Food and Agriculture Organization of the United Nations

AFACI Asian Food & Agriculture Cooperation Initiative

Rural Development Administration

EduSoils

Random Forest

Machine learning for soil mapping

GLOBAL SOIL PARTNERSHIP

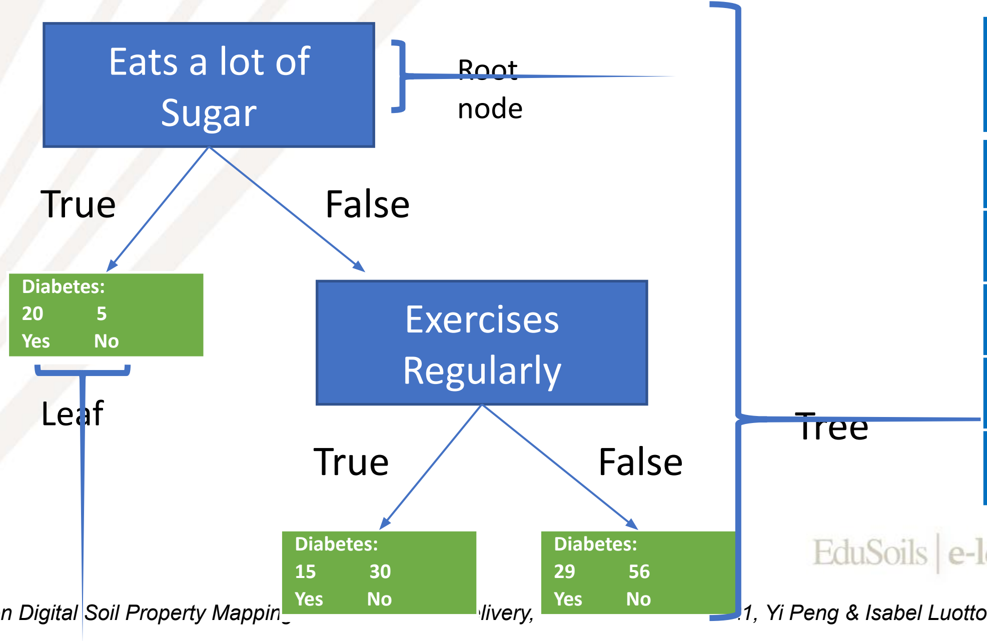
International Training on Digital Soil Property Mapping and Information Delivery, 15-19 November 2021, Yi Peng & Isabel Luotto

Random Forest

- Random forest is a type of machine learning algorithm
- Machine learning algorithms build a model based on training data in order to make predictions without being explicitly programmed to perform the task
- Machine learning methods represent a branch of statistics that can be used to automatically extract information from available data, including the **non-linear** and **hidden** relationships
- It belongs to the decision-tree class of models
- This method is suitable for digital soil mapping under limited and sparse scenarios of data availability

Decision trees

- Random forests are composed of decision trees
- Figuratively speaking the trees are upside down, with the root node (the first variable) at the top and the leaves (the predictions) at the bottom



