Predicting The Functional Status of Water Pumps in Tanzania

Krishna Sai Dheeraj Kalluri

Water is one of the minimum need for human beings, it is not only used for our daily needs but also for agriculture and industrial purposes. Though Tanzania has access to a lot of water, the country still faces the dilemmas of many African countries where many areas have no reliable access to water. We are looking at the dataset of water pumps in Tanzania to predicting the operating condition of a water point.

“By predicting status of the functioning of pumps, the Tanzanian Ministry of Water can improve the maintenance operations of the water pumps and make sure that clean, potable water is available to communities across Tanzania.”

Data description

- amount_tsh – Total static head (amount water available to water point)
- 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 -No description
- 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 water point
- scheme_name – Who operates the water point

- permit – If the water point is permitted
- construction_year – Year the water point was constructed
- extraction_type – The kind of extraction the water point uses
- extraction_type_group – The kind of extraction the water point uses
- extraction_type_class – The kind of extraction the water point uses
- management – How the water point is managed
- management_group – How the water point 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

Most of the features are categorical.

Our target is status_group and it is classified into one of the three categories:

1) functional,
2) non-functional, or
3) functional but need repair.

Our goal is to predict the labels of status_group for test data using the predictor variables. This is a classic classification problem.

Data cleaning:

The data on the first look looks like it is clean, but there's lot of missing values and ariety in data and we have to clean before actually using the dataset for training.

1. By eyeballing all the features, there were some of the features which didn’t have any significance, such as water point name, water point ID which can’t possibly affect the functionality of water point. Such features were removed.
2. Few features with same information content were repeated such as (extraction_type, extraction_type_group, extraction_type_class), (payment, payment_type), (water_quality, quality_group), (source, source_class), (subvillage, region, region_code, district_code, ward), all contain similar representation of data in different grains. Hence, we risk overfitting our data during training by including all the features in our dataset.

```
training_df = training_df.drop(['id', 'source', 'wpt_name',  'num_private', 'region',

                                           'quantity'], axis = 1)

training_df = training_df.drop(['quality_group' ,'lga' ,'ward', 'management', 'payment',
                            'extraction_type_group','extraction_type_class'], axis = 1)
```

3. In some of the features, values with more number of levels were converted to variables with lesser number of levels. We tried multiple techniques to reduce the arity such as generation of synthetic levels and exploring dimension reduction techniques such as principal component analysis (PCA).

```
def funder_cl(row):
    if row['funder']=='Government Of Tanzania':
        return 'gov'
    elif row['funder']=='Danida':
        return 'danida'
    elif row['funder']=='Hesawa':
        return 'hesawa'
    elif row['funder']=='Rwssp':
        return 'rwssp'
```

```
  elif row['funder']=='World Bank':
    return 'world_bank'
  elif row['funder']=='Kkkt':
    return 'Kkkt'
  elif row['funder']=='World Vision':
    return 'World Vision'
  elif row['funder']=='Unicef':
    return 'Unicef'
  elif row['funder']=='Tasaf':
    return 'Tasaf'
  elif row['funder']=='District Council':
    return 'District Council'
  else:
    return 'other'
training_df['funder'] = training_df.apply(lambda row: funder_cl(row), axis=1)
```

4. Construction_year, has lot of missing values and they are in string values, first converted them into date time object, and then converted the years into decades to make less levels of values.

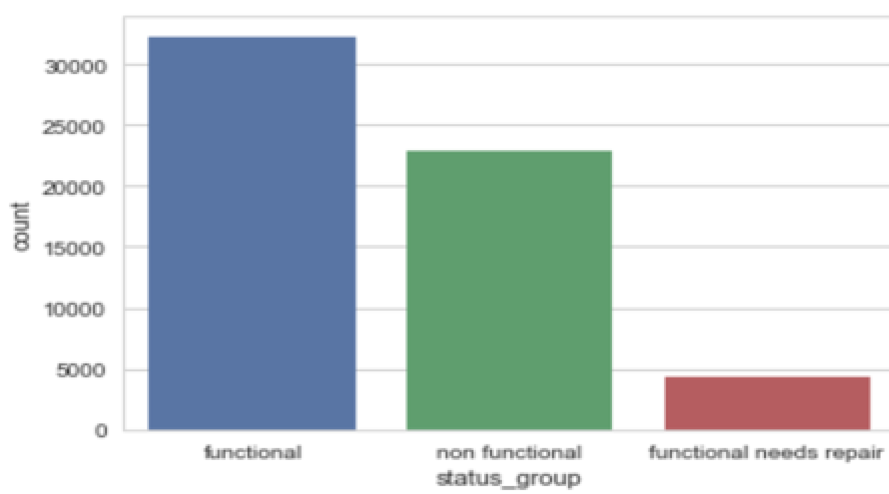
```
def construction_cl(row):
  if row['construction_year'] >= 1960 and row['construction_year'] < 1970:
    return '60s'
  elif row['construction_year'] >= 1970 and row['construction_year'] < 1980:
    return '70s'
  elif row['construction_year'] >= 1980 and row['construction_year'] < 1990:
    return '80s'
```

```
    elif row['construction_year'] >= 1990 and row['construction_year'] < 2000:
        return '90s'
    elif row['construction_year'] >= 2000 and row['construction_year'] < 2010:
        return '00s'
    elif row['construction_year'] >= 2010:
        return '10s'
    else:
        return 'unknown'
training_df['construction_year'] = training_df.apply(lambda row:
construction_cl(row), axis=1)
```

