Capstone 1: Data Mining the Water Table

Project Proposal

Problem:
Many rural communities around the world (especially in developing countries) rely on water pumps as their main source of water. This includes water for drinking, cooking, cleaning, growing crops, and more. Obviously if a community's water pump fails, it can be extremely disruptive to daily life.

So the question is, how can we predict which water pumps are faulty, and which are operational? Using data from the Tanzania Ministry of Water, we are going to make predictions on which water pumps in rural Tanzania are functional, functional but in need of repair, or non functional.

Client:
Delivering more accurate predictions about which pumps might fail can improve maintenance operations, making them more efficient and cost effective. The Tanzania Ministry of Water will be interested in the results of this project, but it may be applicable to any government agency or NGO around the world interested in making sure rural communities have access to clean potable water.

Data:
The data are available here:
https://www.drivendata.org/competitions/7/pump-it-up-data-mining-the-water-table/data/

This project is part of a data science competition on DrivenData.org, as such the data set is downloadable from their website. The data come from Taarifa, which is a platform for aggregating data from the Tanzania Ministry of Water for crowdsourced reporting of infrastructure related issues.

The data come as data-frames stored in several CSV files, one for the training set and the test set, etc. It is ready to be read in as a pandas dataframe, though may require some cleaning/wrangling before it will be ready for analysis.

Methods:
We will begin with some exploratory data analysis to get a better idea of what the data look like, find any patterns in the data, and help inform our analysis. We'll look for

missing values and outliers, and determine what to do with them. We might also look to feature engineering and dimensionality reduction. We'll probably want to scale numerical features, and check for highly correlated features.
This is a supervised learning problem; we'll be making predictions on a categorical variable with three possible outcomes, so it's also a classification problem.

We'll build our model first with more simple methods, starting with a decision tree, before trying other algorithms like random forests, etc.

Deliverables:
The deliverables for this project will be a final report, a presentation slide deck, and the jupyter notebook(s) containing all the code of the analysis.

Data Wrangling

For this project, the data come from DrivenData.org, a non-profit organization which hosts data science competitions that have a social impact. As such, the data are for the most part provided in a clean, tidy format, and require little data wrangling to be ready for analysis. However, the data wrangling steps taken are outlined below:

Cleaning steps

The training set is provided in two separate csv files, one which contained all 40 features, and another which has the outcome variable. We'll read the data in, and then merge the two data frames into one.

Several variables contain redundant information, such as "num_private" which has nothing but zeros, or "recorded_by" which contains the same value for every observation. Some variables have different names but the same information, like "quantity" and "quantity_group", as well as "payment" and "payment_type", so one of each of these columns will be removed. Other columns will be removed from the data as well for similar reasons.

Dealing with missing values

Some variables in this data set encode missing values as zeros. For example, the "construction_year" variable represents the year in which the given water pump was constructed. Any value of zero wouldn't make sense here, so clearly this must be a missing value. The same can be said for the "latitude" and "longitude" variables. Values

of zero would represent the equator and the prime meridian, respectively, neither of which crosses the country of Tanzania, so clearly these must represent missing values.

Other variables, however, may have values of zero which make sense. For example, the “gps_height” variable represents the altitude of the well. So a value of zero may represent a well at zero altitude, or in other words, at sea level. It wouldn’t make sense to treat such a value as missing, so for this variable it is left alone.

For other variables, “amount_tsh” (amount of water available at well) and “population”, it is unclear if the zero values represent missing data or not. So they will be left alone.

For the other variables, where it is clear zeros represent missing data, these will be replaced using the median values of the respective variables.

