A Brief Tutorial on Maxent

Author(s): Steven Phillips

Source: *Lessons in Conservation*, Vol. 3, pp. 108-135

Published by: Network of Conservation Educators and Practitioners, Center for Biodiversity and Conservation, American Museum of Natural History

Stable URL: ncep.amnh.org/linc/

A Brief Tutorial on Maxent

Steven Phillips

INTRODUCTION

This tutorial gives a basic introduction to use of the Maxent program for maximum entropy modelling of species' geographic distributions, written by Steven Phillips, Miro Dudik, and Rob Schapire, with support from AT&T Labs-Research, Princeton University, and the Center for Biodiversity and Conservation, American Museum of Natural History. For more details on the theory maximum entropy modeling as well as a description of the data used and the main types of statistical analysis used here, see:

Phillips, S.J., R.P. Anderson, and R.E. Schapire. 2006. Maximum entropy modeling of species geographic distributions. Ecological Modelling 190: 3-4 pp 231-259.

A second paper describing more recently-added features of the Maxent software is:

Phillips, S.J. and M. Dudik. 2008. Modeling of species distributions with Maxent: new extensions and a comprehensive evaluation. Ecography (31)2: 161-175.

The environmental data we will use consist of climatic and elevational data for South America, together with a potential vegetation layer. Our sample species will be *Bradypus variegatus*, the brown-throated three-toed sloth. This tutorial will assume that all the data files are located in the same directory as the Maxent program files; otherwise you will need to use the path (e.g., c:\data\maxent\tutorial) in front of the file names used here.

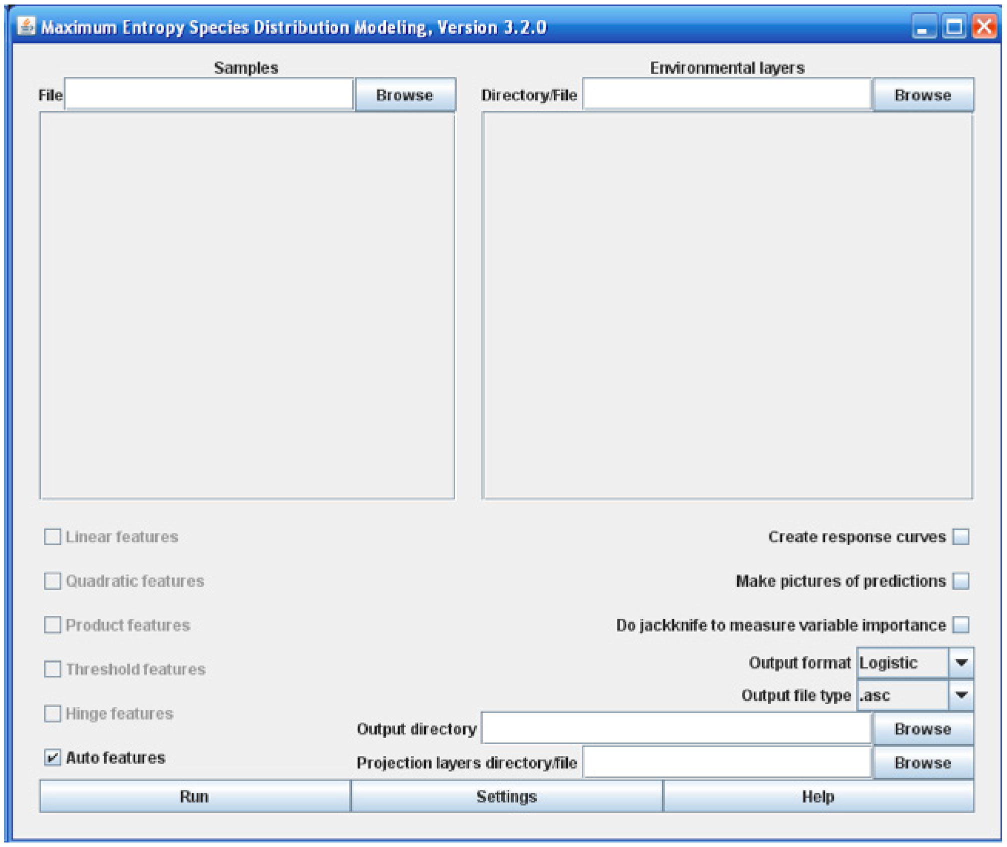
GETTING STARTED

Downloading

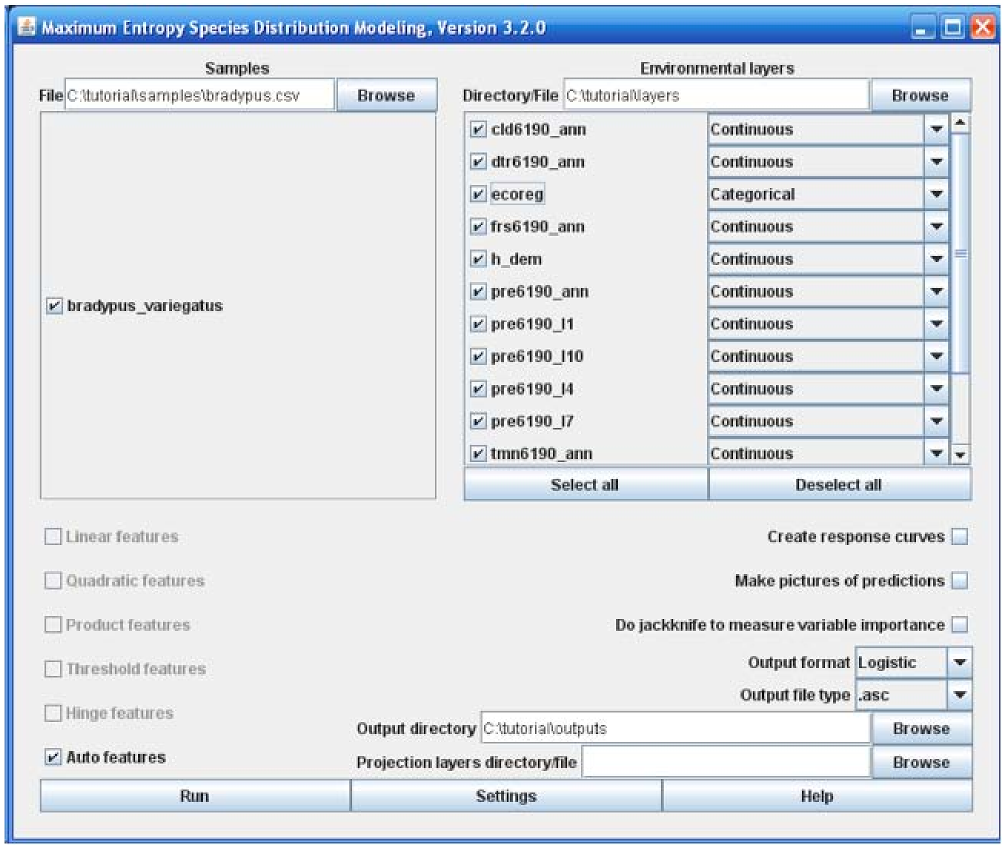
The software consists of a jar file, maxent.jar, which can be used on any computer running Java version 1.4 or later. Maxent can be downloaded, along with associated literature, from www.cs.princeton.edu/~schapire/maxent; the Java runtime environment can be obtained from java.sun.com/javase/downloads. If you are using Microsoft Windows (as we assume here), you should also download the file maxent.bat, and save it in the same directory as maxent.jar. The website has a file called "readme.txt", which contains instructions for installing the program on your computer.

Firing up

If you are using Microsoft Windows, simply click on the file maxent.bat. Otherwise, enter "java -mx512m -jar maxent.jar" in a command shell (where "512" can be replaced by the megabytes of memory you want made available to the program). The following screen will appear:



To perform a run, you need to supply a file containing presence localities ("samples"), a directory containing environmental variables, and an output directory. In our case, the presence localities are in the file "samples\bradypus.csv", the environmental layers are in the directory "layers", and the outputs are going to go in the directory "outputs". You can enter these locations by hand, or browse for them. While browsing for the environmental variables, remember that you are looking for the directory that contains them – you don't need to browse down to the files in the directory. After entering or browsing for the files for *Bradypus*, the program looks like this:



Samples File: C:\tutorial\samples\bradypus.csv

✓ bradypus_variegatus

Environmental layers Directory/File: C:\tutorial\layers

✓ cld6190_ann	Continuous
✓ dtr6190_ann	Continuous
✓ ecoreg	Categorical
✓ frs6190_ann	Continuous
✓ h_dem	Continuous
✓ pre6190_ann	Continuous
✓ pre6190_l1	Continuous
✓ pre6190_l10	Continuous
✓ pre6190_l4	Continuous
✓ pre6190_l7	Continuous
✓ tmn6190_ann	Continuous

☐ Linear features
☐ Quadratic features
☐ Product features
☐ Threshold features
☐ Hinge features
✓ Auto features

Create response curves ☐
Make pictures of predictions ☐
Do jackknife to measure variable importance ☐
Output format: Logistic
Output file type: .asc
Output directory: C:\tutorial\outputs
Projection layers directory/file:

The file "samples\bradypus.csv" contains the presence localities in .csv format. The first few lines are as follows:

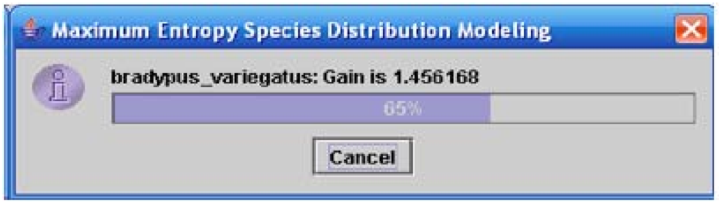
```
species,longitude,latitude
bradypus_variegatus,-65.4,-10.3833
bradypus_variegatus,-65.3833,-10.3833
bradypus_variegatus,-65.1333,-16.8
bradypus_variegatus,-63.6667,-17.45
bradypus_variegatus,-63.85,-17.4
```

There can be multiple species in the same samples file, in which case more species would appear in the panel, along with *Bradypus*. Coordinate systems other than latitude and longitude can be used provided that the samples file and environmental layers use the same coordinate system. The "x" coordinate (longitude, in our case) should come before the "y" coordinate (latitude) in the samples file. If the presence data has duplicate records (multiple records for the same species in the same grid cell), the duplicates can be removed by clicking on the "Settings" button and selecting "Delete duplicates".

The directory "layers" contains a number of ascii raster grids (in ESRI's .asc format), each of which describes an environmental variable. The grids must all have the same geographic bounds and cell size (i.e. all the ascii file headings must match each other perfectly). One of our variables, "ecoreg", is a categorical variable describing potential vegetation classes. The categories must be indicated by numbers, rather than letters or words. You must tell the program which variables are categorical, as has been done in the picture above.

