Pump it Up: Data Mining the Water Table

Jiawei Ma | Renee Sikes | Gina Smith | Stephen Wooley

Submitted: Nov. 26, 2018

STAT 6115 / DSBA 6115

Statistical Learning with Big Data

Dr. Jun Song

University of North Carolina at Charlotte

Table of Contents

1. Introduction

1.1 Abstract

This project aims to use statistical and machine learning methods to accurately classify the operating status of a water access point in the East African country of Tanzania into one of three classes using set of predictor variables. The data and design of the project originates from a public machine learning competition hosted by DrivenData [1]. At the time of writing this paper, the team ranks first in the competition with a bagged model of four different machine learning algorithms: random forest, gradient boosting, extreme gradient boosting, and a neural network.

1.2 Problem Description

The purpose of the project is to accurately predict the operating condition of a waterpoint, located throughout the country of Tanzania into one of three possible operating conditions; therefore, the project is a multi-class classification problem. The three possible classes are: functional, functional needs repair, and non functional. This problem is a part of the "Pump it Up: Data Mining the Water Table" competition hosted by DrivenData.

2. Explanation of Data

2.1 Data Description

Data for this project was recorded and collected by Taarifa [2] and the Tanzanian Ministry of Water [3] and provided to DrivenData for the purposes of the competition.. The dataset can be found at: https://www.drivendata.org/competitions/7/pump-it-up-data-mining-the-water-table/page/23/

The data is collected from various water points throughout the country by the Tanzania Ministry of Water. The competition provides three data files: a training set of the independent variables, a training set of the target labels, and a test set of the independent variables. The training set contains 59,400 rows of data and 40 variables including one identifier, eight numeric variables and 31 categorical variables. The test set includes 14,850 rows of data. There are three possible target variables: functional, functional needs repair, or non-functional. The data types of the features vary from numeric data, Boolean, string, and geolocation.

2.2 Exploratory Data Analysis

The first step in the project was to understand the data provided. Initially, this was accomplished through Exploratory Data Analysis (EDA) which uses visualization techniques to understand the underlying structure of the data, such as frequency of variables or distribution of the data. Visual analysis of the data also aids in identification of data quality problems which may need to be considered and corrected such as missing values, incorrectly recorded values or outliers.

The team also used a statistical package in R, called boruta [4], which is a feature selection method that calculates the importance of the input variables in predicting the target variable. It works by holding one predictive variable steady while randomly shuffling the values of the other remaining variables and then calculates the effect of the steady variable on the target. This process is performed iteratively to progressively eliminate features which are not determined to be influential to the target variable. The results from the boruta package on the waterpoint dataset is shown in figure 1 below.

