R packages for occupancy analysis

In this section, we'll cover the basic initial steps for applying fill patterns to eBird data. In Chapter 4, we used analytical approaches that took into account the change in detection. We modeled covariatas that are known to affect detection (e.g., duration, time of day) alongside covarias that affect occurrence. In contrast, filling models jointly simulate the ecological process of species occurrence and the process of observing the detection of species, but evaluate them as separate processes. This model structure allows us to take into account the change in the likelihood of detection in assessing the occurrence of species. In this section, we will not provide much detail about the theory and assumptions about the placement of models; However, there are plenty of background literature and applications of fill-in models, and readers looking to learn more about this area may want to consult a book on this topic MacKenzie et al. (2017). Applying fill patterns typically requires data from repeated sampling visits (cases) on a single site during a time frame during which the population is closed (e.g., no changes in fill-in between surveys). Although eBird checklists are not designed to meet these requirements, you can apply fill patterns to eBird data by pulling out a subset of data that matches the closing assumptions and corresponding to the duplicate data structure. Here we present a simple example of how to process eBird data to meet these requirements. To illustrate our example, we apply a single-season placement model to assess the filliness and probability of thrush wood detection in the month of June for BCR 27. This section differs from the previous section on simulating meeting speed in two important ways. First, the random forest model used in Chapter 4 exemplifies the approach to machine learning, while the fillability models used in this section take a more traditional approach to probability. This latest class of statistical models is widely used to address specific issues and hypotheses, while the purpose of machine learning is primarily to identify patterns and predictions (Bzdok, Altman and Krzywinski 2018). Second, approaches to machine learning can accommodate complex non-linear effects and interactions between covarias, and are useful in modeling habitat associations that may differ on large spatial and temporal scales. By contrast, placement models are well suited to describe linear effects and simpler interactions. In this example, we specifically focus on the mechanics of filtering and formatting data according to fill patterns, and less on the specifics of selecting appropriate predictor variables to assess the probability of identifying and placing to generate a set of candidate models, used to select a model. The precursors we include are informed by our findings on variable assessments of the importance of випадкову лісову модель у Главі 4, а також наші існуючі знання про види, що моделюються. Якщо ви працювали з попередніми розділами, ви повинні мати всі дані, необхідні для цього розділу. Ви також можете завантажити пакет даних і розпакувати його в каталог проекту. library(auk) library(lubridate) library(sf) library(dggridR) library(unmarked) library(raster) library(ebirdst) library(MuMIn) library(AICcmodavg) library(fields) library(tidyverse) # resolve namespace conflicts select <- dplyr::select projection <- raster::projection # set random number seed to insure fully repeatable results set.seed(1) # setup output directory for saved results if (!dir.exists(output)) { dir.create(output) } # ebird data ebird <- read_csv(data/ebd_woothr_june_bcr27_zf.csv) %>% mutate(year = year(observation_date), # occupancy