```
#GBIF DATA ACCESSED ON 2023-05-22
#GBIF data corresponding to this work are available at DOI: https://doi.org/10.15468/dd.zc83q5

#------------------------------------------------------------------------------
#Title: "Future increase of filamentous cyanobacteria in coastal Baltic Sea predicted by
multiple realm models-
#of marine, terrestrial, and climate change scenarios"
#Author: "Abdelgadir et al.
#The R code file contains Ensemble (ESDM) and stacked species modeling (SSDM) of Cyanobacteria
using stacked multiple realm layers
#------------------------------------------------------------------------------

#######################################
## Upload required packages
#######################################
#BEFORE STARTING:
#Note: replace the working directory folder "R-Directory" (~\\R\\R-Directory\\) with your
desired working path.

#Use Xmx flag to controls the size of the Java runtime heap. This will allow your R code to run
without hitting the ceiling.
#In other word, it specifies the maximum memory allocation pool for a Java Virtual Machine (JVM)
#The memory flag can also be specified in different sizes, such as kilobytes (k), megabytes (m),
and gigabytes (g)
#The following code will increase the heap only for the Java process called by your R script.

options(java.parameters = "-Xmx4g") # or 8g, or larger


library(rgbif)
library(xlsx)
library(XLConnect)
library(XLConnectJars)
library(lme4)
library(car)
library(rJava)
library(xlsx)
library(readxl)
library(ggplot2)
library(sf)
library(rnaturalearth)
library(rnaturalearthdata)
library(ggspatial)
library(sdm)
library(gbm)
library(tree)
library(shiny)
library(usdm)
library(dismo)
library(scrubr)
library(maps)
#library(Rcmdr)
#library(mapview)
library(biomod2)
library(writexl)
library(mapdata)
library(sdmpredictors)
library(SSDM)
library(raster)
library(sp)
library(terra)


#######################################
## DOWNLOAD AND CLEAN DATA FROM GBIF
#######################################

#------------------------------------------------------------------------------
```

```
#myspecies: Nodularia spumigena, Aphanizomenon sp., and Dolichospermum spp.,
#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
#Download occurance records for Nodularia spumigena
#-------------------------------------------------------------------------------

# Download GBIF occurrence data for this species. Set searching limit to 10000 records and
spatial extent to the Baltic Sea: 10, 30,  53, 66
myspecies <- c('Nodularia spumigena')
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE, limit = 10000,
decimalLongitude = "10, 30", decimalLatitude = "53, 66")

# Take a look at the downloaded data:

gbif_data


# Get the DOI for citing these data properly:

gbif_citation(gbif_data)


# Get the columns that matter for mapping and cleaning the occurrence data:
myspecies_coords <- gbif_data$data[ , c("decimalLongitude",
"decimalLatitude","scientificName","occurrenceStatus")]

####################################
#CLEAN THE DATASET!
####################################

# Mind that data often contain errors, so careful inspection and cleaning are necessary!
# Here we'll first remove records of absence or zero-abundance (if any):
names(myspecies_coords)
sort(unique(myspecies_coords$individualCount))  # notice if some points correspond to zero
abundance
sort(unique(myspecies_coords$occurrenceStatus))  # check for different indications of "absent",
which could be in different languages! and remember that R is case-sensitive
absence_rows <- which(myspecies_coords$individualCount == 0 | myspecies_coords$occurrenceStatus
%in% c("absent", "Absent", "ABSENT", "ausente", "Ausente", "AUSENTE"))
length(absence_rows)
if (length(absence_rows) > 0) {
  myspecies_coords <- myspecies_coords[-absence_rows, ]
}

#Let's do some further data cleaning with functions of the 'scrubr' package
nrow(myspecies_coords)
myspecies_coords <-
coord_incomplete(coord_imprecise(coord_impossible(coord_unlikely(myspecies_coords))))
nrow(myspecies_coords)

####################################
#Remove duplicates
####################################

dups <- duplicated(myspecies_coords[,c("decimalLatitude","decimalLongitude")])

#Check how many duplicates did you get?

sum(dups)

#Remove
myspecies_coords <-myspecies_coords[!dups,]

#Check final data
myspecies_coords
########################################

#Convert to excel file
```

```
write_xlsx(myspecies_coords, "~\\R\\R-Directory\\Nodularia spumigena.xlsx")


####################################
#Map the cleaned occurrence data
####################################

world <- rnaturalearth::ne_countries(scale = "medium", returnclass = "sf")
class(world)

ggplot(data=world)+
  geom_sf()+
  geom_point(data = myspecies_coords, mapping = aes(x = decimalLongitude  , y = decimalLatitude
), colour = "red",size = 1) +
  annotation_scale(location = "br", width_hint = 0.2)+
  annotation_north_arrow(location = "br", width_north = "true",pad_x = unit(0.1,"in"),pad_y =
unit(0.5,"in"),style = north_arrow_orienteering)+
  coord_sf(xlim = c(10, 30), ylim = c(53, 66),expand = FALSE)

#############################################################################################

#-----------------------------------------------------------------------
#Download occurance records for Aphanizomenon sp.
#-----------------------------------------------------------------------

# Download GBIF occurrence data for this species. Set searching limit to 10000 records and
spatial extent to the Baltic Sea: 10, 30,  53, 66
myspecies <- c('Aphanizomenon sp.')
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE, limit = 10000,
decimalLongitude = "10, 30", decimalLatitude = "53, 66")

# Take a look at the downloaded data:

gbif_data


# Get the DOI for citing these data properly:

gbif_citation(gbif_data)


# Get the columns that matter for mapping and cleaning the occurrence data:
myspecies_coords <- gbif_data$data[ , c("decimalLongitude",
"decimalLatitude","scientificName","occurrenceStatus")]

####################################
#CLEAN THE DATASET!
####################################

# Mind that data often contain errors, so careful inspection and cleaning are necessary!
# Here we'll first remove records of absence or zero-abundance (if any):
names(myspecies_coords)
sort(unique(myspecies_coords$individualCount))  # notice if some points correspond to zero
abundance
sort(unique(myspecies_coords$occurrenceStatus))  # check for different indications of "absent",
which could be in different languages! and remember that R is case-sensitive
absence_rows <- which(myspecies_coords$individualCount == 0 | myspecies_coords$occurrenceStatus
%in% c("absent", "Absent", "ABSENT", "ausente", "Ausente", "AUSENTE"))
length(absence_rows)
if (length(absence_rows) > 0) {
  myspecies_coords <- myspecies_coords[-absence_rows, ]
}

#Let's do some further data cleaning with functions of the 'scrubr' package
nrow(myspecies_coords)
myspecies_coords <-
coord_incomplete(coord_imprecise(coord_impossible(coord_unlikely(myspecies_coords))))
nrow(myspecies_coords)
```

```
####################################
#Remove duplicates
####################################

dups <- duplicated(myspecies_coords[,c("decimalLatitude","decimalLongitude")])

#Check how many duplicates did you get?