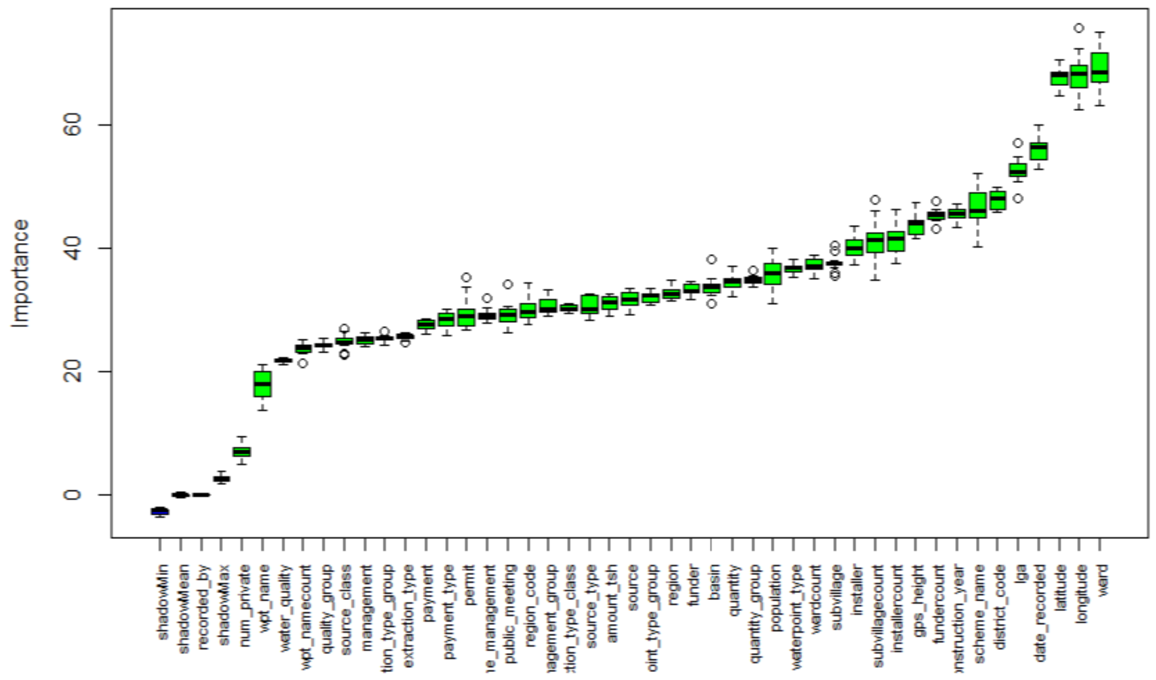


Figure 1: Boruta Importance Plot

2.3 Data Preparation

Prior to beginning analysis, the team tried various methods to clean and prepare the data. Two variables were dropped from the analysis variable set. The variable 'recorded_by' was dropped because it only contained one value and therefore did not hold any predictive information. The 'id' variable was also dropped as an input because it is an identifier and not a predictive variable.

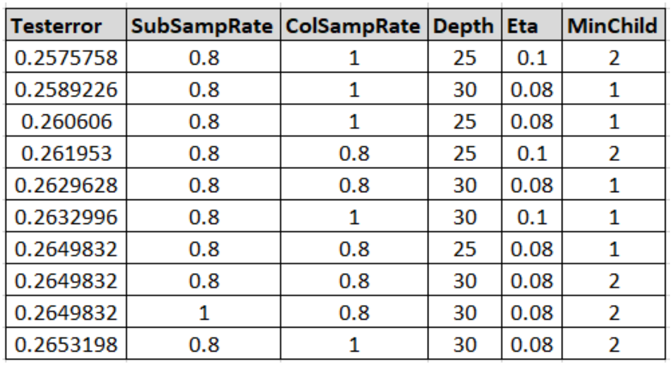
As previously mentioned, the data set included a high number of categorical variables, including four variables with a very high number of unique categories. The team tried various methods for reducing these categories including binning the variables by calculating the 25th, 50th, and 75th percent quartiles and then dropping the original variables. The categorical variables also had to be converted into numeric representations. The team ran models with both the label encoded schema and dummy encoded schema. The final prediction models did not use the binning technique on the categories and did use dummy encoding.

The exploratory analysis also indicated a high number of missing values in some variables. Again, several methods were tested including replacing missing numeric values with the mean and categorical values with the most frequent variable and using a random forest method to impute the missing values. In the final models, the team simply replaced the missing values with a value of "Unknown", which was then used as category during the dummy encoding.

2.4 Tools and Techniques

The team used a variety of tools and techniques during the project. For designing the models, the team used a combination of R language using RStudio or Python using Jupyter notebooks. We also used an open-source AI tool called H20 [5], which provides a locally hosted cluster method for parallel processing of data for faster training of models. The H20 platform will also automatically handle some pre-processing steps, such as dummy encoding of the categorical variables.

Once the initial models were trained the team also worked to improve performance by using 5 fold cross-validation to confirm predictive accuracy. We also performed hyperparameter tuning on our best performing models to further improve accuracy by performing a systematic grid search over possible combinations of the tuning parameters. Figure 2 below show the partial results from performing the grid search on the Extreme Gradient Boosting (XGBoost) model. For example, when setting the SubSampleRate to 0.8, the ColSampRate to 1, the depth to 25, the Eta to 0.1 and the MinChild parameter to 2, the estimated test error was calculated as 0.2575758. The purpose of this methodology is to try different combinations and then select the combination of parameters that produces the lowest test error rate.



