8.5 Logistic Regression

Manual of Applied Spatial Ecology

3/11/2022

Resource selection requires "used" and "available" habitats and the study designs would take up an entire course all on there own. In this section, we hope to show how we can go about this approach all in R and not need to involve excel spreadsheets with multiple columns of data. More details on methods to estimate resource selection functions (RSFs) or resource selection probability functions (RSPFs) can be found in the literature (Manly et al. 2002, Millspaugh et al. 2006, Johnson et al. 2006). We do not expect you to be experts in RSFs after this section but we want you to be able to implement these methods in R after determining study design, data collection protocol, and methodology to best achieve your research objectives.

8.4.1 Logistic regression

As we move forward in this section, we are going to assume that your study design and data assessment prior to this section addresses any collinearity in predictor variables and a priori hypothesis were used to generate your models used in logistic regression. There are several ways to to calculate RSFs in R using logistic functions that can assess population level or intra-population variation. The use of General Linear Models with various function using the lme4 package is often used for estimating population-level models only. Alternatively, we can assess intra-population variation using the glmer function. Assessing intra-population variation is a mixed-model approach that provides a powerful and flexible tool for the analysis of balanced and unbalanced grouped data that are often common in wildlife studies that have correlation between observations within the same group or variation among individuals at the same site (Gillies et al. 2006).

1. Exercise 8.5 - Download and extract zip folder into your preferred location

2. Set working directory to the extracted folder in R under Session - Set Working Directory...

3. Now open the script LogisticRSF.Rmd" and run code directly from the script

4. First we need to load the packages needed for the exercise

```
library(lme4)
library(AICcmodavg)
library(adehabitatHR)
```

5. Load files for each season in the mule deer dataset We may need to identify some numerical data as factors for analysis prior to implementing resource selection analysis.

```
data_1 <- read.csv("MD_winter12.csv",header=T)

################################ MODELING ###############################
data_1$crop=as.factor(data_1$crop)
#dataset$SEASON=factor(dataset$SEASON)
data_1[,2:3]=scale(data_1[,2:3],scale=TRUE)#standardize data to mean of zero
```

6. We may need to use code that changes Reference Categories of our data. For our analysis we are going to define reference category of *used* habitat as crop= 1. Crop category 1 is sunflowere which was the crop of interest but was not selected for based on Selection Ratios in Exercise 8.4.

```r
fit1 = glmer(use ~ relevel(crop,"1")+(1|animal_id), data=data_1, family=binomial(link=
  "logit"),nAGQ = 0)#Sunflower and cover model
fit2 = glmer(use ~ d_cover+(1|animal_id), data=data_1, family=binomial(link="logit"),nAGQ
  = 0)#Distance to cover only model
fit3 = glmer(use ~ d_roads+(1|animal_id), data=data_1, family=binomial(link="logit"),nAGQ
  = 0)#Distance to roads only model
fit4 = glmer(use ~ d_cover+d_roads+(1|animal_id), data=data_1, family=binomial(link="logit"),
  nAGQ = 0)#Distance to cover and roads model
fit5 = glmer(use ~ 1|animal_id, data=data_1, family=binomial(link="logit"),nAGQ = 0)#Intercept model
```

