Pump it Up: Data Mining the Water Table

*The prediction and characteristics of operating condition of waterpoints in Tanzania by using data modelling, analysis and machine learning

1st Wenxuan Zhu
University of Nottingham
20379163
Nottingham, UK
alywz30@nottingham.ac.uk

2nd Hong Li
University of Nottingham
20438363
Nottingham, UK
psxhl10@nottingham.ac.uk

***Abstract*—Pump it Up: Data Mining the Water Table is a dataset about the pump water supply system in Tanzania. In this paper, we use data analysis, data preprocessing, and data classification methods to analyze and process the dataset in detail, and develop Decision tree and XGBoost (eXtreme Gradient Boosting) models to analyze the correlation of data features and predict the operation status of water pumps. And the performance of the two models is compared. The research results show that XGBoost has good performance, with an accuracy of 84.1%. In addition, we also conducted a comparative analysis of natural and human factors and studied their relationship with water point conditions. This has important reference value for decision-makers and water supply managers, which can help them better understand and optimize the operation of pump water supply system and ensure the supply of water resources.**

***Index Terms*—data modelling, data analysis, machine learning**

I. Introduction

Providing clean and reliable water is crucial for community well-being and sustainable development. In many underdeveloped areas, obtaining safe water remains a challenge, with a significant number of water pumps and systems facing issues such as malfunction, breakdowns, and inadequate maintenance. In order to solve these problems, governments and societies in various countries have invested a large amount of manpower and material resources. An open source web app named Taarifa aggregated data from the Tanzania Ministry of Water and the Pump it Up: Data Mining the Water Table dataset was created to explore and analyze factors related to the functionality of Tanzania water pumps.

A. Introduction to the data set

The data set comes from the waterpoints dashboard of an open source web app named "Taarifa" which aggregates data from the Tanzania Ministry of Water. This data set aims to help predict and improve the operational status of water pumps in Tanzania, in order to provide reliable drinking water supply.

The dataset is divided into a training set and a testing set, the training set has 59.4k instances and 41 attributes including the label that save in the a separate file, in the case of testing data has 14.85k instances. The data set contains a large amount of information about pumps and water supply systems, including geographical location, pump type, water source type, construction year, pump management and maintenance status, etc. These data features can be used to predict the functional status of water pumps, help decision-makers and water resource managers understand the operational status of water supply systems, and take corresponding measures to improve water supply services. The target variable of the study is status_group that has three possible values functional, non-functional and functional needs repair. As shown in Figure 1, this is the label display diagram corresponding to the training set, which contains pumps in three states: functional, non-functional and functional needs repair.

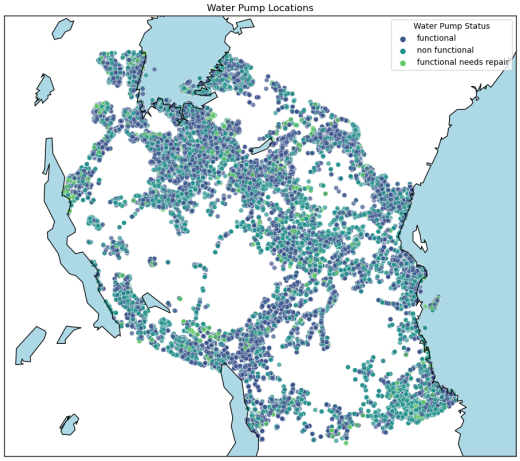

Fig. 1. Water pump locations

B. Research questions

1) How to predict the health of each recorded water point in the data set. (To improve maintenance operations and ensure cleanliness):

- How to ensure that data is accurate, adequate and properly distributed.
- How to judge the impact of features on forecast results and how to select them for prediction.
- How to choose the right machine learning model for prediction.

- How to judge the quality of forecast results.

2) Whether natural factors (Like water quality, source type and so on) have more influence on the condition of a waterpoint or human factors (Like management). (For key decision maker to know they need to pay more attention on natural factors or human factors in the future):

- Which features are natural factors.
- Which features are human factors.
- How to reflect the relationship between a class of eigenvalues and condition of a waterpoint.

In this article, we provide a detailed analysis of this dataset using several different data science and machine learning techniques. Our goal is to identify key characteristics and patterns to predict the function of water pumps to help managers better manage water resources. By understanding the factors that affect pump function, stakeholders can efficiently allocate resources, prioritize maintenance efforts, and ensure sustainable access to clean water for residents. In addition, we will also compare these data processing methods and the performance of models built using machine learning from different aspects, such as the prediction accuracy of training, etc.

The remainder of this paper is organized as follows. Section 2 outlines related work using this dataset for data mining and predicting water pump function. Section 3 introduces the methods we choose for data analysis, preprocessing and data classification and the corresponding results of each method. Section 4 focuses on discussing and comparing the results of different methods with the dataset research results of previous related work. Section 5 presents conclusions and suggestions for future research.

Overall, this paper aims to use a variety of data mining techniques to predict the condition of water supply points, understand the data characteristics that affect the function of water pumps, and provide decision support for key decision makers, so as to contribute to the development of sustainable water resource management strategies and policies.