Testerror	SubSampRate	ColSampRate	Depth	Eta	MinChild
0.2575758	0.8	1	25	0.1	2
0.2589226	0.8	1	30	0.08	1
0.260606	0.8	1	25	0.08	1
0.261953	0.8	0.8	25	0.1	2
0.2629628	0.8	0.8	30	0.08	1
0.2632996	0.8	1	30	0.1	1
0.2649832	0.8	0.8	25	0.08	1
0.2649832	0.8	0.8	30	0.08	2
0.2649832	1	0.8	30	0.08	2
0.2653198	0.8	1	30	0.08	2

Figure 2: Sample Grid Search Parameters

3. Methods

Since the goal of the project is to correctly predict the target variable as belonging to one of three possible classes, we knew we would be exploring various classification methods. We selected several classification techniques to compare. The models will be assessed by the misclassification rate by cross validation to choose the best models. The final accuracy score is the score we obtain from the competition website. The models the team tested are: Logistic Regression, Linear Discriminant Analysis, Support Vector Machines, Adaptive Boosting, Gradient Boosting Machine, Random Forest, Extreme Gradient Boosting, and Neural Network. The following graph in figure 3 shows the overall training and test accuracies for each method. Each method and its results are described in more detail in the following sections.

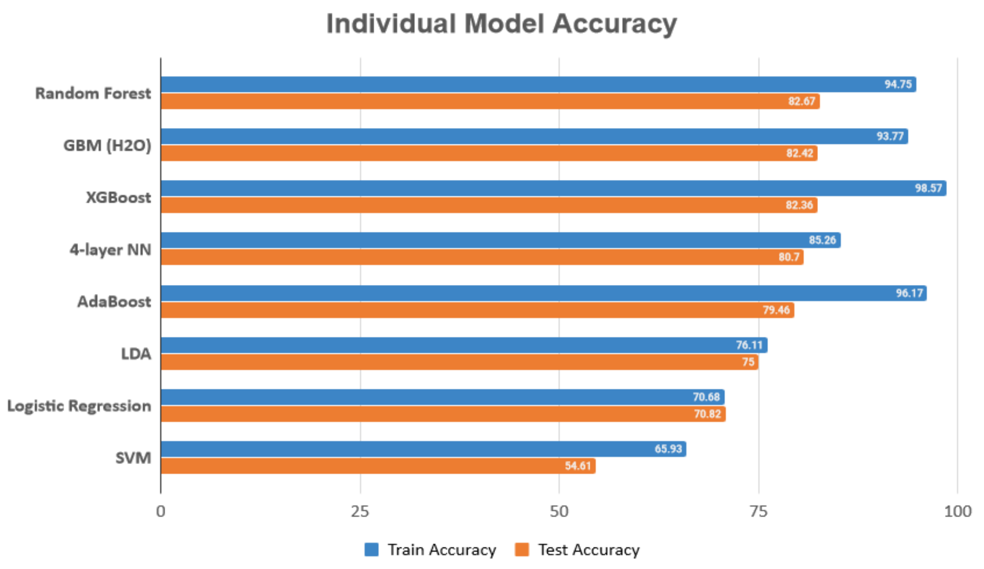


Figure 3: Chart of Model Test and Train Accuracy

3.1 Logistic Regression

Logistic regression is one of the simplest classification algorithms and seeks to classify a target variable, given a set of inputs, by calculating the probability, or log odds, that the target belongs to a particular class. The observation is assigned to the class with the highest probability. We chose to test this method because it is a simple classification method and we felt it would provide a good baseline for understanding how linearly separated our data points were. Our multinomial logistic regression model was built with the H20 package and produced an accuracy of 70.82% on the test set.

