Monitoring suitable habitat of an invasive species into cropland areas in Uganda

Machine Learning A

David Wuepper, Hadi, Wyclife Agumba Oluoch

2025-03-02

Contents

0.1 Background

Invasive species constitute an important factor for consideration in agriculture as they compromise crop performance and have the potential for altering the native ecosystem and species composition of an area. It is estimated that the cost of damage caused by invasive species is 13

times higher than the cost of management (Diagne et al. 2021). The same authors estimated the global cost of controlling invasive species as $1.3 trillion (2017 dollar rate). The control of invasive species also calls for use of herbicides in the cropland that are cost to crop production and could pose residual effects on the final products. By monitoring the current distribution of invasive species and predicting their potential future distribution, decision-makers can optimize strategies that can circumvent the spread of the species and improve agricultural productivity, all while ensuring sustainable land management practices.

In this post, we explore how to create species distribution maps of invasive weed species, *Parthenium hysterophorus* by integrating its occurrence records, bioclimate variables and **R** (R Core Team 2023) for further processing and visualization. Using occurrence data from the Global Biodiversity Information Facility (**GBIF**)(GBIF.org 2020) and climate data from **WorldClim** (Fick and Hijmans 2017), we focus on Uganda as a case study and demonstrate the workflow from data retrieval to final map creation. We chose Uganda since it is one of the countries which has substantial reported records of the invasive species. This approach can be applied to any other species or country or conditions of relevance such as mapping of potential areas of profitable autonomous machine use in agriculture.

0.2 Occurrence records: Global Biodiversity Information Facility (GBIF)

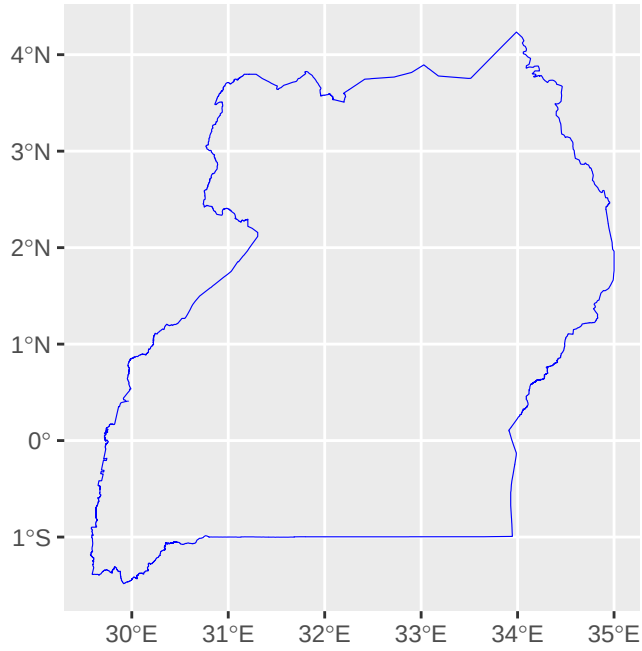
GBIF is a public database which contains a total of 3079892458 occurrence records of species throughout the world as of **2025-03-02**. It is a great database for anyone interested in obtaining occurrence records of a species for modeling the potential suitable habitats and other related analyses. Other databases one can look into include **eBird**, **RAINBIO**, **BIEN**, various herbaria, published reports, and ones own field surveys.

0.2.1 Loading Boundary Data in R

We begin by loading the boundary data for Uganda using the `geodata` package (Hijmans et al. 2024) version 0.6.2, which offers a convenient way to obtain a lot of geospatial data from online databases. There are several other options or packages one can use to get boundary data for their area of interest. We also include `terra` package (Hijmans 2024) version 1.8.29 to aid handling spatial data and `tidyterra` package (Hernangómez 2023) version 0.7.0 for plotting in conjunction with assorted `tidyverse` package 2.0.0 functionalities in the background.

```
library(geodata)
library(terra)
library(tidyterra)
library(tidyverse)

uganda <- gadm(country = "Uganda", # Can change to any country
               level = 0, # For national boundary
```

```
                path = tempdir()) # Temporary storage

ggplot() +
  geom_spatvector(data = uganda,
                  fill = NA,
                  col = "blue")
```



The above code snippet loads Uganda's administrative boundaries, with `level = 0` representing the national boundary and other levels like 1, 2 or 3 depict lower regional divisions. We then visualize/plot it.

0.2.2 Obtaining occurrence records for the species from GBIF

In this next step we again use `geodata` package to obtain species occurrence records for the invasive species of interest to us.

```
occ <- sp_occurrence(
  genus = "Parthenium", # Genus name
  species = "hysterophorus", # Species name
  ext = uganda, # Our country of interest
  removeZeros = TRUE, # To remove points with 0 coordinates
  geo = TRUE, # To obtain only records with geo info.
```

```
  download = TRUE, # To download
  ntries = 5 # To try getting the data at least 5 times
)
```

```
224 records found
```

```
0-224
224 records downloaded
```

0.2.3 Cleaning the occurrence records

Next, we use `dplyr` package to clean the data. In cleaning the data, we are interested in:

- Retaining only the **longitute** and **latitude** columns. In the dataset, these are called *lon* and *lat*, respectively.
- Drop NAs
- Removing duplicated locations.
- Mutate a column for species with values of 1.

```
library(dplyr)
occ_clean <- occ |>
  dplyr::select(lon, lat) |>
  tidyr::drop_na() |> # Play around with drop_na before select
  distinct() |>
  mutate(species = 1)
head(occ_clean)
```

```
       lon       lat species
1 34.74469 -0.117262       1
2 34.31988 -0.520142       1
3 32.65090  0.347100       1
4 32.45997  0.063426       1
5 30.03423 -0.127958       1
6 34.03300 -0.491460       1
```

In other instances, one may be interested in checking also the year column so that only records observed within specific time periods are used for modeling. There are 94 columns in the dataset and one may be interested in filtering for various reasons. For our demonstration here, we are satisfied with lon, lat for distinct records with NAs dropped.

0.2.4 Make the occurrence records geospatial

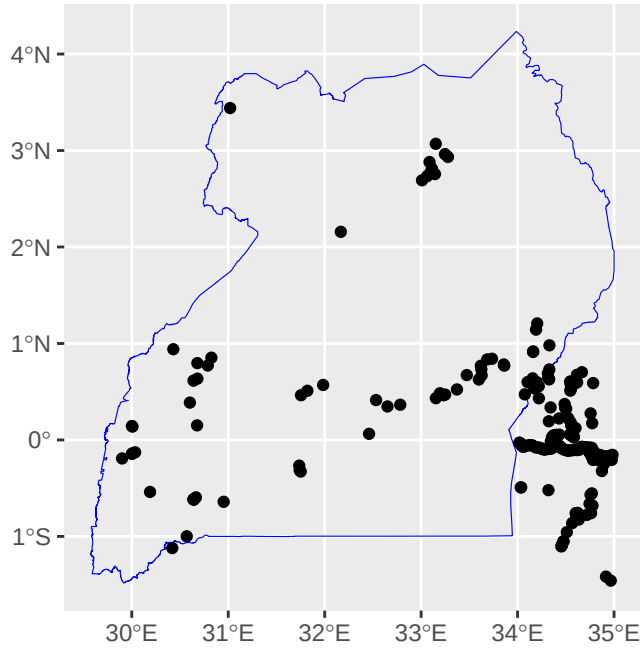
The next step is to make the occurrence records geospatial by using the *vect* function from `terra` package.

```
occ_vect <- occ_clean |>
  vect(geom = c("lon", "lat"), crs = "epsg:4326")
```

0.2.5 Visualize the records in Uganda

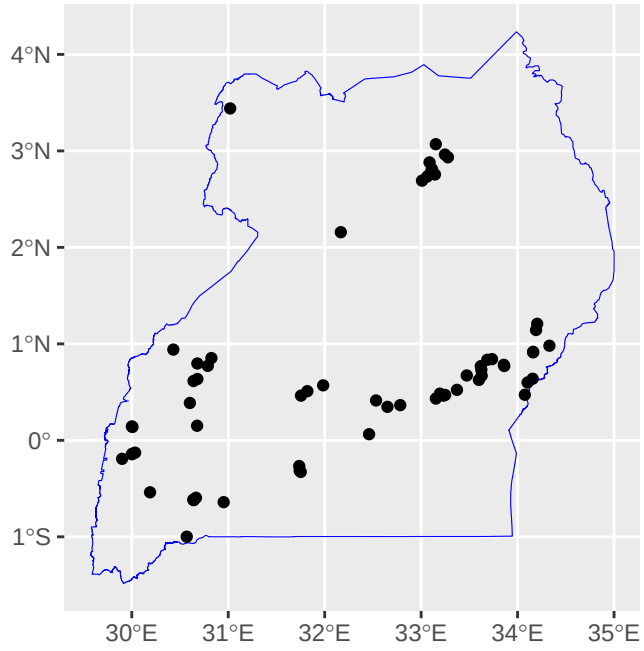
To ensure that the records are located within Uganda as expected, we can overlay the plot of the records on the map of Uganda.

```
ggplot() +
  geom_spatvector(data = uganda, fill = NA, col = "blue") +
  geom_spatvector(data = occ_vect)
```