5. amount_tsh and gps_height had many zeros (‘0’).

Initial Data Exploration:

- By looking at the labels data we can see that there is lot of class imbalance. Like there are 0.543081 54.3% of functional pumps, 38.42 %of non-functional pumps, and 7.2% of the Pumps which are functional but needs repair.



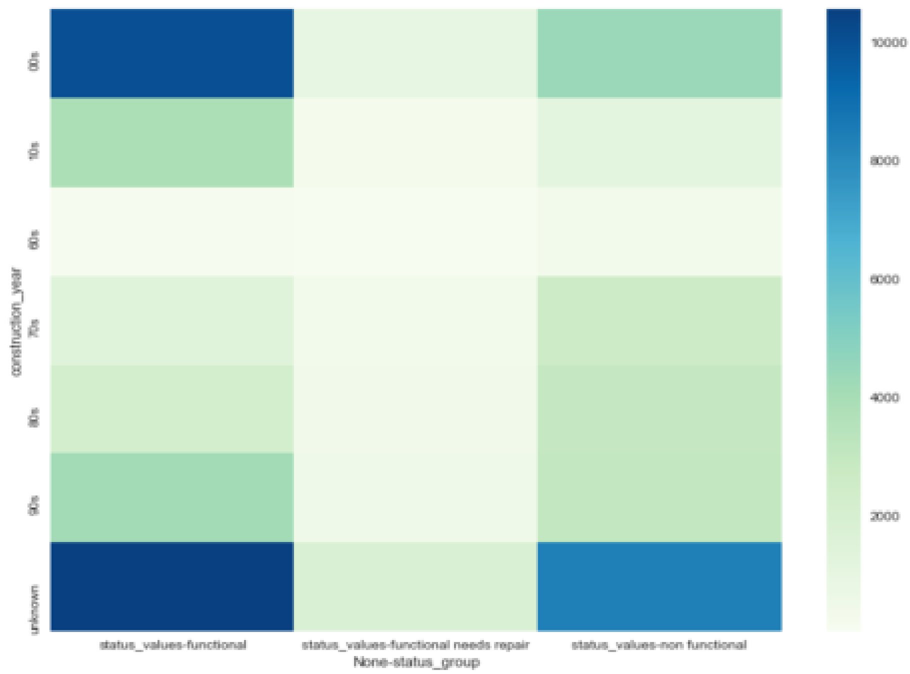
➢ Identified few features that seemed discriminative based on our human intuition. According to me, gps_height, basin, installer, population, scheme_management, construction year, payment type, source, and waterpoint_type seemed like they could be extremely important in identifying the pump status., So I grouped them according to the status group using pivot table.

Construction Year:

```
   piv_df=training_df[['construction_year', 'status_group',
'status_values']]
piv_table = piv_df.pivot_table(index='construction_year',
                    columns='status_group', aggfunc='count')
piv_table
```

	status_values		
status_group	functional	functional needs repair	non functional
construction_year			
00s	9989	977	4364
10s	3794	220	1147
60s	156	42	340
70s	1406	348	2652
80s	2220	423	2935
90s	4139	518	3021
unknown	10555	1789	8365

After creating the Pivot table, I have plotted a Heat map to look at the classification graphically.