modeling requires an integer response species_observed = as.integer(species_observed)) # modis land cover covariates habitat <- read_csv(data/pland-elev_location-year.csv) %>% mutate(year = as.integer(year)) # combine ebird and modis data ebird_habitat <- inner_join(ebird, habitat, by = c(locality_id, year)) # prediction surface pred_surface <- read_csv(data/pland-elev_prediction-surface.csv) # latest year of landcover data max_lc_year <- pred_surface$year[1] r <- raster(data/prediction-surface.tif) # load gis data for making maps map_proj <- st_crs(102003) ne_land <- read_sf(data/gis-data.gpkg, ne_land) %>% st_transform(crs = map_proj) %>% st_geometry() bcr <- read_sf(data/gis-data.gpkg, bcr) %>% st_transform(crs = map_proj) %>% st_geometry() ne_country_lines <- read_sf(data/gis-data.gpkg, ne_country_lines) %>% st_transform(crs = map_proj) %>% st_geometry() ne_state_lines <- read_sf(data/gis-data.gpkg, ne_state_lines) %>% st_transform(crs = map_proj) %>% st_geometry() По-перше, витягне підмножину даних eBird, яка відповідає припущенням моделей заповнюваності, а потім ми виконаємо просторове підмножину для боротьби з просторової упередженістю в даних. Почнемо з фільтрації наших даних, щоб включити тільки контрольні списки з 5 або менше спостерігачів, щоб зменшити джерела зміни в виявленні, і тому, що є дуже мало контрольних списків з більш ніж 5 спостерігачів. Крім того, ми підмножимо спостереження до останнього року, для якого ми маємо дані (2019), щоб відповідати односезонній моделі розміщення. # фільтр перед створенням даних моделі розміщення ebird_filtered <- filter(ebird_habitat, number_observers <= 5, year == max(year)) У деяких ситуаціях ви можете додатково фільтрувати дані за результатами розвідувального аналізу, аналогічного тому, який проводився в розділі 2.5. Незважаючи на те, що більшість контрольних списків у цьому прикладі представлені 2 або меншою кількістю спостерігачів, ми не будемо додатково фільтрувати спостереження для нашого прикладу розміщення. 3 additional limitations for data suitable for filling modeling, filling, to save more checklists at this point. Next, we need to create a discovery history for each location we define as a site. In this example, we define the Month of June as the time period during which we assume that there are no changes in the fulcility between secondary sampling cases for Wood Thrush in BCR 27. The time frame over which closure can be assumed will differ among species and terrains, requiring careful consideration. We define the site as a specific location (latitude/longitude) visited by at least twice the same observer during the closing period defined by us (i.e. the month of June). The auk filter_repeat_visits() function is designed to extract a subset of eBird data suitable for filling modeling. Using the feature, we first filter data only on sites that have at least 2 visits (min_obs). We also define the maximum number of repeat visits (max_obs) as 10 visits or checklists. When a particular site has been visited more than 10 times, the feature randomly selects 10 checklists from all visits to that site. If we had data for more than one year, we could have used annual_closure =TRUE to determine that populations are closed within specified timeframes for a given year, but not closed between years. In other words, the appearance does not change from one re-visit to the next for this sampling event (e.g. year), but may vary between years. If we want to define closing periods over the years, we can identify them in terms of the number of days n_days. For example, n_days = 10 will define adjacent sets of 10 days, starting with the oldest observation date in the data, and use them as consecutive closing periods. Here we do not