3.2 Support Vector Machine (SVM)

Support Vector Machines (SVM) are another method for classifying data that use marginal observations, rather than all observations in the data set, to classify a single observation. The goal of this algorithm is to find linear boundaries that maximizes the margin between groups of different observations after projecting the observations into a higher dimensional space. For this project, we used the Python package scikit learn [7] to generate the model, using the radial basis function (rbf) kernel for the projection. This model did not perform well on the test set, producing and accuracy of only 54.61, which is only a slightly better result than random guessing. We believe this model was not successful for the data set due to the imbalance of classes in the target variable. In other words, since the frequency of one of the classes was so low compared to the other classes, the model was never able to predict that class.

3.3 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a classification method that tries to predict the target by producing a linear combination of the predictive variables and calculates conditional probability density functions to predict probability that the observation belongs to each class. This model produced using the MASS package in R showed some predictive power on the data set by accurately classifying 75% of the test observations. We believe this model was not as successful as some of the others because this algorithm has several underlying assumptions which do not fit our data. For example, the algorithm assumes normality of the variables, which we can tell is not the case for all variables based on the plots generated during the EDA phase of the project. Multicollinearity in the data will also greatly reduce the predictive power of this technique; the dataset contains several hierarchical geographic labels which are very highly correlated, which could result in the lower accuracy compared to other models we tested. Furthermore, the class imbalance may have had a negative effect for this method despite its ability to predict all three classes. One study found that when comparing the performance of LDA on ten different imbalanced datasets versus the balanced dataset using four resampling methods, random over-sampling or random under-sampling outperformed the original dataset for area under the ROC curve (AUC) in nine out of ten cases [11].

3.4 Adaptive Boosting

Adaptive Boosting (AdaBoost) works by applying boosting to a weak-learner base classifier. Boosting is a technique that builds a strong classifier out of a set of weak classifiers by iteratively adjusting the weights of the classifiers to progressively focus on the hardest to predict cases. For this project, we used the multi-class extension of AdaBoost developed by [8], which uses a new algorithm to treat the multi-class problem as a stagewise forward additive problem as opposed to breaking it down into multiple binary classification problems. We used this SAMME.R algorithm as implemented in the scikit-learn Python package. We used a Decision Tree as the weak-learner base classifier. This model correctly classified 79.46 percent of the test observations.

3.5 Gradient Boosting

Gradient boosting is a popular machine learning algorithm that can be used for regression or classification. It works by using a weak learner to make predictions and then creates an ensemble of these weak learners to create a powerful model. The gradient boosting machine model in the H2O package uses many tree models where each model assigns weights to observations misclassified by the previous tree. Our gbm model was built with 5,000 trees. Parameter tuning was initially done for the max depth of the trees which is how many splits are allowed in each of the 5,000 trees. A Cartesian search was done for depths up to 40 using our own train and test split. The optimal max depth was selected from the results. Additional parameter tuning was done using a random grid search for histogram type, sample rate, column sample rate, column sample rate per change level, number of bins for continuous variables, and number of bins for categorical variables. Our best GBM model produced 82.42% accuracy on the competition test data.

3.6 Extreme Gradient Boosting

Extreme gradient boosting (XGBoost) is another popular machine learning algorithm that is similar to the gradient boosting algorithm. However, it is known for its fast speed and accurate predictive power due to some of its notable features including regularization to help reduce overfitting and parallel processing to reduce overall training time. A multiclass classification XGBoost model was created for our dataset in R using the xgboost package. Optimization of the model was obtained through hyperparameter tuning and utilization of its build in cross validation function. The model accuracy received was 82.36%. Additionally, a feature importance plot was obtained, as shown in figure 4 below.

