Technical Note:
Analyzing land cover change with logistic regression in R

D G Rossiter[*]
Analía Loza[†]

Version 2.4; May 14, 2016

Contents

[*] Cornell University, Section of Soil & Crop Sciences
[†] ITC MSc 2004

1 Motivation

This document presents a case study to illustrate how land cover change may be analysed using the R environment for statistical computing and visualisation [9].

These notes only scratch the surface of R's capabilities; the reader is encouraged to consult the on-line help, tutorials, and user's manuals. Similarly, they do not pretend to fully explain the applied statistical methods. A good source for general linear models is Fox [2, 4, 3]; all sorts of advanced methods for R are explained by Venables and Ripley [20]. Hosmer and Lemeshow [8] give an accessible treatment of logistic regression using examples from medical diagnostics.

These notes are designed so that you can *cut* the code shown in the boxes labelled **R code** and *paste* it directly into the R console; you should then see the output in the boxes labelled **R console output**. You can also load the source `.R` files, which should have been provided as a compressed file with this document, into an R environment such as RStudio and run the code there[1]. Of course, you are encouraged to edit the code and experiment.

The aim of this analysis is to quantify and explain observed land cover changes between two dates, based on a set of factors that are *a priori* expected to affect such change. The aim is to predict the *probability* that a plot's land use has changed, given a set of *predictive factors*. The factors that are associated with a higher chance of change are then examined more closely, to see if any reason can be given for the close association.

! → As with any statistical model, this one does *not* say anything about *causation*, only the *degree of association* between change and predictors. Reasoning about causation requires *meta-statistical* analysis, i.e. using statistical *evidence* to support arguments about likely mechanisms of change.

2 Research questions

Although the focus of this note is on the R techniques, it is instructive to place the example used in context by examining some of the original research questions for which the dataset was collected [13, §1.3]; the results have been summarized in a research paper [19].

1. What were the changes in land cover in the Chapare region of Cochabamba Province, Bolivia from 1986 to 2002?
2. What are the causative or controlling factors associated with deforestation?

[1] The sequence of these is `LCC_{0, 1, 21, ROC, 22, 2a, 3, 3a}`

(a) Agricultural colonization (legal and illegal);

(b) Distance to roads and existing settlements;

(c) Land tenure;

(d) Soil suitability and constraints for agriculture.

To answer the second question, we would like to build an **quantitative explanatory model** including one or more of the suspected factors.

3 Example data set

This data set is described by Loza [13]. It consists of 1 064 samples, each representing a single 30×30 m grid cell, from the Chapare region of Cochabamba province, Bolivia. This data set will be used to build and *calibrate* models of land cover change.

The data has been prepared as a *comma-separated value* ("CSV") file. This format is an export option from Excel.

TASK 1 : Open `lcc.csv` in a plain text editor such as Notepad and examine its structure. •

Here are the first few lines.

```
  "cov","dr","ds","t","lspos","text"
"CC", 2924.3,4853,"Nt","A","LC"
"CC", 2535.2,5242,"Nt","A","LC"
"CO", 2146.1,4957,"Ci","A","LC"
```

Fields

The fields are:

cov : Land cover class

dr : Distance to nearest road, meters

ds : Distance to nearest settlement, meters

t : Land tenure class

lspos : Landscape position class (slope and wetness)

text : General soil texture class

Land cover

The first item is a composite code of two letters. The first identifies land cover in 1986 and the second in 2000. The single letters are:

C : closed forest

O : open forest

N : no forest

Distances The distances to road (code `dr`) and settlement (code `ds`) were computed in ILWIS, using a segment map of the road network and a point map of the settlements, respectively.

Land tenure The fourth item is the land tenure [13, Table 6]; this was and is a complicated issue in Bolivia and we just give the most important points about each class, accurate at the date of the study:

- Nt : No established title; this is generally private land that has not yet been regulated under the terms of the 1996 Agrarian Reform law; in some cases this has reverted to the State because of underutilization but in others it is actively managed by its pre-Reform owner;
- CA : Agricultural colonies (private); these are legal tenure agriculture and ranching colonies;
- Ce : Carrasco National Park (protected area);
- Ci : Indigenous communities (traditional rights);
- Cp : Conservation areas (private); in the study area this is a 6300 ha University research station.

The last two items were generalised from classes of the Fertility Capability Classification [17, 18], and are thought to represent important soil-landscape factors that are expected to influence land use decisions.

Landscape position Landscape position (code `lspos`):

- A : flat; seasonally wet (gleyed; LCC code `g`);
- B : 1 – 25% slope gradient;
- C : 25 – 90% slope gradient;
- F : > 90% slope gradient; very steep

Soil texture Generalized soil texture (code `text`):

- L : loamy
- L' : loamy with 10 – 35% gravel
- LC : loamy topsoil, clayey subsoil

3.1 Browsing the example data set

TASK 2 : Start R and switch to the directory where the dataset is stored.

•

How you do this depends on your system; see [16] if you are not clear on how to do this.

> **Note:** The output in some parts of this document was produced by Sweave [11, 10] and R version 3.2.4 (2016-03-10) running on Darwin version 15.4.0. The text and graphical output you see here was automatically generated and incorporated into LaTeX by running code through R and its packages. Then the LaTeX document was compiled into the PDF version you are now reading. Your output may be slightly different on different versions and on different platforms.

TASK 3 : Load the dataset from file `lcc.csv` into variable `dset`, examine its structure, and display the first few observations. •

```
> dset <- read.csv("lcc.csv")
> str(dset)

'data.frame':        1064 obs. of  6 variables:
 $ cov  : Factor w/ 9 levels "CC","CN","CO",..: 1 1 3 6 4 9 2 6 1 1 ...
 $ dr   : num  2924 2535 2146 2442 2831 ...
 $ ds   : int  4853 5242 4957 4690 5079 4283 3999 3732 4121 3976 ...
 $ t    : Factor w/ 5 levels "CA","Ce","Ci",..: 5 5 3 3 3 5 5 5 5 5 ...
 $ lspos: Factor w/ 4 levels "A","B","C","F": 1 1 1 1 1 1 1 1 1 1 ...
 $ text : Factor w/ 3 levels "L","L'","LC": 3 3 3 3 3 3 3 3 3 3 ...

> dset[1:5,]

  cov     dr   ds  t lspos text
1  CC 2924.3 4853 Nt     A   LC
2  CC 2535.2 5242 Nt     A   LC
3  CO 2146.1 4957 Ci     A   LC
4  NO 2442.3 4690 Ci     A   LC
5  NC 2831.4 5079 Ci     A   LC

> summary(dset)

      cov            dr              ds        t       lspos
 CC     :356   Min.   :    0   Min.   :  44   CA: 58   A:371
 CN     :279   1st Qu.:  322   1st Qu.: 813   Ce:132   B:387
 CO     :220   Median :  713   Median :1288   Ci: 37   C:217
 NN     : 86   Mean   : 1215   Mean   :1506   Cp: 63   F: 89
 NO     : 45   3rd Qu.: 1411   3rd Qu.:1928   Nt:774
 OO     : 33   Max.   :12516   Max.   :5750
 (Other): 45
 text
 L  :518
 L' :207
 LC :339
```

Note that the `read.csv` function[2] was able to determine that four of the variables are nominal, or in R terms, *unordered factors*.

[2] A shorthand for the `read.table` function with defaults appropriate for Comma-Separated Values (CSV) files

If the object is attached to the search path, we can refer to the fields by name, rather than use the `frame$field` construction.

TASK 4 : Attach the dataframe to the search path. •

```
> attach(dset)
```

4 Identifying land cover change

Field `cov` codes the land covers at two dates; the first letter represents the cover at the first date and the second letter at the second date. From this we can derive a field that codes whether the sample changed or not between those dates. This is a *logical* response variable, which takes only two values: *True* or *False*. These are often incorrectly coded as *binary* variables, with 1 standing for change and 0 for no change. If this mistake is made, the analyst is tempted to use the numbers 0 and 1 as if they were values of a *ratio* variable[3], and try to apply statistical methods appropriate for ratio response variables, such as linear regression. This is a serious error. R helps here by clearly identifying logical variables as such.

TASK 5 : Compute a logical vector of *True* values for observations with a change in land cover between the two dates, and *False* values for the others. •

Q1 : *What proportion of observations changed land cover between the two dates?* •

```
> changed  <- ! (cov=="CC" | cov=="00" | cov=="NN")
> summary(changed)

   Mode   FALSE    TRUE    NA's
logical     475     589       0

> round(sum(changed)/length(changed),3)

[1] 0.554
```

About 55% of the samples changed their land cover.

TASK 6 : Add the logical vector as a field to the data frame using the `cbind()` method, and then remove the temporary variable. •

[3] a continuous numerical variable with a natural zero

```
> dset <- cbind(dset, changed)
> str(dset)

'data.frame':        1064 obs. of  7 variables:
 $ cov    : Factor w/ 9 levels "CC","CN","CO",..: 1 1 3 6 4 9 2 6 1 1 ...
 $ dr     : num  2924 2535 2146 2442 2831 ...
 $ ds     : int  4853 5242 4957 4690 5079 4283 3999 3732 4121 3976 ...
 $ t      : Factor w/ 5 levels "CA","Ce","Ci",..: 5 5 3 3 3 5 5 5 5 5 ...
 $ lspos  : Factor w/ 4 levels "A","B","C","F": 1 1 1 1 1 1 1 1 1 1 ...
 $ text   : Factor w/ 3 levels "L","L'","LC": 3 3 3 3 3 3 3 3 3 3 ...
 $ changed: logi  FALSE FALSE TRUE TRUE TRUE FALSE ...

> rm(changed)
```

Note that the field `changed` has type 'logical', i.e. it takes the values `TRUE` (a change has occurred) or `FALSE`. Also note that this field was automatically given the same name as the temporary variable.

4.1 Limiting the changes to one original cover class

Different driving forces may be behind different sorts of change. In this example we will only analyze the changes from an initial state of closed forest, i.e. **deforestation**, between the two dates; see §12 for other ideas.

TASK 7 : Make a new data frame as a subset of the full dataset, with only those observations whose initial land cover was 'closed forest'. •

```
> dc <- dset[substring(as.character(cov),1,1)=="C",]
```

This R code uses a *logical expression* to select rows which meet the criterion (note that the criterion is written in the rows position of the array notation `[rows, columns]`, and then includes all columns in the selected rows (note the empty expression in the columns position).

Q2 : *How many observations are in the subset?* •

```
> str(dc)

'data.frame':        855 obs. of  7 variables:
 $ cov    : Factor w/ 9 levels "CC","CN","CO",..: 1 1 3 2 1 1 1 1 3 2 ...
 $ dr     : num  2924 2535 2146 1188 2254 ...
 $ ds     : int  4853 5242 4957 3999 4121 3976 4551 2921 3311 3026 ...
 $ t      : Factor w/ 5 levels "CA","Ce","Ci",..: 5 5 3 5 5 5 5 5 5 5 ...
 $ lspos  : Factor w/ 4 levels "A","B","C","F": 1 1 1 1 1 1 1 1 1 1 ...
 $ text   : Factor w/ 3 levels "L","L'","LC": 3 3 3 3 3 3 3 3 3 3 ...
 $ changed: logi  FALSE FALSE TRUE TRUE FALSE FALSE ...

> dim(dc)[1]

[1] 855
```

The subset has 855 of the original 1064 observations.

TASK 8 : Remove (detach) the complete dataset `dset` from the search path, and replace it (attach) with the subset `dc`. We also re-compute the change for the subset. •

```
> detach(dset); attach(dc)
```

```
> changed <- (cov!="CC"); summary(changed)
```

```
   Mode   FALSE    TRUE    NA's
logical     356     499       0
```

```
> round(sum(changed)/length(changed),3)
```

```
[1] 0.584
```

Note that `changed` now refers to `dc$changed`, not `dset$changed`.

Q3 : *What is the proportion of deforestation?* •

About 58% of the original closed forest has been converted.

5 Probability of change assessed by cross-classification

A simple way to assess the relation between a classified factor and change is to classify the changed sites according to the factor, in a *cross-classification table*. We can examine the numbers of observations that changed, or see these as proportions of each class that changed. The χ^2 statistic for such a table tests whether the uneven proportions in the table could arise by chance.

TASK 9 : Compute a cross-classifiction table of the number of observations deforested, classfied by landscape position; test whether it is different from a chance assignment of change to landscape positions. •

First, the cross-classification table, and the computed χ^2:

```
> (ct <- table(changed, lspos))
```

```
       lspos
changed   A   B   C   F
  FALSE 118 107  87  44
  TRUE  155 198 102  44
```

```
> summary(ct)
```

```
Number of cases in table: 855
Number of factors: 2
```

```
Test for independence of all factors:
        Chisq = 9.7, df = 3, p-value = 0.021
```

```
> (cs <- chisq.test(ct))
```

```
        Pearson's Chi-squared test

data:  ct
X-squared = 9.71, df = 3, p-value = 0.021
```

The expected change if points were randomly assigned to classes is:

```
> round(cs$expected)
```

```
       lspos
changed   A   B   C  F
  FALSE 114 127  79 37
  TRUE  159 178 110 51
```

And the difference between observed and expected is then:

```
> ct - round(cs$expected)
```

```
       lspos
changed   A   B   C   F
  FALSE   4 -20   8   7
  TRUE   -4  20  -8  -7
```

As the example shows, the `summary` method applied to a contingency table calculates the χ^2 statistic.

Q4 : *Does this table show a difference from a chance assignment of change to landscape positions?* •

The difference between observed and expected shows that landscape position B was deforested more than expected; the reverse is true for the other positions. The χ^2 test of independence shows that this is unlikely by chance alone ($p = 0.02 < 0.05$), so landscape position is somewhat associated with different proportions of change. However, this test applies to the whole table, and does not indicate which classes are significantly different from the mean proportion of change (in this case, 0.584).

TASK 10 : Express the contingency table as *proportions* rather than *frequencies* (counts), normalized to 1 for each class. •

```
> (ct.p <- round(t(t(ct)/apply(ct,2,sum)),2))
```

```
       lspos
changed    A    B    C    F
  FALSE 0.43 0.35 0.46 0.50
  TRUE  0.57 0.65 0.54 0.50
```

Note how the column sums are all 1 (normalized).

> This is fairly tricky R code, so some comments are in order. First, the `apply` method applies the function given by its third argument, in this case `sum`, across rows (if the second argument is 1) or columns (if 2), to the matrix named in the first argument, in this case table `ct`. The result is a vector with the same number of rows or columns, containing the value of the function applied to that row or column. So here we compute the *column* sum and then divide this sum into the count of each cell to get the proportion in the column. To make the vector conform with the correct dimension, the table has to be transposed with the `t` method before the division, and then transposed back again for presentation.

Q5 : *Do the four landscape positions have different proportions of* `TRUE` *(plot was converted) and* `FALSE` *(not)? If so, which positions are more and less converted than the average?* •

The proportion of change is clearly highest for position `B` (not wet, not steep), suggesting a slight preference of colonists for these sites.

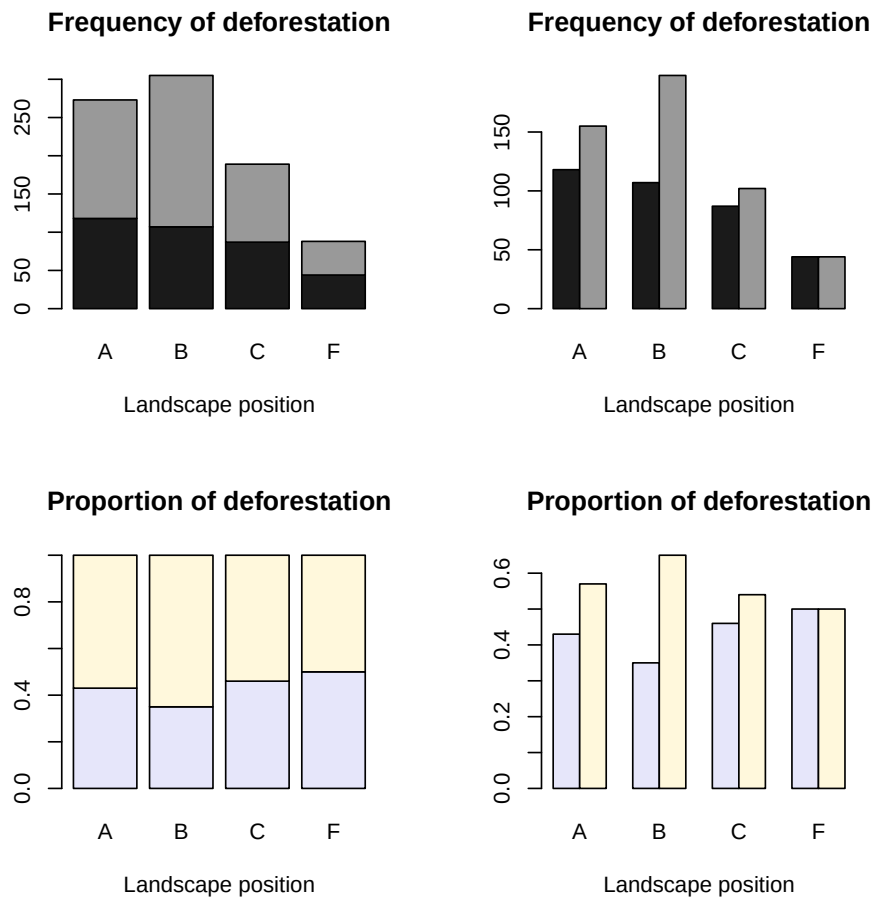
5.1 Visualising cross-classification with a bar plot

Cross-classification tables lend themselves to visualisation with a *bar plot*. This shows bars (rectangles) for each level of the classifying factor, with area proportional to its number or proportion of `TRUE` and `FALSE`. Bar charts of such tables are of two forms: *stacked* (one bar per level, separated into `T` and `F` portions) or *side-by-side* (two bars per level, one each for `T` and `F`).

TASK 11 : Show both the frequencies and proportions of `TRUE` and `FALSE` by landscape position as bar charts, both stacked and side-by-side. •

```
> par(mfrow=c(2,2))
> col.vec <- c("gray10", "gray60")
> barplot(ct, col=col.vec, main="Frequency of deforestation",
+    xlab="Landscape position")
> barplot(ct, beside=T, col=col.vec, main="Frequency of deforestation",
+    xlab="Landscape position")
> col.vec <- c("lavender", "cornsilk")
> barplot(ct.p, col=col.vec, main="Proportion of deforestation",
+    xlab="Landscape position")
> barplot(ct.p, beside=T, col=col.vec,
+    main="Proportion of deforestation", xlab="Landscape position")
```

```
> par(mfrow=c(1,1))
> rm(col.vec)
```



Q6 : *What are the respective advantages of the stacked and side-by-side barcharts?* •

The stacked chart splits the whole into two parts, and at the same time shows the relative totals of the classes. The side-by-side chart makes it easier to compare the changed/not changed within each class.

Q7 : *What are the respective advantages of the frequency and proportions plots?* •

The proportions sum to 1 within each class, so show clearly the relative changes. The frequencies show the actual number changed.

6 The logistic transformation

For a deeper analysis, we will use *logistic regression*, implemented as a special case of *generalised linear models*. As a preliminary, we have to define and explain the logistic transformation; in the following section we will put it to work.

The logistic transformation is often applied to variables which are defined only on the open interval $(0 \ldots 1)$, such as proportions, to variables on $(-\infty \ldots \infty)$. The transformed variable is the natural logarithm of the *odds* corresponding to the proportion; this is called the *log-odds*:

$$\ell = \log \frac{p}{1-p} \tag{1}$$

The odds are simply the ratio of the chance an event will occur, divided by the chance it won't, and is another way to express probability. For example, if for an event $p = 0.8$, the odds of the event occurring are $0.8/(1-0.8) = 0.8/0.2 = 4$, conventionally expressed as "four to one", or $4:1$, in favour of the event's occurrence. In common language, it is four times as likely that the event will occur as that it will not. The log-odds in this case is $\log 4 = 1.386$.

If an event is equally likely and unlikely, $p = 0.5$, the odds are $1:1$, and the log-odds are 0. Thus zero is the centre of a logit-transformed proportion; negative values indicate less than even odds, and positively values more.

> Note: In games of chance the odds ratio is conventionally used in an inverse sense to the above: it expresses the odds that an event *will not* occur. This is because a positive outcome is almost always less likely than a negative one in such games. For example, the gambling odds that a random card drawn from a standard 52 card deck will not be a face card or ace is $(13-5)/5 = 1.6$, conventionally expressed with the smallest possible integer denominator as "eight to five", or $8:5$, "against".

The logistic transformation is undefined for proportions 0 and 1. This makes sense if we consider the proportions as estimates of probabilities: if the event is certain or impossible, there is nothing to analyse. We are only interested in modelling uncertain events. Another way to understand this is the realise that it is meaningless to have infinite odds for or against an event, so the range of the logistic function does *not* include $\pm\infty$.

The inverse transform, from log odds to proportion, is:

$$p = \frac{1}{1+e^{-\ell}} \tag{2}$$

which is often written in the mathematically-equivalent form (obtained by multiplying numerator and denominator by e^{ℓ}):

$$p = \frac{e^{\ell}}{1 + e^{\ell}} \tag{3}$$

To examine the transformation in R, we define two functions, the logistic transformation and its inverse. The forward function goes from a proportion on $(0 \ldots 1)$ to its logistic value on $(-\infty \ldots \infty)$ and the inverse function goes from a logistic variable back to the proportion.

TASK 12 : Define R functions for the forward and inverse logistic transformations. •

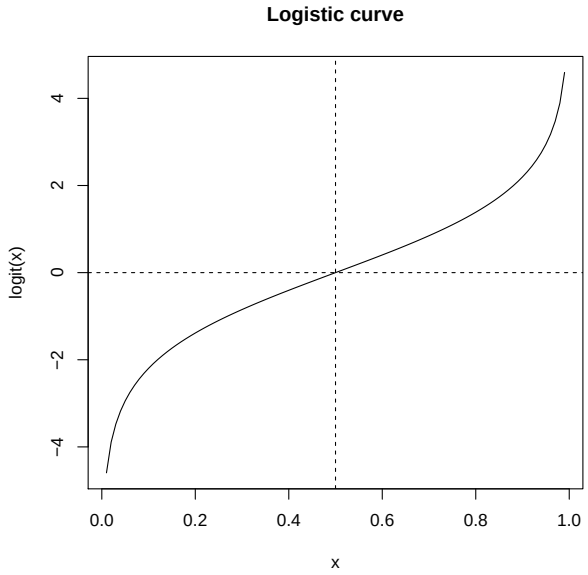
```
> logit <- function(x) log(x/(1-x))
> logit.inv <- function(x) 1/(1+exp(-x))
```

We can examine the effect of these two functions on probabilities from 0.02 to 0.98, in increments of 0.04:

```
> (x <- seq(0.02, 0.98, by=0.04))

 [1] 0.02 0.06 0.10 0.14 0.18 0.22 0.26 0.30 0.34 0.38 0.42 0.46
[13] 0.50 0.54 0.58 0.62 0.66 0.70 0.74 0.78 0.82 0.86 0.90 0.94
[25] 0.98

> logit(x)

 [1] -3.89182 -2.75154 -2.19722 -1.81529 -1.51635 -1.26567
 [7] -1.04597 -0.84730 -0.66329 -0.48955 -0.32277 -0.16034
[13]  0.00000  0.16034  0.32277  0.48955  0.66329  0.84730
[19]  1.04597  1.26567  1.51635  1.81529  2.19722  2.75154
[25]  3.89182

> logit.inv(logit(x))

 [1] 0.02 0.06 0.10 0.14 0.18 0.22 0.26 0.30 0.34 0.38 0.42 0.46
[13] 0.50 0.54 0.58 0.62 0.66 0.70 0.74 0.78 0.82 0.86 0.90 0.94
[25] 0.98
```

We can graph the logistic curve as follows:

```
> x <- seq(0.01, 0.99, by=0.01)
> plot(x,logit(x),type="l", main="Logistic curve")
> abline(h=0, lty=2); abline(v=0.5, lty=2)
```



6.1 A single-predictor logistic model

The logistic model in R is a special case of the *generalised linear model* (GLM), implemented in R by the `glm` method. This method requires a *model specification* using the same S model notation as used in the `lm` method, and in addition a specification of the *model family*. For the logistic model, this family is the *binomial*, also known as the *Bernoulli* family, where the response is a logical variable.

Let's begin by modelling change based on one predictor: landscape position. The question is to what extent the position affects the decision to covert a site from closed forest.

```
> glm.lspos <- glm(changed ~ lspos, family=binomial, data=dc)
> summary(glm.lspos)

Call:
glm(formula = changed ~ lspos, family = binomial, data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
 -1.45   -1.30    0.93    1.06    1.18

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.273      0.122    2.23    0.026 *
lsposB         0.343      0.171    2.00    0.045 *
lsposC        -0.114      0.190   -0.60    0.550
lsposF        -0.273      0.246   -1.11    0.267
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1151.5  on 851  degrees of freedom
AIC: 1159

Number of Fisher Scoring iterations: 4
```

The notation `changed ~ lspos` is an example of the S model specification; it can be read "model variable `changed` as a function of predictor `lspos`").

The intercept (representing the first class, `A`), and the coefficient for class `B` both have probability $p < 0.05$ that they could occur by chance, so the model does find some significant differences (but not much). Positions `A` and `B` are more likely to be converted than the other two.

This looks very much like the summary of a (non-generalised) linear one-way ANOVA model, except there is no goodness-of-fit (R^2). Instead, we are given the *null deviance*, which measures the variability of the dataset, compared to the *residual deviance*, which measures the variability of the *residuals*, after fitting the model. These deviances can be used like the total and residual sum of squares in a linear model to estimate the goodness of fit; this is sometimes referred to as the D^2 (by analogy with R^2):

```
> d2 <- function(model) { round(1-(model$deviance/model$null.deviance),4) }
> d2(glm.lspos)

[1] 0.0084
```

In this code, the notation `glm.lspos$deviance` extracts component `deviance` from fitted model object `glm.lspos`.

> Note: To see the components of an object, use the `str` ('structure') method.

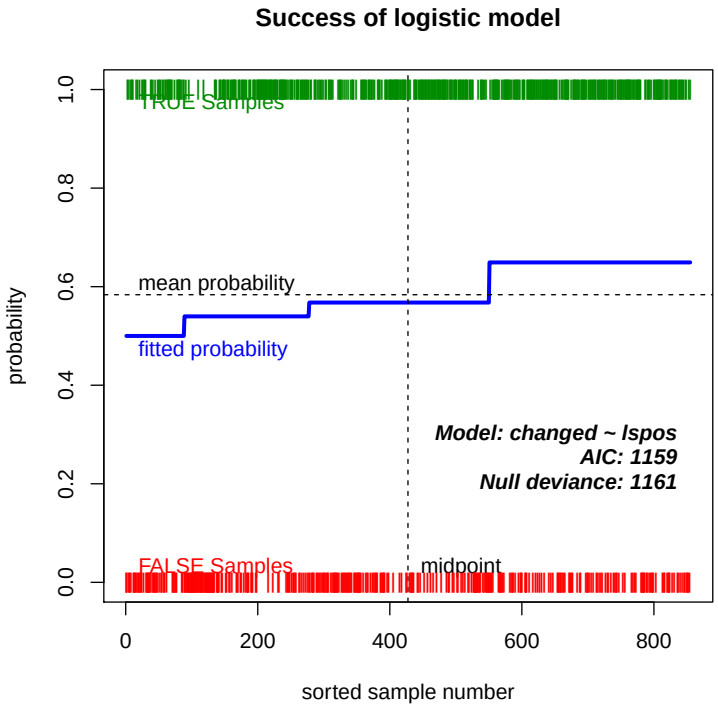
Less than 1% of the deviance in the model has been explained by landscape position. This is not very good! It is an example of a model that is significantly different from random effects, but it not very helpful to understand the underlying process.

6.2 Visualising the logistic model

The logistic model assigns a probability of change, on $(0 \ldots 1)$, for each sample. The sample itself was observed either to change (1) or not (0). We have written a function `logit.plot` (§A.1) to produces a plot of the logistic model's predictions, sorted by probability (so, the less probable changes to the left, the most probable to the right), with the samples corresponding to each probability either at the top (if the site actually changed) or at the bottom (if not).

This function is provided in the `lcc.R` source code file, which can loaded using the R `source()` method, and then applied to any fitted model:

```
> source("lcc.R")
> logit.plot(glm.lspos)
```



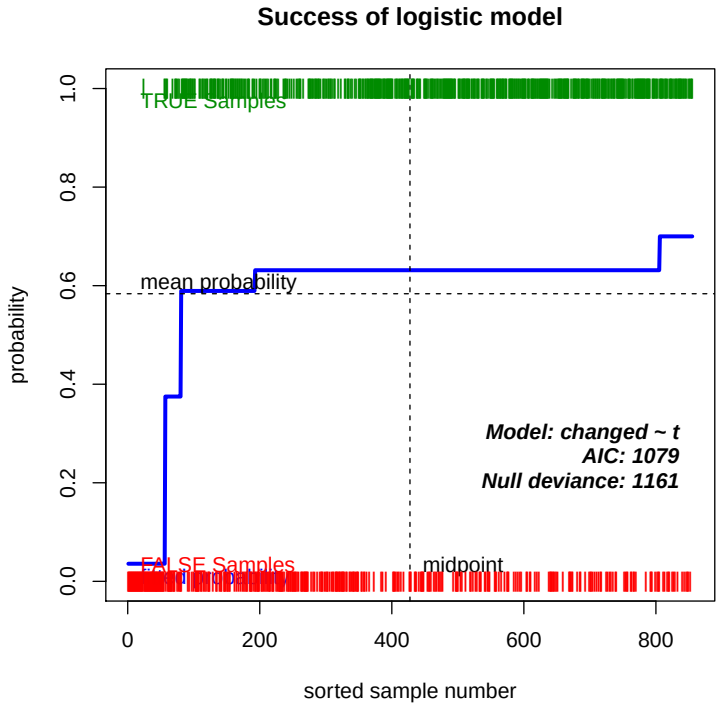
The plot for this model shows that the relation is clearly very poor. The fitted probabilities of change are all close to the mean probability (0.584), shown as a horizontal line. Also, there is very little difference between the density of the changed and unchanged samples.

6.3 Comparing models

Let's try this again,but with a different single predictor, land tenure.

```
> glm.t<-glm(changed ~ t, family=binomial, data=dc)
> summary(glm.t)

Call:
glm(formula = changed ~ t, family = binomial, data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.552  -1.413   0.959   0.959   2.582

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.847      0.309    2.75   0.0060 **
tCe           -0.486      0.363   -1.34   0.1810
```

```
tCi           -1.358      0.523   -2.60   0.0093 **
tCp           -4.143      0.783   -5.29  1.2e-07 ***
tNt           -0.309      0.320   -0.97   0.3332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1068.8  on 850  degrees of freedom
AIC: 1079

Number of Fisher Scoring iterations: 5

> 1-(glm.t$deviance/glm.t$null.deviance)

[1] 0.079627

> logit.plot(glm.t)
```



This is much better: one class (Cp, private conservation area) is very highly significantly different (lower probability of conversion) from the mean probability of conversion (in this case, much lower) and two others (Ci, i.e. indigenous comunities, and the intercept, representing the first-listed class CA, i.e. agricultural colonies) are highly significantly different (lower and higher probability, respectively). The residual deviance is much lower, and almost 8% of the variability has been explained. The plot clearly shows that this relation is much better than when landscape position was used as the single predictor. In particular, some non-

changed samples are fitted very well. There is a clear difference between the density of the changed and unchanged samples; the unchanged are clustered to the

To compare linear models we often use the adjusted R^2. A more general measure for these, which is also applicable to generalised linear models, is the *Akaike Information Criterion*, or AIC. This adjusts the residual deviance for the number of predictors, thus favouring parsimonious models. The AIC for the model from land tenure is 1078.8; for the prediction from landscape position it was 1159.5. Thus the model from land tenure is much better than the model from landscape position.

> Note that the AIC for a null model is the same as the null deviance, since there are no predictors to adjust it, in the present case 1161.3.

6.4 Evaluating models with the ROC curve

The success of logistic regression models may be assessed with the **ROC**, abbreviation for "**R**eceiver **O**perating **C**haracteristic" curve.[4]

This is well-explained for ecological models by Liao and McGee [12]; the same technique has been applied to land cover change models [15]. It was originally elaborated to compare medical diagnostic procedures [6, 7, 5]. ROC curves were used as an R programming example by Lumley [14]. Fawcett [1] presents a tutorial on ROC curves and other diagnostics to select classifiers, and a practical guide for their use research.

> **Note:** R package `ROCR` can be used to produce and evaluate ROC curves and other measures of logistic model performance. It was first placed on CRAN after these notes were complete; I have not evaluated it; probably it does some of what is explained in this section somewhat more elegantly or attractively.

The ROC curve is a plot of the *sensitivity* (proportion of true positives) of the model prediction against the complement of its *specificity* (proportion of false positives), at a series of *thresholds* for a positive outcome. The logistic model gives the *probability* that each location has changed; this can be changed to a *binary* outcome (changed vs. not changed) by selecting a threshold.

To explain this, we first examine the concepts of sensitivity and specificity at a given threshold, and then show how to construct an ROC curve from a series of thresholds.

[4] The name comes from its original application to signal detection; the "receiver" was a radio receiver at different detection thresholds.

6.4.1 Sensitivity and specificity at one threshold

In the present example we have 855 observations, with predicted probabilities of change from 0.0357 (almost zero) to 0.7. If we select a threshold of $p = 0.5$ (change equally likely or not), 775 (of 855) locations are predicted to change; if the threshold is raised to 0.65, only 50 are predicted to change. In fact, 499 changed:

```
> length(glm.t$fitted)

[1] 855

> summary(glm.t$fitted)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0357  0.6310  0.6310  0.5840  0.6310  0.7000

> sum(glm.t$fitted>0.5)

[1] 775

> sum(glm.t$fitted>0.65)

[1] 50

> sum(dc$changed)

[1] 499
```

At any threshold we can compute the sensitivity and specificity, by comparing the predicted with actual change. The *sensitivity* is defined as the ability of the model to find the "positives", i.e. sites that actually changed land use:

$$\text{Sensitivity} \stackrel{\text{def}}{=} \frac{\text{True positives}}{\text{Total positives}}$$

For example, at $p = 0.5$, this model predicts 488 of the 499 sites that changed land use, so the sensitivity is 0.98, which is very good.

```
> sum((glm.t$fitted > 0.5) & dc$changed)

[1] 488

> (sens.5 <- sum((glm.t$fitted > 0.5) & dc$changed)/sum(dc$changed))

[1] 0.97796
```

(A note on this R code: the `sum()` method when applied to a logical vector of `TRUE` and `FALSE` counts the number of `TRUE` in the vector.)

There is another side to a model's performance: the *specificity*, defined as the proportion of "negatives" that are correctly predicted

$$\text{Specificity} \stackrel{\text{def}}{=} \frac{\text{True negatives}}{\text{Total negatives}}$$

In this case 356 sites did not change land use; at at a threshold of $p = 0.5$ the model predicts that 80 sites did not change; of these 69 didn't change in fact; the specificity is thus $69/356 = 0.19$, which is very poor:

```
> sum(!dc$changed)

[1] 356

> sum((glm.t$fitted  < 0.5))

[1] 80

> sum((glm.t$fitted < 0.5) & (!dc$changed))

[1] 69

> (spec.5 <- sum((glm.t$fitted < 0.5) & (!dc$changed))/sum(!dc$changed))

[1] 0.19382
```

This model is quite successful in identifying sites that changed (488 of 499), but quite poor at finding sites that did not change (69 of 356).

The complement of the specificity is the *false positive* rate, that is, the proportion of incorrect predictions of change to the total unchanged. This and the specificity must sum to 1. Similarly, the complement of the sensitivity is the *false negative* rate, that is, the proportion of incorrect predictions of no change to the total changed. This and the sensitivity must sum to 1.

For this model, using a threshold of $p = 0.5$, there is a high rate of false positives (0.81) but a very low rate of false negatives (0.02). The model is predicting too much change.

```
> (fp.5 <- sum((glm.t$fitted > 0.5) & !dc$changed)/sum(!dc$changed))

[1] 0.80618

> spec.5 + fp.5

[1] 1

> (fn.5 <- sum((glm.t$fitted < 0.5) & dc$changed)/sum(dc$changed))

[1] 0.022044

> sens.5 + fn.5

[1] 1
```

We have written a function `logit.plot.quad` (§A.2) to produces a plot of the logistic model's sensitivity, specificity, false and true positives and negatives, at a user-supplied threshold; this is included in source code file `lcc.R`:

```
> source("lcc.R")
```

We plot this for the default threshold ($p = 0.5$); this is shown in Figure 1.

```
> logit.plot.quad(glm.t)
```

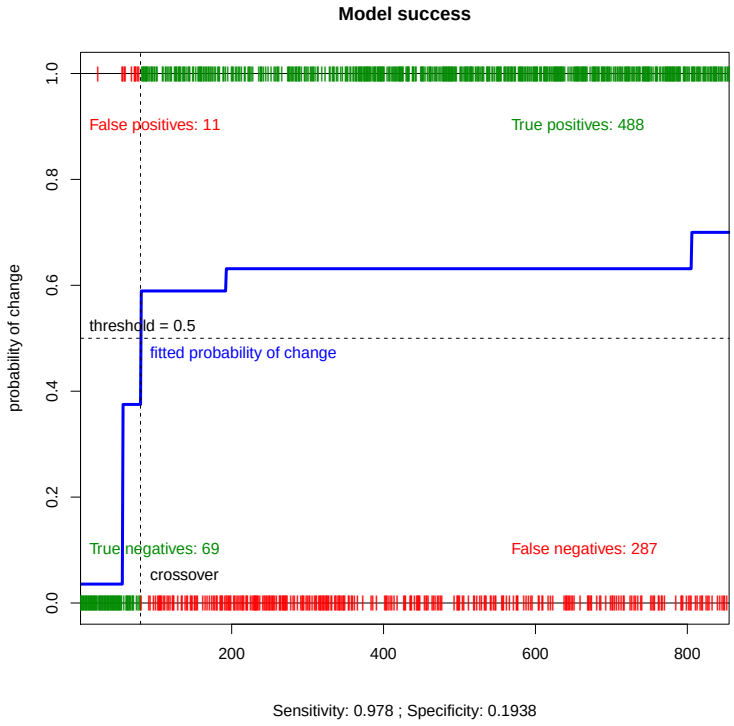


Figure 1: Illustrating the logistic model's performance

The red ticks represent errors: either false positives or false negatives. In this example it's clear that there are many false positives (i.e. negative cases that the model missed), leading to a low specificity, whereas there are few false negatives (i.e. positive cases that the model missed), leading to a high sensitivity.

6.4.2 The ROC curve

These computations can be repeated for any threshold; the sensitivity and specificity will change accordingly. A graph of the *sensitivity*, i.e. true positive rate (on the y-axis) vs. the *false positive* rate (on the x-axis) at different thresholds is called the *Receiver Operating Characteristic* (ROC) curve. Ideally, even at low thresholds, the model would predict most of the true positives with few false positives, so the curve would rise quickly from $(0, 0)$. The closer the curve comes to the left-hand border and then the top border of the graph ("ROC space"), the more accurate is the model; i.e. it has high sensitivity and specificity even at low thresholds. The closer the curve comes to the diagonal, the less accurate

is the model. This is because the diagonal represents the random case: the model predicts at random, so the chance of a true positive is equal to that of a false positive, at any threshold.

The area under the ROC curve (AUC) The ROC curve can be summarised by the *area under the curve* (AUC), computed by the trapezoidal rule (base times the median altitude):

$$A = \sum_{t=1}^{n} [x_{i+1} - x_i][(y_{i+1} + y_i)/2)]$$

where the i are the thresholds where the curve is computed. Note that the area under the diagonal is 0.5, so the ROC curve must define an area at least that large. The ROC area then measures the *discriminating power* of the model: the success of the model in correctly classifying sites that did and did not actually change.

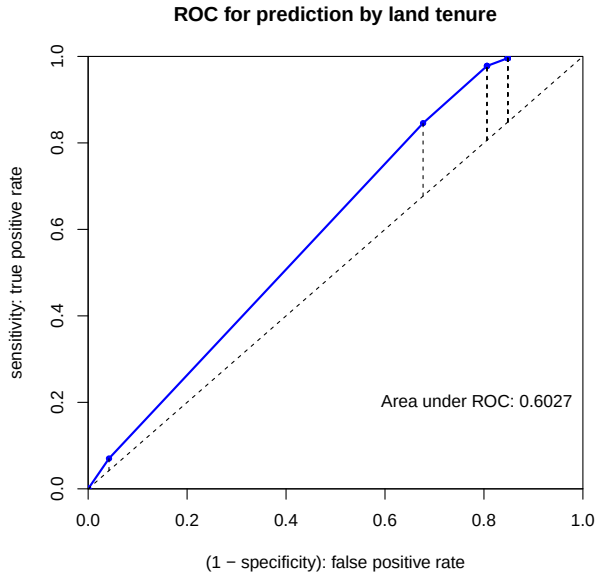
We have written functions to:

1. compute the ROC curve for a model, along with the AUC and the sensitivity/specificity (`logit.roc`; §A.3);
2. compute the area under an ROC curve (AUC) (`logit.roc.area`; §A.4); and
3. plot the curve with its area (`logit.roc.plot`; §A.5).

These are included in source code file `lcc.R`, which was loaded above.

Applying these to the single-predictor model (land tenure):

```
> r <- logit.roc(glm.t)
> logit.roc.area(r)

[1] 0.60266

> logit.roc.plot(r, "ROC for prediction by land tenure")
```



This model is not very successful: the ROC is close to the diagonal and its area is only 0.6, not much better than a random model (area = 0.5).

6.4.3 How to evaluate the AUC

There is no statistical test of the AUC; it depends on the application field, and what other modellers have found in similar studies. As a rule of thumb, the area can be "graded" as an academic grade on the 0-10 scale: 6 is "passing" (model is at least somewhat useful), 7 is 'good", 8 is "very good" and 9 is "excellent".

AUC vs. AIC In §6.3 we explained the use of the *Akaike Information Criterion*, abbreviated AIC, to compare models. Recall that the AIC adjusts the residual deviance for the number of predictors (i.e. model complexity), thus favouring parsimonious models. By contrast, the AUC does *not* take into account the model complexity; adding predictors to the model can never lower the AUC and will usually raise it, with no indication of whether this improvement is fitting signal (true relation) or noise (chance variability in the sample). Therefore, AIC should be used to compare models, and then AUC can be reported for the best parsimonious model.

6.5 Prediction from a single-predictor classified model

The predicted probability of change for a sample from any class is computed from the inverse logistic equation (Equation 2), and the coefficient for the appropriate class. This is the simplest use of the inverse link function. The `glm()` method finds coefficients for each class to predict

the logistic value, so we must apply the inverse transformation (Equation 2) to find the probability. The coefficient for a class can be extracted from the model and used in the inverse logit function.

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_c)}} \tag{4}$$

where β_0 is the log-likelihood of change for the first class (here, CA), β_c is the difference from the first class the log-likelihood of change associated with the named class c.

For example, for land tenure class Cp we compute:

```
> glm.t$coefficients

(Intercept)         tCe         tCi         tCp         tNt
    0.84730    -0.48628    -1.35812    -4.14313    -0.30941

> logit.inv(glm.t$coefficients["(Intercept)"]+glm.t$coefficients["tCp"])

(Intercept)
   0.035714
```

The probability of change for this class as predicted by the model is very small, less than 0.04. This is the first "step" on Figure 1, above.

We can compare this with the probability of change estimated directly from the classification table:

```
> summary(changed[t=="Cp"])

   Mode   FALSE    TRUE    NA's
logical      54       2       0
```

The actual change was $2/56 = 0.0357$, almost exactly the estimate from the GLM, 0.03571431.

To compute the probability of change of a given observation (e.g. a pixel in the image, whether or not it was part of the original sample), the value of the predictor (here, the class) must be extracted from the database, and the appropriate coefficient must be selected. Formally this can be written:

$$p = \frac{1}{1 + e^{-(\beta_0 + \sum_i \beta_{c_i} \delta_{c_i})}} \tag{5}$$

where β_0 is the log-likelihood of change for the first class (i.e. the intercept), β_{c_i} is the difference from the first class the log-likelihood of change associated with class c_i, and δ_{c_i} is the Kronecker delta for class c_i, i.e. 1 if the observation is in the class, 0 otherwise. At most one of the δ_{c_i} will be 1[5].

In practice, R code will be used to extract the correct coefficient from the model summary, knowing the class extracted from the database. For

[5] None will be 1 if the observation is in the first class

example, to compute the probability of change for all the observations in the sample:

```
> p.change <-
+      logit.inv(glm.t$coefficients["(Intercept)"]
+                + ifelse(as.numeric(dc$t)==1, 0,
+                         glm.t$coefficients[as.numeric(dc$t)]));
> summary(p.change)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0357  0.6310  0.6310  0.5840  0.6310  0.7000
```

Note the use of the `as.numeric` function to identify the the position in the array of coefficients for each class. Also, note that the model coefficients for all except the first-named class (here, `CA`) are *offsets* from the intercept, hence the use of the `ifelse` function.

This code can be used for non-sampled locations, replacing the sample dataframe `dc` with a dataframe of the new locations with the same structure. For the original sample, this is much more simply achieved by using the `fitted` method on the model object:

```
> p.change <- fitted(glm.t);
> summary(p.change)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0357  0.6310  0.6310  0.5840  0.6310  0.7000
```

7 Change related to several categorical variables

In most land cover change studies, we suspect that several factors have influenced the change, perhaps additively but more likely with *interactions*. In the previous sections we considered categorical factors one by one, and found the one with strongest predictive power. In this section we model change from two factors at once.

Interpretation of multivariate models is complicated by possible *correlation* between the predictors. For example, if a given tenure class were mostly associated with a given landscape position (e.g. public lands on the steep hills and private lands on the river terraces), then an additive multivariate model with the two predictors might select one or the other as the primary predictor, with the other appearing insignificant (i.e. no additional predictive power). This is mathematically correct, but we miss the insight into the process.

To assess the relation of two classified predictors, we can use a two-way table or *contingency matrix* and also the logistic multiple regression itself.

7.1 Cross-tabulation

The simplest way to study the effect of several predictors on land cover change is with a multi-way table. One way to produce this is with the `xtabs()` method, which uses the S model syntax. First we compute the total number of samples in each cross-tabulation category, then the number of true and the number of false, and finally their ratio, which estimates the likelihood of conversion. This latter is compared to the overall likelihood.

```
> t.all<-xtabs( ~ lspos + t, dc)
> t.t<-xtabs((changed==T) ~ lspos + t, dc)
> t.f<-xtabs((changed==F) ~ lspos + t, dc)
> t.all; t.t; t.f

     t
lspos  CA  Ce  Ci  Cp  Nt
    A   5   5  24   2 237
    B  23  80   0   1 201
    C  12  27   0  42 108
    F  10   0   0  11  67

     t
lspos  CA  Ce  Ci  Cp  Nt
    A   4   3   9   0 139
    B  14  48   0   0 136
    C   9  15   0   2  76
    F   8   0   0   0  36

     t
lspos CA Ce Ci Cp Nt
    A  1  2 15  2 98
    B  9 32  0  1 65
    C  3 12  0 40 32
    F  2  0  0 11 31

> round(t.t/t.f, 3)

     t
lspos    CA    Ce    Ci    Cp    Nt
    A 4.000 1.500 0.600 0.000 1.418
    B 1.556 1.500       0.000 2.092
    C 3.000 1.250       0.050 2.375
    F 4.000             0.000 1.161

> sum(t.t)/sum(t.f)

[1] 1.4017
```

It's clear from the first table that not all combinations are present; in fact tenure class `Ci` (indigenous communities) only occurs on landscape position `A` (flat, wet areas). So any result related to this tenure class automatically affects an interpretation based on landscape position; these two are *confounded*. Similarly, tenure class `Cp` (private conservation) is

almost never found on the lowest two landscape positions, and Ce (national park) never on the steepest slopes and rarely on the flat positions.

The mean likelihood of change is 1.402, i.e. $p = 0.584$, as we've seen before (note: $0.584/(1-0.584) = 1.402$). So if there is no effect of either land use or tenure, all the non-empty cells in the last table should be 1.402. If the factors are significant, the cells will be different, but they should equal the product of the marginal likelihoods, i.e. computed for each class; another way to say this is that any pattern of values in one row or column should be repeated (perhaps at a different level) in all other rows or columns, respectively. Here we see that for land tenure (column) CA, the likelihood is highest for landscape positions A and F, but that this is exactly reversed for land tenure Nt. Hence there is an interaction between the factors. However, we don't yet know if this is statistically significant in the logistic model.

7.2 Logistic regression with several classified predictors

The S model syntax `predictand ~ predictor` may be extended to several predictors, separated by + for additive effects and * for additive effects with interactions. The presence or absence of interactions is very important in interpreting causes of land cover change, so we first see if the interactions are significant:

```
> glm.ls.t<-glm(changed ~ lspos * t, family=binomial)
> summary(glm.ls.t)

Call:
glm(formula = changed ~ lspos * t, family = binomial)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.794  -1.329   0.838   1.011   2.468

Coefficients: (4 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.39e+00   1.12e+00    1.24     0.21
lsposB      -9.44e-01   1.20e+00   -0.79     0.43
lsposC      -2.88e-01   1.30e+00   -0.22     0.83
lsposF      -6.16e-14   1.37e+00    0.00     1.00
tCe         -9.81e-01   1.44e+00   -0.68     0.50
tCi         -1.90e+00   1.19e+00   -1.59     0.11
tCp         -1.70e+01   1.03e+03   -0.02     0.99
tNt         -1.04e+00   1.13e+00   -0.92     0.36
lsposB:tCe   9.44e-01   1.52e+00    0.62     0.54
lsposC:tCe   1.05e-01   1.64e+00    0.06     0.95
lsposF:tCe         NA         NA      NA       NA
lsposB:tCi         NA         NA      NA       NA
lsposC:tCi         NA         NA      NA       NA
lsposF:tCi         NA         NA      NA       NA
lsposB:tCp   9.44e-01   1.78e+03    0.00     1.00
```

```
lsposC:tCp   1.29e+01   1.03e+03    0.01     0.99
lsposF:tCp  -1.63e-08   1.12e+03    0.00     1.00
lsposB:tNt   1.33e+00   1.21e+00    1.10     0.27
lsposC:tNt   8.03e-01   1.33e+00    0.61     0.54
lsposF:tNt  -2.00e-01   1.40e+00   -0.14     0.89

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1056.9  on 839  degrees of freedom
AIC: 1089

Number of Fisher Scoring iterations: 14
```

The key items in this output are the probability values for the interactions. Note that some interactions were not present: there are no samples with both landscape position F and land tenure Ce, so no estimate of the interaction is possible. None of the interactions were anywhere near significant; the lowest probability that the observed interaction was due to chance is for `lsposB:tNt` at $p = 0.272$. So we conclude that the two factors affect deforestation *independently*. So, we recompute the model with only additive effects:

```
> glm.ls.t<-glm(changed ~ lspos + t, family=binomial, data=dc)
> summary(glm.ls.t)

Call:
glm(formula = changed ~ lspos + t, family = binomial, data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.651  -1.338   0.856   1.025   2.541

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.622      0.344    1.81    0.071 .
lsposB         0.333      0.188    1.77    0.076 .
lsposC         0.445      0.224    1.99    0.047 *
lsposF        -0.128      0.265   -0.48    0.630
tCe           -0.605      0.371   -1.63    0.102
tCi           -1.133      0.544   -2.08    0.037 *
tCp           -4.255      0.794   -5.36  8.2e-08 ***
tNt           -0.252      0.325   -0.77    0.439
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1061.9  on 847  degrees of freedom
AIC: 1078

Number of Fisher Scoring iterations: 5
```

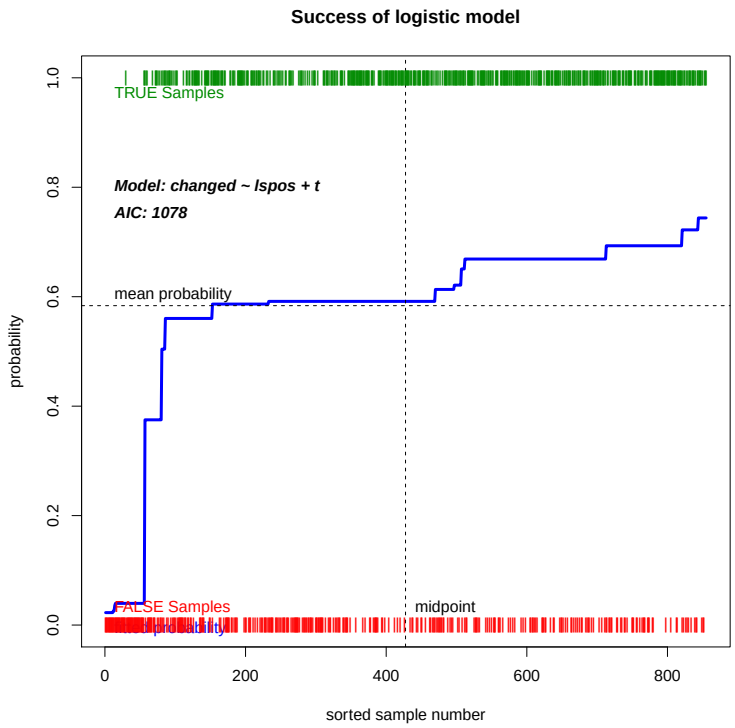
```
> d2(glm.ls.t)
```

```
[1] 0.0855
```

Over 8.5% of the variability has been explained. Tenure class `Cp` has a very highly significant coefficient, as in the single-factor model (§6.3); class `Ci` has a significant coefficient. Landscape positions `C`, `B`, and `A` are significant. Interestingly, `C` is the most significant here, unlike in the single factor model of §6.1. Notice that the AIC is lower than for the model with interactions, although the residual deviance is slightly higher. This shows that the model with interactions was *over-fitting* the data.

We now plot the success of this model:

```
> logit.plot(glm.ls.t)
```



The relation is clearly much better than with any single predictor. The plot looks like a combination of tenure as single predictor, with the smaller steps from the model with landscape position as single predictor superimposed.

7.3 Prediction from multiple-predictor classified model

The predicted probability of change for a sample from any *covariate pattern* (combination of classes) is computed from the inverse logistic equation (Equation 2), with the coefficients for the appropriate classes.

The coefficient for each class is extracted from the model and used in the inverse logit function. For the two-classifier model:

$$p = \frac{1}{1 + e^{-(\beta_0 + \sum_i \beta_{c1_i} \delta_{c2_i} + \sum_j \beta_{c2_j} \delta_{c2_j})}} \tag{6}$$

where:

- β_0 is the log-likelihood of change for samples with the covariate pattern of both first-named classes, i.e. the intercept, (here, tenure class CA and landscape position A);
- β_{c1_i} is the log-likelihood of change associated with the ith class for the first factor (its difference from the intercept);
- δ_{c1_i} is the Kronecker delta for class $c1_i$ of the first factor, i.e. 1 if the observation is in the class, 0 otherwise. At most one of the δ_{c1_i} will be 1; none will be 1 if the observation is in the first class.
- Thus the $\sum_i$ of the products $\beta_{c1_i} \delta_{c2_i}$ will be the value of the appropriate coefficient;
- $\beta_{c2_j}, \delta_{c2_j}, \sum_j$ have the same meaning as $\beta_{c1_i}, \delta_{c1_i}, \sum_i$, but for the second, instead of the first, factor; to avoid confusion we use the index j instead of i for this factor.

These will be different for every combination of predictor classes. For example, for landscape position class A and land tenure class Ci we compute:

```
> glm.ls.t$coefficients["(Intercept)"]

(Intercept)
    0.62197

> glm.ls.t$coefficients["tCi"]

    tCi
-1.1328

> glm.ls.t$coefficients["(Intercept)"] + glm.ls.t$coefficients["tCi"]

(Intercept)
   -0.51083

> logit.inv(glm.ls.t$coefficients["(Intercept)"] +
+                               glm.ls.t$coefficients["tCi"])

(Intercept)
      0.375
```

The intercept represents the covariate pattern for the first level of both factors, i.e. landscape position A and tenure class CA. Here we *add* the

coefficient for tenure class `Ci`, because the model is additive (no interactions), before converting the log-likelihood (here, −0.511) to a probability (here, 0.375). So, the probability of change for this covariate pattern (combination of factors) as predicted by the model is 3/8. The previous figure shows the 24 samples with this pattern as a narrow step at abscissa 0.375.

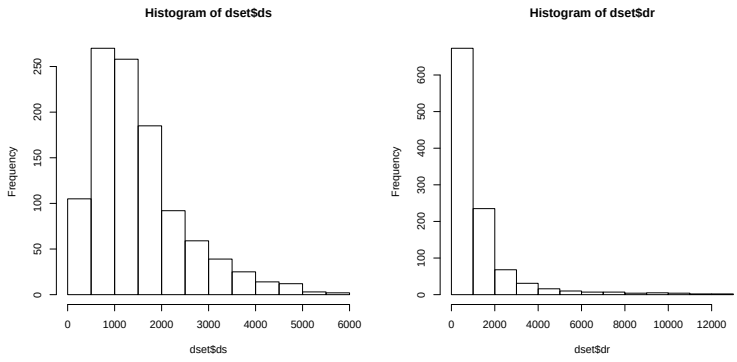
In practice, R code will be used to extract the correct coefficient, as shown for the single-predictor model in §6.5.

8 The logistic model of change for continuous predictors

We also have two possible continuous predictors of land use change: distance to nearest road and to nearest settlement. These can be used as predictors, using exactly the same syntax.

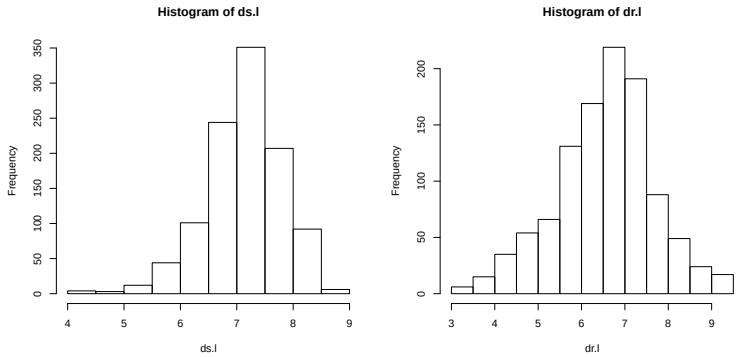
Before we use a continuous predictor, however, it should be approximately symmetric. This helps ensure the numerical stability of the solution, and in particular that the residuals are well-distributed. So we first examine the distribution of the predictors `ds` (distance to settlement) and `dr` (distance to road) in the full dataset, object `dset`:

```
> summary(dset$ds); summary(dset$dr)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     44     813    1290    1510    1930    5750

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0     322     713    1210    1410   12500

> par(mfrow=c(1,2))
> hist(dset$ds); hist(dset$dr)
> par(mfrow=c(1,1))
```



The predictors are quite strongly (positively) skewed (mean > > median). Therefore we first transform them to remove the skew; the log transform is appropriate for this. In both cases we add the size of one plot, 30 m, to avoid samples with zero distance, where the logarithm is undefined:

```
> ds.l<-log(dset$ds+30); dr.l<-log(dset$dr+30)
> summary(ds.l); summary(dr.l)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.30    6.74    7.18    7.13    7.58    8.66

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.40    5.86    6.61    6.54    7.27    9.44

> par(mfrow=c(1,2))
> hist(ds.l); hist(dr.l)
> par(mfrow=c(1,1))
```



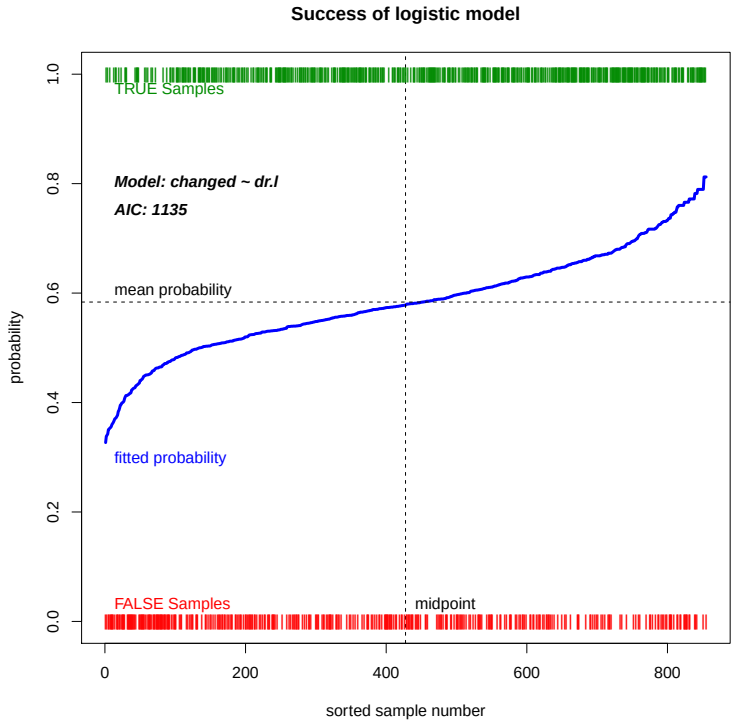
Comparing the four histograms, the effect of the transformation is obvious: after transformation there is almost no skew and the variables are symmetric. We will modify both data frames (the full dataset and the “deforestation" subset) to include these fields. To access the added variables without naming the currently-attached dataframe dc, we must first detach and then re-attach it to the search path.

```
> dset <- cbind(dset, dr.l=log(dset$dr+30), ds.l=log(dset$ds+30))
> dc <- cbind(dc, dr.l=log(dc$dr+30), ds.l=log(dc$ds+30))
> str(dset)

'data.frame':        1064 obs. of  9 variables:
 $ cov    : Factor w/ 9 levels "CC","CN","CO",..: 1 1 3 6 4 9 2 6 1 1 ...
 $ dr     : num  2924 2535 2146 2442 2831 ...
 $ ds     : int  4853 5242 4957 4690 5079 4283 3999 3732 4121 3976 ...
 $ t      : Factor w/ 5 levels "CA","Ce","Ci",..: 5 5 3 3 3 5 5 5 5 5 ...
 $ lspos  : Factor w/ 4 levels "A","B","C","F": 1 1 1 1 1 1 1 1 1 1 ...
 $ text   : Factor w/ 3 levels "L","L'","LC": 3 3 3 3 3 3 3 3 3 3 ...
 $ changed: logi  FALSE FALSE TRUE TRUE TRUE FALSE ...
 $ dr.l   : num  7.99 7.85 7.69 7.81 7.96 ...
 $ ds.l   : num  8.49 8.57 8.51 8.46 8.54 ...

> detach(dc); attach(dc)
```

Now we can compute a one-predictor GLM from the log of distance to roads:

```
> glm.dr <- glm(changed ~ dr.l, family=binomial, data=dc)
> summary(glm.dr)
```

```
Call:
glm(formula = changed ~ dr.l, family = binomial, data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.829  -1.245   0.859   1.049   1.472

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.6969     0.4475    6.03  1.7e-09 ***
dr.l         -0.3623     0.0675   -5.37  7.9e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1130.6  on 853  degrees of freedom
AIC: 1135

Number of Fisher Scoring iterations: 4

> d2(glm.dr)

[1] 0.0264

> logit.plot(glm.dr)
```



This model is also quite poor; it explains less than 3% of the variation, although the predictor is very highly significant. The plot shows a very

"flat" logistic-shaped curve; it doesn't come close to either 0 or 1, meaning that no sample is predicted with any certainty. The distributions of the changed and unchanged samples on the horizontal axes is only slightly different.

8.1 Prediction from a single-predictor continuous model

The predicted probability of change for a sample at any distance is computed from the inverse logistic equation (Equation 2) with the log-likelihood computed by the linear equation found by the model:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \tag{7}$$

where β_0 is the intercept (log-likelihood of change at zero distance), β_1 is the slope of the log-likelihood, and x is the distance of the sample to be predicted. For example, for zero distance (actually 30 m), we compute from the intercept only:

```
> glm.dr$coefficients

(Intercept)        dr.l
    2.69685    -0.36234

> glm.dr$coefficients["(Intercept)"]

(Intercept)
     2.6969

> glm.dr$coefficients[1]

(Intercept)
     2.6969

> logit.inv(glm.dr$coefficients["(Intercept)"])

(Intercept)
    0.93684

> logit.inv(glm.dr$coefficients[1])

(Intercept)
    0.93684
```

The probability of change for a sample at a road is predicted by the model to be quite high, 0.937. Note that we can select a coefficient either by its name or its position in the coefficient array; the second way is shorter but the first way is clearer.

For any distance greater than zero, we must use the linear combination of intercept and coefficient times distance, i.e. the same formula we would use in a linear model, but then as an argument to the inverse logistic transformation. From the slope (-0.362) we can see that increasing distance from road decreases the likelihood of change. For example a

sample 3 000 m from the road has a predicted probability of change of 0.450:

```
> glm.dr$coefficients["dr.l"]

    dr.l
-0.36234

> glm.dr$coefficients[2]

    dr.l
-0.36234

> logit.inv(glm.dr$coefficients["(Intercept)"] +
+       log(3000+30)*glm.dr$coefficients["dr.l"])

(Intercept)
    0.44825

> logit.inv(glm.dr$coefficients[1] +
+       log(3000+30)*glm.dr$coefficients[2])

(Intercept)
    0.44825
```

Note that we had to take the logarithm of the distance plus 30 m, since the coefficient was computed on a variable so transformed.

9 Change related to several continuous variables

Just as in the case of classified predictors, we can use several continuous predictors in the same predictive model. The first task is to assess their relation; perhaps one is superfluous.

9.1 Relation among continuous predictors

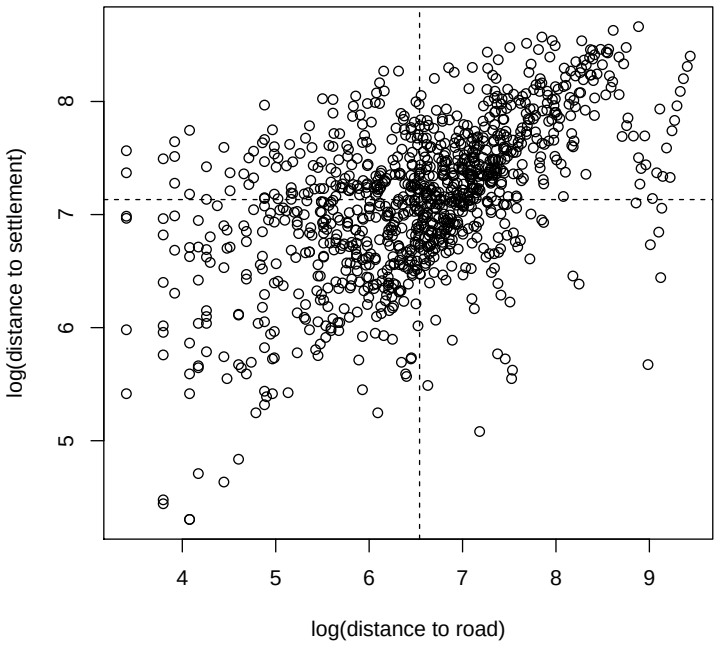
We can visualise the relation between continuous predictors with a scatterplot. If two predictors are not too skewed[6], we can test their relation with the *Pearson's product-moment correlation* $s_{xy}/s_x s_y$.

In this example we have two continuous predictors: distance to roads and to settlements. The log-transform of these resulted in two fairly symmetrical and compact variables; see the histograms in §8. We plot their relation and assess it numerically:

```
> plot(dr.l, ds.l,
+       xlab="log(distance to road)",
+       ylab="log(distance to settlement)")
> abline(h=mean(ds.l),lty=2); abline(v=mean(dr.l),lty=2)
> cor.test(dr.l,ds.l)
```

[6] Actually, they should be bivariate normally distributed, but in practice this measure is somewhat robust

```
        Pearson's product-moment correlation

data:  dr.l and ds.l
t = 21.3, df = 1060, p-value <2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.50394 0.58821
sample estimates:
    cor
0.54746
```



The figure clearly shows that there is some relation; note that the lower-right and upper-left quadrants (defined by the sample means) have many fewer points than the other two.

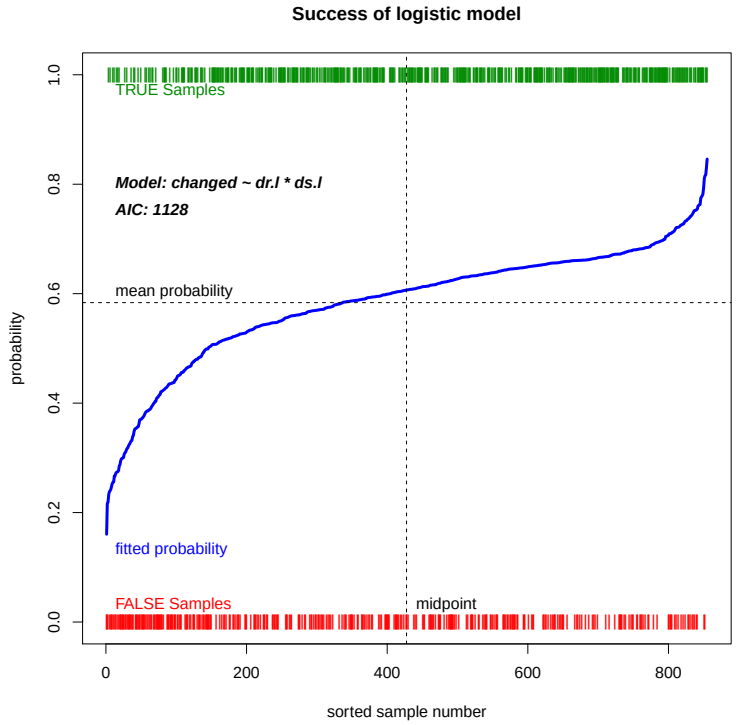
The correlation coefficient is definitely non-zero; note we tested against zero since we had no pre-conception of the relation. However, the coefficient of determination $r^2 \approx (0.547)^2 \approx 0.30$, so the two predictors are not so closely related that one could be discarded. In particular, there are some points that are close to a road but far from a settlement (upper-left quadrant).

> **Note:** The non-parametric Spearman's rank correlation is 0.544, almost identical to the parametric Pearson's correlation of 0.547; this shows that the assumption of bivariate normality was not seriously violated.

9.2 Multiple logistic regression

The S model language works the same for several continuous predictors as for several classified ones. We try the model with interactions first:

```
> glm.dr.ds<-glm(changed ~ dr.l*ds.l, family="binomial", data=dc)
> summary(glm.dr.ds)

Call:
glm(formula = changed ~ dr.l * ds.l, family = "binomial", data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.839  -1.280   0.878   1.004   1.707

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -7.5546     3.4234   -2.21   0.0273 *
dr.l          1.4129     0.5505    2.57   0.0103 *
ds.l          1.4188     0.4919    2.88   0.0039 **
dr.l:ds.l    -0.2441     0.0768   -3.18   0.0015 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1119.8  on 851  degrees of freedom
AIC: 1128

Number of Fisher Scoring iterations: 4

> d2(glm.dr.ds)

[1] 0.0357

> logit.plot(glm.dr.ds)
```



The AIC is lower than for the model using only distance to roads, so this is a better model. Very importantly, the interaction term `dr.l:ds.l` is highly significant. From the coefficients we can see that increasing distance to roads and settlements, each taken individually, increase the likelihood that a site will be changed, in contrast to the single-predictor models, and in contrast to the hypothesis.

However, the negative interaction term shows that as distance from both roads and settlements together increases, the likelihood decrease, in agreement with the hypothesis. Numerically, the interaction term is smaller, but recall that it applies to the *product* of the two distances, so at any substantial distance it will overwhelm the coefficients for the individual distances.

Because the interaction is significant, we do not fit the additive model. The plot of model success here is much better than the previous one; in particular some samples are fitted with probabilities < 0.2 and some > 0.8. These are due to the interaction term, which accounts for points close to (or far from) both roads and settlements.

10 Change related to continuous and classified variables

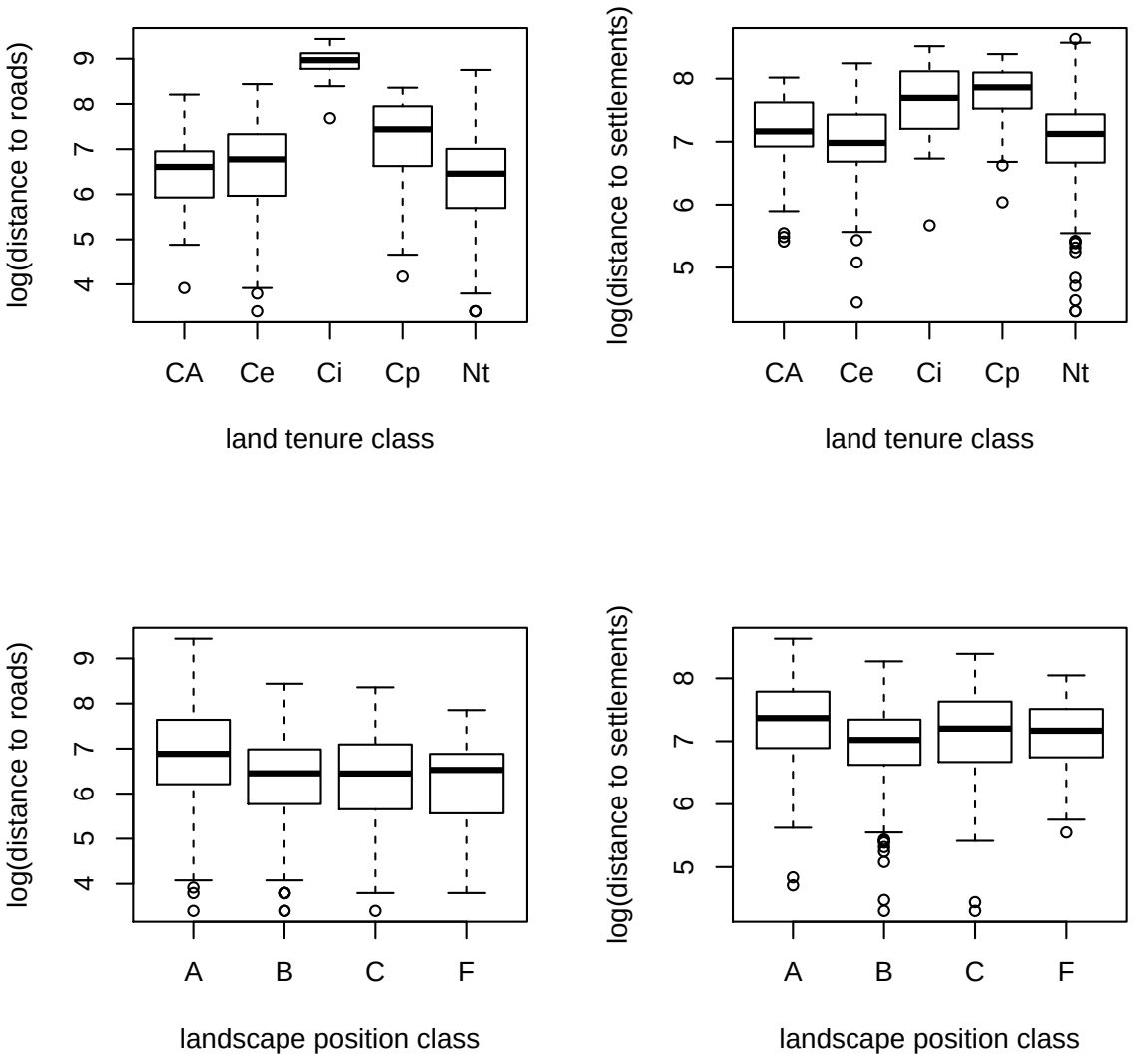
The generalised linear model implemented by the `glm()` method can use both continuous and classified variables together, exactly as the general linear model implemented by the `lm()` method. This leads to greater

predictive power but also difficulties of interpretation and model selection.

10.1 Relation between classified and continuous predictors

To visualise the relation of a continuous to a classified predictor, we produce a *classified boxplot*:

```
> par(mfrow=c(2,2))
> plot(dr.l~t, xlab="land tenure class",
+   ylab="log(distance to roads)", data=dc)
> plot(ds.l~t, xlab="land tenure class",
+   ylab="log(distance to settlements)", data=dc)
> plot(dr.l~lspos, xlab="landscape position class",
+   ylab="log(distance to roads)", data=dc)
> plot(ds.l~lspos, xlab="landscape position class",
+   ylab="log(distance to settlements)", data=dc)
> par(mfrow=c(1,1))
```



The boxplots show some strong interactions between classified and continuous predictors. In particular, tenure class `Ci` (indigenous communities) is only found far from roads, and landscape position `F` (very steep

slopes) is not found near settlements.

To test whether at least one class has a different mean value of the continuous variable from the other, we can compute the *one-way ANOVA*; however it is more informative to compute the *linear model* and interpret its summary:

```
> lm.rt <- lm(dr.l~t, data=dc); summary(lm.rt)

Call:
lm(formula = dr.l ~ t, data = dc)

Residuals:
   Min     1Q Median     3Q    Max
-3.135 -0.573  0.155  0.697  2.446

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.4049     0.1406   45.55  < 2e-16 ***
tCe           0.1308     0.1691    0.77     0.44
tCi           2.4947     0.2469   10.10  < 2e-16 ***
tCp           0.7934     0.1935    4.10  4.5e-05 ***
tNt          -0.0991     0.1462   -0.68     0.50
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.994 on 850 degrees of freedom
Multiple R-squared:  0.183,        Adjusted R-squared:  0.179
F-statistic: 47.7 on 4 and 850 DF,  p-value: <2e-16
```

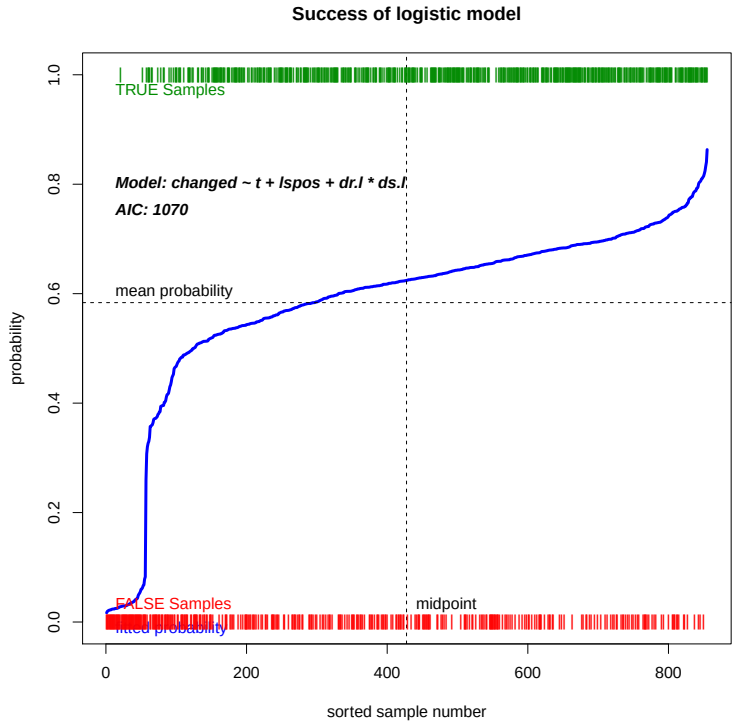
The linear model shows a significant intercept (class `CA`, agricultural colonies); this just means that the mean distance (6.4 km) from the roads is not zero. Classes `Ci` (indigenous communities) and `Cp` (private conservation area) are significantly further from the roads than class `CA`, the others (national park and untitled land) not.

10.2 The combined generalised linear model in R

In previous sections we've seen that landscape position, land tenure, and the distances to roads and settlements (with their interaction) are significant predictors of deforestation. We can combine these all in one model with the S model syntax.

```
> glm.t.ls.drs <- glm(changed ~ t + lspos + dr.l*ds.l, family=binomial,
+    data=dc)
> summary(glm.t.ls.drs)

Call:
glm(formula = changed ~ t + lspos + dr.l * ds.l, family = binomial,
    data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
```

```
-1.836  -1.273   0.819   0.967   2.674

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.9026     3.5701   -1.37    0.170
tCe          -0.5283     0.3735   -1.41    0.157
tCi          -0.4206     0.5799   -0.73    0.468
tCp          -3.9814     0.8003   -4.98  6.5e-07 ***
tNt          -0.2779     0.3286   -0.85    0.398
lsposB        0.1906     0.1981    0.96    0.336
lsposC        0.2772     0.2335    1.19    0.235
lsposF       -0.3212     0.2735   -1.17    0.240
dr.l          0.8082     0.5703    1.42    0.156
ds.l          1.0688     0.5165    2.07    0.039 *
dr.l:ds.l    -0.1556     0.0807   -1.93    0.054 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1047.8  on 844  degrees of freedom
AIC: 1070

Number of Fisher Scoring iterations: 5

> d2(glm.t.ls.drs)

[1] 0.0977

> logit.plot(glm.t.ls.drs)
```



This is the lowest AIC we've seen, and the total variance explained is almost 10%. As before, tenure is the most significant predictor, with a very negative coefficient for class Cp; this shows that the private conservation area is in fact well-protected. Distance to settlement and its interaction with distance to roads are also significant. But no landscape position is now a significant predictor; the association with one of the other predictors has removed its significance.

Therefore we re-compute the model, leaving out this factor.

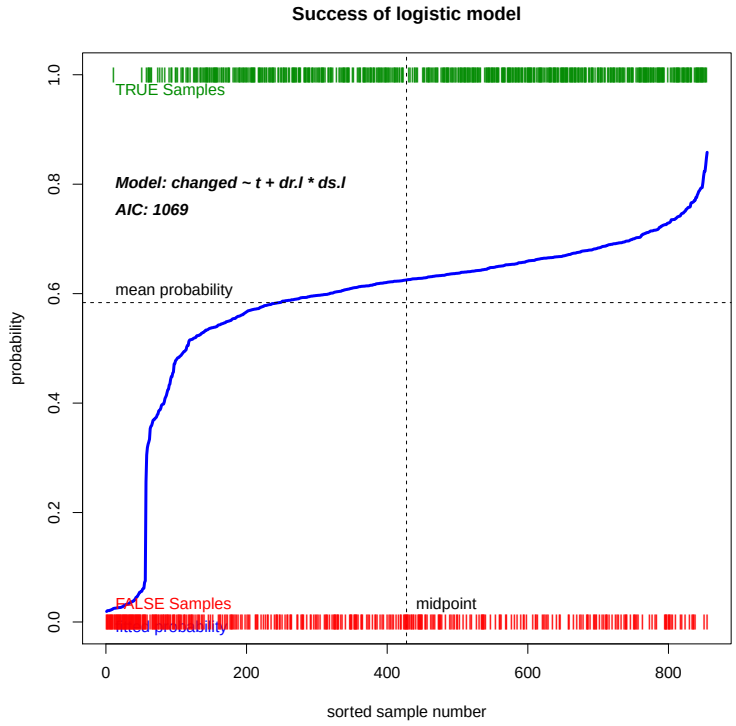
```
> glm.t.drs <- glm(changed ~ t + dr.l*ds.l, family=binomial,
+     data=dc)
> summary(glm.t.drs)

Call:
glm(formula = changed ~ t + dr.l * ds.l, family = binomial, data = dc)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.977  -1.322   0.840   0.966   2.721

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.5029     3.5565   -1.27    0.205
tCe          -0.4150     0.3675   -1.13    0.259
tCi          -0.4836     0.5697   -0.85    0.396
tCp          -3.8909     0.7897   -4.93  8.3e-07 ***
tNt          -0.2952     0.3229   -0.91    0.361
dr.l          0.7896     0.5659    1.40    0.163
```

```
ds.l          1.0291     0.5132    2.01    0.045 *
dr.l:ds.l    -0.1536     0.0799   -1.92    0.055 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1161.3  on 854  degrees of freedom
Residual deviance: 1052.7  on 847  degrees of freedom
AIC: 1069

Number of Fisher Scoring iterations: 5
```

```
> d2(glm.t.drs)
```

```
[1] 0.0934
```

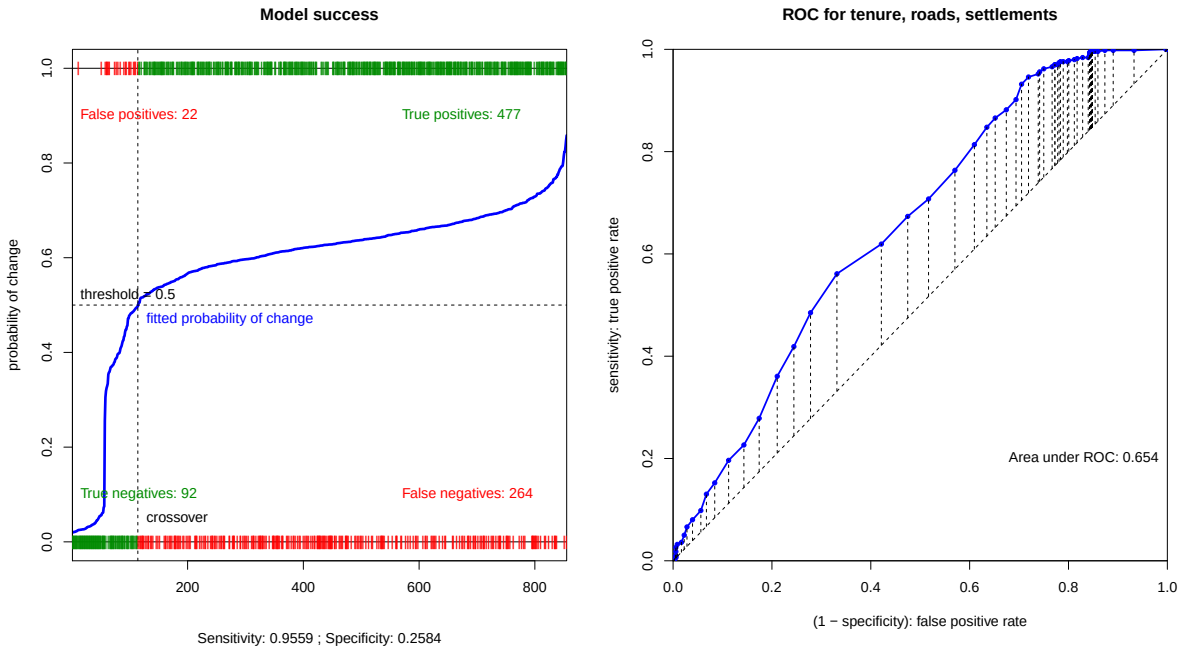
```
> logit.plot(glm.t.drs)
```



The AIC is a bit lower; the total variance explained is a bit lower also, about 9.3%. This is the best parsimonious model. The main predictive factors are land tenure class Cp (strongly reduces likelihood of deforestation), distance to settlements (increases likelihood), and the synergistic relation between distances to roads and settlements (somewhat reduces likelihood as both increase together).

This final model can be evaluated with the area under the ROC curve. We approximate the curve with many steps because it is continuous.

```
> par(mfrow=c(1,2))
> logit.plot.quad(glm.t.drs)
> r <- logit.roc(glm.t.drs, steps=100)
> logit.roc.area(r)

[1] 0.65398

> logit.roc.plot(r, "ROC for tenure, roads, settlements")
> par(mfrow=c(1,2))
```



This area (0.654) is still not very good. We conclude that our set of explanatory variables is only marginally successful in explaining deforestation.

10.3 Prediction from a combined model

This combines the approaches of §7.3 and §8.1. The coefficient for a class can be extracted from the model and used in the inverse logit function along with the slopes of any continuous predictors. Formally:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 \cdots + \sum_i \beta_{c1_i} \delta_{c2_i} + \sum_j \beta_{c2_j} \delta_{c2_j})}} \tag{8}$$

where:

- β_0 is the log-likelihood of change for samples with the covariate pattern of both first-named classes, and values of 0 for the continuous predictors, i.e. the intercept;

- β_1 is the slope of the log-likelihood for continuous predictor with value x_1 (and similarly for x_2 etc.);
- β_{c1_i} is the log-likelihood of change associated with the ith class for the first factor (its difference from the intercept);
- δ_{c1_i} is the Kronecker delta for class $c1_i$ of the first factor, i.e. 1 if the observation is in the class, 0 otherwise. At most one of the δ_{c1_i} will be 1; none will be 1 if the observation is in the first class.
- Thus the $\sum_i$ of the products $\beta_{c1_i}\delta_{c2_i}$ will be the value of the appropriate coefficient;
- $\beta_{c2_j}, \delta_{c2_j}, \sum_j$ have the same meaning as $\beta_{c1_i}, \delta_{c1_i}, \sum_i$, but for the second, instead of the first, factor; to avoid confusion we use the index j instead of i for this factor.

For example, we can calculate the probability of deforestation for a site 1 500 m from a road, 2 500 m from a settlement, with tenure type `Nt` as follows:

```
> attach(glm.t.drs)
> coefficients

(Intercept)         tCe         tCi         tCp         tNt
   -4.50289    -0.41499    -0.48360    -3.89087    -0.29516
       dr.l        ds.l   dr.l:ds.l
    0.78956     1.02909    -0.15361

> logit.inv(coefficients["(Intercept)"]
+    + log(1500+30)*coefficients["dr.l"]
+    + log(2500+30)*coefficients["ds.l"]
+    + log(1500+30)*log(2500+30)*coefficients["dr.l:ds.l"]
+    + coefficients["tNt"]
+ )

(Intercept)
    0.55705

> detach(glm.t.drs)
```

The probability of deforestation is about 0.56 in this case. Note that the interaction coefficient is multiplied by the product of the two distances. In practice, R code will be used to extract the correct coefficient, as shown for the single-predictor model in §6.5.

11 Polytomous response variables

To this point we have been considering a **dichotomous** ("two outcomes") response variable: either something has occurred (e.g. deforestation) or not. There are also situations where the response is **polytomous** ("many outcomes") , i.e. there are several possible outcomes.

For example, starting with closed forest there are three possibilities:

CC : closed → closed forest; no change

CO : closed → open forest (partial deforestation)

CN : closed → no forest (deforestation)

We can see the number of pixels in each of these classes:

```
> table(dc$cov)

 CC  CN  CO  NC  NN  NO  OC  ON  OO
356 279 220   0   0   0   0   0   0
```

Thus, 356 did not change, 220 were partially deforested, and 279 were fully deforested.

It is possible to simultaneously assess polytomous results. Fox [2, §15.2][3, §5.2.2] describes two approaches:

1. **Nested dichotomies**: a series of binary models, based on successive divisions of the classes so that there are only two responses at each level;
2. **Multinomial logistic model**: one category is treated as the base, and the probabilites of the others are simultaneously evaluated with respect to the base.

We will explore the second method here; the "no change" class CC serves logically as the base class.

The nested model is appropriate when the classes form a natural hierarchy. Here that would also be possible: the higher level "any deforestation?" (binary response 'no'; 'yes'), then the second level for deforested 'yes' being "complete deforestation?" ('no', i.e. partial; 'yes'); however we do not pursue it further here.

11.1 Multinomial logistic model: Theory

The mathematical formulation of the multinomial logistic model is more complicated than the binomial model. We want to model the probability π_{ij} that observation i is in each jth class of the m response classes $j = 1 \ldots m$. The first response class $j = 1$ is taken as the base class; so the base probability π_{i1} is computed as the residual probability after the other classes $\pi_{i2} \ldots \pi_{im}$ have been modelled.

Thus the model has $k + 1$ coeffiecients for each of the $j = m - 1$ classes (leaving out the base class): one intercept α_j and one "slope" for each predictor (continuous or each class of a classified predictor) β_{lj}, where $l = 1 \ldots k$ is a column in the model matrix.

The fitted probabilites are then:

$$\pi_{ij} = \frac{e^{(\alpha_j+\beta_{1j}x_{i1}+\cdots\beta_{kj}x_{ik})}}{1+\sum_{l=2}^{m} e^{(\alpha_j+\beta_{1l}x_{i1}+\cdots\beta_{kl}x_{ik})}}, j = 2,\ldots,m \quad (9)$$

$$\pi_{i1} = 1 - \sum_{j=2}^{m} \pi_{ij} \quad (10)$$

This set of equations is fitted by maximum likelihood; see Fox [2, §15.2.1] for details.

The fitted α and β can then be used to assess the **log-odds** of an observation being classified in each class, **relative to the base class**. That is, what is the chance that, instead of the base class (in our example, no deforestation), the observation is in another class (in our example, partial or complete deforestation)? Note these are **not** the odds of being in that class, only the odds relative to the base class: how much more likely is each alternative, compared to the base?

The log-odds are computed as:

$$\ln \frac{\pi_{ij}}{\pi_{i1}} = \alpha_j + \beta_{1j}x_{i1} + \ldots + \beta_{kj}x_{ik}, j = 2,\ldots,m \quad (11)$$

Note that if $m = 2$ this reduces to the binomial logit model, as we have used it up till now.

So, once we fit the model, we can predict the odds of some class, relative to the base. We can also predict the odds between any two classes, by combining the above equation for the two classes j and l:

$$\ln \frac{\pi_{ij}}{\pi_{il}} = \ln\left(\frac{\pi_{ij}/\pi_{i1}}{\pi_{il}/\pi_{i1}}\right) \quad (12)$$

$$= \ln \frac{\pi_{ij}}{\pi_{i1}} - \ln \frac{\pi_{il}}{\pi_{i1}} \quad (13)$$

$$= (\alpha_j - \alpha_l) + (\beta_{1j} - \beta_{1l})x_{i1} + \ldots + (\beta_{kj} - \beta_{kl})x_{ik} \quad (14)$$

To recover the actual odds, the inverse logistic transformation is used, as explained in §6. In practice we are usually interested in the probability of all the classes (which of course sum to 1); these are provided by the `predict` method, as explained in §11.4.

This is a lot of information to interpret properly!

11.2 Multinomial logistic model: R implementation

The `glm` method we have used up until now can not fit multinomial models. Instead, we use the `multinom` method of the `nnet` "Neural networks" package; this is part of the `MASS` "Modern Applied Statistics with S" library provided with the R base distribution; we will also used some methods from `MASS`.

```
> require(MASS)
> require(nnet)
```

The `multinom` method converts the problem to a neural network and then solves it with the `nnet` method.

Before computing the model, we must ensure that the base (reference) class is listed first; in this case it already is (because of the default alphabetic order). If necessary, levels can be reordered with the `relevel` method.

We compute a multinomial logistic model using the same predictors as the best parsimonious binomial model, as determined in §10.2.

```
> mlm.t.drs <- multinom(cov ~ t + dr.l*ds.l, data=dc)

# weights:  27 (16 variable)
initial  value 939.313507
iter  10 value 867.829240
iter  20 value 860.261576
iter  30 value 859.955097
final  value 859.955021
converged

> summary(mlm.t.drs, Wald=T)

Call:
multinom(formula = cov ~ t + dr.l * ds.l, data = dc)

Coefficients:
   (Intercept)      tCe     tCi     tCp       tNt    dr.l
CN     -5.7734 -0.55500 -1.0322 -3.7719 -0.093916 0.95997
CO     -5.4861 -0.30212 -0.3461 -4.0180 -0.524883 0.74894
      ds.l dr.l:ds.l
CN 1.16729  -0.18727
CO 0.99354  -0.13540

Std. Errors:
   (Intercept)     tCe     tCi    tCp     tNt    dr.l    ds.l
CN      4.1135 0.43291 0.87336 1.0770 0.37055 0.67089 0.59321
CO      4.4736 0.41566 0.63498 1.0742 0.36679 0.70269 0.64206
   dr.l:ds.l
CN  0.094958
CO  0.098928

Value/SE (Wald statistics):
   (Intercept)      tCe      tCi     tCp      tNt   dr.l   ds.l
CN     -1.4035 -1.28202 -1.18186 -3.5022 -0.25345 1.4309 1.9677
CO     -1.2263 -0.72685 -0.54506 -3.7404 -1.43102 1.0658 1.5474
   dr.l:ds.l
CN   -1.9721
CO   -1.3686

Residual Deviance: 1719.9
AIC: 1751.9
```

11.3 Multinomial logistic model: Interpretation

The summary gives the coefficients, their standard errors, and the Wald statistics. This latter is printed if the `Wald=T` optional argument to the `summary` method is specified. It is the coefficient divided by its standard error. Thus, if the relative error is high, the Wald statistic is small. This gives an idea of the significance of each predictor: the greater the absolute value, the more significant. Note that the sign of the Wald statistic is the same as that of the coefficient, and thus gives the direction of the effect: increase or decrease in probability due to the predictor.

Here we see that for both deforestation classes, the distances from roads and settlements increase the probability of deforestation, and more so for the complete than the partial deforestation:

```
> coefficients(mlm.t.drs)[, c("dr.l", "ds.l")]

      dr.l    ds.l
CN 0.95997 1.16729
CO 0.74894 0.99354
```

This seems to contradict the hypothesis that roads and settlements lead to deforestation; however recall the discussion in §9.2 on multiple logistic regression: these *positive* coefficients are in fact corrections for the *negative* coefficient for the interaction of these two, which is numerically much larger (recall, the values are multipled):

```
> coefficients(mlm.t.drs)[, "dr.l:ds.l"]

      CN       CO
-0.18727 -0.13540
```

So at a combination of far distances (to settlements and roads), the probability of either kind of deforestation is reduced; the individual coefficients correct this for the case where a pixel is close to one but not the other.

The Wald statistic shows that it is significant:

```
> summary(mlm.t.drs, Wald=T)$Wald.ratios[, c("dr.l", "ds.l", "dr.l:ds.l")]

     dr.l   ds.l dr.l:ds.l
CN 1.4309 1.9677   -1.9721
CO 1.0658 1.5474   -1.3686
```

Considering the land tenure predictor, we again see the strong effect of the conservation area (class `Cp`) on the deforestation probabilities:

```
> coefficients(mlm.t.drs)[, "tCp"]

     CN      CO
-3.7719 -4.0180
```

```
> summary(mlm.t.drs, Wald=T)$Wald.ratios[, "tCp"]

     CN      CO
-3.5022 -3.7404
```

This class has by far the largest coefficient and Wald ratio in the model.

The only major contrast in the factors leading to partial vs. complete deforestation seems to be tenure classes `Ci` "Indigenous communities" and `Nt` "No established title":

```
> coefficients(mlm.t.drs)[, c("tCi", "tNt")]

       tCi       tNt
CN -1.0322 -0.093916
CO -0.3461 -0.524883
```

Other factors being equal, on indigenous land deforestation tends to be complete and on land with no title partial.

As with all R models, we can extract the fitted values with the `fitted` method; here the fits are the probabilities of each response:

```
> head(fitted(mlm.t.drs))

        CC       CN      CO
1  0.54629 0.202551 0.25116
2  0.53192 0.210648 0.25743
3  0.55686 0.097412 0.34573
7  0.43083 0.288256 0.28091
9  0.50742 0.231649 0.26093
10 0.54804 0.204711 0.24724
```

Notice that the probabilities of the three classes for each case sum to 1.

We see that the first pixel has p=0.5463 of remaining as closed forest, p=0.2512 of partial deforestation, and p=0.2512 of complete deforestation. In fact this pixel remained in forest:

```
> dc[1,"cov"]

[1] CC
Levels: CC CN CO NC NN NO OC ON OO
```

The highest probability matches reality in this case.,

11.4 Multinomial logistic model: Prediction

The probabilities of each class, for any combination of predictor values, can be computed with the `predict` generic method, which in this case is implemented by `predict.multinom` in package `nnet`.

We first set up a range of cases to predict, for the best model. There are two continuous and one classified predictor.

The range of the continuous variables is:

```
> range(dc$dr.l)

[1] 3.4012 9.4372

> range(dc$ds.l)

[1] 4.3041 8.6289
```

We make a dataframe with all the combinations of land tenure and a series of distances, and then predict on it with the fitted model:

```
> to.predict <- expand.grid(t = levels(dc$t),
+                           dr.l = seq(3.4, 9.4, by=.2),
+                           ds.l = seq(4.3, 8.8, by=.2))
> head(to.predict, 10)

    t dr.l ds.l
1  CA  3.4  4.3
2  Ce  3.4  4.3
3  Ci  3.4  4.3
4  Cp  3.4  4.3
5  Nt  3.4  4.3
6  CA  3.6  4.3
7  Ce  3.6  4.3
8  Ci  3.6  4.3
9  Cp  3.6  4.3
10 Nt  3.6  4.3

> p.fit <- predict(mlm.t.drs, to.predict, type="probs")
> head(p.fit, 10)

         CC       CN        CO
1  0.43108 0.343175 0.2257416
2  0.54226 0.247818 0.2099182
3  0.60458 0.171448 0.2239721
4  0.97301 0.017822 0.0091657
5  0.49152 0.356208 0.1522767
6  0.42328 0.347551 0.2291708
7  0.53430 0.251851 0.2138489
8  0.59683 0.174569 0.2285992
9  0.97217 0.018366 0.0094682
10 0.48360 0.361489 0.1549065
```

Notice (again) that the probabilities of the three classes for each prediction point sum to 1.

The relative probabilities of any situation can be extracted in the same way. For example, what are the probalities for agricultural colony land 2 km from a settlement and 1 km from a road?

```
> predict(mlm.t.drs, data.frame(t="CA",
+                               dr.l=log(1030),
+                               ds.l=log(2030)), type="probs")
```

```
     CC      CN      CO
0.33120 0.29433 0.37447
```

(Recall we added 30 m to both distance variables before converting to logarithms.)

In this case the probabilities are quite similar, with a slight preference for complete deforestation.

11.5 Multinomial logistic model: Visualization

This is a complex situation to visualize, because there are three outcomes and three predictors.

11.5.1 Probabilities vs. predictors

We graph the relative probabilities for various combinations, using the `p.fit` object. Because one of the predictors is a class, we have to show the results separately for each class. In a 2D plot, we also have to fix one of the distances.

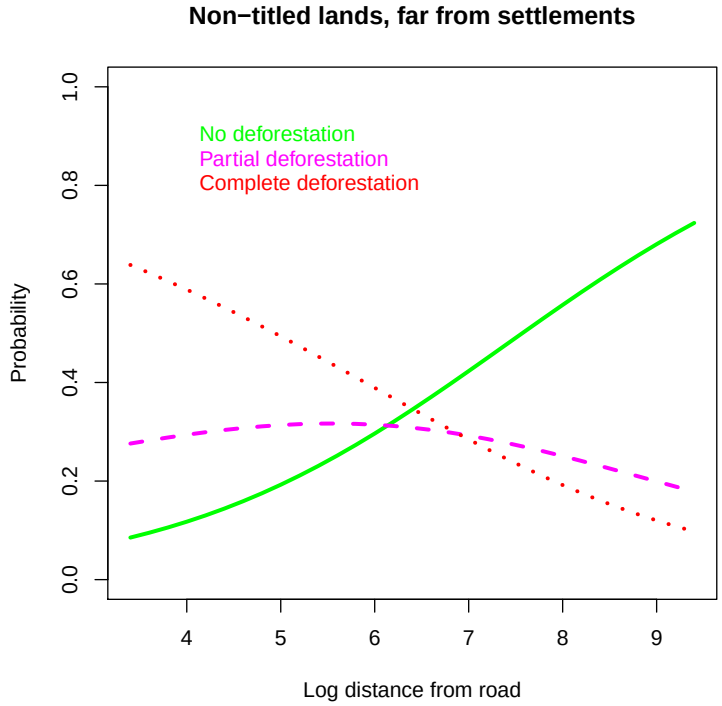
We illustrate this with the probabilities of the three classes for the furthest distance from settlements (`ds.l == 8.7`) and tenure class "no title" (`t == Nt`). We set up a frame with the probabilities for just this situation:

```
> to.graph <- subset(to.predict, (t=="Nt") & (ds.l == 8.7))
> str(to.graph)

'data.frame':        31 obs. of  3 variables:
 $ t   : Factor w/ 5 levels "CA","Ce","Ci",..: 5 5 5 5 5 5 5 5 5 5 ...
 $ dr.l: num  3.4 3.6 3.8 4 4.2 4.4 4.6 4.8 5 5.2 ...
 $ ds.l: num  8.7 8.7 8.7 8.7 8.7 8.7 8.7 8.7 8.7 8.7 ...

> g.fit <- predict(mlm.t.drs, to.graph, type="probs")
```

We first set up the axes (distance on the x-axis and probility on the y-axis) and then draw curves for the three outcomes.

For distance from roads, the limits are `3.4` to `9.4`:

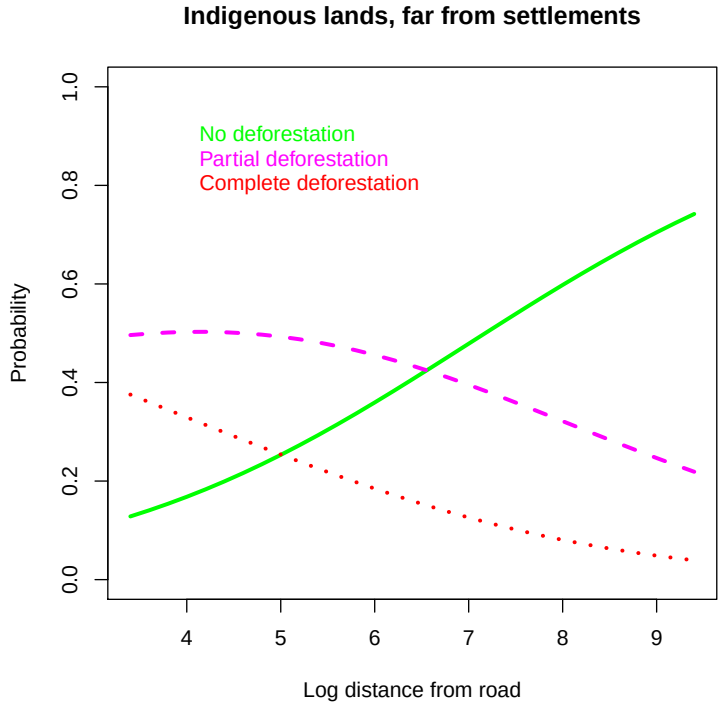
```
> plot(c(3.4, 9.4), c(0,1), type="n",
+      xlab="Log distance from road",
+      ylab="Probability", main="Non-titled lands, far from settlements")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CC"], lty=1, lwd=3, col="green")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CO"], lty=2, lwd=3, col="magenta")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CN"], lty=3, lwd=3, col="red")
> text(4, .9, "No deforestation", col="green", pos=4)
> text(4, .85, "Partial deforestation", col="magenta", pos=4)
> text(4, .8, "Complete deforestation", col="red", pos=4)
```



This clearly shows the increasing probability that forest will be conserved away from roads, for this tenure type and (far) distance from settlements. The probability of partial deforestation is fairly constant in this situation; the trade-off is with complete deforestation.

The picture for the indigenous lands at the same distance is somewhat different:

```
> to.graph <- subset(to.predict, (t=="Ci") & (ds.l == 8.7))
> g.fit <- predict(mlm.t.drs, to.graph, type="probs")
> plot(c(3.4, 9.4), c(0,1), type="n",
+      xlab="Log distance from road",
+      ylab="Probability", main="Indigenous lands, far from settlements")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CC"], lty=1, lwd=3, col="green")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CO"], lty=2, lwd=3, col="magenta")
> lines(seq(3.4, 9.4, by=.2), g.fit[,"CN"], lty=3, lwd=3, col="red")
> text(4, .9, "No deforestation", col="green", pos=4)
> text(4, .85, "Partial deforestation", col="magenta", pos=4)
> text(4, .8, "Complete deforestation", col="red", pos=4)
```



Here the chance of both kinds of deforestation decreases substantially with distance from roads. Complete deforestation is always more likely than partial on these lands.

11.5.2 Model fit to observations

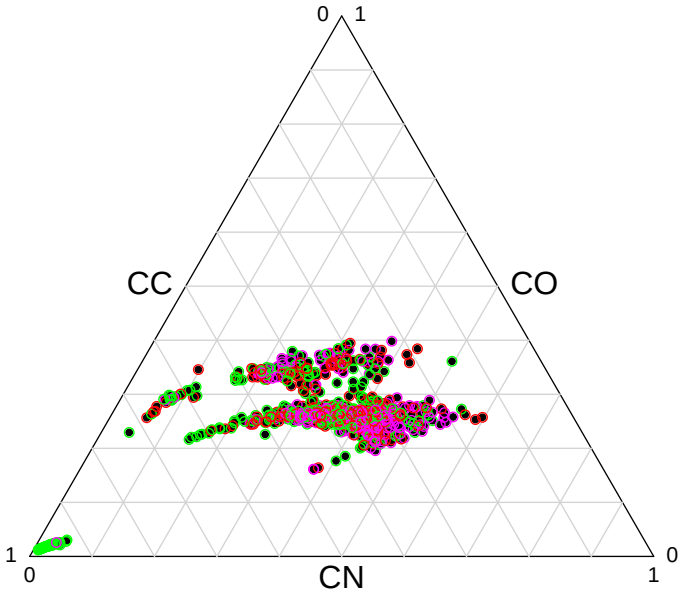
In the case with three classes, the relative probabilities of each observation can be viewed as a **triangle plot**; this is provided by the `triangle.plot` method of the `ade4` package for ecological analysis.

