Capstone Project 2 Milestone Report 2

The question to answer is:

What is the operation status of a water point in Tanzania: functional, needs repair, non functional?

Slightly more than half of the population has access to clean water in Tanzania. The operation and maintenance costs are difficult to bear for local government authorities. Tanzania receives external support from several donor agencies.

The objective of this project is to predict the operation status of water points. This will help reducing operation and maintenance cost, while improving continuity of supply.

This is a DrivenData competition.
The dataset used comes from Taarifa and the Tanzanian Ministry of Water.

The dataset contains records of 59400 water points.
Each record has the following information about the water point:

- `amount_tsh` - Total static head (amount water available to waterpoint)
- `date_recorded` - The date the row was entered
- `funder` - Who funded the well
- `gps_height` - Altitude of the well
- `installer` - Organization that installed the well
- `longitude` - GPS coordinate
- `latitude` - GPS coordinate
- `wpt_name` - Name of the waterpoint if there is one

- `num_private` -
- `basin` - Geographic water basin
- `subvillage` - Geographic location
- `region` - Geographic location
- `region_code` - Geographic location (coded)
- `district_code` - Geographic location (coded)
- `lga` - Geographic location
- `ward` - Geographic location
- `population` - Population around the well
- `public_meeting` - True/False
- `recorded_by` - Group entering this row of data
- `scheme_management` - Who operates the waterpoint
- `scheme_name` - Who operates the waterpoint
- `permit` - If the waterpoint is permitted
- `construction_year` - Year the waterpoint was constructed
- `extraction_type` - The kind of extraction the waterpoint uses
- `extraction_type_group` - The kind of extraction the waterpoint uses
- `extraction_type_class` - The kind of extraction the waterpoint uses
- `management` - How the waterpoint is managed
- `management_group` - How the waterpoint is managed
- `payment` - What the water costs
- `payment_type` - What the water costs
- `water_quality` - The quality of the water
- `quality_group` - The quality of the water
- `quantity` - The quantity of water

- `quantity_group` - The quantity of water
- `source` - The source of the water
- `source_type` - The source of the water
- `source_class` - The source of the water
- `waterpoint_type` - The kind of waterpoint
- `waterpoint_type_group` - The kind of waterpoint

The training dataset is labelled in 3 categories: functional, functional needs repair, non functional.

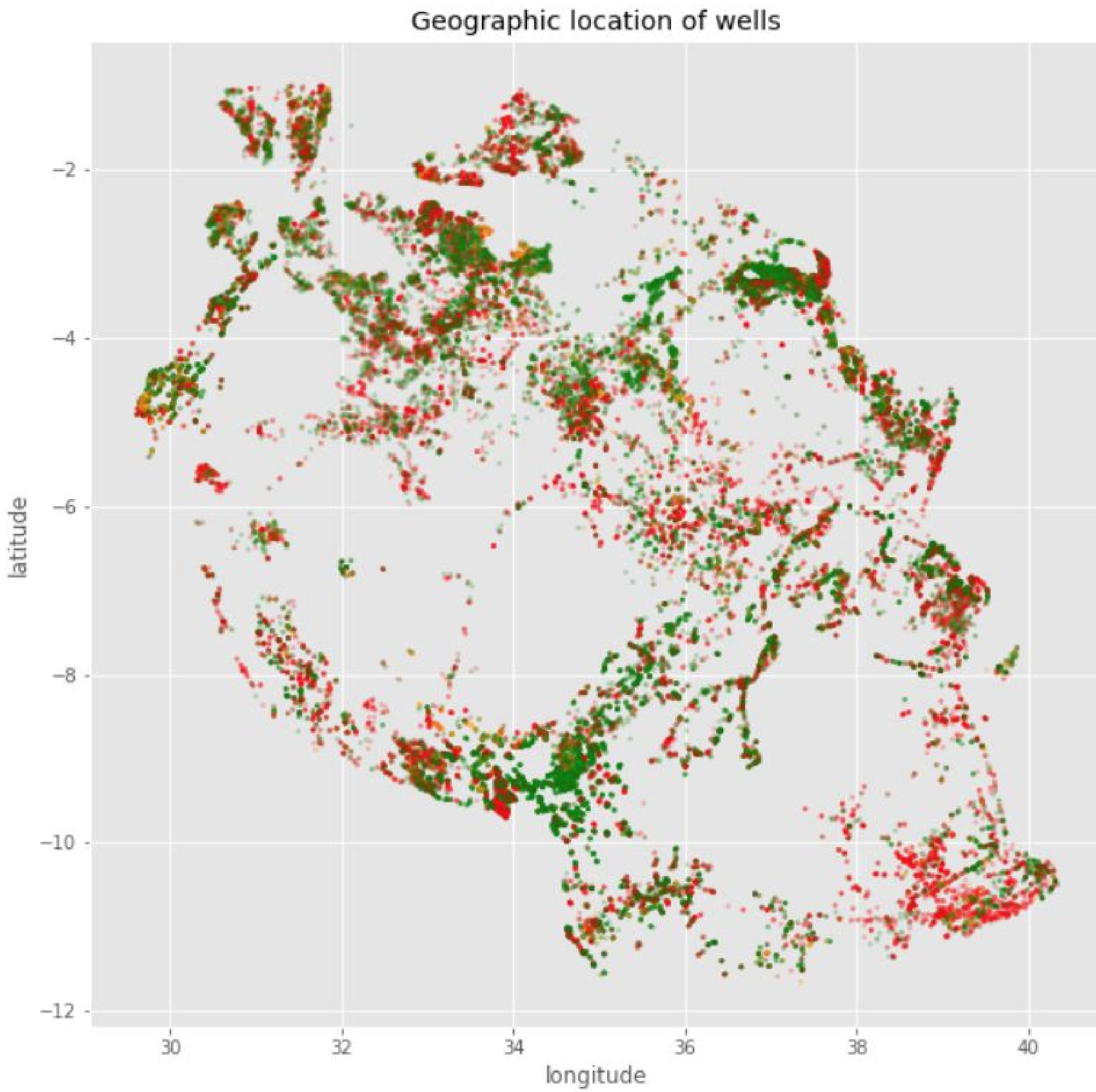
This project consists of data exploration and multiclass classification.
I plan to use one vs. rest classification technique.

The deliverables will be Jupyter Notebook, a technical report and a presentation of the project.

Data exploration and analysis

Geographic features:

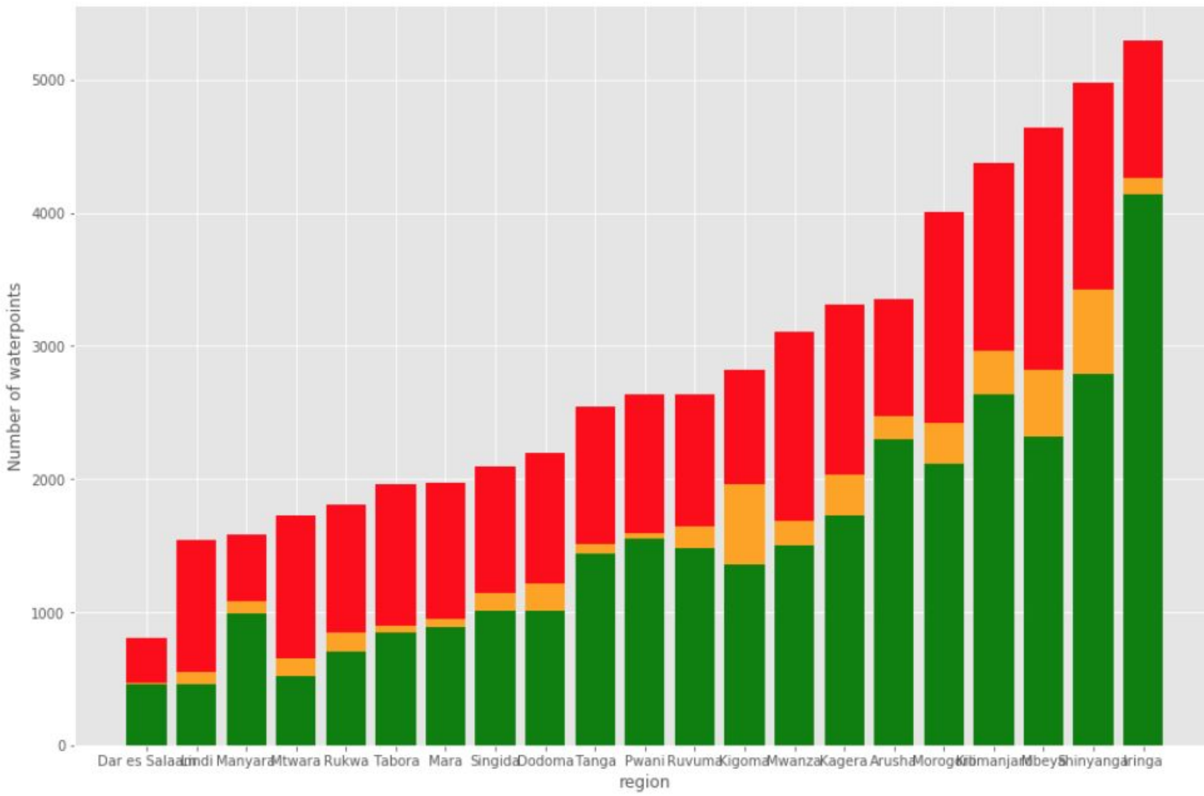
This dataset contains a lot of geographic features.
Let's explore some of them.
First, the latitude and longitude will give us an overview of the country.