7. We can view the results of our modeling procedure to select the best model using Akaike's Information Criteria (AIC; Burnham and Anderson 2002).

```r
fit1
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula: use ~ relevel(crop, "1") + (1 | animal_id)
##    Data: data_1
##       AIC       BIC    logLik  deviance  df.resid
##  31610.18  31659.99 -15799.09  31598.18     29800
## Random effects:
##  Groups    Name        Std.Dev.
##  animal_id (Intercept) 0.3106
## Number of obs: 29806, groups:  animal_id, 14
## Fixed Effects:
##         (Intercept)  relevel(crop, "1")2  relevel(crop, "1")3
##              0.2707               0.6636               0.7483
## relevel(crop, "1")4  relevel(crop, "1")5
##              1.2252               0.9075
```

```r
fit2
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula: use ~ d_cover + (1 | animal_id)
##    Data: data_1
##       AIC       BIC    logLik  deviance  df.resid
##  31649.75  31674.66 -15821.88  31643.75     29803
## Random effects:
##  Groups    Name        Std.Dev.
##  animal_id (Intercept) 0.345
## Number of obs: 29806, groups:  animal_id, 14
## Fixed Effects:
## (Intercept)      d_cover
##      1.1655      -0.3278
```

```r
fit3
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula: use ~ d_roads + (1 | animal_id)
##    Data: data_1
##       AIC       BIC    logLik  deviance  df.resid
##  32194.01  32218.92 -16094.00  32188.01     29803
## Random effects:
##  Groups    Name        Std.Dev.
##  animal_id (Intercept) 0.2799
## Number of obs: 29806, groups:  animal_id, 14
## Fixed Effects:
## (Intercept)      d_roads
##     1.14853      0.05144
```

```
fit4
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula: use ~ d_cover + d_roads + (1 | animal_id)
##    Data: data_1
##       AIC       BIC    logLik  deviance  df.resid
##  31651.72  31684.93 -15821.86  31643.72     29802
## Random effects:
##  Groups    Name        Std.Dev.
##  animal_id (Intercept) 0.3457
## Number of obs: 29806, groups:  animal_id, 14
## Fixed Effects:
## (Intercept)      d_cover      d_roads
##    1.165420    -0.328110    -0.002653
```

```
fit5
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula: use ~ 1 | animal_id
##    Data: data_1
##       AIC       BIC    logLik  deviance  df.resid
##  32202.55  32219.16 -16099.28  32198.55     29804
## Random effects:
##  Groups    Name        Std.Dev.
##  animal_id (Intercept) 0.2867
## Number of obs: 29806, groups:  animal_id, 14
## Fixed Effects:
## (Intercept)
##       1.147
```

```
AIC(fit1,fit2,fit3,fit4,fit5)
```

```
##      df      AIC
## fit1  6 31610.18
## fit2  3 31649.75
## fit3  3 32194.01
## fit4  4 31651.72
## fit5  2 32202.55
```

```
mynames <- paste("fit", as.character(1:5), sep = "")
myaicc <- aictab(list(fit1,fit2,fit3,fit4,fit5), modnames = mynames)
print(myaicc, LL = FALSE)
```

```
##
```

```
## Model selection based on AICc:
##
##      K     AICc Delta_AICc AICcWt Cum.Wt
## fit1 6 31610.18       0.00      1      1
## fit2 3 31649.75      39.57      0      1
## fit4 4 31651.73      41.54      0      1
## fit3 3 32194.01     583.83      0      1
## fit5 2 32202.55     592.37      0      1
```

8. Our top model (fit 1) has all the support in this case indicating that during winter 2012 the mule deer were selecting for each habitat over sunflower. Considering sunflower is not available during the winter months this makes perfect sense. Looking at parameter estimates and confidence intervals for the additional habitat categories in fit 1 we see that forest (category 4) is most selected habitat followed by shrub (category 5). This is only a simply way to look at habitat, however, we used more animals that were on the air for several years and also could look at distance to habitat instead of representing habitat as categorical data.

```
per1_se <- sqrt(diag(vcov(fit1)))
# table of estimates with 95% CI
tab_per1 <- cbind(Est = fixef(fit1), LL = fixef(fit1) - 1.96 * per1_se, UL = fixef(fit1)
  + 1.96 * per1_se)
tab_per1
```

```
##                             Est         LL        UL
## (Intercept)           0.2706915 0.08792356 0.4534594
## relevel(crop, "1")2 0.6636420 0.54049466 0.7867894
## relevel(crop, "1")3 0.7482707 0.61547197 0.8810694
## relevel(crop, "1")4 1.2251593 1.12604249 1.3242761
## relevel(crop, "1")5 0.9074680 0.81258760 1.0023484
```

10. We can then create a surface of predictions from our top model indicating where in our study site we might find the highest probability of use. To do this, we need to export a text file from our “layer” created in Exercise 8.3.

```
layer1 <- read.table("layer1.txt",sep=",")
names(layer1) = c("crop", "d_cover", "d_roads","x", "y")
#Need to standardize the raw distance rasters first to match what we modeled
layer1[,2:3]=scale(layer1[,2:3],scale=TRUE)
head(layer1)
```

```
##   crop  d_cover  d_roads        x       y
## 1    2 7.746856 4.036468 -1143750 1734450
## 2    2 7.724088 3.995143 -1143720 1734450
## 3    2 7.704306 3.954485 -1143690 1734450
## 4    2 7.687531 3.914510 -1143660 1734450
## 5    2 7.673780 3.875234 -1143630 1734450
## 6    2 7.663070 3.836676 -1143600 1734450
```

```
layer1$crop <- as.factor(layer1$crop)

# predictions based on best model
predictions = predict(fit1, newdata=layer1, re.form=NA, type="link")#based on the scale of the
  #linear predictors
predictions = exp(predictions)
range(predictions)
```

```
## [1] 1.310871 4.463132
```