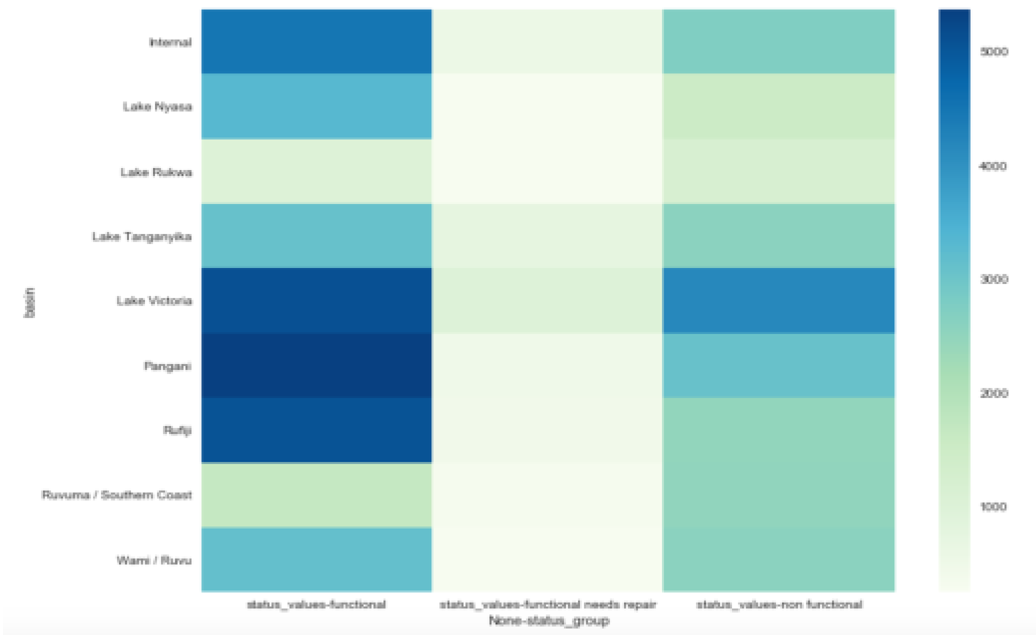


Basin:

```
piv_df= training_df[['basin','status_group','status_values']]
piv_table = piv_df.pivot_table(index='basin',
                          columns='status_group', aggfunc='count')
piv_table
```

	status_values		
status_group	functional	functional needs repair	non functional
basin			
Internal	4482	557	2746
Lake Nyasa	3324	250	1511
Lake Rukwa	1000	270	1184
Lake Tanganyika	3107	742	2583
Lake Victoria	5100	989	4159
Pangani	5372	477	3091
Rufiji	5068	437	2471
Ruvuma / Southern Coast	1670	326	2497
Wami / Ruvu	3136	269	2582

Heat Map



This analysis is conducted on all the features and by looking at the conditions most influencing the status of the pumps, main focus is on detecting the functional needs repair pumps.

Data Modelling and Transformation:

Using this new data with 20 features, tried different ensemble methods to find the model with true positives for the status_group- Functional and needs repair.

Random Forest Classifier

```
rfc = sklearn.ensemble.RandomForestClassifier(n_estimators=1000,
                                              min_samples_split=6,
                                              criterion='gini',
                                              max_features='auto',
                                              oob_score=True,
                                              random_state=1,
                                              n_jobs=-1)
```

Decision Tree:

```
dtc = DecisionTreeClassifier(criterion='gini',
                             max_depth = 10,
                             max_features = 'auto',
                             random_state = 1,
                             splitter = 'best')
```

Extra Tree Classifier:

```
ETC = ExtraTreesClassifier(n_estimators=1000,min_samples_split=10)
```

Gradient Boosting:

```
param_grid = {'learning_rate': [0.075,0.07],
              'max_depth': [6,7],
              'min_samples_leaf': [7,8],
              'max_features': [1.0],
              'n_estimators':[100, 200]}
```

After training all these models the results of each classifier is taken for comparison.

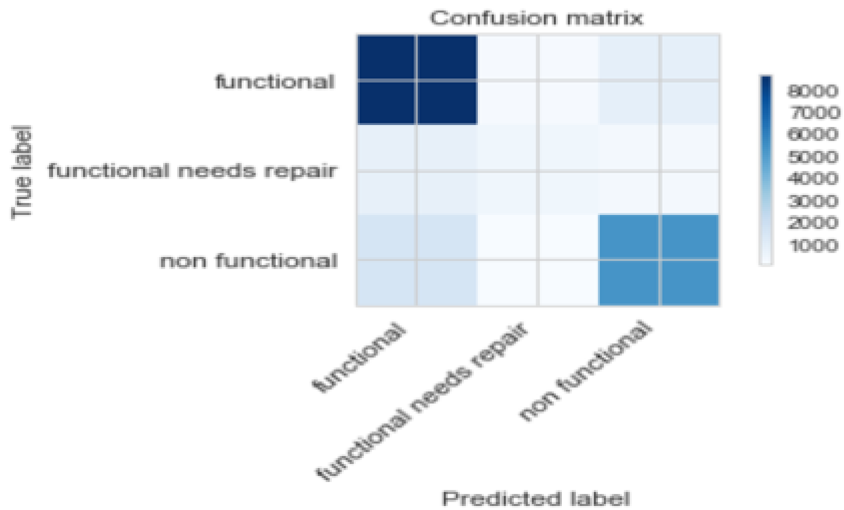
Classifier	**Training Accuracy Score**	**Test Accuracy Score**
Random Forest Classifier	**94.01**	**80.81**
Decision Tree	**75.32**	**72.77**
Extra Tree Classifier	**91.19**	**80.04**
Gradient Boosting	**92.38**	**79.37**