The color code is green for functional, orange for functional but needs repair and red for non functional.
There is a majority of non functional water points at the South East of Tanzania.

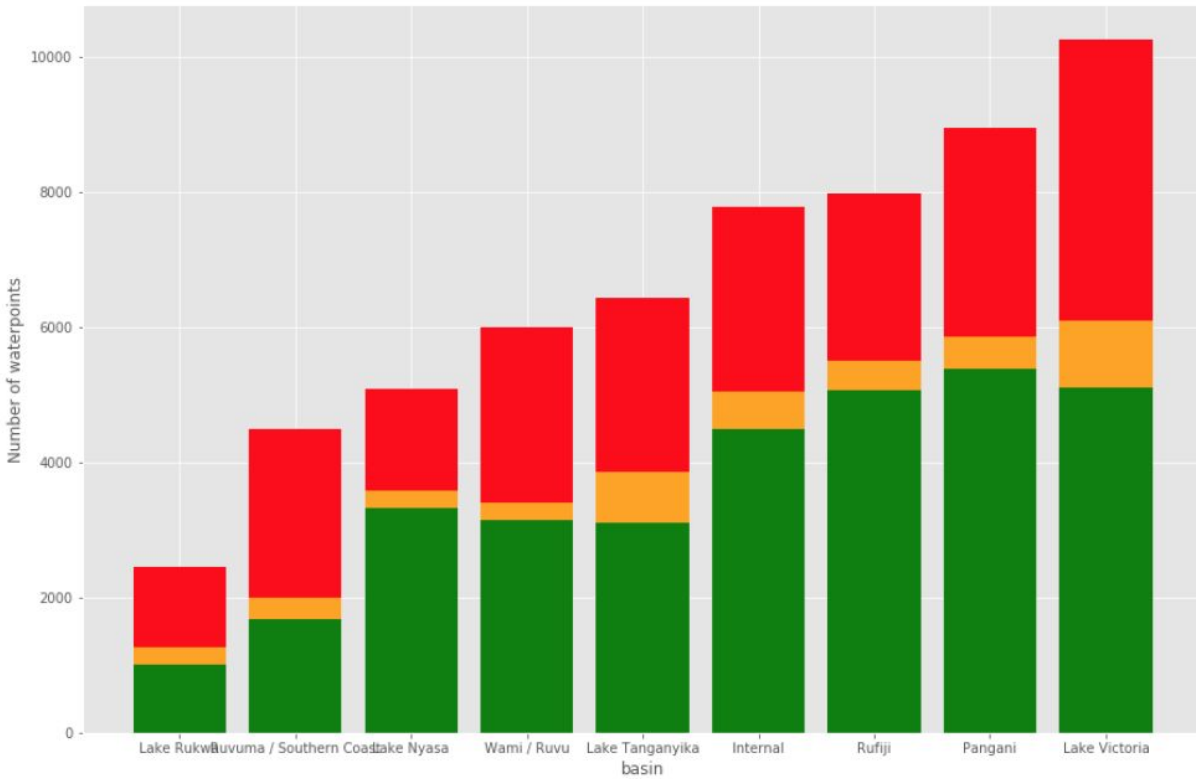
There are 1812 missing coordinates that we replace with the mean of the region.

There are 21 regions in Tanzania. Here is the repartition of water points for each region:



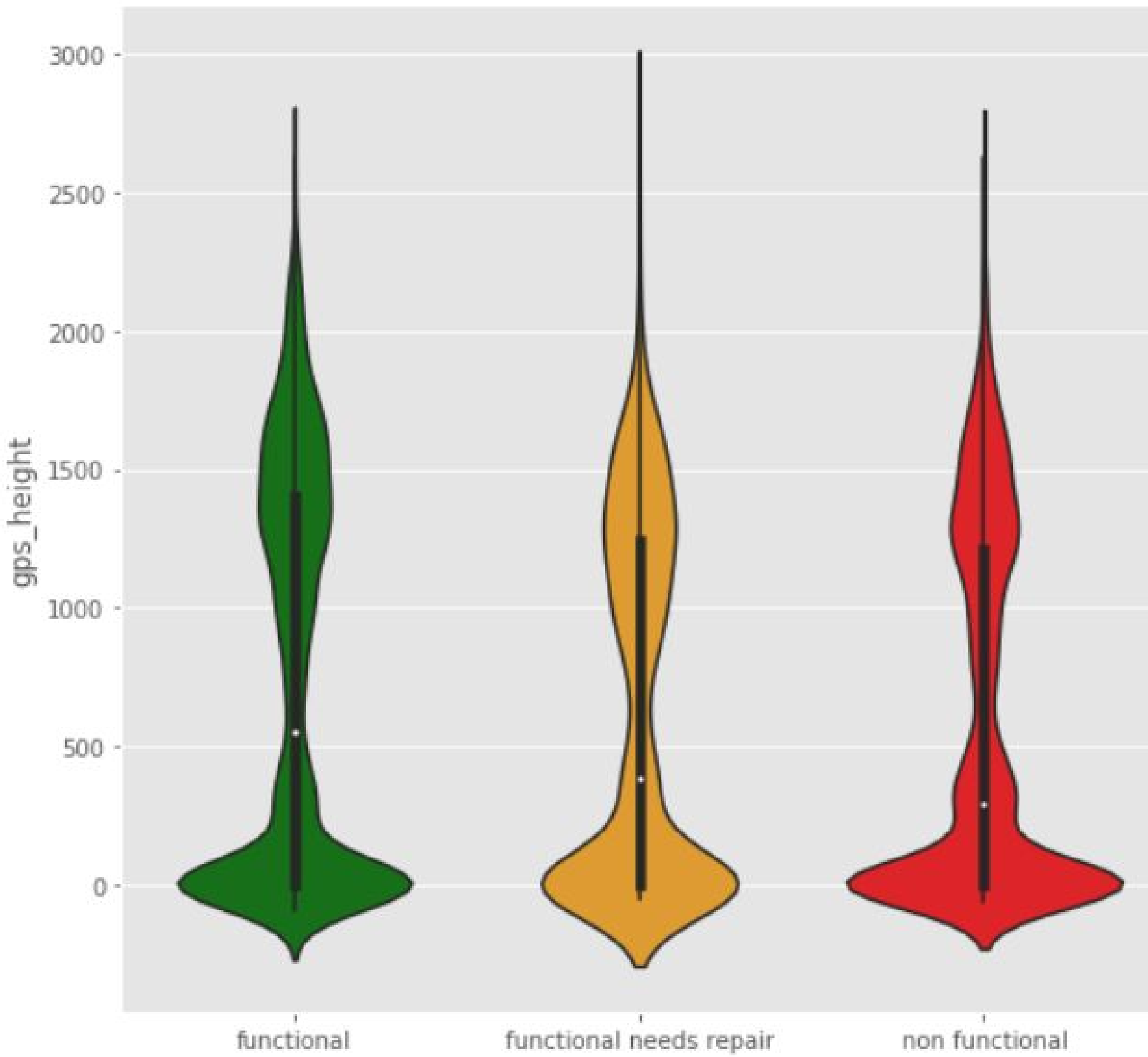
We can see that the proportions of the 3 operational status are different for the different regions. The chi-square test for independence confirms that the operational status repartition is different according to the region.

Tanzania is divided in 9 water basins, and they have a different operation status repartition as well, as shown by the plot and the chi-square test.



The other geographical features: lga, ward, district are also related to the operational status, as proved by the chi-square test for independence.
Ward as 2092 different values, so we will keep the 100 most frequent ones, and create a 'other' category.
We will get rid of region_code because it's similar to region, and of subvillage because it has 19287 different values, with a lot of them referring only to one waterpoint.

The last geographic feature is gps_height:



The violins look very similar, so let’s test the independence of each status against the others.