II. Literature Review

Data mining technology is increasingly being applied in fields such as business and scientific research. It can be predicted and analyzed through historical data analysis and pattern recognition. Therefore, data mining technology has also been applied to water resource management to maximize the utilization of water resources on Earth and ensure the supply of water resources. The following are some studies on using data mining techniques to predict the condition of water pumps in Tanzania.

Darmatasia and Aniati [1] used XGBoost (eXtreme Gradient Boosting) data mining methods to analyze and predict the state of water pumps. At the same time, they also proposed using recursive feature elimination to select important data features to improve the accuracy of the model. They selected 27 optimal features from 39 features in the dataset to train and evaluate the model, and achieved an accuracy of 80.38%.

Karan and L [2] utilize TabNet, a sequential attentive deep neural architecture, for predicting the repair status of water pumps in Tanzania. TabNet utilizes a sequence attention mechanism to model table data, extracting key features through feature selection and automatic learning processes. TabNet also provides model interpretability, explaining the predicted results of the model through attention weights and feature selection. They also use α-Balanced variant of focal loss to address category imbalance issues.

Jacob [3] used classification tree and RandomForest algorithm to create different models, and implemented the all-in-one classification method. In addition, he also established multiple models such as logical regression and the boosted logistic regression for comparison and evaluation. The final result is that the RandomForest algorithm performs best in AUC (area under the curve) and classification rate, which are 0.91 and 0.8209 respectively, and the model can also help to understand why pumps are more likely to belong to a certain category and the relationship between variables.

Indra and Vivek [4] created several models by using SAS® Enterprise Miner, including decision trees, neural networks, multinomial logistic regression and random forest models, to classify and predict whether pumps are in functional state, non-functional state or in need of maintenance. They evaluate the performance of each model by verifying the misclassification rate, and the best model is the RandomForest model.

III. Methodology and Results from each of the stages

In order to carry out effective data mining and analysis on the data set, we started from three aspects: data analysis, data preprocessing and data classification, and we tried at least two methods in each aspect, and each method showed different results. Below, we will introduce the methodology and display the results from these three aspects.

A. Data analysis

In terms of data analysis, we used two analysis methods, namely descriptive statistical analysis and data visualization, by using these methods, we performed data analysis on the data set before and after data imputation.

1) Descriptive statistical analysis (before data imputation): As shown in Fig. 2, first we use the .info command of pandas to count the general information of the original data set, which helps us to observe the features and feature types in the data set more intuitively.

Second, as shown in Fig. 3, we distinguish all features according to their data types, which will benefit classification research for data visualization and subsequent data preprocessing.

Finally, as shown in Fig. 4, we find out all the NaNs in the dataset and find their corresponding feature names.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 59400 entries, 0 to 59399
Data columns (total 41 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     59400 non-null  int64
 1   amount_tsh             59400 non-null  float64
 2   date_recorded          59400 non-null  object
 3   funder                 55765 non-null  object
 4   gps_height             59400 non-null  int64
 5   installer              55745 non-null  object
 6   longitude              59400 non-null  float64
 7   latitude               59400 non-null  float64
 8   wpt_name               59400 non-null  object
 9   num_private            59400 non-null  int64
 10  basin                  59400 non-null  object
 11  subvillage             59029 non-null  object
 12  region                 59400 non-null  object
 13  region_code            59400 non-null  int64
 14  district_code          59400 non-null  int64
 15  lga                    59400 non-null  object
 16  ward                   59400 non-null  object
 17  population             59400 non-null  int64
 18  public_meeting         56066 non-null  object
 19  recorded_by            59400 non-null  object
 20  scheme_management      55523 non-null  object
 21  scheme_name            31234 non-null  object
 22  permit                 56344 non-null  object
 23  construction_year      59400 non-null  int64
 24  extraction_type        59400 non-null  object
 25  extraction_type_group  59400 non-null  object
 26  extraction_type_class  59400 non-null  object
 27  management             59400 non-null  object
 28  management_group       59400 non-null  object
 29  payment                59400 non-null  object
 30  payment_type           59400 non-null  object
 31  water_quality          59400 non-null  object
 32  quality_group          59400 non-null  object
 33  quantity               59400 non-null  object
 34  quantity_group         59400 non-null  object
 35  source                 59400 non-null  object
 36  source_type            59400 non-null  object
 37  source_class           59400 non-null  object
 38  waterpoint_type        59400 non-null  object
 39  waterpoint_type_group  59400 non-null  object
 40  status_group           59400 non-null  object
dtypes: float64(3), int64(7), object(31)
memory usage: 19.0+ MB
```

Fig. 2. General information about the dataset

```
categorical_columns = [c for c in df.columns if df[c].dtype.name == 'object']
numerical_columns   = [c for c in df.columns if df[c].dtype.name != 'object']
print ('Categorical features: ',categorical_columns)
print ('Numerical features: ',numerical_columns)
```

