Bush density mapping

CIAT

02 November, 2016, 17:32

Objectives

This manual will help you calculate bush density using count data with ndvi and crown cover data as covariates. At the end of this session, you will be able to:

1. Import excel data into R.
2. Import raster data into R.
3. Convert data from excel to ESRI point shapefile.
4. Use the point and raster data to construct a random forest model.
5. Use the RF model to predict bush density.
6. Test model accuracy.

Download the sample dataset we will use in this session from this link https://drive.google.com/open?id=0B__Gkb__0tNKkQcGZGc1ZWRnpZc0U

Before you start this session, it is important you have (i) the latest R software and (ii) Rstudio installed in your computer.

Start the session but first clear your work space.

```
rm(list = ls(all = TRUE))
```

To enable us reproduce the results next time, let's set the seed.

```
set.seed(211134)
```

Set the start of data processing.

```
startTime <- Sys.time()
cat("Start time", format(startTime),"\n")
```

```
## Start time 2016-11-02 17:32:06
```

Set working directory.

```
setwd("C:/LDN_Workshop/Sample_dataset/Bush_Density_Mapping")
```

List down the packages to be used in this session. Packages will be installed if not already installed. They will then be loaded into the session.

```
.packages = c("sp","rgdal","raster","randomForest","plyr","xlsx","xlsxjars",
              "dplyr","caret","car", "e1071","snow")
.inst <- .packages %in% installed.packages()
if(length(.packages[!.inst]) > 0) install.packages(.packages[!.inst])
lapply(.packages, require, character.only=TRUE)
```

```
## Loading required package: sp
```

```
## Loading required package: rgdal

## rgdal: version: 1.1-10, (SVN revision 622)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.0.1, released 2015/09/15
##  Path to GDAL shared files: C:/Users/jymutua/Documents/R/win-library/3.3/rgdal/gdal
##  Loaded PROJ.4 runtime: Rel. 4.9.2, 08 September 2015, [PJ_VERSION: 492]
##  Path to PROJ.4 shared files: C:/Users/jymutua/Documents/R/win-library/3.3/rgdal/proj
##  Linking to sp version: 1.2-3

## Loading required package: raster

## Loading required package: randomForest

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

## Loading required package: plyr

## Loading required package: xlsx

## Loading required package: rJava

## Loading required package: xlsxjars

## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:randomForest':
##
##     combine

## The following objects are masked from 'package:raster':
##
##     intersect, select, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Loading required package: caret

## Loading required package: lattice

## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:randomForest':
##
##     margin

## Loading required package: car

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## Loading required package: e1071

##
## Attaching package: 'e1071'

## The following object is masked from 'package:raster':
##
##     interpolate

## Loading required package: snow

## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
##
## [[6]]
## [1] TRUE
##
## [[7]]
## [1] TRUE
##
## [[8]]
## [1] TRUE
##
## [[9]]
```

```
## [1] TRUE
##
## [[10]]
## [1] TRUE
##
## [[11]]
## [1] TRUE
##
## [[12]]
## [1] TRUE
```

To get help on the functions and data sets in R, use `help()` or `?`. For example, to view the help file for the `calc` function, type one of the following:

```
help(calc)
?calc
```

Reading your data

Read in data from excel sheet

```
d <- read.xlsx("Field_data/Otji_BD_Sampling_Points.xlsx", sheetName =
                        "Sheet1", header=TRUE)
```

Calculate values by finding the median in crown cover and mean of values for counts.Note that 1 plot = 0.01 ha; 4 plots = 0.04 ha

```
d$shrubs_less_1.5 <- apply(d[,8:11], 1, sum, na.rm=TRUE)
d$shrubs_more_1.5_no_stem <- apply(d[,16:19], 1, sum, na.rm=TRUE)
d$shrubs_more_1.5_stem <- apply(d[,24:27], 1, sum, na.rm=TRUE)
```

Create new 'data.frame' with the columns you need

```
d<-d[,c("Waypoint_No","Latitude","Longitude",
                            "shrubs_less_1.5","shrubs_more_1.5_no_stem",
                            "shrubs_more_1.5_stem")]
```

Add two new columns of shrubs more than 1.5 and all shrubs in general

```
d$shrubs_more_1.5 <- apply(d[,5:6], 1, sum, na.rm=TRUE)
d$shrubs_all <- apply(d[,4:6], 1, sum, na.rm=TRUE)
```

Round columns and create new 'data.frame with the columns you need.

```
d<-d %>%
  mutate_each(funs(round(.,0)), shrubs_less_1.5, shrubs_more_1.5,
              shrubs_all)
d<-d[,c("Waypoint_No","Latitude","Longitude","shrubs_less_1.5",
        "shrubs_more_1.5", "shrubs_all")]
```

Compute shrubs per hectare.

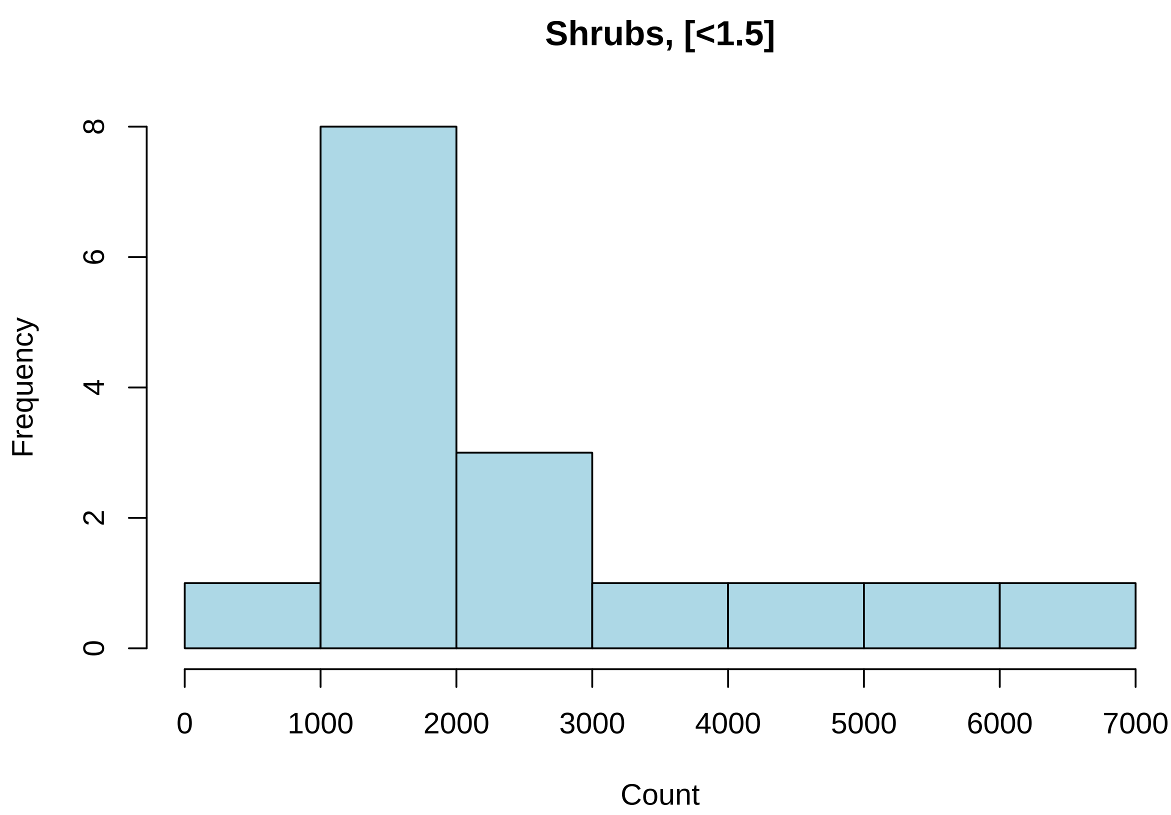
```
d$shrubs_less_1.5 <- d$shrubs_less_1.5*25
d$shrubs_more_1.5 <- d$shrubs_more_1.5*25
d$shrubs_all <- d$shrubs_all*25
```

Remove all NAs.