sum(dups)

#Remove
myspecies_coords <-myspecies_coords[!dups,]

#Check final data
myspecies_coords
########################################

#Convert to excel file
write_xlsx(myspecies_coords, "~\\R\\R-Directory\\Aphanizomenon sp.xlsx")


####################################
#Map the cleaned occurrence data
####################################

world <- rnaturalearth::ne_countries(scale = "medium", returnclass = "sf")
class(world)

ggplot(data=world)+
  geom_sf()+
  geom_point(data = myspecies_coords, mapping = aes(x = decimalLongitude  , y = decimalLatitude
), colour = "red",size = 1) +
  annotation_scale(location = "br", width_hint = 0.2)+
  annotation_north_arrow(location = "br", width_north = "true",pad_x = unit(0.1,"in"),pad_y =
unit(0.5,"in"),style = north_arrow_orienteering)+
  coord_sf(xlim = c(10, 30), ylim = c(53, 66),expand = FALSE)


#############################################################################################

#-----------------------------------------------------------------------------
#Download occurrence records for Dolichospermum spp.
#-----------------------------------------------------------------------------
# Download GBIF occurrence data for this species. Set searching limit to 10000 records and
spatial extent to the Baltic Sea: 10, 30,  53, 66
myspecies <- c('Dolichospermum spp.')
gbif_data <- occ_data(scientificName = myspecies, hasCoordinate = TRUE, limit = 10000,
decimalLongitude = "10, 30", decimalLatitude = "53, 66")

# Take a look at the downloaded data:

gbif_data


# Get the DOI for citing these data properly:

gbif_citation(gbif_data)


# Get the columns that matter for mapping and cleaning the occurrence data:
myspecies_coords <- gbif_data$data[ , c("decimalLongitude",
"decimalLatitude","scientificName","occurrenceStatus")]

####################################
#CLEAN THE DATASET!
####################################

# Mind that data often contain errors, so careful inspection and cleaning are necessary!
# Here we'll first remove records of absence or zero-abundance (if any):
```

```
names(myspecies_coords)
sort(unique(myspecies_coords$individualCount))  # notice if some points correspond to zero
abundance
sort(unique(myspecies_coords$occurrenceStatus))  # check for different indications of "absent",
which could be in different languages! and remember that R is case-sensitive
absence_rows <- which(myspecies_coords$individualCount == 0 | myspecies_coords$occurrenceStatus
%in% c("absent", "Absent", "ABSENT", "ausente", "Ausente", "AUSENTE"))
length(absence_rows)
if (length(absence_rows) > 0) {
  myspecies_coords <- myspecies_coords[-absence_rows, ]
}

#Let's do some further data cleaning with functions of the 'scrubr' package
nrow(myspecies_coords)
myspecies_coords <-
coord_incomplete(coord_imprecise(coord_impossible(coord_unlikely(myspecies_coords))))
nrow(myspecies_coords)

####################################
#Remove duplicates
####################################

dups <- duplicated(myspecies_coords[,c("decimalLatitude","decimalLongitude")])

#Check how many duplicates did you get?

sum(dups)

#Remove
myspecies_coords <-myspecies_coords[!dups,]

#Check final data
myspecies_coords
########################################

#Convert to excel file
write_xlsx(myspecies_coords, "~\\R\\R-Directory\\Dolichospermum spp.xlsx")


####################################
#Map the cleaned occurrence data
####################################

world <- rnaturalearth::ne_countries(scale = "medium", returnclass = "sf")
class(world)

ggplot(data=world)+
  geom_sf()+
  geom_point(data = myspecies_coords, mapping = aes(x = decimalLongitude  , y = decimalLatitude
), colour = "red",size = 1) +
  annotation_scale(location = "br", width_hint = 0.2)+
  annotation_north_arrow(location = "br", width_north = "true",pad_x = unit(0.1,"in"),pad_y =
unit(0.5,"in"),style = north_arrow_orienteering)+
  coord_sf(xlim = c(10, 30), ylim = c(53, 66),expand = FALSE)

#############################################################################################

#From your directory, merge the three excel files (Nodularia spumigena.xlsx, Aphanizomenon
sp.xlsx, and Dolichospermum spp.xlsx)-
#into one file holds three coloumn:species, decimalLongitude and decimalLatitude. Convert the
'xlst' file ('save as' function in excel) to 'csv'.
#Name the file 'combinedDf.csv'

######################
## Import the data file 'combinedDf.csv'from directory
######################

#Choose file #
```

```
combinedDf<- read.csv("~/R/R-Directory/combinedDf.csv")

#-------------------------------------------------------------------------------

#######################
## PLOT MAP ## to see where are the locations of records
#######################

world <- rnaturalearth::ne_countries(scale = "medium", returnclass = "sf")
class(world)

ggplot(data=world)+
  geom_sf()+
  geom_point(data = combinedDf, mapping = aes(x = decimalLongitude  , y = decimalLatitude ),
colour = "blue",size = 0.5) +
  annotation_scale(location = "br", width_hint = 0.2)+
  annotation_north_arrow(location = "br", width_north = "true",pad_x = unit(0.1,"in"),pad_y =
unit(0.5,"in"),style = north_arrow_orienteering)+
  coord_sf(c(10, 30), ylim = c(53, 66),expand = FALSE)
#-------------------------------------------------------------------------------


#######################
##PREPARING AND MAKING A NEW DATAFRAME FOR MODELING  # Let's name it (newssp)
#######################

combinedDf<- read.csv("~/R/R-Directory/combinedDf.csv")

newssp<-combinedDf

#Check the new file

newssp

#Adding one column and name it species
newssp$species<-1
newssp<-newssp[,c("decimalLongitude","decimalLatitude","species")]
head(newssp)
class(newssp)

#Check the new dataframe and make SpatialPointsDataFrame
newssp$decimalLongitude <- as.numeric(newssp$decimalLongitude)
newssp$decimalLatitude <- as.numeric(newssp$decimalLatitude)
coordinates(newssp) = ~ decimalLongitude + decimalLatitude

#newssp = SpatialPoints(newssp, proj4string = CRS("+proj=longlat +datum=WGS84"))

class(newssp)
head(newssp)

newssp
plot(newssp)


#-------------------------------------------------------------------------------
#Make sure those packages are uploaded
library(sdm)
library(gbm)
library(tree)
library(shiny)
library(usdm)
getmethodNames()

#-------------------------------------------------------------------------------

#######################
## DOWNLOADING PREDICTORS from Bio-Oracle 2.2 database: https://bio-oracle.org/ -
# and Bio-climatic variables from WorldClim database: https://www.worldclim.org/data/bioclim
#######################
```

```
#Creat a folder in a desired locatin where you can save all raster layers (to avoid re-
downloading data in next sessions).