Handling outliers

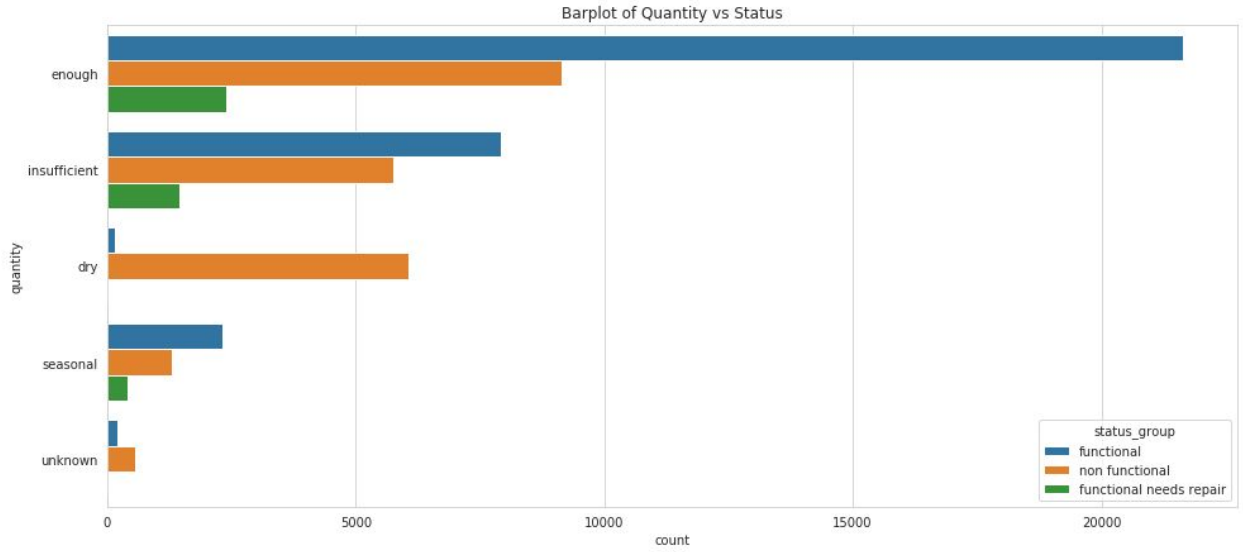
Most of the data are made up of categorical variables, so it is only necessary to look at the numerical variables, most of which contain few extreme values. The variables which do not contain outliers were “construction_year”, “latitude”, and “longitude”. The other variables may contain a handful of outliers, but it is unclear if these are mistakes in data collection, or just extreme values that are just very rare. There does not appear to be any obvious errors: for example “amount_tsh” value less than zero, which would indicate a negative amount of water in a given well, and would make no sense. So the extreme values are left alone for now.