define n_days and consider all checklists of June 2019 as repeat visits for one season. Finally, the site_vars defines a set of variables that defines the site. In this example, the site is shared by location and observers. Any set of data variables can be used to identify sites. For example, site_vars = locality_id can be used to identify sites that use the location independently of the observer. occ <- filter_repeat_visits(ebird_filtered, min_obs = 2,max_obs = 10, annual_closure = TRUE, date_var = observation_date, site_vars = c(locality_id, observer_id)) # the entire data set nrow(ebird_habitat) #> [1] 48450 # reduced data set nrow(occ) #> [1] 3724 # number of individual sites n_distinct(occ$) #> [1] 988 This feature n_distinct(occ$) #> [1] 988 filter_repeat_visits() added three new columns to the data set: the site is a unique site IDENTIFIER (here, location and observer), closure_id defines the primary closing period (in this example, year), and n_observations determines the number of visits to each site. Our capture stories now are formatted for the single-season placement model and are ready for analysis. Note that we made a compromise on the sample size by dropping the reset 10,415 checklists to 3,724 checklists at 988 sites. We will use our filtered observations in accordance with the single-season placement model using unmarked R packaging. For more information about the type of data format required for this package, refer to the documentation for the unmarkedWide() function format. The auk format_unmarked_occu() function converts data from a vertical format in which each row is observed (as in EBD) into a horizontal detection history, where each row is a site. In this format, each column represents a repeat visit– for this example, we will have up to 10 columns of detection events. This data format is commonly used for most applications of the placement model, including without markup. At this stage, we need to specify what variables will be the ecological process (i.e. filling) with covarias, and which will be the observation process (i.e. detection) of kovariat. Covariates (site_covs) fill will be unique at the site level, while covariates (obs_covs) detection can also be unique to each site, as well as the case of sampling (i.e. checklist). For this example, we will use MODIS land cover variables as habitat covariates to model the likelihood of Wood Thrush filling. Based on measures of prognostic importance from Chapter 4, we include deciduous broadleaf forests and mixed forests as types of habitats for which we expect positive relationships with filling, and farmland and urban ones for which we expect negative relationships. To assess the likelihood of detection, we include three variable efforts related to the detection process. The type of habitat has been shown to affect detection in bird species, such as some species harder to detect in densely forested habitats relative to more open habitat types. As such, we also include deciduous broadleaf forest and mixed forest as covariates to detect probability. Filling patterns allow us to tease out the different effects of habitat on both detection and probability of filling. # format for unmarked occ_wide <- format_unmarked_occu(occ, site_id = site, reply = species_observed, site_covs = c(n_observations, latitude, longitude, pland_04_deciduous_broadleaf, pland_05_mixed_forest, pland_12_cropland , pland_13_urban), obs_covs = c(time_observations_started, duration_minutes, effort_distance_km, number_observers, protocol_type, pland_04_deciduous_broadleaf, pland_05_mixed_forest)) As discussed in section 4.3, spatial beeping observations eBird reduces spatial bias. We will use the same hexagonal approach to subcamping