Eats a lot of Sugar

Root node

True

False

Diabetes:
20 Yes 5 No

Leaf

Exercises Regularly

True

False

Diabetes:
15 Yes 30 No

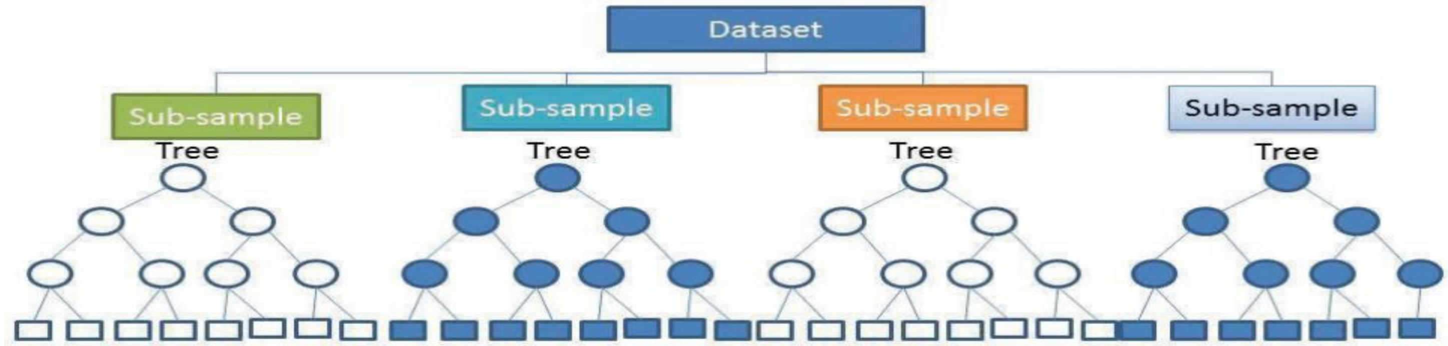
Diabetes:
29 Yes 56 No

Tree

Sugar	Exercise	Diabetes
Yes	No	Yes
Yes	Yes	No
No	Yes	No
No	No	No
Yes	No	?

Random Forest

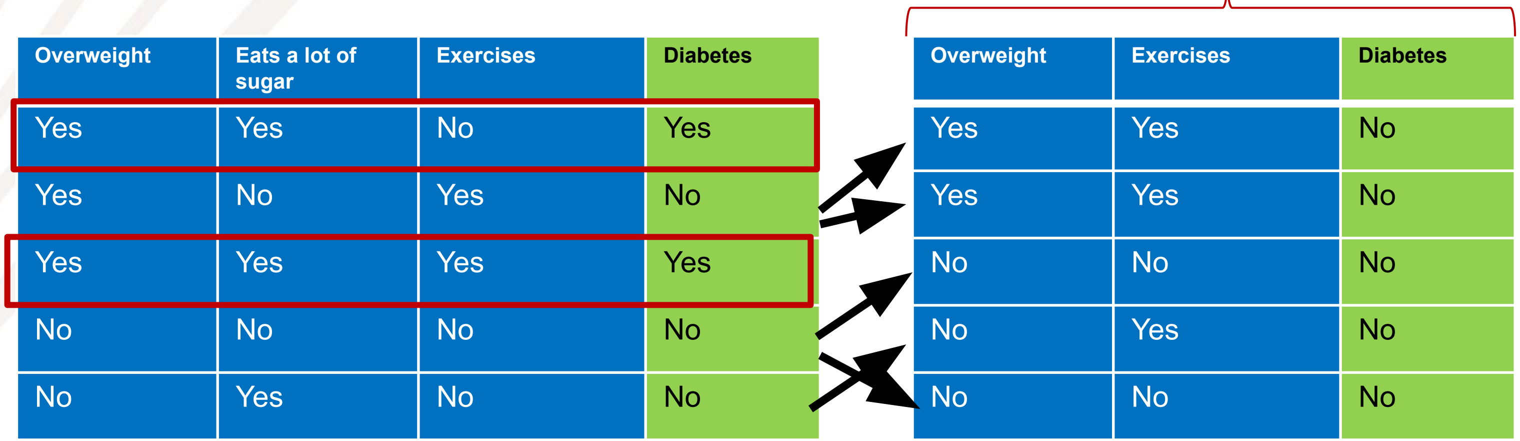
- Random forests are composed of hundreds of these decision trees consisting of randomly selected predicting variables and randomly selected subsamples
- Each single tree makes a prediction and the final average is obtained by taking the average of each tree



Random Forest - Bootstrapping

- Each tree (generated using a different subset of available data and random combinations of the prediction factors) is internally evaluated by an **out-of-bag** cross validation

These two data sets were not used to build the tree and will be used for validation



Training subset (bootstrapping)