# Download rasters to sdmpredictors/Meta folder and import into R

options(sdmpredictors_datadir = "C:/R-4.0.2/library/sdmpredictors/Meta/")

#First: upload the marine raster layers. Name them mar1 and mar2

mar1 <- load_layers( layercodes = c('BO22_parmax','BO22_parmean','BO22_nitratemax_ss',
                                    'BO22_nitratemean_ss','BO22_nitratemin_ss',
                                    'BO22_nitraterange_ss', 'BO22_nitrateltmin_ss',
                                    'BO22_nitrateltmax_ss','BO22_phosphatemax_ss',
                                    'BO22_phosphatemean_ss','BO22_phosphatemin_ss',
                                    'BO22_phosphaterange_ss','BO22_phosphateltmax_ss',
                                    'BO22_phosphateltmin_ss','BO22_ppmax_ss','BO22_ppmean_ss',
                                    'BO22_ppltmin_ss','BO22_ppmin_ss','BO22_pprange_ss',
                                    'BO22_ppltmax_ss','BO22_salinityltmax_ss',
                                    'BO22_salinityltmin_ss','BO22_salinitymax_ss',
                                    'BO22_salinitymean_ss','BO22_salinitymin_ss',
                                    'BO22_salinityrange_ss',
                                    'BO22_dissoxmax_ss','BO22_dissoxmean_ss',
                                    'BO22_dissoxmin_ss','BO22_dissoxrange_ss',
                                    'BO22_dissoxltmax_ss','BO22_dissoxltmin_ss'),
                     equalarea=FALSE, rasterstack=TRUE)

mar2 <- load_layers( layercodes = c("MS_biogeo13_sst_mean_5m","MS_biogeo14_sst_min_5m",
                                    "MS_biogeo15_sst_max_5m","MS_biogeo16_sst_range_5m",
                                    "MS_biogeo17_sst_variance_5m","MS_bathy_5m"),
                     equalarea=FALSE, rasterstack=TRUE)

#Stack mar1 and mar2

mar= stack(mar1,mar2)

#Secondly: upload the terrestrial raster layers. Name it (terr)

terr <- load_layers( layercodes = c("WC_bio1","WC_bio5","WC_bio6","WC_bio7",
                                    "WC_bio8","WC_bio9","WC_bio10","WC_bio11",
                                    "WC_bio12","WC_bio13","WC_bio14","WC_bio16",
                                    "WC_bio17","WC_bio18","WC_bio19",
                                    "WC_alt"),equalarea=FALSE,rasterstack=TRUE)


############################################################
#Resample the marine by the terrestrial layers and stack into one raster stack
#Here we will make two raster stacks: marine by the terrestrial (mar/terr) and terrestrial by
the marine (terr/mar)
#Method of resampling is nearest neighbor "ngb"
############################################################

t<-extent(-180, 180, -90, 90) #layer extent from terrestrial stack
m<-extent(-180, 180, -90, 90) #layer extent from marine stack
#no need to edit the following 6 lines
extent_list<-list(t, m)
extent_list<-lapply(extent_list, as.matrix)
matrix_extent<-matrix(unlist(extent_list), ncol=length(extent_list))
rownames(matrix_extent)<-c("xmin", "ymin", "xmax", "ymax")
best_extent<-extent(min(matrix_extent[1,]), max(matrix_extent[3,]), min(matrix_extent[2,]),
max(matrix_extent[4,]))
ranges<-apply(as.matrix(best_extent), 1, diff)
reso<-res(mar) #choose layer you want to keep resolution
nrow_ncol<-ranges/reso
s2<-raster(best_extent, nrows=nrow_ncol[2], ncols=nrow_ncol[1], crs=mar@crs) #choose layer crs
you want to keep

t.2 <-resample(terr, s2, method="ngb") #resample terr by s2
m.2 <-resample(mar, s2, method="ngb") #resample mar by s2
```

```
#Create mar/terr
env1=stack(m.2, t.2) #stack resampled layers

#Create terr/mar
env2=stack(t.2, m.2) #stack resampled layers

#Check raster stacks
env1
env2

############################################################
#Crop the raster layers to study extent of the Baltic Sea
############################################################

# Set study extent (# 10, 30,  53, 66 (Baltic Sea))
SA.ext <- extent(10, 30, 53, 66)

# Crop raster layer by extent & plot. Name the current raster stacks as 'env.c1' (for mar/terr)
and 'env.TerrMar' (for terr/mar)
env.c1 <- crop(env1, SA.ext)
env.c2 <- crop(env2, SA.ext)

#Check again the new ratser stacks
env.c1
env.c2

#Plot each raster stack
plot(env.c1)
plot(env.c2)

################################################################
#Now we will start the modeling, first by removing the multicollinearity between predictors.
#We will run two different models using the two raster stacks: mar/terr (env.c1) and terr/mar
(env.c2)
#We will assign one name 'env.c' for the raster stacks.
################################################################


#-------------------------------------------------------------------------------
#We start the first model with "mar/terr" using the raster stack 'env.c1'
#-------------------------------------------------------------------------------

#Assign the name env.c (stands for current marine environmental variables)

env.c<-env.c1

env.c= as.data.frame(env.c)

#######################
## Removing multicollinearity between predictors #VERY IMPORTANT#
#######################

#Variance Inflation Factor and test for multicollinearity using VIF function of usdm package
("vifstep" and "vifcor")-
# (th) stand for correlation threshold.
#Here, any variable with high correlation coefficient (grater than threshold 0.7) will be
removed.