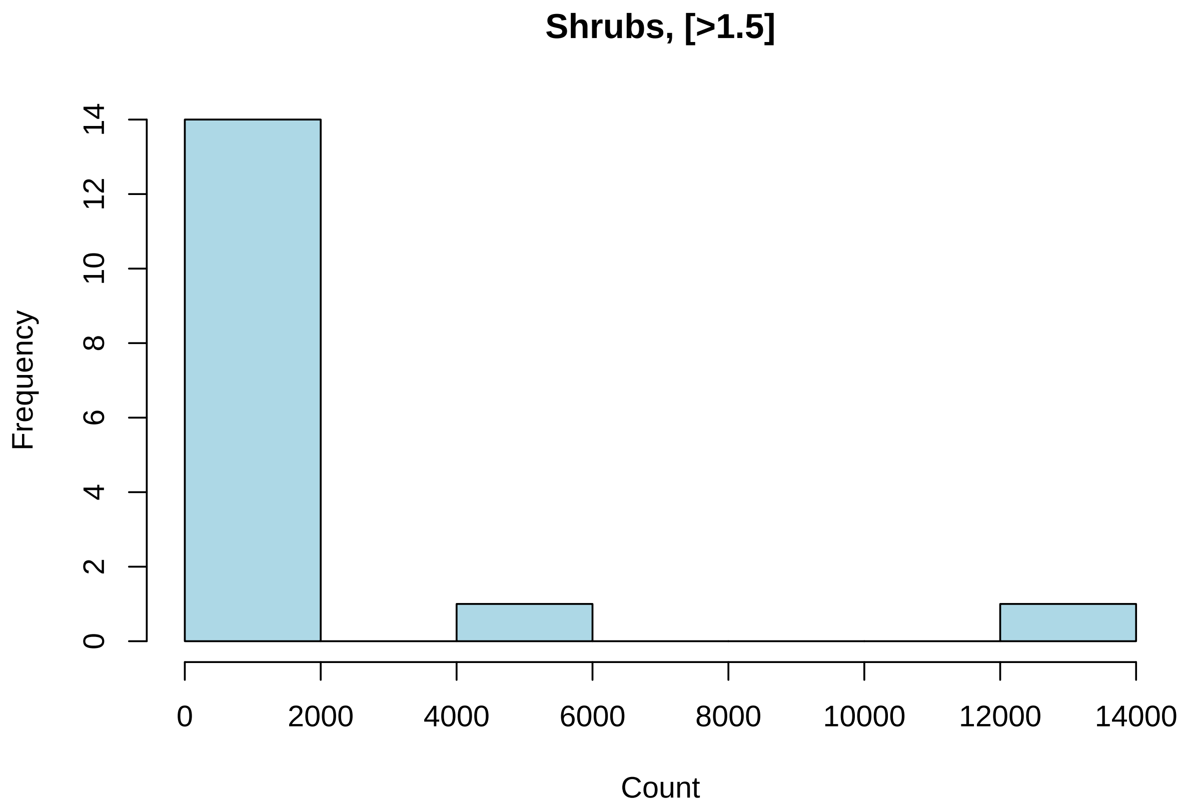
```
d<-d[complete.cases(d),]
```

Plot histograms of the three variables

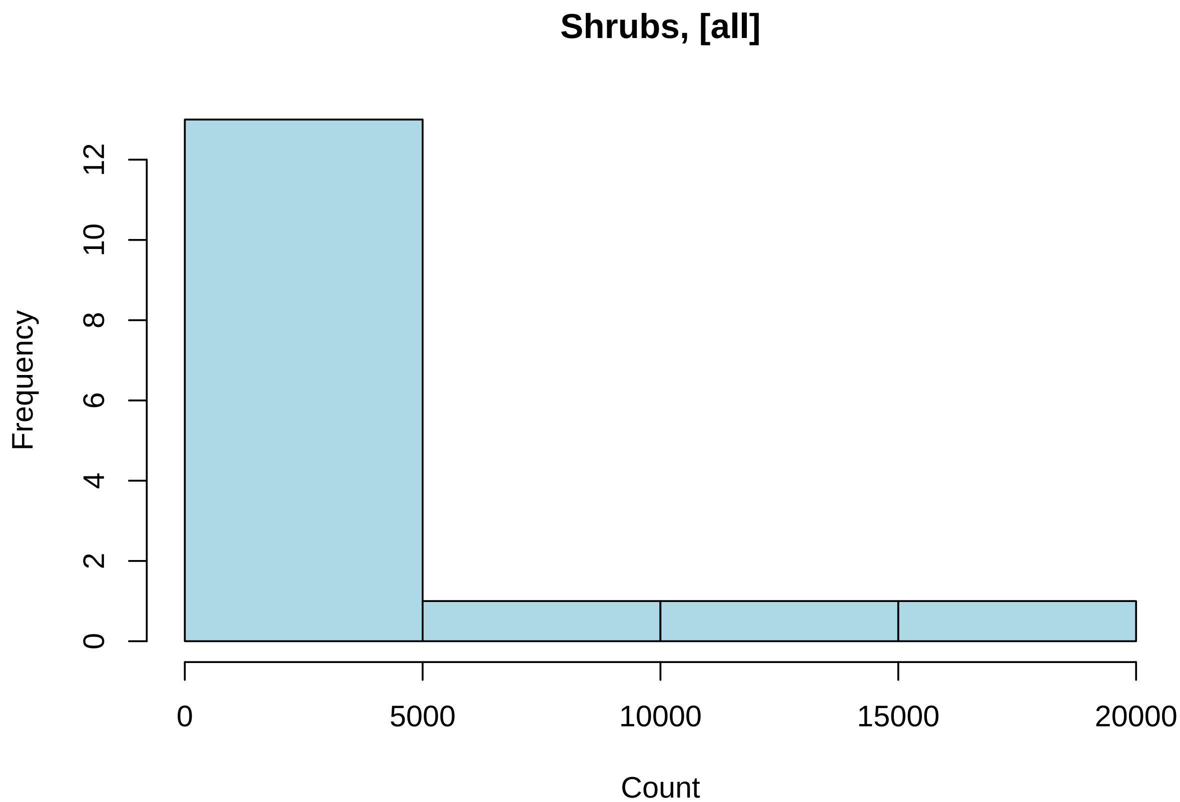
```
hist(d$shrubs_less_1.5, col = "lightblue", xlab="Count", main="Shrubs, [<1.5]")
```




```
hist(d$shrubs_more_1.5, col = "lightblue", xlab="Count", main="Shrubs, [>1.5]")
```




```
hist(d$shrubs_all, col = "lightblue", xlab="Count", main="Shrubs, [all]")
```



Export data to .csv

```
write.csv(d, file = "Otji_BushData_trainData.csv",row.names=FALSE)
```

Get long and lat from your data.frame. Make sure that the order is in lon/lat. Convert the dataraframe into a spatial point dataframe.

```
xy <- d[,c(3,2)]
trainDatageo <- SpatialPointsDataFrame(coords = xy, data = d,
                                       proj4string = CRS("+proj=longlat
                                                         +datum=WGS84"))
trainData <- spTransform(trainDatageo, CRS('+proj=utm +zone=33 +south
                                             +datum=WGS84'))
```

Let's do some background checking of the field names and rename trainData fields

```
names(trainData)
```

```
## [1] "Waypoint_No"     "Latitude"        "Longitude"       "shrubs_less_1.5"
## [5] "shrubs_more_1.5" "shrubs_all"
```

```
names(trainData) <- c("Waypoint_No","Latitude","Longitude","shrubs_less_1.5",
                      "shrubs_more_1.5", "shrubs_all")
```