Overweight	Eats a lot of sugar	Exercises	Diabetes
Yes	Yes	No	Yes
Yes	No	Yes	No
Yes	Yes	Yes	Yes
No	No	No	No
No	Yes	No	No

Overweight	Exercises	Diabetes
Yes	Yes	No
Yes	Yes	No
No	No	No
No	Yes	No
No	No	No

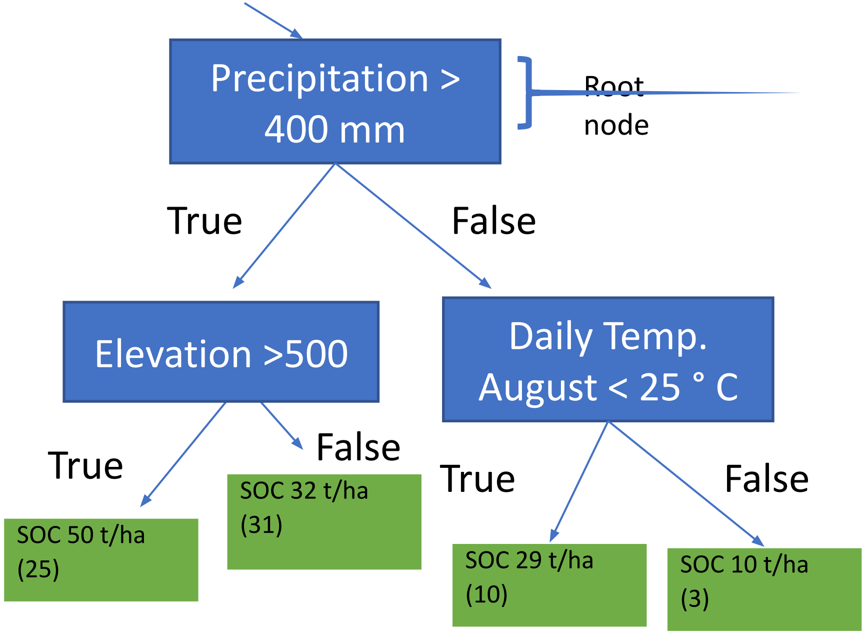
The trees in a Random Forest

This Photo by Unknown Author is licensed under CC BY

- Random forests are used both for predicting categorical outcomes (e.g. to diagnose medical conditions) and for predicting continuous data like Soil Organic Carbon
- Classification trees are used to predict categorical data
- Regression trees are used for continuous data by recursively splitting the data

Random Forest for continuous data

- In linear regression a model is fitted to the entire dataset □it cannot detect complex relationships
- In a regression tree the data is split into specific groups that correspond to certain thresholds of predictors (covariates) of each tree
- The prediction of the single tree is made based on the mean of the observed samples in the leaf
- The prediction of the random forest is made by taking the average of the predictions of the single trees



( ) → number of observations in each final split

EduSoils | e-learning soil educational platform

GLOBAL SOIL PARTNERSHIP

Random Forest workflow:

1. Get the previously prepared covariates and data to create a regression matrix
2. Create a random forest model and explore parameters
3. Explore which variables are more relevant in the model
4. Predict SOC with our random forest model

Random Forest -Data prep

```
# Set working directory
setwd("C:/Users/hp/Documents/FAO/EduSoils/AFACI_training/Training_material")

# Load the covariates stack. It was was prepared in the
#'data_preparation_covariates' script
load(file = "02-Outputs/covariates.RData")
names(covs)

# Load the processed data for digital soil mapping. This table was #prepared in the
'data_preparation_profiles' script
dat <- read.csv("02-Outputs/dat_train.csv")
names(dat)
```

Random Forest - Data prep

```r
# extract values from covariates to the soil points
coordinates(dat) <- ~ X + Y
dat <- extract(x = covs, y = dat, sp = TRUE)
summary(dat)
# Remove NA values
dat<-as.data.frame(dat)
dat <- dat[complete.cases(dat),]
str(dat)
# LandCover and soilmap are categorical variables, they need to be #'factor' type
dat$LandCover <- as.factor(dat$LandCover)
dat$soilmap <- as.factor(dat$soilmap)
str(dat)
```