v1c<-vifstep(env.c)
v1c
v2c<-vifcor(env.c,th=0.7)
v2c
env.c<-exclude(env.c,v2c)

#Have a look on the new predictors
env.c

#########################################################
#After removing multicollinearity:
```

```
#---------- VIFs of the remained variables --------
#  Variables              VIF
#BO22_parmax              2.332311
#BO22_parmean             3.842137
#BO22_nitratemean_ss      3.339599
#BO22_nitraterange_ss     3.577396
#BO22_phosphaterange_ss   5.055620
#BO22_phosphateltmin_ss   4.661173
#BO22_ppmean_ss           3.567557
#BO22_ppmin_ss            3.210874
#BO22_salinityrange_ss    3.259078
#MS_biogeo13_sst_mean_5m  6.833804
#MS_biogeo15_sst_max_5m   5.387835
#MS_bathy_5m              1.966168
#WC_bio5                  6.253519
#WC_bio8                  3.045540
#WC_bio9                  4.312384
#WC_bio18                 4.278053
#WC_alt                   2.908686
############################################################

#Download required variables again:
options(sdmpredictors_datadir = "C:/R-4.0.2/library/sdmpredictors/Meta/")

mar1 <- load_layers(layercodes = c('BO22_parmax','BO22_parmean',
                                   'BO22_nitratemean_ss',
                                   'BO22_nitraterange_ss',
                                   'BO22_phosphaterange_ss',
                                   'BO22_phosphateltmin_ss',
                                   'BO22_ppmean_ss',
                                   'BO22_ppmin_ss',
                                   'BO22_salinityrange_ss'),
                    equalarea=FALSE, rasterstack=TRUE)

mar2 <- load_layers(layercodes = c('MS_biogeo13_sst_mean_5m',
                                   'MS_biogeo15_sst_max_5m',
                                   'MS_bathy_5m'),
                    equalarea=FALSE, rasterstack=TRUE)

terr <- load_layers(layercodes = c('WC_bio5',
                                   'WC_bio8',
                                   'WC_bio9',
                                   'WC_bio18',
                                   'WC_alt'),
                    equalarea=FALSE, rasterstack=TRUE)

mar = raster::stack(mar1,mar2)
terr= raster::stack(terr)

#####################################################################
#Resample the mar by terr and vice versa and great a new raster stack
#####################################################################
t<-extent(-180, 180, -90, 90) #layer extent from terrestrial stack
m<-extent(-180, 180, -90, 90) #layer extent from marine stack
#no need to edit the following 6 lines
extent_list<-list(t, m)
extent_list<-lapply(extent_list, as.matrix)
matrix_extent<-matrix(unlist(extent_list), ncol=length(extent_list))
rownames(matrix_extent)<-c("xmin", "ymin", "xmax", "ymax")
best_extent<-extent(min(matrix_extent[1,]), max(matrix_extent[3,]), min(matrix_extent[2,]),
max(matrix_extent[4,]))
ranges<-apply(as.matrix(best_extent), 1, diff)
reso<-res(mar) #choose layer you want to keep resolution
nrow_ncol<-ranges/reso
s2<-raster(best_extent, nrows=nrow_ncol[2], ncols=nrow_ncol[1], crs=mar@crs) #choose layer crs
you want to keep

t.2 <-resample(terr, s2, method="ngb") #resample terr by s2
m.2 <-resample(mar, s2, method="ngb") #resample mar by s2
```

```
#Create mar/terr
env.MarTerr=stack(m.2, t.2) #stack resampled layers

#Create terr/mar
env.TerrMar=stack(t.2, m.2) #stack resampled layers

#Check ratser stacks
env.MarTerr
env.TerrMar

#################################################################
# Set study extent (# 10, 30,  53, 66 (Baltic Sea))
SA.ext <- extent(10, 30, 53, 66)

# Crop raster layer by extent & plot. Name the current raster stacks as 'env.c1' (for mar/terr)
and 'env.TerrMar' (for terr/mar)

env.MarTerr <- crop(env.MarTerr, SA.ext)
env.TerrMar <- crop(env.TerrMar, SA.ext)


#Check again the new ratser stacks
env.MarTerr
env.TerrMar

#Plot each raster stack
plot(env.MarTerr)
plot(env.TerrMar)

#-------------------------------------------------------------------------

########################
## LET'S START MODELING
######################

#Now we are ready for modeling. First we prepare the data object with 'sdmData' function
d <- sdmData(species~., train=newssp, predictors= env.MarTerr)
d
#Here, we will get presence-only

#We need to add pseudo-absence data at background. I will add 10000 pseudo-points as background
using "bg" argument
d <- sdmData(species~., train=newssp, predictors= env.MarTerr, bg=list(n=10000))
d

#Now we have everything !!
#Let's fit the model by adding different algorithm, replication techniques and K-fold cross-
validation
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.
#m <- sdm(species ~ . , d, filename='mC.img',overwrite=TRUE,
methods=c("GLM","RF","MAXENT","MARS","CART","GBM"), replication=c("boot"),k=10,test.percent=30)

#Alternative code:without (filename='')
m <- sdm(species ~ . , d, methods=c("GLM","RF","MAXENT","MARS","CART","GBM"),
replication=c("boot"),k=10,test.percent=30)

m

#Explore all model values by running "gui" function

gui(m)

########################
## MODEL EVALUATION
######################

#Generating response curve with all IDs
```

```
rcurve(m)

#----------------------------------------

#Get relative variable importance for all the models (Overall)

sdm::getVarImp(m)
plot(sdm::getVarImp(m))

#now we can evaluate the models using getEvaluation function
getEvaluation(m, stat='AUC')

#OR
getEvaluation(m, stat=c('AUC','TESS','Threshold'), opt=2)

#which threshold we need to evaluate and compare Presence and Absence (pa)?

ev<- getEvaluation(m, stat=c('AUC','TESS','Threshold'), opt=2)

mean(ev$threshold)

########################
## ENSEMBLE PREDICTION
######################

#-------------------------------------------------------------------------------------------#
#Ensemble modeling is an approach combines several individual models to produce more accurate
predictions than a single model alone.
#-------------------------------------------------------------------------------------------#
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.