Import the rest of input data, stack and rename contents

```
r.list<-list.files(path = ".", pattern = ".tif$", full.names = TRUE)
r.stack <- stack(r.list)
names(r.stack) <- c("crown_cover","NDVI","band2","band3","band4","band5",
                    "band6","band7")
```

Note that we are only calculating bush density in the bush area LULC catageory. Import the bush area mask

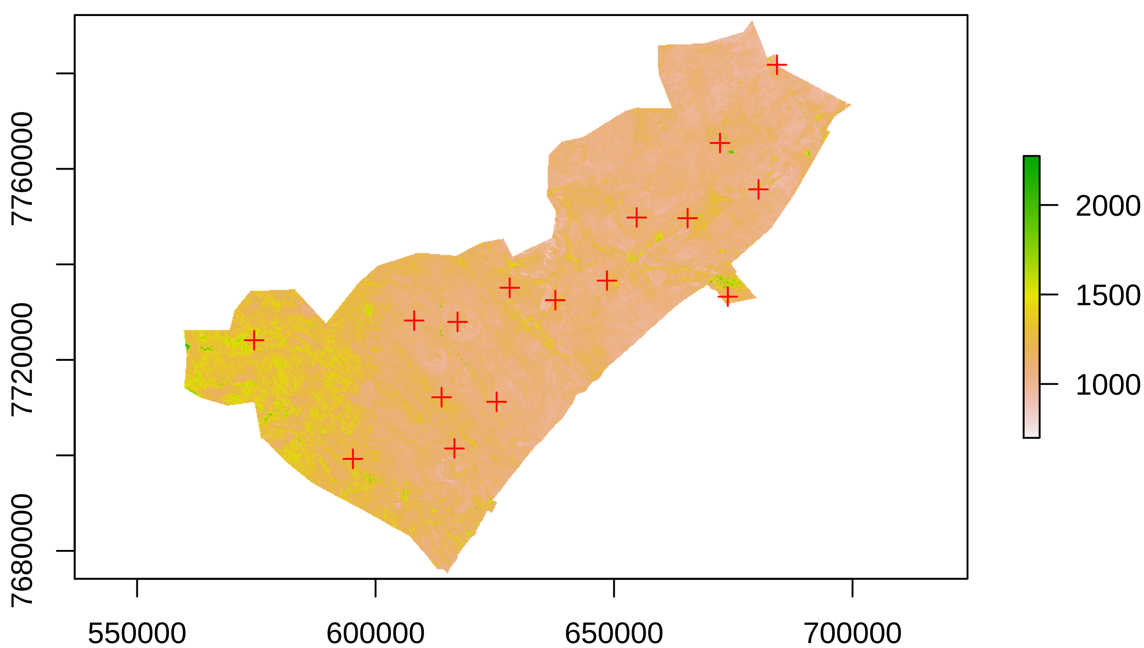
```
o.mask <- raster("Other_data/Otji_BushArea_2016.tif")
```

Set extent of the training data to match covs

```
trainData@bbox <- bbox(o.mask )
```

Plot the points on top of `layer 3` of the raster stack

```
plot(r.stack[[3]])
plot(trainData, add=TRUE, col = "red", pch = 3)
```



Mask, read and stack the covariates. Remove NA values (otherwise RF cannot predict)

```
covs <- mask(r.stack, o.mask )
covs <- na.omit(covs)
```

Assign raster values to the training data.

```
v<-as.data.frame(extract(covs,trainData))
trainData@data=data.frame(trainData@data, v[match(rownames(trainData@data),
                                                  rownames(v)),])
```

Rename fields in the training dataset, remove NAs and write the dataset as a .csv

```
names(trainData) <- c("waypoint_no","latitude","longitude","shrubs_less_1.5",
                      "shrubs_more_1.5","shrubs_all","crown_cover","NDVI",
                      "band2","band3","band4","band5","band6","band7")
trainData@data<-trainData@data[complete.cases(trainData@data),]
write.csv(trainData@data, file = "Otji_MF_trainData.csv",row.names=FALSE)
```

Compute summary statistics.

```
summary(trainData$shrubs_all)
```

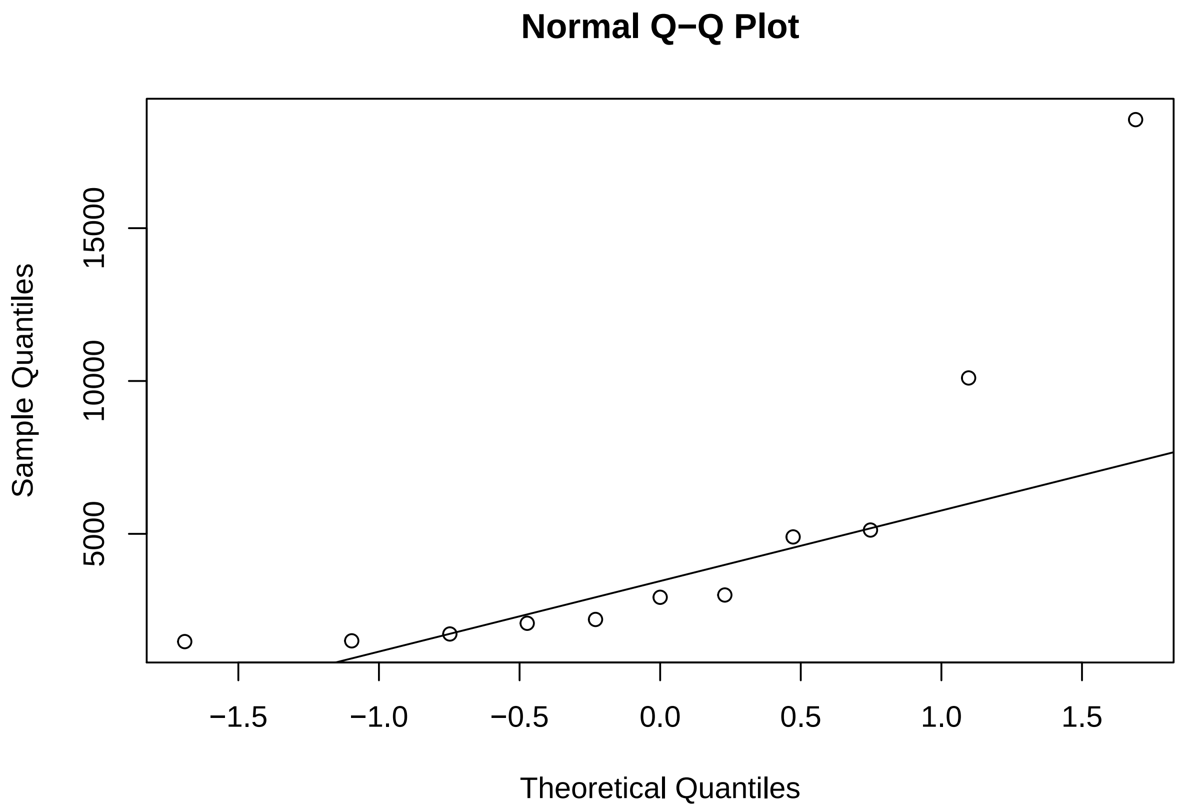
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1475    1900    2925    4870    5012   18550
```

```
skewness(trainData$shrubs_all, na.rm=T)
```

```
## [1] 1.649636
```

QQ plot.