```
Categorical features:  ['date_recorded', 'funder', 'installer', 'wpt_name', 'basin', 'subvillage', 'region', 'lga',
 'ward', 'public_meeting', 'recorded_by', 'scheme_management', 'scheme_name', 'permit', 'extraction_type', 'extraction
_type_group', 'extraction_type_class', 'management', 'management_group', 'payment', 'payment_type', 'water_quality',
 'quality_group', 'quantity', 'quantity_group', 'source', 'source_type', 'source_class', 'waterpoint_type', 'waterpoin
t_type_group', 'status_group']
Numerical features:  ['id', 'amount_tsh', 'gps_height', 'longitude', 'latitude', 'num_private', 'region_code', 'distr
ict_code', 'population', 'construction_year']
```

Fig. 3. Feature types of the dataset

```
values_df.apply(lambda x: sum(x.isnull()))
```

```
id                         0
amount_tsh                 0
date_recorded              0
funder                  3635
gps_height                 0
installer               3655
longitude                  0
latitude                   0
wpt_name                   0
num_private                0
basin                      0
subvillage               371
region                     0
region_code                0
district_code              0
lga                        0
ward                       0
population                 0
public_meeting          3334
recorded_by                0
scheme_management       3877
scheme_name            28166
permit                  3056
construction_year          0
extraction_type            0
extraction_type_group      0
extraction_type_class      0
management                 0
management_group           0
payment                    0
payment_type               0
water_quality              0
quality_group              0
quantity                   0
quantity_group             0
source                     0
source_type                0
source_class               0
waterpoint_type            0
waterpoint_type_group      0
dtype: int64
```

Fig. 4. Check for NaNs

2) Data visualization (before data imputation): Next, we perform visual analysis on some important features of the raw data.

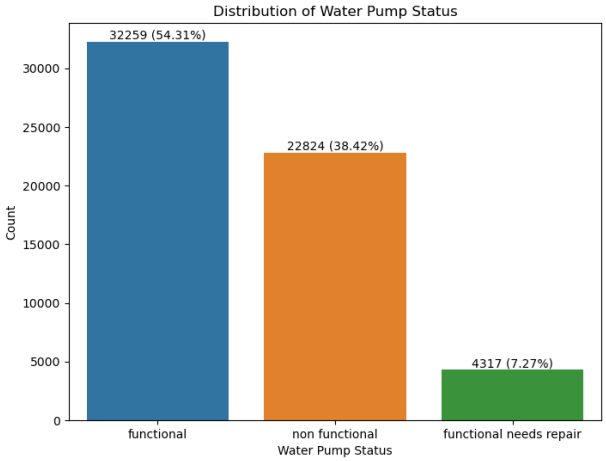

Fig. 5. Distribution of water pump status

In Fig. 5, it can be clearly seen that functional and non functional water pumps account for the majority, with a total of 32259 functional water pumps accounting for 54.31%, and 22824 non-functional water pumps accounting for 38.42%. However, only a small portion of functional water pumps needs repair, with only 4317 pumps accounting for 7.27%. This is a category imbalance situation that may lead to model bias during training and prediction. This may reduce the accuracy and recall rate of the model for rare categories. To solve this problem, we need to adopt some strategies, such as undersampling or oversampling, to balance the number of samples in each category.

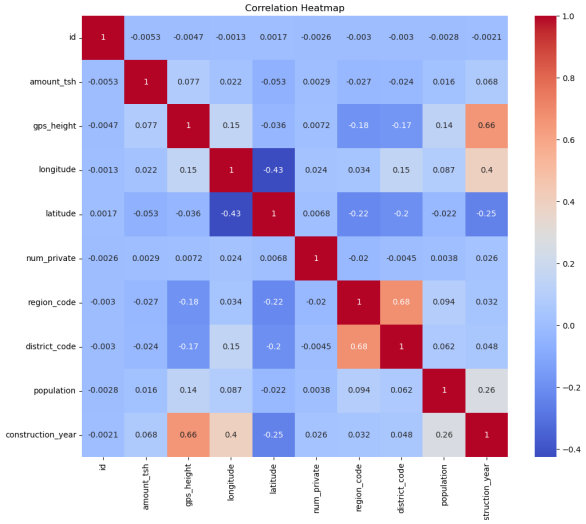

Fig. 6. Correlation heatmap

From Fig. 6, it can be seen that the correlation between data features of numerical types, with darker colors indicating a high degree of correlation between features, such as GPS_ height and construction_ year, longitude and construction_ year. At the same time, it should be noted that there may be multicollinearity between these highly correlated features, which may have an impact on the performance and interpretability of the model. Therefore, these data features need to be identified and further processed. In addition, some features may not have a significant correlation with the state of the water pump, which does not have a strong influence on predicting the state of the water pump. Therefore, these features may not be considered during training and testing.

3) Data visualization (after data imputation): In view of the fact that some NaNs were found in descriptive statistics, and in order to find more data problems and analyze them, we slightly modified the data set, used the mode to imput NaNs, and conducted data analysis for research questions.

In order to study question 1, it is necessary to find which features are more relevant to the target features and select which ones are used for prediction. Therefore, after data interpolation, in order to visualize the characteristics of the object data type, it is necessary to visualize These features were encoded before so that they can be displayed more intuitively.