Exploratory Data Analysis/Data Story

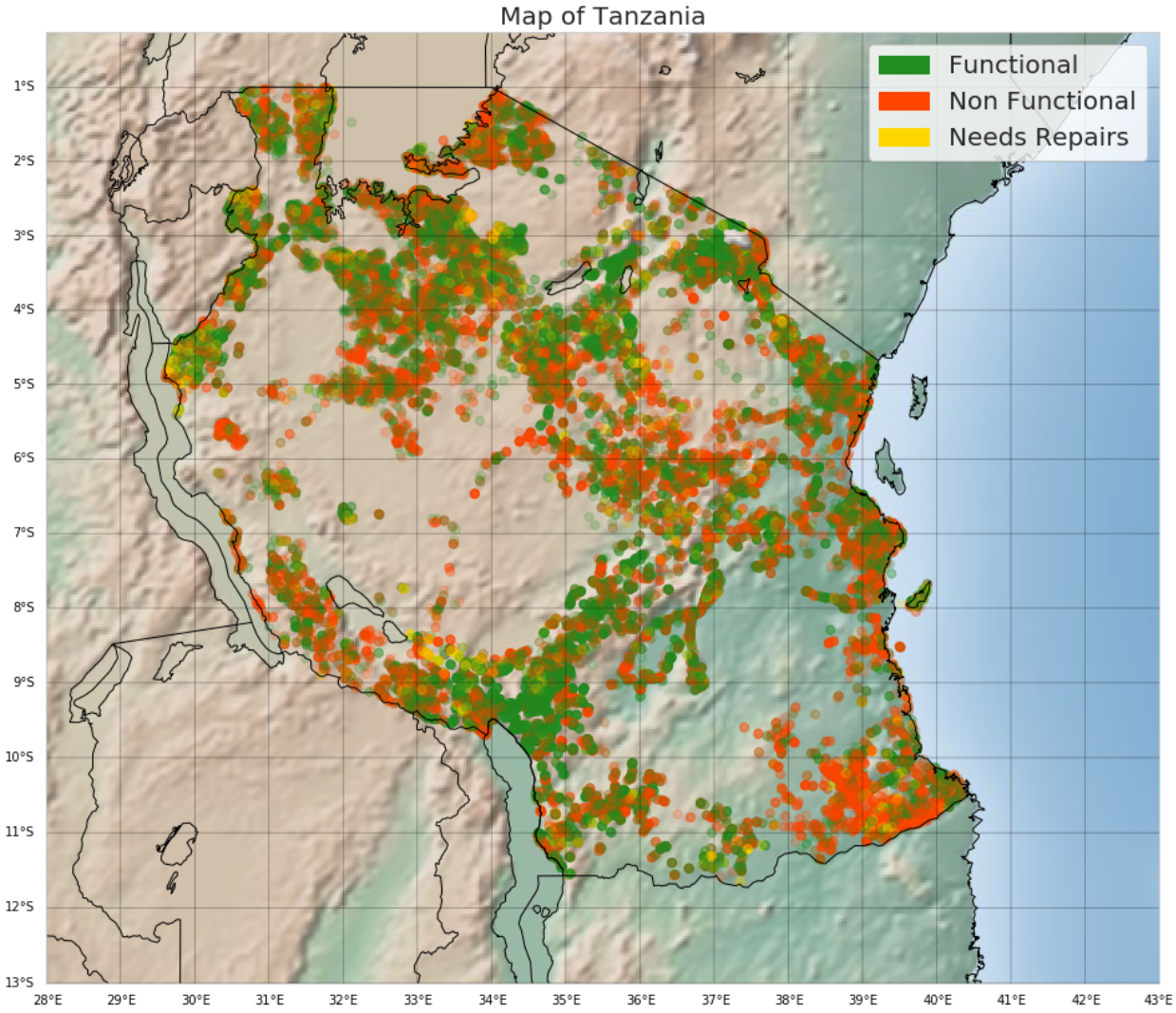
We’ll take a look at the data set and see if we can find some interesting insights. First we notice that the two variables amount_tsh and num_private are made up almost entirely of zero values, which might not be very useful in predicting status_group. Most likely we will want to remove these columns when building our model. Second, we’ll create some visuals to help us notice some trends.

We’ll look at some violin plots of construction_year grouped by status_group, and we can see that it seems wells which are functional are more likely to have been constructed more recently. The plot is given below.

We'll also look at several categorical variables. It appears basin, Region, extraction_type, payment, water_quality, quantity, source, and waterpoint_type might be important predictors for determining if a well is functional or not, or in need of repairs. For example, the plot shown below shows that wells which are 'dry' (from the quantity variable) are much more likely to be non functional.



Next we'll look at a map of Tanzania with the wells represented as points. The points are color coded to show which are functional, non functional, and in need of repairs.



It's unclear if there are any meaningful patterns in this map. We can clearly see at least a few patches where there appear to be more functional or non functional wells. However, it's possible this information is already captured in categorical variables like "region" and "basin" or other variables which correspond to geographic location.

The violin plots of latitude, longitude, and gps_height did not appear to reveal any significant patterns. We'll do some statistical inference and hypothesis testing to see if there are meaningful differences in group averages.

Statistical Inference

Based on the results of our exploratory data analysis, we are interested in doing a but more investigation into which variables might be important in predicting the outcome

variable. We first graphed the distributions of numerical variables such as construction_year, gps_height, latitude, and longitude and group them by status_group (functional, non function, needs repairs). Now we want to see if there are any significant differences across these groups.

