**PROJECT REPORT**

**PUMP IT UP: DATAMINING THE WATER TABLE**

**Team**

**Gayatri Balakumar(gxb170030)**

**Mohana Prudvi Dongala(mxd165330)**

**Sowmya Bhupathiraju(sxb162130)**

**Teja Kiran Chunduri(txc163430)**

**1. INTRODUCTION**

As a part of our project we have taken part in the driven data competition: “Pump it Up: Data Mining the Water Table”. For this challenge we have to develop a classifier that can classify various water points of Tanzania water resources as “Functional”, “Non-Functional” and “Functional needs repair”. These values should be predicted upon the large array of attributes provided to us in the dataset. Then we have developed various classifiers and submitted an .csv file for the test values that have been provided. The motto of the classification is to know the understanding and maintenance of the water points to provide better service across the communities of Tanzania.

**2. PROBLEM DESCRIPTION**

The learner we are developing for the project is a part of the competition Pump it up –Driven Data challenge. The objective of this challenge is to predict the working condition of the water point whether it is functional, functional needs repair or non-functional. The data for this competition comes from the “Tariff water point’s dashboard”, which aggregates data from the Tanzania Ministry of Water.

They have provided with

* Training set values -The variables for the training set
* Training set labels-The class label for each variable in training set
* Test set values- The variables that need predictions

The problem definition is to find out whether or not a water pipe if functional or needs a repair or not functioning

**3. DATASET DESCRIPTION**

The training data set given for the given competition had 59400 training values where each has one ID and class label.

* Total features : 39 attributes and 1 class label
* Total number of training instances: 59400
* Total number of testing instances: 14850

**3.1 Type of Attributes**

* Number of categorical attributes: 31
* Number of numeric attributes: 7
* Number of data attributes: 2

Fig 3.1. Types of Attributes

**Attributes Description**

|  |  |
| --- | --- |
| **Name of the attribute** | **Attribute description** |
| Amount\_tsh | The amount of total static head |
| Date\_recorded | The date when the data point was entered |
| Funder | The organization funding the well |
| Gps\_height | Altitude of the well |
| Installer | The organization which installed the well |
| Latitude | GPS coordinate of the well |
| Longitude | GPS coordinate of the well |
| Wpt\_name | Name of the water point if present else null |
| Num\_private |  |
| Basin | Geographic location of the basin |
| Sub\_village | Geographic location i.e., name of village |
| Region | Geographic location i.e., region name |
| Region\_code | Geographic location i.e., zip code of region |
| District\_code | Geographic location i.e., unique region code |
| Lga | Geographic location |
| Ward | Geographic location |
| Population | Population around the water point |
| Public\_meeting | True /False |
| Recorded\_by | Group which entered this row of data |
| Scheme\_management | Organization that operates the water point |
| Scheme\_name | Name of organization that operates the water point |
| Permit | Whether the water point is permitted |
| Construction\_year | Year of water point construction |
| Extraction\_type | Extraction method used by water point |
| Extraction\_type\_group | Group of Extraction method used by waterpoint |
| Extraction\_type\_class | The kind of extraction the waterpoint uses |
| Management | In what manner the waterpoint is managed |
| Management\_group | In what manner the waterpoint is managed |
| Payment | Cost of the water |
| Payment\_type | Cost of the water |
| Water\_quality | Quality of the water |
| Quality\_group | Quality of the water |
| Quantity | Quantity of the water |
| Quantity\_group | The quantity of water |
| Source | The source of waterpoint |
| Source\_type | The source of waterpoint |
| Source\_class | The source of waterpoint |
| Waterpoint\_type | The kind of waterpoint |
| Waterpoint\_type\_group | The kind of waterpoint |

**3.2 Class Labels**

The Class labels in this dataset:

* functional - the water point is functional and no repairs needed
* functional needs repair - the water point is functional, but repairs needed
* non-functional - the water point is not functional

**Class Labels Distribution**

* Number of functional labels: 32259
* Number of functional needs repair: 4317
* Number of nonfunctional: 22824

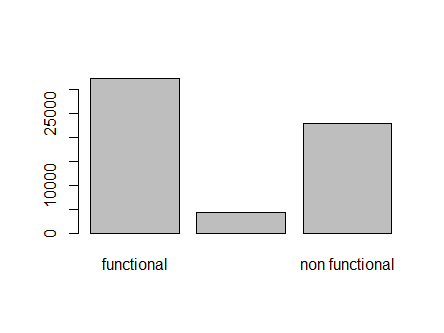


Fig 3.2 Distribution of labels

**4. PRE-PROCESSING TECHNIQUES**

**4.1 ATTRIBUTE ANALYSIS**

The raw data may contain lot of noise that is, it may contain number of attributes which may not contribute to classification or it may mislead the learner. So we checked each and every attribute to know how much important it is and how it effects the learner’s accuracy .We performed a number of operations like calculating correlation ,plotting histograms against attributes and class labels ,checking their null values percentage etc.