```
qqnorm(trainData$shrubs_all)
qqline(trainData$shrubs_all)
```



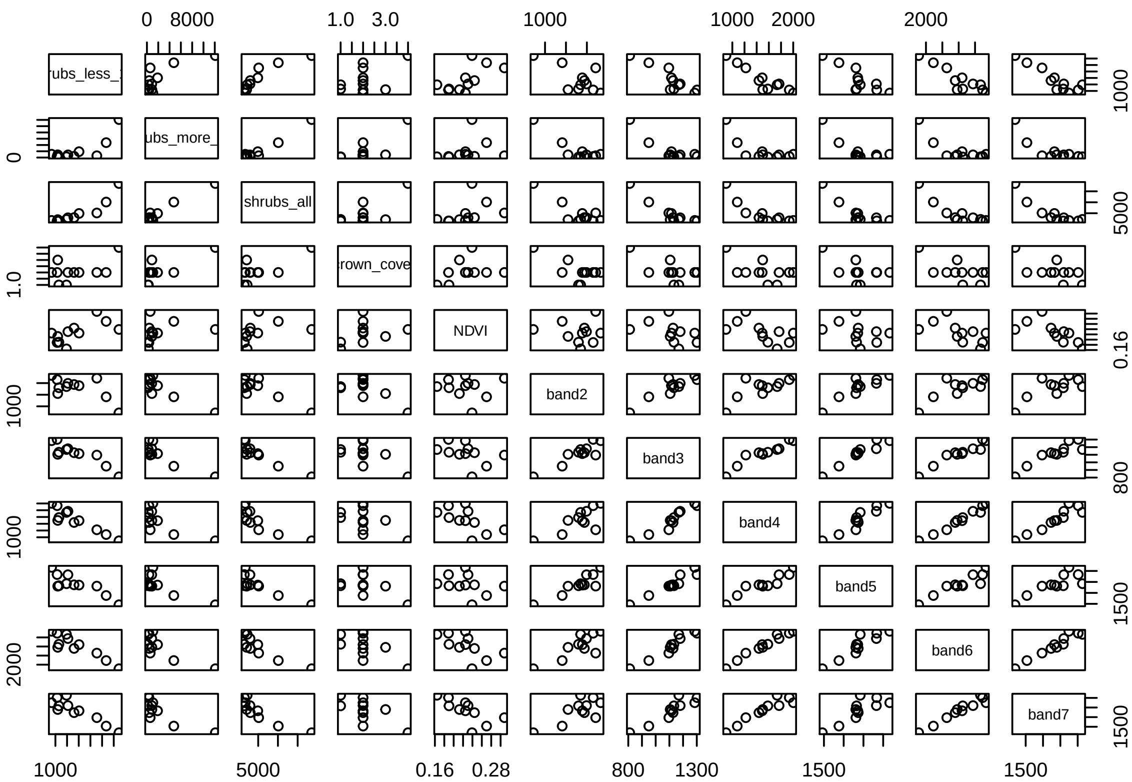
Compute correlation coefficients and plot correlations

```
cor(trainData@data[,4:14])
```

```
##                 shrubs_less_1.5 shrubs_more_1.5 shrubs_all crown_cover
## shrubs_less_1.5       1.0000000       0.7870470  0.9062045   0.5113658
## shrubs_more_1.5       0.7870470       1.0000000  0.9740724   0.7618735
## shrubs_all            0.9062045       0.9740724  1.0000000   0.7097503
## crown_cover           0.5113658       0.7618735  0.7097503   1.0000000
## NDVI                  0.6444677       0.2479644  0.4063124   0.3291826
## band2                -0.6673499      -0.8699912 -0.8410627  -0.6596828
## band3                -0.8883771      -0.8492370 -0.9078956  -0.6004480
## band4                -0.9182825      -0.7182936 -0.8291098  -0.5363318
## band5                -0.8334936      -0.7572776 -0.8247357  -0.4960407
## band6                -0.8988966      -0.7742281 -0.8603397  -0.6219464
## band7                -0.9030470      -0.7987772 -0.8786887  -0.6675928
##                        NDVI       band2      band3      band4      band5
## shrubs_less_1.5  0.64446772 -0.66734993 -0.8883771 -0.9182825 -0.8334936
## shrubs_more_1.5  0.24796438 -0.86999115 -0.8492370 -0.7182936 -0.7572776
## shrubs_all       0.40631237 -0.84106273 -0.9078956 -0.8291098 -0.8247357
## crown_cover      0.32918259 -0.65968278 -0.6004480 -0.5363318 -0.4960407
## NDVI             1.00000000 -0.04182892 -0.4215472 -0.6054592 -0.2995566
## band2           -0.04182892  1.00000000  0.8998662  0.7552128  0.8846406
```

```
## band3           -0.42154722  0.89986622  1.0000000  0.9515501  0.9578004
## band4           -0.60545915  0.75521283  0.9515501  1.0000000  0.9400076
## band5           -0.29955664  0.88464059  0.9578004  0.9400076  1.0000000
## band6           -0.59969871  0.79484537  0.9568856  0.9851810  0.9237968
## band7           -0.70287392  0.72027622  0.9121251  0.9365657  0.8204991
##                      band6      band7
## shrubs_less_1.5 -0.8988966 -0.9030470
## shrubs_more_1.5 -0.7742281 -0.7987772
## shrubs_all      -0.8603397 -0.8786887
## crown_cover     -0.6219464 -0.6675928
## NDVI            -0.5996987 -0.7028739
## band2            0.7948454  0.7202762
## band3            0.9568856  0.9121251
## band4            0.9851810  0.9365657
## band5            0.9237968  0.8204991
## band6            1.0000000  0.9682497
## band7            0.9682497  1.0000000
```

```r
pairs(trainData@data[,4:14])
```



Correlate count of shrubs with NDVI and Landsat 8 band 2-7.

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$NDVI)
```

```
## [1] 0.6444677
```

```r
cor(trainData@data$shrubs_all,trainData@data$NDVI)
```

```
## [1] 0.4063124
```

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$crown_cover)
```

```
## [1] 0.5113658
```

```r
cor(trainData@data$shrubs_all,trainData@data$crown_cover)
```

```
## [1] 0.7097503
```

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$band2)
```

```
## [1] -0.6673499
```

```r
cor(trainData@data$shrubs_all,trainData@data$band2)
```

```
## [1] -0.8410627
```

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$band3)
```

```
## [1] -0.8883771
```

```r
cor(trainData@data$shrubs_all,trainData@data$band3)
```

```
## [1] -0.9078956
```

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$band6)
```

```
## [1] -0.8988966
```

```r
cor(trainData@data$shrubs_all,trainData@data$band6)
```

```
## [1] -0.8603397
```

```r
cor(trainData@data$shrubs_less_1.5,trainData@data$band7)
```

```
## [1] -0.903047
```

```r
cor(trainData@data$shrubs_all,trainData@data$band7)
```

```
## [1] -0.8786887
```

Select covariates based on correlation analysis and save as a ‘data.frame’.

```r
d <- trainData@data[,c("waypoint_no","latitude","longitude",
                       "shrubs_less_1.5","shrubs_all",
                       "crown_cover", "NDVI")]
names(d)
```

```
## [1] "waypoint_no"     "latitude"        "longitude"       "shrubs_less_1.5"
## [5] "shrubs_all"      "crown_cover"     "NDVI"
```