NOTE: When we provided the extent as uganda, the *sp_occurrence* function used the bounding box of Uganda, not the actual border hence some of the points fell into western Kenya. Therefore, we need to run a spatial analysis which drops all the points outside the border of Uganda. This is *crop* function from `terra` package.

```r
occ_ug <- crop(occ_vect, uganda)

ggplot() +
  geom_spatvector(data = uganda, fill = NA, col = "blue") +
  geom_spatvector(data = occ_ug)
```



0.3 Predictor variables

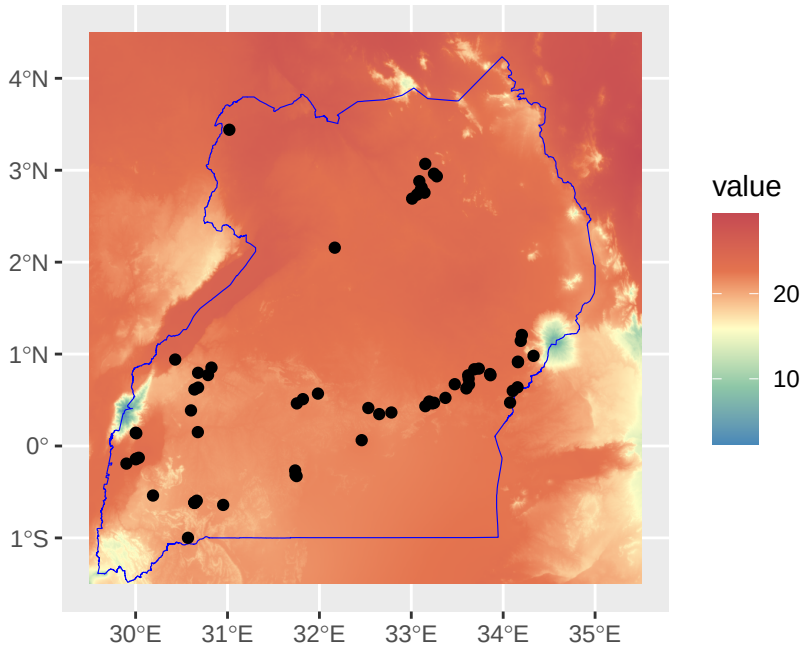
In this next step, we are going to obtain predictor variables from worldclim. We will use *worldclim_country* function from `geodata` package which is super convenient.

```r
predictors <- worldclim_country(country = "Uganda",
                                var = "bio",
                                path = tempdir(),
                                version = "2.1")
writeRaster(predictors, "data/current.tif")
```

0.3.1 Visualize one of the predictors

We then confirm that the predictor variable covers our country of interest and both country boundary and occurrence records overlay perfectly.

```
predictors <- rast("data/current.tif")
ggplot() +
  geom_spatraster(data = predictors[[1]]) +
  scale_fill_whitebox_c(
    palette = "muted"
  ) +
  geom_spatvector(data = uganda, fill = NA, col = "blue") +
  geom_spatvector(data = occ_ug)
```

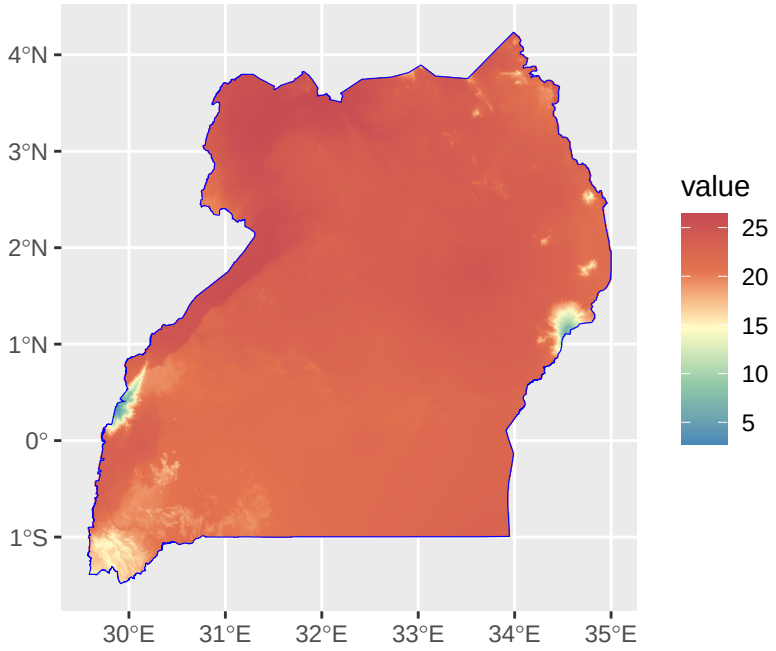


Just like with *sp_occurrence* which returned occurrence records based on the bounding box of Uganda instead of the country boundary, *worldclim_country* also returned the predictor variables based on the bounding box of Uganda. This means we need to run another code to clip the predictors to the boundaries of Uganda only.

```
preds_ug <- crop(predictors, uganda, mask = TRUE)
```

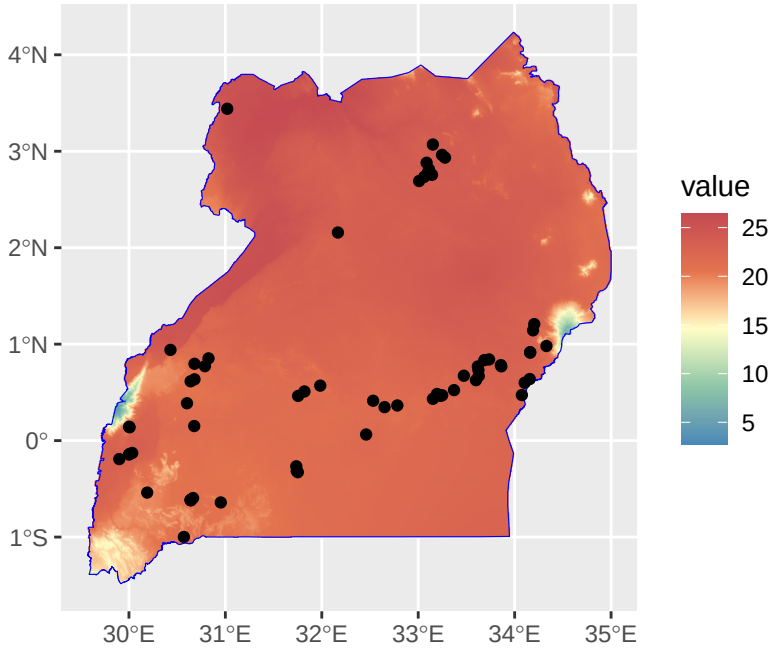
Confirm that the shape of the predictor variables is now coinciding with the country border.

```
ggplot() +
  geom_spatraster(data = preds_ug[[1]]) +
  scale_fill_whitebox_c(palette = "muted") +
  geom_spatvector(data = uganda, fill = NA, col = "blue")
```



Now we can also confirm that the occurrence records are overlaying the predictor variables perfectly. Sometimes one may get occurrence records, study area boundaries and predictor variables that are on different coordinate reference systems which may call for re-projection. However, in our current case, everything is in `epsg:4326`.

```
ggplot() +
  geom_spatraster(data = preds_ug[[1]]) +
  scale_fill_whitebox_c(palette = "muted") +
  geom_spatvector(data = uganda, fill = NA, col = "blue") +
  geom_spatvector(data = occ_ug)
```



0.3.2 Check and drop highly correlated variables (Multi-collinearity check)

It is always important to drop all variables that are highly correlated from the modeling workflow. This we can do for the whole of the study area or specific to the predictor variables at the occurrence points. Either way is acceptable. For that, we will use `usdm` (Naimi et al. 2014) package version 2.1.7.

```
library(usdm)
correlated <- vifcor(preds_ug)
```

Excluding the highly correlated variables in order to remain with those that are valuable in predicting the potential suitable habitat for the species.

```
preds_ug_used <- exclude(preds_ug, correlated)
```

Let us rename the predictor layers to have concise names:

```
names(preds_ug_used)
```

```
[1] "wc2.1_30s_bio_2"  "wc2.1_30s_bio_4"  "wc2.1_30s_bio_8"  "wc2.1_30s_bio_9"
[5] "wc2.1_30s_bio_13" "wc2.1_30s_bio_14" "wc2.1_30s_bio_15" "wc2.1_30s_bio_18"
[9] "wc2.1_30s_bio_19"
```

```
names(preds_ug_used) <- paste0("bio", c(2,4,8,9,13,14,15,18,19))
names(preds_ug_used)
```

```
[1] "bio2"  "bio4"  "bio8"  "bio9"  "bio13" "bio14" "bio15" "bio18" "bio19"
```