as in Chapter 4; however, here we will obey at the level of sites, not observations. For this example, we will try one site for 5 cell of the grid. Note that because we are not observer_id the site definition, this sublinguation process will select only one line or set of visits from one observer to the within each cell 5 km. # generate hexagonal grid with ~5 km between dggs < -dgconstruct(spacing=5) #=get=hexagonal=cell=id=for= each= site= occ_wide_cell=></-> <- occ_wide= %=>%mutate(cell=dgGEO_to_SEQNUM(dggs, longitut latitude)$seqnum) # sample of one site per grid cell occ_ss <- occ_wide_cell= %=>%group_by(cell) %>%sample_n(size = 1) %>% ungroup() %>%select(-cell) # calculate the percentage of decrease in the number of sites 1 - nrow(occ_ss) / nrow(occ_wide) This led to a decrease in the number of sites by 41% Finally, we will turn this observation data frame into an unmarked object so that we can start adapting the placement models. <- formatwide(occ_ss,= type=unmarkedFrameOccu)= summary(occ_um)= #=>occ_um object #> #> 584 site #> Maximum number of site observations: 10 #> Total number of site observations: 3.86 #> Sites with at least one detection: 64 #> #> Tab y observations: #> 0 1 <NA>#> 2125 128 3587 #> #> Site-level covariates: #> n_observations latitude longitude pland_04_deciduous_broadleaf pland_05_mixed_forest pland_12_cropland #> Min. :29.7 mins. :-91.4 min. :0.000 Min >. 000 1st qu.:0. 000 1st qu.:0. 000 #> Median : 2.00 Median :33.1 Median :-81.4 Median :0.000 Median :0.000 Median :0.000 #> Average : 3.86 Average:33.2 Medium :-82.0 Average:0.054 Average:0.047 Average:0.025 #> 3rd Qu.: 5.00 3 Qurd.:35.1 3rd Qu.:-78.3 3 Qu.:0. 032 3rd Qu.:0. 000 3rd Qu.:0. 000 #> Max.: 10.00 Max.> > > pland_13_urban: 37.4 Max. 000 #> Median:0.000 #> Medium:0.153 #> Q3:0. 194 #> Max.: 1.000 #> #> Observation-level covariates: #> time_observations_started duration_minutes effort_distance_km number_observers protocol_type pland_04_deciduous_broadleaf #> Min. Min. :1 Length:5840 Min >. 1st Qu.:1 class:symbol 1st Qu.:0 #> Median :10 Median : 35 Median :0 Median :1 Mode:symbol Median:0 #> Average:12 Average: 53 Average:1 Average:1 Average:0 #> 3rd Qu.:16 3rd Qu.: 70 3rd Qu.Qu:1 3rd Qu.:1 3rd Qu.:0 #> Max.:24 Max.:300 Max. :5 Max. :4 Max. :1 #> NA:3587 NA:3587 NA:3587 NA:3587 NA's :3587 #> pland_05_mixed_forest #> Min. :0 #> 1 thousand: #> Median:0 #> Average:0 #> 3rd Qu.:0 #> Max. :1 #> NA:3587 Now that we've created a data frame with detection history and covarias, we can use unmarked to fit the single-season fill model. As mentioned above, readers can learn more about fill-in models and the variety of modeling approaches at MacKenzie et al. (2017). Here we are simply approaching the single-season model of placing our data using the occu() function, specifying the detection and filling of the covarias, respectively, using the dual right form ~ коваріатів ~ заповнюваність коваріатів. # </NA></-></-></-></-> </NA></-></-></-></-> model occ_model <- occu(~ time_observations_started + duration_minutes + effort_distance_km + number_observers + protocol_type + pland_04_deciduous_broadleaf + pland_05_mixed_forest ~ pland_04_deciduous_broadleaf + pland_05_mixed_forest + pland_12_cropland + pland_13_urban, data = occ_um) # look at the regression coefficients from the model summary(occ_model) #> #> Call: #> occu(formula = ~time_observations_started + duration_minutes + #> effort_distance_km + number_observers + protocol_type + pland_04_deciduous_broadleaf + #> pland_05_mixed_forest ~ pland_04_deciduous_broadleaf + pland_05_mixed_forest + #> pland_12_cropland + pland_13_urban, data = occ_um) #> #> Occupancy (logit-scale): #> Estimate SE z P(>|z|) #> (Intercept) -2.04 0.242 -8.399 4.49e-17 #> pland_04_deciduous_broadleaf 5.48 1.897 2.889 3.86e-03 #> pland_05_mixed_forest 1.27 0.828 1.536 1.25e-01 #> pland_12_cropland -1.07 1.992 -0.538 5.90e-01 #> pland_13_urban -1.91 0.978 -1.956 5.05e-02 #> #> Detection (logit-scale) : #> Estimate SE z P(>|z|) #> (Intercept) -1.16869 0.56714 -2.061 0.03934 #> time_observations_started -0.00305 0.03038 -0.100 0.91996 #> duration_minutes 0.00634 0.00318 1.992 0.04633 #> effort_distance_km -0.22876 0.13539 -1.690 0.09109 #> number_observers 0.07085 0.30928 0.229 0.81882 #> protocol_typeTraveling 0.71787 0.39297 1.827 0.06773 #> pland_04_deciduous_broadleaf -0.26478 0.64762 -0.409 0.68265 #> pland_05_mixed_forest 2.66054 1.01091 2.632 0.00849 #> #> AIC : 694 #> Кількість сайтів: 584 #> optim код зближення: 0 #> оптимальні ітерації: 55 #> Початкове завантаження ітерацій: 0 Хоча мало доброти придатних тестів існують для моделей заповнюваності, ми демонструємо, як виконати MacKenzie і Bailey (2004) хороший тест. Такий підхід обчислює статистику чи-квадрата Пірсона з спостережуваних і очікуваних частот історії виявлення для даної моделі. У цьому прикладі ми використовуємо тестову функцію mb.gof.test() у пакеті AICcmodavg, яка може обробляти моделі заповнюваності, вироблені функцією occu() без розмітки. Зверніть увагу, що для отримання точних результатів цей процес вимагає імітації великої кількості зразків завантажунки, що може зайняти багато часу для запуску. Щоб зберегти час виконання розумним, ми встановлюємо nsim = 10, щоб імітувати 10 зразків для цього прикладу; однак, при запуску цього за звичайних обставин, ви повинні збільшити це до набагато більшої кількості симуляцій (наприклад, nsim = 1000). occ_gof <- mb.gof.test(occ_model, nsim = 10, plot.hist = FALSE) # приховати таблицю chisq, щоб дати більш простий вихід occ_gof $ chisq.table <- NULL print(occ_gof) #> #> MacKenzie і Бейлі доброта-підходить для односезонної моделі заповнюваності #> #> Статистика Chi-square = 1555 #> Кількість зразків початкового завантаження = 1000 #> = 0.621 #> #> Квантилі статистики завантаження: #> 0% 25% 50% 75% 100% #> > <1> <6> 537 1387 1715 2254 35800 #> #> Оцінка c-hat = 0,73 Для цього прикладу прикладу obtaining calculated chi-square statistics under the zero sample distribution is indicated by a p-value of 0.621. How > 0.1 no reason to consider the lack of a fit. We also get an overreail value (c-hat) for a model that comes by dividing observed chi-square statistics using statistics derived from modeling. In this example, c-hat = 0.73, which is very close to c-hat = 1, indicates that the variance is no greater than the perch, and that there is no evidence of excessive overreaction. Again, under normal circumstances we would like to run a lot more syming, but based on this smaller run, the test statistics suggest that there is no evidence of a lack of suitability for this model to this data. See MacKenzie and Bailey (2004) for more information on this test. So far, we have a single global model that includes all covariates that we believe will affect the likelihood of filling up and detecting. In general, we propose to carefully consider when choosing a set of models of candidates that will be launched and comparable during the selection of the model. In this example, we will use the dredge() function to create a set of candidate models using different combinations of covarias in the global model. It will take a while to run, and therefore recommend choosing carefully candidate model sets instead of this approach. # dredge all possible combinations of the occupancy covariates occ_dredge <- dredge(occ_model) # model comparison to explore the results for occupancy mc <- as.data.frame(occ_dredge) %>% select(starts_with(psi(p), df, AICc, delta, weight) # shorten names for printing names(mc) <- names(mc) %>% str_extract((?