Fitting the Random Forest regression models

You can now fit the models using the 'randomForest' model i.e. Specify the model as a formula with the dependent variable (i.e., count of shrubs).

```
model1 <- randomForest(x = d[,c(6:7)], y = d[,"shrubs_less_1.5"])
model3 <- randomForest(x = d[,c(6:7)], y = d[,"shrubs_all"])
```

```
str(model1, max.level=2)
```

```
## List of 17
##  $ call           : language randomForest(x = d[, c(6:7)], y = d[, "shrubs_less_1.5"])
##  $ type           : chr "regression"
##  $ predicted      : Named num [1:11] 3376 2542 1826 2522 1627 ...
##   ..- attr(*, "names")= chr [1:11] "1" "4" "5" "7" ...
##  $ mse            : num [1:500] 1639688 2954267 2362450 2278256 2309962 ...
##  $ rsq            : num [1:500] 0.502 0.104 0.283 0.309 0.299 ...
##  $ oob.times      : int [1:11] 171 186 180 180 183 179 167 165 164 182 ...
##  $ importance     : num [1:2, 1] 8526728 19041880
##   ..- attr(*, "dimnames")=List of 2
##  $ importanceSD   : NULL
##  $ localImportance: NULL
##  $ proximity      : NULL
##  $ ntree          : num 500
##  $ mtry           : num 1
##  $ forest         :List of 11
##   ..$ ndbigtree    : int [1:500] 5 7 5 5 9 7 9 5 7 7 ...
##   ..$ nodestatus   : int [1:11, 1:500] -3 -3 -1 -1 -1 0 0 0 0 0 ...
##   ..$ leftDaughter : int [1:11, 1:500] 2 4 0 0 0 0 0 0 0 0 ...
##   ..$ rightDaughter: int [1:11, 1:500] 3 5 0 0 0 0 0 0 0 0 ...
##   ..$ nodepred     : num [1:11, 1:500] 3270 2564 6450 1300 2925 ...
##   ..$ bestvar      : int [1:11, 1:500] 1 1 0 0 0 0 0 0 0 0 ...
##   ..$ xbestsplit   : num [1:11, 1:500] 3 1.5 0 0 0 0 0 0 0 0 ...
##   ..$ ncat         : Named int [1:2] 1 1
##   .. ..- attr(*, "names")= chr [1:2] "crown_cover" "NDVI"
##   ..$ nrnodes      : int 11
##   ..$ ntree        : num 500
##   ..$ xlevels      :List of 2
##  $ coefs          : NULL
##  $ y              : num [1:11] 5350 2075 1150 675 1300 ...
##  $ test           : NULL
##  $ inbag          : NULL
##  - attr(*, "class")= chr "randomForest"
```

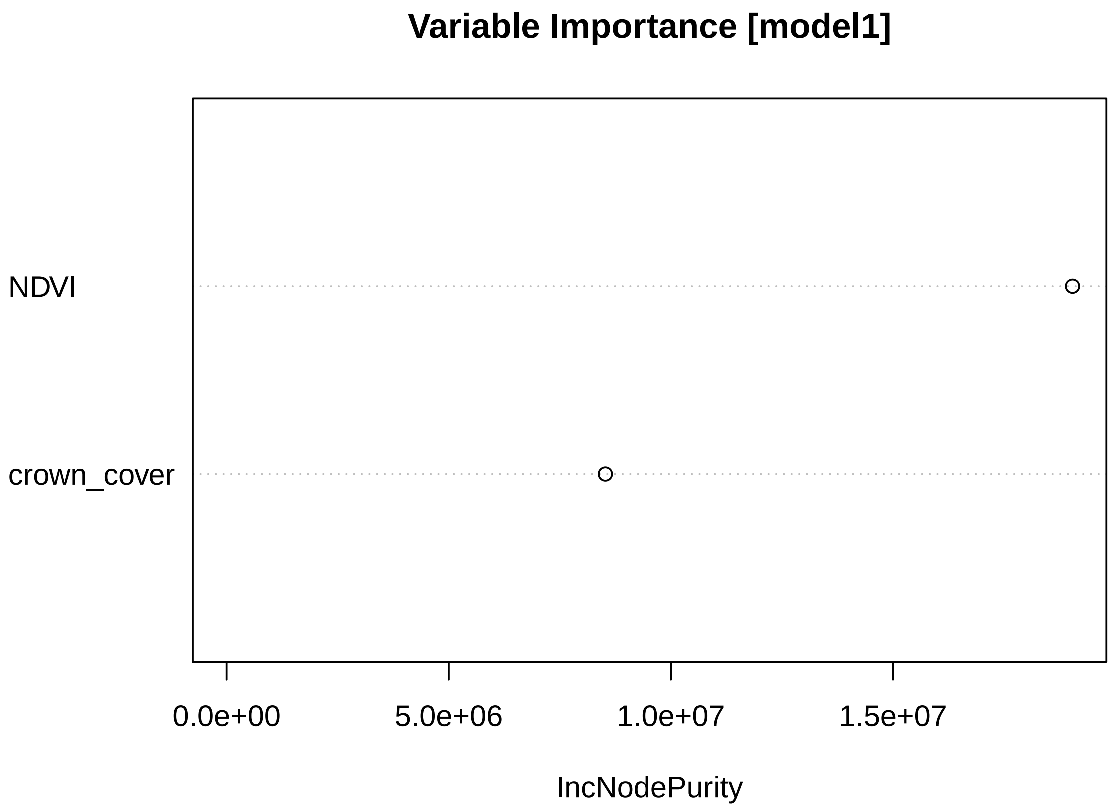
```
str(model3, max.level=2)
```

```
## List of 17
##  $ call           : language randomForest(x = d[, c(6:7)], y = d[, "shrubs_all"])
##  $ type           : chr "regression"
```

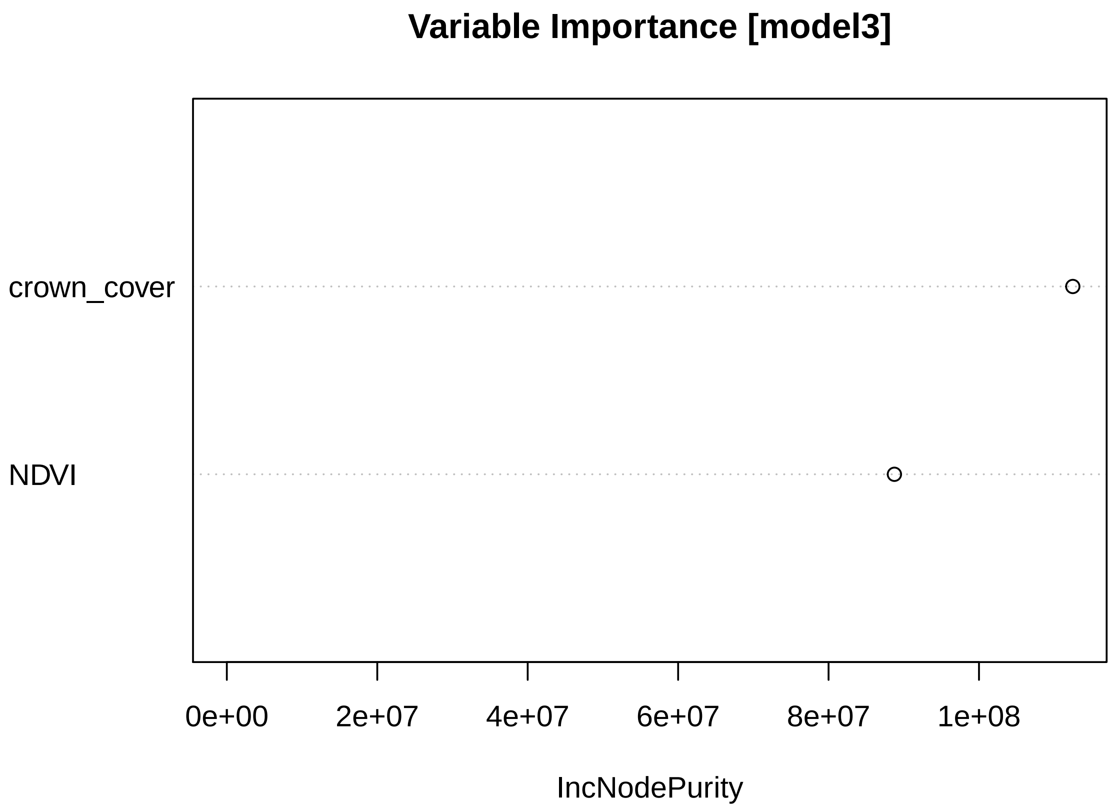
```
## $ predicted      : Named num [1:11] 5993 4224 2666 3642 2271 ...
##  ..- attr(*, "names")= chr [1:11] "1" "4" "5" "7" ...
## $ mse            : num [1:500] 65712656 39118979 34747932 30240141 30963518 ...
## $ rsq            : num [1:500] -1.682 -0.596 -0.418 -0.234 -0.264 ...
## $ oob.times      : int [1:11] 176 164 167 169 165 178 180 178 177 166 ...
## $ importance     : num [1:2, 1] 1.12e+08 8.88e+07
##  ..- attr(*, "dimnames")=List of 2
## $ importanceSD   : NULL
## $ localImportance: NULL
## $ proximity      : NULL
## $ ntree          : num 500
## $ mtry           : num 1
## $ forest         :List of 11
##  ..$ ndbigtree    : int [1:500] 5 5 7 7 5 9 7 5 3 5 ...
##  ..$ nodestatus   : int [1:11, 1:500] -3 -1 -3 -1 -1 0 0 0 0 0 ...
##  ..$ leftDaughter : int [1:11, 1:500] 2 0 4 0 0 0 0 0 0 0 ...
##  ..$ rightDaughter: int [1:11, 1:500] 3 0 5 0 0 0 0 0 0 0 ...
##  ..$ nodepred     : num [1:11, 1:500] 3243 1717 3816 2550 7612 ...
##  ..$ bestvar      : int [1:11, 1:500] 1 0 2 0 0 0 0 0 0 0 ...
##  ..$ xbestsplit   : num [1:11, 1:500] 1.5 0 0.257 0 0 ...
##  ..$ ncat         : Named int [1:2] 1 1
##  .. ..- attr(*, "names")= chr [1:2] "crown_cover" "NDVI"
##  ..$ nrnodes      : int 11
##  ..$ ntree        : num 500
##  ..$ xlevels      :List of 2
## $ coefs          : NULL
## $ y              : num [1:11] 10100 2925 1500 1725 1475 ...
## $ test           : NULL
## $ inbag          : NULL
## - attr(*, "class")= chr "randomForest"
```