Random Forest

```r
# Save the final table and all the covariates
write.csv(dat, "02-Outputs/SOC_RegMatrix.csv", row.names = FALSE)
```

- Due to the randomness component of Random Forest we’re going to use the set.seed() function

```r
# Define the random numbers table (to get reproducible result)
set.seed(12042019)
```

Random Forest

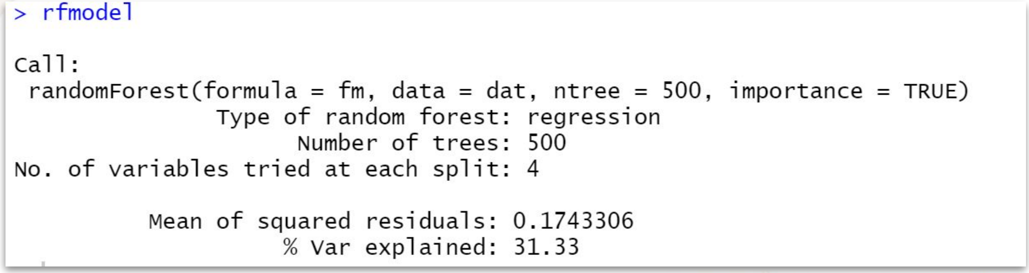
- Now we’re going to transform our data frame into a SpatialPointsDataFrame and set the coordinate system to WGS84
- To use the randomForest() function we first need to define a formula of the data and covariates with the function fm()

```
library(sp)
# Promote to spatialPointsDataFrame and set the coordinate system
coordinates(dat) <- ~ X + Y
proj4string(dat) = CRS("+init=epsg:4326") # WGS84 ;names(data)
# We need to define a formula for the model
fm = as.formula(paste("OCSlog ~", paste0(names(covs),
                                              collapse = "+")))
fm #check the model
```

Random Forest - model

- Now we’re going to create the random forest model with the randomForest() function

```
# Run the Random Forest model and explore the results
library(randomForest)

rfmodel <- randomForest(fm, data=dat, ntree=500, importance=TRUE)
rfmodel
```

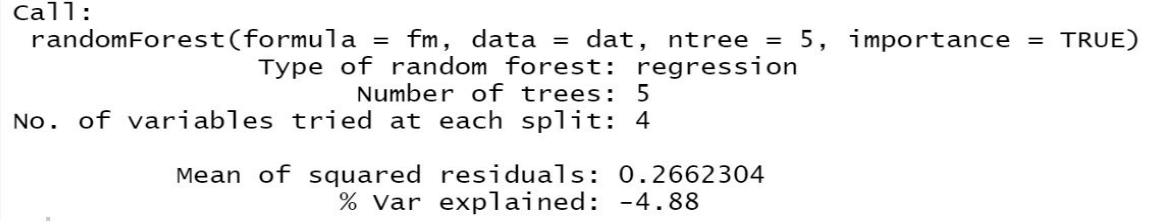


```
> rfmodel

Call:
 randomForest(formula = fm, data = dat, ntree = 500, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 4

          Mean of squared residuals: 0.1743306
                    % Var explained: 31.33
```

Explore the model

- Now we will explore what happens when we change the number of decision trees within out random forest
- The default of 500 trees produces a better model than a random forest with only 5, please change it back to 500
- If we were using another package (caret) we could change mtry (number of variables per tree)

```
rfmodel <- randomForest(fm, data=dat, ntree=5, importance=TRUE)

rfmodel
```



```
Call:
 randomForest(formula = fm, data = dat, ntree = 5, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 5
No. of variables tried at each split: 4

          Mean of squared residuals: 0.2662304
                    % Var explained: -4.88
```