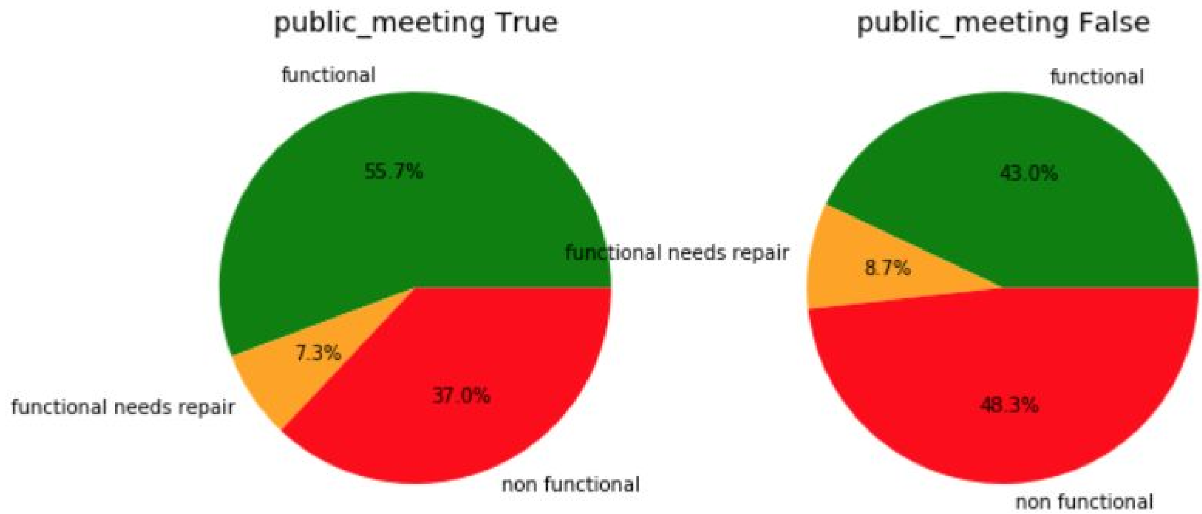
	t_stat	p_value
functional vs other	27.7151	5.42774e-168
functional needs repair vs other	-4.00602	6.18255e-05
non functional vs other	-26.2139	1.33281e-150

The low p_values show the repartition of gps_height for these groups is actually very different.

Construction features:

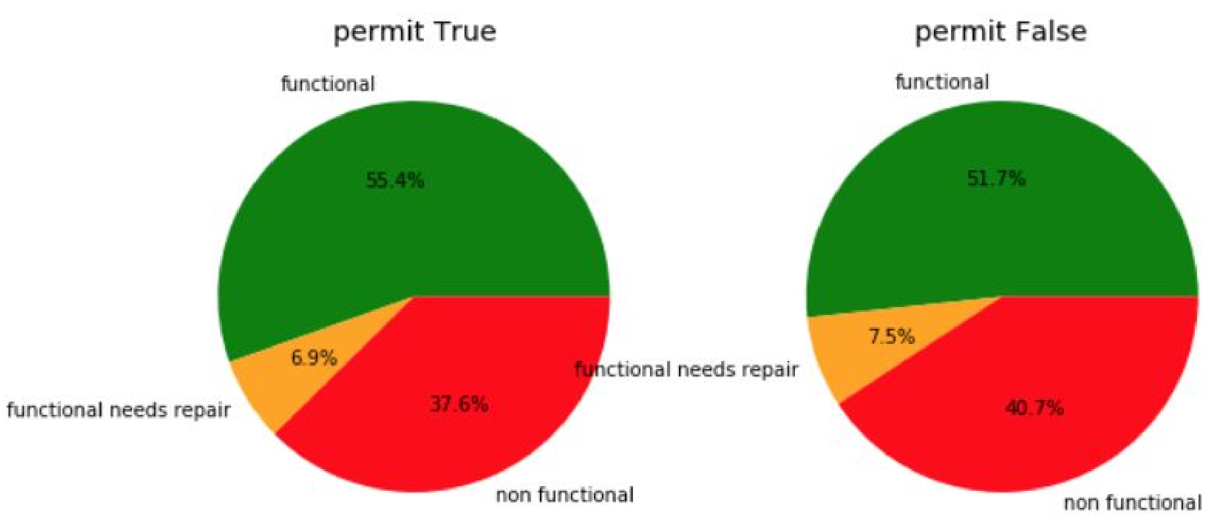
The funder and installer features are the names of the organizations that funded and installed the well. They have each about 2000 different values, with missing values and spelling mistakes. Let's fix some of the spelling mistakes, then keep the 100 most frequent organizations and name the rest of them 'other'.

56066 records report if there was a public meeting:



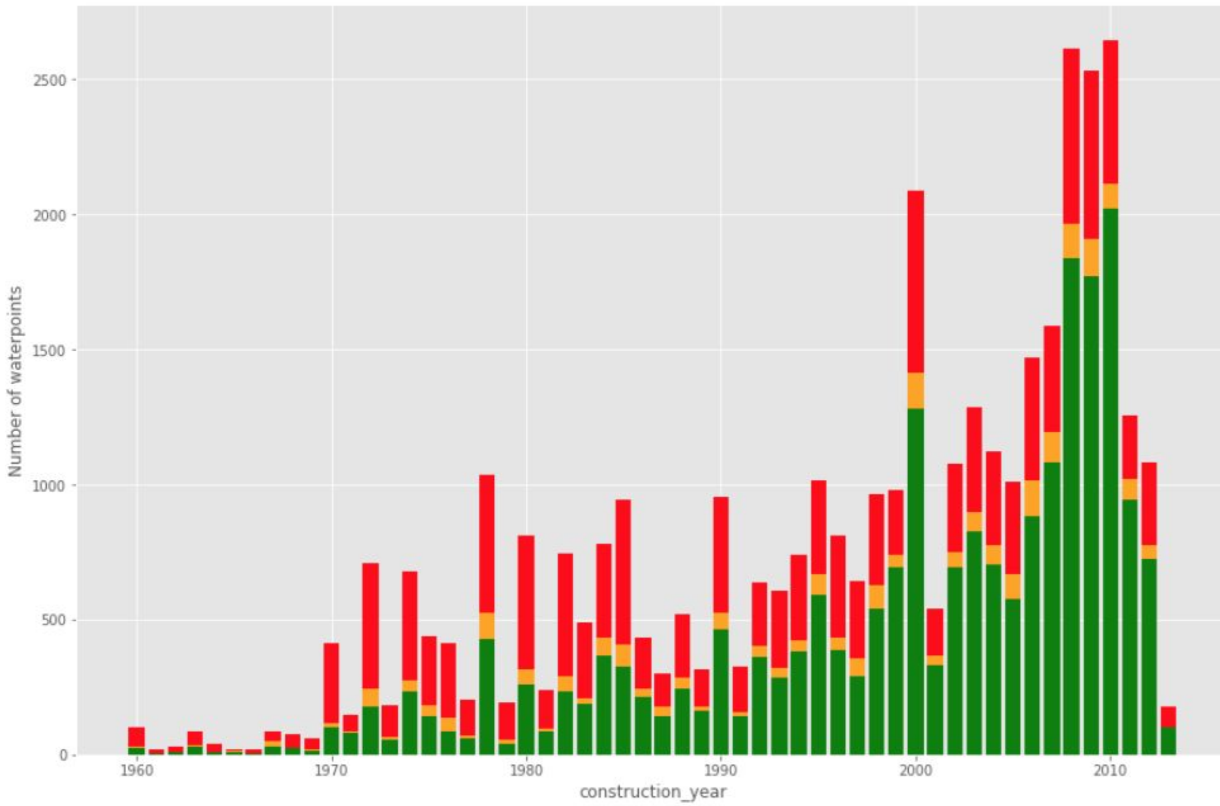
The proportion of functional water points is higher when there was a public meeting, so we will use this feature. Where it's missing, we set it to False.

56344 records report if there is a permit:



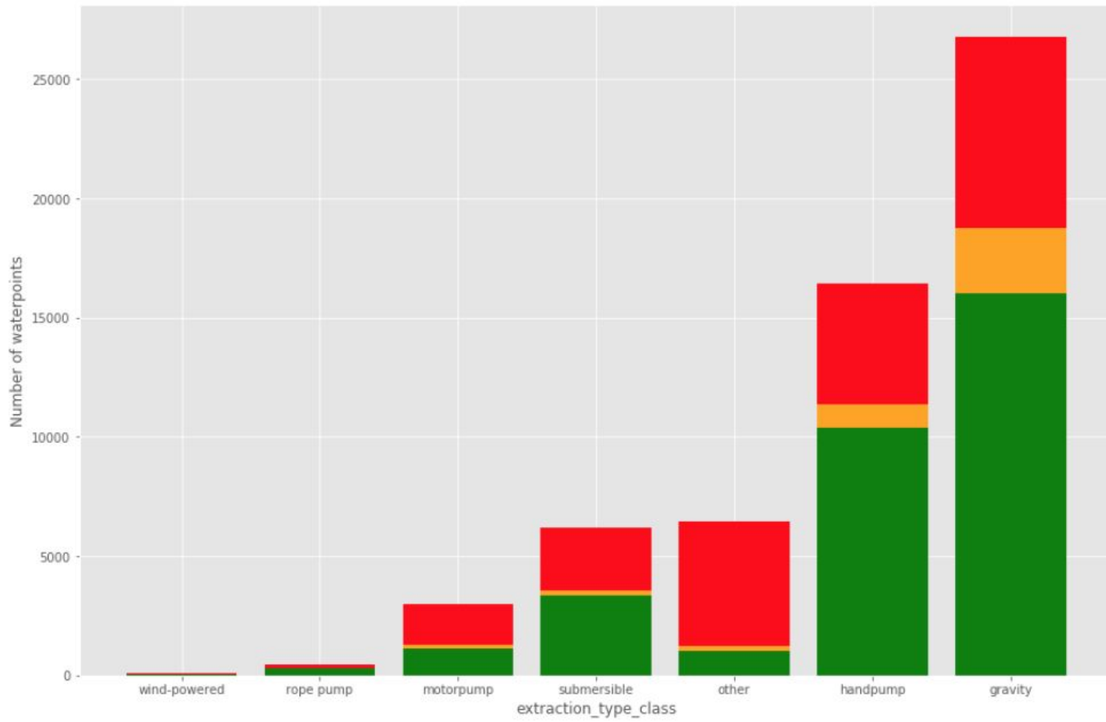
The proportions are not very different, but with the chi-square test, we can conclude that the permit and operational status are related.
Where this information is missing, we set it to False.