Plot variable importance

```
varImpPlot(model1, main="Variable Importance [model1]")
```

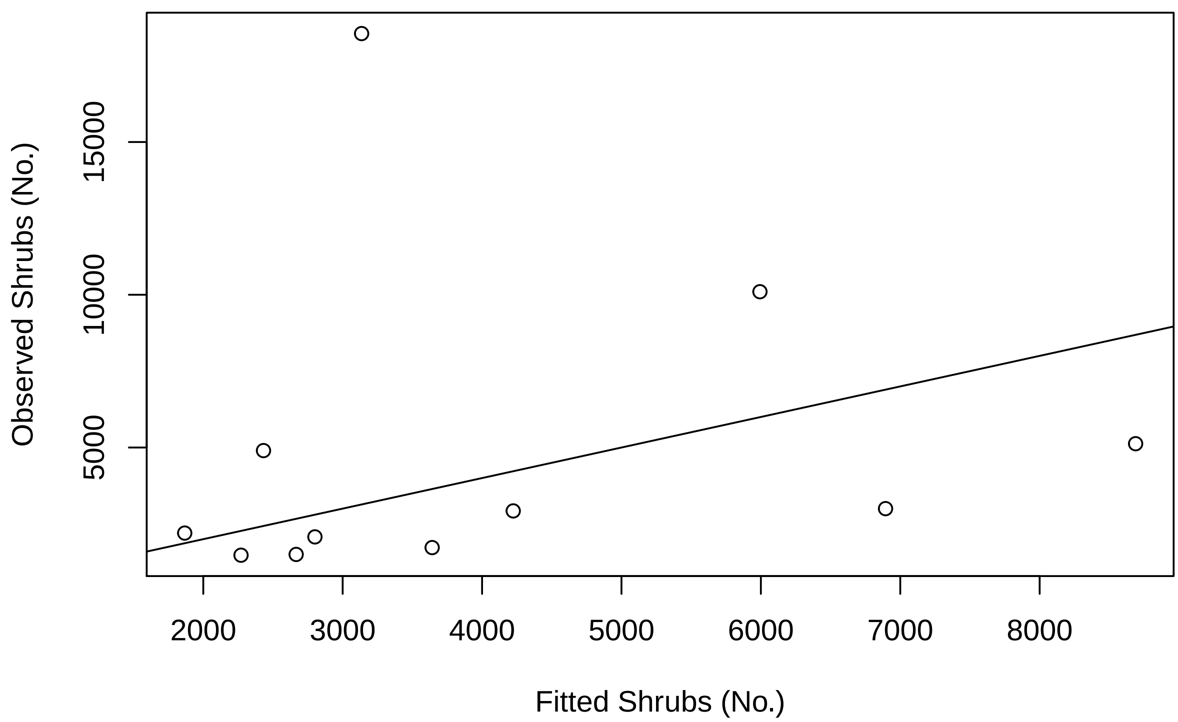



```r
varImpPlot(model3, main="Variable Importance [model3]")
```



Plot correlation

```
plot(model3$predicted, model3$y, xlab="Fitted Shrubs (No.)",
     ylab="Observed Shrubs (No.)")
abline(0,1)
```



Round the r2 value

```
round(cor(model1$predicted,model1$y)**2,3)
```

```
## [1] 0.161
```

```
round(cor(model3$predicted,model3$y)**2,3)
```

```
## [1] 0.016
```

Let's look at the root mean squared error

```
round(sqrt(mean((model1$predicted-model1$y)**2)), digits=1)
```

```
## [1] 1705.5
```

```
round(sqrt(mean((model3$predicted-model3$y)**2)), digits=1)
```