#en<- ensemble(m, env.MarTerr,
filename='enC.img',overwrite=TRUE,setting=list(method='weighted',stat='AUC'))

en<- ensemble(m, env.MarTerr,setting=list(method='weighted',stat='AUC'))

en

plot(en)

plot(en,main="Current ensemble for Cyanobacteria\nwith GBM, RF, GLM, MAXENT, CTA and MARS")

################################
#Alternative Ensemble (ESDM) modeling
################################
#----------------------------------------
Occ<- read.csv("~/R/R-Directory/combinedDf.csv")

ESDMcurrent <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                  env.MarTerr, rep = 1, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                  k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)

plot(ESDMcurrent@projection,main = 'current ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

#Get model evulation
knitr::kable(ESDMcurrent@evaluation)
#-----------------------------------------------

################################
#Now we will do the Stacked species distribution models (SSDMs)
################################

Occ2<- read.csv("~/R/R-Directory/combinedDf.csv")
```

```
SSDM <- stack_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), Occ2, env.MarTerr, rep = 10,
ensemble.thresh = c(0.7),
                        Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
                        k=10, nb=10000, strat='random',Spcol = 'species', method = "pSSDM",
verbose = FALSE)

plot(SSDM@diversity.map,main = 'Current SSDM for Cyanobacteria with\nMAXENT, MARS, GLM, RF,\nGBM
, and CTA algorithms')

knitr::kable(SSDM@evaluation)

knitr::kable(SSDM@variable.importance)

plot(SSDM)

##############################################################################################

SDMcurrent_MAXENT<- modelling(c('MAXENT'),Occ,
                              env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_MAXENT@projection, main = 'Current SDM\nfor Cyanobacteria\nwith MAXENT
algorithm')

knitr::kable(SDMcurrent_MAXENT@evaluation)

#-------------------------------------------------

SDMcurrent_MARS<- modelling(c('MARS'),Occ,
                            env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_MARS@projection, main = 'Current SDM\nfor Cyanobacteria\nwith MARS algorithm')

knitr::kable(SDMcurrent_MARS@evaluation)
#-------------------------------------------------
SDMcurrent_GLM<- modelling(c('GLM'), Occ,
                           env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_GLM@projection, main = 'Current SDM\nfor Cyanobacteria\nwith GLM algorithm')

knitr::kable(SDMcurrent_GLM@evaluation)
#-------------------------------------------------

SDMcurrent_RF<- modelling(c('RF'), Occ,
                          env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_RF@projection, main = 'Current SDM\nfor Cyanobacteria\nwith RF algorithm')

knitr::kable(SDMcurrent_RF@evaluation)
#-------------------------------------------------

SDMcurrent_GBM<- modelling(c('GBM'), Occ,
                           env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_GBM@projection, main = 'Current SDM\nfor Cyanobacteria\nwith GBM algorithm')

knitr::kable(SDMcurrent_GBM@evaluation)
#-------------------------------------------------

SDMcurrent_CTA<- modelling(c('CTA'), Occ,
                           env.MarTerr, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_CTA@projection, main = 'Current SDM\nfor Cyanobacteria\nwith CTA algorithm')
```

```
knitr::kable(SDMcurrent_CTA@evaluation)

#-------------------------------------------------------------------------------
#Now we do the second model with "terr/mar" using the raster stack 'env.TerrMar'
#-------------------------------------------------------------------------------

########################
## LET'S START MODELING
########################

#Now we are ready for modeling. First we prepare the data object with sdmData function
d <- sdmData(species~., train=newssp, predictors= env.TerrMar)
d
#Here, we will get presence-only

#We need to add pseudo-absence data at background. I will add 10000 pseudo-points as background
using "bg" argument
d <- sdmData(species~., train=newssp, predictors= env.TerrMar, bg=list(n=10000))
d

#Now we have everything !!
#Let's fit the model by adding different algorithm, replication techniques and K-fold cross-
validation
m <- sdm(species ~ . , d, methods=c("GLM","RF","MAXENT","MARS","CART","GBM"),
replication=c("boot"),k=10,test.percent=30)

m

#Explore all model values by running "gui" function

gui(m)

########################
## MODEL EVALUATION
########################

#Generating response curve with all IDs
rcurve(m)

#-----------------------------------------

#Get relative variable importance for all the models (Overall)

sdm::getVarImp(m)
plot(sdm::getVarImp(m))

#now we can evaluate the models using getEvaluation function
getEvaluation(m, stat='AUC')

#OR
getEvaluation(m, stat=c('AUC','TESS','Threshold'), opt=2)

#which threshold we need to evaluate and compare Presence and Absence (pa)?

ev<- getEvaluation(m, stat=c('AUC','TESS','Threshold'), opt=2)

mean(ev$threshold)

########################
## ENSEMBLE PREDICTION
########################

#-------------------------------------------------------------------------------------------#
#Ensemble modeling is an approach combines several individual models to produce more accurate
predictions than a single model alone.
