Crowdsourced Mapping Data Set

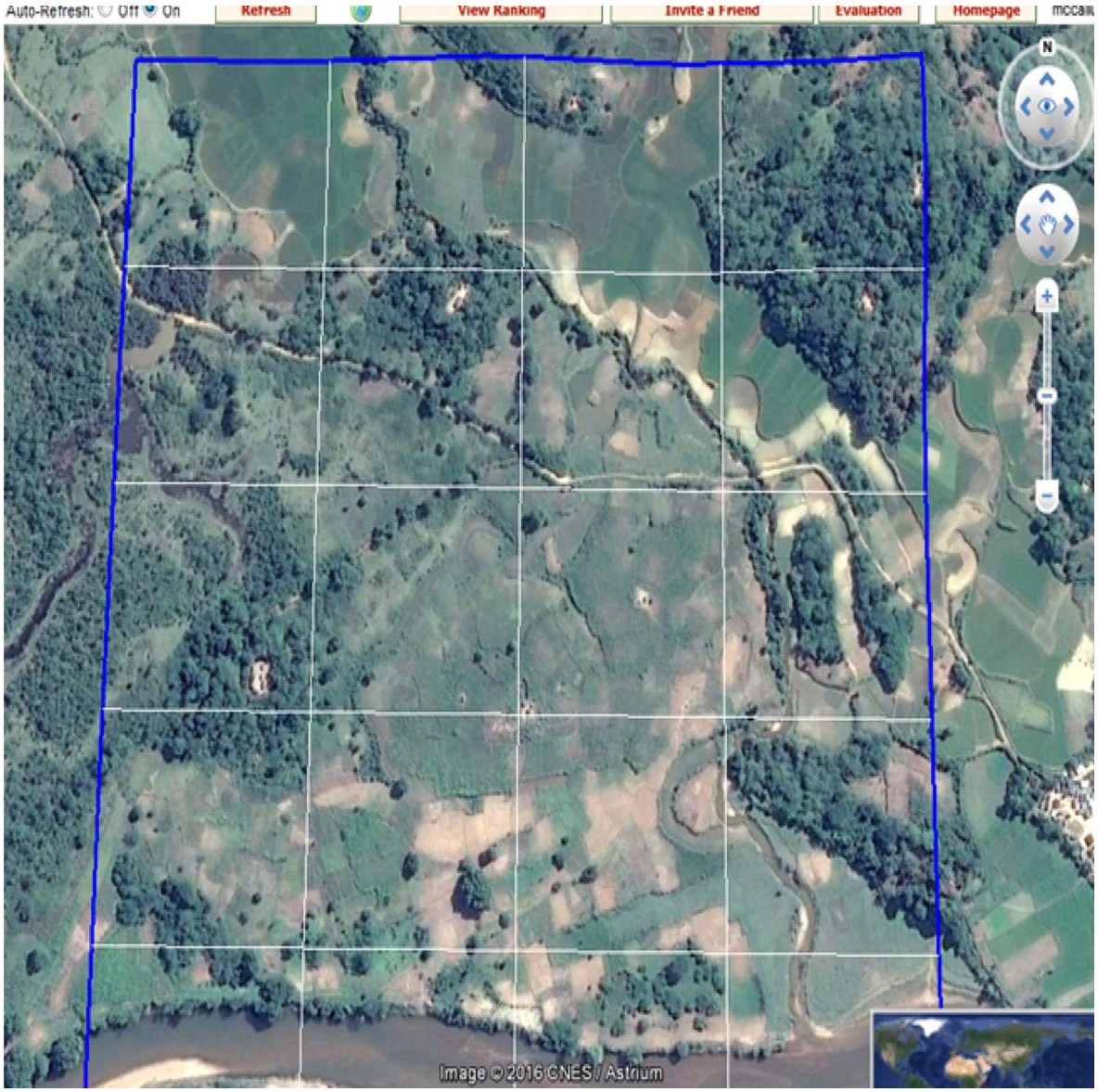
Abstract

Crowdsourced data from OpenStreetMap is used to automate the classification of satellite images into different land cover classes (impervious, farm, forest, grass, orchard, water). Global land cover is an essential climate variable and a key biophysical driver for earth system models.

While remote sensing technology, particularly satellites, have played a key role in providing land cover datasets, large discrepancies have been noted among the available products. Global land use is typically more difficult to map and in many case cannot be remotely sensed. *In-situ* or ground-based data and High “resolution imagery” are thus an important requirement for producing accurate land cover and land use datasets and this is precisely what is lacking.

Here we describe the global land cover and land use reference data derived from the Geo-Wiki crowdsourcing platform via four campaigns. These global datasets provide information on human impact, land cover disagreement, wilderness and land cover and land use. Hence, they are relevant for the scientific community that requires reference data for global satellite-derived products, as well as those interested in monitoring global terrestrial ecosystems in general.

Sample Picture



Methods

The global crowdsourcing derived dataset for land cover and land use is comprised of four campaigns: human impact, land cover disagreement, wilderness and reference. Although related, the campaigns were underpinned by different research questions so the result has been the acquisition of different geographically distributed data sets that are relevant not only for land cover and land use science but also for the creation of remotely-sensed and hybrid products. An overview of the four campaigns is described in detail below. Over 150,000 samples of land cover and land use were acquired globally at more than 100,000 unique locations.