Methods

For this we will be using Welch's T-test to compare the averages between two groups. Since we have three outcomes for status_group, we will do the tests pairwise to check if there is a significant difference between each group. We use the Welch's T-test because we want to compare and see if the means of two data samples differ, and we are not necessarily going to assume the variances are the same across the three groups. For each test, our null hypothesis will be that the means of the two groups do not differ from one another, and we will select alpha to be 0.01.

Hypothesis Tests

For the first set of tests, we'll look at the means of contruction_year. The mean for the functional group is 1999.94, the mean for non functional is 1992.40, and the mean for needs repairs is 1995.26. All the pairwise T-tests are significant in this first example, indicating that construction_year might be an important variable in predicting status_group. The averages also seem to make intuitive sense: it might be the case that wells constructed more recently are more likely to be functioning properly, while wells constructed further in the past would probably have a higher chance of breaking down and either needing repairs, or not functioning altogether.

For the second set of tests we'll look at the means of gps_height. The mean for functional is 740.13, non functional is 574.46, and needs repairs is 627.61. Once again all pairwise tests are significant, with non functional vs needs repairs being the least significant, at least in terms of having the highest p-value. This indicates gps_height might also be important in predicting status_group.

The third set of tests looks at longitude. The means are as follows; functional: 35.191149; non functional: 35.239450; needs repairs: 34.309867. Interestingly, the T-test comparing functional and non functional groups is not significant, with a p-value of 0.035, while the other two tests are significant. However, this might be an indication that longitude might not be as helpful for predicting status group, since there is relatively little difference between the two groups functional and non functional.

The final set of tests is for the latitude variable. All pairwise tests showed significance, with the means given as: functional: -5.704921; non functional: -5.810394; needs repairs: -5.162580.

It's worth noting that we have a very large sample size, roughly 60,000 total data observations. So any significant p-values we obtain might be a reflection of this large sample size, rather than a reflection of any practical significance in differences. Thus, based on the results we obtained, it might be the case that the construction_year and gps_height could be important variables in predicting status_group, but it less clear that latitude and longitude will be all that helpful, at least without further analysis.

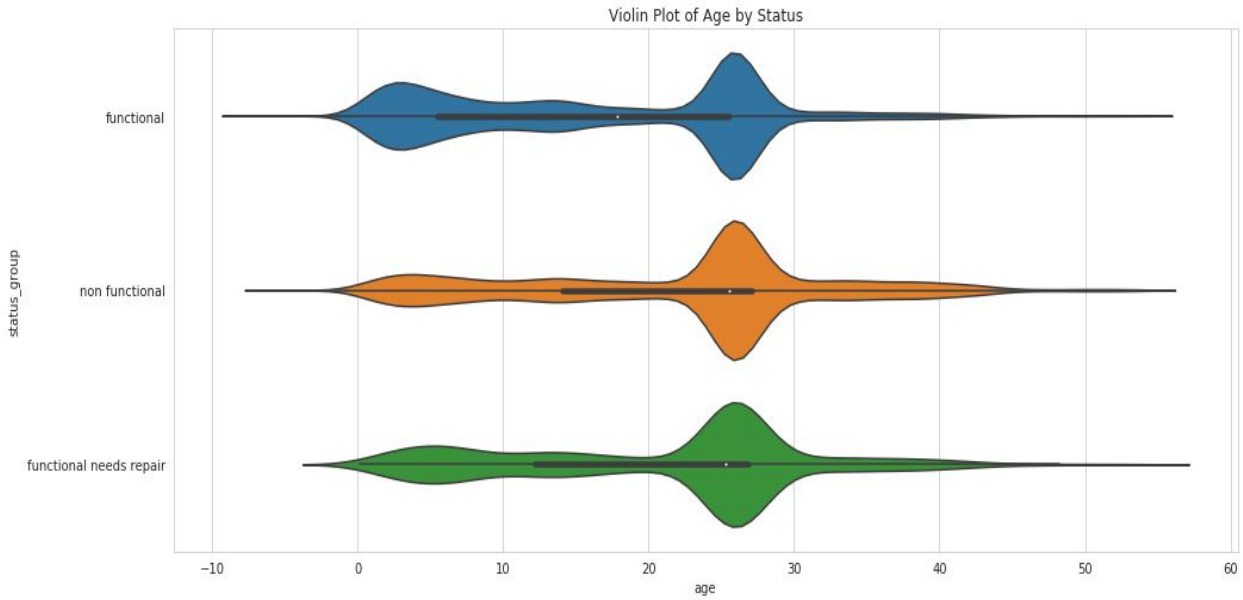
Machine Learning

Before we begin building and testing our machine learning model, we need to do a bit of extra work. First we're going to do some feature engineering, and then we'll need to reformat the data to get it ready to work with SciKitLearn.