#-------------------------------------------------------------------------------------------#

en<- ensemble(m, env.TerrMar, setting=list(method='weighted',stat='AUC'))
```

```
en

plot(en)

plot(en,main="Current ensemble for Cyanobacteria\nwith GBM, RF, GLM, MAXENT, CTA and MARS")

################################
#Alternative Ensemble (ESDM) modeling
################################
#--------------------------------------
Occ<- read.csv("~/R/R-Directory/cyanoTAXA2.csv")

ESDMcurrent <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                  env.TerrMar, rep = 10, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                  k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)

plot(ESDMcurrent@projection,main = 'current ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

#Get model evaluation
knitr::kable(ESDMcurrent@evaluation)
#-----------------------------------------------

################################
##Now we will do the Stacked species distribution modeling (SSDM)
################################

Occ2<- read.csv("~/R/R-Directory/cyanoTAXA.csv")


SSDM <- stack_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), Occ2, env.TerrMar, rep = 10,
ensemble.thresh = c(0.7),
                        Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
                        k=10, nb=10000, strat='random',Spcol = 'species', method = "pSSDM",
verbose = FALSE)

plot(SSDM@diversity.map,main = 'Current SSDM for Cyanobacteria with\nMAXENT, MARS, GLM, RF,\nGBM
, and CTA algorithms')

knitr::kable(SSDM@evaluation)

knitr::kable(SSDM@variable.importance)

plot(SSDM)

#############################################################################################

######################
## Now we can test performance and get the evaluation of each
algorithm:'MAXENT','MARS','GLM','RF','GBM','CTA'
# VERY IMPORTANT: remember to run this test for each raster stack: mar/terr and terr/mar
######################

Occ<- read.csv("~/R/R-Directory/cyanoTAXA.csv")


SDMcurrent_MAXENT<- modelling(c('MAXENT'),Occ,
                              env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_MAXENT@projection, main = 'Current SDM\nfor Cyanobacteria\nwith MAXENT
algorithm')

knitr::kable(SDMcurrent_MAXENT@evaluation)

#-----------------------------------------------
```

```
SDMcurrent_MARS<- modelling(c('MARS'),Occ,
                            env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_MARS@projection, main = 'Current SDM\nfor Cyanobacteria\nwith MARS algorithm')

knitr::kable(SDMcurrent_MARS@evaluation)
#-------------------------------------------------
SDMcurrent_GLM<- modelling(c('GLM'), Occ,
                           env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_GLM@projection, main = 'Current SDM\nfor Cyanobacteria\nwith GLM algorithm')

knitr::kable(SDMcurrent_GLM@evaluation)
#-------------------------------------------------

SDMcurrent_RF<- modelling(c('RF'), Occ,
                          env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_RF@projection, main = 'Current SDM\nfor Cyanobacteria\nwith RF algorithm')

knitr::kable(SDMcurrent_RF@evaluation)
#-------------------------------------------------

SDMcurrent_GBM<- modelling(c('GBM'), Occ,
                           env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_GBM@projection, main = 'Current SDM\nfor Cyanobacteria\nwith GBM algorithm')

knitr::kable(SDMcurrent_GBM@evaluation)
#-------------------------------------------------

SDMcurrent_CTA<- modelling(c('CTA'), Occ,
                           env.TerrMar, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
verbose = FALSE)

plot(SDMcurrent_CTA@projection, main = 'Current SDM\nfor Cyanobacteria\nwith CTA algorithm')

knitr::kable(SDMcurrent_CTA@evaluation)


##########################################################################################################

################################
#FUTURE PREDICTION
################################

#-------------------------------------------------------
#Download the Future MAR raster layers for years 2050 and 2100 using at RCP85
#-------------------------------------------------------

# Download rasters to sdmpredictors/Meta folder and import into R
options(sdmpredictors_datadir = "C:/R-4.0.2/library/sdmpredictors/Meta/")

future1 <- load_layers( layercodes = c("BO22_RCP85_2050_templtmax_ss",
"BO22_RCP85_2050_templtmin_ss",
                                       "BO22_RCP85_2050_tempmax_ss",
"BO22_RCP85_2050_tempmin_ss",
                                       "BO22_RCP85_2050_tempmean_ss",
"BO22_RCP85_2050_temprange_ss"),
                        equalarea=FALSE, rasterstack=TRUE)

future2 <- load_layers( layercodes = c("BO22_RCP85_2100_templtmax_ss",
"BO22_RCP85_2100_templtmin_ss",
                                       "BO22_RCP85_2100_tempmax_ss",
```

```
"BO22_RCP85_2100_tempmin_ss",
                                        "BO22_RCP85_2100_tempmean_ss",
"BO22_RCP85_2100_temprange_ss"),
                        equalarea=FALSE, rasterstack=TRUE)

#Stack mar layers
marf = stack(future1, future2)

#-----------------------------------------------------------
#Download the Future TERR raster layers for years 2050 and 2070 using at RCP85
#-----------------------------------------------------------

# Download rasters to sdmpredictors/Meta folder and import into R
options(sdmpredictors_datadir = "C:/R-4.0.2/library/sdmpredictors/Meta/")

future3<- raster::getData('CMIP5', var='bio', res=2.5, rcp=85, year=50,model='AC')

#Drop layers Bio02, Bio03, Bio04, Bio15
future3<-dropLayer(future3, c("ac85bi502", "ac85bi503", "ac85bi504","ac85bi5015"))

future4<- raster::getData('CMIP5', var='bio', res=2.5, rcp=85, year=70,model='AC')

#Drop layers Bio02, Bio03, Bio04, Bio15
future4<-dropLayer(future4, c("ac85bi702", "ac85bi703", "ac85bi704","ac85bi7015"))

#Stack terr layers
terrf = stack(future3, future4)


#-----------------------------------------------------------

# Set study extent (# 10, 30,  53, 66 (Baltic Sea))
SA.ext <- extent(10, 30, 53, 66)

marf <- crop(marf, SA.ext)
terrf <- crop(terrf, SA.ext)

plot(marf)
plot(terrf)

#################################################################
#Now we will start the modeling, first by removing the multicollinearity between future
predictors.
#We will run two different models using the two raster stacks: future marine (marf) and future
terrestrial (terrf)
#################################################################


########################
## Removing multicollinearity between future predictors #VERY IMPORTANT#
########################

#Variance Inflation Factor and test for multicollinearity using VIF function of usdm package
("vifstep" and "vifcor")-
# (th) stand for correlation threshold.
#Here, any variable with high correlation coefficient (grater than or equal to threshold 0.7)
will be removed.

v1f<-vifstep(marf)
v1f
v2f<-vifcor(marf,th=0.7)
v2f
marf<-exclude(marf,v2f)

marf

#Have a look on the new predictors

plot(marf)
```

```
############################################################
#After removing multicollinearity:
#---------- VIFs of the remained variables --------
#Variables                    VIF
#BO22_RCP85_2050_tempmin_ss 6.161892
#BO22_RCP85_2050_tempmean_ss 6.286015
#BO22_RCP85_2100_templtmax_ss 5.076330
############################################################

#######################
## Removing multicollinearity between terrestrial predictors #VERY IMPORTANT#
#######################

v1f<-vifstep(terrf)
v1f
v2f<-vifcor(terrf,th=0.7)
v2f
terrf<-exclude(terrf,v2f)

terrf

#Have a look on the new predictors

plot(terrf)

############################################################
#After removing multicollinearity:
#---------- VIFs of the remained variables --------
#Variables      VIF
#ac85bi508 1.349662
#ac85bi509 1.591481
#ac85bi707 1.460179
#ac85bi708 1.263614
#ac85bi709 2.036646
#ac85bi7014 1.788487
############################################################
#-------------------------------------------------------------------------------
#To make sure:
# Set study extent (# 10, 30,  53, 66 (Baltic Sea))
SA.ext <- extent(10, 30, 53, 66)

marf <- crop(marf, SA.ext)
terrf <- crop(terrf, SA.ext)

plot(marf)
plot(terrf)
#######################
## LET'S START FUTURE MARINE (marf) MODELING
#######################

#Now we are ready for modeling. First we prepare the data object with sdmData function
df <- sdmData(species~., train=newssp, predictors= marf)
df
#Here, we will get presence-only

#We need to add pseudo-absence data at background. I will add 10000 pseudo-points as background
using "bg" argument
df <- sdmData(species~., train=newssp, predictors= marf, bg=list(n=10000))
df

#Now we have everything !!