**4.1.1 Eliminating attributes**

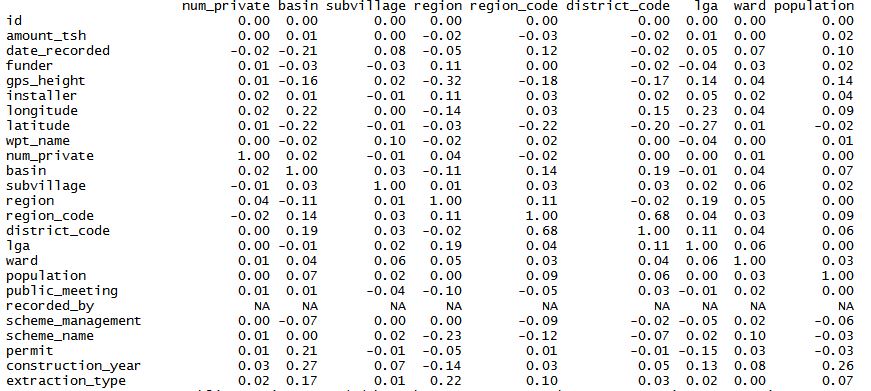
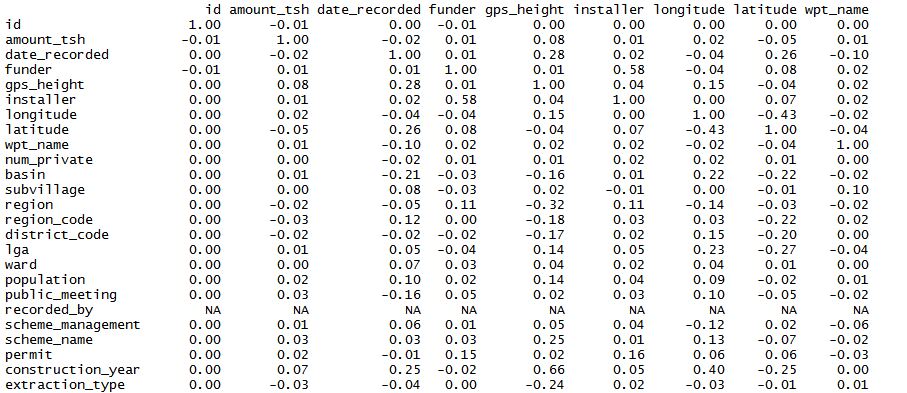
**Calculating correlations**

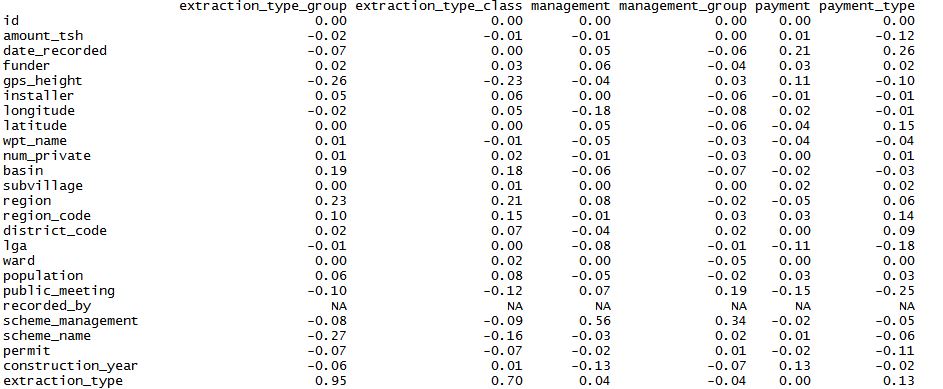
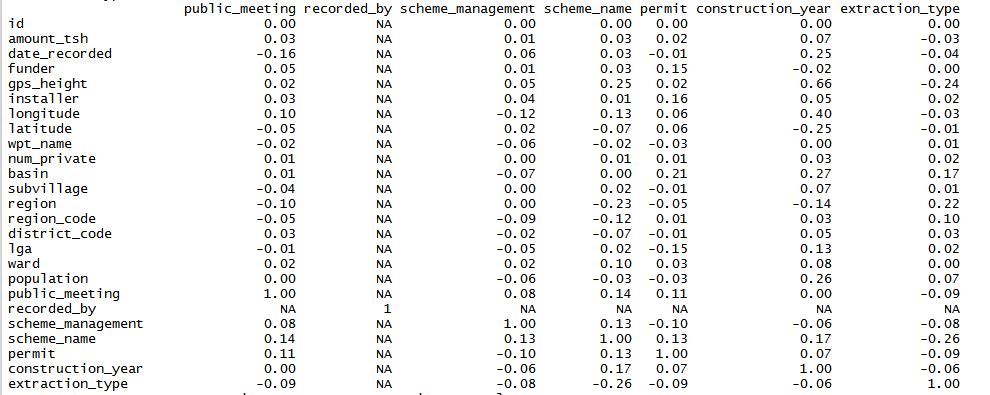
A correlation matrix is constructed for all the attributes to check out the correlations .It is huge 40/40 matrix. From the correlation matrix we tried to observe the correlation with the class label as well as the correlation with the other attributes.

We observed the following:

* Checking with the class labels
  + - Good correlation with the class labels :Theses attributes help in classification
    - Low correlation with the class labels: Further check the reason behind low correlation
* Checking amongst other attributes
  + - Redundant attributes

By carefully checking the correlations values obtained, we began by assuming a threshold of 0.8. The attribute pairs which have more than 0.8 are further checked to see which one of them is more useful. We analyzed the raw data to find out the reason behind the high correlation. In case the attributes were way too similar .i.e., their contribution to classification is same (from values and their distribution) we concluded such attributes as redundant ones.





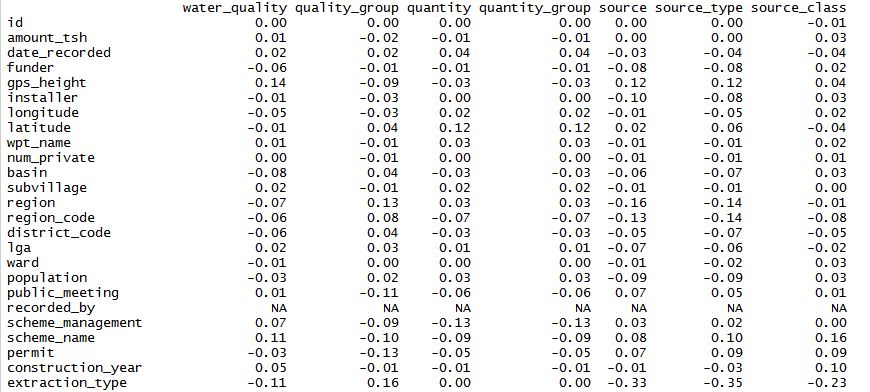


Fig:4.1 Correlation matrices

To conclude upon which ones to retain, we used variable importance on different classifiers like random forests and chose the better one. We started from the least ranked attributes and followed the following steps to see whether to retain or remove

Fig :4.2 Variable importance of attributes on random forests

**Checking for NULL values**

We checked for the number of NULL values in the two attributes, the one which has most number of NULL values is removed. If we could not relate by checking NULL values, then we have further processed to check their factoring levels.

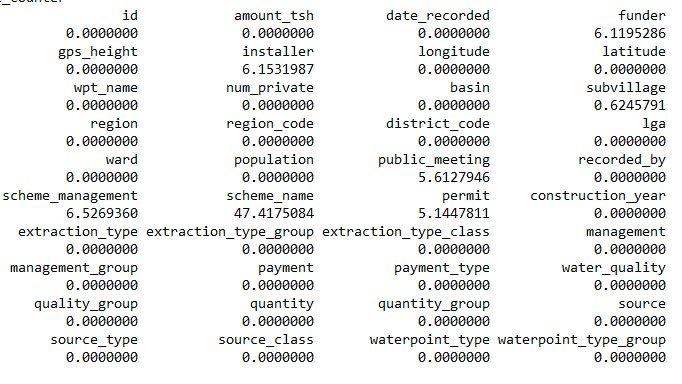


Fig 4.3 Null value percentages for each Attributes

**Plotting Histograms**

We have plotted histograms and checked for the number of levels. If they have a considerable distribution on levels then the attribute which does not have proper distribution is removed.

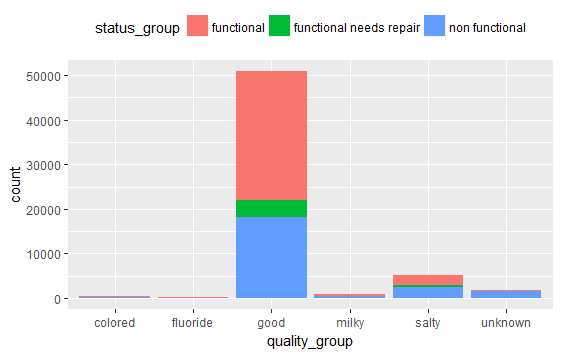


Fig: 4.4 Histogram plotted for quality\_group

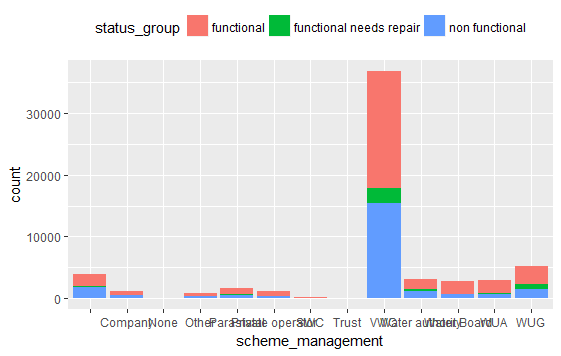
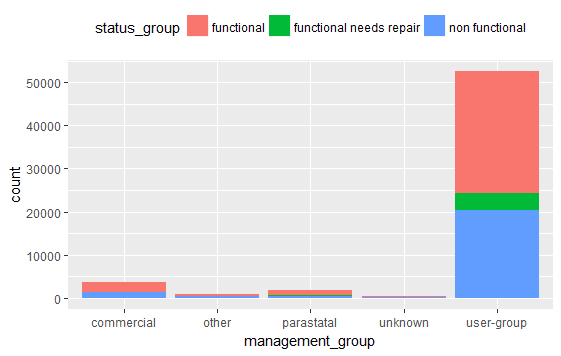


Fig:4.5 histograms plotted for management\_group and schema\_management

We have seen by plotting histogram for quality group that most of the rows value is “good “,so this does not contribute much for classification. The same with the other two attributes also.

**Another conclusions obtained by plotting the histograms**

By plotting histograms we also observed many of the attributes with too many levels. Then we had to see the usefulness of these attributes before disregarding them completely because we can handle such attributes by bucketing them. Few of them were

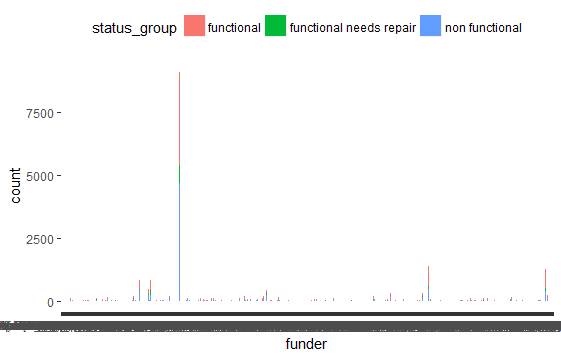


Fig:4.6 histograms plotted for funder

There were cases where the attributes had too many levels and did not have much significance with the class label such as:

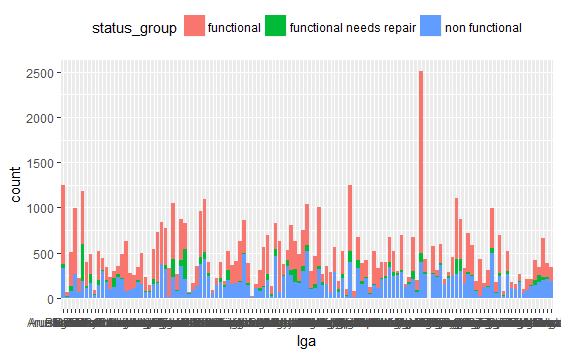


Fig: 4.7 histograms plotted for lga

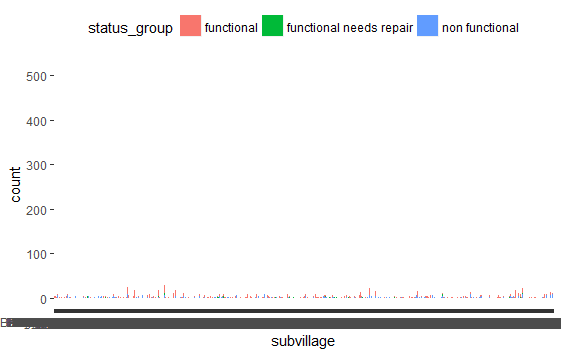


Fig 4.8 Histogram of sub village

**Impact of removing attributes on classifier (random forests)**

For impact on classifier we proceeded by running on random forest to cross verify the results and decide on few uncertain attributes. For those, we incrementally removed attributes and checked the impact on accuracy by retaining and removing them. This was performed by running experiments on a classifier (random forest) by providing attributes on all confusing combinations and recording the accuracy values. Doing this also helped us confirm on equivalent data attributes.

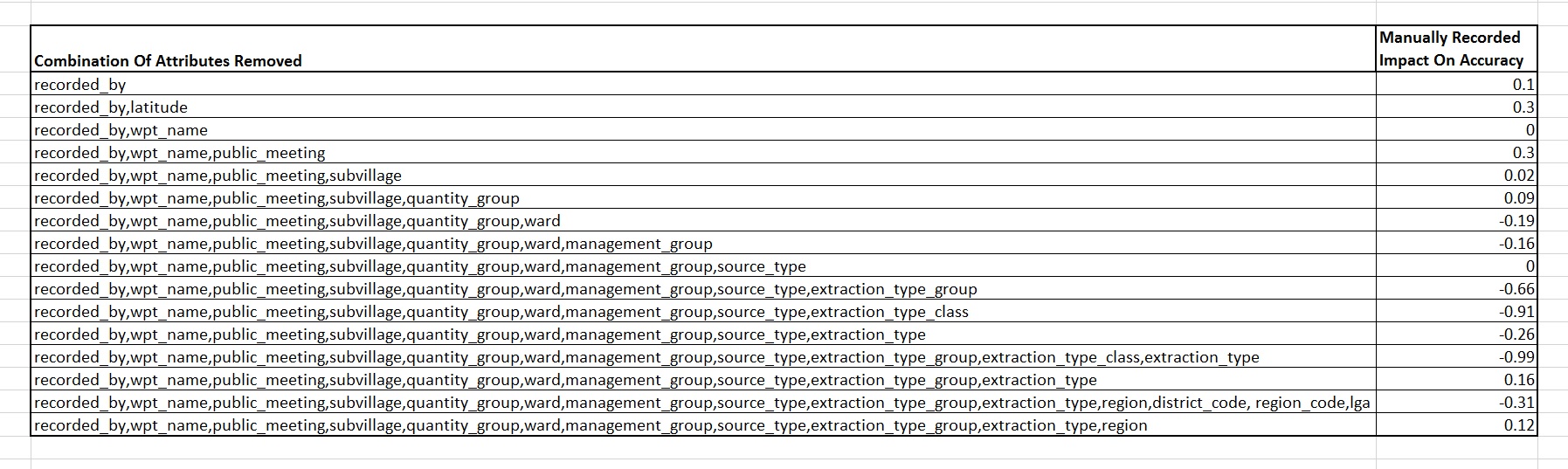


Fig 4.9 Impact of Accuracy on classifier when adding and removing the attributes

After the above procedures, the following attributes are removed.

|  |  |  |
| --- | --- | --- |
| **index** | **Attribute** | **Reason for removal** |
| 1 | quantity\_goup | redundant  high correlation with quantity |
| 2 | extraction\_type\_group | redundant  high correlation with extraction\_type |
| 3 | source\_type | redundant  high correlation with source |
| 4 | payment\_type | redundant  high correlation with payment |
| 5 | quality\_group | less variable importance  most of the rows have one value  effecting accuracy negatively |
| 6 | waterpoint\_type\_group | redundant  high correlation with waterpoint\_type |
| 7 | scheme\_name | high NULL value percentage |
| 8 | recorded\_by | same value in entire data |
| 9 | num\_private | very less variable importance  no significance  more number of NULL values |
| 10 | sub\_village | too many levels  less variable importance |
| 11 | district\_code | too many levels  less variable importance |
| 12 | wpt\_name | less variable importance |
| 13 | extraction\_type | less consistent and similar data compared to extraction\_type\_class |
| 14 | Lga | too many levels |
| 15 | region\_code | similar to region |
| 16 | management\_group | similar to management |
| 17 | Installer | similar to funder |
| 18 | scheme\_management | no significance |
| 19 | Ward | Similar to subvillage |
| 20 | Date\_recorded | Modified to operation\_age |

**4.1.2 Modification of existing attributes**

**Operations Age**

Using date recorded and construction year. We haves seen that date recorded attribute have numerous level and construction years are mostly around 2000. To make these attributes useful we have constructed a new attribute called operationyear. This attribute gives the operation duration of water point in years. This is the subtraction of year from the date recorded and the construction year. This attribute now makes sense as we are taking in regard the  
feature which describes how old the water point is. This may help us  
to know about the working condition of the water point.

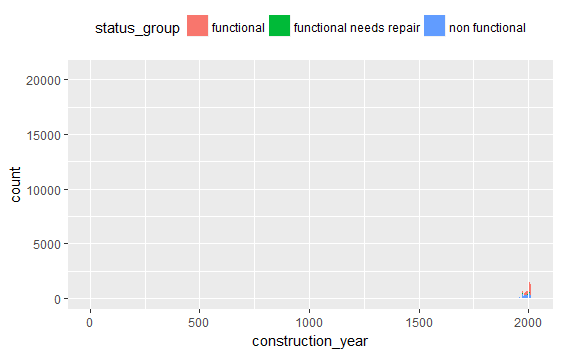
****

Fig 4.10 histogram of construction year showing all concentrated around 2000

**Funder**

The funder attribute is the name of the organization funding the water point. The data of funder has many human errors. Mostly punctuations and spelling mistakes. If these are not handled the number of levels is at 1898. So we have modifies the funder column and then factored it to 11 levels

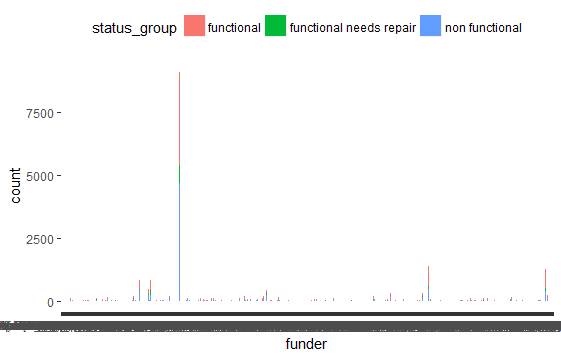
****

Fig 4.11 Histogram of Funder

Finally, after the attribute we considered the following attributes.

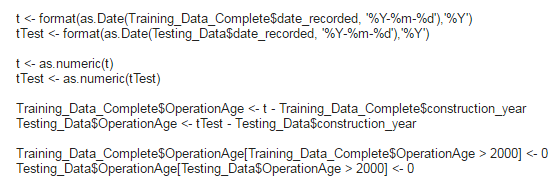
|  |  |
| --- | --- |
| **Index** | **Attributes** |
| 1 | Amount\_tsh |
| 2 | Funder |
| 3 | Gps\_height |
| 4 | Longitutude |
| 5 | Latitude |
| 6 | Basin |
| 7 | Region |
| 8 | Population |
| 9 | Public\_meeting |
| 10 | Permit |
| 11 | Extraction\_type\_class |
| 12 | Management |
| 13 | Payment |
| 14 | Water\_quality |
| 15 | Quantity |
| 16 | Source |
| 17 | Source\_class |
| 18 | Waterpoint\_type |
| 19 | Operation\_age |

**4.2 Pseudo code**

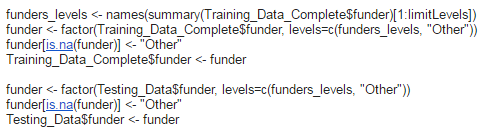
* Firstly we have removed the attributes that we selected keeping those we want to change.

****

* We have changed the attribute construction year and date recorded into a new attribute OperationAge using the year part of the attributes. Then removed the attributes construction year and date\_recorded

****

* On the column Funder: This column has too many levels and errors. TO handle them we identified similar values for this column we have firstly changed the case to lower case then removed the spaces in the values, lastly trimmed the values into first 3 characters in R . Also we have considered row values with garbage such as“.”,””,””,” ”,”-“,”\_” as others. This way we are handling even null values here.  Now the levels have reduced to around 400 .Further we have bucketed these levels to 11.

****

* Changed gps\_levels by bucketing similar to that done on funder column.
* Then we have run the classifier on the dataset. Few changes like making a data matrix for xgboost have been done according to the classifier functions and library used.

**5. PROPOSED SOLUTIONS AND METHODS**

Following Methods and classifiers have been used to check the best possible classifiers for the given problem. The classifiers which gave better accuracy results were further considered and worked with the parameter tuning in the next session

**Decision trees**

Decision trees are flexible .They are very good with categorical values. It is comparatively faster than the neural nets and deep learning techniques. As our task was classification we tried on decision trees and they gave considerate results

R package used: rpart

**Random forests**

Random forests are ensemble methods for classification and regression problems. The base classifiers used are decision trees. They construct multiple decision trees at the training time and gives the node of the trees as output in the testing time

R package used: randomforest

**Bagging**

Bootstrap aggregating or bagging is an ensemble methods .This method helps to avoid over fitting. Bootstrapping might not have a problem with noisy data. It does not work with stable methods.

R package used: adaBag, Ipred

**Naïve Bayes**

Naïve bayes are simply probabilistic classifiers based on the Bayes theorem with strong independence assumptions. It is very fast because it has no model building.

R package used: mass

**Deeplearning**

Deep learning is a new machine learning technique. These cascade to many layers of non-processing units for feature extraction and transformation

Rpackage used: h20

**Xtreme Gradientboosting**

Gradient Boosting is a machine learning technique which produces a prediction model in the form of weak prediction models .Generally decision trees are used as the classifiers. This is a very powerful technique.

Rpackage used: xgboost

The following table gives the initial accuracy which we tested on different classifiers

|  |  |
| --- | --- |
| **Classifier** | **Average Accuracy** |
| Decision tree | 76.517 |
| Gradient Boosting | 80.71 |
| Random Forests | 76.05 |
| Deep Learning | 68.185 |
| Naïve Bayes | 20 |
| Bagging | 54.34 |

Fig 5.1 Average accuracies for classifiers

**6. EXPERIMENTALRESULTS AND ANALYSIS**

**CLASSIFIER ANALYSIS AND PARAMETRE TUNING**

Once we are done with the attribute selection and removal, we need to choose a learner to classify. We used Deep Learning techniques, Gradient Boosting, Decision Trees, and Random Forests

**GRADIENT BOOSTING**

XGBoost is one of the classifiers implementing .Ensemble methods and has proven to has better performance on various datsets.We’ve used “xgboost” library for implementing the xgboost algorithm, that includes a linear model and a tree learning algorithm. The following parameters are considered while implementing the “xgboost” method.

General Parameters

nthread = number of threads to be run in parallel

Tree Parameters

nfold = number of validation folds

max\_depth = maximum depth of the trees

eta = step size to be shrinked after each boosting step

min\_child\_weight = minimum sum of instance weight in a child node( to stop further partition of the node)

subsample = sample ratio of training data

colsample\_bytree = sample ratio of columns needed for tree construction

nrounds = number of passes on data

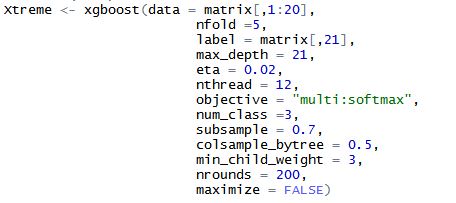
Learning Parameters

objective = learning task (Binary Classification or multiclass class classification)

num\_class = number of classes

The following parameter values are fixed as these gives the best results

nthread=12,subsample=0.7,min\_child\_weight=3,colsample\_bytree=0.5



Different runs for the Gradient Boosting

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No | max\_depth | Eta | nrounds | Training Accuracy | Validation Accuracy |
| 1 | 20 | 0.2 | 400 | 100 | 80.54 |
| 2 | 15 | 0.2 | 400 | 100 | 80.257 |
| 3 | 21 | 0.2 | 400 | 100 | 80.31 |
| 4 | 21 | 0.5 | 350 | 100 | 79.57 |
| 5 | 21 | 0.1 | 250 | 99.99 | 80.74 |
| **6** | **21** | **0.02** | **200** | **93.833** | **81.63** |
| 7 | 21 | 0.02 | 150 | 93.57 | 81.97 |

By implementing the above tuned parameters on the testing data, we got highest accuracy

for the parameters of the 6th run, which is **81.7**

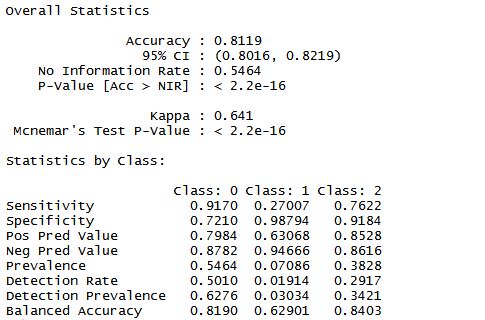


Fig 6.1 Confusion matrix for the best run

Along with the accuracy, we have also considered other performance metrics from the confusion matrix such as precision, recall and ppv and npv

Fig 6.2 Graph comparing accuracy with the parameters

**DECISION TREES**

We have used rpart package for building the decision tree. The parameters tuned for improving with accuracy of the decision tree were:

* Minsplit- It is the minimum number of observations per leaf node.
* Cp- This controls the size of the decision tree by deciding the number of splits.
* Maxdepth- The maximum depth of the decision tree.

Different runs for the decision tree:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #run number | minsplit | cp | maxdepth | Accuracy |
| 1. | 10 | 0.0001 | default | 78.0812 |
| 2. | 10 | 0.0001 | 5 | 70.83123 |
| 3. | 10 | 0.0001 | 10 | 74.841 |
| 4. | 10 | 0.0001 | 20 | 77.963 |
| 5. | 100 | 0.0001 | 20 | 76.36 |
| 6. | 5 | 0.0001 | default | 78.08 |
| 7. | 5 | 0.00005 | default | 78.2835 |
| **8.** | **5** | **0.000075** | **30** | **78.5** |
| 9. | 2 | 0.00001 | 30 | 75.46 |
| 10. | 20 | 0.00001 | 30 | 77.255 |
| 11. | 25 | 0.00001 | 30 | 77.00 |
| 12 | 25 | 0.00005 | 30 | 77.524 |

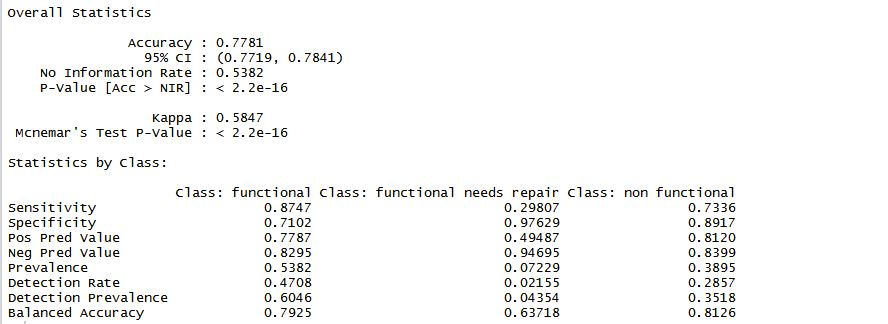
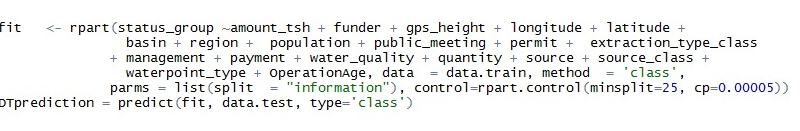


Fig 6.3 Confusion Matrix for best run

Along with the accuracy ,we have also considered other performance metrics from the confusion matrix such as precision,recall and ppv and npv

The highest accuracy on training data set we achieved is 78.5. We have decided these as our best set of parameters. We have then populated the submission file for this and achieved an accuracy of **78.09** on submission.

Fig 6.4 Graph comparing accuracy with the parameters



**RANDOM FORESTS**

Random Forest

The parameters tuned for improving with accuracy of the Random Forest were:

Parameters:

* mtry- the number of variables randomly selected at each split
* n-tree- number of trees to grow

Different runs for the Random Forest

|  |  |  |  |
| --- | --- | --- | --- |
| #no of run | mtry | n-tree | Accuracy |
| 1 | 10 | 1 | 74.08 |
| 2 | 15 | 1 | 73.75 |
| 3 | 5 | 1 | 74.01 |
| 4 | 5 | 2 | 72.01 |
| 5 | 10 | 10 | 79.32 |
| 6 | 10 | 15 | 79.55 |
| **7** | **10** | **50** | **79.65** |

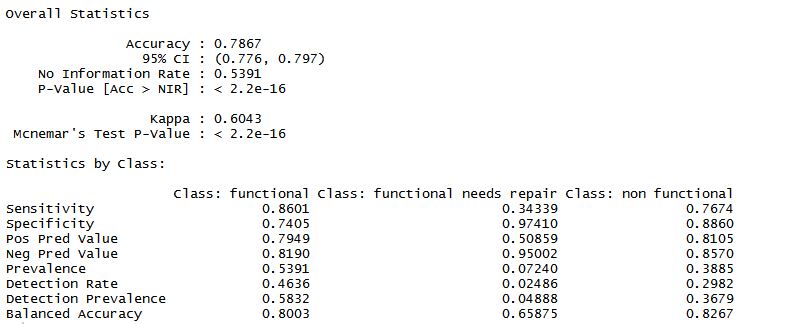


Fig 6.5 Confusion Matrix for the best run

Along with the accuracy ,we have also considered other performance metrics from the confusion matrix such as precision,recall and ppv and npv

The highest accuracy on training data set we achieved is 78.5. We have decided these as our best set of parameters. We have then populated the submission file for this and achieved an accuracy of **78.09** on submission.

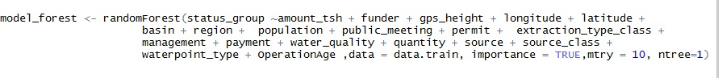
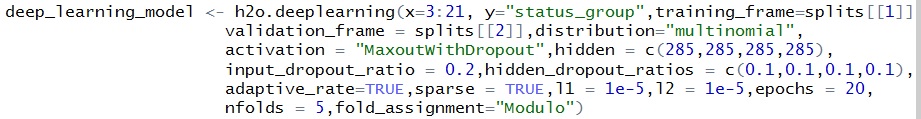


Fig 6.6 Graph comparing accuracy with the parameters

**DEEP LEARNING**

We used H20 package to build a multi-layer, feed forward, back-propagating neural network as our deep learning model. Below are the parameters we considered for building and tuning the model.

* Distribution – determines the type of predictions and the output nodes required.
* Activation Function – Weighted combination of input signals are activated through this function.
* Hidden – defines how many hidden layers to be built and the sets the number of neurons for each layer. This accounts to the complexity of the model.
* Epochs – Specifies how many times the network iterates thorough the dataset.
* Adaptive rate : Boolean attributes that is set to true or false for enabling adaptive learning rate.



Different runs for the Deep Learning

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| #no of run | Distribution | Activation Function | No. of hidden layers | No. of nodes in hidden layer | epochs | accuracy | Time taken  (min) |
| 1. | Quantile | RectifierWithDropout | 3 | 250 | 50 | 67.01 | 130 |
| 2. | Quantile | MaxoutWIthDropout | 4 | 220 | 100 | 70.6 | 190 |
| 3. | Quantile | MaxoutWIthDropout | 5 | 200 | 100 | 63.12 | 240 |
| 4. | Multinomial | MaxoutWIthDropout | 4 | 175 | 20 | 71.8 | 62 |
| 5. | Multinomial | MaxoutWIthDropout | 3 | 205 | 20 | 64.02 | 68 |
| 6. | Multinomial | MaxoutWIthDropout | 4 | 250 | 20 | 72.56 | 113 |

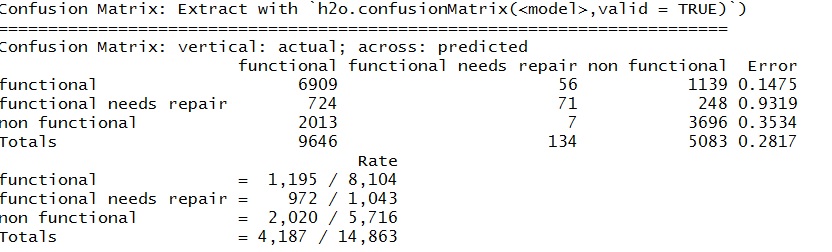


Fig 6.7 Confusion Matrix

Along with the accuracy ,we have also considered other performance metrics from the confusion matrix .

The highest accuracy on training data set we achieved is 72.56. We have decided these as our best set of parameters. We have then populated the submission file for this and achieved an accuracy of **50.67**on submission.

Fig: 6.8 Graph comparing accuracy with the parameters

**7. CONCLUSION**

The final algorithm used for submitting in the competition is gradient boosting. For Deep learning the tradeoff between accuracy of the model and the time consumed is not good enough to consider .Decision trees did not give good accuracies. Random forests was a strong learner and has a good theoretical base for prediction .But extreme gradient boosting gave promising results.

Fig: 7.1 Time graphs for all classifiers

Fig: 7.2 Accuracy graphs for all classifiers

**8. FUTURE WORK**

In the future the data preprocessing can be handled even more sophisticatedly. For example the attributes longitude and latitude could be treated as (x,y) coordinates that can be used to convert into distance from a fixed point . This way we are reducing the attributes as well as making use of their values.

Even the attributes such as regions could be bucketed into north, south, east and west zones by gaining proper geographical knowledge of Tanzania.

Many more such improvisations can improve the classification power of the classifier. In future we can improve our classifier further by using such techniques

**9. CONTRIBUTION OF TEAM MEMBERS**

|  |  |
| --- | --- |
| Understanding the datasets | Teja,Sowmya,Mohana,Gayathri |
| Preprocessing data | Teja,Sowmya,Mohana,Gayathri |
| Status report | Teja,Mohana |
| Proposed solutions | Teja,Sowmya,Mohana,Gayathri |
| Decision tree | Mohana |
| Random Forests | Sowmya |
| Xgboost | Teja |
| Deep Learning | Gayathri |
| Report | Teja,Sowmya,Mohana,Gayathri |

**10. REFERENCES**

[1] “Machine Learning Report format” CS 39ILl: Machine Learning

[2] “R news and tutorials” R Bloggers

[3] “Welcome to deep Learning” Deep Learning

[4] “Correlation based feature selection for Machine Learning”, Mark A Hill, April 1999