Table 1. Validation Accuracy scores of different classifiers.

Once training is done, constructed the confusion matrixes and the classification reports for each of the classifier to select the model which has more tpr for the functional but needs repair group.

Random Forest Classifier

```
Confusion matrix
[[8650  157  805]
 [ 736  385  212]
 [1512   58 5305]]
```

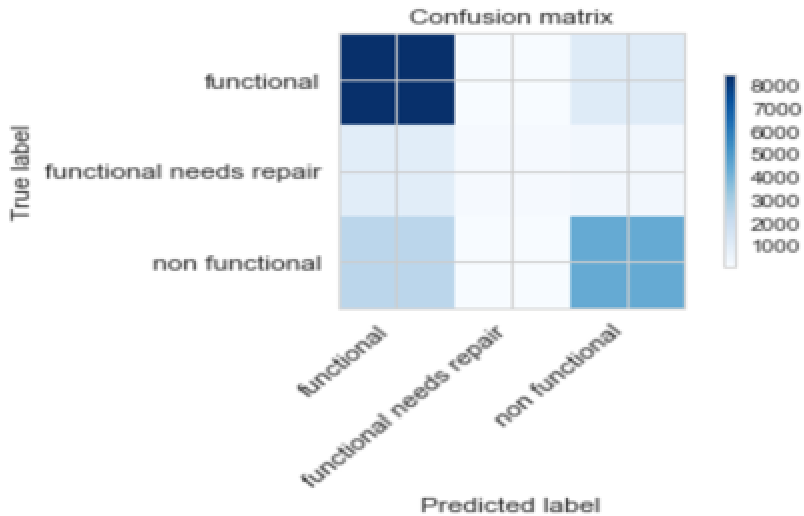


Classification Report

	precision	recall	f1-score	support
functional	0.79	0.90	0.84	9612
functional needs repair	0.64	0.29	0.40	1333
non functional	0.84	0.77	0.80	6875
avg / total	0.80	0.80	0.79	17820

Decision Tree:

```
Confusion matrix
[[8449   70 1093]
 [ 940  135  258]
 [2441   49 4385]]
```

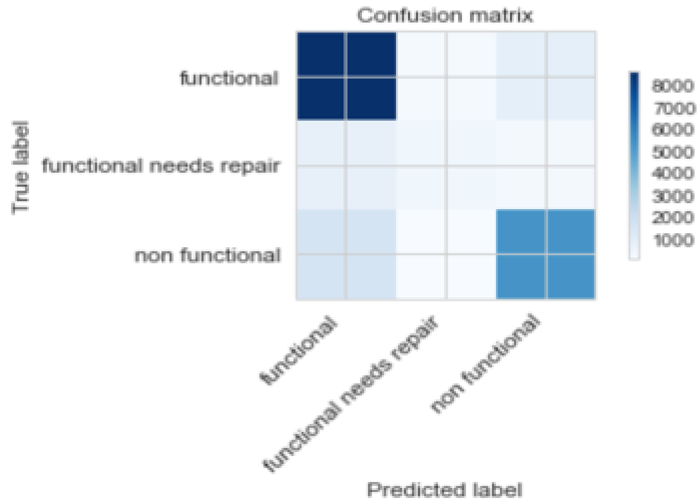


Classification report

	precision	recall	f1-score	support
functional	0.71	0.88	0.79	9612
functional needs repair	0.53	0.10	0.17	1333
non functional	0.76	0.64	0.70	6875
avg / total	0.72	0.73	0.71	17820

Extra Tree Classifier:

```
Confusion matrix
[[8607  163  842]
 [ 734  387  212]
 [1544   61 5270]]
```