#Let's fit the model by adding different algorithm, replication techniques and K-fold cross-
validation
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.
mf <- sdm(species ~ . , df,
filename='mf.img',overwrite=TRUE,methods=c("GLM","RF","MAXENT","MARS","CART","GBM"),
replication=c("boot"),k=10,test.percent=30)
```

```
mf

#Explore all model values by running "gui" function

gui(mf)

#######################
## MODEL EVALUATION
#######################

#Generating response curve with all IDs
rcurve(mf)

#----------------------------------------

#Get relative variable importance for all the models (Overall)

sdm::getVarImp(mf)
plot(sdm::getVarImp(mf))

#now we can evaluate the models using getEvaluation function
getEvaluation(mf, stat='AUC')

#OR
getEvaluation(mf, stat=c('AUC','TESS','Threshold'), opt=2)

#which threshold we need to evaluate and compare Presence and Absence (pa)?

evf<- getEvaluation(mf, stat=c('AUC','TESS','Threshold'), opt=2)

mean(evf$threshold)

#######################
## ENSEMBLE PREDICTION
#######################

#-------------------------------------------------------------------------------------------#
#Future Ensemble models is a ML approach combines multiple models in one prediction process.
#-------------------------------------------------------------------------------------------#
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.

#enf<- ensemble(mf, marf,
filename='enF.img',overwrite=TRUE,setting=list(method='weighted',stat='AUC'))

enf<- ensemble(mf, marf, setting=list(method='weighted',stat='AUC'))

enf

plot(enf)

plot(enf,main="Future ensemble for Cyanobacteria\nwith GBM, RF, GLM, MAXENT, CTA and MARS")

#################################
#Alterantive Future Ensemble (ESDM) modeling
#################################
#----------------------------------------


Occ<- read.csv("~/R/R-Directory/CyanoTAXA2.csv")

ESDMfuture <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                 marf, rep = 10, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                 k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)
```

```
plot(ESDMfuture@projection,main = 'Future ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

#Get model evaluation
knitr::kable(ESDMfuture@evaluation)
#-----------------------------------------------

################################
##Now we will do the Stacked species distribution modeling (SSDM)
################################

Occ2<- read.csv("~/R/R-Directory/cyanoTAXA.csv")


SSDMf <- stack_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), Occ2, marf, rep = 1,
ensemble.thresh = c(0.7),
                         Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
                         k=10, nb=100, strat='random',Spcol = 'species', method = "pSSDM",
verbose = FALSE)

plot(SSDMf@diversity.map,main = 'Future SSDM for Cyanobacteria with\nMAXENT, MARS, GLM, RF,\nGBM
, and CTA algorithms')

knitr::kable(SSDMf@evaluation)

knitr::kable(SSDMf@variable.importance)

plot(SSDMf)

#########################################################################################################

########################
## Now we can test performance and get the evaluation of each
algorithm:'MAXENT','MARS','GLM','RF','GBM','CTA'
# VERY IMPORTANT: remember to run this test for each raster stack: FUTUREmar/FUTUREterr and
FUTUREterr/FUTUREmar
######################

########################
## PREDICTION - future 'MAXENT','MARS','GLM','RF','GBM','CTA'
########################

Occ<- read.csv("~/R/R-Directory/CyanoTAXA2.csv")

SDMfuture_MAXENT<- modelling('MAXENT', subset(Occ, Occ$species == 'Cyanobacteria'),
                             marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose
= FALSE)

plot(SDMfuture_MAXENT@projection, main = 'Future SDM\nfor Cyanobacteria\nwith MAXENT algorithm')

knitr::kable(SDMfuture_MAXENT@evaluation)

#-----------------------------------------------

SDMfuture_MARS<- modelling('MARS', subset(Occ, Occ$species == 'Cyanobacteria'),
                           marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_MARS@projection, main = 'Future SDM\nfor Cyanobacteria\nwith MARS algorithm')

knitr::kable(SDMfuture_MARS@evaluation)
#-----------------------------------------------
SDMfuture_GLM<- modelling('GLM', subset(Occ, Occ$species == 'Cyanobacteria'),
                          marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_GLM@projection, main = 'Future SDM\nfor Cyanobacteria\nwith GLM algorithm')

knitr::kable(SDMfuture_GLM@evaluation)
```

```
#-------------------------------------------------

SDMfuture_RF<- modelling('RF', subset(Occ, Occ$species == 'Cyanobacteria'),
                         marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_RF@projection, main = 'Future SDM\nfor Cyanobacteria\nwith RF algorithm')

knitr::kable(SDMfuture_RF@evaluation)
#-------------------------------------------------

SDMfuture_GBM<- modelling('GBM', subset(Occ, Occ$species == 'Cyanobacteria'),
                          marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_GBM@projection, main = 'Future SDM\nfor Cyanobacteria\nwith GBM algorithm')

knitr::kable(SDMfuture_GBM@evaluation)
#-------------------------------------------------

SDMfuture_CTA<- modelling('CTA', subset(Occ, Occ$species == 'Cyanobacteria'),
                          marf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_CTA@projection, main = 'Future SDM\nfor Cyanobacteria\nwith CTA algorithm')

knitr::kable(SDMfuture_CTA@evaluation)
#-------------------------------------------------

ESDMfuture <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                 marf, rep = 10, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                 k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)

plot(ESDMfuture@projection,main = 'Future ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

knitr::kable(ESDMfuture@evaluation)

#############################################################################
######################
## LET'S START FUTURE MARINE (terrf) MODELING
######################

#Now we are ready for modeling. First we prepare the data object with sdmData function
df <- sdmData(species~., train=newssp, predictors= terrf)
df
#Here, we will get presence-only

#We need to add pseudo-absence data at background. I will add 10000 pseudo-points as background
using "bg" argument
df <- sdmData(species~., train=newssp, predictors= terrf, bg=list(n=10000))
df

#Now we have everything !!