Feature Engineering

We're going to create a new variable from the data, which hopefully will have new information that will help our model performance. We'll create a variable called 'age' which will represent the age of the well in years at the time the data observation was recorded. We'll calculate this number by taking the difference between the 'date_recorded' variable and the 'construction_year' variable. We first need to convert the variables to datetime objects, and then get the timedelta into the right frequency.

We'll make a violin plot of age by status, just as we did before, to get a better idea of what the data look like.



It looks like functional wells are more likely to be younger than non functional wells, or wells that need repairs. We also ran a hypothesis test as we did earlier, and discovered all p-values were significant. This might not mean much since as stated before we have so many observations, almost any difference might appear significant, but we'll see if this new feature may help our model performance in the end.

Next we remove unnecessary columns which contain redundant information, and those which have almost nothing but missing data. The final step is to convert all the categorical variables to numerical values, so that it will work with SciKitLearn.

Benchmark Model

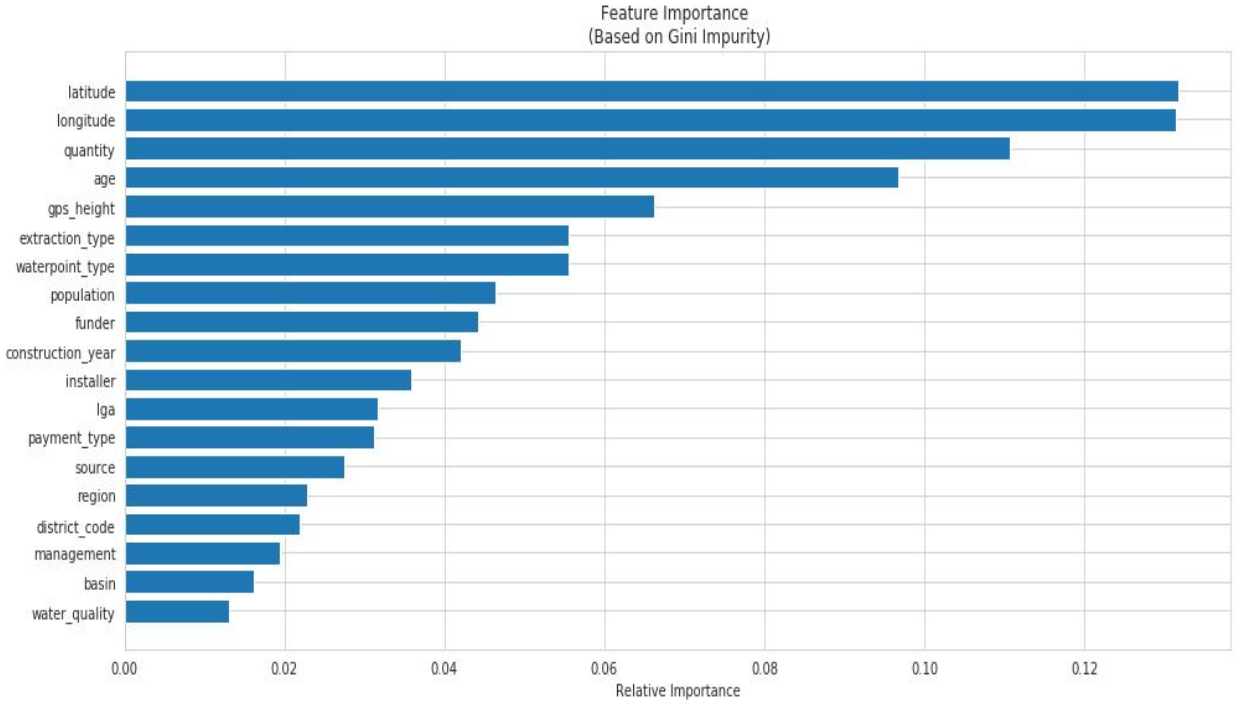
The first model we'll build will be a simple decision tree. This will give us a baseline from which we can compare our results. If we build a model which performs worse than this simple decision tree, then we may have a problem. We'll also build a simple random forest model using 250 trees, and compare the two.