DOING A RUN

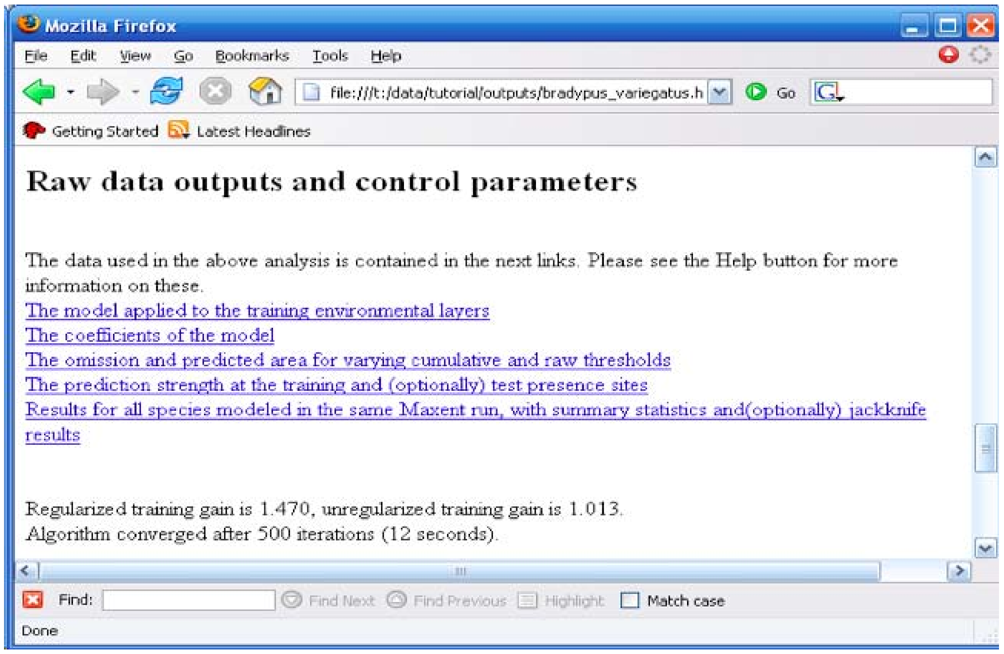
Simply press the "Run" button. A progress monitor describes the steps being taken. After the environmental layers are loaded and some initialization is done, progress towards training of the maxent model is shown like this:


Maximum Entropy Species Distribution Modeling

bradypus_variegatus: Gain is 1.456168

The gain is closely related to deviance, a measure of goodness of fit used in generalized additive and generalized linear models. It starts at 0 and increases towards an asymptote during the run. During this process, Maxent is generating a probability distribution over pixels in the grid, starting from the uniform distribution and repeatedly improving the fit to the data. The gain is defined as the average log probability of the presence samples, minus a constant that makes the uniform distribution have zero gain. At the end of the run, the gain indicates how closely the model is concentrated around the presence samples; for example, if the gain is 2, it means that the average likelihood of the presence samples is exp(2) ≈ 7.4 times higher than that of a random background pixel. Note that Maxent isn't directly calculating "probability of occurrence". The probability it assigns to each pixel is typically very small, as the values must sum to 1 over all the pixels in the grid (though we return to this point when we compare output formats).

The run produces multiple output files, of which the most important for analyzing your model is an html file called "bradypus.html". Part of this file gives pointers to the other outputs, like this:



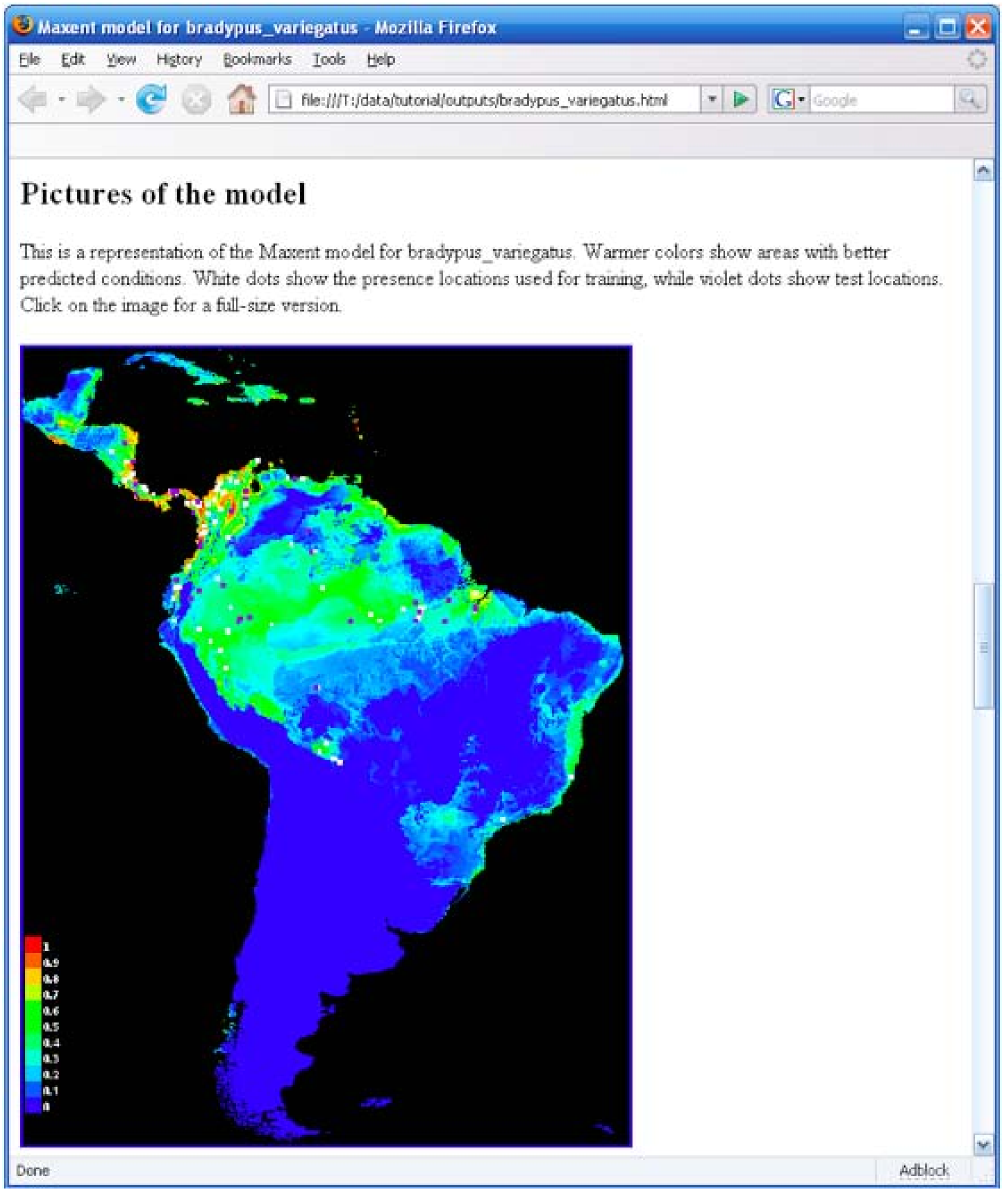
Raw data outputs and control parameters

The data used in the above analysis is contained in the next links. Please see the Help button for more information on these.
The model applied to the training environmental layers
The coefficients of the model
The omission and predicted area for varying cumulative and raw thresholds
The prediction strength at the training and (optionally) test presence sites
Results for all species modeled in the same Maxent run, with summary statistics and(optionally) jackknife results

Regularized training gain is 1.470, unregularized training gain is 1.013.
Algorithm converged after 500 iterations (12 seconds).

LOOKING AT A PREDICTION

To see what other (more interesting) output there can be in bradpus.html, we will turn on a couple of options and rerun the model. Press the "Make pictures of predictions" button, then click on "Settings", and type "25" in the "Random test percentage" entry. Then, press the "Run" button again. After the run completes, the file bradypus.html contains a picture like this:


Maxent model for bradypus_variegatus - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///T:/data/tutorial/outputs/bradypus_variegatus.html
Google
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Done
Adblock

Pictures of the model

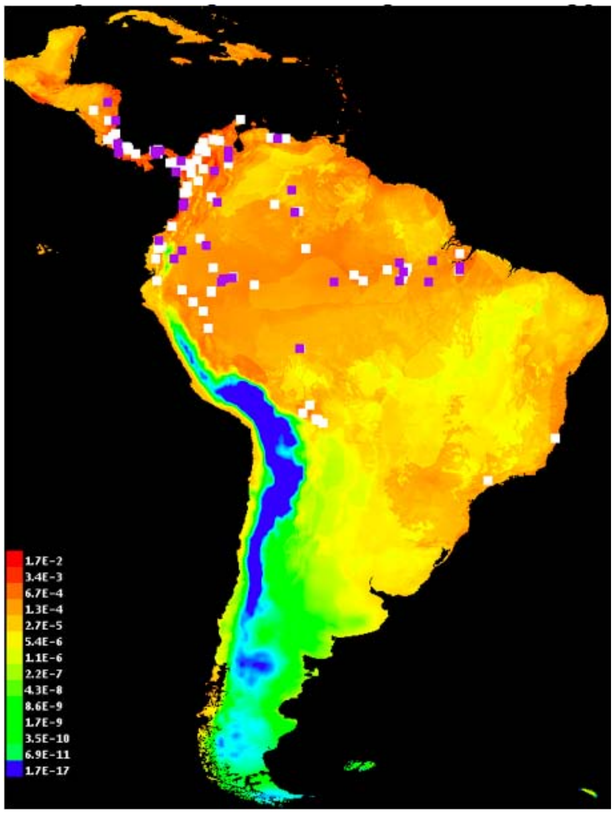
This is a representation of the Maxent model for bradypus_variegatus. Warmer colors show areas with better predicted conditions. White dots show the presence locations used for training, while violet dots show test locations. Click on the image for a full-size version.

The image uses colors to indicate predicted probability that conditions are suitable, with red indicating high probability of suitable conditions for the species, green indicating conditions typical of those where the species is found, and lighter shades of blue indicating low predicted probability of suitable conditions. For *Bradypus*, we see that suitable conditions are predicted to be highly probable through most of lowland Central America, wet lowland areas of northwestern South America, the Amazon basin, Caribean islands, and much of the Atlantic forests in south-eastern Brazil. The file pointed to is an image file (.png) that you can just click on (in Windows) or open in most image processing software. If you want to copy these images, or want to open them with other software, you will find the .png files in the directory called "plots" that has been created as an output during the run.

The test points are a random sample taken from the species presence localities. The same random sample is used each time you run Maxent on the same data set, unless you select the "random seed" option on the settings panel. Alternatively, test data for one or more species can be provided in a separate file, by giving the name of a "Test sample file" in the Settings panel.

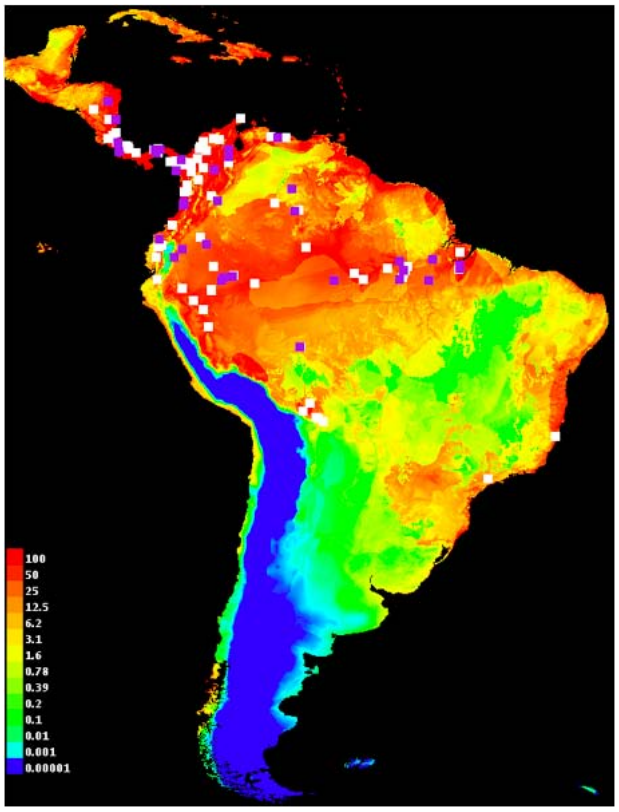
OUTPUT FORMATS

Maxent supports three output formats for model values: raw, cumulative, and logistic. First, the raw output is just the Maxent exponential model itself. Second, the cumulative value corresponding to a raw value of r is the percentage of the Maxent distribution with raw value at most r. Cumulative output is best interpreted in terms of predicted omission rate: if we set a cumulative threshold of c, the resulting binary prediction would have omission rate c% on samples drawn from the Maxent distribution itself, and we can predict a similar omission rate for samples drawn from the species distribution. Third, if c is the exponential of the entropy of the maxent distribution, then the logistic value corresponding to a raw value of r is $c{\cdot}r/(1+c{\cdot}r)$. This is a logistic function, because the raw value is an exponential function of the environmental variables. The three output formats are all monotonically related, but they are scaled differently, and have different interpretations. The default output is logistic, which is the easiest to conceptualize: it gives an estimate between 0 and 1 of probability of presence. Note that probability of presence depends on details of the sampling design, such as the plot size and (for vagile organisms) observation time; logistic output estimates probability of presence assuming that the sampling design is such that typical presence localities have probability of presence of about 0.5. The picture of the *Bradypus* model above uses the logistic format. In comparison, using the raw format gives the following picture:



Note that we have used a logarithmic scale for the colors. A linear scale would be mostly blue, with a few red pixels (you can verify this by deselecting "Logscale pictures" on the Settings panel) since the raw format typically gives a small number of sites relatively large values – this can be thought of as an artifact of the raw output being given by an exponential distribution.

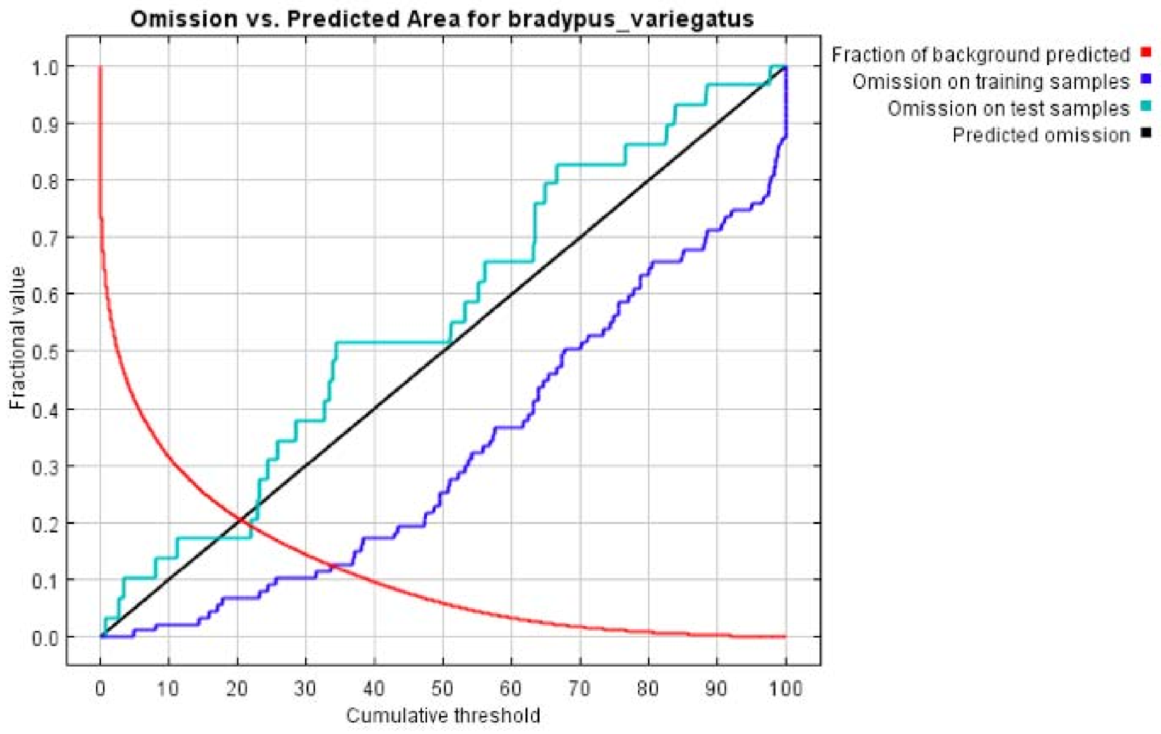
Using the cumulative output format gives the following picture:



As with the raw output, we have used a logarithmic scale for coloring the picture in order to emphasize differences between smaller values. Cumulative output can be interpreted as predicting suitable conditions for the species above a threshold in the approximate range of 1-20 (or yellow through orange, in this picture), depending on the level of predicted omission that is acceptable for the application.

STATISTICAL ANALYSIS

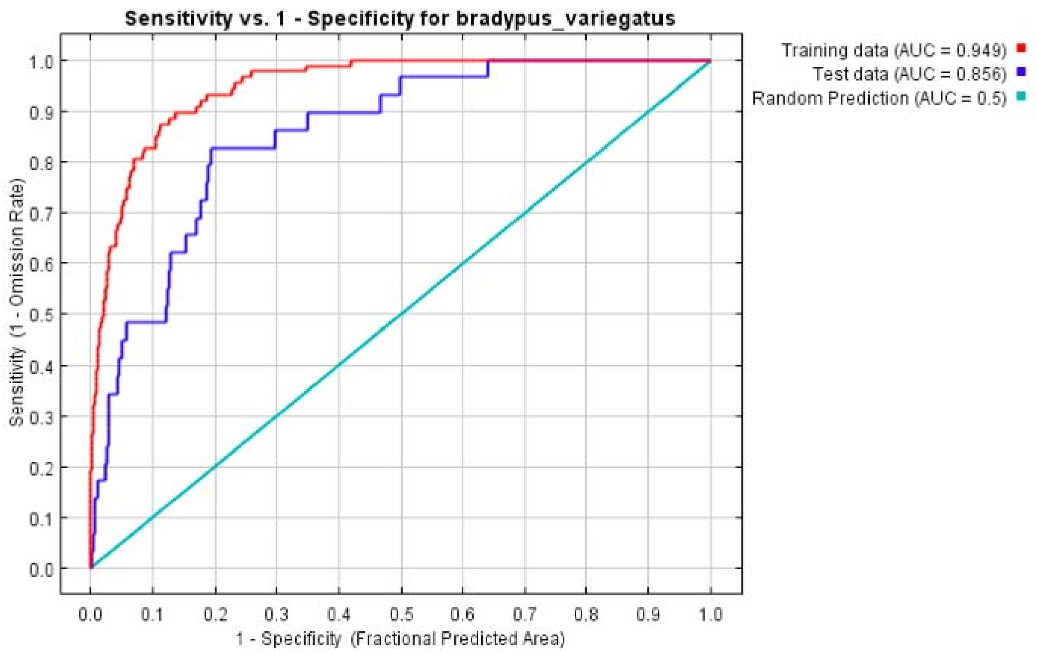
The "25" we entered for "random test percentage" told the program to randomly set aside 25% of the sample records for testing. This allows the program to do some simple statistical analysis. Much of the analysis made use of a threshold to make a binary prediction, with suitable conditions predicted above the threshold and unsuitable below. The first plot shows how testing and training omission and predicted area vary with the choice of cumulative threshold, as in the following graph:



Here, we see that the omission on test samples is a very good match to the predicted omission rate; the omission rate for test data drawn from the Maxent distribution itself. The predicted omission rate is a straight line, by definition of the cumulative output format. In some situations, the test omission line lies well below the predicted omission line: a common reason is that the test and training data are not independent, for example if they derive from the same spatially-autocorrelated presence data.

The next plot gives the receiver operating curve for both training and test data, shown below. The area under the ROC curve (AUC) is also given here; if test data are available, the standard error of the AUC on the test data is given later on in the web page.

If you use the same data for training and for testing then the red and blue lines will be identical. If you split your data into two partitions, one for training and one for testing

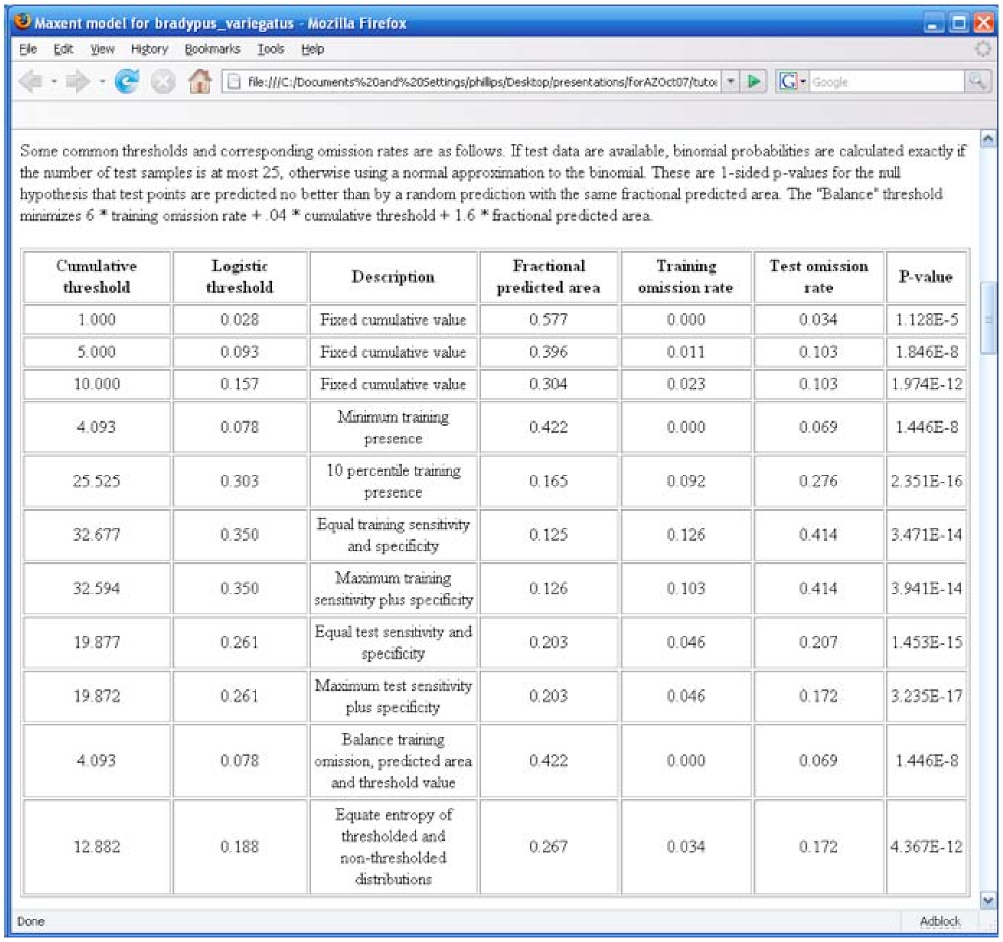


it is normal for the red (training) line to show a higher AUC than the blue (testing) line. The red (training) line shows the "fit" of the model to the training data. The blue (testing) line indicates the fit of the model to the testing data, and is the real test of the models predictive power. The turquoise line shows the line that you would expect if your model was no better than random. If the blue line (the test line) falls below the turquoise line then this indicates that your model performs worse than a random model would. The further towards the top left of the graph that the blue line is, the better the model is at predicting the presences contained in the test sample of the data. For more detailed information on the AUC statistic a good starting reference is:

Fielding, A.H. and J.F. Bell. 2007. A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24(1): 38-49.

Because we have only occurrence data and no absence data, "fractional predicted area" (the fraction of the total study area predicted present) is used instead of the more standard commission rate (fraction of absences predicted present). For more discussion of this choice, see the paper in *Ecological Modelling* mentioned on page 1 of this tutorial. It is important to note that AUC values tend to be higher for species with narrow ranges, relative to the study area described by the environmental data. This does not necessarily mean that the models are better; instead this behavior is an artifact of the AUC statistic.

If test data are available, the program automatically calculates the statistical significance of the prediction, using a binomial test of omission. For *Bradypus*, this gives:


Maxent model for bradypus_variegatus - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phillips/Desktop/presentations/forAZOct07/tuto
Google
Done
Adblock

Some common thresholds and corresponding omission rates are as follows. If test data are available, binomial probabilities are calculated exactly if the number of test samples is at most 25, otherwise using a normal approximation to the binomial. These are 1-sided p-values for the null hypothesis that test points are predicted no better than by a random prediction with the same fractional predicted area. The "Balance" threshold minimizes 6 * training omission rate + .04 * cumulative threshold + 1.6 * fractional predicted area.

Cumulative threshold	Logistic threshold	Description	Fractional predicted area	Training omission rate	Test omission rate	P-value
1.000	0.028	Fixed cumulative value	0.577	0.000	0.034	1.128E-5
5.000	0.093	Fixed cumulative value	0.396	0.011	0.103	1.846E-8
10.000	0.157	Fixed cumulative value	0.304	0.023	0.103	1.974E-12
4.093	0.078	Minimum training presence	0.422	0.000	0.069	1.446E-8
25.525	0.303	10 percentile training presence	0.165	0.092	0.276	2.351E-16
32.677	0.350	Equal training sensitivity and specificity	0.125	0.126	0.414	3.471E-14
32.594	0.350	Maximum training sensitivity plus specificity	0.126	0.103	0.414	3.941E-14
19.877	0.261	Equal test sensitivity and specificity	0.203	0.046	0.207	1.453E-15
19.872	0.261	Maximum test sensitivity plus specificity	0.203	0.046	0.172	3.235E-17
4.093	0.078	Balance training omission, predicted area and threshold value	0.422	0.000	0.069	1.446E-8
12.882	0.188	Equate entropy of thresholded and non-thresholded distributions	0.267	0.034	0.172	4.367E-12

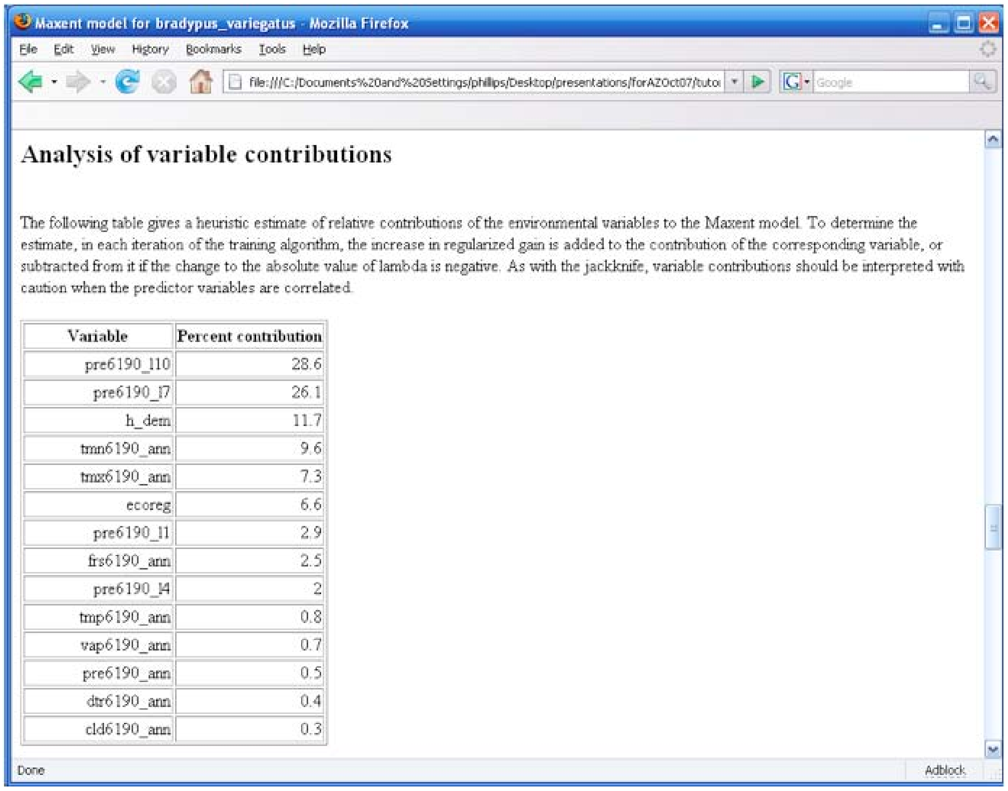
For more detailed information on the binomial statistic, see the *Ecological Modelling* paper mentioned above.

WHICH VARIABLES MATTER MOST?

A natural application of species distribution modeling is to answer the question, which variables matter most for the species being modeled? There is more than one way to answer this question; here we outline the possible ways in which Maxent can be used to address it.

While the Maxent model is being trained, we can keep track of which environmental variables are making the greatest contribution to the model. Each step of the Maxent algorithm increases the gain of the model by modifying the coefficient for a single fea–

ture; the program assigns the increase in the gain to the environmental variable(s) that the feature depends on. Converting to percentages at the end of the training process, we get the following table:


Maxent model for bradypus_variegatus - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phillips/Desktop/presentations/forAZOct07/tuto
Google
Done
Adblock

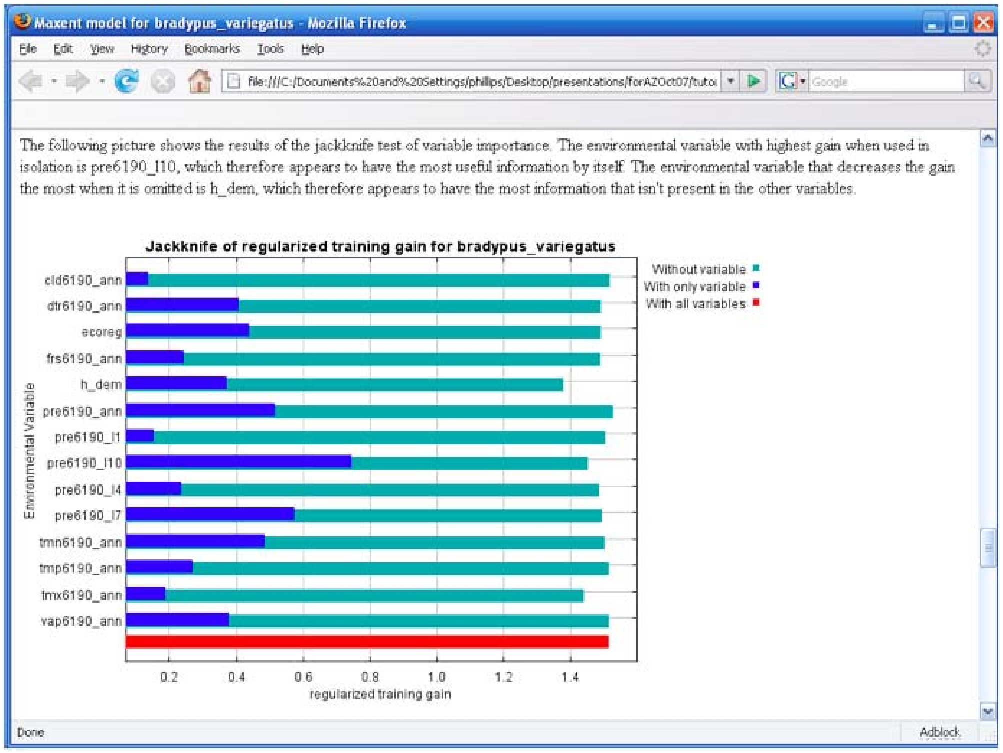
Analysis of variable contributions

The following table gives a heuristic estimate of relative contributions of the environmental variables to the Maxent model. To determine the estimate, in each iteration of the training algorithm, the increase in regularized gain is added to the contribution of the corresponding variable, or subtracted from it if the change to the absolute value of lambda is negative. As with the jackknife, variable contributions should be interpreted with caution when the predictor variables are correlated.

Variable	Percent contribution
pre6190_l10	28.6
pre6190_l7	26.1
h_dem	11.7
tmn6190_ann	9.6
tmx6190_ann	7.3
ecoreg	6.6
pre6190_l1	2.9
frs6190_ann	2.5
pre6190_l4	2
tmp6190_ann	0.8
vap6190_ann	0.7
pre6190_ann	0.5
dtr6190_ann	0.4
cld6190_ann	0.3

These percent contribution values are only heuristically defined: they depend on the particular path that the Maxent code uses to get to the optimal solution, and a different algorithm could get to the same solution via a different path, resulting in different percent contribution values. In addition, when there are highly correlated environmental variables, the percent contributions should be interpreted with caution. In our *Bradypus* example, annual precipitation is highly correlated with October and July precipitation. Although the above table shows that Maxent used the October precipitation variable more than any other, and hardly used annual precipitation at all, this does not necessarily imply that October precipitation is far more important to the species than annual precipitation.

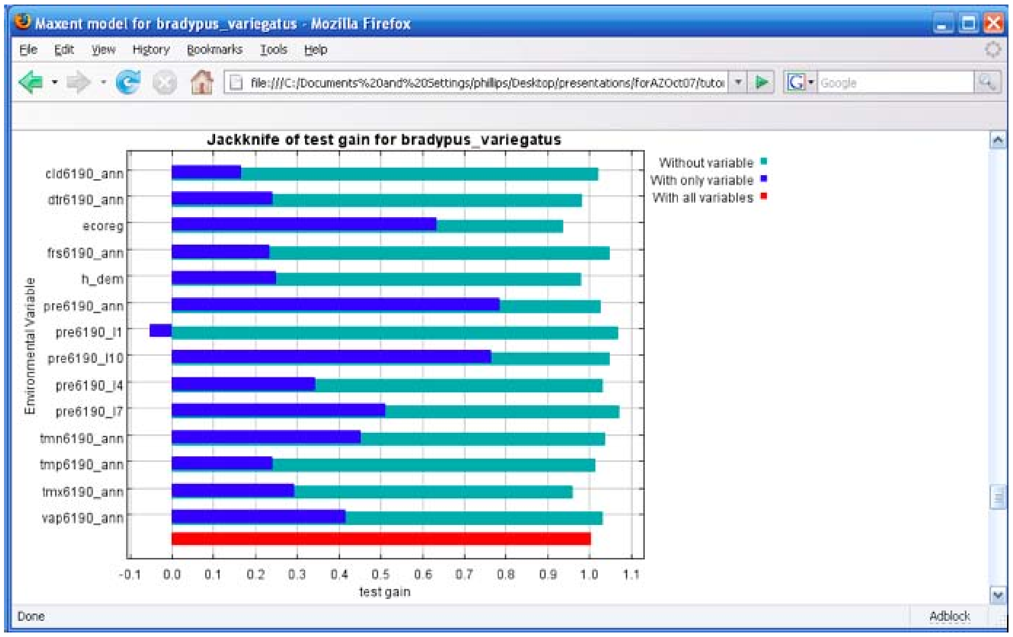
To get alternate estimates of which variables are most important in the model, we can also run a jackknife test by selecting the "Do jackknife to measure variable important" checkbox. When we press the "Run" button again, a number of models are created. Each variable is excluded in turn, and a model created with the remaining variables. Then a model is created using each variable in isolation. In addition, a model is created using all variables, as before. The results of the jackknife appear in the "bradypus.html" files in three bar charts, and the first of these is shown below.

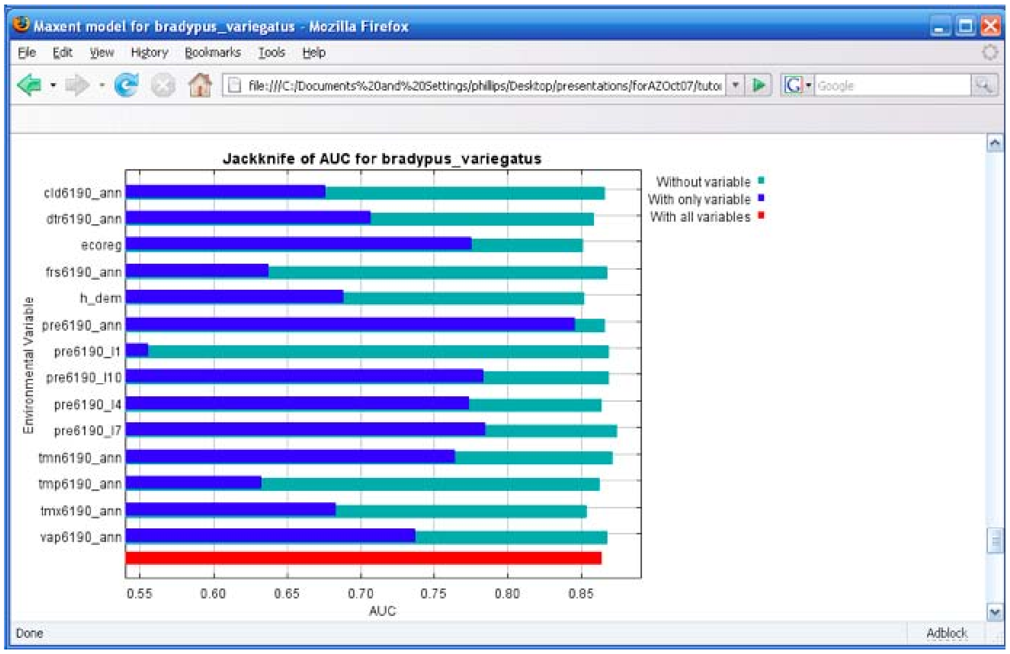

Maxent model for bradypus_variegatus - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phillips/Desktop/presentations/forAZOct07/tuto
Google
Jackknife of regularized training gain for bradypus_variegatus
cld6190_ann
dtr6190_ann
ecoreg
frs6190_ann
h_dem
pre6190_ann
pre6190_l1
pre6190_l10
pre6190_l4
pre6190_l7
tmn6190_ann
tmp6190_ann
tmx6190_ann
vap6190_ann
Environmental Variable
0.2
0.4
0.6
0.8
1.0
1.2
1.4
regularized training gain
Without variable
With only variable
With all variables
Done
Adblock

The following picture shows the results of the jackknife test of variable importance. The environmental variable with highest gain when used in isolation is pre6190_l10, which therefore appears to have the most useful information by itself. The environmental variable that decreases the gain the most when it is omitted is h_dem, which therefore appears to have the most information that isn't present in the other variables.

We see that if Maxent uses only pre6190_l1 (average January rainfall) it achieves almost no gain, so that variable is not (by itself) useful for estimating the distribution of *Bradypus*. On the other hand, October rainfall (pre6190_l10) allows a reasonably good fit to the training data. Turning to the lighter blue bars, it appears that no variable contains a substantial amount of useful information that is not already contained in the other variables, because omitting each variable in turn did not decrease the training gain considerably.

The bradypus.html file has two more jackknife plots, which use either test gain or AUC in place of training gain, shown below.





Comparing the three jackknife plots can be very informative. The AUC plot shows that annual precipitation (pre6190_ann) is the most effective single variable for predicting the distribution of the occurrence data that was set aside for testing, when predictive performance is measured using AUC, even though it was hardly used by the model built using all variables. The relative importance of annual precipitation also increases in the test gain plot, when compared against the training gain plot. In addition, in the test gain and AUC plots, some of the light blue bars (especially for the monthly precipitation variables) are longer than the red bar, showing that predictive performance improves when the corresponding variables are not used.

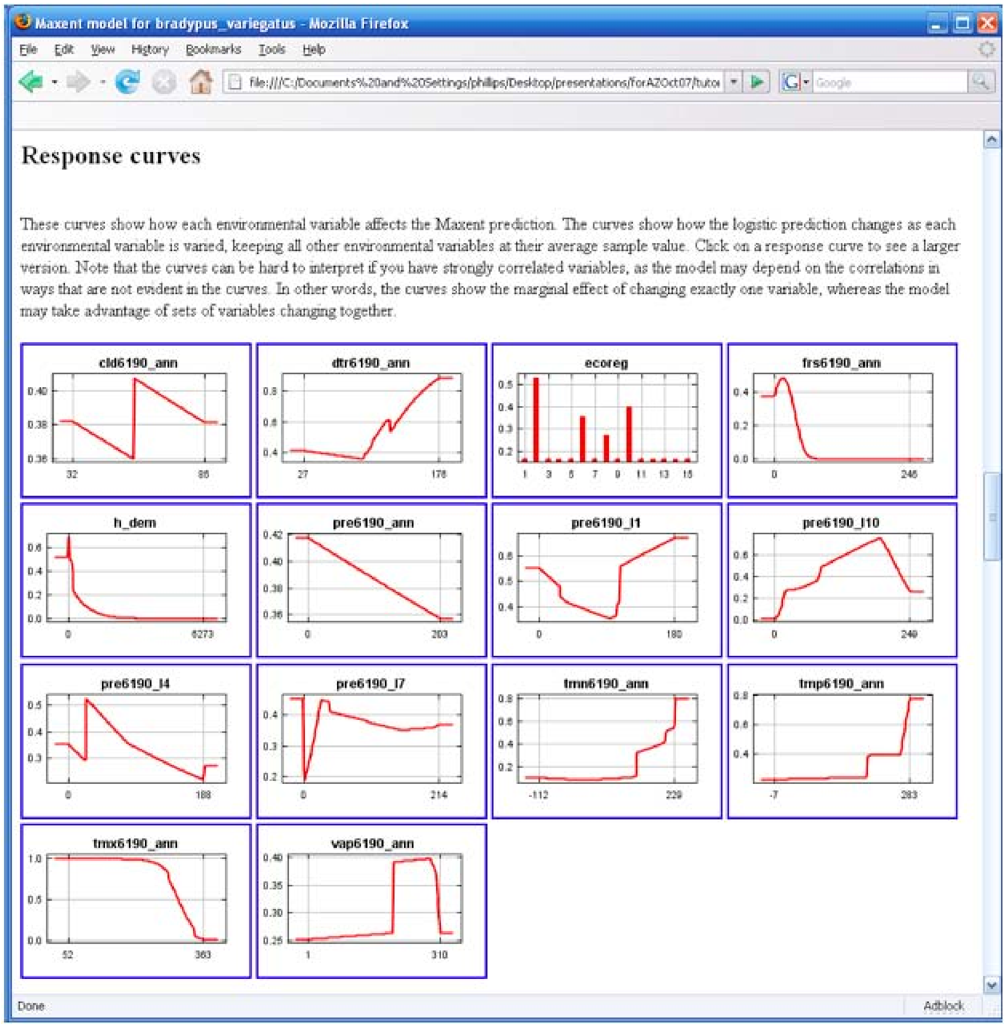
This tells us that monthly precipitation variables are helping Maxent to obtain a good fit to the training data, but the annual precipitation variable generalizes better, giving comparatively better results on the set-aside test data. Phrased differently, models made with the monthly precipitation variables appear to be less transferable. This is important if our goal is to transfer the model, for example by applying the model to future climate variables in order to estimate its future distribution under climate change. It makes sense that monthly precipitation values are less transferable: likely suitable conditions for *Bradypus* will depend not on precise rainfall values in selected months, but on the aggregate average rainfall, and perhaps on rainfall consistency or lack of extended dry periods. When we are modeling on a continental scale, there will probably be shifts in the precise timing of seasonal rainfall patterns, affecting the monthly precipitation but not suitable conditions for *Bradypus*.

In general, it would be better to use variables that are more likely to be directly relevant to the species being modeled. For example, the Worldclim website (www.worldclim.org) provides "BIOCLIM" variables, including derived variables such as "rainfall in the wettest quarter", rather than monthly values.

A last note on the jackknife outputs: the test gain plot shows that a model made only with January precipitation (pre6190_l1) results in a negative test gain. This means that the model is slightly worse than a null model (i.e., a uniform distribution) for predicting the distribution of occurrences set aside for testing. This can be regarded as more evidence that the monthly precipitation values are not the best choice for predictor variables.

HOW DOES THE PREDICTION DEPEND ON THE VARIABLES?

Now press the "Create response curves", deselect the jackknife option, and rerun the model. This results in the following section being added to the "bradypus.html" file:



Maxent model for bradypus_variegatus - Mozilla Firefox

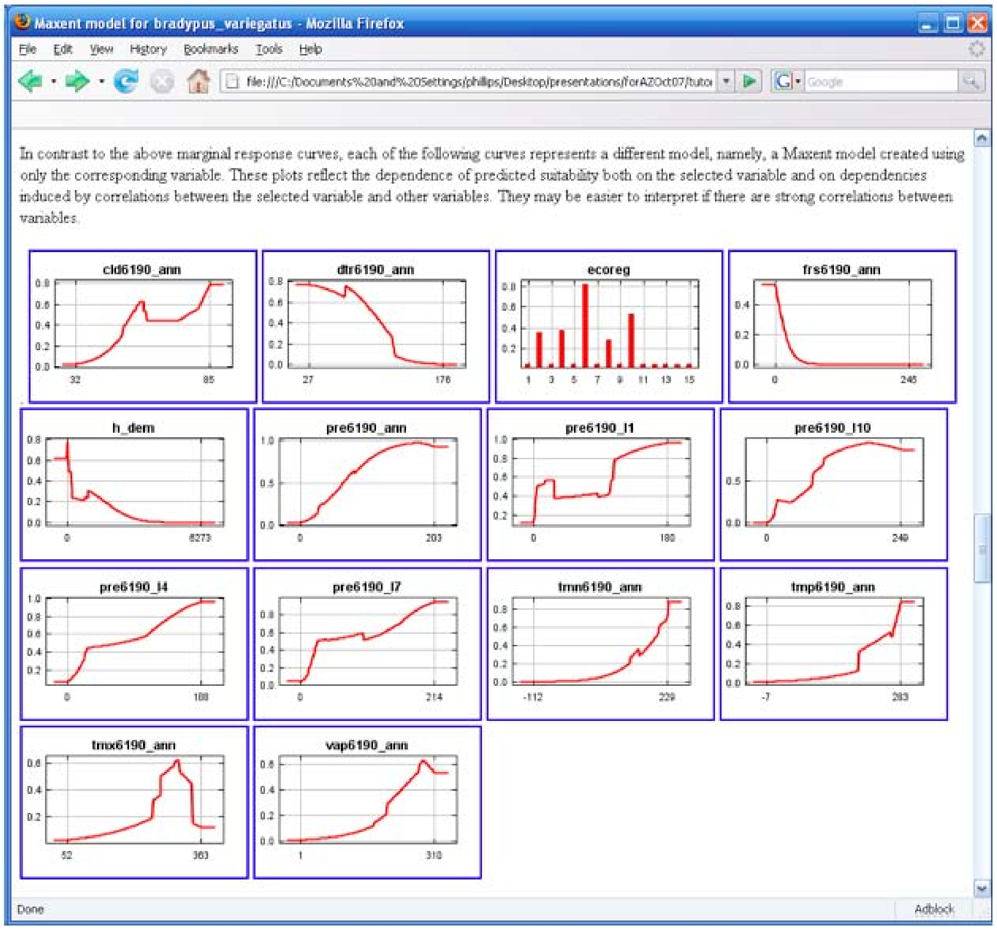
file:///C:/Documents%20and%20Settings/phillips/Desktop/presentations/forAZOct07/tuto

Response curves

These curves show how each environmental variable affects the Maxent prediction. The curves show how the logistic prediction changes as each environmental variable is varied, keeping all other environmental variables at their average sample value. Click on a response curve to see a larger version. Note that the curves can be hard to interpret if you have strongly correlated variables, as the model may depend on the correlations in ways that are not evident in the curves. In other words, the curves show the marginal effect of changing exactly one variable, whereas the model may take advantage of sets of variables changing together.

Each of the thumbnail images can be selected (by clicking on them) to obtain a more detailed plot, and if you would like to copy or open these plots with other software, the .png files can be found in the "plots" directory. Looking at vap6190_ann, we see that the response is low for values of vap6190_ann in the range 1-200, and is higher for values in the range 200-300. The value shown on the y-axis is predicted probability of suitable conditions, as given by the logistic output format, with all other variables set to their average value over the set of presence localities.

Note that if the environmental variables are correlated, as they are here, the marginal response curves can be misleading. For example, if two closely correlated variables have response curves that are near opposites of each other, then for most pixels, the combined effect of the two variables may be small. As another example, we see that predicted suitability is negatively correlated with annual precipitation (pre6190_ann), if all other variables are held fixed. In other words, once the effect of all the other variables has already been accounted for, the marginal effect of increasing annual precipitation is to decrease predicted suitability. However, annual precipitation is highly correlated with the monthly precipitation variables, so in reality we cannot easily hold the monthly values fixed while varying the annual value. The program therefore produces a second set of response curves, in which each curve is made by generating a model using only the corresponding variable, disregarding all other variables:

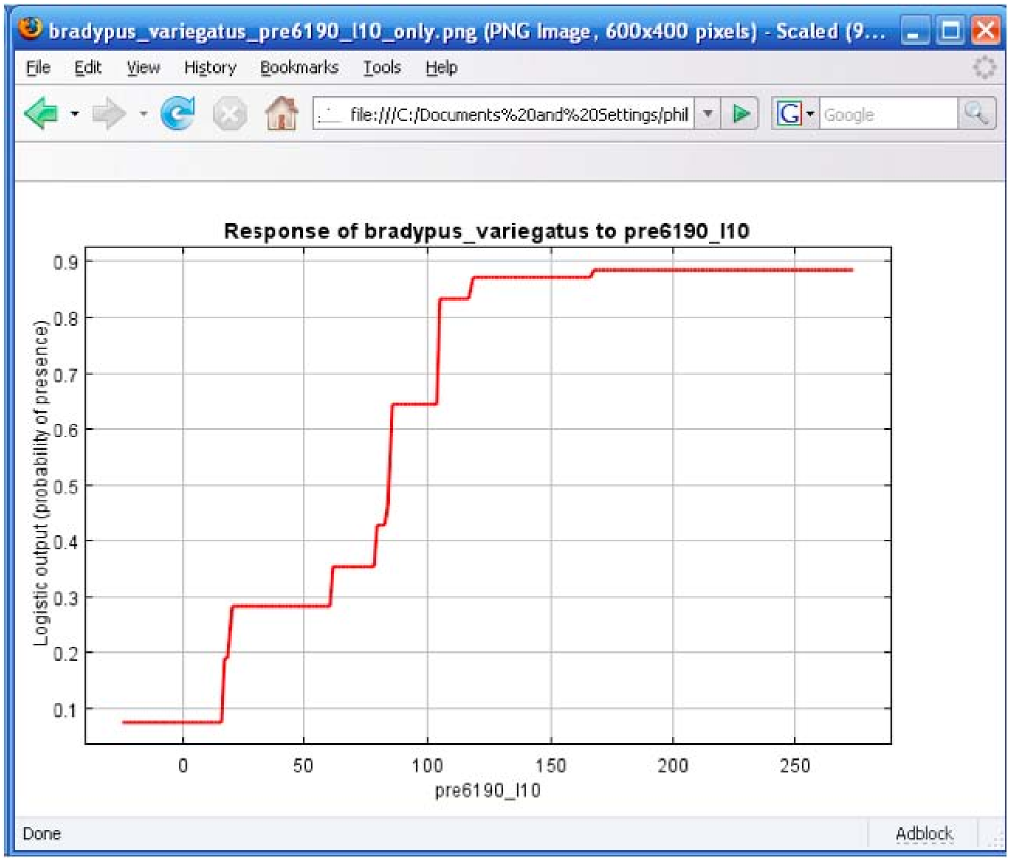

In contrast to the above marginal response curves, each of the following curves represents a different model, namely, a Maxent model created using only the corresponding variable. These plots reflect the dependence of predicted suitability both on the selected variable and on dependencies induced by correlations between the selected variable and other variables. They may be easier to interpret if there are strong correlations between variables.

Maxent model for bradypus_variegatus - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phillips/Desktop/presentations/forAZOct07/tuto
cld6190_ann
dtr6190_ann
ecoreg
frs6190_ann
h_dem
pre6190_ann
pre6190_l1
pre6190_l10
pre6190_l4
pre6190_l7
tmn6190_ann
tmp6190_ann
tmx6190_ann
vap6190_ann
Done
Adblock

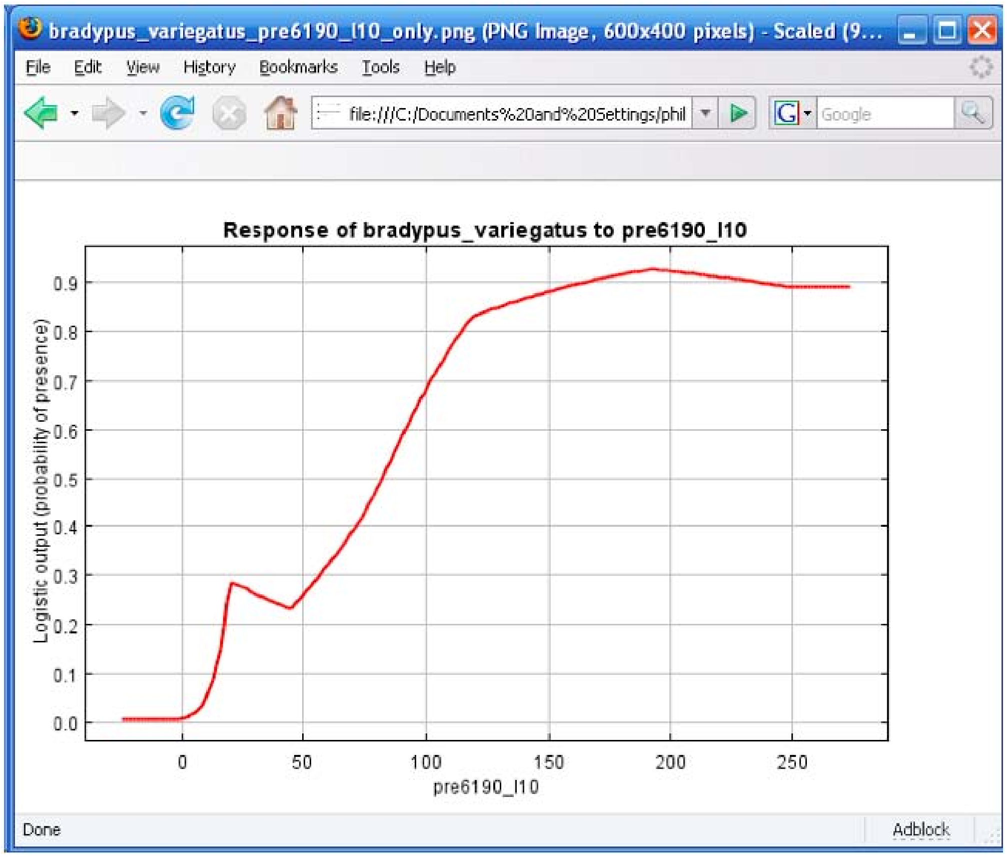
In contrast to the marginal response to annual precipitation in the first set of response curves, we now see that predicted suitability generally increases with increasing annual precipitation.

FEATURE TYPES AND RESPONSE CURVES

Response curves allow us to see the difference among different feature types. Deselect the "auto features", select "Threshold features", and press the "Run" button again. Take a look at the resulting feature profiles – you'll notice that they are all step functions, like this one for pre6190_l10:


bradypus_variegatus_pre6190_l10_only.png (PNG Image, 600x400 pixels) - Scaled (9...
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phil
Google
Response of bradypus_variegatus to pre6190_l10
Logistic output (probability of presence)
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
pre6190_l10
Done
Adblock

If the same run is done using only hinge features, the resulting feature profile looks like this:


bradypus_variegatus_pre6190_l10_only.png (PNG Image, 600x400 pixels) - Scaled (9...
File Edit View History Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/phil
Google
Response of bradypus_variegatus to pre6190_l10
Logistic output (probability of presence)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
50
100
150
200
250
pre6190_l10
Done
Adblock

The outlines of the two profiles are similar, but they differ because different feature types allow different possible shapes of response curves. The exponent in a Maxent model is a sum of features, and a sum of threshold features is always a step function, so the logistic output is also a step function (as are the raw and cumulative outputs). In comparison, a sum of hinge features is always a piece-linear function, so if only hinge features are used, the Maxent exponent is piece-wise linear. This explains the sequence of connected line segments in the second response curve above. Using all classes together (the default, given enough samples) allows many complex responses to be accurately modeled. A deeper explanation of the various feature types can be found by clicking on the "help" button.

SWD FORMAT

Another input format can be very useful, especially when your environmental grids are very large. For lack of a better name, it's called "samples with data", or just SWD. The SWD version of our *Bradypus* file, called "bradypus_swd.csv", starts like this:

```
species,longitude,latitude,cld6190_ann,dtr6190_ann,ecoreg,frs6190_ann,h_dem,pre6190_ann,pre6190_l10,pre6190_
l1,pre6190_l4,pre6190_l7,tmn6190_ann,tmp6190_ann,tmx6190_ann,vap6190_ann
bradypus_variegatus,-65.4,-10.3833,76.0,104.0,10.0,2.0,121.0,46.0,41.0,84.0,54.0,3.0,192.0,266.0,337.0,279.0
bradypus_variegatus,-65.3833,-10.3833,76.0,104.0,10.0,2.0,121.0,46.0,40.0,84.0,54.0,3.0,192.0,266.0,337.0,279.0
bradypus_variegatus,-65.1333,-16.8,57.0,114.0,10.0,1.0,211.0,65.0,56.0,129.0,58.0,34.0,140.0,244.0,321.0,221.0
bradypus_variegatus,-63.6667,-17.45,57.0,112.0,10.0,3.0,363.0,36.0,33.0,71.0,27.0,13.0,135.0,229.0,307.0,202.0
bradypus_variegatus,-63.85,-17.4,57.0,113.0,10.0,3.0,303.0,39.0,35.0,77.0,29.0,15.0,134.0,229.0,306.0,202.0
```

It can be used in place of an ordinary samples file. The difference is only that the program doesn't need to look in the environmental layers to obtain values for the variables at the sample points. The environmental layers are thus only used to get "background" pixels – pixels where the species hasn't necessarily been detected. In fact, the background pixels can also be specified in a SWD format file. The file "background.csv" contains 10,000 background data points. The first few look like this:

```
background,-61.775,6.175,60.0,100.0,10.0,0.0,747.0,55.0,24.0,57.0,45.0,81.0,182.0,239.0,300.0,232.0
background,-66.075,5.325,67.0,116.0,10.0,3.0,1038.0,75.0,16.0,68.0,64.0,145.0,181.0,246.0,331.0,234.0
background,-59.875,-26.325,47.0,129.0,9.0,1.0,73.0,31.0,43.0,32.0,43.0,10.0,97.0,218.0,339.0,189.0
background,-68.375,-15.375,58.0,112.0,10.0,44.0,2039.0,33.0,67.0,31.0,30.0,6.0,101.0,181.0,251.0,133.0
background,-68.525,4.775,72.0,95.0,10.0,0.0,65.0,72.0,16.0,65.0,69.0,133.0,218.0,271.0,346.0,289.0
```

We can run Maxent with "bradypus_swd.csv" as the samples file and "background.csv" (both located in the "swd" directory) as the environmental layers file. Try running it – you'll notice that it runs much faster, because it doesn't have to load the large environmental grids. The downside is that it can't make pictures or output grids, because it doesn't have all the environmental data. The way to get around this is to use a "projection", described below.

BATCH RUNING

Sometimes you need to generate multiple models, perhaps with slight variations in the modeling parameters or the inputs. Generation of models can be automated with command-line arguments, obviating the need to click and type repetitively at the program interface. The command line arguments can either be given from a command window (a.k.a. shell), or they can be defined in a batch file. Take a look at the file "batchExample.bat" (for example, right click on the .bat file inWindows Explorer and open it using Notepad). It contains the following line:

```
java -mx512m -jar maxent.jar environmentallayers=layers togglelayertype=ecoreg
```

```
samplesfile=samples\bradypus.csv outputdirectory=outputs redoifexists autorun
```

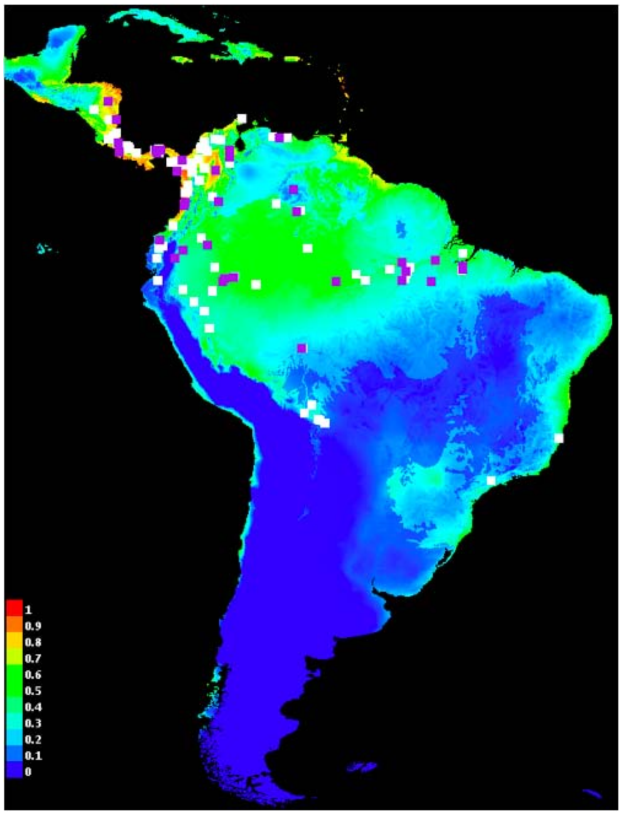
The effect is to tell the program where to find environmental layers and samples file and where to put outputs, to indicate that the ecoreg variable is categorical. The "autorun" flag tells the program to start running immediately, without waiting for the "Run" button to be pushed. Now, try double clicking on the file to see what it does.

Many aspects of the Maxent program can be controlled by command-line arguments – press the "Help" button to see all the possibilities. Multiple runs can appear in the same file, and they will simply be run one after the other. You can change the default values of most parameters by adding command-line arguments to the "maxent.bat" file. Many of the command-line arguments also have abbreviations, so the run described in batchExample.bat could also be initiated using this command:

```
java -mx512m -jar maxent.jar –e layers –t eco –s samples\bradypus.csv –o outputs –r -a
```

REGULARIZATION

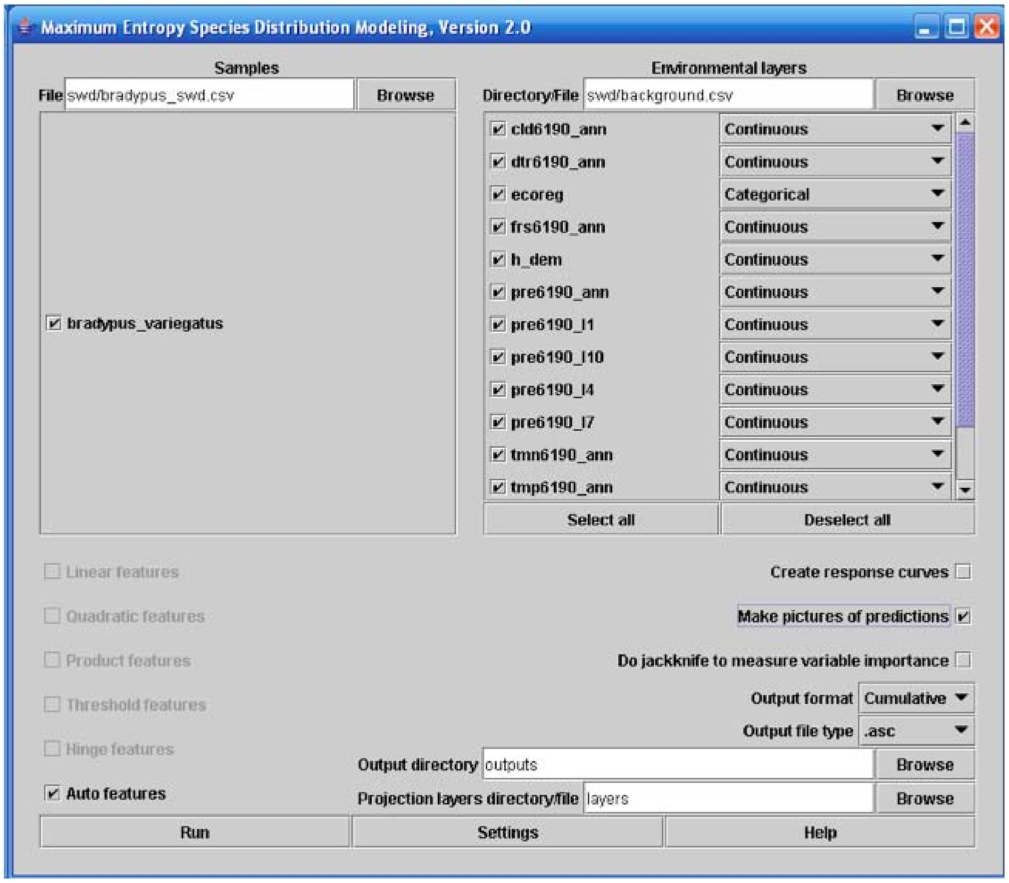
The "regularization multiplier" parameter on the settings panel affects how focused or closely-fitted the output distribution is – a smaller value than the default of 1.0 will result in a more localized output distribution that is a closer fit to the given presence records, but can result in overfitting (fitting so close to the training data that the model doesn't generalize well to independent test data). A larger regularization multiplier will give a more spread out, less localized prediction. Try changing the multiplier, and examine the pictures produced and changes in the AUC. As an example, setting the multiplier to 3 makes the following picture, showing a much more diffuse distribution than before:



The potential for overfitting increases as the model complexity increases. First try setting the multiplier very small (e.g. 0.01) with the default set of features to see a highly overfit model. Then try the same regularization multiplier with only linear and quadratic features.

PROJECTING

A model trained on one set of environmental layers (or SWD file) can be "projected" by applying it to another set of environmental layers (or SWD file). Situations where projections are needed include modeling species distributions under changing climate conditions, applying a model of the native distribution of an invasive species to assess invasive risk in a different geographic area, or simply evaluating the model at a set of test locations in order to do further statistical analysis. Here, we're going to use projection for a very simple task: to make an output ascii grid and associated picture when the samples and background are in SWD format. Type in, or browse for, the samples file "swd\bradypus_swd.csv" and the environmental layers in "swd\background.csv", then enter the "layers" directory in the "Projection Layers Directory", as pictured below. (More generally, the projection layers directory would contain grids with the same names as the variables used for training the model, but describing a different geographic region or different climatic conditions.)



Maximum Entropy Species Distribution Modeling, Version 2.0

Samples
File: swd/bradypus_swd.csv
☑ bradypus_variegatus

Environmental layers
Directory/File: swd/background.csv

☑ cld6190_ann	Continuous
☑ dtr6190_ann	Continuous
☑ ecoreg	Categorical
☑ frs6190_ann	Continuous
☑ h_dem	Continuous
☑ pre6190_ann	Continuous
☑ pre6190_l1	Continuous
☑ pre6190_l10	Continuous
☑ pre6190_l4	Continuous
☑ pre6190_l7	Continuous
☑ tmn6190_ann	Continuous
☑ tmp6190_ann	Continuous

☐ Linear features
☐ Quadratic features
☐ Product features
☐ Threshold features
☐ Hinge features
☑ Auto features

Create response curves ☐
Make pictures of predictions ☑
Do jackknife to measure variable importance ☐
Output format: Cumulative
Output file type: .asc
Output directory: outputs
Projection layers directory/file: layers

When you press “Run”, a model is trained on the SWD data, and then projected onto the full ascii grids in the “layers” directory. The output ascii grid is called “bradypus_variegatus_layers.asc”, and in general, the projection directory name is appended to the species name, in order to distinguish it from the standard (un-projected) output. If “make pictures of predictions” is selected, a picture of the projected model will appear in the “bradypus.html” file.

ANALYZING MAXENT OUTPUT IN R

Maxent produces a number of output files for each run. Some of these files can be imported into other programs if you want to do your own analysis of the predictions. Here, we demonstrate the use of the free statistical package R on Maxent outputs: this section is intended for users who have experience with R. We will use the following two files produced by Maxent:

bradypus_variegatus.csv
bradypus_variegatus_samplePredictions.csv

The first of these is produced when the background data are given in SWD format, and the second is always produced. Make sure you have test data (for example, by setting the random test percentage to 25); we will be evaluating the Maxent outputs using the same test data Maxent used. First, we start R, and install some packages (assuming this is the first time we’re using them) and then load them by typing (or pasting):

```
install.packages(“ROCR”, dependencies=TRUE)
install.packages(“vcd”,  dependencies=TRUE)
library(ROCR)
library(vcd)
library(boot)
```

Throughout this section, we will use blue text to show R code and commands and green to show R outputs. Next we change directory to where the Maxent outputs are, for example:

```
setwd(“c:/maxent/tutorial/outputs”)
```

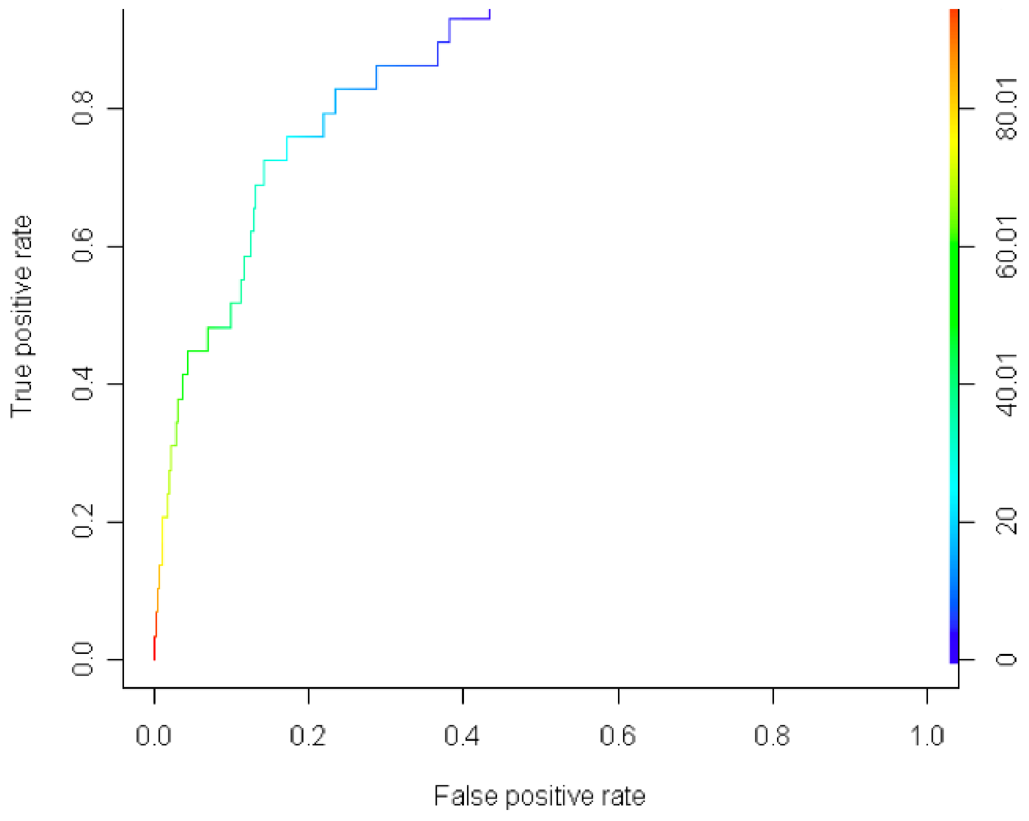
then read the Maxent predictions at the presence and background points, and extract the columns we need:

```
presence <- read.csv("bradypus_variegatus_samplePredictions.csv")
background <- read.csv("bradypus_variegatus.csv")
pp <- presence$Cumulative.prediction                # get the column of predictions
testpp <- pp[presence$Test.or.train=="test"]        # select only test points
trainpp <- pp[presence$Test.or.train=="train"]      # select only test points
bb <- background$Maxent.cumulative.values.at.background.points
```

Now we can put the prediction values into the format required by ROCR, the package we will use to do some ROC analysis, and generate the ROC curve:

```
combined <- c(testpp, bb)                              # combine into a single vector
label <- c(rep(1,length(testpp)),rep(0,length(bb)))   # labels: 1=present, 0=random
pred <- prediction(combined, label)                    # labeled predictions
perf <- performance(pred, "tpr", "fpr")                # True / false positives, for ROC curve
plot(perf, colorize=TRUE)                              # Show the ROC curve
performance(pred, "auc")@y.values[[1]]                 # Calculate the AUC
```

The plot command gives the following result:



while the "performance" command gives an AUC value of 0.8677759, consistent with the AUC reported by Maxent. Next, as an example of a test available in R but not in Maxent, we will make a bootstrap estimate of the standard deviation of the AUC.

```
AUC <- function(p,ind) {
pres <- p[ind]
combined <- c(pres, bb)
label <- c(rep(1,length(pres)),rep(0,length(bb)))
predic <- prediction(combined, label)
return(performance(predic, "auc")@y.values[[1]])
}

 b1 <- boot(testpp, AUC, 100)  # do 100 bootstrap AUC calculations
 b1                             # gives estimates of standard error and bias
```

This gives the following output:

```
 ORDINARY NONPARAMETRIC BOOTSTRAP

  Call:
  boot(data = testpp, statistic = AUC, R = 100)

  Bootstrap Statistics :
       original       bias     std. error
t1* 0.8677759 -0.0003724138  0.02972513
```

and we see that the bootstrap estimate of standard error (0.02972513) is close to the standard error computed by Maxent (0.028). The bootstrap results can also be used to determine confidence intervals for the AUC:

```
boot.ci(b1)
```

gives the following four estimates – see the resources section at the end of this tutorial for references that define and compare these estimates.

```
Intervals :
  Level      Normal              Basic
  95%   ( 0.8099,  0.9264 )   ( 0.8104,  0.9291 )
```

```
Level     Percentile          BCa
95%   ( 0.8064,  0.9252 )  ( 0.7786,  0.9191 )
```

Those familiar with use of the bootstrap will notice that we are bootstrapping only the presence values here. We could also bootstrap the background values, but the results would not change much, given the very large number of background values (10000).

As a final example, we will investigate the calculation of binomial and Cohen's Kappa statistics for some example threshold rules. First, the following R code calculates Kappa for the threshold given by the minimum presence prediction:

```
confusion <- function(thresh) {
    return(cbind(c(length(testpp[testpp>=thresh]), length(testpp[testpp<thresh])),
             c(length(bb[bb>=thresh]), length(bb[bb<thresh]))))
  }
  mykappa <- function(thresh) {
    return(Kappa(confusion(thresh)))
  }
  mykappa(min(trainpp))
```

which gives a value of 0.0072. If we want to use the threshold that minimizes the sum of sensitivity and specificity on the test data, we can do the following, using the true positive rate and false positive rate values from the "performance" object used above to plot the ROC curve:

```
fpr = perf@x.values[[1]]
 tpr = perf@y.values[[1]]
 maxsum = 0
 for (i in 1:length(perf@alpha.values[[1]])) {
   sum = tpr[[i]] + (1-fpr[[i]])
   if (sum > maxsum) {
     maxsum = sum
     cutoff = perf@alpha.values[[1]][[i]]
     index = i
   }
 }
 mykappa(cutoff)
```

This gives a Kappa value of 0.0144. To determine binomial probabilities for these two threshold values, we can do:

```
mybinomial <- function(thresh) {
   conf <- confusion(thresh)
   trials <- length(testpp)
   return(binom.test(conf[[1]][[1]], trials, conf[[1,2]] / length(bb), "greater"))
  }
 mybinomial(min(trainpp))
 mybinomial(cutoff)
```

This gives p-values of 5.979e-09 and 2.397e-11 respectively, which are both slightly larger than the p-values given by Maxent. The reason for the difference is that the number of test samples is greater than 25, the threshold above which Maxent uses a normal approximation to calculate binomial p-values.

R RESOURSES

Some good introductory material on using R can be found at:

http://spider.stat.umn.edu/R/doc/manual/R-intro.html, and other pages at the same site.

http://www.math.ilstu.edu/dhkim/Rstuff/Rtutor.html

TERMS OF USE

This material is based on work supported by the National Science Foundation under the Course, Curriculum and Laboratory Improvement program (NSF 0127506).