Here is a bar plot of the construction years:



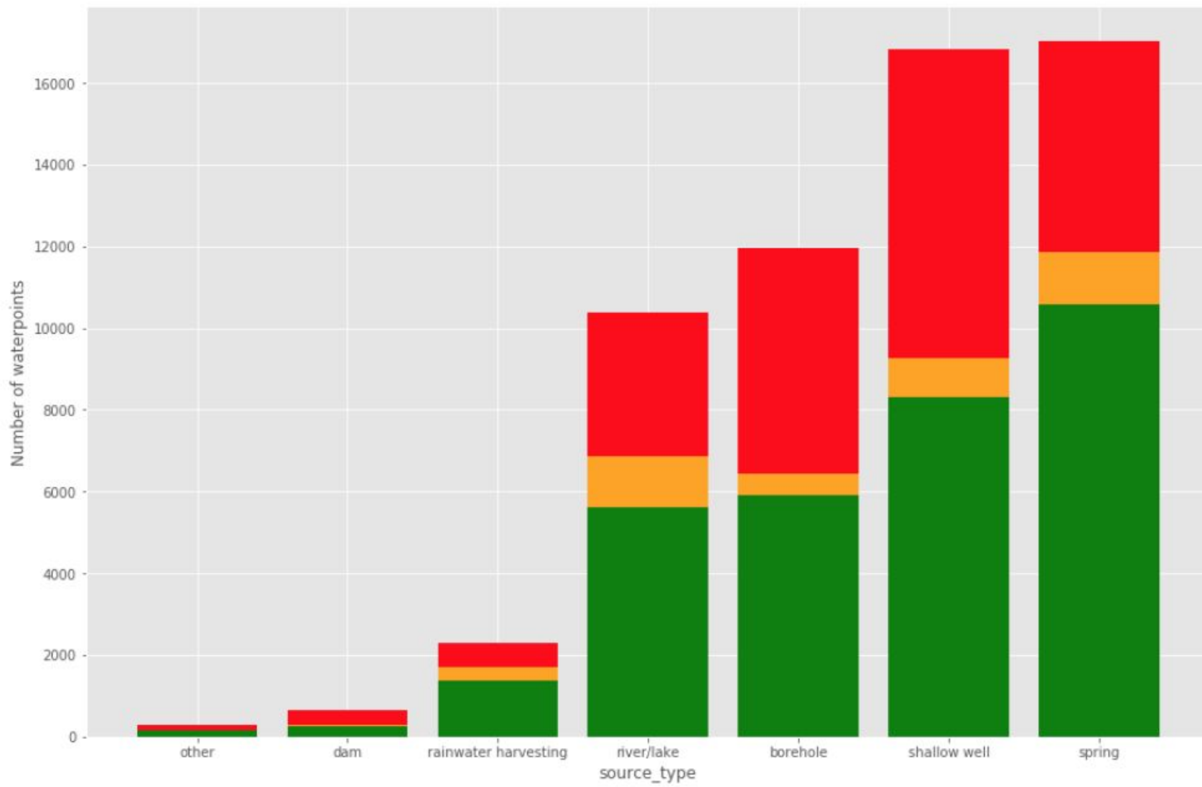
Older water points are more likely to be non functional.
Some construction years are missing. We set them to the mean of the construction years: 1997.

3 features describe the extraction type. We will keep only the extraction type class:



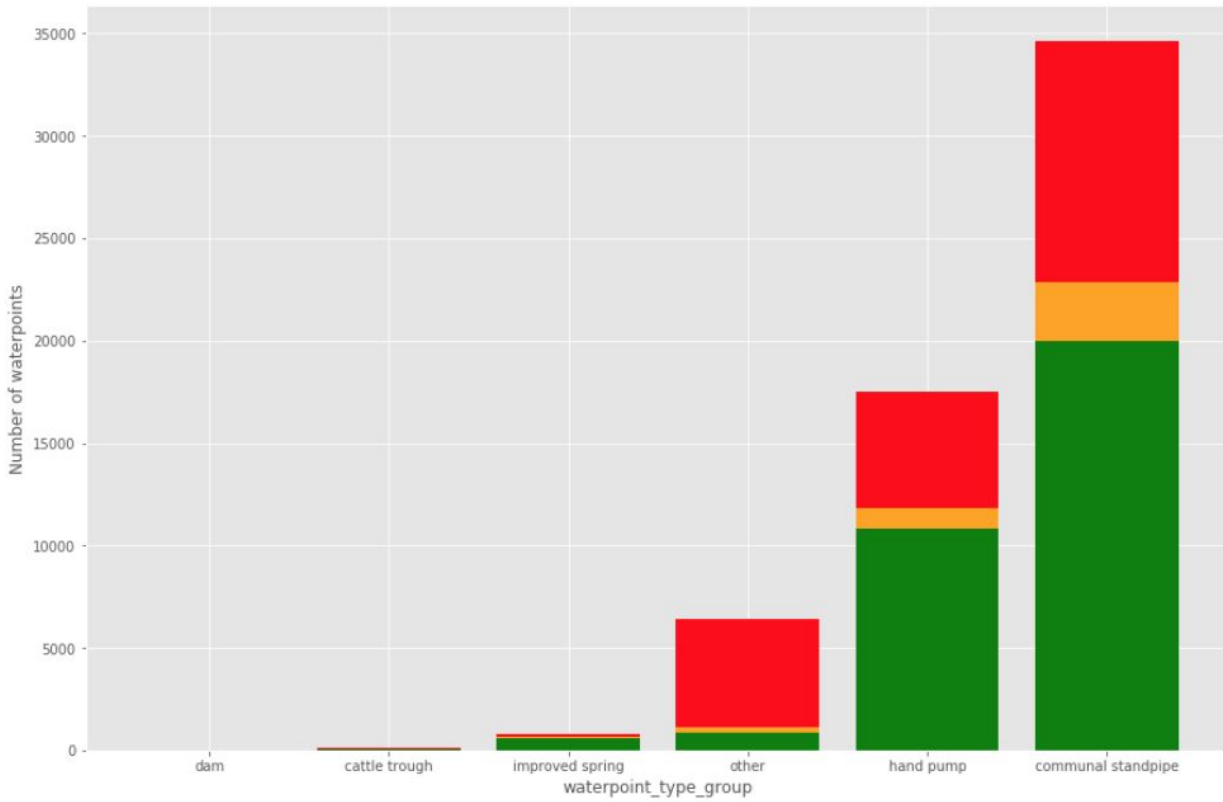
The repartition for these extraction types are very different. The chi-square test for independence shows a relation with the operational status.

3 features describe the source of the water. We will keep only the source type:



Shallow wells and boreholes are more likely to be non functional. Here again, the chi-square shows relation with the operational status.