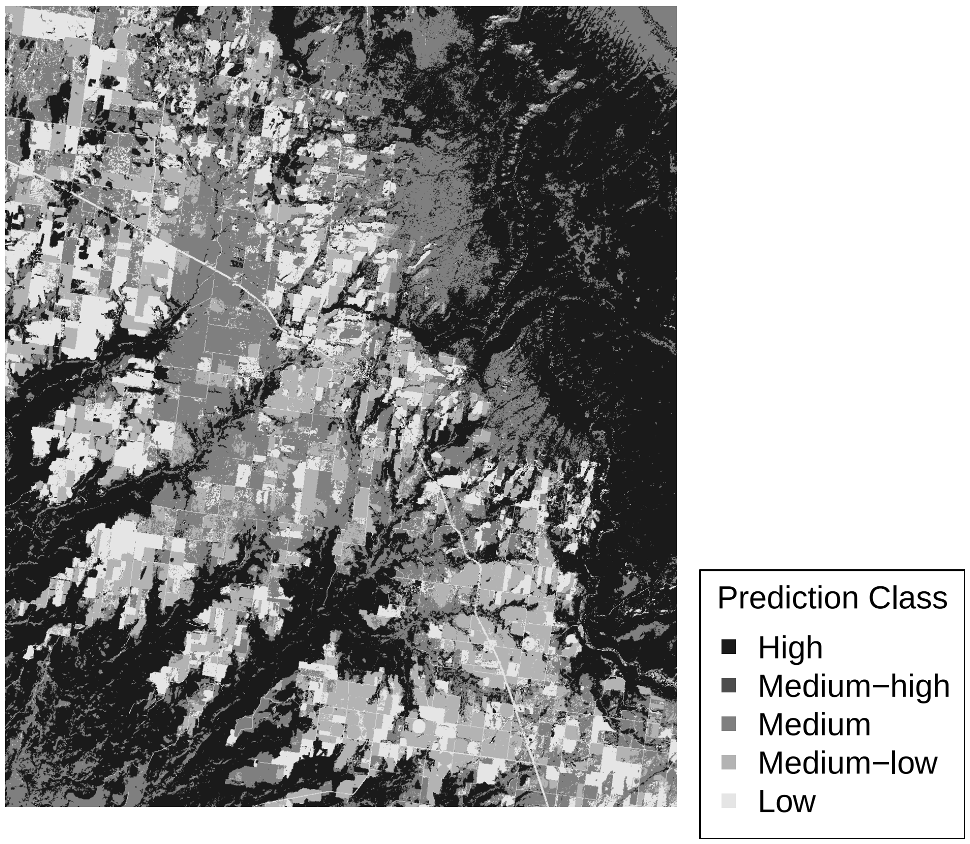
```r
#------------------------------------------------------------------------------------
# create Ascii grid of raw predictions if needed
layer1$predictions = predictions
#preds = layer1
#preds = SpatialPixelsDataFrame(points=preds[c("x", "y")], data=preds)
#preds = as(preds, "SpatialGridDataFrame")
#names(preds)
#writeAsciiGrid(preds, "predictions.asc", attr=13)#attr should be column number for 'predictions'

#------------------------------------------------------------------------------------
# assign each cell or habitat unit to a 'prediction class'.
# classes have (nearly) equal area, if the cells or habitat units have equal areas.
# output is a vector of class assignments (higher is better).
F.prediction.classes <- function(raw.prediction, n.classes){
  # raw.prediction = vector of raw (or scaled) RSF predictions
  # n.classes = number of prediction classes.
  pred.quantiles = quantile(raw.prediction, probs=seq(1/n.classes, 1-1/n.classes,
  by=1/n.classes))
  ans = rep(n.classes, length(raw.prediction))
  for(i in (n.classes-1):1){
    ans[raw.prediction < pred.quantiles[i]] = i
  }
  return(ans)
}

layer1$prediction.class = F.prediction.classes(layer1$predictions, 5)
table(layer1$prediction.class)
```

```
##
##      1      2      3      5
## 307690 261406 640591 872463
```

```r
###############################################
# create map of RSF prediction classes in R
m = SpatialPixelsDataFrame(points = layer1[c("x", "y")], data=layer1)
names(m)
```

```
## [1] "crop"             "d_cover"          "d_roads"          "x"
## [5] "y"                "predictions"      "prediction.class"
```

```r
par(mar=c(0,0,0,0))
image(m, attr=7, col=c("grey90", "grey70", "grey50", "grey30", "grey10"))
par(lend=1)
legend("bottomright", col=rev(c("grey90", "grey70", "grey50", "grey30", "grey10")),
       legend=c("High", "Medium-high", "Medium", "Medium-low", "Low"),
       title="Prediction Class", pch=15, cex=1.0,bty != "n", bg="white")
```



```
# create Ascii grid of prediction classes
#m = as(m, "SpatialGridDataFrame")
#names(m)
#writeAsciiGrid(m, "PredictionClassess.asc", attr=7)
```