The decision tree had an accuracy score of about 0.757, while the random forest model had a score of about 0.806. So already we've seen an improvement.

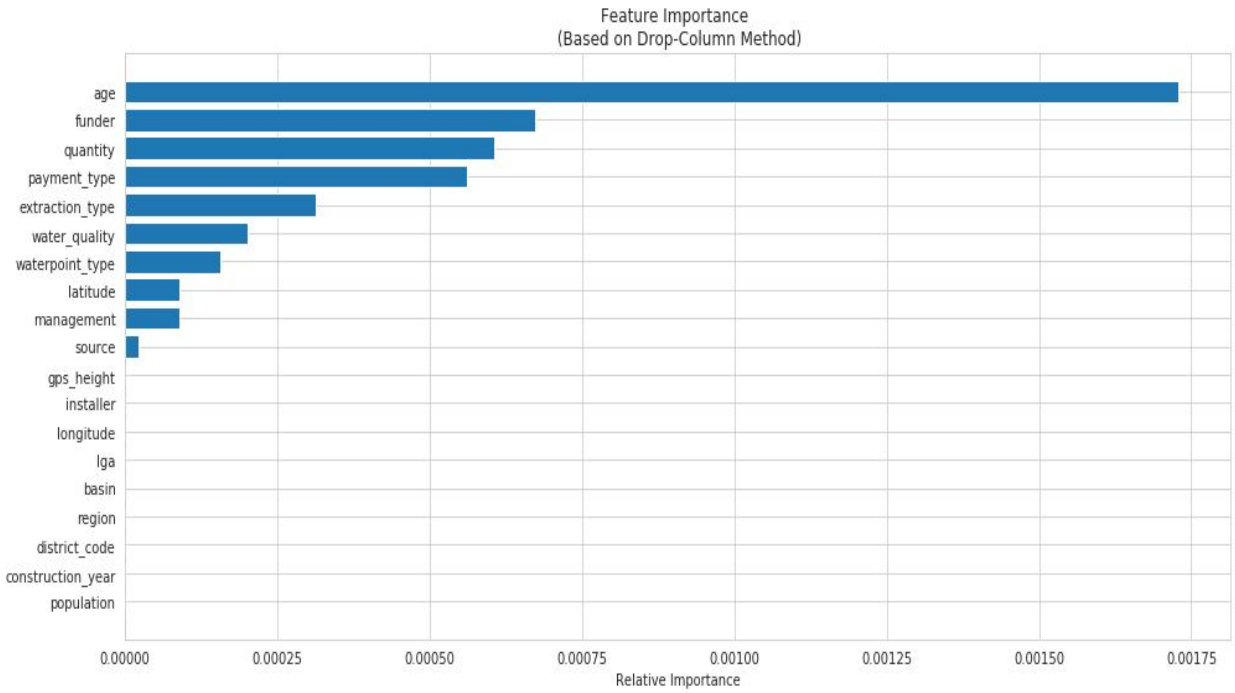
Feature importance

We want to avoid problems with over-fitting, so to do that, we'll analyze feature importance. This will help us see which variables are actually important for our model

performance. We'll use SciKitLearn's built in methods for analyzing feature importance, which use a method that takes Gini Impurity into account.