0.4 Build the sdmData object and model

In this next step, we will build the machine learning model which will be able to take the current occurrence points and predictor variables to calibrate a model that can predict where else the species could occur. For that, we will use `sdm` package (Naimi and Araujo 2016) version 1.2.56. There are several other packages that can achieve this, such as `ENMeval` (Kass et al. 2021), `wallace` (et al. 2023), `ENMTools` (Warren and Dinnage 2024), `biomod2` (Thuiller et al. 2024), `dismo` (Hijmans et al. 2023) among others.

```
library(sdm)
```

```
sdm 1.2-56 (2025-01-17)
```

```
sdm_data <- sdmData(formula = species ~.,
                    train = occ_ug,
                    predictors = preds_ug_used,
                    bg = list(method = "gRandom",
                              n = 1000))
```

Have a glance at the sdm_data object:

```
sdm_data
```

```
class                                 : sdmdata
============================================================
number of species                     :  1
species names                         :  species
number of features                    :  9
feature names                         :  bio2, bio4, bio8, ...
type                                  :  Presence-Background
has independent test data?             :  FALSE
number of records                     :  1065
has Coordinates?                      :  TRUE
```

Then the model calibration:

```
sdm_model <- sdm(formula = species ~.,
                 data = sdm_data,
                 methods = "maxent",
                 replications = "boot",
                 test.p = 30,
                 n = 3)
write.sdm(sdm_model, "model/sdm_model.sdm")
```

View the model summary:

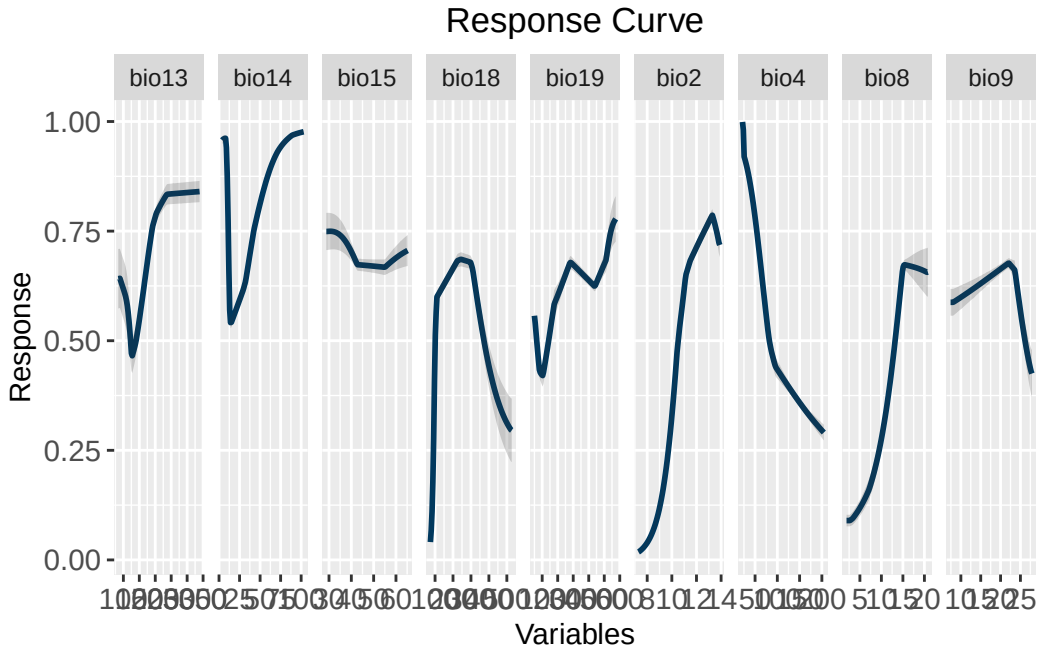
```
sdm_model <- read.sdm("model/sdm_model.sdm")
sdm_model
```

```
class                                  : sdmModels
=======================================================
number of species                      :  1
number of modelling methods            :  1
names of modelling methods             :  maxent
replicate.methods (data partitioning) :  bootstrap
number of replicates (each method)     :  3
toral number of replicates per model   :  3 (per species)
----------------------------------------
model run success percentage (per species)  :
----------------------------------------
method          species
----------------------
maxent     :        100   %

#########################################################################
model Mean performance (per species), using test dataset (generated using partitioning):
-------------------------------------------------------------------------------

 ## species   :  species
=========================

methods    :     AUC     |     COR     |     TSS     |     Deviance
-------------------------------------------------------------------------
maxent     :     0.81    |     0.31    |     0.52    |     1.04
```

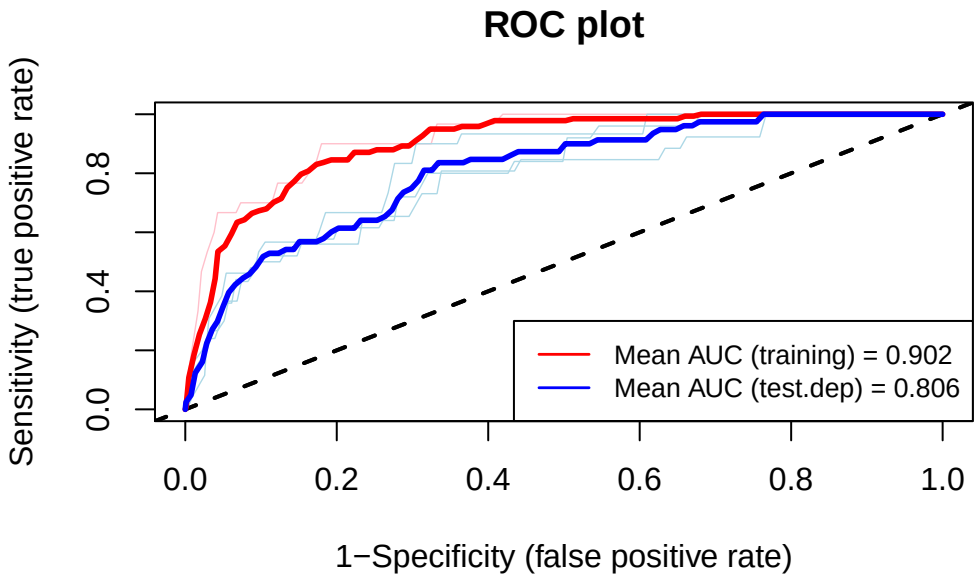
Other checks on the model:

```
rcurve(sdm_model)
```

```
The id argument is not specified; The modelIDs of 3 successfully fitted models are assigned t
```

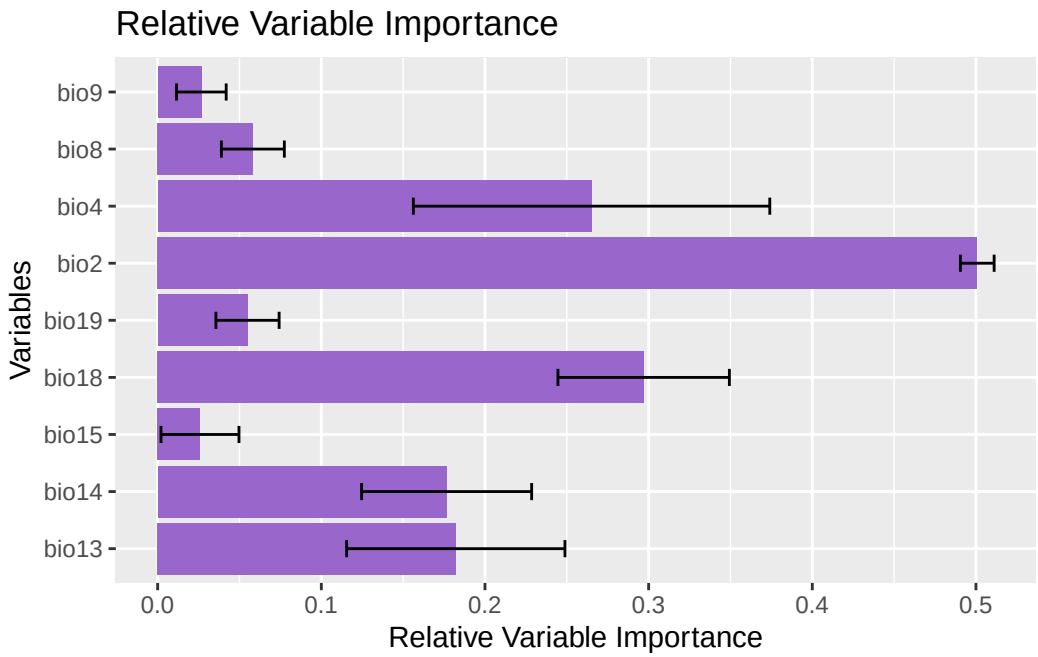



```
roc(sdm_model)
```




```
plot(getVarImp(sdm_model))
```

```
The variable importance for all the models are combined (averaged)...
```



0.5 NOTE

One can redo what we have done up to here several times, tuning the modelling process until a model with desirable characteristics is attained. Basically, this is a model that meets the specific needs of the project. For example, one can change the number of replications (20 etc), replication methods (sub, cv), model methods (brt, gam, glm, rf, svm). One can also add occurrence records (e.g. from BIEN), one can also add predictor variables (soil, elevation…but be careful with remote sensing data as they are object specific and less of environment).