ntree parameter

```
plot(rfmodel)
```

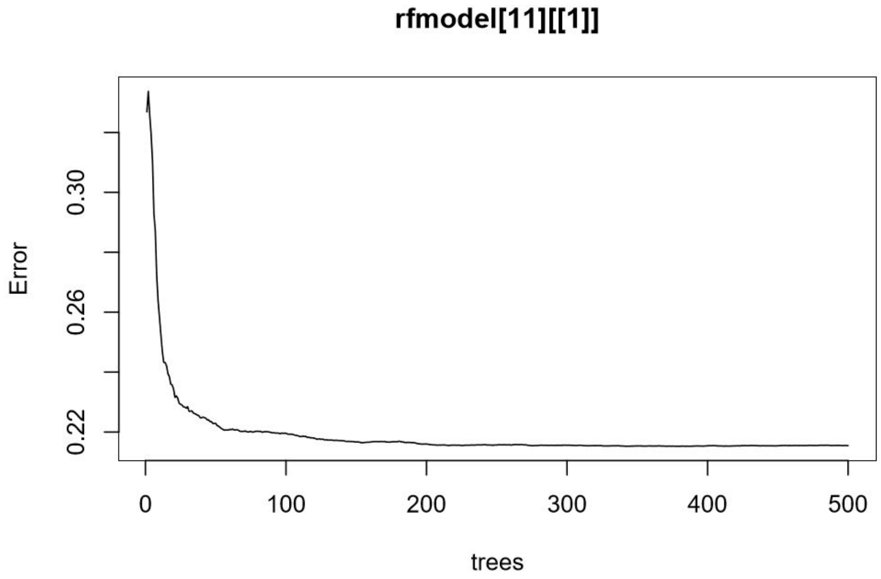


Figure 6.18: Select ntree

lucational platform

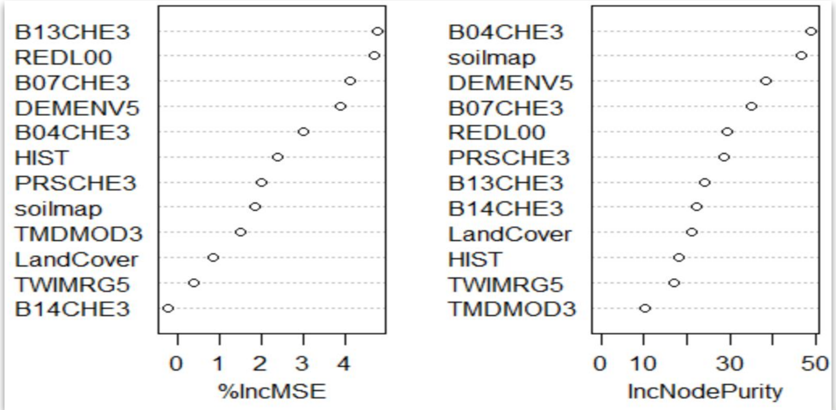
Explore the model

- The default of 500 trees produces a better model than a random forest with only 5, please change it back to 500

```
rfmodel <- randomForest(fm, data=dat, ntree=500, importance=TRUE)
rfmodel
```

Covariate Importance

- Now we will explore which covariate is more relevant within the model with the varImpPlot() function
- The first graph shows how the mean squared error increases within the entire model if a covariate is excluded
- The second graph shows which covariate is more decisive for splitting the data into homogenous data groups (measured with residual sum of squares (RSS)