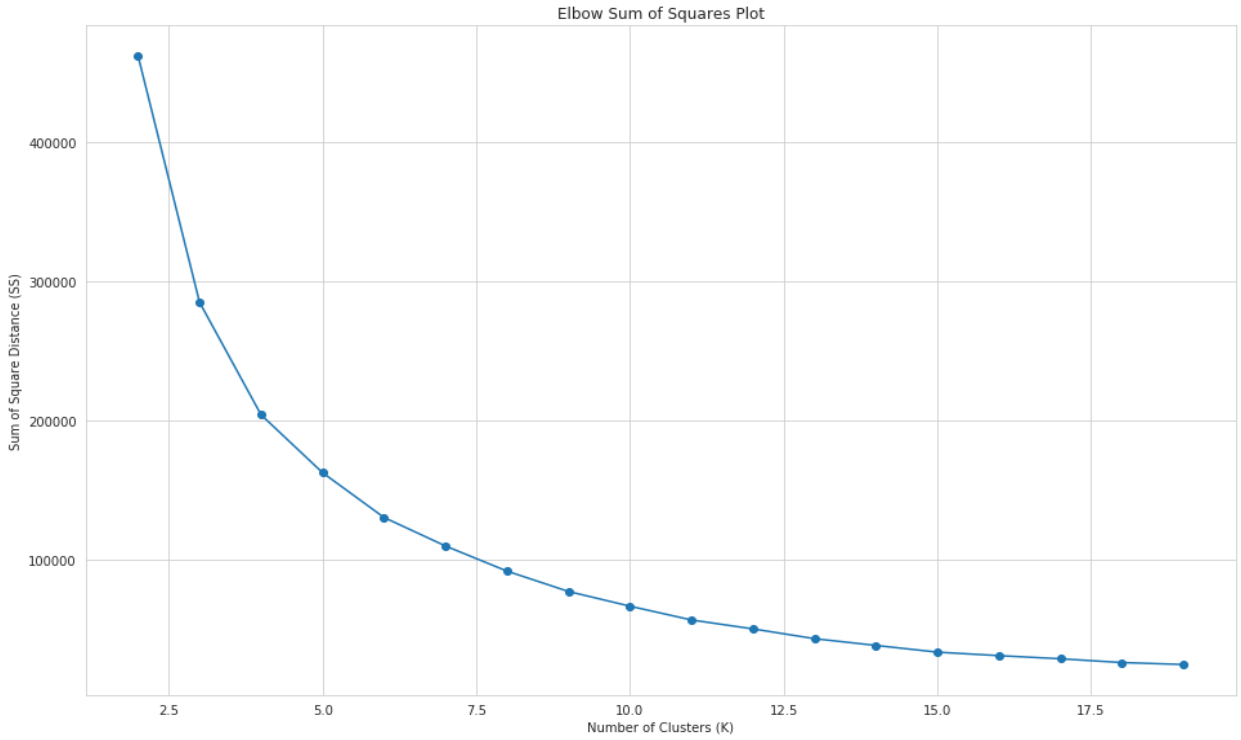


Interesting that this seems to indicate longitude and latitude are the two most important features, even though we saw earlier that when we did a hypothesis test for longitude, we did not get a significant p-value when comparing functional and non functional wells. So what happened? Apparently, this method can over emphasize the importance of continuous variables, as well as categorical variables with a high number of levels. So, to double check we'll use a second method for analyzing feature importance. We will do this by dropping each column one at a time, and comparing the difference in performance to the full model. This is computationally more costly, but should yield better results. We do this manually by creating a for loop over all the columns, and drops the column, builds a new model and checks the performance. We'll compare the difference to our benchmark score based on the model with all features, and plot the results.