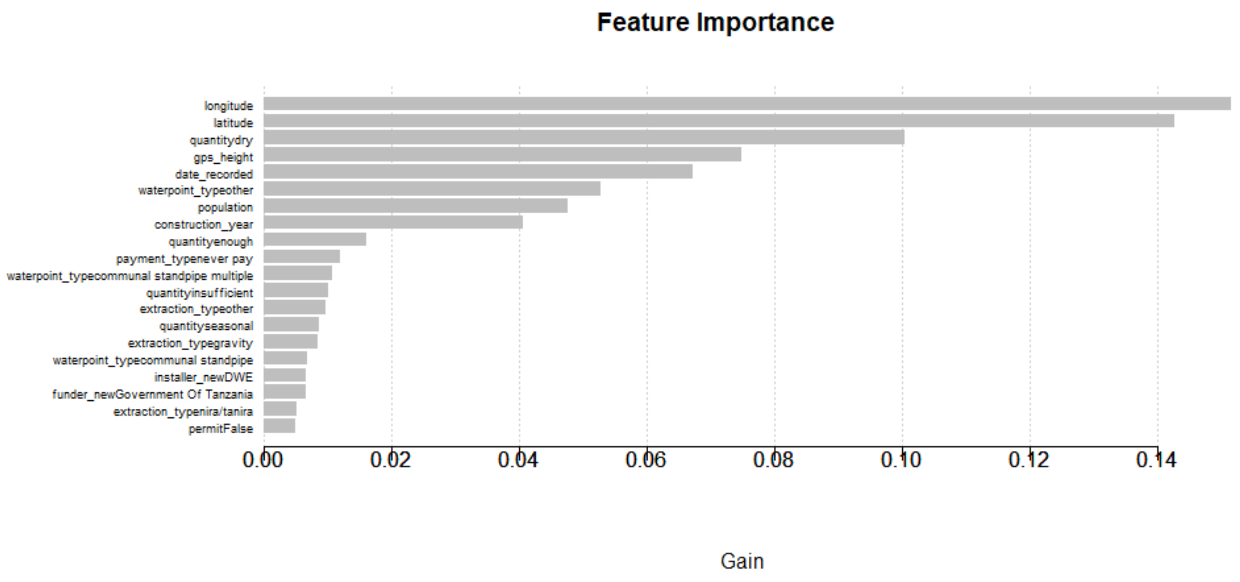


Figure 4: Feature Importance Plot from Extreme Gradient Boosting Model

3.7 Neural Network

A feed forward neural network is a type of machine learning algorithm with three main components. The input layer receives the data then passes it to the first hidden layer. Hidden layers consist of layered sets of nodes where every node on one hidden layer is connected to every node on the next hidden layer by a weight. The final hidden layer is connected to the output layer. This structure is advantageous as it allows the neural network to generalize by learning from the initial inputs and their relationships to infer unseen relationships on new data. We used the H20 package to construct and evaluate predictions for many neural networks. Two to five layers were tested with three and four layers producing the strongest results. Layers of decreasing size outperformed networks with standardized hidden layer size, and 400 nodes in the first hidden layer produced the strongest results. Another important parameter to test was the activation function. Rectifier, Tanh, and Maxout along with the versions allowing dropout were all tested, and Rectifier with dropout generally outperformed the rest. In addition, tuning was done for sample rate and the l1 and l2 regularization penalties. Our best performing neural network had a (400, 200, 100, 50) hidden layer and scored 80.7% on the test data.

3.5 Random Forest

Random Forest is a machine learning technique which aims to reduce variance in a dataset by generating multiple decision trees with different subsets of observations and variables and then combines the predictions of the individual trees into a single prediction. For this project we leveraged the H20 platform in R for building the random forest. At each iteration we selected 60 percent of the training samples and 80 percent of the available columns. The random forest was the best individual performing model, correctly predicting 82.67% of the test observations.

4. Analysis / Results

4.1 Final Approach

The single best performing model was the random forest model which accurately predicted 82.67% of the observations. However, to further improve the score, we chose to use a bagging method consisting of the top four performing models: random forest, gradient boosting, XGBoost, and the neural network. The tree-based models had high accuracy scores in the 82 percentiles while the neural network had an accuracy of 80.7%. Bagging is a statistical technique the combines the prediction results for multiple machine learning models and takes a "majority vote" from the predictions of each observation to determine the final prediction. In any cases where there may have been a tie in the voting, the random forest prediction was used to break the tie. This technique allowed us to improve the accuracy score to 82.86%.