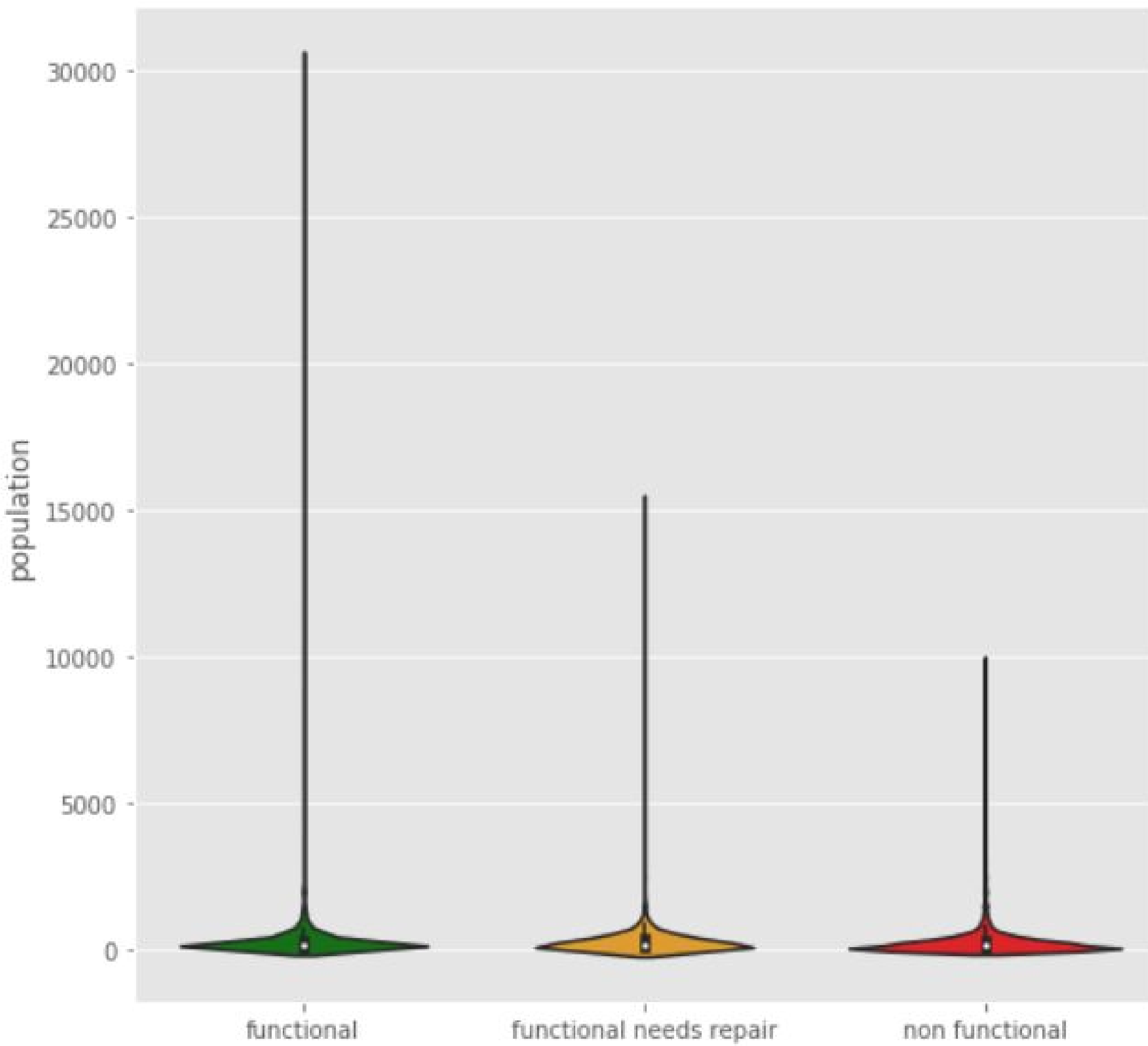
2 features describe the water point type. We will keep only waterpoint_type_group:



Chi-square test for independence shows the water point type and operational status are related.

Exploitation features:

Here is a violin plot of the population around the water points:

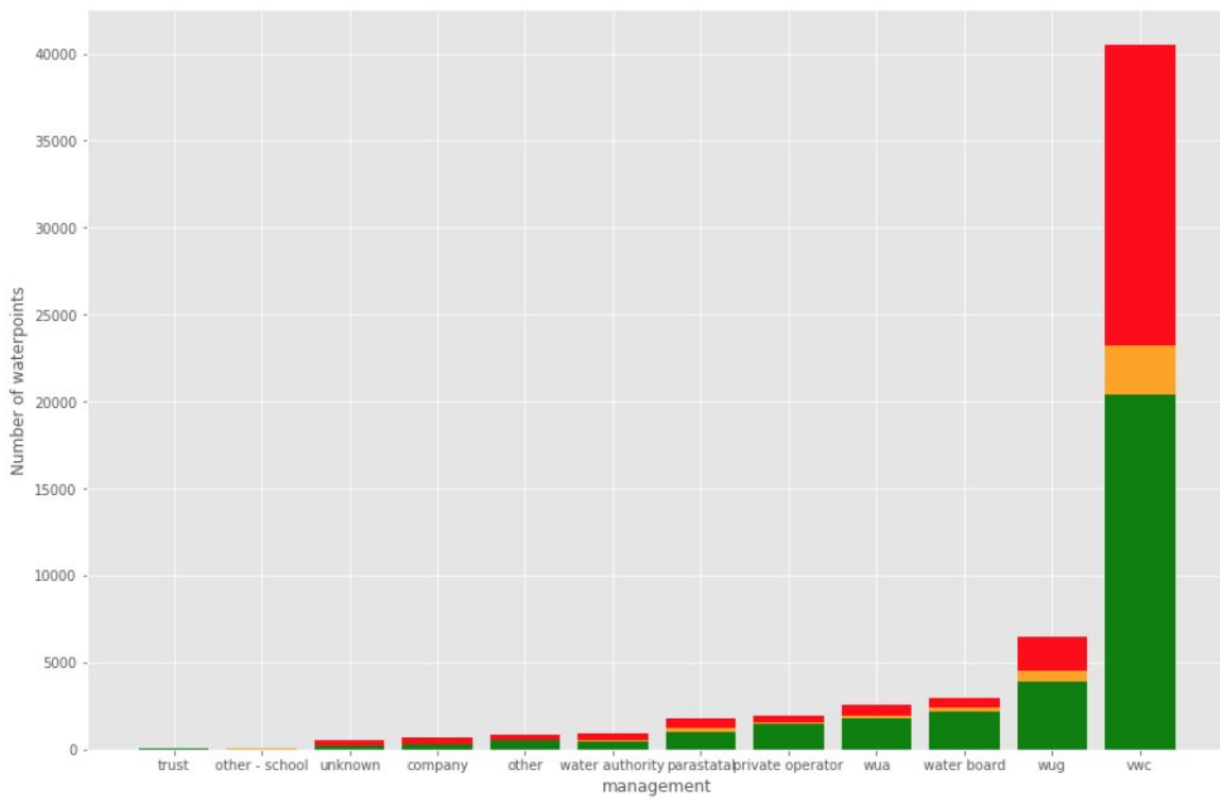


Let’s test the independence of each category against the others:

	t_stat	p_value
functional vs other	2.7702	0.00560495
functional needs repair vs other	1.50508	0.132312
non functional vs other	-3.61082	0.00030562

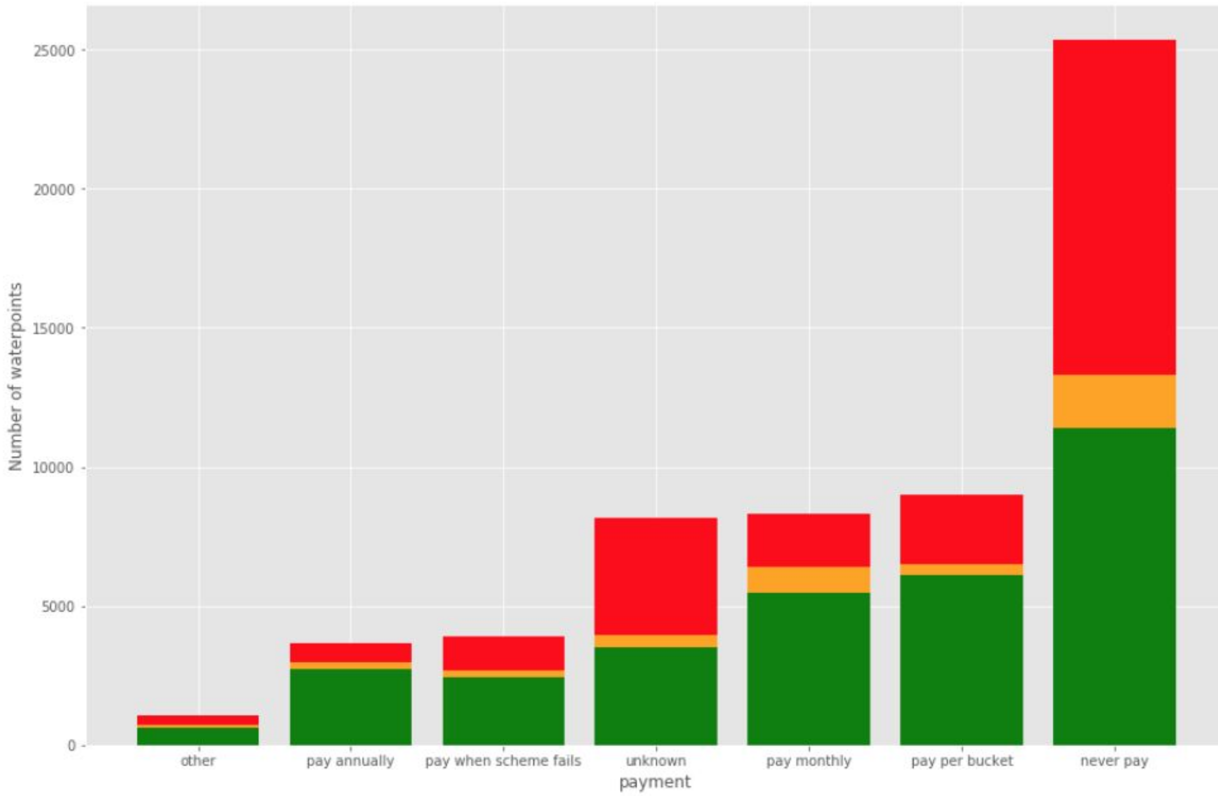
The low p_values for functional and non functional water points shows the population for these groups is very different.
The high p_value for functional needs repair shows the population for this group is not different from the rest of the waterpoints. It means it will be more difficult to predict this category accurately.
21381 values are missing, we set them to the median of the non zero values, because median is less sensitive to extreme values.