Classification report

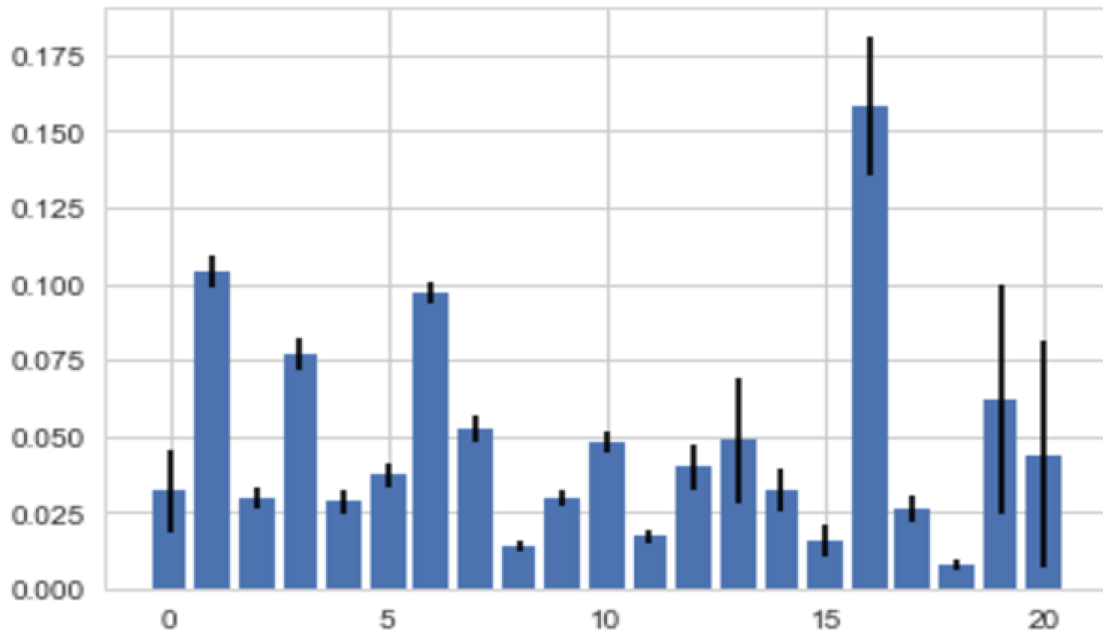
```
                         precision    recall  f1-score   support

             functional       0.79      0.90      0.84      9612
functional needs repair       0.63      0.29      0.40      1333
         non functional       0.83      0.77      0.80      6875

            avg / total       0.80      0.80      0.79     17820
```

Looking at the results Confusion Matrix and the classification reports of each classifier, the Random Forest classifier and the Extra Tree Classifier are performing almost similar when compared between Functional needs repair.

The feature importance is obtained and compared by plotting a bar graph. And the most important features are as follows, Quantity group, waterpoint_tyoe, days since recorded, gps_height, sub-village,



['amount_tsh': 0.03890503, 'days_since_recorded': 0.10767824

'funder': 0.03318091, 'gps_height': 0.08149866, 'installer': 0.02799432,

'basin': 0.0403402, 'subvillage': 0.09869852, 'population': 0.05493696,

'public_meeting': 0.01485134, 'scheme_management': 0.03185649,

'scheme_name': 0.05003365, 'permit': 0.01792227,

'construction_year': 0.03814958, 'extraction_type': 0.04968215,

'payment_type': 0.03841178, 'water_quality': 0.01915135,

'quantity_group': 0.13095799, 'source_type': 0.03165223,

'source_class': 0.01228557, 'waterpoint_type': 0.0476140,

'waterpoint_type_group': 0.03419865]

Final Results:

Analysis leads us to believe that the dataset was difficult to classify. Prioritized the classification to identify the pumps that needs repair. The cost of a standard pump ranges from $100 - $2000. Installing this pump requires drilling which can be anything between $1000- $3000. On the other hand maintaining the pump would only cost tens of dollar, Which could help the Tanzanian water industry millions of dollars.

Future Work:

1) We could gather the exact population data from external sources and add it to our dataset, and check if the population in particular areas is affecting the functional status of the pump, which can help them to predict the areas and take more care in that area.

2) We can identify the life time of a pump by using the historical data, which could help the Tanzanian water ministry to help them repair them before they get non-functional.

3) Density Cube to enhance our general performance we think another technique worth investigating would be to build a density cube utilizing negative examples, and after that testing using positive examples.This strategy might help us identify pattern in distribution of positive and negative examples.

Project Location: https://github.com/ksdkalluri/identifying_faulty_pumps