Unless otherwise noted, the following ten land cover/land use classes were used: [1] impervious cover; [2] grass cover; [3] farm vegetation/grassland; [4] Orchard ; [5] forest; [6] water. These form a basic description of global land cover and land use, providing a compromise between retrieving enough information while maintaining simplicity for the contributor. Furthermore, these were chosen to be consistent with generalized land cover classes21, which allows for comparison of existing land cover products. In particular, we collapsed all tree classes into a single class, added a mosaic class of cultivated and managed/natural vegetation and added a flooded wetland class. This set of ten classes was deemed optimal for retrieval of information from high to medium resolution satellite imagery.

This dataset was derived from geospatial data from two sources:

1) Landsat time-series satellite imagery from the years 2014-2015, and

2) crowdsourced georeferenced polygons with land cover labels obtained from OpenStreetMap.

Human impact campaign:

A campaign to evaluate a map of land availability for biofuel production was undertaken using the Geo-Wiki crowdsourcing platform. A stratified random sample of 32,946 pixels at a resolution of 1 km was extracted from two sources: the original biofuel land availability map and cropland locations from the FROM-GLC land cover validation dataset. These were then provided randomly to the participants for assessment. For each pixel, the volunteer was asked to indicate the dominant land cover from a set of 6 basic classes. Additionally, volunteers were asked to provide their level of confidence: unsure; less sure; quite sure; and sure. Furthermore, if the volunteer selected a land cover class containing cultivated vegetation, they were prompted to select a field size: very small; small; medium; and large.

The volunteer was then asked to indicate the overall degree of human impact (on a scale of 0 to 100%) that was visible from the satellite images in the same 1 km pixel along with their confidence in their choice as above. Human impact in this context refers to the degree to which the landscape has been modified by humans as visible from satellite imagery. In a final interpretation question, volunteers were asked to determine if the land was abandoned along with an indication of confidence as above. Finally they were asked to record the satellite image date that was visible at the bottom of the screen. A total of 299 control pixels were independently assessed by three experts for quality assessment purposes.

Download the data from UIC website, save the data into one file they already gave the data into split like train and test sets. By data description it is supervise model. Now we have to do machine learning models, based upon target variable we have to do regression or classification problem we have to choose. Our task is classification problem we have to predict the numeric variables.

My data consists of 29 variables and 10845 observations, target variable is class. Remaining variables all are predictors they are numeric values these values are arranged by NDVI it is calculated by the ration of difference between NIR and red by sum of NIR and red values. It these are image capture values.

$$NDVI = (NIR- red) / (NIR + red)$$

Where NDVI - Normalized difference vegetation index

NIR – Near infrared regions.

- Set the directory.
- Read the train and test data sets.
- Combined the train and test sets.
- See the summary of data.
- Then structure of data.
- Now check the class imbalance in data (i.e, frustration thing).
- By table of class it shows the values of each levels in target variables.
- We have to do smote operation to balance the target variables.
- The data set contain both the positive and negative values except class variables.
- Visualization the variables how they are distributed.
- Pre-processing the data.
- Split the data into train and test sets.
- After completion of pre-processing the it is easy way to do machine models.
- And final we have to compare the accuracy for each and every models in that which one is the best model, also precision and recall.
- Final conclusion on project.

Set the working directory it is used to call the data into R or python.
They already data is split into train and test sets.
Combine the data because we have to split the data into the ratio of 70 and 30.
Before the combined the data it is in the ratio of 97.2 and 2.8 with respective to train and test sets. For that we have to combine the data after completion of combine we have to split the data into train and test in 70 and 30.

While seeing the summary of data we see that min, 1st quartile, Median, Mean, 3rd quartile, Max values of each variables.
Structure of data we can find that factors, numeric variables.

Class imbalance is not easy to the balance target variables, I am frustration with that. The target variables has six levels, in that one class take 75% of records, and remaining levels are share by 25%.

By the smote operation we have to set the oversampling and under sampling it took lot of time to set the values to balance the target variable.

The data set contain both positive and negative values it is arranged by NDVI computation. Visualization is main task how the distribution happened in the data set. We have to visualize the data in pictorial representation.