4 features describe how the water point is managed.
We drop scheme_name because half of the values are missing, scheme_management because it's redundant and has missing values, management_group because it's redundant.
We will keep only the management feature:



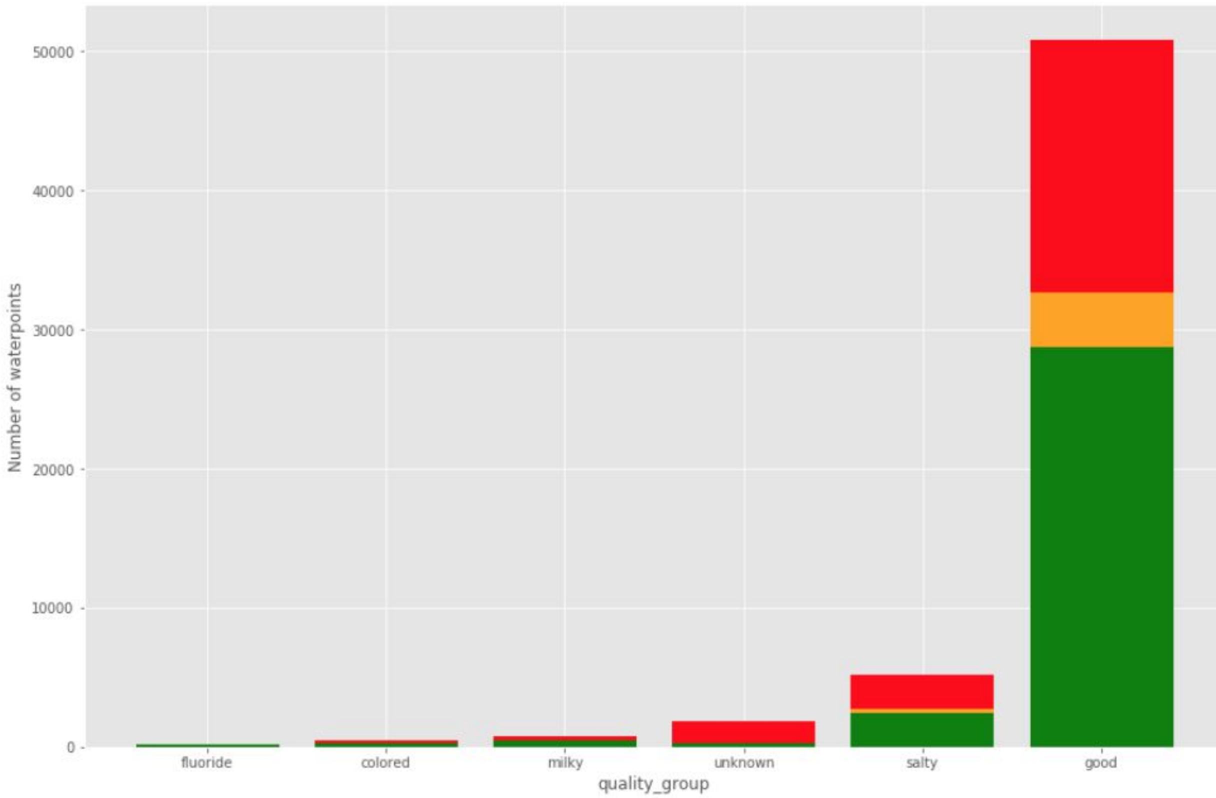
The chi-square test for independence shows that the management is related to the operational status.

2 features describe the payment type. We will keep only the payment, and order the categories: other < never pay < pay when scheme fails < pay annually < pay monthly < pay per bucket



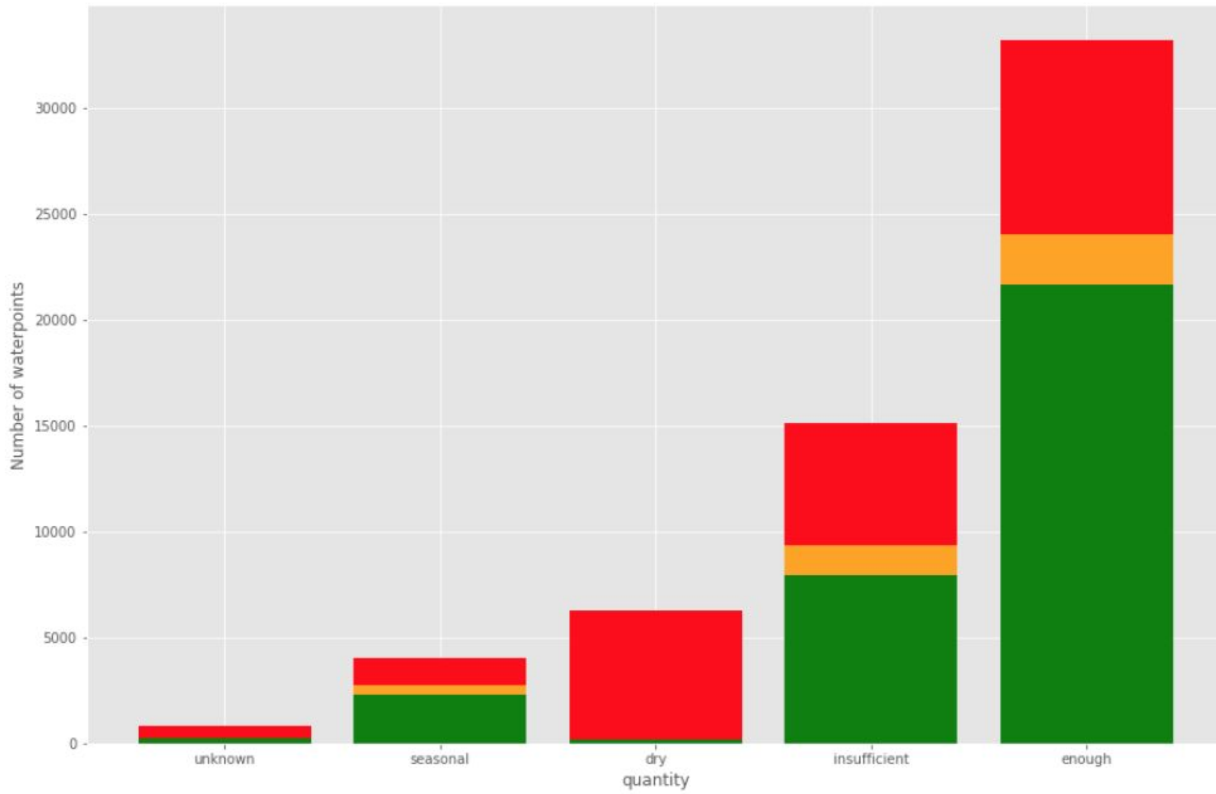
It seems that when there is payment for the water (either periodically, per bucket or on failure) the waterpoint is more likely to be functional. The chi-square test for independence confirms the relation.

2 features describe the water quality. We will keep only the quality_group, and order the categories: unknown < fluoride < salty < colored < milky < good.