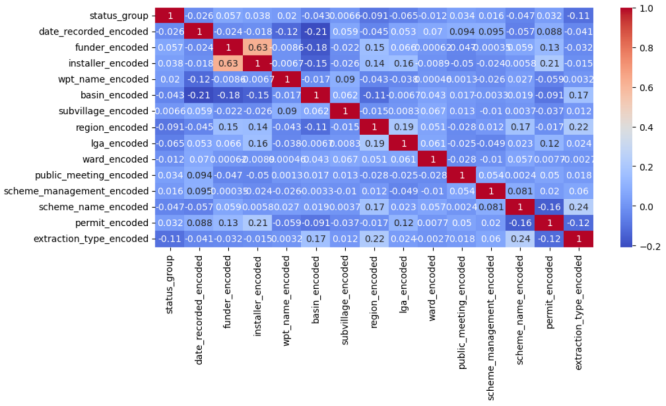

Fig. 7. Heatmap for 'object' features

As shown in Fig. 7, the logic is the same as that in Fig. 6. The darker the color, the higher the correlation between the features. For example, there may be a higher correlation between 'extraction_type' and the status of the water pump.

In order to analyze research question 2, we need to find out the characteristics of nature factors and human factors separately, and try to find out the connection between them through visualization. For example, if we choose 'water quality' as the nature factors and 'management' as the human factors, try to draw a stacked histogram. As shown in Fig. 8, it seems that the better the 'water quality', the more water pumps can be used, and the nature factors may have a greater impact.

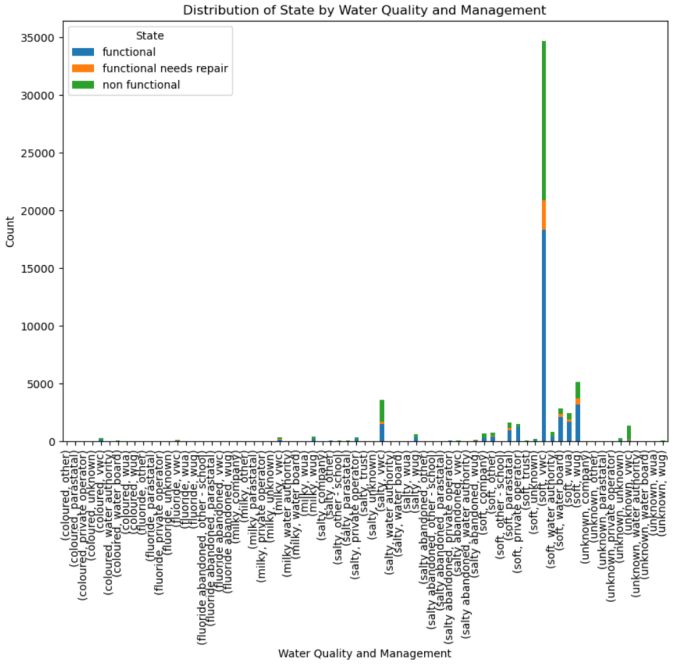

Fig. 8. Distribution of state by water quality and management

In addition, we also want to analyze the correlation between the features with NaN values and the target features. Since there are so many categories, we first use feature engineering to reduce the number of categories before performing data analysis. Take the three characteristics of funders, installers and scheme managements as examples.

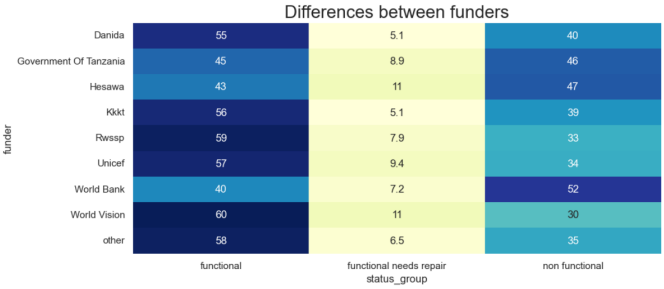

Fig. 9. Differences between funders

As shown in Fig. 9, after reducing the number of categories, the state data distribution of different funders to water pumps can be displayed very well. Darker colors represent more distribution.

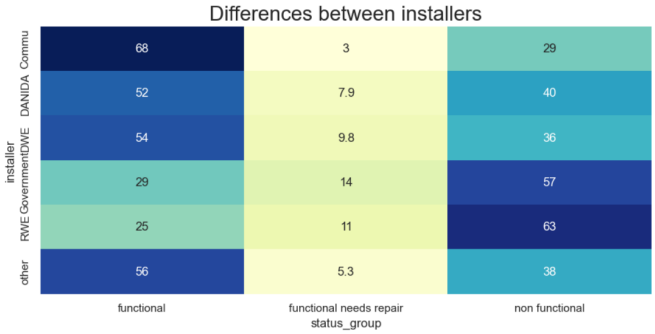

Fig. 10. Differences between installers

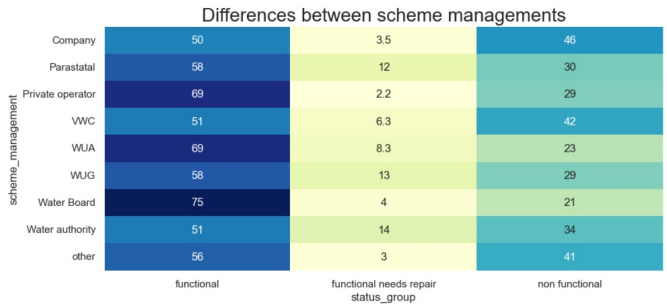

Fig. 11. Differences between scheme managements

Similarly, Fig. 10 and Fig. 11 also show the data distribution of installers and scheme managements in the three states of water pumps, such as 'RWE', which is more related to water pumps that cannot work, and 'Water authority', a management organization that can the water pump used is more relevant.

What needs to be added is that although we have used two methods of data analysis, we can notice that descriptive statistical analysis will focus more on the information of the data itself, which is an effective method for basic information collection, while data visualization is more targeted , which can better reflect the relationship between data and is very flexible.

Therefore, in summary, we finally chose to use these two methods for data analysis at the same time, which can help us to find problems in the original data set while analyzing data relationships and laws, and prepare for subsequent data preprocessing.

B. Data pre-processing

Although some data preprocessing methods have appeared in data analysis, they are only a small part, so this section will introduce the schemes we use in data preprocessing in detail. We tried a total of 4 categories of data preprocessing methods, including data cleaning, data dimensionality reduction, data balancing, and category reduction. These four types of feature engineering methods. Among them, data cleaning includes methods such as missing value processing, feature selection and feature encoding. Data imbalance includes two methods: SMOTE and RandomOverSampler. Data dimensionality reduction uses principal component analysis(PCA), and category reduction uses new classification.

1) Category reduction: Category reduction is generally used to deal with categorical data, especially when dealing with features with a large number of unique classes. It reduces the complexity of this feature, reduces the risk of overfitting, and is easy to visualize and analyze, making the data less sparse. This method was mentioned in the previous data analysis, which is used to study the relationship between some feature columns containing NaN values and the target features. Taking the analysis of 'installers' and water pump status as an example, we found that there are many categories with a count of 1, so we classify the categories with a count of less than 1000 into one category and name them as others. As shown in Fig. 12, this is the data distribution after category reduction processing, which is very clear and solves the problem of data sparseness.