5. Conclusion

For this particular dataset, the objective was quite clear: Predict the operating condition of a waterpoint in Tanzania. Therefore, our best model was the one that would provide the highest prediction accuracy. After testing multiple types of classification models the team determined the best methods for modeling the given data set were tree-based methods. Additional accuracy was afforded by boosting methods to further tune predictions. We believe tree-based methods work well for the dataset due to the high number of categorical variables.

The resulting model will ultimately assist Taarifa and the Tanzania Ministry of Water in improving the maintenance operations of the pumps. Next steps would involve determining the relationship of the pump status and the attributes through additional modeling using the important variables extracted from the random forest model or by narrowing the focus and concentrating on each region. This would produce an increase in the number of functional water pumps and drive improvement for the water conditions within Tanzania and similar countries.

6. References

[1] https://www.drivendata.org/competitions/7/pump-it-up-data-mining-the-water-table/
[2] http://taarifa.org/
[3] http://maji.go.tz/
[4] https://www.rdocumentation.org/packages/Boruta/versions/6.0.0/topics/Boruta

[6] https://www.h2o.ai/
[7] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[8] https://scikit-learn.org/stable/modules/feature_selection.html
[9]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html#sklearn.ensemble.GradientBoostingClassifier.fit
[10] Zhu, Ji &amp; Rosset, Saharon &amp; Zou, Hui &amp; Hastie, Trevor. "Multi-class AdaBoost." Statistics and its interface, 2, 2006, doi:10.4310/SII.2009.v2.n3.a8.
[11] Xie, J., & Qiu, Z. (2007). The effect of imbalanced data sets on LDA: A theoretical and empirical analysis. *Pattern Recognition, 40*, 557-562.

7. Appendix

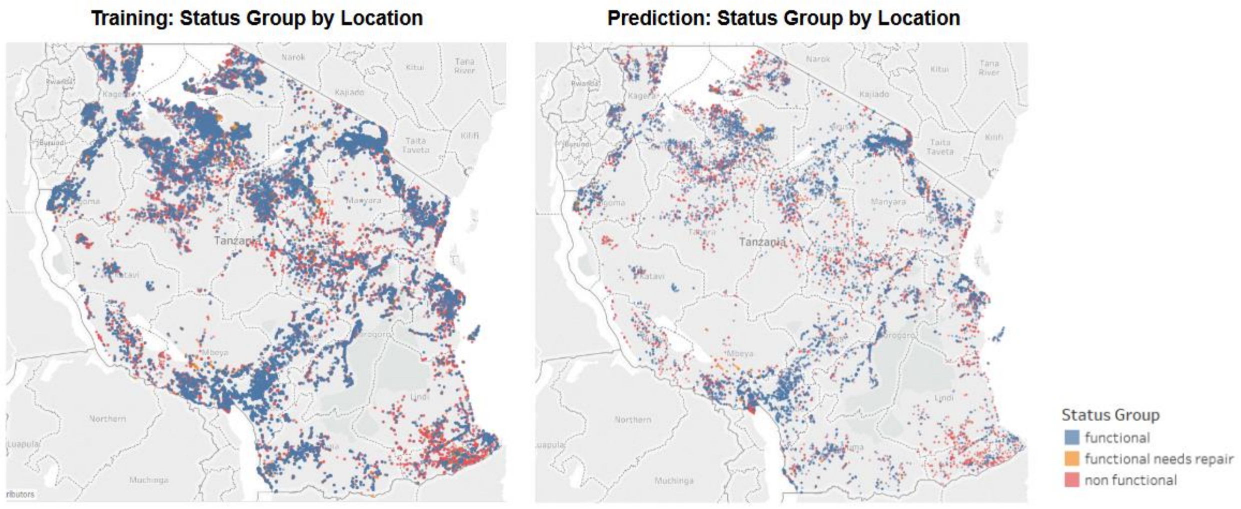

Training: Status Group by Location
Prediction: Status Group by Location
Tanzania
Status Group
functional
functional needs repair
non functional

Figure 5: Map of Waterpoints and Status from Training Data (left), Map of Waterpoints and Predicted Status from Test set (right)