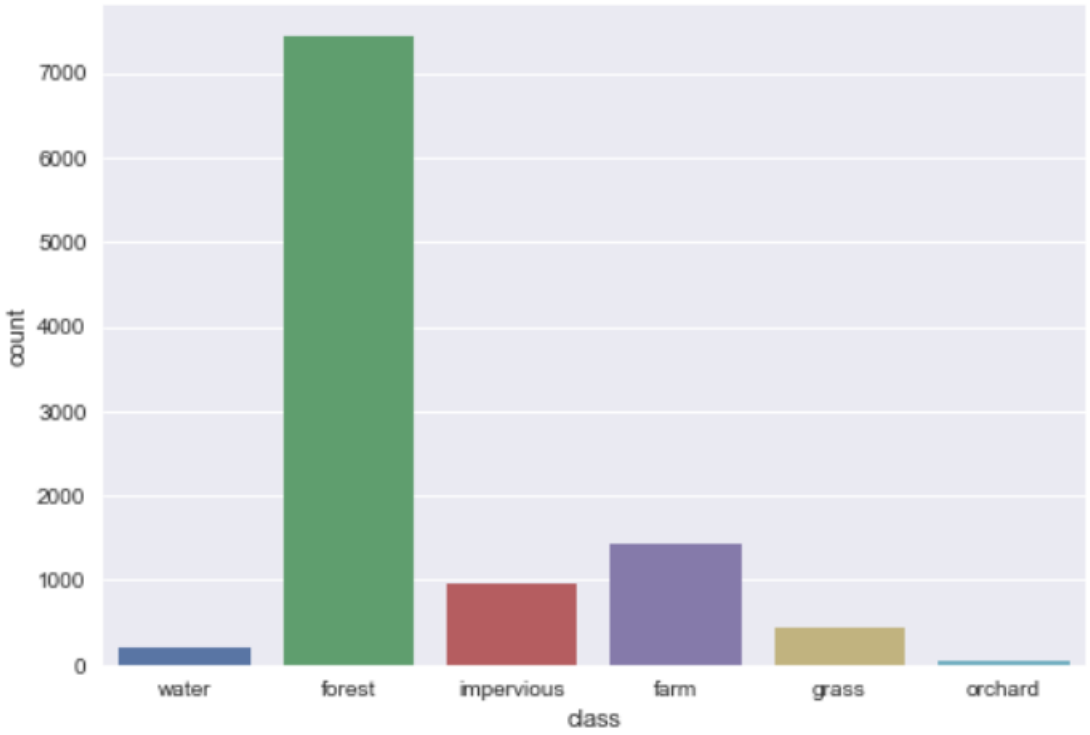
Pre-processing the data is the major task to build the machine learning models.
For pre-processing the data I need to used pre-processing function to standardize and normalized the data, then numeric values are comes in normal form, it is useful for increase the accuracy of models.

Now we have to split the data into train and test, here we have to build the model on train set with respective of there model function and required libraries. And predict on test set with class variable. Finally we have to check the confusion matrix in that our error matrix has accuracy, precision and recall. Based upon the all the models accuracy, precision and recall we have to check the which one is the best model to our data set.

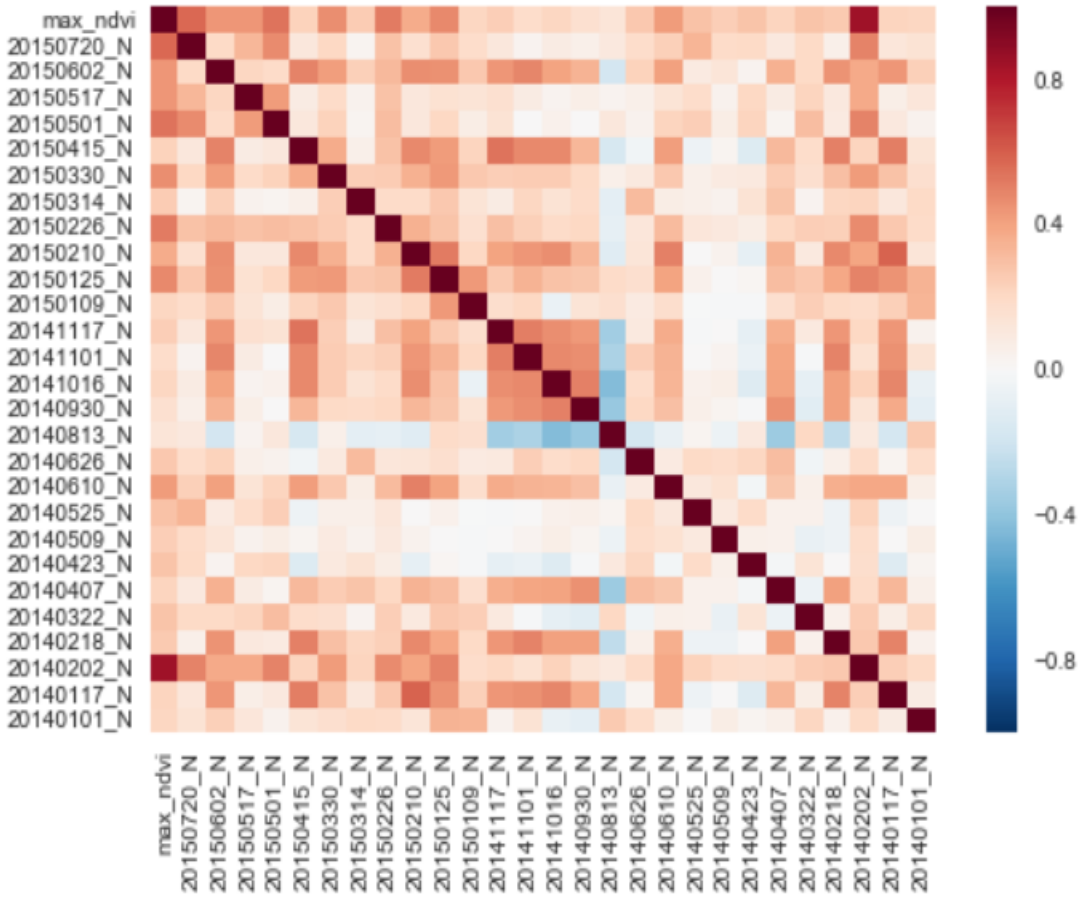
Finally we have to conclude the project domain with help of error matrix.