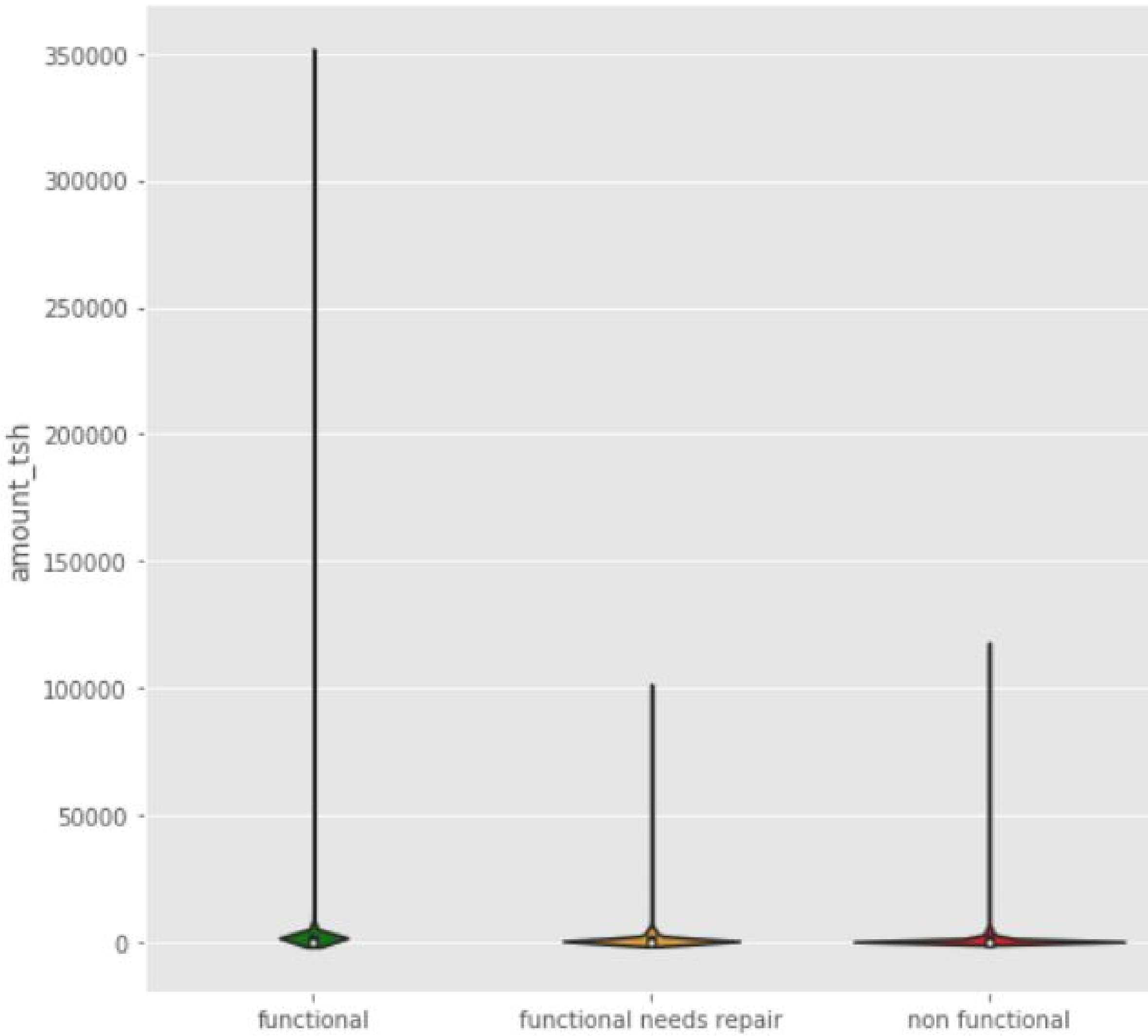
The chi-square test for independence shows the operational status is related to the water quality group.

2 features describe the water quantity. We will keep the quantity and order the categories: unknown < dry < insufficient < seasonal < enough.



The chi-square test for independence shows the operational status is related to the water quantity.

Amount_tsh is the amount of water available to waterpoint. Most of the values are 0. When we keep only non zero values, we get this plot:

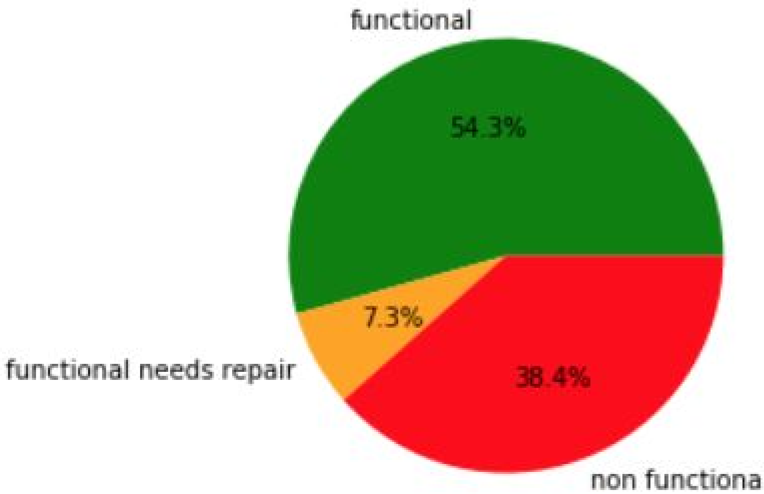


The t-test for independence of each operational status vs the others shows the functional and non functional groups are very different.
However, the p-value for functional needs repair vs the others is 0.29, which means the amount_tsh for this group is not different from the rest of the waterpoints.

We will drop the record features because they have no relation with the operational status: date_recorded, wpt_name, num_private, recorded_by.

Operational Status:

Here is the repartition of the status_group we try to predict:



We will order the categories: non functional < functional needs repair < functional.

The labels are not balanced, there is very few functional needs repair.

Machine Learning

Prepare data

The first step for machine learning is to prepare the data

Label encoding

We use label encoding for the categorical columns.
Some of the categories are ordered: payment, quality_group, quantity.
Some of the categories have 100 values: funder, installer, ward.
The operational status is label encoded with order.

Scaling

We scale the numeric columns because some machine learning algorithms work better with scaled features. For this operation we use scikit-learn MinMaxScaler to have only positive values.

Separate training from test data

We use 80% of the dataset to train models and keep 20% to calculate metrics on unseen data.
We stratify the split to keep the same proportion of each label in the datasets.

Metrics

To evaluate the classifications, we will compute metrics for a perfect model (where the predictions equal the real values), and for a dummy model (where the predictions are random values according to the proportions in the sample).
The metrics will be:

Confusion matrix

Dummy model:

```
Confusion matrix:
[[1708  333 2524]
 [ 334   58  471]
 [2523  472 3457]]
```

Perfect model:

```
Confusion matrix:
[[4565    0    0]
 [   0  863    0]
 [   0    0 6452]]
```

F1 score

Dummy model: 0.4396 Perfect model: 1.0

Kendall Tau

Kendall Tau measures the correspondence between 2 rankings. Since our labels are ordered, they are a ranking.
Dummy model: 0.0051 Perfect model: 1.0

Spearman rank order correlation

Dummy model: 0.0054 Perfect model: 1.0

Models

We will use one vs. rest classifier with different models. We will train them using our train dataset and calculate metrics on predictions from the test dataset.

Linear Logistic Regression

This is a simple linear model. We use dataset where numeric columns are scaled for this model. The resulting metrics are:

```
Confusion matrix:
[[2980  619  966]
 [ 231  307  325]
 [1403 1223 3826]]

F1 score:  0.6235282226649532

 KendalltauResult(correlation=0.3958756909764103, pvalue=0.0)

 SpearmanrResult(correlation=0.4241028317285591, pvalue=0.0)
```

The results are better than those of the dummy model. Most of the 'functional needs repair' and a lot of 'functional' are misclassified.
It seems the linear model is not so good for this project.

Support Vector Machine

This model is more complex and needs more computation. But because it finds a separating hyperplane, it could be good to predict the 'functional needs repair' class that is in between.
We use scaled numeric values for this model.
The results are:

```
Confusion matrix:
[[3062  644  859]
 [ 148  521  194]
 [1105 1102 4245]]

F1 score:  0.68160077339881

 KendalltauResult(correlation=0.48290909863813036, pvalue=0.0)

 SpearmanrResult(correlation=0.5109300473115832, pvalue=0.0)
```

More labels are well classified, and all the metrics are better.

Random Forest

We use non-scaled numeric values because Decision Trees are not based on distance, hence are not sensitive to features with different scales.
Here are the results:

```
Confusion matrix:
[[3535   83  947]
 [ 122  263  478]
 [ 563  205 5684]]

F1 score:  0.7915386179714657

 KendalltauResult(correlation=0.6519381180625193, pvalue=0.0)

 SpearmanrResult(correlation=0.6726210579677491, pvalue=0.0)
```

‘non functional’ and ‘functional’ are predicted more accurately, but the random forest has difficulties predicting the ‘functional needs repair’ class.
However, the metrics are better.

K Neighbors

We use scaled numeric values for this model.
Here are the results:

```
Confusion matrix:
[[3237  121 1207]
 [ 144  261  458]
 [ 733  173 5546]]

F1 score:  0.7542770773884885

 KendalltauResult(correlation=0.5756045832396937, pvalue=0.0)

 SpearmanrResult(correlation=0.5940009697597216, pvalue=0.0)
```

The metrics are better than SVM but the ‘functional needs repair’ class is not well predicted.