```
df.installer.value_counts()
```

```
other          33202
DWE            21057
Government      1825
RWE             1206
Commu           1060
DANIDA          1050
Name: installer, dtype: int64
```

Fig. 12. The result of category reduction for 'installer'

2) Data cleaning: Because there will always be some errors, duplicates, inconsistencies, or incomplete data in the data set, data cleaning can improve the quality of the data set and improve the performance of subsequent model training. Here, we tried and adopted three methods of missing value handling, feature selection, and feature encoding.

- **Missing value handling**
  Generally, missing data processing methods include deletion, filling, interpolation, etc. We used the filling method in this data set to process missing data. By statistically calculating the content of the feature column with missing data, we can obtain the mode of the feature column, and use the mode to fill the missing data. As shown in Fig. 13, after the mode filling, the number of all missing values in the data set is 0.

```
df.apply(lambda x: sum(x.isnull()))
```

```
id                     0
amount_tsh             0
date_recorded          0
funder                 0
gps_height             0
installer              0
longitude              0
latitude               0
wpt_name               0
num_private            0
basin                  0
subvillage             0
region                 0
region_code            0
district_code          0
lga                    0
ward                   0
population             0
public_meeting         0
recorded_by            0
scheme_management      0
scheme_name            0
permit                 0
construction_year      0
extraction_type        0
extraction_type_group  0
extraction_type_class  0
management             0
management_group       0
payment                0
payment_type           0
water_quality          0
quality_group          0
quantity               0
quantity_group         0
source                 0
source_type            0
source_class           0
waterpoint_type        0
waterpoint_type_group  0
status_group           0
dtype: int64
```

Fig. 13. The result of missing value handling

- **Feature selection**
  Feature selection is a relatively important data processing step and a key step in machine learning. It can effectively reduce training time and improve interpretability and model performance. Generally, it can be performed by embedding or filtering methods. In this data set, in order to reduce the complexity, we adopted a relatively simple method. First, we used artificial screening to delete all the feature columns with the same value, and then analyzed some feature columns that were processed by category reduction, and deleted the feature columns that were consistent with the target Feature columns with little correlation between features. In addition, we also remove some duplicate feature columns and meaningless feature columns from the dataset. As shown in Fig. 14, these are the feature names left after feature selection.

