R-exercise for Friday 12 Dec 2014

December 2014
http://BendixCarstensen.com/Epi
Version 5

Compiled Thursday 11th December, 2014, 19:25
from: `C:/Bendix/undervis/SPE/Intro/R-intro-SDC.tex`

Bendix Carstensen Steno Diabetes Center, Gentofte, Denmark
& Department of Biostatistics, University of Copenhagen
bxc@steno.dk
www.pubhealth.ku.dk/~bxc

Edition 2014 by Bendix Carstensen

Contents

Chapter 1

Epidemiological calculations

1.1 Diagnostic criteria in the NDR

This exercise is based on the Danish National Diabetes Register (NDR) — or to be precise a 10% bogus sample. The term ”bogus” refers to the fact that the database we shall use is a 10% sample of the NDR, were all dates have been changes randomly by a quantity in the range from -7 to 7 days, so no person is identifiable, but we still have a database that behaves as the NDR in terms of incidence, mortality etc.

1. First load the Epi-package, and then get the 10% sample of the NDR from the web — the variable names are as in the original NDR, but all dates have been altered. Note that you *must* include the web-address in a `url()`-command, moreover the web-address is case-sensitive:

```
> library( Epi )
> clear()
> load( file=url("http://BendixCarstensen.com/DMreg/data/NDR2011-bogus.Rda") )
> lls()
```

2. Now take a look at the data frame, and convince yourself that the missingness-pattern is ok:

```
> str( ndr )
> summary( ndr )
```

3. Note that all the dates are factors, which is a bit impractical, so we convert them to fractional years by `cal.yr`. This is most easily done in a loop over all the date variables. `R` is so friendly that you can use the variable names in commands too; the date varaibels are those that start with `D_`; and `grep` is the function that seraces for a particular string in a character vector; try:

```
> names( ndr )
> grep( "D_", names(ndr) )
```

   This is a way to get the *position number* of the date variables, so conversion is dead-easy now:

```
> wh <- grep( "D_", names(ndr) )
> for( i in wh ) ndr[,i] <- cal.yr( ndr[,i] )
> head( ndr )
```

We can use `cal.yr` without firther arguments because the dates are in standard ISO-format (`YYYY-MM-DD`), othrwise we would need the `format` argument.

4. It is a bit annoying with upper-case names and with the underscores, so we can just change the variable names by removing the first two letters and putting the rest to lower case (the 10 is just to cut the names short for convenience). And then we save the groomed register

```
> names( ndr ) <- tolower( substr(names(ndr),3,10) )
> names( ndr )
> save( ndr, file="ndr.Rda" )
```

1.1.1 Blood glucose criteria in NDR

It has been debated whether the blood glucose criteria shoudl be used at all, and we therefore create a new version of the register by computing a new date of inclusion and a new inclusion criterion, *as if* the two glucose citeria were omitted:

5. Now create two new variables, `new.aars` and `new.dto`, based only on LPR, foot therapy and medication:

```
> load( file="ndr.Rda" )
> ndr$new.dto <- with( ndr, pmin( fodt, lpr, oad, ins, na.rm=TRUE ) )
> # Assigning levels is a bit more tricky
> ndr$new.aars                         <- NA
> ndr$new.aars[ndr$new.dto==ndr$fodt]<- "fodt"
> ndr$new.aars[ndr$new.dto==ndr$oad] <- "oad"
> ndr$new.aars[ndr$new.dto==ndr$ins] <- "ins"
> ndr$new.aars[ndr$new.dto==ndr$lpr] <- "lpr"
> summary( ndr )
```

6. Now make a table of changes:

```
> addmargins( with( ndr, table( inklaars, new.aars, useNA="ifany" ) ) )
```

Why have some non-clucose inclusions changed?

7. The question of real interest is another: How does the exclusion from the register depend on age and date. This is just a logistic regression:

```
> ndr <- transform( ndr, A = inkldto-foddto )
> mF <- glm( is.na(new.aars) ~ A + I(A^2) + I(A^3),
+            family = binomial,
+            data = subset( ndr, sex=="K" ) )
> summary( mF )
```

We dont get much wiser from that.

8. Instead, predict the probability as a function of `A`, that we put in a *prediction data frame* — it muts contain variables with the same names as the *explanatory* variables in the model:

```
> nd <- data.frame( A=5:90 )
> pr.F <- predict( mF, newdata=nd, type="response" )
> plot( nd$A, pr.F, type="l" )
```

9. But we would like to make it a bit more flexible, so we put in a natural spline of age instaed:

```
> library( splines )
> a.kn <- c(10,20,25,30,35,4:9*10)
> mF <- glm( is.na(new.aars) ~ Ns( A, knots=a.kn ),
+            family = binomial,
+            data = subset( ndr, sex=="K" ) )
> summary( mF )
> pr.F <- predict( mF, newdata=nd, type="response" )
> plot( nd$A, pr.F, type="l" )
```

10. But we would alos like to see if it depends on time, so we include the data of diagnosis, `inkldto`, in the model too:

```
> p.kn <- seq(1996,2010,,4)
> mF <- glm( is.na(new.aars) ~ Ns( A, knots=a.kn ) +
+                              Ns( inkldto, knots=p.kn ),
+            family = binomial,
+            data = subset( ndr, sex=="K" ) )
> summary( mF )
> nd <- data.frame( A=5:90, inkldto=2000 )
> pr.F <- predict( mF, newdata=nd, type="response" )
> plot( nd$A, pr.F, type="l" )
```

11. But this only gives the prediction for the year 2000, we would like to see it for all years. In order to use `matplot`, we would like to have the predictions of the age-specific probabilities next to each oter in a matrix. We use `cbind` for that:

```
> pr.F <- NULL
> for( p in 1996:2011 )
+    {
+    nd <- data.frame( A=5:90, inkldto=p )
+    pr.F <- cbind( pr.F, predict( mF, newdata=nd, type="response" ) )
+    }
> matplot( nd$A, pr.F,
+          type="l", lty=1, lwd=1:2, col=heat.colors(25)[1:16] )
> text( c(80,80), c(55,60)/100, c(1996,2011),
+       col=heat.colors(25)[c(1,16)], font=2 )
```

12. But this is a model that assumes that the age effects is the same in all years, so we explore an inateraction model:

```
> iF <- glm( is.na(new.aars) ~ Ns( A, knots=a.kn )*Ns( inkldto, knots=p.kn ),
+            family = binomial,
+            data = subset( ndr, sex=="K" ) )
> summary( iF )
> anova( mF, iF, test="Chisq" )
```

The nice thing here is that the prediction machinery is eactly the same:

```
> pr.F <- NULL
> for( p in 1996:2011 )
+    {
+    nd <- data.frame( A=5:90, inkldto=p )
+    pr.F <- cbind( pr.F, predict( iF, newdata=nd, type="response" ) )
+    }
> matplot( nd$A, pr.F,
+          type="l", lty=1, lwd=1:2, col=heat.colors(25)[1:16] )
> text( c(80,80), c(55,60)/100, c(1996,2011),
+       col=heat.colors(25)[c(1,16)], font=2 )
```

The graph now really indicates that there is an interaction!