#Let's fit the model by adding different algorithm, replication techniques and K-fold cross-
validation
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.
mf <- sdm(species ~ . , df,
filename='mf.img',overwrite=TRUE,methods=c("GLM","RF","MAXENT","MARS","CART","GBM"),
replication=c("boot"),k=10,test.percent=30)

mf

#Explore all model values by running "gui" function
```

```
gui(mf)

#######################
## MODEL EVALUATION
#######################

#Generating response curve with all IDs
rcurve(mf)

#----------------------------------------

#Get relative variable importance for all the models (Overall)

sdm::getVarImp(mf)
plot(sdm::getVarImp(mf))

#now we can evaluate the models using getEvaluation function
getEvaluation(mf, stat='AUC')

#OR
getEvaluation(mf, stat=c('AUC','TESS','Threshold'), opt=2)

#which threshold we need to evaluate and compare Presence and Absence (pa)?

evf<- getEvaluation(mf, stat=c('AUC','TESS','Threshold'), opt=2)

mean(evf$threshold)

#######################
## ENSEMBLE PREDICTION
#######################

#-------------------------------------------------------------------------------------------#
#Future Ensemble models is a ML approach combines multiple models in one prediction process.
#-------------------------------------------------------------------------------------------#
#Note: printing image file (filename='mC.img',overwrite=TRUE) function is extensive command and
uses a lot of time and memory. Use only if necessary or simply remove it.

#enf<- ensemble(mf, terrf,
filename='enF.img',overwrite=TRUE,setting=list(method='weighted',stat='AUC'))

enf<- ensemble(mf, terrf, setting=list(method='weighted',stat='AUC'))

enf

plot(enf)

plot(enf,main="Future ensemble for Cyanobacteria\nwith GBM, RF, GLM, MAXENT, CTA and MARS")

#################################
#Alterantive Future Ensemble (ESDM) modeling
#################################
#----------------------------------------


Occ<- read.csv("~/R/R-Directory/CyanoTAXA2.csv")

ESDMfuture <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                 terrf, rep = 10, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                 k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)

plot(ESDMfuture@projection,main = 'Future ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

#Get model evulation
knitr::kable(ESDMfuture@evaluation)
```

```
#-------------------------------------------------

################################
##Now we will do the Stacked species distribution modeling (SSDM)
################################

Occ2<- read.csv("~/R/R-Directory/cyanoTAXA.csv")



SSDMf <- stack_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), Occ2, terrf, rep = 1,
ensemble.thresh = c(0.7),
                         Xcol = 'decimalLongitude', Ycol = 'decimalLatitude',
                         k=10, nb=100, strat='random',Spcol = 'species', method = "pSSDM",
verbose = FALSE)

plot(SSDMf@diversity.map,main = 'Future SSDM for Cyanobacteria with\nMAXENT, MARS, GLM, RF,\nGBM
, and CTA algorithms')

knitr::kable(SSDMf@evaluation)

knitr::kable(SSDMf@variable.importance)

plot(SSDMf)

#########################################################################################################

########################
## Now we can test performance and get the evaluation of each
algorithm:'MAXENT','MARS','GLM','RF','GBM','CTA'
# VERY IMPORTANT: remember to run this test for each raster stack: FUTUREmar/FUTUREterr and
FUTUREterr/FUTUREmar
######################

########################
## PREDICTION - future 'MAXENT','MARS','GLM','RF','GBM','CTA'
########################

Occ<- read.csv("~/R/R-Directory/CyanoTAXA2.csv")

SDMfuture_MAXENT<- modelling('MAXENT', subset(Occ, Occ$species == 'Cyanobacteria'),
                             terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose
= FALSE)

plot(SDMfuture_MAXENT@projection, main = 'Future SDM\nfor Cyanobacteria\nwith MAXENT algorithm')

knitr::kable(SDMfuture_MAXENT@evaluation)

#-------------------------------------------------

SDMfuture_MARS<- modelling('MARS', subset(Occ, Occ$species == 'Cyanobacteria'),
                           terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_MARS@projection, main = 'Future SDM\nfor Cyanobacteria\nwith MARS algorithm')

knitr::kable(SDMfuture_MARS@evaluation)
#-------------------------------------------------
SDMfuture_GLM<- modelling('GLM', subset(Occ, Occ$species == 'Cyanobacteria'),
                          terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_GLM@projection, main = 'Future SDM\nfor Cyanobacteria\nwith GLM algorithm')

knitr::kable(SDMfuture_GLM@evaluation)
#-------------------------------------------------

SDMfuture_RF<- modelling('RF', subset(Occ, Occ$species == 'Cyanobacteria'),
                         terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
```

```
FALSE)

plot(SDMfuture_RF@projection, main = 'Future SDM\nfor Cyanobacteria\nwith RF algorithm')

knitr::kable(SDMfuture_RF@evaluation)
#-------------------------------------------------

SDMfuture_GBM<- modelling('GBM', subset(Occ, Occ$species == 'Cyanobacteria'),
                          terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_GBM@projection, main = 'Future SDM\nfor Cyanobacteria\nwith GBM algorithm')

knitr::kable(SDMfuture_GBM@evaluation)
#-------------------------------------------------

SDMfuture_CTA<- modelling('CTA', subset(Occ, Occ$species == 'Cyanobacteria'),
                          terrf, Xcol = 'decimalLongitude', Ycol = 'decimalLatitude', verbose =
FALSE)

plot(SDMfuture_CTA@projection, main = 'Future SDM\nfor Cyanobacteria\nwith CTA algorithm')

knitr::kable(SDMfuture_CTA@evaluation)
#-------------------------------------------------

ESDMfuture <- ensemble_modelling(c('MAXENT','MARS','GLM','RF','GBM','CTA'), subset(Occ,
Occ$species == 'Cyanobacteria'),
                                 terrf, rep = 10, Xcol = 'decimalLongitude', Ycol =
'decimalLatitude',
                                 k=10, nb=10000, strat= 'random', ensemble.thresh = c(0.7),
verbose = FALSE)

plot(ESDMfuture@projection,main = 'Future ESDM for cyanobacteria with\nMAXENT, MARS, GLM,
RF,\nGBM , and CTA algorithms')

knitr::kable(ESDMfuture@evaluation)
```