```
df.columns
```

```
Index(['id', 'amount_tsh', 'gps_height', 'installer', 'longitude', 'latitude',
       'wpt_name', 'basin', 'region', 'district_code', 'lga', 'ward',
       'population', 'public_meeting', 'scheme_management', 'permit',
       'construction_year', 'extraction_type', 'extraction_type_group',
       'extraction_type_class', 'management', 'management_group', 'payment',
       'water_quality', 'quality_group', 'quantity', 'source', 'source_class',
       'waterpoint_type_group', 'status_group', 'year_recorded',
       'month_recorded'],
      dtype='object')
```

Fig. 14. The result of missing feature selection

- **Feature encoding**
  By observing the data set after data filling and feature selection, we can find that the format of most feature data is object. In order to facilitate subsequent data classification and model training, we need to use feature encoding to convert the classified data into a format that can be used in machine learning algorithms. Common feature encoding methods include label encoding, one-hot encoding, target encoding, etc. Here, we use label encoding. It can map each category to an integer, for example, 'functional' corresponds to 2, 'functional needs repair': corresponds to 1, 'non functional' corresponds to 0. As shown in Figure 15, it shows the result of label encoding for a column of categorical data.

status_group	...	extraction_type_group_encoded	extraction_type_class_encoded	management_encoded	payment_encoded	water_quality_encoded	management_g
2	...	1	0	7	2	6	
2	...	1	0	11	0	6	
2	...	1	0	7	4	6	
0	...	10	5	7	0	6	
2	...	1	0	1	0	6	
...	...	...	...	...	...	...	
2	...	1	0	9	4	6	
2	...	1	0	7	2	6	
2	...	11	1	7	3	1	
2	...	5	1	7	0	6	
2	...	5	1	7	5	4	

Fig. 15. The result of feature encoding

3) Data dimensionality reduction: Although feature selection has screened out part of the data, there are still many problems in the high-dimensional feature space. Therefore, in order to solve this potential risk, data dimensionality reduction is a more effective method. It can convert high-dimensional feature space into low-dimensional feature space, reduce computational complexity, and allow us to better understand the structure of the data. Common data dimensionality reduction techniques include principal component analysis (PCA) and linear discriminant analysis (LDA). Here, we use PCA for data dimension reduction, which transforms the original data into a set of linearly independent representations of each dimension through linear transformation, which can be used to extract the main feature components of the data. As shown in Fig. 16, it shows the result of transforming the raw data into 20 feature columns.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
0	-2.248530	0.364313	-0.513760	-1.160734	0.389995	-1.173774	0.893205	0.850317	0.268377	-0.975970	0.638933	-0.843587	2.057015	1.862080	0
1	-2.185826	0.270701	2.278308	0.685330	-0.703407	2.508129	0.124704	-0.461203	2.384861	0.826479	0.641434	0.889290	-0.573634	0.900010	-1
2	-1.325088	1.728274	0.049237	0.933570	-0.191985	2.649586	-0.476101	-0.217013	-0.804726	0.079295	0.078274	0.009171	-0.369207	0.783428	-0
3	2.673594	3.542596	-0.778398	-2.509143	0.271701	1.213873	-0.426577	-0.726844	0.459932	-1.590909	-0.048631	-1.474421	-0.058247	2.452509	0
4	-0.888296	-1.003288	-3.656448	3.882854	0.828904	-0.006494	-1.030872	-0.215071	0.994057	1.381373	0.264038	0.523810	-0.753641	1.242179	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
59395	-2.123575	-0.058870	2.096122	-0.076389	0.311889	0.351381	-2.081261	-1.291312	-0.027686	-0.683992	-0.621004	-0.879329	0.514517	-0.653829	0
59396	-2.216839	0.836490	-1.409346	-0.825409	0.957132	1.005106	0.432222	-0.527496	-0.211023	-0.009840	1.079045	0.894273	1.238718	-0.216727	-0
59397	3.559840	-0.088879	-0.894560	-2.094250	0.921640	-0.204952	-0.635733	1.693019	0.101521	0.631417	0.361671	0.198022	1.104707	-0.837751	-1
59398	0.547272	-0.743640	-1.151487	-0.793078	1.469676	-1.139978	-1.037320	-0.561959	1.210913	0.674834	0.069112	0.096703	-0.581309	0.630454	0
59399	1.085860	1.122466	-0.098651	-1.376456	-0.371999	-0.699800	-1.360765	0.031611	-1.325077	1.771913	-1.364944	0.628736	0.270338	0.191768	-0

59400 rows × 20 columns

Fig. 16. The result of PCA

4) Data imbalance: In the above data analysis, by observing Fig. 5, we can find that the number of 'functional needs repair' of the target feature is significantly less than that of the other two feature classes, and the gap is very wide, which often leads to a significant decline in the performance of machine learning. And it will be biased towards the category with a large number of predictions. In order to solve this hidden danger, we need to use some strategies, such as oversampling, undersampling, modifying performance indicators, etc. Here,

we tried two methods for data imbalance processing, they are SMOTE and RandomOverSampler.

- **SMOTE**
  SMOTE is a kind of oversampling technology. Its basic idea is to generate new minority class samples by interpolating minority class samples, and increase the number of minority class samples to achieve class balance. It is generally implemented in two steps. First, for each minority class sample, calculate the distance between it and other minority class samples, and find its k nearest neighbor samples, and then select one of them according to the set oversampling ratio or Multiple nearest neighbor samples, and then for each selected nearest neighbor sample, randomly select a point on the connection line between it and the original sample as a newly generated minority class sample [5].