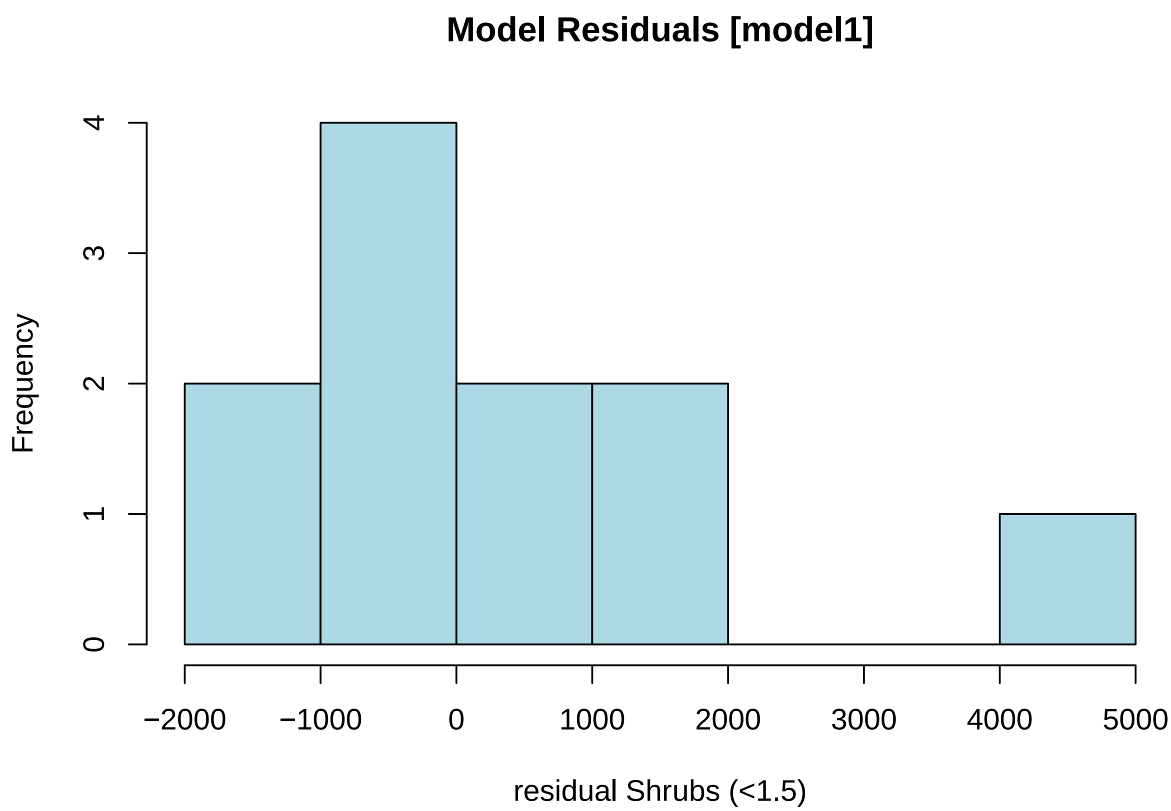
```
## [1] 5191
```

Compute residuals

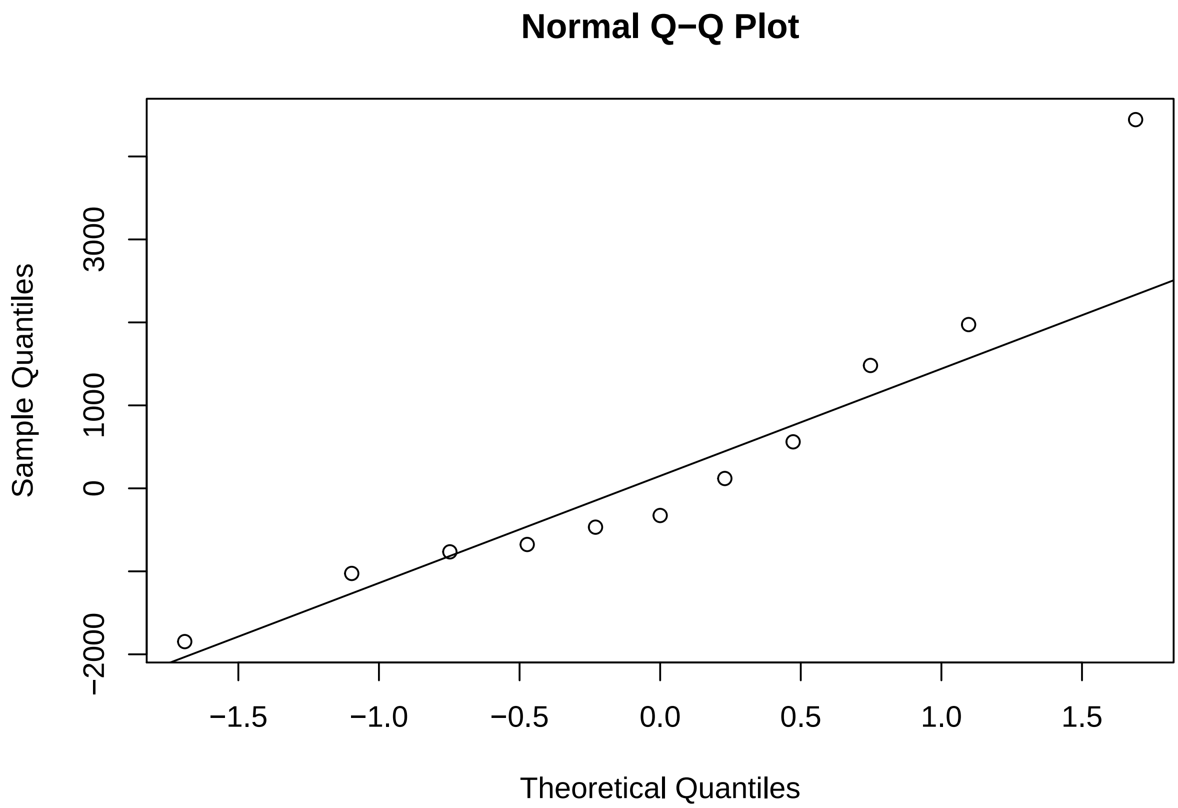
```
d$resid.rf1 <- model1$y - model1$predicted
d$resid.rf3 <- model3$y - model3$predicted
```

Check residual distribution

```r
hist(d$resid.rf1, col = "lightblue", main = "Model Residuals [model1]", xlab =
       "residual Shrubs (<1.5)")
```

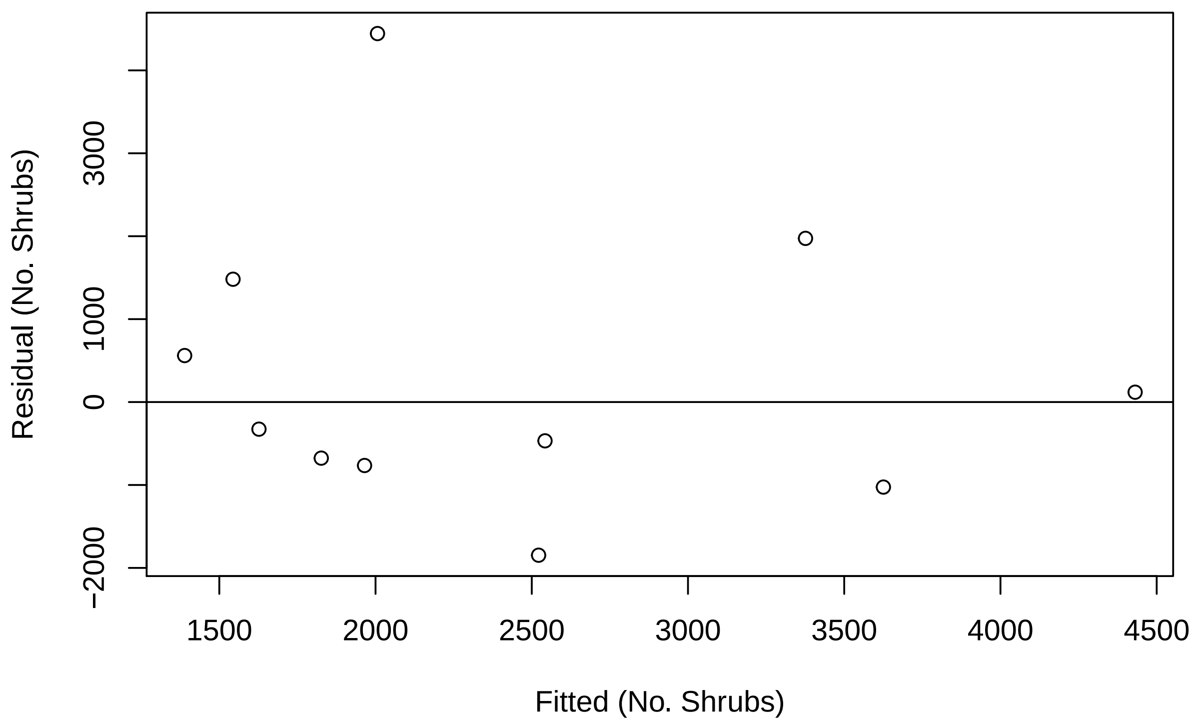



```r
qqnorm(d$resid.rf1)
qqline(d$resid.rf1)
```



Plot residuals

```
plot(model1$predicted, d$resid.rf1, xlab = "Fitted (No. Shrubs)",
     ylab = "Residual (No. Shrubs)")
abline(0,0)
```



Next, before you predict the models, you can print out the RMSE and Rsquared . R squared is a number that indicates the proportion of the variance in the dependent variable that is predictable from the independent variable.

```
print(model1)
```

```
##
## Call:
##  randomForest(x = d[, c(6:7)], y = d[, "shrubs_less_1.5"])
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 2908675
##                     % Var explained: 11.75
```

```
print(model3)
```

```
##
## Call:
##  randomForest(x = d[, c(6:7)], y = d[, "shrubs_all"])
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
```

```
##
##           Mean of squared residuals: 26946172
##                     % Var explained: -9.96
```

Use the ‘predict’ command to make rasters with predictions from the fitted models. To speed up computations use the ‘clusterR’ function from the ‘raster’ package which supports multi-core computing for functions such as predict (NB: install ‘snow’ package).

```
beginCluster()
```

```
## 4 cores detected, using 3
```

```
prediction1 <- clusterR(covs, raster::predict, args = list(model = model1))
prediction3 <- clusterR(covs, raster::predict, args = list(model = model3))
endCluster()
```

Compute the density for shrubs above 1.5m. We can do this by substracting shrubs less than 1.5m from all shrubs.

```
prediction2 <- prediction3 - prediction1
```

Multiply the output rasters by 25 to convert the units from shrubs/0.04ha to shrubs/1ha, round raster values to whole numbers and save the predicted images as GeoTIFFs.