Visualization

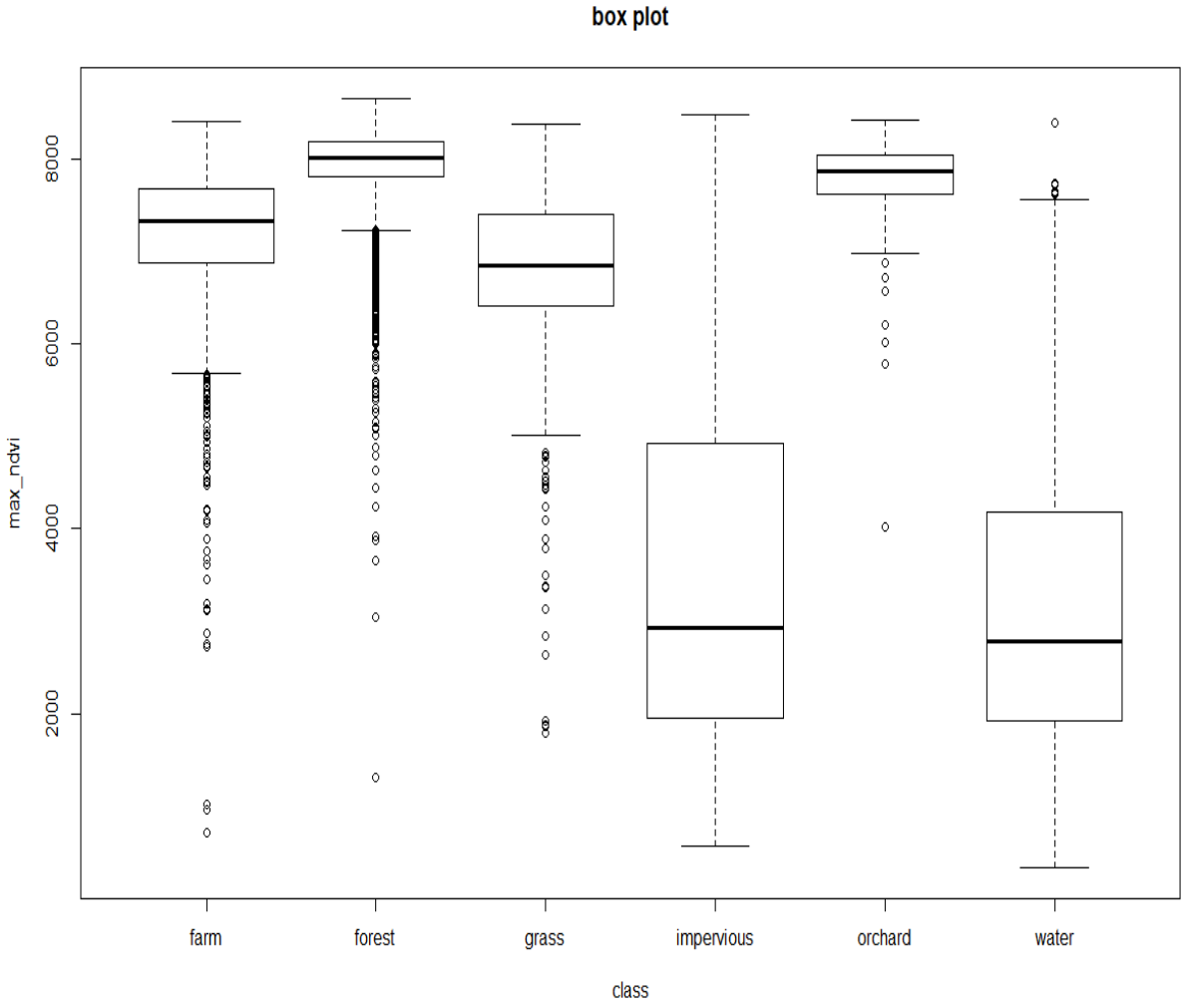
Bar plot:



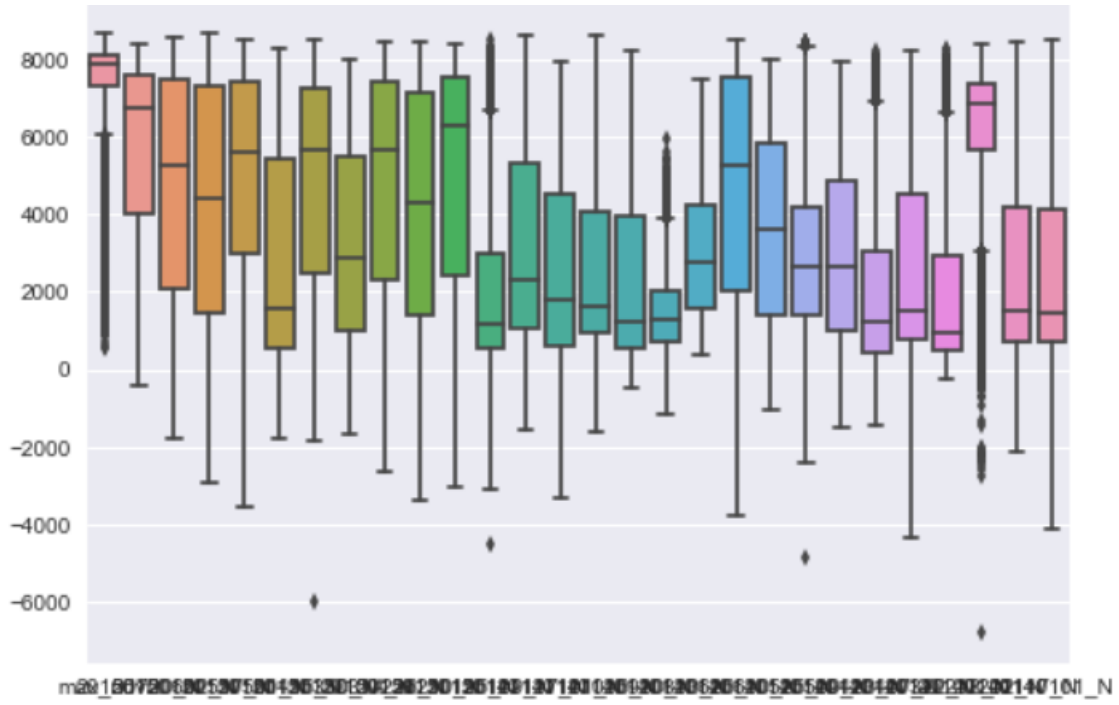
Correlation Plot:



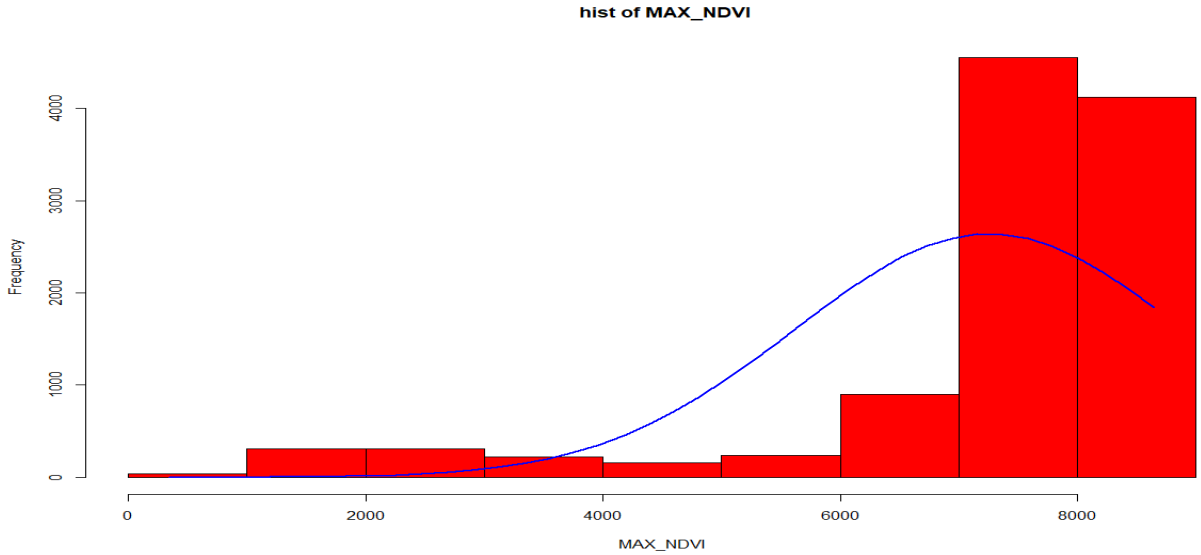
Box Plot:


box plot
max_ndvi
2000
4000
6000
8000
farm
forest
grass
impervious
orchard
water
class

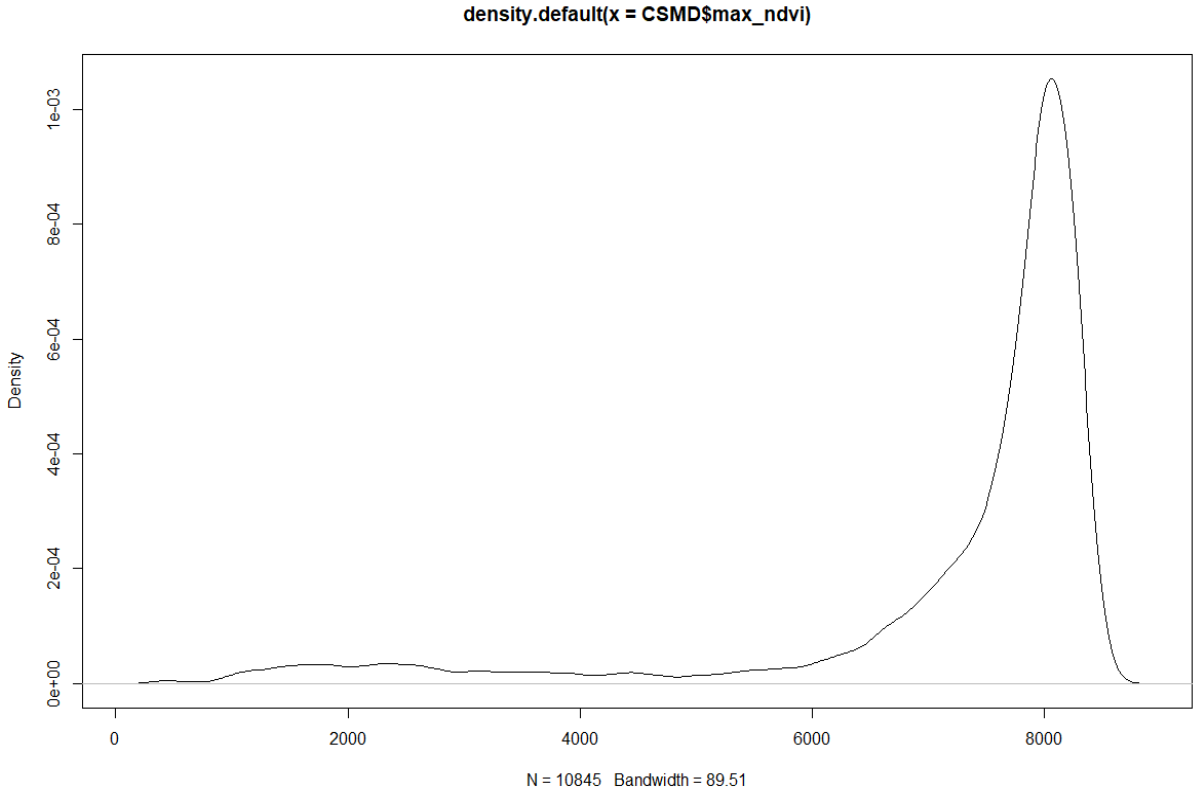
Box plot of all numeric variables:



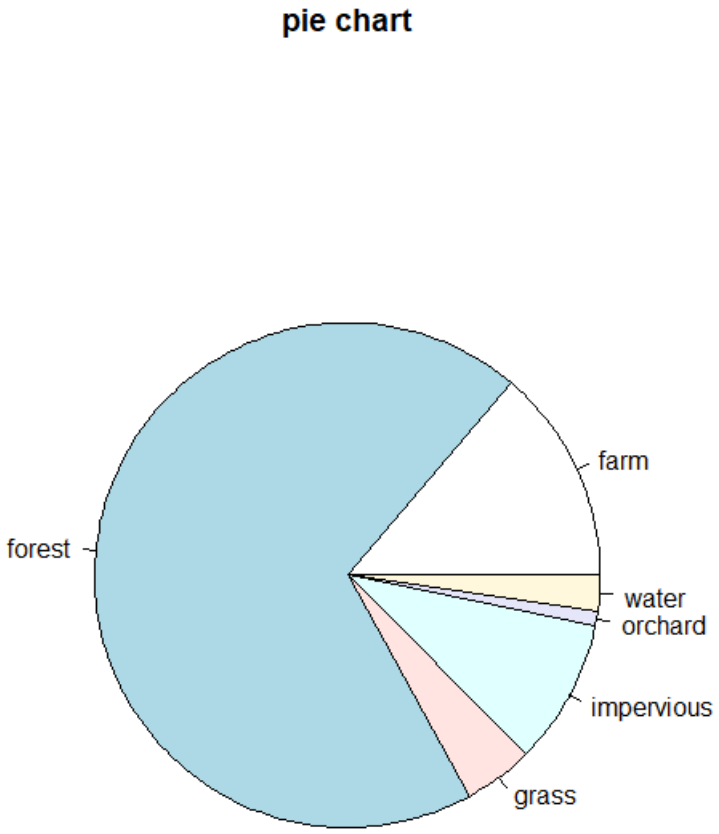
Density with histogram plot:

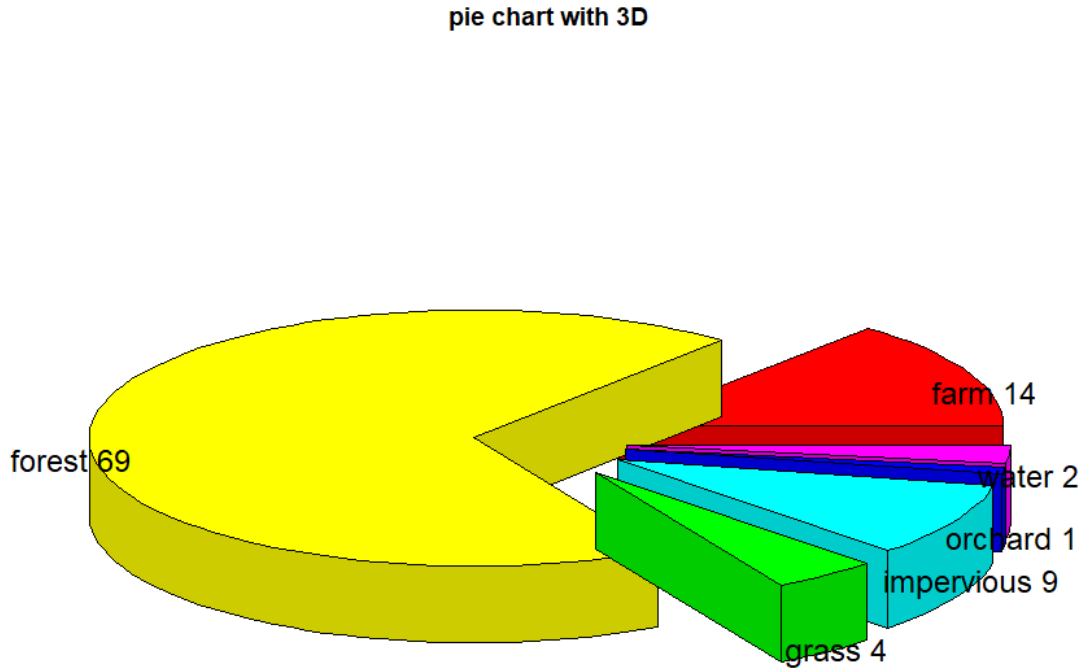


Density plot:



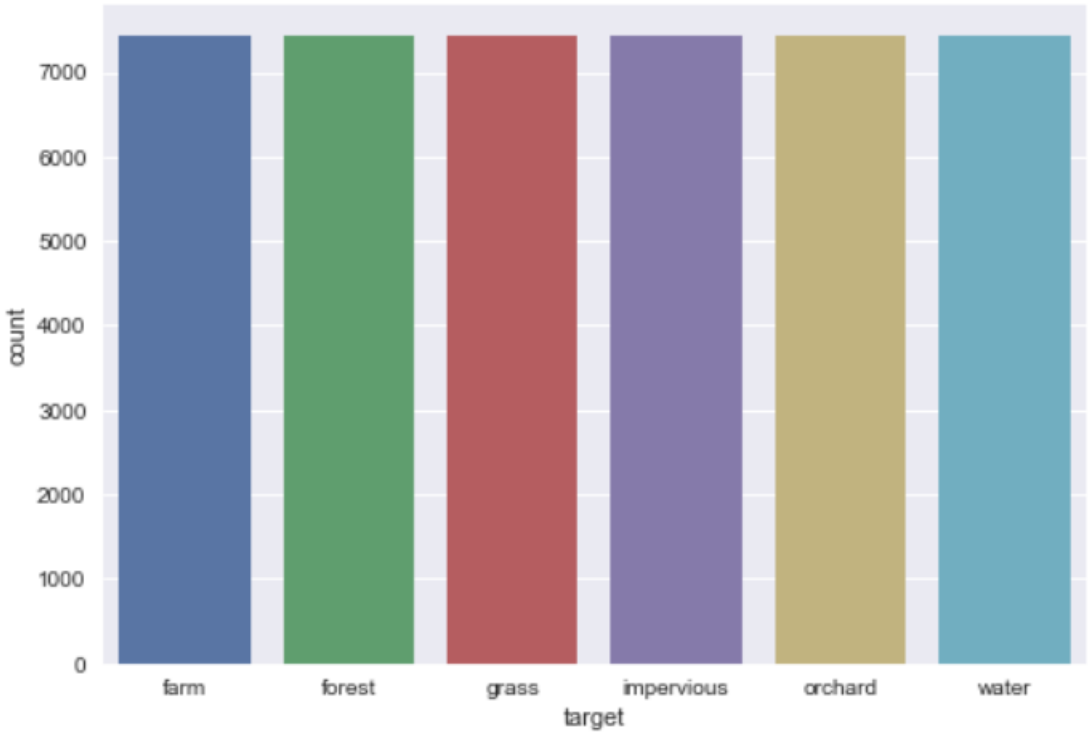
Pie:


pie chart
farm
forest
water
orchard
impervious
grass

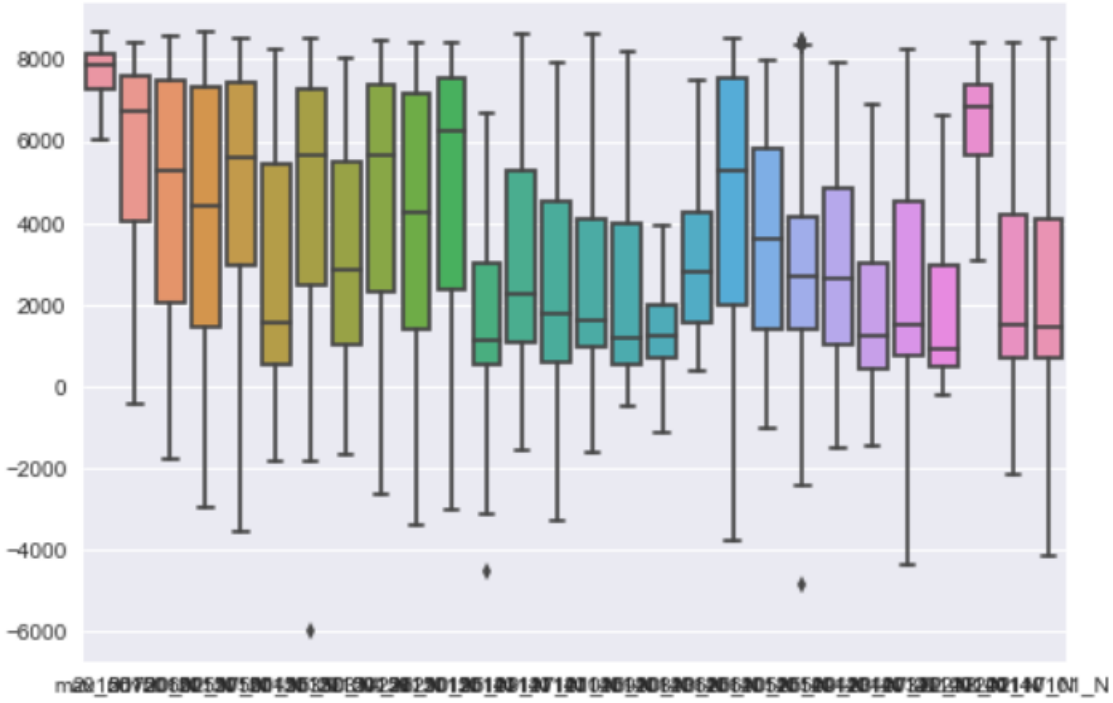

pie chart with 3D
farm 14
forest 69
water 2
orchard 1
impervious 9
grass 4

After removing the all outliers and class imbalance visualization.