0.6 Predict the potential suitable habitat for the species

Finally, we get to the point where we use the trained model to predict the potential suitable habitat of the species over the study area. For this, we use *predict* function.

```
pred <- predict(object = sdm_model, preds_ug_used)
pred
```

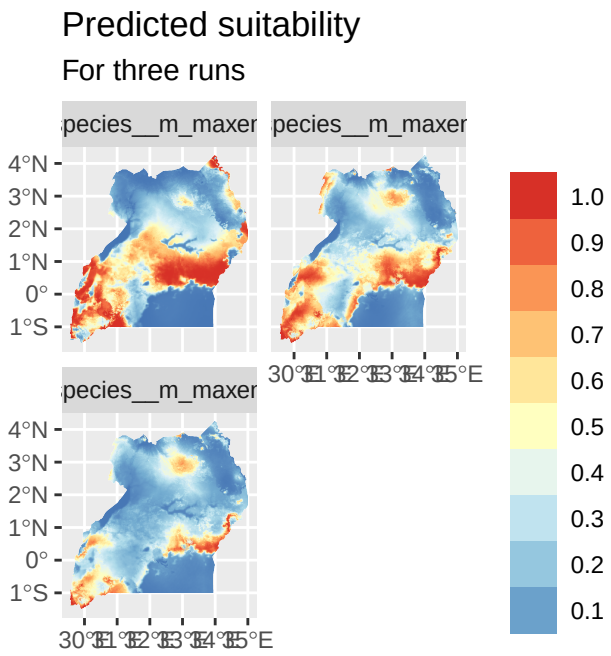
```
class       : SpatRaster
dimensions  : 686, 651, 3  (nrow, ncol, nlyr)
resolution  : 0.008333333, 0.008333333  (x, y)
extent      : 29.575, 35, -1.483333, 4.233333  (xmin, xmax, ymin, ymax)
```

```
coord. ref. : lon/lat WGS 84 (EPSG:4326)
source(s)   : memory
names       : id_1__sp_s~t__re_boot, id_2__sp_s~t__re_boot, id_3__sp_s~t__re_boot
min values  :           0.002192378,           0.005477519,           0.005691341
max values  :           1.000000000,           0.999999702,           0.999999762
```

This is giving us three layers of `spatRaster` corresponding to the three runs.

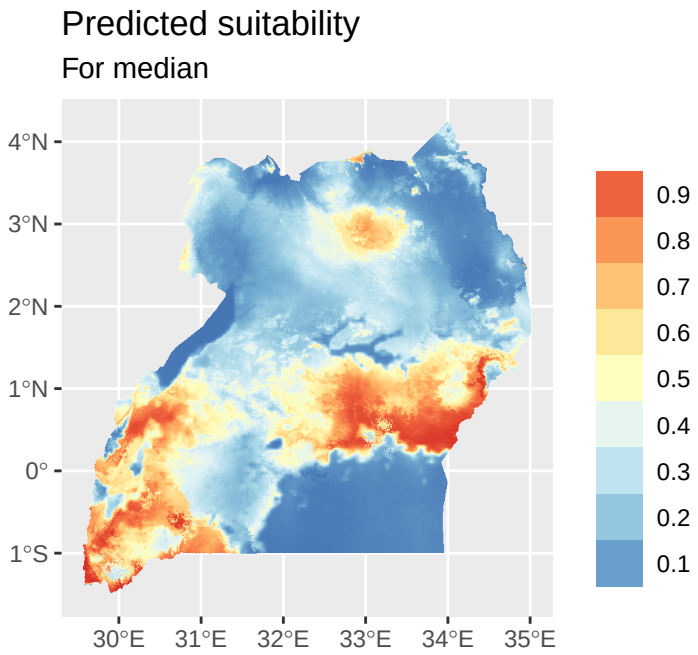
We can visualize them:

```
ggplot() +
  geom_spatraster(data = pred) +
  facet_wrap(~lyr, ncol = 2) +
  scale_fill_whitebox_c(
    palette = "bl_yl_rd",
    n.breaks = 10,
    guide = guide_legend(reverse = TRUE)
  ) +
  labs(
    fill = "",
    title = "Predicted suitability",
    subtitle = "For three runs"
  )
```



Normally, we take the median of the predictions since we only have one method 'maxent'. When more than one method, we may be careful to only get median of each method separately.

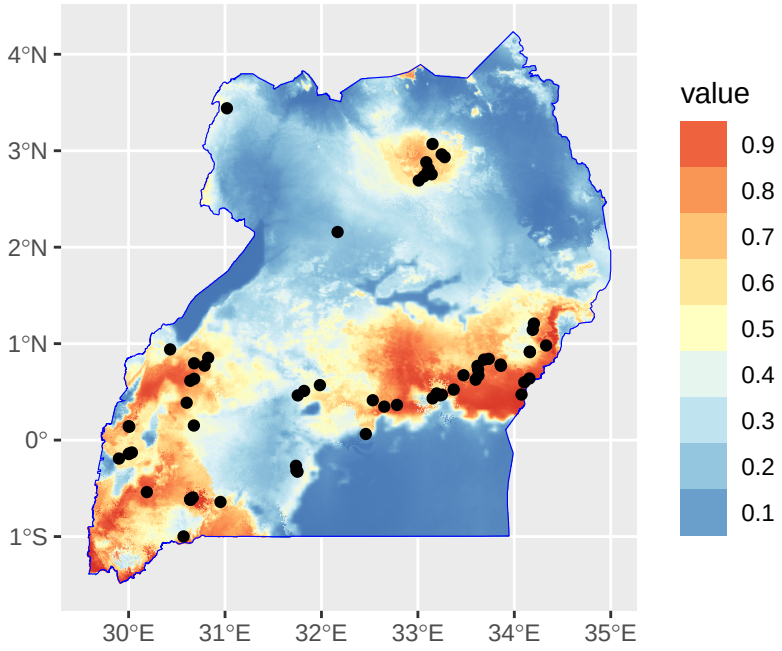
```
pred_median <- median(pred)
ggplot() +
  geom_spatraster(data = pred_median) +
  scale_fill_whitebox_c(
    palette = "bl_yl_rd",
    n.breaks = 10,
    guide = guide_legend(reverse = TRUE)
  ) +
  labs(
    fill = "",
    title = "Predicted suitability",
    subtitle = "For median"
  )
```



We can also visualize the map of the occurrence points on top of this prediction so that we visually inspect how well the model performed.

```
ggplot() +
  geom_spatraster(data = pred_median) +
  scale_fill_whitebox_c(
```

```
    palette = "bl_yl_rd",
    n.breaks = 10,
    guide = guide_legend(reverse = TRUE)) +
  geom_spatvector(data = uganda, fill = NA, col = "blue") +
  geom_spatvector(data = occ_ug)
```



0.6.1 Binarize the suitability map for presence absence map

We now need to make an important step which is to binarize the map into presence-absence map. This is a map which we can therefore use to tell how much land is under the weed. However, since we actually have not been in the field to make the observation of the fields, we use threshold based evaluation metrics to create a cut-off point along the 0 to 1 suitability scale. To get the evaluation metrics, we are going to compare the true presences and the background points with the predicted values on the map. This involves the following steps:

i. Getting the observed records and their locations (x, y, species)

```
obs_coords <- coords(sdm_data) # Get all coordinates
obs_species <- as.data.frame(sdm_data) |> dplyr::select(species) # Get species; 1 or 0
```

ii. Getting predicted values at the observed locations. Not so good with presence-background data.

```
pred_values <- terra::extract(pred_median, obs_coords)$median
```

Now we can use the two (observed values and predicted values) to run evaluation of the model.

```
eva <- evaluates(x = obs_species$species, p = pred_values)
```

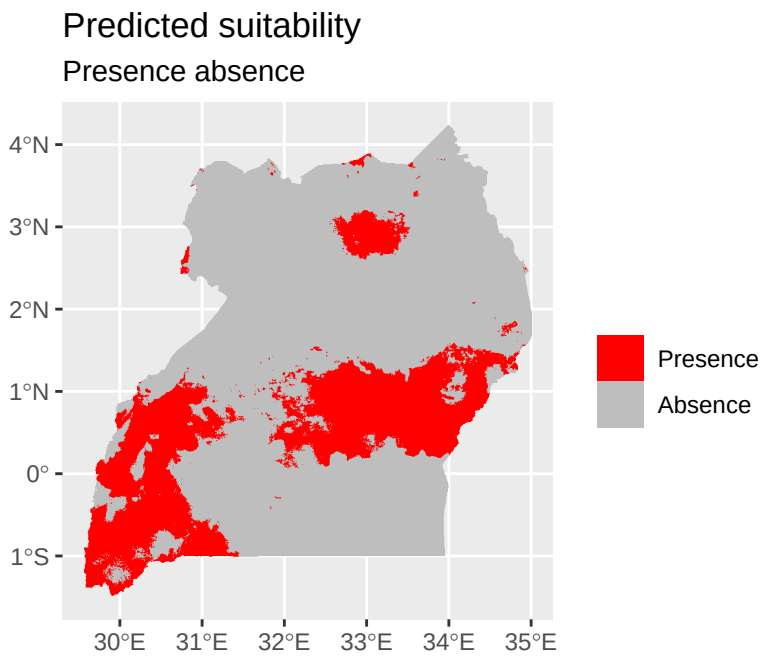
There are up to 15 threshold-based evaluation metrics which one can pick from. For this case we will go by the one which maximizes sensitivity and specificity. We will call it *th* for threshold.

```
th <- eva@threshold_based[2, 2]
th
```

```
[1] 0.48422
```

Now, with this threshold we can predict the potential presence points for the species.

```
pa <- pred_median
pa[] <- ifelse(pa[] >= th, 1, 0)
pa <- pa |> rename(present = median)
ggplot() +
  geom_spatraster(data = as.factor(pa)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "red"), # Map levels to colors
    labels = c("Absence", "Presence"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  labs(
    fill = "",                                # No label for legend
    title = "Predicted suitability",
    subtitle = "Presence absence"
  )
```



0.7 Predictions into the future

Since we have built our model, we can use it to make predictions across space and time. For instance, we can use the model to make prediction in another country, say Kenya, in case we currently did not have occurrence records in Kenya to train the model on. We can also use the same model to make predictions into the future.

```
fut_pred <- cmip6_tile(lon = 32,
                       lat = 1,
                       model = "HadGEM3-GC31-LL",
                       ssp = "245",
                       time = "2041-2060",
                       var = "bioc",
                       path = tempdir())
```

Let us visualize one layer from the downloaded tile.

```
plot(fut_pred[[1]])
```

Seems the tile does not cover the whole of our study area of interest as it covers only above the equator. Now we need additional three tiles in order to cover the whole of Uganda.

```
tile_ne <- cmip6_tile(lon = 32,
                      lat = 1,
                      model = "HadGEM3-GC31-LL",
                      ssp = "245",
                      time = "2041-2060",
                      var = "bioc",
                      path = tempdir())
tile_se <- cmip6_tile(lon = 32,
                      lat = -1,
                      model = "HadGEM3-GC31-LL",
                      ssp = "245",
                      time = "2041-2060",
                      var = "bioc",
                      path = tempdir())
tile_sw <- cmip6_tile(lon = 28,
                      lat = -1,
                      model = "HadGEM3-GC31-LL",
                      ssp = "245",
                      time = "2041-2060",
                      var = "bioc",
                      path = tempdir())
tile_nw <- cmip6_tile(lon = 28,
                      lat = 1,
                      model = "HadGEM3-GC31-LL",
                      ssp = "245",
                      time = "2041-2060",
                      var = "bioc",
                      path = tempdir())
```

Merging all the tiles:

```
# First we subset only the layers we used for training the model
sub_index <- c(2, 4, 8, 9, 13:14, 15, 18:19)
tile_ne_sub <- subset(tile_ne, sub_index)
tile_se_sub <- subset(tile_se, sub_index)
tile_sw_sub <- subset(tile_sw, sub_index)
tile_nw_sub <- subset(tile_nw, sub_index)
```

Now merge them:

```
merged_fut <- merge(tile_ne_sub, tile_se_sub, tile_sw_sub, tile_nw_sub)
```

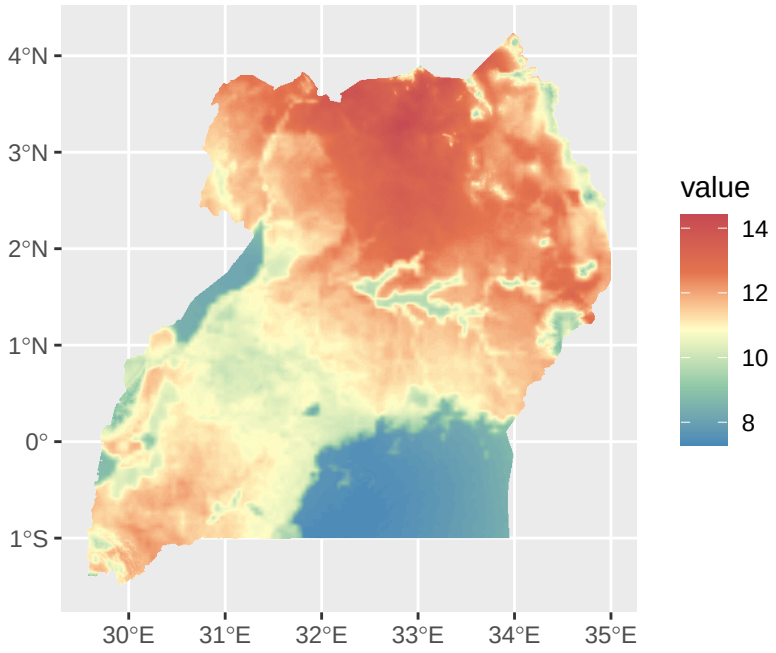
Visualize the one of the layers in the merged _fut:

```
plot(merged_fut[[1]])
```

Now we can crop it to the extent of Uganda and visualize.

```
merged_fut_cropped <- crop(merged_fut, uganda, mask = TRUE)
plot(merged_fut_cropped[[1]])
writeRaster(merged_fut_cropped, "data/future.tif")
```

```
merged_fut_cropped <- rast("data/future.tif")
ggplot() +
  geom_spatraster(data = merged_fut_cropped[[1]]) +
  scale_fill_whitebox_c(palette = "muted")
```



Now we can use out trained model to predict the potential suitable habitat for the species by the year 2041-2060 under shared socio-economic pathway 2.45 and using the **HadGEM3-GC31-LL** Global Circulation Model. But first we need to assign the same names to this future layers as that of the current layers where the model was trained.

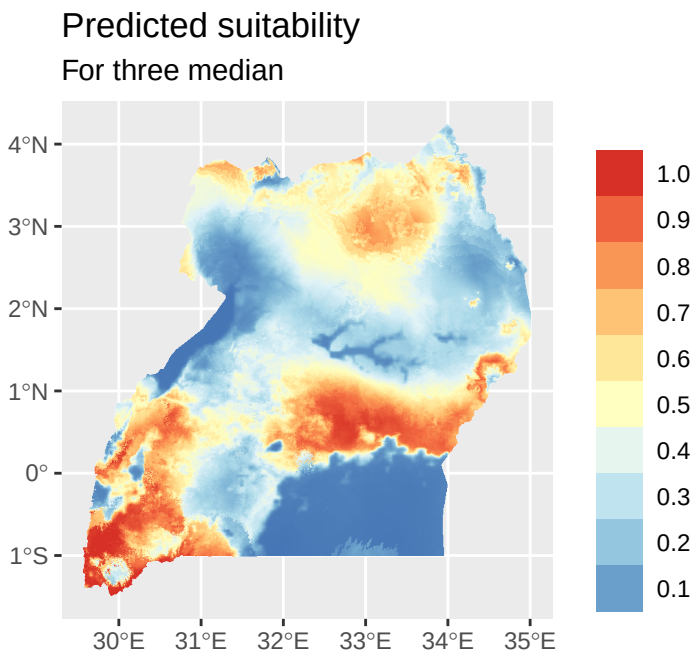
```
names(merged_fut_cropped) <- names(preds_ug_used)
```

Now we can go ahead with the prediction step.

```
fut_pred <- median(predict(object = sdm_model,
                           newdata = merged_fut_cropped))
```

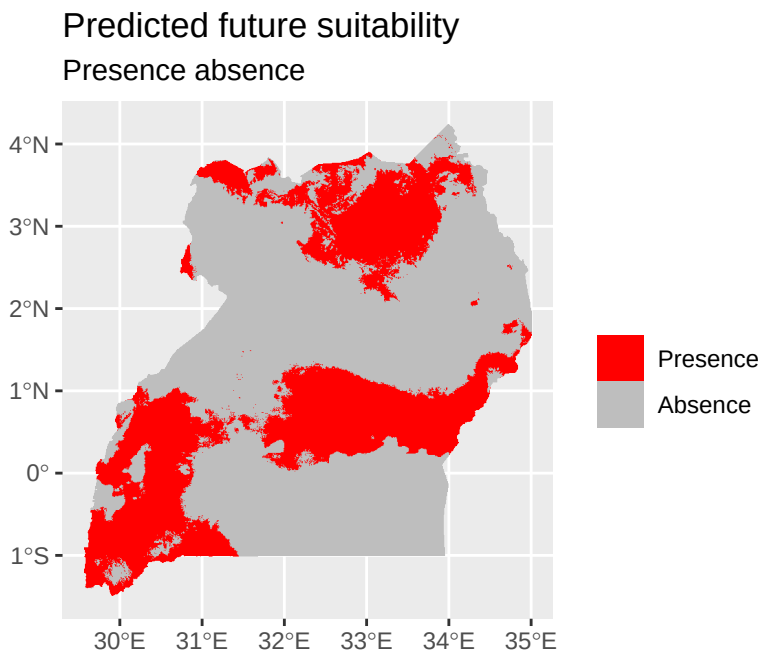
Visualize the future habitat suitability map.

```
ggplot() +
  geom_spatraster(data = fut_pred) +
  scale_fill_whitebox_c(
    palette = "bl_yl_rd",
    n.breaks = 10,
    guide = guide_legend(reverse = TRUE)
  ) +
  labs(
    fill = "",
    title = "Predicted suitability",
    subtitle = "For three median"
  )
```



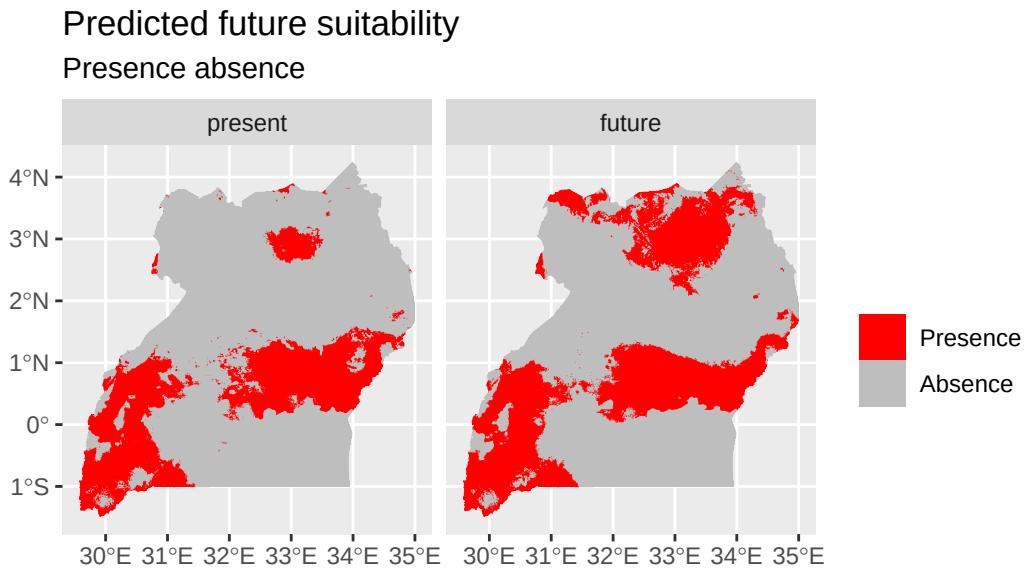
Using the same threshold we can create a binary map for the future.

```r
fut_pa <- fut_pred
fut_pa[] <- ifelse(fut_pa[] >= th, 1, 0)

fut_pa <- fut_pa |> rename(future = median)

ggplot() +
  geom_spatraster(data = as.factor(fut_pa)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "red"), # Map levels to colors
    labels = c("Absence", "Presence"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  labs(
    fill = "",                                  # No label for legend
    title = "Predicted future suitability",
    subtitle = "Presence absence"
  )
```



We can look at both maps side by side:

```r
#par(mfrow = c(1, 2))
pa <- as.factor(pa)
fut_pa <- as.factor(fut_pa)

curr_fut <- c(pa, fut_pa)

# plot(pa, main = "Current")
# plot(fut_pa, main = "Future")

ggplot() +
  geom_spatraster(data = curr_fut) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "red"), # Map levels to colors
    labels = c("Absence", "Presence"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  facet_wrap(~lyr, ncol = 2) +
  labs(
    fill = "",                                # No label for legend
    title = "Predicted future suitability",
    subtitle = "Presence absence"
  )
```

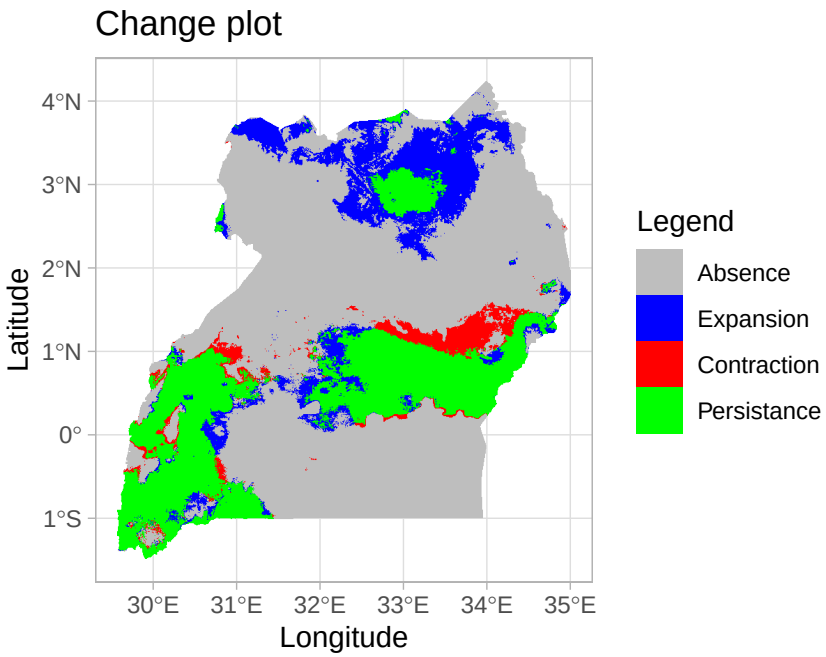


We can do a little logical operation to see which areas will experience expansion, contraction, persistence and absence.

```
change <- pa # ch for change
change[] <- ifelse(pa[] == 0 & fut_pa[] == 0, 0,
                ifelse(pa[] == 0 & fut_pa[] == 1, 1,
                       ifelse(pa[] == 1 & fut_pa[] == 0, 2, 3)))
```

We can plot the change.

```
ggplot() +
  geom_spatraster(data = as.factor(change)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "blue", "2" = "red", "3" = "green"),
    labels = c("Absence", "Expansion", "Contraction", "Persistance"),
    name = "Legend",
    na.translate = FALSE
  ) +
  labs(title = "Change plot",
       x = "Longitude",
       y = "Latitude") +
  theme_light()
```

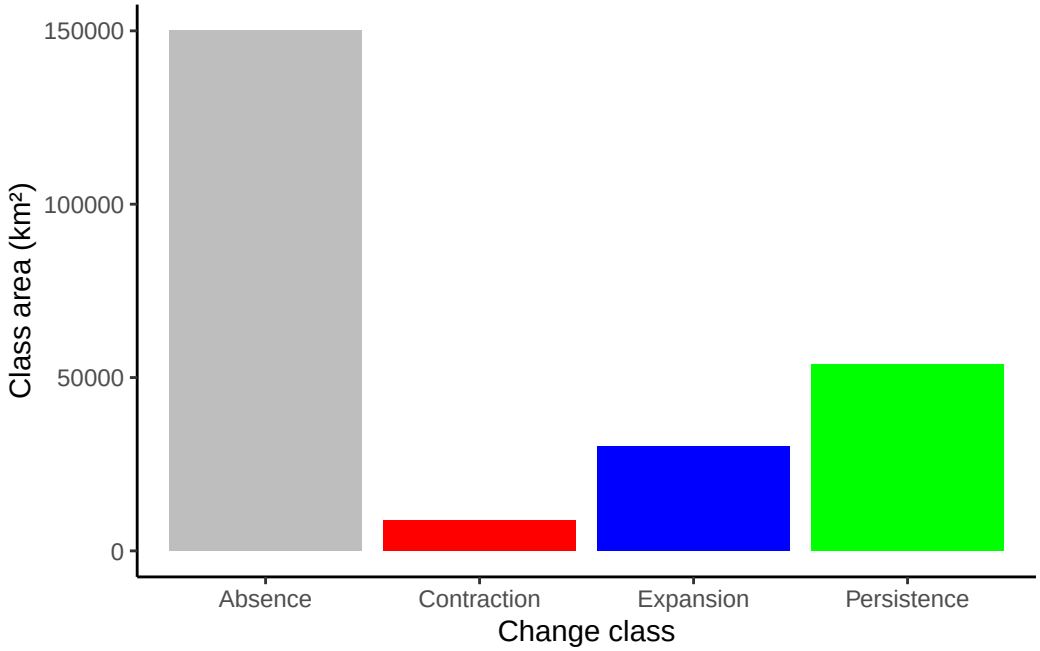


To know the area under each class we can use zonal statistics:

```
zonal_areas <- zonal(cellSize(change, unit = "km"), change, fun = "sum")
zonal_areas
```

```
  present       area
1       0 150064.609
2       1  30264.421
3       2   8836.178
4       3  53712.095
```

```
zonal_areas |>
  mutate(class = c("Absence", "Expansion", "Contraction", "Persistence")) |>
  ggplot() +
  geom_col(aes(x = class, y = area), fill = c("gray", "blue", "red", "green")) +
  labs(x = "Change class", y = "Class area (km²)") +
  theme_classic()
```

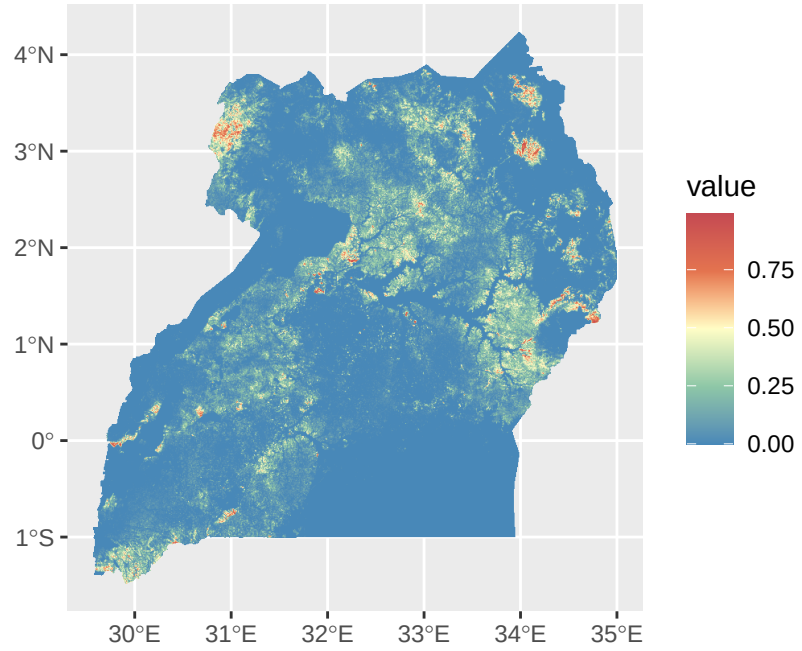


0.8 Cropland overlay

Lastly, we can find out how much of the Ugandan cropland will potentially be affected by the species in future. For that we will need the cropland map of Uganda. We can obtain cropland data by using *cropland* function in `geodata` package. Specifically, we use the WorldCover (Zanaga et al. 2021) fractional cropland data.

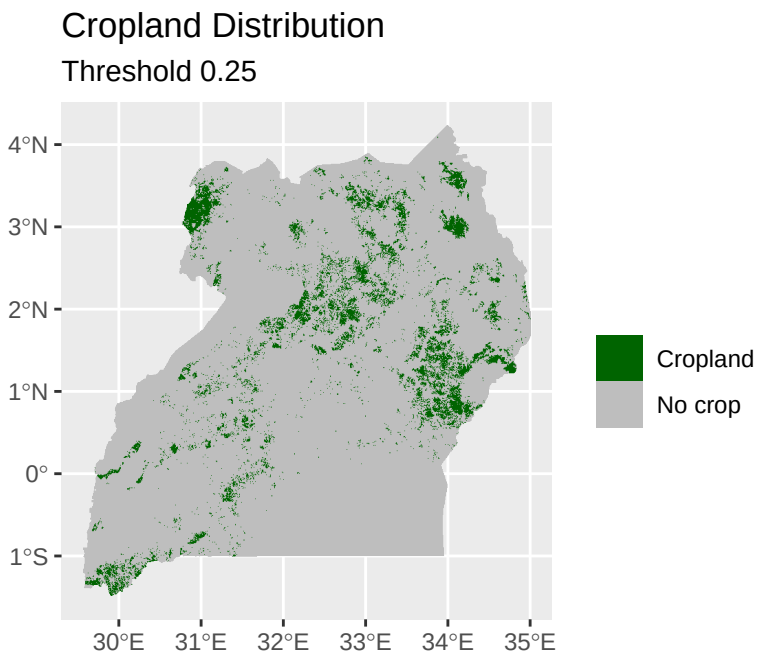
```
cropland_ug <- cropland(source = "WorldCover", path = "data/cropland_ug.tif")
cropland_ug_ <- crop(cropland_ug, uganda, mask = TRUE)
writeRaster(cropland_ug_, "data/cropland_ug_cropped.tif")
```

```
ug_cropland <- rast("data/cropland_ug_cropped.tif")
ggplot() +
  geom_spatraster(data = ug_cropland) +
  scale_fill_whitebox_c(palette = "muted")
```



The pixel values represent the fraction of cropland in each cell. Let us assume that where the fraction is above 0.25 is considered cropland. The aim is to have a binary map of cropland or no cropland.

```
crop_bin <- ug_cropland
crop_bin[] <- ifelse(crop_bin[] >= 0.25, 1, 0)
ggplot() +
  geom_spatraster(data = as.factor(crop_bin)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "darkgreen"), # Map levels to colors
    labels = c("No crop", "Cropland"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  labs(
    fill = "",                                # No label for legend
    title = "Cropland Distribution",
    subtitle = "Threshold 0.25"
  )
```

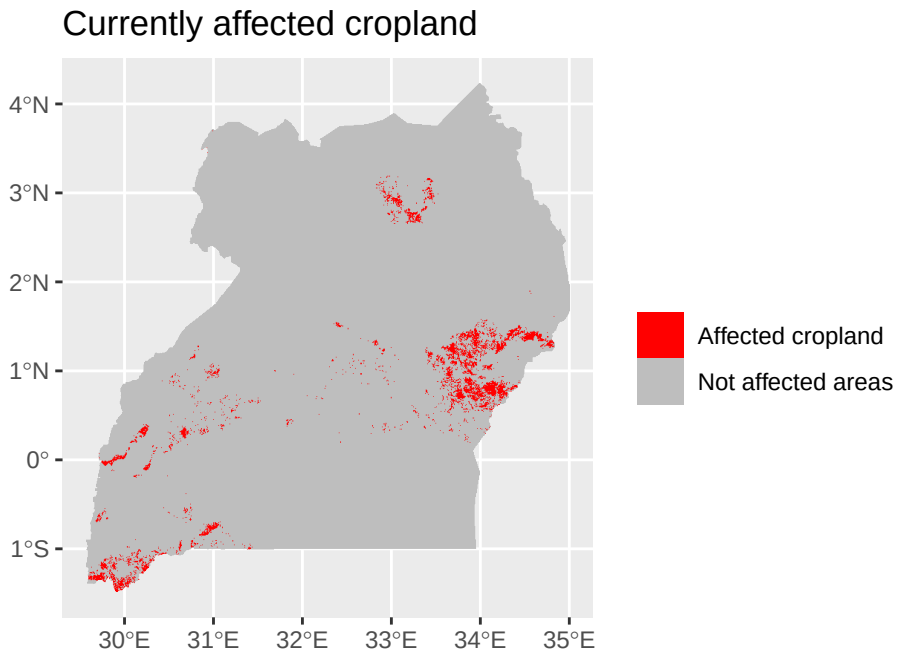


Now we can determine how much of the current cropland is affected by the weed and also we can predict how much of the future land may likely be affected by the invasive species.

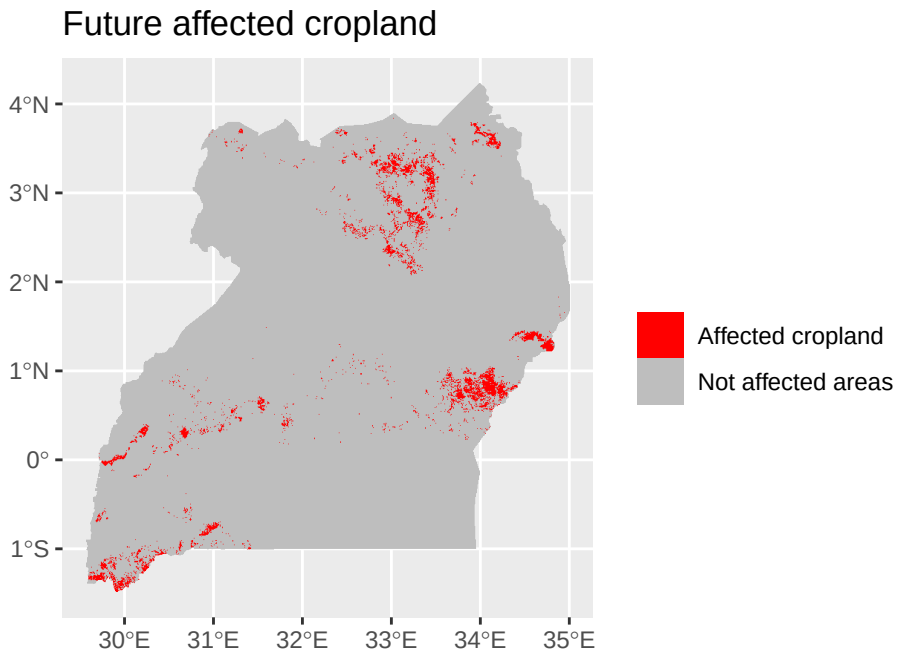
```
current_affected <- pa
future_affected <- pa

current_affected[] <- ifelse(pa[] == 1 & crop_bin[] == 1, 1, 0)
future_affected[] <- ifelse(fut_pa[] == 1 & crop_bin[] == 1, 1, 0)
```

Visualize them

```
ggplot() +
  geom_spatraster(data = as.factor(current_affected)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "red"), # Map levels to colors
    labels = c("Not affected areas", "Affected cropland"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  labs(
    fill = "",                                # No label for legend
    title = "Currently affected cropland",
  )
```

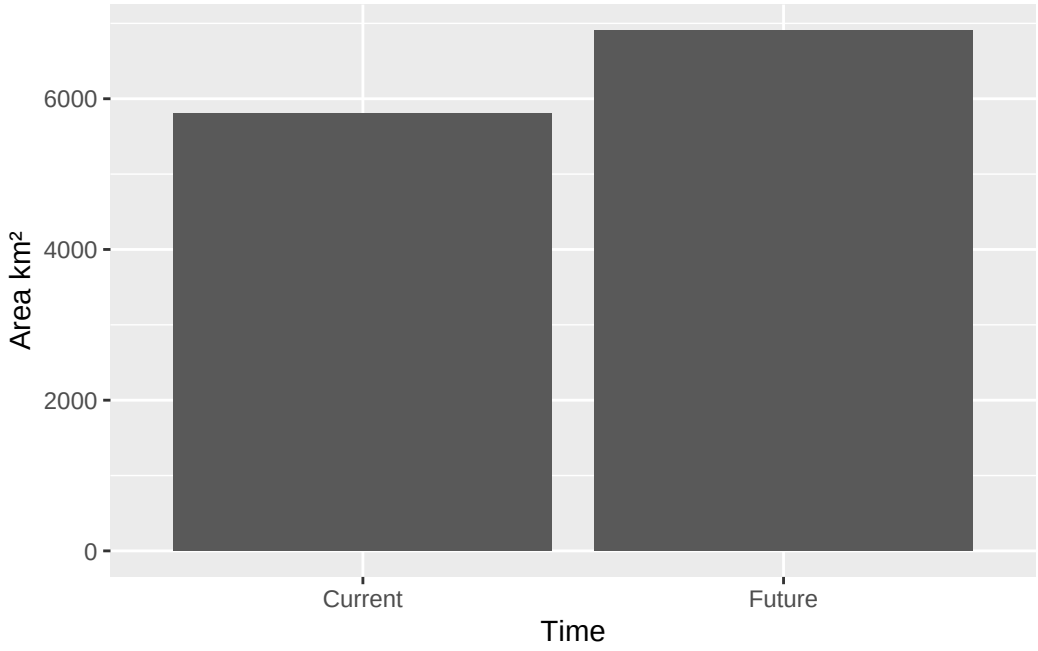



```
ggplot() +
  geom_spatraster(data = as.factor(future_affected)) +
  scale_fill_manual(
    values = c("0" = "gray", "1" = "red"), # Map levels to colors
    labels = c("Not affected areas", "Affected cropland"),
    guide = guide_legend(reverse = TRUE),
    na.translate = FALSE# Reverse legend order
  ) +
  labs(
    fill = "",                                # No label for legend
    title = "Future affected cropland",
  )
```




```
curr_affected_area <- zonal(cellSize(current_affected, unit = "km"), current_affected, fun =
fut_affected_area <- zonal(cellSize(future_affected, unit = "km"), future_affected, fun = "su
```

```
df_affected <- tibble(x = c("Current", "Future"),
                      y = c(curr_affected_area$area[2], fut_affected_area$area[2]))

df_affected |> ggplot(aes(x, y)) +
  geom_col() +
  labs(x = "Time",
       y = "Area km²")
```



0.9 Conclusion: Key Observations from the *Parthenium hysterophorus*

The map of potential suitable habitat of *Parthenium hysterophorus* suggest expansion with climate change. In the **Cental and Southwestern regions** the species is likely to pose more harm to farmers as it characterizes highly suitable environment for the species. Even though northern Uganda is currently less suitable for the species, the model predicts rapid expansion in habitat suitability for the species in the region into the 2041 - 2060 period. Using the cropland map from WorldCover, we predict that the invasive species is set to pose more havoc for croplands in Uganda in the future.

Feel free to reach me out if you got any questions.

Diagne, Christophe, Boris Leroy, Anne-Charlotte Vaissière, Rodolphe E. Gozlan, David Roiz, Ivan Jarić, Jean-Michel Salles, Corey J. A. Bradshaw, and Franck Courchamp. 2021. "High and Rising Economic Costs of Biological Invasions Worldwide." *Nature* 592 (7855): 571–76. https://doi.org/10.1038/s41586-021-03405-6.

Fick, Stephen E., and Robert J. Hijmans. 2017. "WorldClim 2: New 1-Km Spatial Resolution Climate Surfaces for Global Land Areas." *International Journal of Climatology*. https://doi.org/10.1002/joc.5086.

GBIF.org. 2020. "GBIF Home Page." https://www.gbif.org.

Hernangómez, Diego. 2023. "Using the {Tidyverse} with {Terra} Objects: The {Tidyterra} Package" 8: 5751. https://doi.org/10.21105/joss.05751.

Hijmans, Robert J. 2024. "Terra: Spatial Data Analysis." https://CRAN.R-project.org/package=terra.

Hijmans, Robert J., Márcia Barbosa, Aniruddha Ghosh, and Alex Mandel. 2024. "Geodata: Download Geographic Data." https://CRAN.R-project.org/package=geodata.

Hijmans, Robert J., Steven Phillips, John Leathwick, and Jane Elith. 2023. "Dismo: Species Distribution Modeling." https://CRAN.R-project.org/package=dismo.

J. M. Kass, G. E. Pinilla-Buitrago, A. Paz, B. A. Johnson, V. Grisales-Betancur, S. I. Meenan, D. Attali, et al. 2023. "Wallace 2: A Shiny App for Modeling Species Niches and Distributions Redesigned to Facilitate Expansion via Module Contributions" 2023(3): 1–9. https://onlinelibrary.wiley.com/doi/10.1111/ecog.06547.

Kass, J. M., R. Muscarella, P. J. Galante, C. L. Bohl, G. E. Pinilla-Buitrago, R. A. Boria, M. Soley-Guardia, and R. P. Anderson. 2021. "ENMeval 2.0: Redesigned for Customizable and Reproducible Modeling of Species' Niches and Distributions" 12: 1602–8. https://doi.org/10.1111/2041-210X.13628.

Naimi, Babak, and Miguel B. Araujo. 2016. "Sdm: A Reproducible and Extensible r Platform for Species Distribution Modelling" 39: 368–75. https://doi.org/10.1111/ecog.01881.

Naimi, Babak, Nicholas a.s. Hamm, Thomas A. Groen, Andrew K. Skidmore, and Albertus G. Toxopeus. 2014. "Where Is Positional Uncertainty a Problem for Species Distribution Modelling" 37: 191–203. https://doi.org/10.1111/j.1600-0587.2013.00205.x.

R Core Team. 2023. "R: A Language and Environment for Statistical Computing." https://www.r-project.org/.

Thuiller, Wilfried, Damien Georges, Maya Gueguen, Robin Engler, Frank Breiner, Bruno Lafourcade, Remi Patin, and Helene Blancheteau. 2024. "Biomod2: Ensemble Platform for Species Distribution Modeling." https://CRAN.R-project.org/package=biomod2.

Warren, Dan, and Russell Dinnage. 2024. "ENMTools: Analysis of Niche Evolution Using Niche and Distribution Models." https://CRAN.R-project.org/package=ENMTools.

Zanaga, Daniele, Ruben Van De Kerchove, Wanda De Keersmaecker, Niels Souverijns, Carsten Brockmann, Ralf Quast, Jan Wevers, et al. 2021. "ESA WorldCover 10 m 2020 V100." Zenodo. https://doi.org/10.5281/ZENODO.5571936.