```
2    32259
0    22824
1     4317
Name: status_group, dtype: int64
2    32259
0    32259
1    32259
Name: status_group, dtype: int64
2    22535
0    16021
1     3024
Name: status_group, dtype: int64
0    22535
2    22535
1    22535
Name: status_group, dtype: int64
SMOTE
```

Fig. 17. The result of SMOTE

  As shown in Fig. 17, it shows the number of different classes of target features after SMOTE processing on the training set. Obviously, the numbers of the three classes are all filled to the same number.
- **RandomOverSampler**
  RandomOverSampler is also a kind of oversampling, which increases the number of samples by randomly copying the samples of the minority class, so as to achieve class balance. As shown in Fig. 18, it shows the results of data imbalance processing using RandomOverSampler, and the number of completions for each category is the same as the previous method.

```
2    32259
0    22824
1     4317
Name: status_group, dtype: int64
2    32259
0    32259
1    32259
Name: status_group, dtype: int64
2    22535
0    16021
1     3024
Name: status_group, dtype: int64
0    22535
2    22535
1    22535
Name: status_group, dtype: int64
RandomOverSampler
```

Fig. 18. The result of RandomOverSampler

  It should be noted that random oversampling may increase the risk of overfitting of the model, because it increases the number of minority class samples by duplicating them, which may cause the model to be overly sensitive to these duplicated samples [6]. Therefore, this data set does not consider the use of RandomOverSampler for data imbalance processing.

So far, all the data preprocessing methods we have tried have been introduced. Because they need to be divided into two groups for comparison, we will divide the above data preprocessing methods into two categories. The first category includes category reduction, data cleaning and Data dimensionality reduction (PCA), the second category includes category reduction, data cleaning and data balancing (SMOTE). Due to the relationship of the original data set, both methods need to include category reduction and data cleaning.

What needs to be added is that for the accuracy of the model, we finally integrated and used four steps of category reduction, data cleaning, data dimensionality reduction (PCA) and data balance (SMOTE) to preprocess the data, and obtained the highest prediction accuracy.

C. Data classification

In order to classify and predict the preprocessed dataset, we need to select an appropriate model for training and performance evaluation. Here, we have selected two popular models for training, they are DecisionTree and XGBoost, which just correspond to the training set after the preprocessing of the previously divided two types of data, and then carry out model training and evaluation.

1) DecisionTree: For the model training of the DecisionTree, we choose the first type of preprocessed data set, that is, the data set processed by category reduction, data cleaning and Data dimensionality reduction (PCA).

5.2 DecisionTree classifier is evaluated for performance with the above dataset.

```
clf_dt = DecisionTreeClassifier(max_depth = 16)
dt_scores = cross_val_score(clf_dt, X_d_train, y_d_train, cv=5, scoring='accuracy')
print(dt_scores.mean(), '+/-', dt_scores.std())
```

```
0.6801346801346801 +/- 0.0037759303675102287
```

Fig. 19. Accuracy of DecisionTree in the first type of preprocessed data set

As shown in Fig. 19, the accuracy rate of the model is about 68%, which is not high.

2) XGBoost: For the model training of the XGBoost, we choose the second type of preprocessed data set, that is, the data set processed by category reduction, data cleaning and data balancing (SMOTE).

5.1 XGBoost classifier is evaluated for performance with the above dataset.

```
clf_xg = XGBClassifier(objective = 'multi:softmax', booster = 'gbtree',
                       num_class = 3, eval_metric = 'merror', eta = .1,
                       max_depth = 16, colsample_bytree = .4, n_jobs = -1)
xg_scores = cross_val_score(clf_xg, X_train_resampled, y_train_resampled, cv=5, scoring='accuracy')
print(xg_scores.mean(), '+/-', xg_scores.std())
```

```
0.8333407292360032 +/- 0.006681652822433565
```

Fig. 20. Accuracy of decision tree in the second type of preprocessed data set

As shown in Fig. 20, the accuracy rate of this model is about 83.3%, which is relatively high.

In addition, in order to have a high accuracy rate, we have gone through various data preprocessing and model combinations and tests, and finally found the best performance match. The data preprocessing includes category reduction, data cleaning, data dimensionality reduction (PCA) and data balance (SMOTE) four steps, the model finally chose XGBoost.

5.1 XGBoost classifier is evaluated for performance with the above dataset.

```
clf_xg = XGBClassifier(objective = 'multi:softmax', booster = 'gbtree',
                       num_class = 3, eval_metric = 'merror', eta = .1,
                       max_depth = 16, colsample_bytree = .4, n_jobs = -1)
xg_scores = cross_val_score(clf_xg, X_d_train_resampled, y_d_train_resampled, cv=5, scoring='accuracy')
print(xg_scores.mean(), '+/-', xg_scores.std())
```

```
0.8418312255010724 +/- 0.015580595522714723
```

Fig. 21. Final accuracy

As shown in Fig. 21, the final accuracy of the model is as high as 84.1%.