We first load the library:

```
> require(ade4)
```

We then plot the fits as a triangle, colour-coded by the actual class:

```
> fits <- data.frame(fitted(mlm.t.drs))
> str(fits)

'data.frame':        855 obs. of  3 variables:
 $ CC: num  0.546 0.532 0.557 0.431 0.507 ...
 $ CN: num  0.2026 0.2106 0.0974 0.2883 0.2316 ...
 $ CO: num  0.251 0.257 0.346 0.281 0.261 ...

> pts <- triangle.plot(fits, scale=F, show.position=F)
> points(pts, col=c("green","magenta","red")[as.numeric(dc$cov)])
> levels(dc$cov)[1:3]

[1] "CC" "CN" "CO"
```



A few points have very high probability of not being deforested; the green colour corresponding to the first class CC shows the correct classification. The rest of the diagram is disappointing. The middle of the triangle is where all classes are equally-likely. The ideal would be to have the first class (no deforestation, coloured green) in the lower-left corner (corresponding to class CC=1), the second class (full deforestation, red) in the lower-right (CN=1), and the third (partial deforestation, magenta) at the top (CO=1).

Cleaning up from this section:

```
> rm(mlm.t.drs, to.predict, p.fit, to.graph, g.fit, fits, pts)
```

12 Suggestions for further study

The easiest way to make sure you understand the techniques presented in this note is to *re-analyze* the dataset, but another sort of change rather than deforestation (from closed forest to any other type):

- **Rapid deforestation**: conversion *from* closed forest *to* non-forest (i.e. conversion to open forest is not considered a conversion);
- **Clearing**: conversion *from* either closed *or* open forest*to* non-forest;
- **Reforestation**: conversion *from* non-forest to either closed *or* open forest.

These require a change in the subset selection command in §4.1.

If you are more ambitious, you can use your own dataset. All that is required is that it have, or that you can create, a logical field (values `TRUE` and `FALSE`) stating the condition you want to model. For example, this can be presence/absence of a species in a set of plots.

> **Note:** If you have a binary (values `0` and `1`) field you can use it directly as a response variable in `glm`.

References

[1] Tom Fawcett. ROC graphs: Notes and practical considerations for researchers. Technical Report HP Labs Tech Report HPL-2003-4, HP Labs, 16-March-2004 2004. URL http://binf.gmu.edu/mmasso/ROC101.pdf. PDF:. 19

[2] J Fox. *Applied regression, linear models, and related methods*. Sage, Newbury Park, 1997. 3, 48, 49

[3] J Fox. *An R and S-PLUS Companion to Applied Regression*. Sage, Newbury Park, 2002. 3, 48

[4] John Fox. *Applied regression analysis and generalized linear models*. SAGE, 3rd ed edition, 2016. ISBN 978-1-4522-0566-3. 3

[5] JA Hanley. Receiver operating characteristic (ROC) methodology: the state of the art. *Critical Reviews in Diagnostic Imaging*, 29(3): 307–335, 1989. 19

[6] JA Hanley and BJ McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, 1982. 19

[7] JA Hanley and BJ McNeil. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. *Radiology*, 148:839–843, 1983. 19

[8] DW Hosmer and S Lemeshow. *Applied logistic regression*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., New York, 2000. 3

[9] R Ihaka and R Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996. 3

[10] F Leisch. Sweave, part I: Mixing R and LaTeX. *R News*, 2(3):28–31, December 2002. URL http://CRAN.R-project.org/doc/Rnews/. 6

[11] F Leisch. *Sweave User's Manual*. TU Wein, Vienna (A), 2.7.1 edition, 2006. URL http://www.statistik.lmu.de/~leisch/Sweave/Sweave-manual.pdf. 6

[12] JG Liao and D McGee. Adjusted coefficients of determination for logistic regression. *American Statistician*, 57(3):161–165, 2003. 19

[13] AV Loza U. *A spatial logistic model of tropical forest conservation. The case of Carrasco province - Cochabamba (1986-2002)*. Unpublished MSc thesis, University of Twente, Faculty of Geo-Information Science & Earth Observation (ITC), Enschede (NL), 2004. 3, 4, 5

[14] T Lumley. Programmers' niche: A simple class, in S3 and S4. *R News*, 4(1):33–36, June 2004. URL http://CRAN.R-project.org/doc/Rnews/. 19

[15] RG Pontius Jr. and LC Schneider. Land-cover change model validation by an ROC method for the Ipswich watershed, Massachusetts, USA. *Agriculture, Ecosystems & Environment*, 85(1-3):239–248, 2001. 19

[16] DG Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC (Revision 3.9)*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 2011. 6

[17] PA Sánchez, W Couto, and SW Buol. The fertility capability soil classification system: interpretation, applicability and modification. *Geoderma*, 27(4):283–309, 1982. 5

[18] PA Sánchez, CA Palm, and SW Buol. Fertility capability soil classification: a tool to help assess soil quality in the tropics. *Geoderma*, 114(3-4):157–185, 2003. 5

[19] HAMJ van Gils and AV Loza Armand Ugon. What drives conversion of tropical forest in Carrasco province, Bolivia? *Ambio*, 35(2):81–85, 2006. 3

[20] WN Venables and BD Ripley. *Modern applied statistics with S*. Springer-Verlag, New York, fourth edition, 2002. 3

A R code for visualising the logistic model

This section show the R code for six functions for visualising the performance of logistic regression models; these are used in the text. The code may be cut-and-paste from here; however, it is easier to load them from the `lcc.R` source code file with the R `source()` method:

R code:

```
source("lcc.R")
ls()
```

R console output:

```
"logit.plot"       "logit.plot.quad"     "logit.plot.ss"
"logit.roc"        "logit.roc.area"      "logit.roc.plot"
```

These can be applied to any fitted logistic model, for example:

R code:

```
glm.lspos <- glm(changed ~ lspos, family=binomial, data=dc)
logit.plot(glm.lspos)
logit.plot.quad(glm.lspos)
r <- logit.roc(glm.lspos)
logit.roc.area(r)
logit.roc.plot(r)
logit.plot.ss(r)
```

A.1 logit.plot

R code:

```
## visualise the success of a logistic model
## plot logistic curve, mean change, T/F, AIC, null deviance
## arguments
##   model   a fitted glm
##   title (optional)
logit.plot <- function(model, title="Success of logistic model") {
                  # sort the fitted values
    sf <- sort(fitted(model), index=T)
    plot(sf$x, ylim=c(0,1), type="l", col="blue", lwd=3,
         xlab="sorted sample number", ylab="probability")
    text(0, min(fitted(model))-.03,
         "fitted probability",col="blue",pos=4)
    title(title)
    abline(h=mean(fitted(model)), lty=2)
    text(0, mean(fitted(model))+.02, "mean probability", pos=4)
                  # name of the response field
    field.name <- attr(attr(terms(formula(model)), "factors"),
                       "dimnames")[[1]][1]
                  # extract the T/F vector
                  # depends on whether glm was called
                  #    with a data argument
    eval(parse(text=paste("tmp <- ",
               ifelse(class(model$data) == "data.frame", "model$data$", ""),
               field.name, sep="")))
    abline(v=length(tmp)/2,lty=2)
    text(length(tmp)/2,.03,"midpoint",pos=4)
                  # show T/F
    points(1:length(tmp), tmp[sf$ix],
           pch="|",cex=1,col=ifelse(tmp[sf$ix], "green4", "red"))
    text(0,.03,"FALSE Samples",col="red",pos=4)
    text(0,.97,"TRUE Samples",col="green4",pos=4)
                  # print model and fit
    text(length(tmp),0.30,paste(
         "Model:", formula(model)[2], formula(model)[1],
         formula(model)[3],sep=" "), pos=2,font=4)
    text(length(tmp),0.25,paste(
          "AIC:", round(model$aic,0), sep=" "),pos=2,font=4)
    text(length(tmp),0.20,paste(
          "Null deviance:", round(model$null.deviance,0), sep=" "),
          pos=2,font=4)
}
```

A.2 logit.plot.quad

R code:

```
## plot logistic curve, threshold, T/F +/-, sensitivity, specificity
## arguments
##   model   a fitted glm
##   threshold  cutoff for sensitivity/specificity, default 0.5
##   title   (optional)
logit.plot.quad <- function(model, threshold=0.5, title="Model success") {
    sf<-sort(fitted(model), index=T)
                  # leave extra space at bottom
    par(mar=c(6,4,4,2)+.1); par(xaxs="i", yaxs="r")
    plot(sf$x, ylim=c(0,1), type="l", col="blue", lwd=3, xlab="",
         ylab="probability of change")
    abline(h=c(0,1), lty=1)
                  # show threshold and crossover point
    abline(h=threshold,lty=2); text(0,threshold+.02,
                        paste("threshold =", threshold), pos=4)
    crossover <- sum(fitted(model) < threshold)
    abline(v=crossover,lty=2)
    text(crossover,.05,"crossover",pos=4)
    text(crossover, threshold-.03,
         "fitted probability of change",col="blue",pos=4)
                  # name of the response field
    field.name <- attr(attr(terms(formula(model)), "factors"),
                       "dimnames")[[1]][1]
                  # extract the T/F from it
    eval(parse(text=paste("tmp <- ",
                ifelse(class(model$data) == "data.frame", "model$data$", ""),
                field.name, sep="")))
                  # show T/F as vertical bars at the index
                  # colours differ with T/F predictions
    points(1:length(tmp),tmp[sf$ix],
           pch="|",cex=1,
           col=ifelse((tmp[sf$ix] == (sf$x>threshold)),"green4","red"))
                  # compute proportions
    tn <- sum((!tmp[sf$ix]) & (sf$x < threshold))
    fn <- sum((!tmp[sf$ix]) & (sf$x >= threshold))
    tp <- sum(tmp[sf$ix] & (sf$x >= threshold))
    fp <- sum(tmp[sf$ix] & (sf$x < threshold))
    right  <- length(sf$x)*.65
    text(0,.1,paste("True negatives:",tn), col="green4",pos=4)
    text(right,.1,paste("False negatives:", fn), col="red",pos=4)
    text(right,.9,paste("True positives:", tp), col="green4",pos=4)
    text(0,.9,paste("False positives:", fp), col="red",pos=4)
    title(main=title)
    title(sub=paste("Sensitivity:", round(tp/(tp+fp),4),
                    "; Specificity:", round(tn/(tn+fn),4)), line=4)
}
```

A.3 logit.roc

R code:

```
## compute an empirical ROC curve from a fitted logistic model
## and a data frame with the modelled logical field
##
## results are used as an argument to logit.plot.ss, logit.roc.plot
##
## arguments
##   model   a fitted glm
##   steps   how many thresholds; default 20
## returns
##   logit.roc     data frame with three fields:
##      pts          vector of points along the curve
##      sens, spec   sensitivity, specificity
logit.roc <- function(model, steps=20) {
                  # get the response field
                  # from the model object
    field.name <- attr(attr(terms(formula(model)), "factors"),
                       "dimnames")[[1]][1]
                  # and extract the T/F from it
    eval(parse(text=paste("tmp <- ",
                ifelse(class(model$data) == "data.frame", "model$data$", ""),
                field.name, sep="")))
    r <- data.frame(pts = seq(0, 1-(1/steps), by=1/steps),
                    sens = 0, spec=0);
    for (i in 0:steps) {
      thresh <- i/steps;
      r$sens[i] <- sum((fitted(model) >= thresh) & tmp)/sum(tmp);
      r$spec[i] <- sum((fitted(model) < thresh) & !tmp)/sum(!tmp)
      }
    return(r)
}
```

A.4 logit.roc.area

R code:

```
## compute area under a ROC curve computed by logit.roc()
## typical usage: logit.roc.area(roc(model, dataset))
## argument
##   r  an ROC curve returned by logit.roc()
logit.roc.area <- function(r) {
    area <- 0;
    for (i in 1:(length(r$pts)-1))
        area <- area + ((1 - r$sens[i+1]) - (1 - r$sens[i])) *
                       ((r$spec[i+1] + r$spec[i])/2);
    return(area)
    }
```

A.5 logit.roc.plot

R code:

```
## plot an ROC curve computed by logit.roc()
## typical usage: logit.roc.plot(roc(model, dataset))
## argument
##   r  an ROC curve returned by logit.roc()
logit.roc.plot <- function(r, title="ROC curve") {
    old.par <- par(no.readonly = TRUE); on.exit(par(old.par))
    par(xaxs="i", yaxs="i")
    plot(1 - r$spec, r$sens, xlim=c(0, 1), ylim=c(0,1), type="l",
        xlab="(1 - specificity): false positive rate",
        ylab="sensitivity: true positive rate",
        col="blue", lwd=2);
    points(1 - r$spec, r$sens, pch=20, cex=2, col="blue");
    abline(0, 1, lty=2);
    segments(1-r$spec, 1-r$spec, 1-r$spec, r$sens, lty=2)
    text(0, 0.9, paste("Area under ROC:",round(logit.roc.area(r),4)), pos=4)
    title(main = title)
    }
```

A.6 logit.plot.ss

R code:

```
## plot sensitivity, specificity vs. threshold for a logistic model
## typical usage: logit.plot.ss(logit.roc(model, dataset))
## argument
##    r  ROC curve computed by logit.roc()
logit.plot.ss <- function(r) {
    plot(r$pts, r$spec, type="n",
         xlab="Threshold",  ylab="value", ylim=c(0,1));
    title(main = "Sensitivity and specificity vs. threshold");
    abline(h=seq(0, 1, by=0.1, lty="dashed", lwd=0.5));
    lines(r$pts, r$spec, col="blue", lwd=1.5);
    lines(r$pts, r$sens, col="green4", lwd=1.5);
    text(0.05, 0.05, pos=4, col="blue", "Specificity");
    text(0.05, 0.95, pos=4, col="green4", "Sensitivity");
    }
```

A.7 Programmer's notes

This paragraph is for those who are curious about some of the less-obvious aspects of the R code.

The main trick in this code is to look inside the fitted model object using extractor methods such as `fitted` (to get the fitted probabilities of change), and `formula` (to get the model formula). In some cases we have

to access the model object directly, for example the `$data` field (to get the data used to fit the model).

The `terms` method is used to extract the terms of the formula; this then has named attribute `"factors"`, which we extract with the `attr` method; this attribute itself has attribute `"dimnames"` which we also extract with `attr`; finally we get the first element `[1]` of the first list `[[1]]`:

R code:

```
formula(model)
terms(formula(model))
attr(terms(formula(model)), "factors")
attr(attr(terms(formula(model)), "factors"),"dimnames")
attr(attr(terms(formula(model)), "factors"),"dimnames")[[1]]
attr(attr(terms(formula(model)), "factors"),"dimnames")[[1]][1]
```

R console output:

```
changed ~ lspos

attr(,"variables")
list(changed, lspos)
attr(,"factors")
        lspos
changed     0
lspos       1
attr(,"term.labels")
[1] "lspos"
attr(,"order")
[1] 1
attr(,"intercept")
[1] 1
attr(,"response")
[1] 1
attr(,".Environment")
<environment: R_GlobalEnv>

        lspos
changed     0
lspos       1

[[1]]
[1] "changed" "lspos"

[[2]]
[1] "lspos"

[1] "changed" "lspos"

[1] "changed"
```

An especially tricky aspect is getting the data itself, because there are two ways that `glm` could have been called: with or without an explicit `data=` argument. We can tell which way it was called by applying the `class` method to the `$data` field of the model. The results will either be `data.frame` (if called with an explicit `data=` argument) or `environment` (if data was taken from the environment, e.g. if the data frame were `attach`ed before the call).

To copy the data into a local vector, we build up a command as a text string (using the `paste` method), parse it into an R language object (using the `parse` method), and finally evaluate it in the environment (using the `eval` method). This illustrates the tremendous flexibility of the S language:

R code:

```
eval(parse(text=paste("tmp <- ",
       ifelse(class(model$data) == "data.frame", "model$data$", ""),
       field.name, sep="")))
```

For example, suppose the model was built in the local environment (no `data=` argument to `glm`), and the field is `changed`; this will be evaluated as if the following code were in the program:

R code:

```
tmp <- changed
```