<=psi\\(pland_[0-9]{2}_)[a-z_]+) %>% coalesce(names(mc)) # take a quick peak at the model selection table mutate_all(mc, ~ round(., 3)) %>% head(18) %>% knitr::kable() 5.43 1.34 -1.84 7 686 0.000 0.028 5.16 1.31 -1.88 9 687 0.390 0.023 5.76 -2.03 6 687 0.397 0.023 5.54 -2.07 8 687 0.687 0.020 5.45 1.28 -1.85 8 687 1.038 0.017 5.78 -2.04 7 687 1.225 0.015 5.40 1.30 -1.88 7 688 1.408 0.014 5.34 1.37 -1.85 8 688 1.546 0.013 5.74 -2.07 6 688 1.656 0.012 5.33 1.30 -1.05 -1.90 8 688 1.733 0.012 5.41 1.34 -1.84 8 688 1.967 0.011 5.62 -1.26 -2.10 7 688 1.981 0.011 5.68 -2.05 7 688 2.033 0.010 5.38 1.41 -1.87 6 688 2.045 0.010 5.34 1.34 -1.85 8 688 2.047 0.010 5.43 1.34 -1.84 8 688 2.056 0,010 5,06 1,27 -1,09 -1,94 10 688 2,105 0,010 5,39 -1,30 -1,30 -0,010 2.14 9 688 2.249 0.009 Виправлений інформаційний критерій Akaike (AICc) вимірює ймовірність того, що кожна модель генерує дані, які ми спостерігали, скориговані на кількість параметрів у моделі. Lower values indicate models with better fits to data, punishing for the added number of parameters. Delta is the difference in AICc values between this model and the model that most likely generated data (i.e. one that the lowest AICc), and is relatively conditional on the set of model candidates. Finally, the weight of the AIK is a delta transformation that can be interpreted as the likelihood that this model is the most likely one of the models of candidates who generated data, and is also conditional on a set of candidate models. A quick glance at dredging results, focusing on visualizing what most affects the likelihood of filling up, shows that for Wood Thrush's example, there is no clear single model, or even a small set of models, that are more likely to spawn our data. This is evident from the number of models that all fall under the Delta AICc~2-3 value, and the large number of models with moderate AIC weights. With that in mind, and the fact that all of our effects are linear and use the same family and communication feature, we'll average all AICc-weighted models to get an average prediction model. However, there may be scenarios in which there is a clear set of highly effective models, in which case you can use the get.models() function to extract only these models to averaging. For the sake of efficiency, we will only average the top models that we will define as those collectively composed of 95% weight. This will trivialize the results because models with less support have very low weights and therefore contribute little to weighted average projections. # select models with the most support for model averaging (< 2.5 delta aicc) occ_dredge_delta <- get.models(occ_dredge, subset = delta <= 2.5) # average models based on model weights occ_avg <- model.avg(occ_dredge_delta, fit = TRUE) # model averaged coefficients for occupancy and detection probability coef(occ_avg) #> psi(Int) psi(pland_04_deciduous_broadleaf) psi(pland_05_mixed_forest) #> -1.99418 5.47373 1.32462 #> psi(pland_13_urban) p(Int) p(duration_minutes) #> -1.94454 -1.05931 0.00573 #> p(pland_05_mixed_forest) p(effort_distance_km) p(protocol_typeTraveling) #> 2.84740 -0.18494 0.55552 #> psi(pland_12_cropland) p(time_observations_started) p(pland_04_deciduous_broadleaf) #> -1.17179 0.00520 -0.04471 #> p(number_observers) #> 0.01248 In this table , the psi() coefficients are from the occupancy submodel and the p() coefficients are from the detection submodel. based on these results, we see that an increase in deciduous broadleaf forest has a much stronger effects on occupancy probability than an increase in mixed forest. As expected, the increase in the percentage of urban area and