Deforestation-to-biodiversity

Ulas Ayyilmaz and Ishika

```r
#change between 2000-2020 #https://www.globalforestwatch.org/dashboards/global/?category=fore
forest_data <- read_csv("net_tree_change.csv")
head(forest_data)
```

```
# A tibble: 6 x 8
  iso   stable                    loss    gain  disturb net   change gfw_area__ha
  <chr> <chr>                     <chr>   <chr> <chr>   <chr>  <dbl>        <dbl>
1 ABW   113.038689599999998       6.7644~ 19.2~ 0.6764~ 12.4~  10.4         18194.
2 AFG   368081.866700000013225    16604.~ 1074~ 1146.2~ -586~  -1.52    64385933.
3 AGO   44546560.189999997615814 341261~ 1224~ 162392~ -218~  -4.41   124742581.
4 AIA   890.152399199999991       51.507~ 69.8~ 9.7319~ 18.3~   1.93         8330.
5 ALA   76419.028500000000349     7426.9~ 2582~ 12417.~ -484~  -5.03       150631.
6 ALB   814631.915999999968335    40701.~ 1647~ 41219.~ -242~  -2.70      2873543.
```

```r
top_5_loss <- forest_data |>
  arrange(desc(as.numeric(loss))) |>
  slice_head(n = 20)
top_5_gain <- forest_data |>
  arrange(desc(as.numeric(gain))) |>
  slice_head(n = 20)

top_5_net_desc <- forest_data |>
  arrange(desc(as.numeric(net))) |>
  slice_head(n = 20)

top_5_net_asc <- forest_data |>
  arrange(as.numeric(net)) |>
  slice_head(n = 20)

top_5_net_desc # poland, ukraine, uruguay, ireland, bangladesh
```

```
# A tibble: 20 x 8
   iso   stable                     loss  gain  disturb net   change gfw_area__ha
   <chr> <chr>                      <chr> <chr> <chr>   <chr>  <dbl>        <dbl>
 1 CHN   202215007.599999994039536 4544~ 6689~ 171769~ 2144~  0.958   938225140.
 2 IND   59241391.750000000000000  1007~ 1881~ 617156~ 8740~  1.32    315280003.
 3 URY   659122.222900000051595    1127~ 6544~ 230107~ 5417~ 54.1      17746563.
 4 BLR   7407088.671000000089407   4421~ 9636~ 683220~ 5214~  6.11     20706981.
 5 UKR   9065233.101999999955297   4998~ 9263~ 895296~ 4265~  4.08     60136264.
 6 POL   9114498.453999999910593   4853~ 8920~ 975192~ 4066~  3.85     31240006.
 7 SSD   25841911.940000001341105  8019~ 1082~ 96714.~ 2809~  1.05     62810211.
 8 BGD   3313032.510999999940395   1154~ 3178~ 612964~ 2024~  5.01     13938445.
 9 IRL   388037.844199999992270    9463~ 2842~ 77309.~ 1896~ 33.9       7036677.
10 SDN   3715927.944000000134110   8530~ 2165~ 2345.3~ 1312~  3.45    187212843.
11 LTU   1722243.843000000109896   1166~ 2415~ 315129~ 1248~  5.80      6501684.
12 ROU   7427728.728000000119209   1057~ 2238~ 534902~ 1181~  1.46     23833860.
13 NPL   6718407.851999999955297   7372~ 1740~ 55029.~ 1002~  1.46     14766264.
14 PAK   2057786.046000000089407   3869~ 1334~ 7457.5~ 9478~  4.51     87417714.
15 THA   18845385.170000001788139  1747~ 1842~ 405750~ 9441~  0.383    51405456.
16 TUR   12756770.300000000745058  3850~ 4737~ 510036~ 8876~  0.650    78070436.
17 HUN   1677658.310000000055879   1226~ 2073~ 249845~ 8463~  4.13      9305287.
18 GBR   2421686.652999999932945   3915~ 4731~ 184814~ 8156~  2.72     24548584.
19 GEO   3238385.424999999813735   1795~ 9353~ 7227.5~ 7558~  2.32      6984461.
20 PRK   6061637.154000000096858   2135~ 2884~ 394941~ 7497~  1.12     12275513.
```

```
top_5_net_asc#tanzania, Mozambique, indonesia, DCcongo, paraguay
```

```
# A tibble: 20 x 8
   iso   stable                     loss  gain  disturb net   change gfw_area__ha
   <chr> <chr>                      <chr> <chr> <chr>   <chr>  <dbl>        <dbl>
 1 BRA   413722809.300000011920929 3614~ 8062~ 234216~ -280~  -5.93   850036547.
 2 CAN   257102961.500000000000000 2516~ 1696~ 148636~ -820~  -2.76   995519376.
 3 COD   146862160.800000011920929 7593~ 1591~ 145662~ -600~  -3.55   232913006.
 4 PRY   14232500.500000000000000  5810~ 6424~ 883902~ -516~ -24.7     39958592.
 5 MOZ   35317855.280000001192093  4897~ 5831~ 202499~ -431~ -10.2     78729703.
 6 IDN   124187188.599999994039536 9000~ 4882~ 240569~ -411~  -2.62   189024479.
 7 TZA   27968218.850000001490116  4374~ 5572~ 143782~ -381~ -11.3     94057605.
 8 ARG   27610827.399999998509884  4664~ 1107~ 174446~ -355~ -10.5    278009661.
 9 USA   237725811.800000011920929 1747~ 1398~ 283653~ -348~  -1.23   947301497.
10 BOL   52561114.219999998807907  3941~ 6172~ 270692~ -332~  -5.61   108339299.
11 ZMB   34159461.780000001192093  3640~ 7681~ 151400~ -287~  -7.31    75049045.
12 KHM   6195015.302000000141561   2787~ 1473~ 133709~ -264~ -25.6     18135962.
13 AGO   44546560.189999997615814  3412~ 1224~ 162392~ -218~  -4.41   124742581.
```

```
14 COL   73177488.349999994039536  2825~ 1085~ 328654~ -174~  -2.20   113675882.
15 CIV   16844645.399999998509884  2530~ 8816~ 437764~ -164~  -6.94    32165807.
16 MMR   35420448.899999998509884  2227~ 6366~ 839121~ -159~  -3.46    66929741.
17 GIN   14569792.289999999105930  1862~ 2726~ 262048~ -158~  -8.34    24481617.
18 LAO   13290214.820000000298023  1573~ 1017~ 568687~ -147~  -7.16    23000163.
19 NGA   20636992.129999998956919  2396~ 9278~ 118123~ -146~  -6.07    90841014.
20 VEN   51803026.079999998211861  1860~ 4913~ 133297~ -136~  -2.49    91249272.
```

order of interest (check exist for each country)

Most net negative: PRY, COD, MOZ, IDN, TZA

most positive: URY, UKR, POL,IRL, BGD

All columns of a RGBIF EOD dataset #"gbifID", "datasetKey", "occurrenceID", "kingdom", "phylum", "class", "order", "family", #"genus", "species", "infraspecificEpithet", "taxonRank", "scientificName", #"verbatimScientificName", "verbatimScientificNameAuthorship", "countryCode", "locality", #"stateProvince", "occurrenceStatus", "individualCount", "publishingOrgKey", #"decimalLatitude", "decimalLongitude", "coordinateUncertaintyInMeters", #"coordinatePrecision", "elevation", "elevationAccuracy", "depth", "depthAccuracy", #"eventDate", "day", "month", "year", "taxonKey", "speciesKey", "basisOfRecord", #"institutionCode", "collectionCode", "catalogNumber", "recordNumber", "identifiedBy", #"dateIdentified", "license", "rightsHolder", "recordedBy", "typeStatus", #"establishmentMeans", "lastInterpreted", "mediaType", "issue")

Ok, brainstorm time. Deforestation means removel of forests over time due to mostly due to human interference. Deforestation affects many things that contribute to human's well-being indirectly in a negative way. A direct effect is observed on the biodiversity - specifically birds who roam freely in forests. Other

```r
bird_data<-name_suggest(q = "Aves", rank = "class", curlopts = list(timeout = 60))
bird_taxon_key <- bird_data$data$key
eod_dataset_key <- "4fa7b334-ce0d-4e88-aaae-2e0c138d049e"  # EOD datasetKey
```

```r
# Define the target countries and years
years <- c(2000, 2010, 2020)  # Year range
eod_dataset_key <- "4fa7b334-ce0d-4e88-aaae-2e0c138d049e"  # EOD datasetKey
orders <- c("Coraciiformes","Strigiformes","Galliformes","Ciconiiformes")
```

```r
# Load necessary library
# Initialize an empty dataframe to store the combined data
combined_data <- data.frame()

# List of country names (folders inside the "data" directory)
```

```r
countries <- c( "poland","ukraine", "uruguay", "ireland", "bangladesh",
               "tanzania", "mozambique", "indonesia", "dcongo", "paraguay")
# Loop through each country's folder and combine all CSV files
for (country in countries) {

  # Path to the country's folder
  country_path <- file.path("data", country)

    # Get the list of CSV files in the country's folder
  csv_files <- list.files(country_path, pattern = "\\.csv$", full.names = TRUE)

    # Read each CSV file and add its content to the combined dataframe
  for (csv_file in csv_files) {
   tryCatch({
      # Read the CSV file (use read_delim for robustness)
      data <- read_delim(csv_file, delim = NULL, show_col_types = FALSE)

      # Add a column to identify the country
      data$country <- country

      # Append the data to the combined dataframe
      combined_data <- bind_rows(combined_data, data)
    }, error = function(e) {
      cat("Error reading file:", csv_file, "\n")
    })
  }
}

# Save the combined dataframe as a CSV file in the main directory
write.csv(combined_data, "all_bird_orders_data.csv", row.names = FALSE)

cat("Combined CSV file created as 'all_bird_orders_data.csv' in the main directory.\n")
```

```
Combined CSV file created as 'all_bird_orders_data.csv' in the main directory.
```

m

```r
bird_orders <- read.csv("all_bird_orders_data.csv")
head(bird_orders)
```

```
     gbifID                                 datasetKey
```

```
1 972752070 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
2 962063576 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
3 956424085 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
4 956423830 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
5 956334143 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
6 584375475 4fa7b334-ce0d-4e88-aaae-2e0c138d049e
                        occurrenceID  kingdom   phylum class        order
1 URN:catalog:CLO:EBIRD:OBS219042257 Animalia Chordata  Aves Ciconiiformes
2 URN:catalog:CLO:EBIRD:OBS207389323 Animalia Chordata  Aves Ciconiiformes
3 URN:catalog:CLO:EBIRD:OBS200802257 Animalia Chordata  Aves Ciconiiformes
4 URN:catalog:CLO:EBIRD:OBS200798945 Animalia Chordata  Aves Ciconiiformes
5 URN:catalog:CLO:EBIRD:OBS200758554 Animalia Chordata  Aves Ciconiiformes
6  URN:catalog:CLO:EBIRD:OBS96224279 Animalia Chordata  Aves Ciconiiformes
     family   genus         species infraspecificEpithet taxonRank
1 Ciconiidae Ciconia Ciconia ciconia                   NA   SPECIES
2 Ciconiidae Ciconia   Ciconia nigra                   NA   SPECIES
3 Ciconiidae Ciconia Ciconia ciconia                   NA   SPECIES
4 Ciconiidae Ciconia Ciconia ciconia                   NA   SPECIES
5 Ciconiidae Ciconia Ciconia ciconia                   NA   SPECIES
6 Ciconiidae Ciconia Ciconia ciconia                   NA   SPECIES
                 scientificName verbatimScientificName
1 Ciconia ciconia (Linnaeus, 1758)       Ciconia ciconia
2   Ciconia nigra (Linnaeus, 1758)         Ciconia nigra
3 Ciconia ciconia (Linnaeus, 1758)       Ciconia ciconia
4 Ciconia ciconia (Linnaeus, 1758)       Ciconia ciconia
5 Ciconia ciconia (Linnaeus, 1758)       Ciconia ciconia
6 Ciconia ciconia (Linnaeus, 1758)       Ciconia ciconia
  verbatimScientificNameAuthorship countryCode
1                               NA          PL
2                               NA          PL
3                               NA          PL
4                               NA          PL
5                               NA          PL
6                               NA          PL
                             locality  stateProvince occurrenceStatus
1                           Sandomierz Swietokrzyskie          PRESENT
2              Highway, southern Poland    Malopolskie          PRESENT
3 Białowieża National Park (IBA PL046)      Podlaskie          PRESENT
4               Biebrza NP--Dobarz area      Podlaskie          PRESENT
5             Reserwat Mierzeja Sarbska      Pomorskie          PRESENT
6                 Poleski National Park      Lubelskie          PRESENT
  individualCount                     publishingOrgKey decimalLatitude
1              NA e2e717bf-551a-4917-bdc9-4fa0f342c530        50.67987
```

```
2               1 e2e717bf-551a-4917-bdc9-4fa0f342c530        49.64294
3              NA e2e717bf-551a-4917-bdc9-4fa0f342c530        52.77079
4              NA e2e717bf-551a-4917-bdc9-4fa0f342c530        53.36978
5               2 e2e717bf-551a-4917-bdc9-4fa0f342c530        54.77334
6               2 e2e717bf-551a-4917-bdc9-4fa0f342c530        51.37454
  decimalLongitude coordinateUncertaintyInMeters coordinatePrecision elevation
1         21.76563                            NA                  NA        NA
2         19.92021                            NA                  NA        NA
3         23.85561                            NA                  NA        NA
4         22.59019                            NA                  NA        NA
5         17.62782                            NA                  NA        NA
6         23.04717                            NA                  NA        NA
  elevationAccuracy depth depthAccuracy  eventDate day month year taxonKey
1                NA    NA            NA 2010-07-18  18     7 2010  2481912
2                NA    NA            NA 2010-07-21  21     7 2010  2481909
3                NA    NA            NA 2000-07-28  28     7 2000  2481912
4                NA    NA            NA 2000-07-28  28     7 2000  2481912
5                NA    NA            NA 2000-07-24  24     7 2000  2481912
6                NA    NA            NA 2010-07-30  30     7 2010  2481912
  speciesKey     basisOfRecord institutionCode collectionCode catalogNumber
1    2481912 HUMAN_OBSERVATION             CLO          EBIRD  OBS219042257
2    2481909 HUMAN_OBSERVATION             CLO          EBIRD  OBS207389323
3    2481912 HUMAN_OBSERVATION             CLO          EBIRD  OBS200802257
4    2481912 HUMAN_OBSERVATION             CLO          EBIRD  OBS200798945
5    2481912 HUMAN_OBSERVATION             CLO          EBIRD  OBS200758554
6    2481912 HUMAN_OBSERVATION             CLO          EBIRD   OBS96224279
  recordNumber identifiedBy dateIdentified   license rightsHolder recordedBy
1           NA           NA             NA CC_BY_4_0           NA obsr426265
2           NA           NA             NA CC_BY_4_0           NA obsr116419
3           NA           NA             NA CC_BY_4_0           NA obsr119031
4           NA           NA             NA CC_BY_4_0           NA obsr119031
5           NA           NA             NA CC_BY_4_0           NA obsr119031
6           NA           NA             NA CC_BY_4_0           NA obsr116419
  typeStatus establishmentMeans         lastInterpreted mediaType
1         NA                 NA 2024-04-17 08:23:22.915        NA
2         NA                 NA 2024-04-17 08:24:17.292        NA
3         NA                 NA 2024-04-17 09:07:13.585        NA
4         NA                 NA 2024-04-17 08:45:14.385        NA
5         NA                 NA 2024-04-17 09:51:22.303        NA
6         NA                 NA 2024-04-17 09:02:29.332        NA
                                                                    issue
1 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
2 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
```

```
3 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
4 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
5 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
6 CONTINENT_DERIVED_FROM_COORDINATES;TAXON_MATCH_TAXON_CONCEPT_ID_IGNORED
  country
1  poland
2  poland
3  poland
4  poland
5  poland
6  poland
```

```
#596024662,584212444
```

info: all belong to aves class #important columns to keep: order, family, genus, species, stateProvince, individualCount, decimalLatitude, elevation,decimalLongitude, day,month, year, country

```
filtered_bird_orders <- bird_orders |>
  filter(order %in% orders)|>
  select(order, family, genus, species, stateProvince, individualCount, decimalLatitude, ele
head(filtered_bird_orders)
```

```
          order     family   genus         species  stateProvince
1 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia Swietokrzyskie
2 Ciconiiformes Ciconiidae Ciconia   Ciconia nigra    Malopolskie
3 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Podlaskie
4 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Podlaskie
5 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Pomorskie
6 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Lubelskie
  individualCount decimalLatitude elevation decimalLongitude day month year
1              NA        50.67987        NA         21.76563  18     7 2010
2               1        49.64294        NA         19.92021  21     7 2010
3              NA        52.77079        NA         23.85561  28     7 2000
4              NA        53.36978        NA         22.59019  28     7 2000
5               2        54.77334        NA         17.62782  24     7 2000
6               2        51.37454        NA         23.04717  30     7 2010
  country
1  poland
2  poland
3  poland
4  poland
```

```
5  poland
6  poland
```

```r
country_mapping <- data.frame(
  iso = c("POL", "UKR", "URY", "IRL", "BGD", "TZA", "MOZ", "IDN", "COD", "PRY"),
  country = c("poland", "ukraine", "uruguay", "ireland", "bangladesh",
              "tanzania", "mozambique", "indonesia", "dcongo", "paraguay")
)

filtered_forest_data <- forest_data %>%
  filter(iso %in% country_mapping$iso) %>%   # Keep only rows with matching ISO codes
  left_join(country_mapping, by = "iso")
```

```r
# Combine the datasets based on the "country" column
joined_data <- left_join(filtered_bird_orders, filtered_forest_data, by = "country")

# Output the combined dataset
write.csv(joined_data, "filtered_forest_data.csv")
head(joined_data)
```

```
          order     family   genus         species  stateProvince
1 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia Swietokrzyskie
2 Ciconiiformes Ciconiidae Ciconia   Ciconia nigra    Malopolskie
3 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Podlaskie
4 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Podlaskie
5 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Pomorskie
6 Ciconiiformes Ciconiidae Ciconia Ciconia ciconia      Lubelskie
  individualCount decimalLatitude elevation decimalLongitude day month year
1              NA        50.67987        NA         21.76563  18     7 2010
2               1        49.64294        NA         19.92021  21     7 2010
3              NA        52.77079        NA         23.85561  28     7 2000
4              NA        53.36978        NA         22.59019  28     7 2000
5               2        54.77334        NA         17.62782  24     7 2000
6               2        51.37454        NA         23.04717  30     7 2010
  country iso                 stable                  loss
1  poland POL 9114498.453999999910593 485398.353599999973085
2  poland POL 9114498.453999999910593 485398.353599999973085
3  poland POL 9114498.453999999910593 485398.353599999973085
4  poland POL 9114498.453999999910593 485398.353599999973085
5  poland POL 9114498.453999999910593 485398.353599999973085
6  poland POL 9114498.453999999910593 485398.353599999973085
                 gain                disturb                    net   change
```

```
1 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
2 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
3 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
4 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
5 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
6 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
  gfw_area__ha
1     31240006
2     31240006
3     31240006
4     31240006
5     31240006
6     31240006
```

```
#get rid of NA's in joined data
joined_data1 <- joined_data |>
  filter(!is.na(individualCount))
```

```
library(sf)
library(tidyr)
library(rnaturalearth)
```

```
order_counts <- joined_data1 %>%
  group_by(order, country, year) %>%
  summarize(order_count = sum(individualCount), .groups = "drop")
library(dplyr)
```

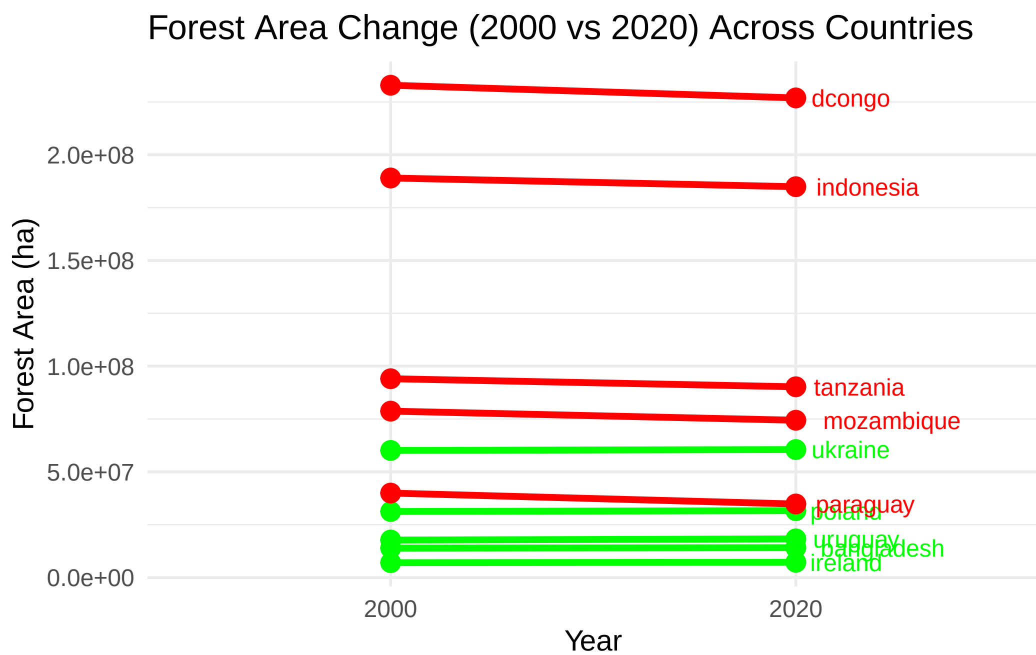
```
#plot bird number changes
plot <-
  ggplot(order_counts %>% filter(country == "bangladesh"), aes(x = year, y = order_count, lal
    geom_point() +
    geom_text(vjust = -0.5, hjust = 0.5, size = 3) +
    labs(
      title = paste("Counts for", "Bangladesh"),
      x = "Year",
      y = "Order Count"
    ) +
    theme_bw()

ggsave(paste("plots/Bangladesh.png"))
```

```
# country_density_map <- joined_data1 |>
#   group_by(country,order,year)
#
# # Determine the global range of individualCount for consistent scaling
# global_size_range <- range(country_density_map$individualCount, na.rm = TRUE)
#
# # Iterate through each bird order and create maps for each country and year
# bird_orders <- unique(country_density_map$order)
#
# # Create a list to store plots
# plots <- list()
# c = 0
# world_map <- rnaturalearth::ne_countries(scale = "medium", returnclass = "sf")
# country_maps <- world_map |> filter(name_long %in% countries)
#
# for (order1 in bird_orders) {
#   for (country1 in countries) {
#     for (year1 in years) {
#       c <- c + 1
#       # Filter data for the specific bird order, year, and country
#       specific_data <- country_density_map |>
#         filter(order == order1, year == year1, country == country1)
#
#       # Join country map with bird observation data
#       country_map <- country_maps |> filter(name_long == country1)
#
#       # Create the plot
#       plots[[c]] <-
#         ggplot() +
#         geom_sf(data = country_map, fill = "gray90", color = "black") +
#         geom_point(data = specific_data,
#                    aes(x = decimalLongitude, y = decimalLatitude, size = individualCount),
#                    alpha = 0.7) +
#         scale_size_continuous(name = "Count", range = c(1, 10), limits = global_size_range
#         ggtitle(paste("Bird Order:", order1, "Year:", year1, "Country:", country1)) +
#         theme_bw()
#
#       ggsave(paste("plot_geodistribution/bird_order:", order1, "year:", year1, "country:",
#
#     }
#   }
# }
```

```
# ggsave("bird_density_map.png", plots[[1]], width = 10, height = 7)
# Example: Display the first plot
# print(length(plots))
# print(plots[[1]])
# print(plots[[2]])
# print(plots[[3]])
```

```
# Calculate the global maximum change
max_change <- filtered_forest_data |>
  mutate(
    initial_forest = as.numeric(gfw_area__ha),
    final_forest = as.numeric(gfw_area__ha) - as.numeric(loss) + as.numeric(gain),
    change = final_forest - initial_forest
  ) |>
  summarise(max_change = max(abs(change), na.rm = TRUE)) |>
  pull(max_change)

forest_change <- filtered_forest_data |>
  mutate(
    initial_forest = as.numeric(gfw_area__ha),
    final_forest = as.numeric(gfw_area__ha) - as.numeric(loss) + as.numeric(gain),
    change = final_forest - initial_forest,
    color_intensity = abs(change) / max_change,
    line_color = ifelse(change > 0,
                        scales::col_numeric("green", domain = c(0, 1))(color_intensity),
                        scales::col_numeric("red", domain = c(0, 1))(color_intensity))
  ) |>
  select(country, initial_forest, final_forest, change, line_color) |>
  pivot_longer(cols = c(initial_forest, final_forest),
               names_to = "year",
               values_to = "forest_area") |>
  mutate(year = recode(year,
                       initial_forest = "2000",
                       final_forest = "2020"))

# Create a single ggplot for forest change across 10 countries
forest_change_plot <- ggplot(forest_change, aes(x = year, y = forest_area, group = country))
  geom_line(aes(color = line_color), size = 1.2) +
  geom_point(aes(color = line_color), size = 3) +
  scale_color_identity() +
  geom_text(data = forest_change |> filter(year == "2020"),
            aes(label = country, x = year, y = forest_area, color = line_color),
```

```
            hjust = -0.2, size = 3) +
  ggtitle("Forest Area Change (2000 vs 2020) Across Countries") +
  xlab("Year") +
  ylab("Forest Area (ha)") +
  theme_minimal()

# Display the plot
print(forest_change_plot)
```




```
# #visualize geographical distribution with log normalized data
# country_density_map_logscale <- country_density_map|>
#   mutate(
#     log_count = log1p(individualCount)  # log1p handles zero counts
#   )
#
# global_size_range <- range(country_density_map_logscale$log_count, na.rm = TRUE)
#
# # Iterate through each bird order and create maps for each country and year
# bird_orders <- unique(country_density_map_logscale$order)
#
# # Create a list to store plots
# plots1 <- list()
```

```r
# c = 0
# for (order1 in bird_orders) {
#   for (country1 in countries) {
#     for (year1 in years) {
#       c <- c + 1
#       # Filter data for the specific bird order, year, and country
#       specific_data <- country_density_map_logscale |>
#         filter(order == order1, year == year1, country == country1)
#
#       # Join country map with bird observation data
#       country_map <- country_maps |> filter(name_long == country1)
#
#       # Create the plot
#       plots1[[c]] <-
#         ggplot() +
#         geom_sf(data = country_map, fill = "gray90", color = "black") +
#         geom_point(data = specific_data,
#                    aes(x = decimalLongitude, y = decimalLatitude, size = individualCount),
#                    alpha = 0.7) +
#         scale_size_continuous(name = "Count", range = c(1, 10), limits = global_size_range
#         ggtitle(paste("Bird Order:", order1, "Year:", year1, "Country:", country1)) +
#         theme_bw()
#
#       ggsave(paste("plot_geodistribution_logscale/bird_order:", order1, "year:", year1, "co
#
#     }
#   }
# }
```

```r
# plots1[[1]]
# plots1[[2]]
# plots1[[3]]
```

```r
# # Merge aggregated counts back with deforestation data
model_data <- joined_data1 %>%
  select(country, year, iso, stable, loss, gain, disturb, net, change, gfw_area__ha) %>%
  distinct() %>%
  inner_join(order_counts, by = c("country", "year"))

#log normalizes the counts to take into account discrepancies in effort put into observation
model_data_log <- model_data|>
  mutate(
```

```r
    log_count = log1p(order_count)  # log1p handles zero counts
  )
head(model_data_log)
```

```
  country year iso                  stable                 loss
1  poland 2010 POL 9114498.453999999910593 485398.353599999973085
2  poland 2010 POL 9114498.453999999910593 485398.353599999973085
3  poland 2010 POL 9114498.453999999910593 485398.353599999973085
4  poland 2010 POL 9114498.453999999910593 485398.353599999973085
5  poland 2000 POL 9114498.453999999910593 485398.353599999973085
6  poland 2000 POL 9114498.453999999910593 485398.353599999973085
                   gain                disturb                    net  change
1 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
2 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
3 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
4 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
5 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
6 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
  gfw_area__ha         order order_count log_count
1     31240006 Ciconiiformes         244  5.501258
2     31240006 Coraciiformes          29  3.401197
3     31240006   Galliformes         153  5.036953
4     31240006  Strigiformes          22  3.135494
5     31240006 Ciconiiformes         617  6.426488
6     31240006 Coraciiformes          30  3.433987
```

```r
model_data_log_normalized <- model_data_log |>
  mutate(
    scaled_log_count = (log_count - min(log_count, na.rm = TRUE)) /
                       (max(log_count, na.rm = TRUE) - min(log_count, na.rm = TRUE))
  )

head(model_data_log_normalized)
```

```
  country year iso                  stable                 loss
1  poland 2010 POL 9114498.453999999910593 485398.353599999973085
2  poland 2010 POL 9114498.453999999910593 485398.353599999973085
3  poland 2010 POL 9114498.453999999910593 485398.353599999973085
4  poland 2010 POL 9114498.453999999910593 485398.353599999973085
5  poland 2000 POL 9114498.453999999910593 485398.353599999973085
6  poland 2000 POL 9114498.453999999910593 485398.353599999973085
```

```
                    gain                disturb                    net  change
1 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
2 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
3 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
4 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
5 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
6 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
  gfw_area__ha         order order_count log_count scaled_log_count
1     31240006 Ciconiiformes         244  5.501258        0.4992061
2     31240006 Coraciiformes          29  3.401197        0.2811656
3     31240006   Galliformes         153  5.036953        0.4509992
4     31240006  Strigiformes          22  3.135494        0.2535787
5     31240006 Ciconiiformes         617  6.426488        0.5952689
6     31240006 Coraciiformes          30  3.433987        0.2845700
```

```
model_data_log_normalized_bins <- model_data_log_normalized |>
  mutate(
    scaled_log_count_bins = cut(
      scaled_log_count,
      breaks = seq(0, 1, length.out = 11),  # 10 intervals
      labels = paste0("b", 1:10),       # Create labels for the bins
      include.lowest = TRUE
    )
  )
head(model_data_log_normalized_bins)
```

```
  country year iso                 stable                  loss
1  poland 2010 POL 9114498.453999999910593 485398.353599999973085
2  poland 2010 POL 9114498.453999999910593 485398.353599999973085
3  poland 2010 POL 9114498.453999999910593 485398.353599999973085
4  poland 2010 POL 9114498.453999999910593 485398.353599999973085
5  poland 2000 POL 9114498.453999999910593 485398.353599999973085
6  poland 2000 POL 9114498.453999999910593 485398.353599999973085
                    gain                disturb                    net  change
1 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
2 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
3 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
4 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
5 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
6 892077.288900000043213 975192.795700000016950 406678.935300000011921 3.84563
  gfw_area__ha         order order_count log_count scaled_log_count
1     31240006 Ciconiiformes         244  5.501258        0.4992061
```

```
2     31240006 Coraciiformes          29  3.401197         0.2811656
3     31240006   Galliformes         153  5.036953         0.4509992
4     31240006  Strigiformes          22  3.135494         0.2535787
5     31240006 Ciconiiformes         617  6.426488         0.5952689
6     31240006 Coraciiformes          30  3.433987         0.2845700
  scaled_log_count_bins
1                    b5
2                    b3
3                    b5
4                    b3
5                    b6
6                    b3
```

```
model_data_log_normalized_bins|> group_by(country) |>
  summarize(year)
```

```
# A tibble: 107 x 2
# Groups:   country [10]
   country     year
   <chr>      <int>
 1 bangladesh  2020
 2 bangladesh  2020
 3 bangladesh  2020
 4 bangladesh  2020
 5 bangladesh  2010
 6 bangladesh  2010
 7 bangladesh  2010
 8 bangladesh  2010
 9 bangladesh  2000
10 bangladesh  2000
# i 97 more rows
```

```
set.seed(12)  # For reproducibility
library(caret)

# Perform an initial split
split_index_bins <- initial_split(model_data_log_normalized_bins, prop = 0.8)

# Extract training and test sets
train_data_bins <- training(split_index_bins)
test_data_bins <- testing(split_index_bins)
```

```r
set.seed(23)
# Perform an initial split
split_index <- initial_split(model_data_log_normalized, prop = 0.8)

# Extract training and test sets
train_data <- training(split_index)
test_data <- testing(split_index)
```

```r
library(e1071)

svm_recipe <-
  recipe(scaled_log_count_bins ~ stable + loss + gain + net + change + gfw_area__ha + order +
         data = model_data_log_normalized_bins) |>
  step_mutate(country = as.factor(country)) |>
  step_dummy(country, one_hot = TRUE) |>
  step_mutate(order = as.factor(order)) |>
  step_dummy(order, one_hot = TRUE) |>
  step_mutate(year = factor(year, levels = c(2000, 2010, 2020)))


svm_pol <- svm_poly(cost = tune(), degree= tune()) |>
  set_engine("kernlab") |>
  set_mode("classification")

svm_pol_wflow <- workflow() |>
  add_model(svm_pol) |>
  add_recipe(svm_recipe)

folds_pol <- vfold_cv(train_data_bins, v = 4)

# the tuned parameters also have default values you can use
grid_pol <- grid_regular(cost(), degree(c(1,5)), levels = 5)

svm_pol_tune <-
  svm_pol_wflow |>
  tune_grid(resamples = folds_pol, grid = grid_pol)


svm_metrics_pol <- collect_metrics(svm_pol_tune)
accuracy_results_pol <- svm_metrics_pol |>
  filter(.metric == "accuracy")
```

```r
print(accuracy_results_pol)
```

```
# A tibble: 25 x 8
       cost degree .metric  .estimator  mean     n std_err .config
      <dbl>  <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
 1  0.000977      1 accuracy multiclass 0.202     3  0.0543 Preprocessor1_Model~
 2  0.0131        1 accuracy multiclass 0.202     3  0.0543 Preprocessor1_Model~
 3  0.177         1 accuracy multiclass 0.202     3  0.0543 Preprocessor1_Model~
 4  2.38          1 accuracy multiclass 0.220     3  0.0588 Preprocessor1_Model~
 5 32             1 accuracy multiclass 0.220     3  0.0588 Preprocessor1_Model~
 6  0.000977      2 accuracy multiclass 0.233     3  0.0507 Preprocessor1_Model~
 7  0.0131        2 accuracy multiclass 0.157     3  0.0171 Preprocessor1_Model~
 8  0.177         2 accuracy multiclass 0.204     3  0.0437 Preprocessor1_Model~
 9  2.38          2 accuracy multiclass 0.204     3  0.0437 Preprocessor1_Model~
10 32             2 accuracy multiclass 0.204     3  0.0437 Preprocessor1_Model~
# i 15 more rows
```

```r
library(tidyr)
library(ggplot2)
#
# # Step 3: Create a Confusion Matrix with Explicit Levels
# confusion_data <- svm_predictions |>
#   conf_mat(truth = scaled_log_count_bins, estimate = .pred_class)
#
# # Extract the confusion matrix counts as a data frame
# confusion_data_df <- as_tibble(confusion_data$table) |>
#   complete(Truth = paste0("b", 1:10), Prediction = paste0("b", 1:10), fill = list(n = 0))
#
# # Step 4: Plot the Heatmap
# heatmap_plot <- ggplot(confusion_data_df, aes(x = Prediction, y = Truth, fill = n)) +
#   geom_tile() +
#   scale_fill_gradient(low = "white", high = "blue") +
#   labs(
#     title = "Confusion Matrix Heatmap",
#     x = "Predicted Bin",
#     y = "Actual Bin",
#     fill = "Count"
#   ) +
#   theme_minimal()
#
# print(heatmap_plot)
```

```
# # Check prediction distribution
# svm_predictions |>
#   count(.pred_class) |>
#   arrange(desc(n)) |>
#   mutate(proportion = n / sum(n))
#
# # Compare predicted vs actual class distribution
# predicted_distribution <- svm_predictions |>
#   count(.pred_class) |>
#   rename(Predicted = n)
#
# actual_distribution <- test_data_bins |>
#   count(scaled_log_count_bins) |>
#   rename(Actual = n)
#
# comparison <- full_join(predicted_distribution, actual_distribution,
#                          by = c(".pred_class" = "scaled_log_count_bins")) |>
#   replace_na(list(Predicted = 0, Actual = 0))
#
# print(comparison)
```

```
#SVM RBF

svm_rbf <- svm_rbf(cost = tune(), rbf_sigma= tune()) |>
  set_engine("kernlab") |>
  set_mode("classification")

svm_rbf_wflow <- workflow() |>
  add_model(svm_rbf) |>
  add_recipe(svm_recipe)

folds_rbf <- vfold_cv(train_data_bins, v = 4)

# the tuned parameters also have default values you can use
grid_rbf <- grid_regular(cost(), rbf_sigma(), levels = 5)

svm_rbf_tune <-
  svm_rbf_wflow |>
  tune_grid(resamples = folds_rbf, grid = grid_rbf)

svm_metrics_rbf <- collect_metrics(svm_rbf_tune)
accuracy_results_rbf <- svm_metrics_rbf |>
```

```r
  filter(.metric == "accuracy")
print(accuracy_results_rbf)
```

```
# A tibble: 25 x 8
       cost    rbf_sigma .metric  .estimator  mean     n std_err .config
      <dbl>        <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>
 1  0.000977 0.0000000001 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 2  0.0131   0.0000000001 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 3  0.177    0.0000000001 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 4  2.38     0.0000000001 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 5 32        0.0000000001 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 6  0.000977 0.0000000316 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 7  0.0131   0.0000000316 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 8  0.177    0.0000000316 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
 9  2.38     0.0000000316 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
10 32        0.0000000316 accuracy multiclass 0.222     3  0.0972 Preprocessor1~
# i 15 more rows
```

```r
# # Step 3: Create a Confusion Matrix with Explicit Levels
# confusion_data <- svm_predictions |>
#   conf_mat(truth = scaled_log_count_bins, estimate = .pred_class)
#
# # Extract the confusion matrix counts as a data frame
# confusion_data_df <- as_tibble(confusion_data$table) |>
#   complete(Truth = paste0("b", 1:10), Prediction = paste0("b", 1:10), fill = list(n = 0))
#
# # Step 4: Plot the Heatmap
# heatmap_plot <- ggplot(confusion_data_df, aes(x = Prediction, y = Truth, fill = n)) +
#   geom_tile() +
#   scale_fill_gradient(low = "white", high = "blue") +
#   labs(
#     title = "Confusion Matrix Heatmap",
#     x = "Predicted Bin",
#     y = "Actual Bin",
#     fill = "Count"
#   ) +
#   theme_minimal()
#
# print(heatmap_plot)
```

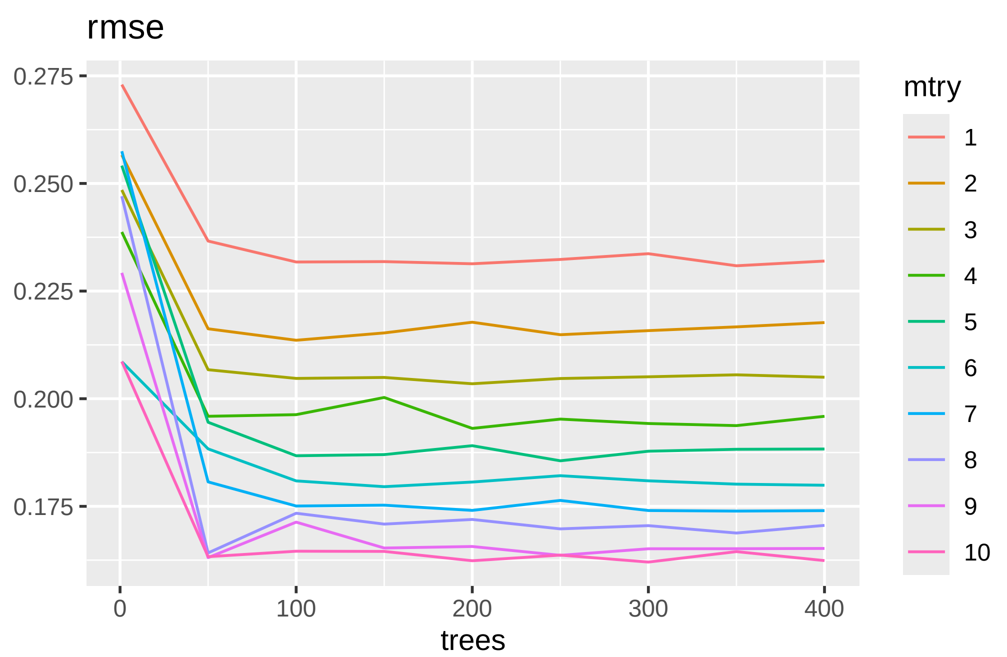
```r
model_data_log_normalized <- model_data_log_normalized |>
  mutate(
    country = as.factor(country),
    order = as.factor(order),
    year = factor(year, levels = c(2000, 2010, 2020))
  )

# Step 2: Define Recipe
rf_recipe <- recipe(scaled_log_count ~ stable + loss + gain + net + change + gfw_area__ha + c
                    data = train_data) |>
  step_dummy(all_nominal(), -all_outcomes())  # Dummy-encode categorical variables

# Step 3: Define Model and Workflow
rf_model <- rand_forest(mtry = tune(), trees = tune()) |>  # Define Random Forest model
  set_engine("ranger", importance = "permutation") |>  # Set engine with permutation importar
  set_mode("regression")  # Set mode to regression

rf_wflow <- workflow() |>
  add_model(rf_model) |>
  add_recipe(rf_recipe)
```

```r
rf_folds <- vfold_cv(train_data,
                          v = 5)
rf_grid <- expand_grid(
  mtry = seq(1, 10, by = 1),  # You can adjust this range as needed
  trees = c(1, 50, 100, 150, 200, 250, 300, 350, 400)
)
rf_tune <-
  rf_wflow |>
  tune_grid(resamples = rf_folds,
            grid = rf_grid)

tuning_results <- rf_tune |> collect_metrics()
tuning_results
```

```
# A tibble: 180 x 8
    mtry trees .metric .estimator  mean     n std_err .config
   <dbl> <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
 1     1     1 rmse    standard   0.273     5  0.0212 Preprocessor1_Model01
 2     1     1 rsq     standard   0.130     5  0.0682 Preprocessor1_Model01
 3     1    50 rmse    standard   0.237     5  0.0140 Preprocessor1_Model02
```

```
 4      1     50 rsq     standard   0.230      5  0.0962 Preprocessor1_Model02
 5      1    100 rmse    standard   0.232      5  0.0119 Preprocessor1_Model03
 6      1    100 rsq     standard   0.289      5  0.111  Preprocessor1_Model03
 7      1    150 rmse    standard   0.232      5  0.0139 Preprocessor1_Model04
 8      1    150 rsq     standard   0.272      5  0.0908 Preprocessor1_Model04
 9      1    200 rmse    standard   0.231      5  0.0138 Preprocessor1_Model05
10      1    200 rsq     standard   0.288      5  0.107  Preprocessor1_Model05
# i 170 more rows
```

```r
tuning_results <- rf_tune |> collect_metrics()

tuning_results|>
  filter(.metric == "rmse") |>
  mutate(mtry = as.factor(mtry)) |>
  ggplot() +
  geom_line(aes(x = trees, y = mean, color = mtry)) +
  labs(title = "rmse", y = "")
```




```r
# Step 4: Select Best Parameters
best_params <- rf_tune |> select_best()

# Step 5: Finalize Workflow
```

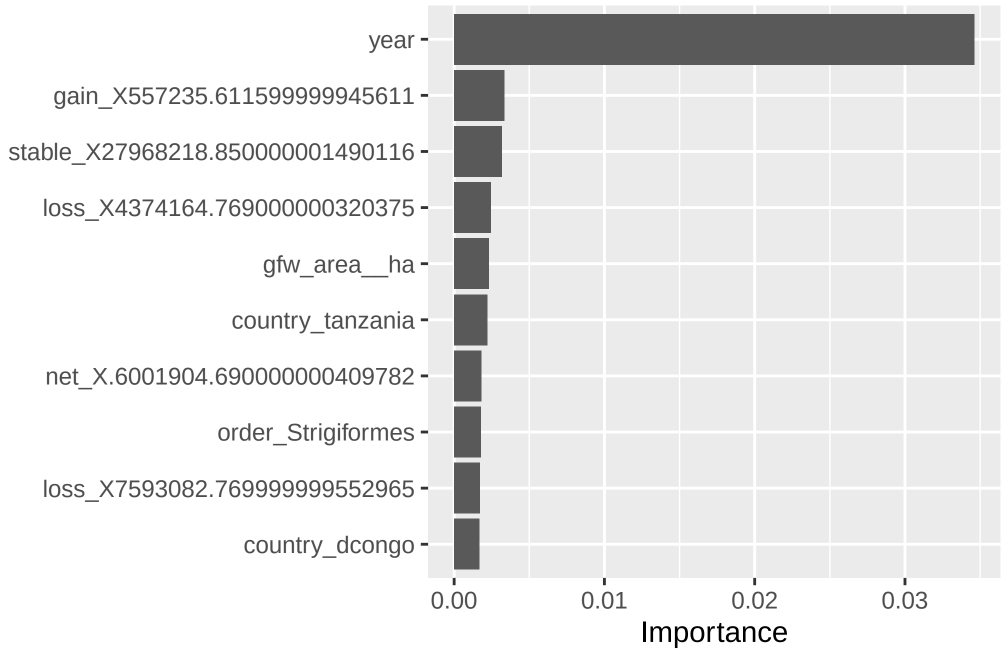
```r
final_rf_wflow <- rf_wflow |>
  finalize_workflow(best_params)

# Step 6: Fit Final Model
final_rf_fit <- final_rf_wflow |>
  fit(data = train_data)

# Step 7: Predict on Test Data
rf_preds <- predict(final_rf_fit, new_data = test_data) |>
  bind_cols(test_data)

# Step 8: Evaluate Model Performance (MSE)
rf_mse <- rf_preds |>
  metrics(truth = scaled_log_count, estimate = .pred) |>
  filter(.metric == "rmse") |>
  mutate(mse = .estimate^2) |>
  pull(mse)

rf_mse
```

```
[1] 0.03565733
```

```r
best_params
```

```
# A tibble: 1 x 3
   mtry trees .config
  <dbl> <dbl> <chr>
1    10   300 Preprocessor1_Model88
```

```r
library(vip)
vip(final_rf_fit)
```



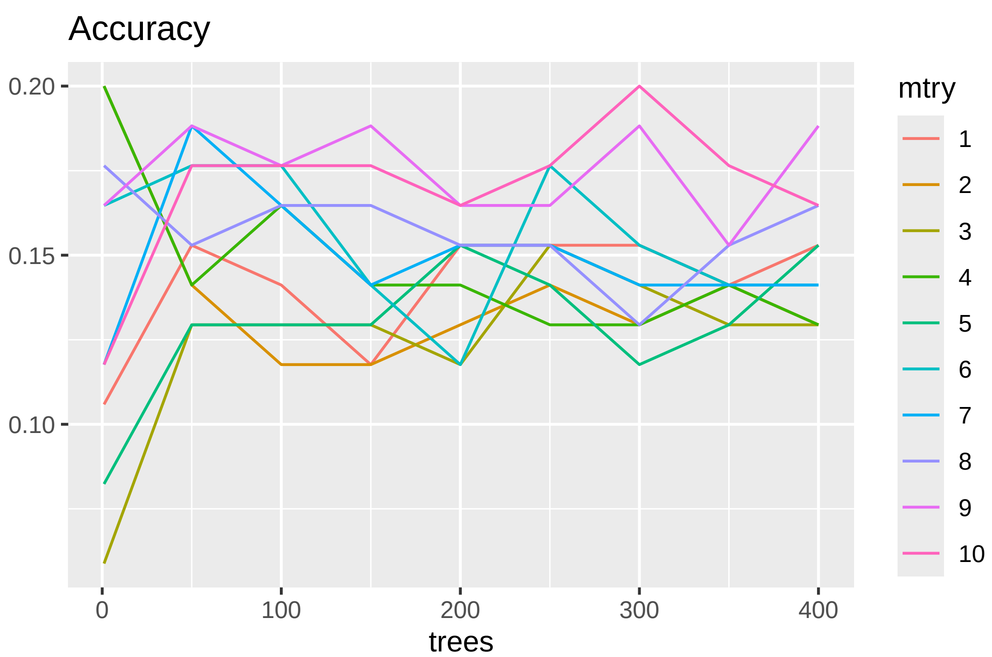
I realize that the year 2020 is very predictive, and the country tanzania very predictive, as both variables have distinctive natures appearantly. The variable that exists for all entries that is the most important is "net". Net change in forest size is somewhat indicative of predicting count.

```
# Split Data
set.seed(123)  # Ensure reproducibility

# Step 2: Define Recipe
rf_recipe1 <- recipe(scaled_log_count_bins ~ stable + loss + gain + net + change + gfw_area_
                    data = train_data_bins) |>
  step_dummy(all_nominal(), -all_outcomes())  # Dummy-encode categorical variables

# Step 3: Define Model and Workflow
rf_model1 <- rand_forest(mtry = tune(), trees = tune()) |>  # Define Random Forest model
  set_engine("ranger", importance = "permutation") |>  # Set engine with permutation importa
  set_mode("classification")  # Set mode to regression

rf_wflow1 <- workflow() |>
  add_model(rf_model1) |>
  add_recipe(rf_recipe1)
```

```r
rf_folds1 <- vfold_cv(train_data_bins,
                          v = 5)
rf_grid1 <- expand_grid(
  mtry = seq(1, 10, by = 1),  # You can adjust this range as needed
  trees = c(1, 50, 100, 150, 200, 250, 300, 350, 400)
)
rf_tune1 <-
  rf_wflow1 |>
  tune_grid(resamples = rf_folds1,
            grid = rf_grid1)

rf_tune1 |> collect_metrics()
```

```
# A tibble: 270 x 8
    mtry trees .metric     .estimator  mean     n std_err .config
   <dbl> <dbl> <chr>       <chr>      <dbl> <int>   <dbl> <chr>
 1     1     1 accuracy    multiclass 0.106     5  0.0432 Preprocessor1_Model01
 2     1     1 brier_class multiclass 0.553     5  0.0307 Preprocessor1_Model01
 3     1     1 roc_auc     hand_till  0.534     5  0.0238 Preprocessor1_Model01
 4     1    50 accuracy    multiclass 0.153     5  0.0660 Preprocessor1_Model02
 5     1    50 brier_class multiclass 0.459     5  0.0108 Preprocessor1_Model02
 6     1    50 roc_auc     hand_till  0.587     5  0.0361 Preprocessor1_Model02
 7     1   100 accuracy    multiclass 0.141     5  0.0440 Preprocessor1_Model03
 8     1   100 brier_class multiclass 0.456     5  0.0109 Preprocessor1_Model03
 9     1   100 roc_auc     hand_till  0.600     5  0.0448 Preprocessor1_Model03
10     1   150 accuracy    multiclass 0.118     5  0.0456 Preprocessor1_Model04
# i 260 more rows
```

```r
tuning_results1 <- rf_tune1 |> collect_metrics()

tuning_results1|>
  filter(.metric == "accuracy") |>
  mutate(mtry = as.factor(mtry)) |>
  ggplot() +
  geom_line(aes(x = trees, y = mean, color = mtry)) +
  labs(title = "Accuracy", y = "")
```

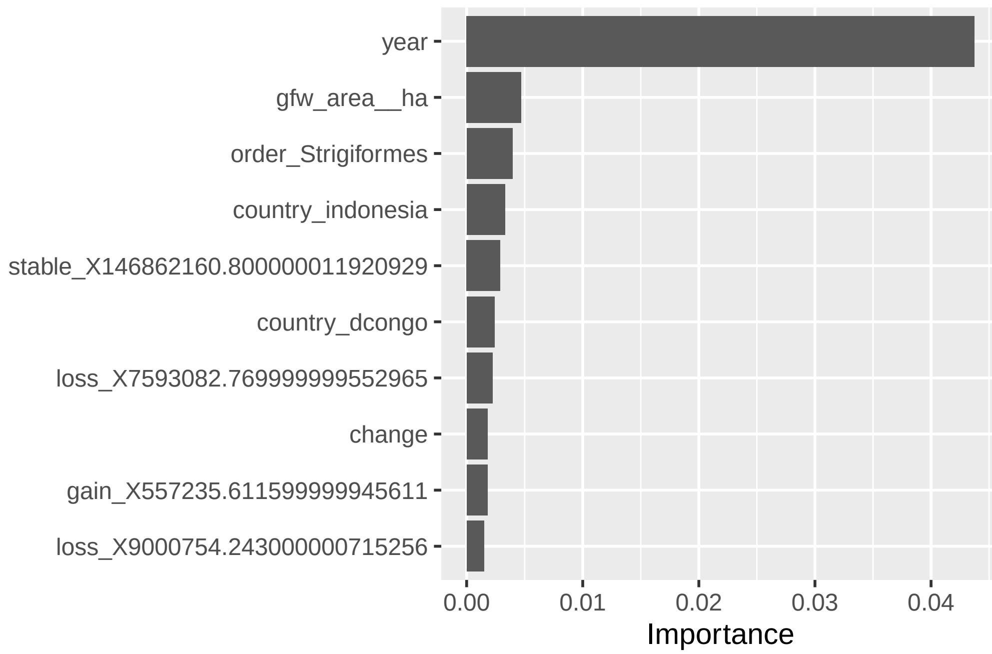



```
final_rf_wflow1 <- rf_wflow1 |>
  finalize_workflow(best_params)

# Step 6: Fit Final Model
final_rf_fit1 <- final_rf_wflow1 |>
  fit(data = train_data_bins)

# Step 5: Predict on Train Data
rf_preds1 <- predict(final_rf_fit1, new_data = train_data_bins) |>
  bind_cols(train_data_bins)

# Step 6: Evaluate Model Performance (MSE)
rf_metrics1 <- rf_preds1 |>
  metrics(truth = scaled_log_count_bins, estimate = .pred_class)

# Extract MSE
rf_mse1 <- rf_metrics1 |>
  filter(.metric == "rmse") |>  # You can replace this with "mse" if you need it
  pull(.estimate)

rf_mse1
```

```
numeric(0)
```

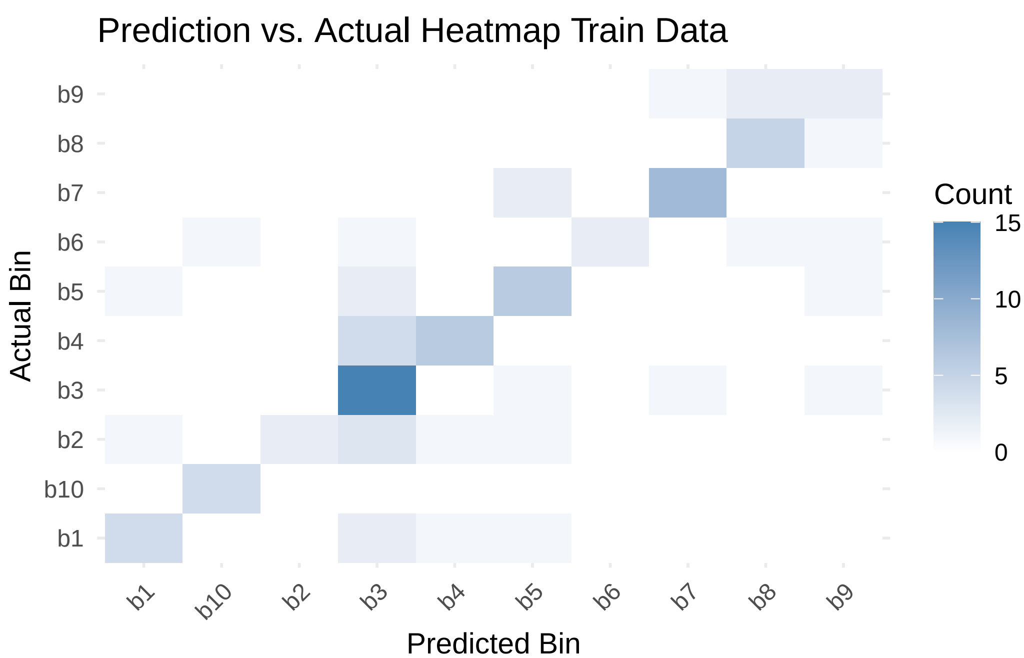
```
# Step 5: Predict on Test Data
rf_preds2 <- predict(final_rf_fit1, new_data = test_data_bins) |>
  bind_cols(test_data_bins)

# Step 6: Evaluate Model Performance (MSE)
rf_metrics2 <- rf_preds2 |>
  metrics(truth = scaled_log_count_bins, estimate = .pred_class)

# Extract MSE
rf_mse2 <- rf_metrics2 |>
  filter(.metric == "rmse") |>  # You can replace this with "mse" if you need it
  pull(.estimate)
```

```
library(vip)
vip(final_rf_fit1)
```



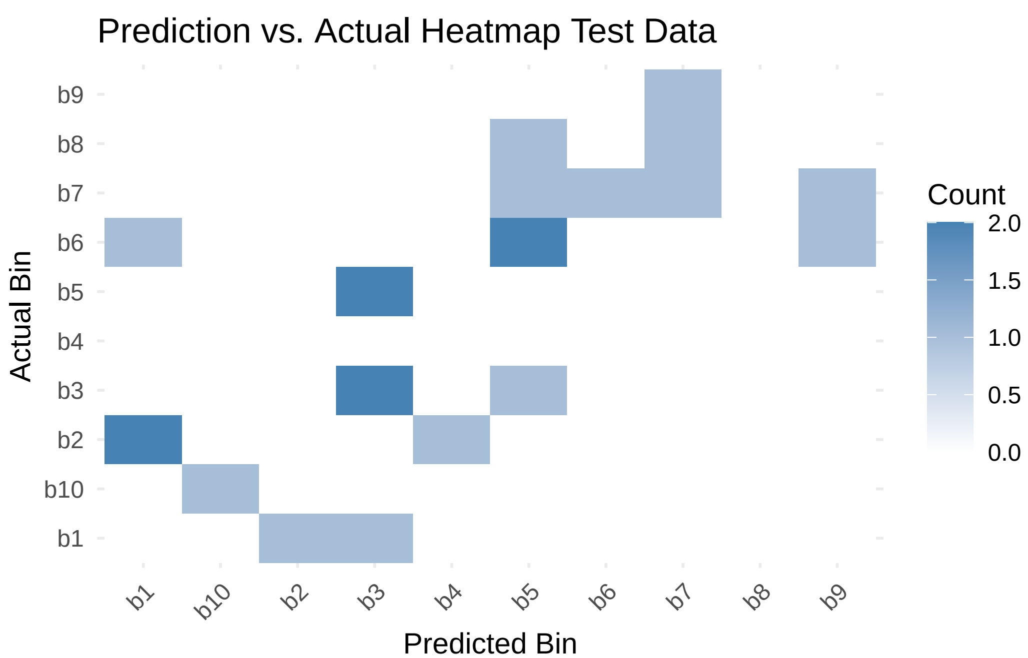

```
library(ggplot2)
library(tidyr)

# Step 6: Generate Confusion Matrix Data
conf_matrix <- rf_preds1 |>
  conf_mat(truth = scaled_log_count_bins, estimate = .pred_class)
```

```r
# Convert Confusion Matrix to Tibble for Plotting
conf_matrix_tibble <- as_tibble(conf_matrix$table) |>
  complete(Truth = paste0("b", 1:10), Prediction = paste0("b", 1:10), fill = list(n = 0))

# Step 7: Heatmap of Predictions vs. Actuals
ggplot(conf_matrix_tibble, aes(x = Prediction, y = Truth, fill = n)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(
    title = "Prediction vs. Actual Heatmap Train Data",
    x = "Predicted Bin",
    y = "Actual Bin",
    fill = "Count"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```r
# Step 6: Generate Confusion Matrix Data
conf_matrix <- rf_preds2 |>
  conf_mat(truth = scaled_log_count_bins, estimate = .pred_class)

# Convert Confusion Matrix to Tibble for Plotting
```

```
conf_matrix_tibble <- as_tibble(conf_matrix$table) |>
  complete(Truth = paste0("b", 1:10), Prediction = paste0("b", 1:10), fill = list(n = 0))

# Step 7: Heatmap of Predictions vs. Actuals
ggplot(conf_matrix_tibble, aes(x = Prediction, y = Truth, fill = n)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(
    title = "Prediction vs. Actual Heatmap Test Data",
    x = "Predicted Bin",
    y = "Actual Bin",
    fill = "Count"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# KNN model with tuning
knn_recipe <- recipe(scaled_log_count_bins ~ stable + loss + gain + net + change + gfw_area_
                     data = train_data_bins)

knn_model <- nearest_neighbor() |>
  set_engine("kknn") |>
```

```
  set_mode("classification")


knn_wflow <- workflow() |>
  add_model(knn_model) |>
  add_recipe(knn_recipe)

fit_knn <- knn_wflow |> fit(data = train_data_bins)

set.seed(470)
knn_vfold <- vfold_cv(train_data_bins,
                          v = 5, strata = scaled_log_count_bins)
k_grid <- data.frame(neighbors = c(1,3,5))

knn_tune <- nearest_neighbor(neighbors = tune()) |>
  set_engine("kknn") |>
  set_mode("classification")

knn_wflow_tune <- workflow() |>
  add_model(knn_model) |>
  add_recipe(knn_recipe)
```

```
knn_wflow_tune |>
  tune_grid(resamples = knn_vfold,
            grid = k_grid) |>
  collect_metrics() |>
  filter(.metric == "accuracy")
```

```
# A tibble: 1 x 6
  .metric  .estimator  mean     n std_err .config
  <chr>    <chr>      <dbl> <int>   <dbl> <chr>
1 accuracy multiclass 0.175     5  0.0335 Preprocessor1_Model1
```