Naive Bayes

This model usually does well with categorical features. We use scaled numeric values because Multinomial Naive Bayes only work with positive values.
Here are the results:

```
Confusion matrix:
[[2770  189 1606]
 [ 253  111  499]
 [2424  245 3783]]

F1 score:  0.5570665433208284

 KendalltauResult(correlation=0.21028350312591165, pvalue=1.0028021515916738e-127)

 SpearmanrResult(correlation=0.21935655415749256, pvalue=2.1365232447842717e-129)
```

This model is not good for this project.

AdaBoost

Since the best results were obtained with Random Forest, we try AdaBoost to see if it can better classify the ‘functional needs repair’ labels.
We use non-scaled numeric values for this model.
Here are the results:

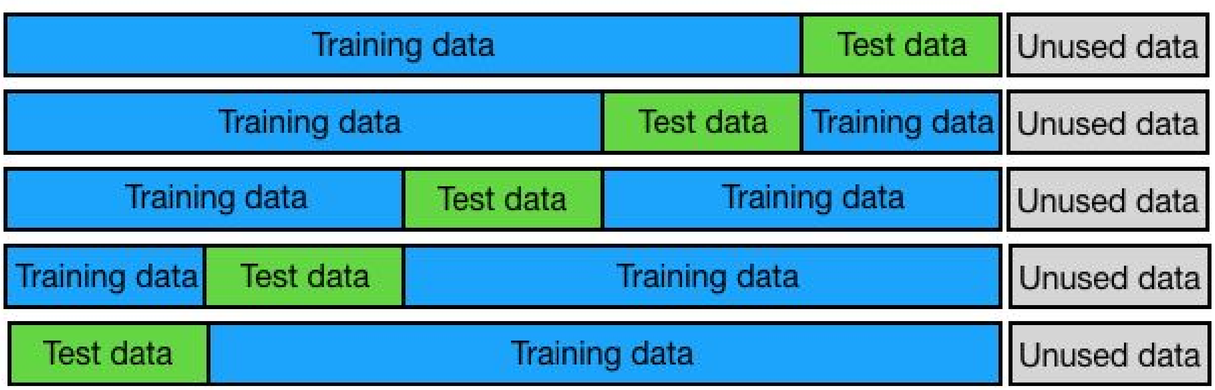
```
Confusion matrix:
[[2807   14 1744]
 [ 142   48  673]
 [ 597   29 5826]]

F1 score:  0.7039081699067322

 KendalltauResult(correlation=0.5173871119371819, pvalue=0.0)

 SpearmanrResult(correlation=0.5324603365590895, pvalue=0.0)
```

No, Random Forest was a better model.

Optimize the best model: One vs Rest Random Forest Classifier

First, let's have a look at the model. Looking at the features importance, public_meeting and permit have less importance. To simply the model, we remove these features.

Then, let's use K Folds on the training data to optimize the parameters.
We will use 5 splits so that every time 80% of the training data is used to train and 20% is used to test.



Here are the results for n_estimators:

	f1 score	kendall tau	spearman rank
10	0.790242	0.644971	0.664640
50	0.794610	0.652242	0.671832
100	0.794887	0.652630	0.672193
150	0.794774	0.651849	0.671210
200	0.794847	0.652376	0.671844
300	0.795418	0.653653	0.673222

There is a good improvement from 10 to 50, so I choose 50 for n_estimators.

Here are the results for max_depth:

	f1 score	kendall tau	spearman rank
10	0.739290	0.586286	0.602420
50	0.792067	0.649317	0.669217
100	0.791718	0.647850	0.667536
150	0.792005	0.648011	0.667634
200	0.793095	0.650841	0.670610

There is a good improvement between 10 and 50, so I choose to use max_depth=50.

Test the optimized model on the held out test data

Let's test the optimized model: One vs Rest Random Forest Classifier with 50 estimators and 50 max depth.
We train the model on all the training data, and make predictions on the test data.

We get the results:

```
Confusion matrix:
[[3613   86  866]
 [ 136  305  422]
 [ 545  207 5700]]

F1 score:  0.8044211481385531

 KendalltauResult(correlation=0.6728608366847327, pvalue=0.0)

 SpearmanrResult(correlation=0.693259182944975, pvalue=0.0)
```

These are the best results!

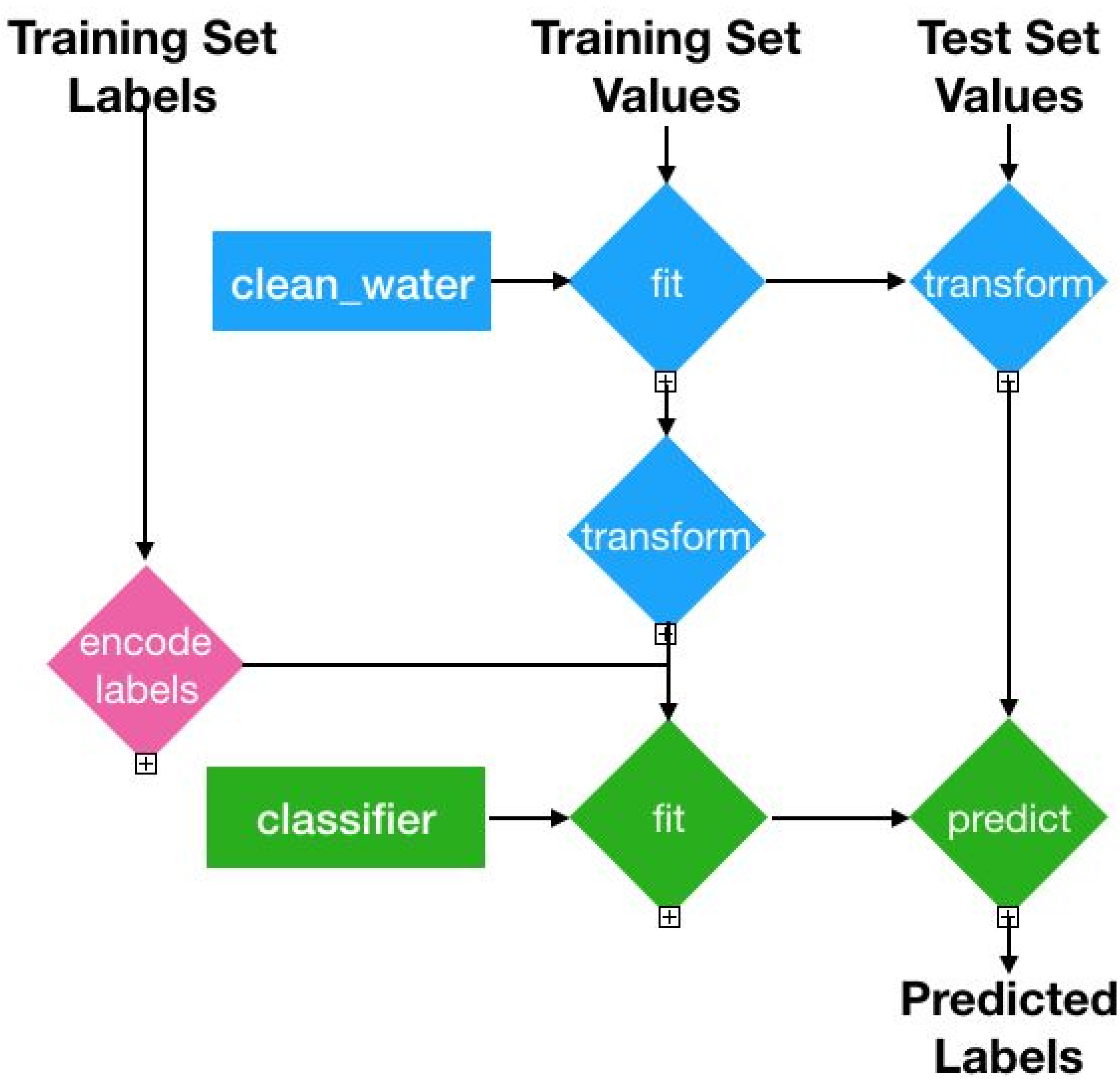
Synthesis

Now, we synthesize the snippets of code for data cleaning, model training and prediction. This will allow us to easily work on any training and test datasets.

We will create a class clean_water with methods:

- fit(X) : gets the mean, median and most frequent values from the training dataset.
- transform(X): transforms the dataset to clean and encode values.

Now, we can train the model using the whole Training Set provided, and predict values for the Test Set provided.
Here are the steps of the whole process:



Now we can submit the predicted labels to DrivenData and get a score:

Submissions

BEST	CURRENT RANK	# COMPETITORS	SUBS. TODAY
0.8109	940	6643	1 / 3

EVALUATION METRIC

Classification Rate $= \frac{1}{N} \sum_{i=0}^{N} I(y_i = \hat{y}_i)$

The metric used for this competition is the classification rate, which calculates the percentage of rows where the predicted class $\hat{y}$ in the submission matches the actual class, y in the test set. The maximum is 1 and the minimum is 0. The goal is to maximize the classification rate.