```
Accuracy Score:  0.9700318023814807
```

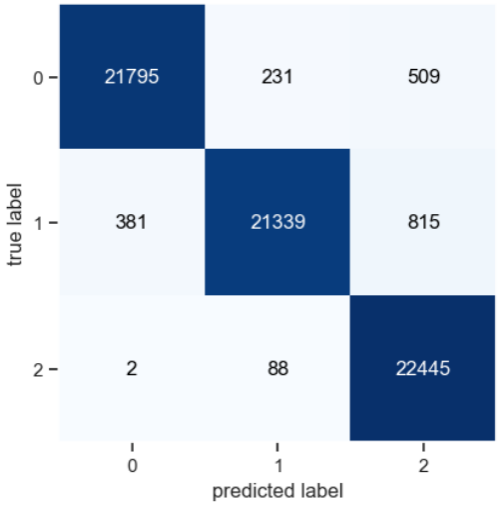

Fig. 22. Confusion matrix image

Fig. 22 shows the performance of the model on the oversampled training dataset after training with appropriate XGBoost parameters. It can be seen that the accuracy rate is very high.

Finally, we applied the model to the test data set, and the final classification prediction results are shown in Fig. 23.

```
functional                8600
non functional            5982
functional needs repair    268
dtype: int64
```

Fig. 23. Prediction on test set

IV. Discussion

A. Comparing and critiquing (partners in pair)

As mentioned above, the first solution is to use the first type of preprocessed data set to train the DecisionTree model, and its final accuracy rate is 68%; the second solution is to use the second type of preprocessed data set to train XGBoost model, its final accuracy rate is 83.3%.

Obviously, the accuracy rate of option 1 is very low, and the accuracy rate of option 2 is higher. The reason for the low accuracy of Scheme 1 may be that the DecisionTree model itself has poor performance in this data set, and data imbalance and PCA have negatively affected the data set. Although the accuracy of Scheme 2 is higher, there are still cases of overfitting after data imbalance processing, and the feature dimension is high, and the calculation is complicated and takes a long time.

B. Comparing and critiquing (previous research)

In this section, we use the scheme with the highest final accuracy for comparison with previous studies.

Darmatasia and Aniati also used XGBoost to create a model in their study, but the accuracy of their model was only 80.38%, lower than our experimental result of 84%, which

may be because they did not consider the imbalance of the categories of data sets when processing the data. This may lead to overfitting problems and deviations in the predicted results of the model [1].

Karan and L's TabNet model also obtained good prediction results, with an accuracy of 83.6%, which may be attributed to their substitution strategies for some columns with category skew. TabNet can also learn nonlinearities in the data and avoid the problem of over-fitting. At the same time, TabNet uses sequential attention to select prominent features in each decision step, which contributes to the interpretability of the model, which is not involved in our study [2].

In Jacob's study, a random forest model with an accuracy of 82.09% was created. During the study, he conducted correlation assessment for each single feature, and conducted correlation statistics by chi-square test or analysis of variance test according to the type of variable. This is quite sufficient for the data analysis and selection, to ensure the reliability of the data, for the training model is very helpful. At the same time, this is what we need to improve in our research work [3].

Among the models created by Indra and Vivek, the one that achieved the best results was also the RandomForest model, but neither the accuracy of the model nor the specific data processing process were given, so there may be insufficient data processing, so the trained model is not reliable and persuasive enough [4].

V. Conclusions and recommendation for future research

This study uses appropriate data analysis, data preprocessing, and data classification methods to process and select features to ensure the integrity and reliability of the data set, and selects the most relevant features to construct prediction models, thereby ensuring that the trained models can obtain reliable results during prediction.

We use different data mining techniques to construct models to predict the operating status of water supply points, using methods such as Decision tree and XGBoost. In addition, the performance of two different models was compared, such as the time required for training and the accuracy of prediction results. The final experimental results indicate that the prediction accuracy of the XGBoost model is higher than that of the decision tree.

In addition, we also compared and analyzed the impact of natural factors (such as water quality, water source types, geographical location) and human factors (such as management, management level) on the status of waterpoints, to understand the impact of these two factors on water supply point conditions, and provide important reference information for decision-makers to improve the effect of water supply management and maintenance and operation.

In summary, both of the prediction models we have constructed can predict the state of water pumps, but the XGBoost model has better performance and higher accuracy. Our analysis of data can also provide managers with some reference information, allowing them to understand which factors have a significant impact on the status of water pumps, and to pay more attention in future maintenance and management.

References

[1] Arymurthy, A.M., 2016, October. Predicting the status of water pumps using data mining approach. In 2016 International Workshop on Big Data and Information Security (IWBIS) (pp. 57-64). IEEE.

[2] Pathak, K. and Shalini, L., 2023. Pump It Up: Predict Water Pump Status using Attentive Tabular Learning. arXiv preprint arXiv:2304.03969.

[3] Benoot, J., PREDICTING THE FUNCTIONAL STATE OF TANZANIAN WATER PUMPS.

[4] Chowdavarapu, I.K. and Manikandan, V.D., 2016. Data Mining the Water Pumps: Determining the functionality of Water Pumps in Tanzania using SAS Enterprise Miner. SAS South Central User Group Forum.

[5] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.

[6] Lemaitre, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. Journal of Machine Learning Research, 18(17), 1-5.