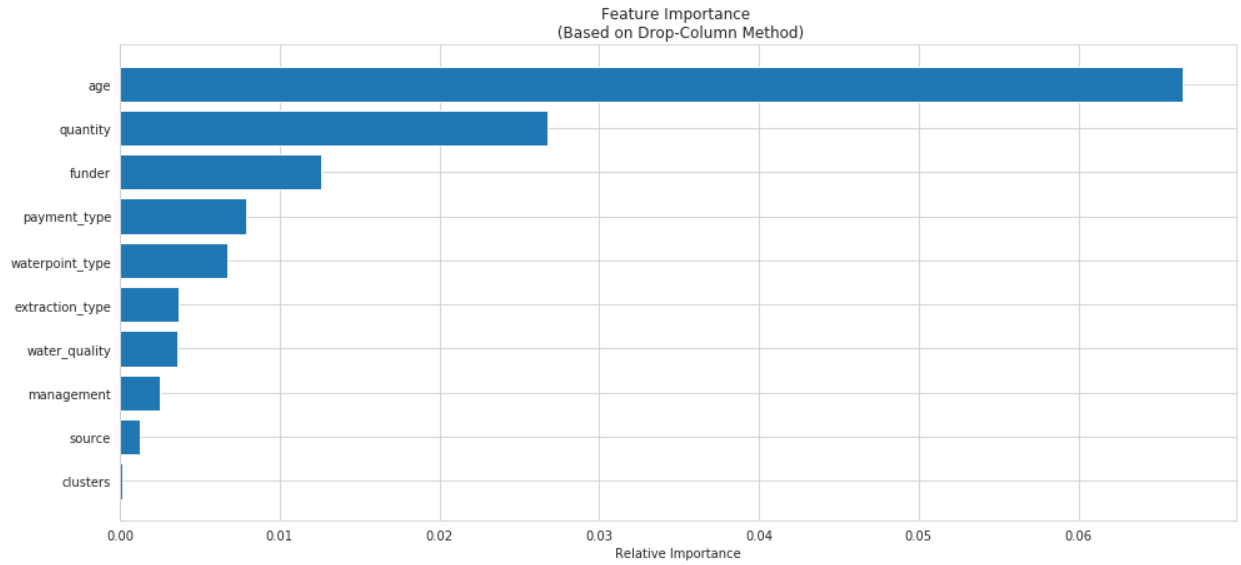


Now we've gotten completely different results. Longitude is no longer an important feature at all, along with several other variables. Also interesting, the age variable we created earlier, now appears to be the most important feature! So it's a good thing we took the time to do some extra feature engineering.

With that in mind, we'll attempt to do some extra feature engineering, and see if we can take the latitude and longitude data and transform it into a feature that might be useful. We'll try using KMeans clustering to transform the data by organizing it into clusters. We'll need to test out different values for k (the number of clusters) and compare the sum of squares using an "elbow" plot. Where the bend appears in the plot will indicate which value we might want to choose for k.



There appears to be a bend or “elbow” in the curve at k = 4, so that’s what we’ll choose for our algorithm. We’ll organize the data into these four clusters, and add the cluster assignment as a new feature in our data set. We’ll then drop the features that were determined to be unimportant, and take another look at feature importance.



These results show our feature engineering did not prove to be effective. The "clusters" feature we created based on latitude and longitude was almost completely unimportant

in our model. For this reason we will not be using this feature going forward, and we will use only the features found to be important for the next step.

Hyper-Parameter Tuning

The next step we'll want to take is to fine-tune some hyperparameters of our model. We'll be using a new data set with all the unimportant features (which we've just uncovered) dropped from the dataframe. We'll first train a model with the same parameters as before, to get an idea of how the model performs after dropping several columns. Then we'll run a grid search over several different parameter options, and compare results.

Final Results

The benchmark yields a score of 0.784, due to training on less data,while the model obtained by the grid search yields a score of about 0.801. So, parameter tuning definitely was an improvement, even though it is slightly worse than the original random forest model we ran. Hopefully we managed to avoid overfitting the data. This model should perform better on new data it's never seen before than the original model would.

Below is a table summarizing the results we've obtained. It has all the models trained, and the corresponding accuracy score, based on the validation set.

Model	**Accuracy**
Decision Tree	0.757
First Random Forest	0.806
RF with gps "clusters"	0.789
RF without unimportant features	0.784
RF with hyperparameter tuning	0.801

All that is left is to make a prediction using the final test set, convert the results to the proper format, and submit it for final scoring in the competition.