farmland as a decrease in the likelihood of accommodation. A unique feature of placement models is that we can investigate whether certain covarias specifically affect the likelihood of detection separately from any impact on occurrence that was not possible through a machine learning approach in Chapter 4. In particular, we have already seen that коваріати впливають на заповнюваність, але ми можемо оцінити, як ці коваріати впливають на ймовірність виявлення, використовуючи ту саму модель, що усереднює результати, які ми згенерували вище, умовні на наявній і потенційно виявленій птиці. Ми включили листяний широколистий ліс (pland_04) із змішаний ліс (pland_05) як виявлення коваріатів у глобальній моделі. # порівняння моделі для вивчення результатів для виявлення md <- as.data.frame(occ_dredge)= %=>% select(starts_with(p(), df, AICc, дельта, вага) # скоротити імена для друку імен (md) <- names(md)= %=>% str_extract((?<=p\\(pland_[0-9]{2}_)[a-z_]+) %=> % coalesce(names(md)) # take a quick peak at the model selection table md[1:18,] #> p(Int) p(duration_minutes) p(effort_distance_km) p(number_observers) deciduous_broadleaf mixed_forest p(protocol_type) #> 1426 -0.972 0.00530 NA NA NA 2.71 <NA>#> 1460 -1.166 0.00652 -0.2158 NA NA 2.75 + #> 1170 -1.043 0.00546 NA NA NA 3.02 <NA>#> 1204 -1.245 0.00653 -0.2096 NA NA 3.04 + #> 1458 -1.120 0.00436 NA NA NA 2.87 + #> 1202 -1.202 0.00441 NA NA NA 3.16 + #> 1457 -0.926 NA NA NA NA 2.75 + #> 1428 -0.929 0.00645 -0.0742 NA NA 2.62 <NA>#> 1201 -1.007 NA NA NA NA 3.04 + #> 1938 -0.967 0.00527 NA NA NA 2.71 <NA>#> 1490 -1.079 0.00549 NA NA NA 2.74 <NA>#> 1682 -1.033 0.00541 NA NA NA 3.00 <NA>#> 1172 -1.004 0.00650 -0.0670 NA NA 2.94 <NA>#> 1425 -0.598 NA NA NA NA 2.43 <NA>#> 1434 -0.984 0.00531 NA NA 0.0596 2.73 <NA>#> 1430 -0.974 0.00530 NA 0.00154 NA 2.71 <NA>#> 1972 -1.162 0.00648 -0.2165 NA NA 2.74 + #> 1716 -1.237 0.00648 -0.2104 NA NA 3.02 + #> p(time_observations_started) df AICc delta weight #> 1426 NA 7 686 0.000 0.02827 #> 1460 NA 9 687 0.390 0.02326 #> 1170 NA 6 687 0.397 0.02318 #> 1204 NA 8 687 0.687 0.02005 #> 1458 NA 8 687 1.038 0.01683 #> 1202 NA 7 687 1.225 0.01533 #> 1457 NA 7 688 1.408 0.01398 #> 1428 NA 8 688 1.546 0.01305 #> 1201 NA 6 688 1.656 0.01235 #> 1938 8 688 1.733 0.01189 #> 1490 0.00888 8 688 1.967 0.01057 #> 1682 7 688 1.981 0.01050 #> 1172 7 688 2.033 0.01023 #> 1425 6 688 2.045 0.01017 #> 1434 8 688 2.047 0.047 0.01016 #> 1430 8 688 2.056 0.01011 #> 1972 10 688 2.105 0.000987 #> 1716 9 688 2.249 0.00918 From these results, it's clear that several of our work covariates do not influence protein-mainly protocol type and the number of observers. Однак, тому числі середовище проживання Covariates-особливо змішані ліси вплинули на виявлення ймовірності, як це спостерігається відмінності AIC to AIC ваги для моделей, які включають цей коваріат. Давайте подивимося на узагалення коефіцієнтів моделі, щоб побачити, як вони впливають на виявлення та заповнюваність. coef(occ_avg) %>% enframe() #> # Tibble: x 2 #> значення імені #> <chr> <dbl>#> 1 psi(Int) -1.99 #> 2 psi(pland_04_deciduous_broadleaf) 5.47 #> 3 psi(pland_05_mixed_forest) 1.32 #> 4 psi(pland_13_urban) -1.94 #> 5 p(Int) -1.06 #> 6 p(duration_minutes)</dbl> </chr> </NA> </NA> </NA> </NA> </NA> </NA> </NA> </NA> </NA> </NA> </=p\\(pland_[0-9]{2}_)[a-z_]+)> </-> </-> </-> #> # ... з ще 7 рядків Знову ж таки, psi() коефіцієнти з підмоделі заповнюваності і p() коефіцієнти з виявлення субмоделі. Для виявлення ймовірності ми бачимо, що хоча листяний ліс збільшує ймовірність заповнюваності деревини молочниці, це зменшує ймовірність виявлення, оскільки птахів важче бачити і чути в густому лісі. Однак змішаний ліс позитивно впливає як на заповнюваність, так і на ймовірність