not significantly impact the ROC curves. When we started working on this dataset, we were focusing only on the accuracy score (since that's what the competition on Driven Data was measuring). After looking into our data in more detail and realizing the imbalance present, we focused our efforts in looking at the F1 Score as a measuring tool.
- **Multi-level Models:** The lme4 Multi-level model package did not work well with our categorical variables. This could be the result of using a sparse data set of dummy variables as inputs.
- **New Packages:**
    - We discovered the H20 package that performed well on the Random Forest model

Code:

See https://github.com/julianghadially/Predictive-Modeling/tree/master/Tanzania%20Project for the code we developed for modeling and processing of the data. Data sets can be found in on this github page inside the data folder.

References:

1. How Tanzania Failed to Fix its Water Access Problem: http://www.humanosphere.org/world-politics/2014/12/tanzania-failed-fix-water-access-problem/
2. Driven Data: http://www.drivendata.org/
3. Tanzania Population: http://www.citypopulation.de/php/tanzania-admin.php
4. Tanzania Open Dataset: http://opendata.go.tz/dash
5. Water Point Mapping: http://wpm.maji.go.tz/
6. Cartodb.com:- https://cartodb.com/