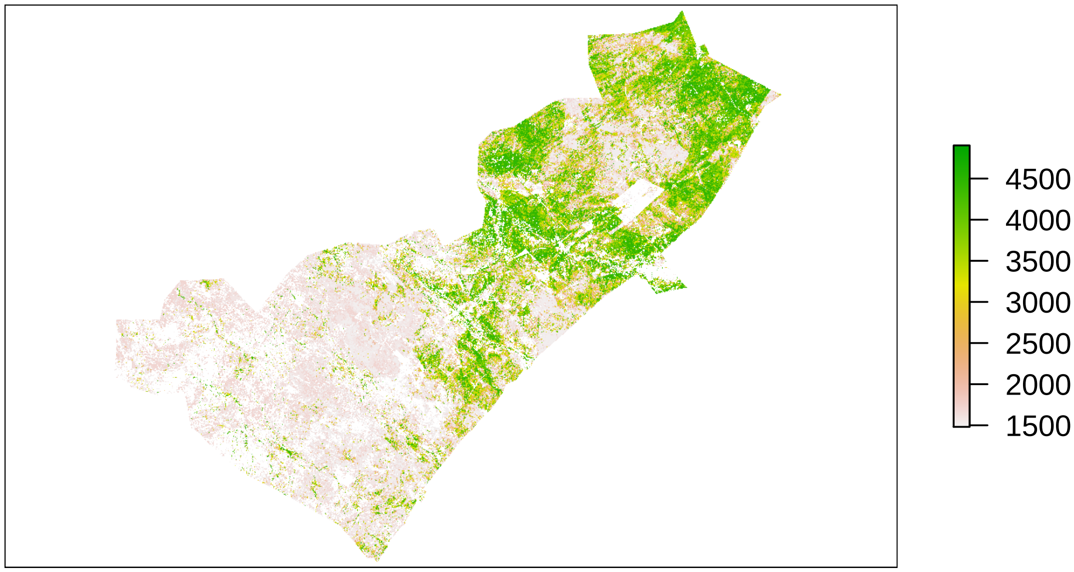
```
prediction1<-round(prediction1, digits = 0)
prediction2<-round(prediction2, digits = 0)
prediction3<-round(prediction3, digits = 0)
writeRaster(prediction1, "otji_bd1.tif", overwrite=TRUE)
writeRaster(prediction2, "otji_bd2.tif", overwrite=TRUE)
writeRaster(prediction3, "otji_bd3.tif", overwrite=TRUE)
```

Results

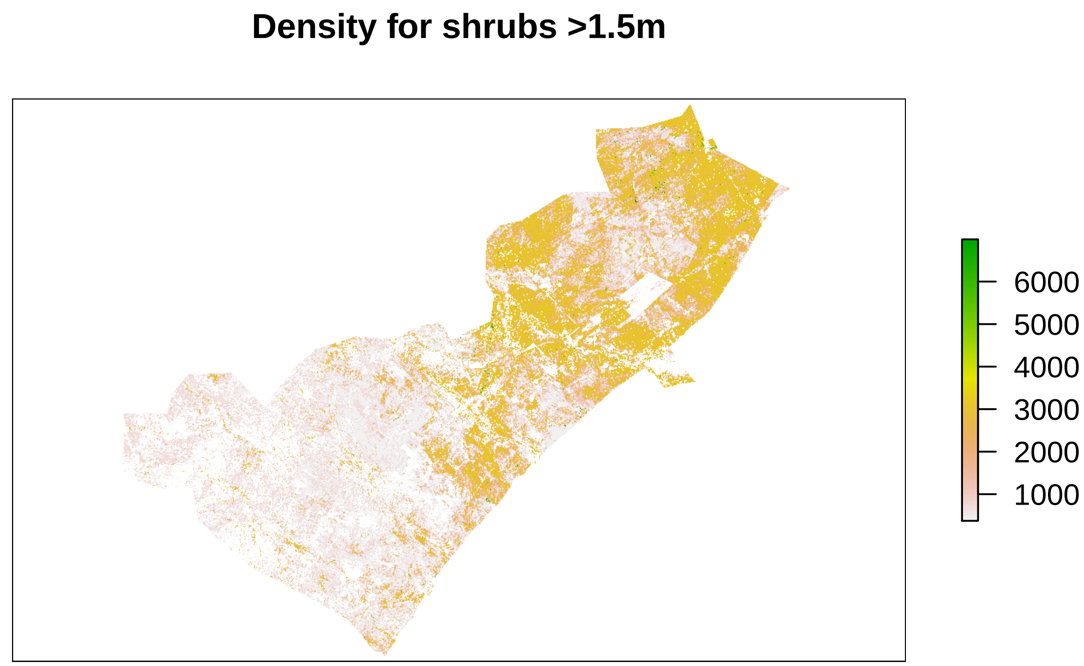
Plot the three maps.

```
plot(prediction1, main="Density for shrubs <1.5m", axes=FALSE)
```

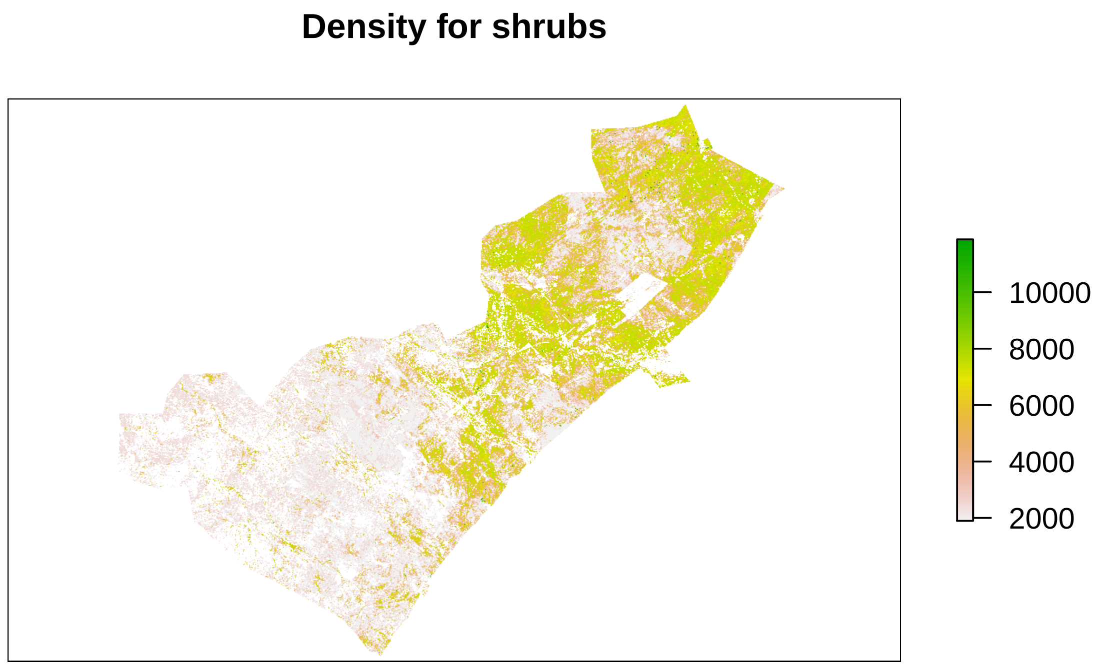
Density for shrubs <1.5m




```
plot(prediction2, main="Density for shrubs >1.5m", axes=FALSE)
```




```
plot(prediction3, main="Density for shrubs", axes=FALSE)
```



Finally, check the amount of time you spent conducting this analysis

```
timeDiff <- Sys.time() - startTime
cat("\nProcessing time", format(timeDiff), "\n")
```

```
##
## Processing time 5.970125 mins
```