Bar Plot:



After removing outliers Box plot:



Modelling:

Logistic model:

For Logistic model the train data I used multinomial function, because it is multiclass problem. I just applied the model on train data and predict the model on the test data for accuracy and precision. It is how model is perform well. And here our error metric is both Accuracy, Precision and Recall,

After building the model I predict the model on train data it gives the accuracy 81.5 %.

Then I check on the test data it gives the accuracy 58.6 %.

Precision on test data 60.1 %.
Recall on test data 60.3 %.

Code for Logistic model:

```
from sklearn import linear_model

from sklearn import metrics

lr = linear_model.LogisticRegression()

mul_lr = linear_model.LogisticRegression(multi_class='multinomial', solver='newton-cg')

mul_lr.fit(X_train, Y_train)

prediction_lr = mul_lr.predict(X_test)

prediction_lr

from sklearn.metrics import accuracy_score,precision_score, recall_score, confusion_matrix

metrics.accuracy_score(Y_train, mul_lr.predict(X_train))

metrics.accuracy_score(Y_test, mul_lr.predict(X_test))

precision = precision_score(Y_test, prediction_lr,average='macro')

precision

recall = recall_score(Y_test,prediction_lr, average='macro')

recall

print(precision,recall)
```

Decision tree:

For decision tree we can't do the class imbalance, they we will take, if we done doesn't matter. It will take both the balanced and unbalance class level.

While doing the model fit on train data it gives the 100 % accuracy it is over fitting.

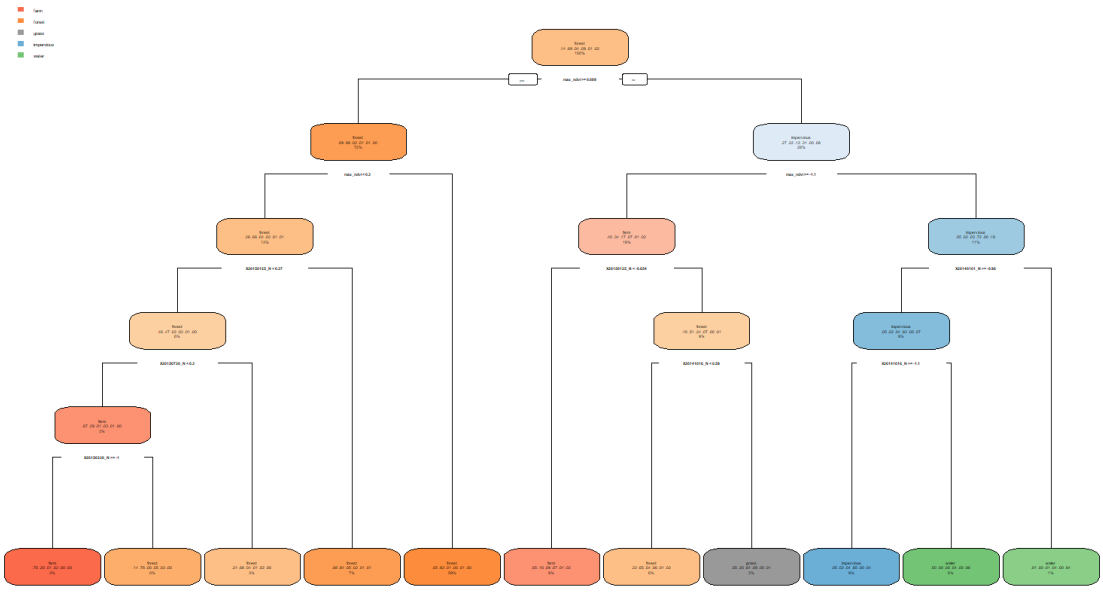
Prediction on test data it gives the accuracy 55 %.

Precision, Recall 58.8 % , 52.4 %.

Code for Decision tree:

```
from sklearn import tree
tree = tree.DecisionTreeClassifier()
tree_model = tree.fit(X_train, Y_train)
metrics.accuracy_score(Y_train, tree_model.predict(X_train))
prediction_tr = tree.predict(X_test)
acc = accuracy_score(prediction_tr, Y_test)
acc
precision = precision_score(Y_test, prediction_tr,average='macro')
precision
recall = recall_score(Y_test,prediction_tr, average='macro')
recall
print(precision,recall)
```

Decision tree plot:



Random Forest:

It is tree, it will generate the number of tree based upon the n values, it will take random values as root node. They will occur any number of different root nodes.

By the build the random forest as the function, build on the train data.

Prediction on model it get the accuracy of 99.9 %.

Prediction on test set the accuracy 60.3 %.

Precision, Recall 73.2 % and 57.1 %.

Code for Random forest:

```
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
rf = RandomForestClassifier()
rf = rf.fit(X_train, Y_train)
prediction_rf = rf.predict(X_test)
metrics.accuracy_score(Y_train, rf.predict(X_train))
acc = accuracy_score(Y_test, prediction_rf)
acc
metrics.accuracy_score(Y_test, rf.predict(X_test))
precision = precision_score(Y_test, prediction_rf, average='macro')
precision
recall = recall_score(Y_test,prediction_rf,average='macro')
recall
print(precision,recall)
```

Random Forest Plot: