ARTIFICIAL INTELLIGENCE PROJECT

1. Tracking a changing climate

The climate is changing around the world. The impacts of climate change are felt in many different areas, but they are particularly noticeable in their effects on birds. Many bird species are moving north, if they can, to stay in climatic conditions that are suitable for them.

Our analysis will use data from the UK Met Office together with records from the Global Biodiversity Information Facility to build our very own species distribution model using machine learning. This model will be able to predict where our bird species of interest is likely to occur in the future - information that is invaluable to conservation organization working on the ground to preserve these species and save them from extinction.

We will start by importing the climate data from a local `rds` file.

```
Start coding or generate with AI.
```

```
# Importing tidyverse, raster, and sf packages
library(tidyverse)
library(sf)
library(raster)

# Importing the climate data from an rds file
climate  <- read_rds('datasets/climate_raster.rds')

# Have a look at the variables in the climate data
colnames(climate)

# Converting the dataframe to SpatialPixelDataFrame for plotting
climate_df <- mutate(
  .data = climate,
  rasters = map(
    .x = rasters,
    ~ as_tibble(as(.x, "SpatialPixelsDataFrame")))) %>%
  unnest(cols = c(rasters))
```

```
-- Attaching packages --------------------------------------- tidyverse 1.2.1 --
v ggplot2 3.2.1     v purrr   0.3.3
v tibble  2.1.3     v dplyr   0.8.3
v tidyr   1.0.0     v stringr 1.4.0
v readr   1.1.1     v forcats 0.3.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
Linking to GEOS 3.5.1, GDAL 2.2.2, PROJ 4.9.2
Loading required package: sp

Attaching package: 'raster'

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

    select

The following object is masked from 'package:tidyr':

    extract
```

1. 'decade'
2. 'rasters'

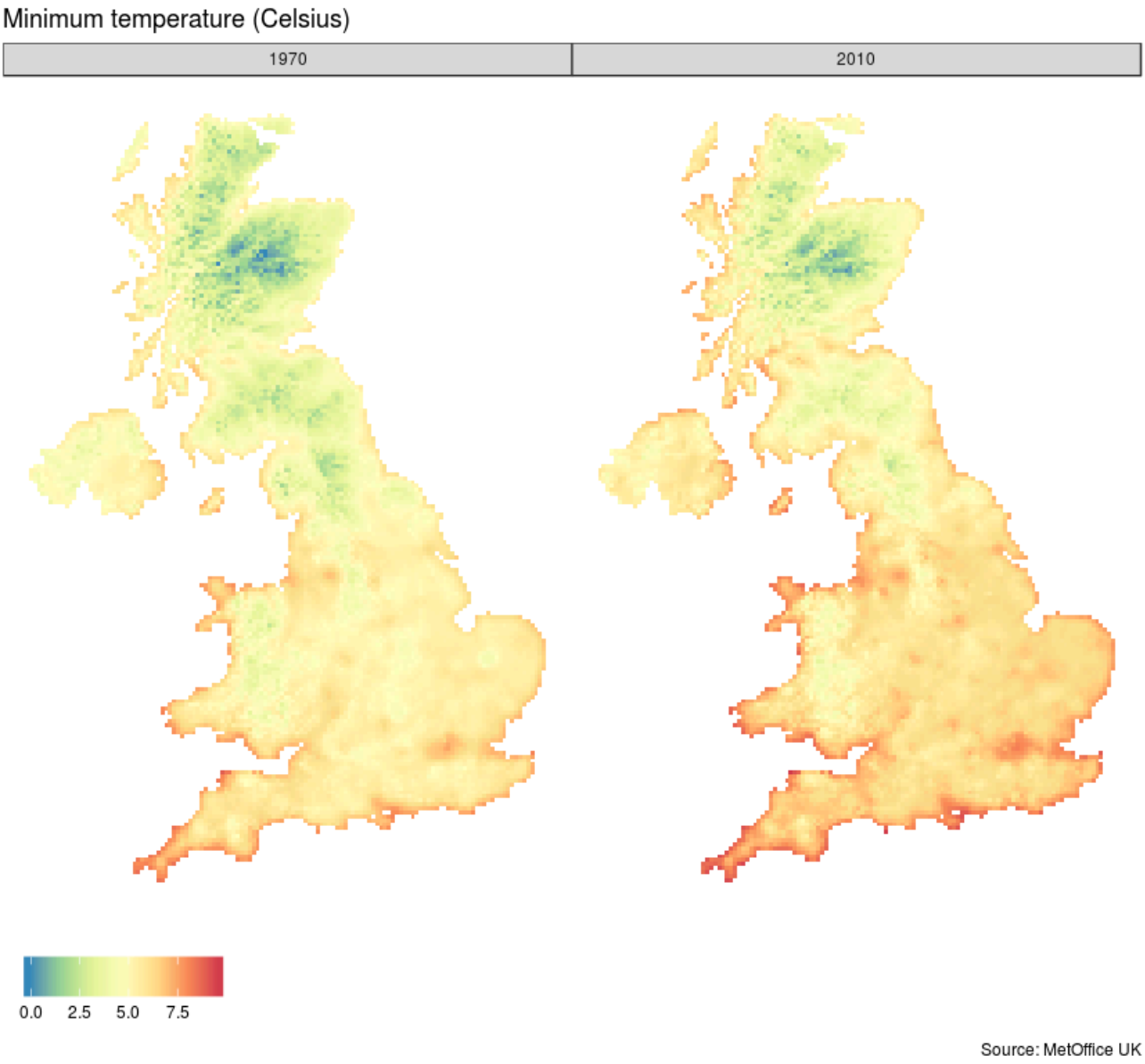
2. Mapping a changing climate

We have loaded the pre-processed climate data and converted it to a `SpatialPixelDataFrame`. This data frame now contains all the information we need:

- the `decade` of observation,
- spatial coordinates (`x`, `y`)
- six selected climatic variables (`minimum.temperature`, `maximum.temperature`, `rainfall`, `wind.speed`, `snow.lying`, `air.frost`)

The first step in any analysis is visualizing the data. Visualizing the data makes sure the data import worked, and it helps us develop intuition about the patterns in our dataset. Here we are dealing with spatial datas. We will start with two maps: one map of the climatic conditions in 1970, and one map of the climatic conditions in 2010. Our climate data has several variables, so let us pick `minimum.temperature` for now.

```
library(ggthemes)

# Creating the plot
ggp_temperature <- climate_df %>%
  filter(decade == 1970 | decade == 2010)%>%
  ggplot(aes(x = x, y = y)) + geom_tile(aes(fill = minimum.temperature)) +
  theme_map()+
  coord_equal()+
  facet_grid(~ decade) + scale_fill_distiller(palette = "Spectral") +
  theme(legend.title = element_blank(), legend.position = "bottom") +
  labs(title = "Minimum temperature (Celsius)", caption = 'Source: MetOffice UK')

ggp_temperature
```


Minimum temperature (Celsius)
1970
2010
0.0
2.5
5.0
7.5
Source: MetOffice UK

3. Fieldwork in the digital age – download the data

Now we need to obtain species occurrence records. This used to be the main challenge in biogeography. Natural historians, such as Charles Darwin and Alexander von Humboldt, traveled around the globe for years on rustic sail ships collecting animal and plant specimens to understand the natural world.

```
library(rgbif)
source("datasets/occ_search.R")

# Calling the API to get the occurrence records of Scottish crossbill
gbif_response <- occ_search(
  scientificName = "Loxia scotica", country = "GB",
  hasCoordinate = TRUE, hasGeospatialIssue = FALSE, limit = 2000)

# Inspecting the class and names of gbif_response
class(gbif_response)
names(gbif_response)

head(data,n = 6)
```

'gbif'

1. 'meta'
2. 'hierarchy'
3. 'data'
4. 'media'
5. 'facets'

```
1 function (..., list = character(), package = NULL, lib.loc = NULL,
2     verbose = getOption("verbose"), envir = .GlobalEnv)
3 {
4     fileExt <- function(x) {
5         db <- grepl("\\\\.[^.]+\\\\.(gz|bz2|xz)$", x)
6         ans <- sub(".*\\\\.", "", x)
```

4. Sorting out the bad eggs – data cleaning

In particular, data collected at this large scale can have issues. Luckily, GBIF provides some useful metadata on each record.

Here are some criteria we can use:

1. "issues" - We will only use records where no doubts about the observation were listed.
2. "license" - We will only use records under a creative commons license.
3. "date" - We will only use records between 1965 and 2015 because that matches our climate dataset.

```
library(lubridate)

birds_dated <- mutate(
  .data = gbif_response$data,
# Creating a new column which specifies the decade of observation
  decade = ymd_hms(eventDate) %>% round_date("10y") %>% year())

birds_cleaned <- birds_dated %>%
  filter(
    issues == "" &
    str_detect(license, "http://creativecommons.org/") &
    decade >= 1970, decade <= 2010
  ) %>%
  transmute(decade = decade, x = decimalLongitude, y = decimalLatitude) %>%
  arrange(decade)
```

```
Attaching package: 'lubridate'

The following object is masked from 'package:base':

    date
```

5. Nesting the data

We have cleaned the data, but there is a problem. We want to know the climatic conditions at the location of the bird observation **at the time** of the observation. This is tricky because we have climate data from multiple decades. How do we match each bird observation to the correct climate raster cell?

We can `nest()` data in a list column. The result is a data frame where the grouping columns do not change, and a list column of aggregated data from each group is added. List columns can hold many different kinds of variables such as vectors, data frames, and even objects. For example, the climate data that we imported earlier was already nested by decade and had a list column (`rasters`) that contained a `rasterStack` object for each decade.

```
# "Nesting" the bird data
birds_nested <- birds_cleaned %>%
    group_by(decade) %>%
    nest(.key = "presences")
```

```
head(birds_nested)
```

```
# Calculating the total number of records per decade
birds_counted <- birds_nested %>%
  mutate(n = map_dbl(presences,nrow))
```

```
head(birds_counted)
```

```
Warning message:
"`.key` is deprecated"
```

decade	presences
1970	-3.7742, -3.7996, -3.7424, -3.7742, -3.7536, 57.1484, 57.1748, 57.1368, 57.1484, 57.2490
1980	-3.805402, -3.604524, -4.323717, -3.712700, 57.164144, 57.327101, 56.670186, 57.163600
1990	-3.957270, -3.626685, -3.702624, -3.957270, -3.957270, -3.604524, -3.680281, -3.175040, -3.604524, 54.930540, 57.227079, 57.241025, 54.930540, 54.930540, 57.327101, 57.236949, 57.043710, 57.327101
2000	-2.866320, -4.081160, -4.269620, -4.101760, -4.269620, -4.061040, -4.106990, -4.106990, -4.106990, -4.106990, -4.904820, -3.594050, -4.579720, -3.577180, -3.567030, -3.382740, -2.909360, -3.466780, 1.619770, -3.740130, -4.443510, -4.443510, -4.136400, -3.900980, -4.443510, -3.880710, -4.431430, -4.355060, -4.269620, -4.112250, -4.101760, -4.269620, -3.740130, -4.269620, -3.740130, -4.269620, -4.269620, -3.740130, -4.443510, -4.112250, -4.269620, -4.101760, -3.740130, -4.101760, -4.269620, -4.269620, -4.112250, -4.112250, -4.269620, -4.101760, -4.269620, -4.353930, -4.269620, -4.112250, -4.269620, -4.269620, -4.389310, -3.880710, -3.704550, -3.737600, -3.737600, -4.914770, -4.269620, -4.285090, -5.002750, -3.880710, -3.880710, -3.880710, -3.880710, -3.737600, -4.897130, -4.353930, -2.875080, -3.498970, -2.874660, -3.502590, -3.335770, -3.140920, -2.842130, -3.302210, -4.611970, -4.355060, -4.413570, -4.413570, -4.247380, -4.413570, -4.567190, -3.740130, -4.080580, -3.740130, -4.443510, -4.443510, -3.740130, -3.574630, -3.574630, -4.179830, -3.574630, -4.357480, -4.104890, -3.880710, -4.709970, -4.709970, -3.574630, -3.674750, -3.880710, -3.574630, -4.515830, -4.561540, -3.880710, -3.240560, -4.399130, -3.880710, -3.880710, -3.284810, -2.942360, -2.516040, -2.481440, -3.318130, -2.776590, -3.335130, -3.317490, -3.019240, -3.494250, -3.105760, -3.460720, -3.499690, -3.740130, -3.919610, -4.076090, -2.736160, -3.246610, -4.076090, -4.258440, -4.081160, -3.740130, -3.740130, -3.740130, -4.745840, -4.247380, -3.740130, -4.081160, -3.740130, -3.740130, -3.574630, -3.574630, -3.574630, -4.081160, -3.198920, -3.232400, -3.558800, -4.081160, -2.743260, -4.431430, -3.072810, -4.076090, -2.577070, -2.582690, -3.078020, -4.413570, -2.910240, -2.414660, -4.759340, -3.574630, -3.626685, -3.574630, -3.701856, -3.574630, -3.574630, -3.752920, -3.574630, -3.740130, -3.900980, -3.740130, -3.574630, -3.896400, -3.574630, -3.574630, -3.582330, -3.574630, -3.574630, -4.209700, -3.574630, -4.225660, -4.367460, -4.086270, -3.422970, -2.921430, -3.255810, -3.252720, -3.419480, -3.735910, -4.611720, -3.281160, -4.051160, -4.618230, -3.783680, -4.264010, -4.443510, -4.315730, -3.740130, -3.630480, -4.474360, -4.106990, -3.626330, -4.548660, -4.101760, -4.480640, -4.883790, -4.315730, -4.269620, -4.269620, -4.367460, -2.914670, -4.280940, -3.626330, -2.748690, -2.578460, -4.624790, -3.953280, -2.412660, -4.275270, -4.766150, -4.611720, -4.081160, -3.896400, -2.752360, -4.586050, -2.908040, -4.292400, -4.455740, -3.900980, -4.766150, -4.786820, -4.258440, -4.598810, -4.611720, -3.416000, -3.426490, -3.735910, -4.264010, -3.594050, -3.973160, -3.744370, -3.574630, -3.255810, -4.419490, -3.586349, -3.752920, -3.744370, -3.578470, -4.745840, -3.582330, -3.740130, -3.574630, -3.574630, -3.910240, -3.590120, -4.112250, -3.910240, -3.249660, -4.275270, -3.402310, -3.574630, -3.574630, -2.923700, -2.745060, -4.904820, -2.579860, -3.078020, -2.919160, -2.910240, -2.581270, -3.943520, -3.088630, -2.916910, -3.586210, -3.702393, -4.904820, -3.914910, -4.081160, -3.744370, -4.081160, -4.076090, -4.258440, -3.740130, -3.740130, -3.896400, -3.900980, -4.247380, -4.247380, -4.592410, -3.574630, -4.086270, -3.744370, -3.578470, -3.735910, -3.574630, -3.085950, -3.919610, -3.541860, -3.419480, -3.735910, -4.081160, -3.541860, -3.085950, -3.574630, -3.919610, -3.409120, -3.900980, -3.574630, -3.740130, -4.101760, -3.574630, -3.948380, -3.249660, -4.579720, -3.626685, -3.900980, -3.412550, -3.409120, -4.739150, -3.761580, -3.740130, -3.574630, -4.911920, -3.757240, -3.574630, -3.574630, -3.574630, -2.750520, -3.757240, -3.088630, -4.586050, -3.567030, -3.567030, -3.757240, -3.567030, -3.567030, -3.910240, -3.761580, -4.443510, -3.563260, -4.598810, -4.247380, -4.431430, -4.081160, -3.735910, -4.455740, -3.761580, -4.076090, -3.567030, -3.567030, -3.575800, -3.567030, -3.575800, -3.567030, -3.567030, -3.582080, -3.567030, -3.567030, -3.072810, 57.020620, 57.388200, 57.744350, 57.747250, 57.744350, 57.029110, 57.837010, 57.837010, 57.837010, 57.837010, 57.282000, 57.664820, 57.378960, 57.023120, 57.036270, 57.397800, 57.096680, 57.001550, 52.281590, 57.213680, 57.830970, 57.830970, 57.764640, 57.121500, 57.830970, 58.020250, 57.651490, 56.987980, 57.744350, 57.926770, 57.747250, 57.744350, 57.213680, 57.744350, 57.213680, 57.744350, 57.744350, 57.213680, 57.830970, 57.926770, 57.744350, 57.747250, 57.213680, 57.747250, 57.744350, 57.744350, 57.926770,

57.926770, 57.744350, 57.747250, 57.744350, 56.970030, 57.744350, 57.926770, 57.744350, 57.744350, 58.029680, 58.020250, 57.160260, 57.159810, 57.159810, 57.407600, 57.744350, 57.456540, 57.261730, 58.020250, 58.020250, 58.020250, 58.020250, 57.159810, 57.605730, 56.970030, 57.043020, 56.983200, 57.025050, 57.073000, 57.020980, 57.112750, 57.043250, 57.003370, 56.860210, 56.987980, 57.382260, 57.382260, 57.385340, 57.382260, 57.199490, 57.213680, 57.963230, 57.213680, 57.830970, 57.830970, 57.213680, 57.215860, 57.215860, 57.368550, 57.215860, 57.545070, 57.801110, 58.020250, 57.340280, 57.340280, 57.215860, 57.232540, 58.020250, 57.215860, 56.930930, 57.595010, 58.020250, 57.129770, 57.670080, 58.020250, 58.020250, 57.452740, 57.096430, 57.296540, 57.188880, 57.452400, 57.061640, 57.003020, 57.434440, 57.491100, 57.684010, 57.041190, 57.684400, 57.001160, 57.213680, 57.480610, 57.298430, 56.684560, 57.309390, 57.298430, 57.564850, 57.388200, 57.213680, 57.213680, 57.213680, 57.375450, 57.385340, 57.213680, 57.388200, 57.213680, 57.213680, 57.215860, 57.215860, 57.215860, 57.388200, 57.628760, 57.628440, 57.354100, 57.388200, 57.043870, 57.651490, 57.041480, 57.298430, 56.954910, 57.314220, 57.221110, 57.382260, 57.132600, 57.135220, 57.554890, 57.215860, 57.227079, 57.215860, 57.168575, 57.215860, 57.215860, 57.483040, 57.215860, 57.213680, 57.121500, 57.213680, 57.215860, 57.031720, 57.215860, 57.215860, 57.395450, 57.215860, 57.215860, 56.757000, 57.215860, 57.026290, 56.664230, 57.477960, 57.577030, 57.581680, 57.578800, 57.489000, 57.487230, 57.123890, 57.827620, 58.297170, 56.849560, 57.917350, 58.111480, 57.654600, 57.830970, 58.462310, 57.213680, 58.472850, 58.279650, 57.837010, 58.383080, 56.930260, 57.747250, 58.369380, 57.012860, 58.462310, 57.744350, 57.744350, 56.664230, 57.312240, 57.923850, 58.383080, 57.313340, 57.044740, 58.007070, 58.109000, 56.955560, 57.834100, 57.644610, 57.827620, 57.388200, 57.031720, 57.492970, 57.468700, 57.042780, 58.103350, 58.010450, 57.121500, 57.644610, 57.913760, 57.564850, 57.648160, 57.827620, 57.397430, 57.666820, 57.123890, 57.654600, 57.664820, 58.468040, 57.303470, 57.215860, 57.578800, 57.472010, 56.983981, 57.483040, 57.303470, 57.305660, 57.375450, 57.395450, 57.213680, 57.215860, 57.215860, 57.301060, 57.575030, 57.926770, 57.301060, 57.399190, 57.834100, 57.038220, 57.215860, 57.215860, 57.671490, 57.133690, 57.282000, 57.134570, 57.221110, 57.491870, 57.132600, 57.224390, 57.929460, 57.580350, 57.402060, 57.485240, 57.241109, 57.282000, 57.390830, 57.388200, 57.303470, 57.388200, 57.298430, 57.564850, 57.213680, 57.213680, 57.031720, 57.121500, 57.385340, 57.385340, 57.558430, 57.215860, 57.477960, 57.303470, 57.305660, 57.123890, 57.215860, 57.490540, 57.480610, 57.005590, 57.487230, 57.123890, 57.388200, 57.005590, 57.490540, 57.215860, 57.480610, 57.217830, 57.121500, 57.215860, 57.213680, 57.747250, 57.215860, 58.019230, 57.399190, 57.378960, 57.227079, 57.121500, 57.307630, 57.217830, 57.285720, 57.662600, 57.213680, 57.215860, 57.371720, 57.572820, 57.215860, 57.215860, 57.215860, 57.403160, 57.572820, 57.580350, 57.468700, 57.036270, 57.036270, 57.572820, 57.036270, 57.036270, 57.301060, 57.662600, 57.830970, 56.946470, 57.648160, 57.385340, 57.651490, 57.388200, 57.123890, 58.010450, 57.662600, 57.298430, 57.036270, 57.036270, 57.029420, 57.036270, 57.029420, 57.036270, 57.036270, 57.022160, 57.036270, 57.036270, 57.041480

-3.587464, -3.588014, -3.698168, -3.713181, -3.713181, -3.588310, -2.438670, -5.136430, -3.492850, -3.524930, -2.339130, -3.701856, -3.646110, -2.415100, -3.524900, -3.525230, -3.574270, -3.509600, -3.510370, -3.064200, -3.669971, -4.056040, -2.241220, -3.709385, -3.180430, -3.180430, -3.228770, -3.046700, -2.857940, -3.337390, -3.086960, -3.086960, -3.256280, -3.158300, -3.769470, -3.769470, -3.051270, -4.403490, -4.419840, -3.326300, -4.456830, -3.178400, -3.939140, -3.326300, -3.619360, -3.320480, -3.211180, -3.924580, -3.256280, -2.772930, -3.256590, -4.642930, -2.844000, -2.734400, -3.086960, -2.895130, -2.841110, -3.325980, -4.340790, -4.044750, -3.178400, -3.411310, -3.289200, -3.178400, -3.046700, -3.343950, -3.289200, -3.922050, -2.861950, -2.789700, -4.246400, -3.113570, -3.580010, -3.922050, -3.063680, -4.246400, -3.320480, -4.456830, -3.228770, -4.236960, -3.988930, -3.988930, -4.340790, -3.051270, -2.844000, -3.063680, -2.924210, -3.256590, -3.398420, -3.939140, -3.411310, -4.419840, -4.173920, -3.777710, -3.924580, -4.403490, -3.180720, -2.556630, -2.920840, -3.843020, -4.522560, -2.552340, -3.311570, -4.481410, -3.176960, -3.327930, -2.870530, -2.940600, -3.713140, -3.327600, -3.311570, -2.984690, -3.730800, -2.956750, -2.796900, -3.379320, -3.031000, -2.372230, -2.593670, -2.984690, -4.446980, -3.843920, -4.404090, -2.771050, -3.078540, -2.804720, -3.196270, -4.271470, -2.593670, -2.486940, -4.126420, -2.863650, -3.031000, -4.170040, -2.761540, -4.228490, -2.870530, -2.552340, -2.372230, -3.992490, -2.938310, -2.804720, -3.843920, -2.925120, -4.271470, -3.713140, -4.958390, -3.844380, -3.392250, -2.454150, -4.512080, -4.228490, -4.498350, -4.460490, -4.880330, -2.585920, -3.730800, -4.818090, -4.445140, -4.475050,

-3.992490, -4.862360, -4.979000, -3.679960, -2.796900, -3.104330, -2.771050, -2.638530,
-4.880330, -4.404090, -3.327930, -2.519210, -2.421030, -3.176380, -4.445140, -2.940600,
-3.395670, -4.862360, -3.392250, -3.379320, -4.170040, -2.984450, -2.881470, -2.519210,
-3.311900, -3.449300, -4.894780, -3.634260, -4.680250, -4.490320, -3.495430, -4.483900,
-3.296800, -3.031500, -3.282940, -2.618390, -3.678520, -3.031500, -2.773300, -4.780230,
-2.963820, -4.662710, -3.461700, -2.585780, -3.387830, -4.616010, -4.781520, -2.540730,
-3.387830, -3.461700, -2.690170, -4.732250, -4.483900, -4.177020, -4.535580, -3.096860,
-4.636260, -2.849850, -4.526440, -3.678520, -3.296800, -3.495430, -4.564020, -2.540730,
-3.387830, -4.556650, -3.051780, -3.242150, -4.765330, -4.699160, -4.576400, -3.994920,
-3.557850, -3.051780, -3.597040, -3.733333, -3.622260, -3.580470, -3.580470, -4.691050,
-4.064030, -4.708650, -4.644880, -3.104700, -4.312370, -4.877300, -3.959420, -4.969590,
-2.681960, -4.709970, -3.912630, -4.711300, -4.654610, -3.076970, -3.039860, -6.068250,
-2.809170, -3.841200, -4.399130, -3.334480, -4.523140, -3.074890, -3.736760, -4.294100,
-3.916400, -3.895480, -3.678010, -3.204040, -3.138170, -4.823450, -3.986200, -4.355060,
-3.580940, -2.956280, -3.532610, -4.323290, -4.299770, -3.138170, -4.179830, -4.863330,
-3.671500, -3.566270, -4.277280, -3.693730, -3.546040, -4.280620, -3.900980, -4.723470,
-3.676380, -3.250860, -3.105230, -3.802820, -3.612440, -2.875080, -2.874660, -4.168770,
-3.625510, -3.737600, -2.781500, -3.980450, -3.486900, -3.687070, -4.386790, -2.841720,
-2.516040, -3.453540, -3.608520, -3.674750, -3.641630, -4.757240, -4.830290, -4.303190,
-4.555890, -3.707860, -3.804570, -3.932130, -3.204040, -3.274830, -3.072810, -4.127920,
-4.358640, -3.769790, -4.385650, -4.357480, -3.548310, -3.241170, -4.565340, -4.548990,
-3.241170, -3.010810, -3.899790, -3.675560, -2.742900, -4.149760, -2.577620, -3.288610,
-4.827490, -4.861940, -3.187180, -4.543780, -2.548950, -3.708690, -4.247380, -4.333290,
-4.394110, -4.699220, -3.770650, -4.518320, -3.547550, -4.215220, -3.997110, -3.498970,
-4.514500, -4.906240, -3.138720, -3.704550, -2.844580, -4.291500, -4.036630, -3.907450,
-3.106300, -3.443620, -4.789610, -3.989200, -2.955340, -3.043390, -3.819430, -3.580010,
-4.492050, -3.640040, -4.708880, -4.743160, -3.104170, -3.855250, -4.627930, -3.835850,
-4.933670, -4.495920, -3.010320, -4.693900, -4.624790, -4.584510, -4.419660, -2.610240,
-3.379360, -3.104170, -2.809560, -3.247220, -4.772290, -3.089700, -5.002750, -3.836740,
-2.809560, -4.688360, -3.056230, -4.014670, -4.903410, -3.104700, -4.228320, -4.082180,
-2.956280, -4.498380, -2.940540, -3.807200, -4.150810, -4.942250, -3.076970, -3.334480,
-4.695520, -4.573440, -3.088630, -3.740130, -3.252720, -3.252720, -3.740130, -3.740130,
-4.413570, -3.735910, -3.574630, -4.413570, -3.735910, -3.740130, -3.735910, -3.252720,
-4.904820, -4.413570, -4.106990, -4.413570, -4.106990, -4.106990, -3.112220, -3.344210,

2010 -3.269290, -2.809170, 57.336231, 57.344383, 57.246052, 57.171620, 57.171620, 57.023430,
57.062580, 55.599930, 57.015170, 57.034550, 57.095470, 57.168575, 57.225330, 56.996500,
57.033650, 57.041730, 57.032140, 57.022150, 57.041010, 57.603020, 57.140273, 57.087600,
57.140170, 57.236863, 57.575040, 57.575040, 57.520670, 57.576210, 57.371030, 57.294980,
57.243490, 57.243490, 57.349720, 57.404560, 57.307610, 57.307610, 57.153960, 58.114780,
58.105480, 57.447820, 58.149700, 57.512170, 57.376990, 57.447820, 57.291660, 57.286170,
57.493890, 57.098670, 57.349720, 57.281750, 57.358700, 57.678700, 57.487900, 57.012480,
57.243490, 57.532470, 57.362160, 57.438840, 57.415090, 58.076120, 57.512170, 57.491810,
57.340400, 57.512170, 57.576210, 57.474580, 57.340400, 57.368260, 57.541680, 57.290630,
57.776220, 57.575640, 57.534730, 57.368260, 57.585060, 57.776220, 57.286170, 58.149700,
57.520670, 57.623640, 57.987160, 57.987160, 57.415090, 57.153960, 57.487900, 57.585060,
57.361580, 57.358700, 57.590780, 57.376990, 57.491810, 58.105480, 57.409090, 57.478200,
57.098670, 58.114780, 57.584020, 57.237980, 57.226850, 57.450310, 57.636230, 56.950530,
57.501870, 58.014440, 57.467270, 57.492720, 57.200260, 57.352470, 57.524010, 57.483740,
57.501870, 57.136530, 57.191350, 57.334380, 57.622950, 57.528100, 57.612280, 57.023070,
57.480350, 57.136530, 58.006120, 57.468260, 58.123750, 57.191930, 57.522050, 57.218670,
57.547940, 58.440640, 57.480350, 56.977760, 58.308370, 57.613530, 57.612280, 57.624800,
57.533340, 57.758560, 57.200260, 56.950530, 57.023070, 57.133550, 57.262660, 57.218670,
57.468260, 57.397500, 58.440640, 57.524010, 58.049180, 57.477240, 57.429140, 56.986860,
57.968910, 57.758560, 57.771500, 57.709340, 57.287060, 56.995300, 57.191350, 57.989450,
57.979200, 58.167320, 57.133550, 57.269500, 57.275770, 57.165080, 57.622950, 57.270290,
57.191930, 57.183690, 57.287060, 58.123750, 57.492720, 56.932710, 56.969010, 57.449310,
57.979200, 57.352470, 57.518940, 57.269500, 57.429140, 57.528100, 57.624800, 57.127550,
57.658320, 56.932710, 57.510850, 57.608170, 57.259770, 58.369500, 57.956490, 57.654840,
57.508810, 58.050340, 57.555930, 57.037340, 57.160790, 56.968190, 57.497530, 57.037340,
57.299710, 57.280330, 57.603820, 57.947870, 57.500220, 56.986310, 57.312400, 58.002760,
57.954210, 57.282960, 57.212400, 57.500220, 57.291210, 57.072260, 58.050240, 58.208520,

57.954310, 57.282960, 57.312400, 57.500220, 57.291210, 57.973360, 58.050340, 58.298530,
58.300910, 57.018810, 57.355330, 57.020730, 57.932690, 57.497530, 57.555930, 57.508810,
57.985850, 57.282960, 57.312400, 57.644530, 57.171920, 58.400850, 57.963650, 57.983050,

6. Making things spatial - projecting our observations

Both our datasets are nested by decade now. We have one more step before we extract the climatic conditions at bird locations. Locations in `birds_counted` are latitude and longitude coordinates. R doesn't know that these are spatial information. We need to **convert** and **project** our data.

Projections are necessary because maps are 2-dimensional, but the earth is 3-dimensional. There is no entirely accurate way to represent the surface of a 3D sphere in 2D. Projections are sets of conventions to help us with this issue. GBIF hosts data from around the world and uses a global projection (WGS84). The Met Office is a UK organization and provides data in the British National Grid projection (BNG).

To project spatial data, use Coordinate Reference System (CRS) strings.

```
# Define geographical projections
proj_latlon <- st_crs("+proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0")
proj_ukgrid <- st_crs("+init=epsg:27700")

# Converting records to spatial points and projecting them
birds_presences <- mutate(birds_counted,
  presences = map(presences, ~ .x %>%
    # Specifying the current projection
    st_as_sf(coords = c("x", "y"), crs = proj_latlon) %>%
    # Transforming to new projection
    st_transform(crs = proj_ukgrid)))
```

7. Extracting key information

Now we are ready to combine the two datasets and extract the climatic conditions at each location for the given decade. This is where the nested structure comes in handy. We join the data frames by their grouping column and can rest assured that the data in the list columns are matched up correctly. This allows us to operate on the list column variables element-wise using the `map()` family functions.

```
# Combining the bird data and the climate data in one data frame
birds_climate <- full_join(birds_presences, climate, by = "decade")

presence_data <- map2_df(
  .x = birds_climate[["rasters"]],
  .y = birds_climate[["presences"]],
# extracting the raster values at presence locations
  ~ raster::extract(x = .x, y = .y) %>%
    as_tibble() %>%
    mutate(observation = "presence"))
```

8. Pseudo-absences

To run a machine learning model, the classification algorithm needs two classes: presences and absences. Our presences are the observations from GBIF. Absences are a lot harder to get.

The difficulty is because of information asymmetry between the presences and absences. With a bird observation we are sure it occurred at that location, but to be certain the bird does **not** occur somewhere, we would have to continuously monitor the site.

One way to deal with this problem is to generate "pseudo-absences". Pseudo-absences are a random sample from the entire study area. We assume that the species does not occur at the random locations and our hope is that the average actual probability of occurrence for the bird in these random locations is low enough to give our algorithm something to learn

```
# Defining helper function in order to create pseudo-absence data
create_pseudo_absences <- function(rasters, n, ...) {
    set.seed(12345)
    sampleRandom(rasters, size = n * 5, sp = TRUE) %>%
    raster::extract(rasters, .) %>% as_tibble() %>%
    mutate(observation = "pseudo_absence")
}

# Creating pseudo-absence proportional to the total number of records per decade
pseudo_absence_data <- pmap_df(.l = birds_climate, .f = create_pseudo_absences)

# Combining the two datasets
model_data <- full_join(presence_data, pseudo_absence_data) %>%
  mutate(observation = factor(observation)) %>% na.omit()
```

```
Joining, by = c("minimum.temperature", "maximum.temperature", "rainfall", "wind.speed
```

9. Making models - with caret

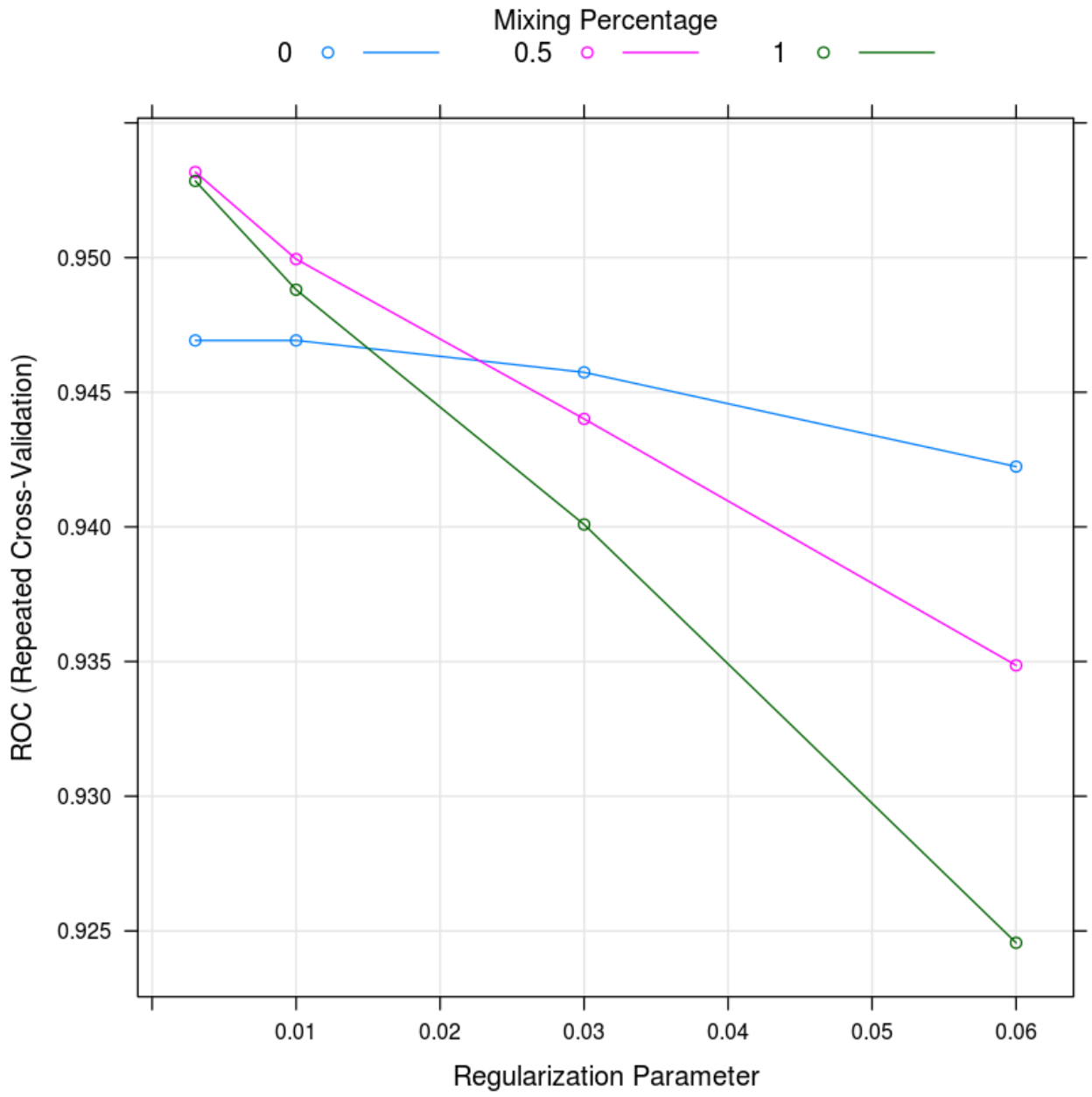
We are ready to train our model. We will use `glmnet`, which fits a generalized logistic regression (*glm*) with elastic net regularization (*net*). Our algorithm has several "hyperparameters". These are variables used by the machine learning algorithm to learn from the data. They influence the

performance of the model and often interact with one another, so it is difficult to know the right settings *apriori*.

To figure out a good set of hyperparameters, we need to try several possible scenarios to see which ones work best. `caret` makes this easy. All we need to do is define a "tuning grid" with sets of possible values for each training parameter. Then use cross-validation to evaluate how well the different combinations of hyperparameters did building the predictive model.

```
# Importing caret and setting a reproducible seed
library(caret)
set.seed(12345)

# Create a tuning grid with sets of hyperparameters to try
tuneGrid <- expand.grid(alpha = c(0, 0.5, 1), lambda = c(.003, .01, .03, .06))

# Create settings for model training
trControl <- trainControl(method = 'repeatedcv', number = 5, repeats = 1,
  classProbs = TRUE, verboseIter = FALSE, summaryFunction = twoClassSummary)

# Fitting a statistical model to the data and plot
model_fit <- train(
  observation ~ ., data = model_data,
  method = "glmnet", family = "binomial", metric = "ROC",
  tuneGrid = tuneGrid, trControl = trControl)

plot(model_fit)
```

```
Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift
```



10. Prediction probabilities

We have now built our first species distribution model. Next, we will use it to predict the probability of occurrence for Scottish crossbill across the UK. We will make a prediction for each decade and each cell of the grid. Since we fit a logistic regression model, we can choose to predict the probability. In our case, this becomes the "probability of occurrence" for our species.

```
# Use the model to make a prediction
climate_df[["prediction"]] <- predict(
    object = model_fit,
    newdata = climate_df,
    type = "prob")[["presence"]]

head(climate_df)
```