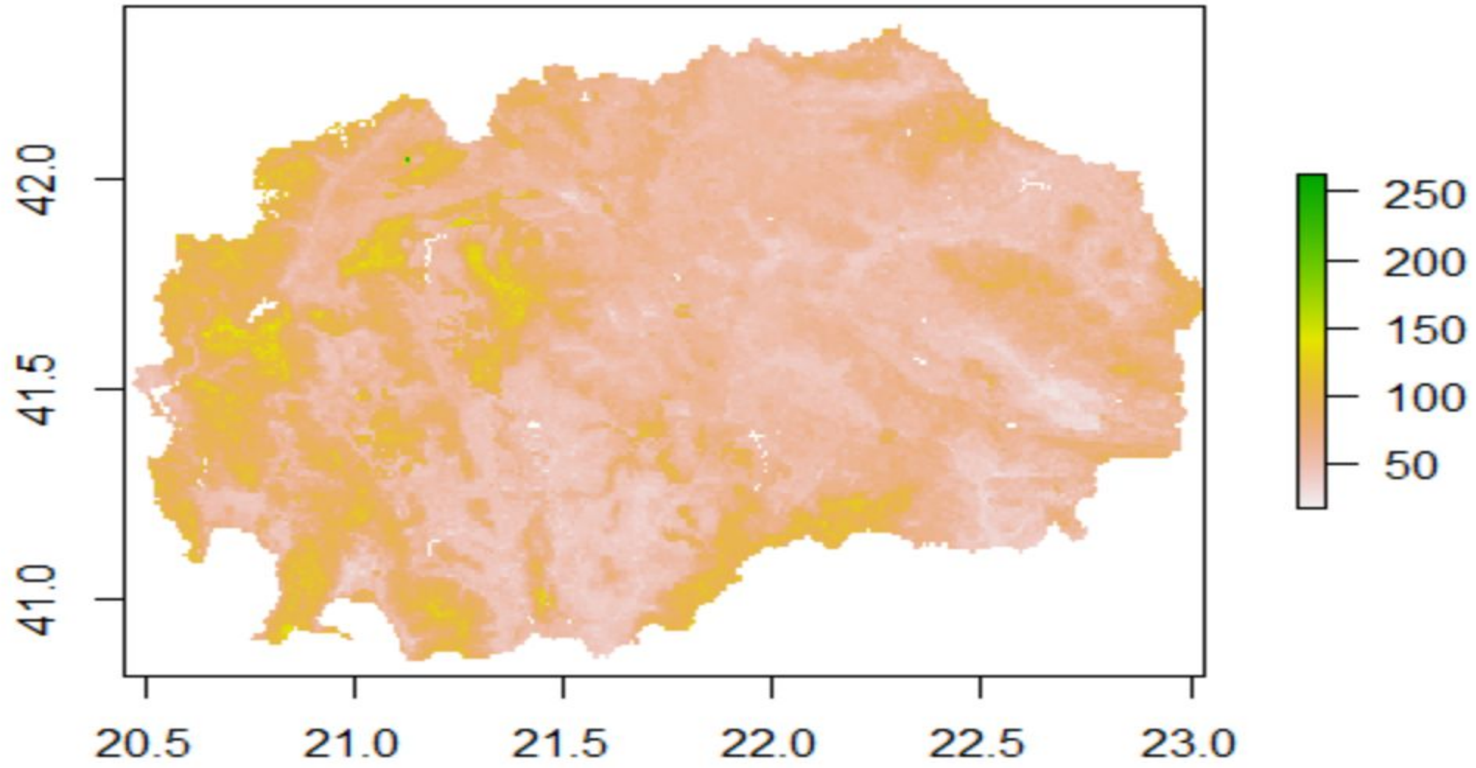


```
# Explore the importance of
#covariates in the model
varImpPlot(rfmodel)
```

Predict SOC with RF model

- Finally we will run our model to predict SOC
- A uncertainty map can be created by using quantile regression forest (for more info refer to the SOC mapping cookbook)

```
# Make a prediction across all Macedonia
pred <- predict(covs, rfmodel)


# Back transform predictions log transformed
pred <- exp(pred)


# Explore and save the result as a tiff file
plot(pred)
writeRaster(pred, filename = "02-Outputs/Final Maps/MKD_OCS_RF.tif", overwrite=TRUE)
```


42.0
41.5
41.0
20.5
21.0
21.5
22.0
22.5
23.0
250
200
150
100
50