виявлення. Здатність дражнити різні ефекти, які коваріати мають на виявлення і заповнюваність є однією з сильних сторін заповнюваності моделювання. У цьому розділі ми оцінимо розподіл Wood Thrush в BCR 27. Подібно до розділу 3.4, ми будемо генерувати поверхню прогнозування за допомогою covariates земельного покриву PLAND, зведеного на звичайній сітці точок через BCR 27. Для цього ми використовуватимемо функцію predict() для оцінки ймовірності заповнюваності, стандартних помилок і інтервалів довіри. Ми будемо використовувати predict() на виході model.avg() для прогнозування, використовуючи усереднію коефіцієнти моделі від моделі кандидата, встановленої з принаймні 95% кумулятивної ваги AIC. # примітка: код нижче може зайняти до години, щоб запустити! occ_pred <- predict(occ_avg, newdata = as.data.frame(pred_surface), type = state) # add to prediction surface pred_occ <- bind_cols(pred_surface, occ_prob = occ_pred$fit, occ_se = occ_pred$se.fit) %>% select(latitude, longitude, occ_prob, occ_se) Next, we'll convert this data frame to spatial features using sf, then rasterize the points using the prediction surface raster template. r_pred <- pred_occ %>% # convert to spatial features st_as_sf(coords = c(longitude, latitude), crs = 4326) %>% st_transform(crs = projection(r)) %>% # rasterize rasterize(r) r_pred <- r_pred[[c(occ_prob, occ_se)]] # save the raster tif_dir <- output if (!dir.exists(tif_dir)) { dir.create(tif_dir) } writeRaster(r_pred[[occ_prob]], filename = file.path(tif_dir, occupancy-model_prob_woothr.tif), overwrite = TRUE) writeRaster(r_pred[[occ_se]], filename = file.path(tif_dir, occupancy-model_se_woothr.tif), overwrite = TRUE) Finally, we can map these predictions! # project predictions r_pred_proj <- projectRaster(r_pred , crs = map_proj $proj4string, метод = ngb) par(mfrow = c(2, 1)) для (нм в іменах(r_pred)) { r_plot <- r_pred_proj[[nm]] par(mar = c(3.5, 0.25, 0.25, 0.25)) # встановити ділянку ділянки(bcr)), col = NA, border = NA) plot(ne_land, col = #dddddd, border = #888888, lwd = 0.5, add = TRUE) # ймовірність розміщення або стандартна помилка, якщо (nm == occ_prob) { назва <- Ймовірність заповнюваності деревини молочниці brks <- seq(0, 1, length.out = 21) lbl_brks <- seq(0, 1, length.out = 11) %>% round(2) } else { title <- Невизначеність заповнюваності деревини молочниці (SE) mx <- стеля (1000 * cellStats(r_plot, max)) / 1000 brks <- seq(0, mx, length.out = 21) lbl_brks <- mx, length.out = 11) %>% round(2) } pal <- <- - 1) plot(r_plot, col = pal, breaks = brks, maxpixels = ncell(r_plot), легенда = FALSE, додати = TRUE) # межі сюжету (bcr, межа = #000000, col = NA, lwd = 1, додати = TRUE) сюжет (ne_state_lines, col = #ffffff, lwd = 0.75, додати = TRUE) сюжет (ne_country_lines, col = #ffffff, lwd = 1.5, додати = TRUE) box() # legend par(new = TRUE, mar = c(0, 0, 0, 0)) image.plot(zlim = range(brks), legend.only = TRUE, breaks = brks, col = pal, smallplot = c(0.25, 0.75, 0.06, 0.09), горизонтальний = TRUE, axis.args = list(at = lbl_brks, етикетки = lbl_brks, fg = чорний, col.axis = чорний, cex.axis = 1000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000.75, lwd.ticks = 0.5, padj = -1.5), legend.args = list(text = title, side = 3, col = black, cex = 1, line = 0)) }

Bzdok, Danilo, Naomi Altman і Martin Krzywinski. 2018. «Точки значущості: статистика проти машинного навчання». Природні методи 15 (квітень): 233–34. . Маккензі, Деріл І., і Лариса Л. Бейлі. 2004. «Оцінка придатності моделей сайту-розміщення». Журнал сільськогосподарської, біологічної та екологічної статистики 9 (3): 300–318. Маккензі, Деріл І., Джеймс Д. Ніколс, Дж. 2017. Оцінка та моделювання заповнюваності: виводити закономірності та динаміку виникнення видів. Elsevier. Elsevier.