Nonlinear models in R

Table of Contents

### Load libraies for plots and data analysis  
library(tidyr)  
library(ggplot2)  
library(sjPlot)  
library(sjmisc)

##   
## Attaching package: 'sjmisc'

## The following object is masked from 'package:tidyr':  
##   
## replace\_na

library(sjlabelled)  
library(MASS)  
library(tidyverse)

## -- Attaching packages --------------------------------------------------------------------------------- tidyverse 1.3.0 --

## v tibble 3.0.0 v dplyr 0.8.5  
## v readr 1.3.1 v stringr 1.4.0  
## v purrr 0.3.4 v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------ tidyverse\_conflicts() --  
## x tibble::add\_case() masks sjmisc::add\_case()  
## x forcats::as\_factor() masks sjlabelled::as\_factor()  
## x dplyr::as\_label() masks sjlabelled::as\_label()  
## x dplyr::filter() masks stats::filter()  
## x purrr::is\_empty() masks sjmisc::is\_empty()  
## x dplyr::lag() masks stats::lag()  
## x sjmisc::replace\_na() masks tidyr::replace\_na()  
## x dplyr::select() masks MASS::select()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(leaps)  
library(glmnet)

## Loading required package: Matrix

##   
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack

## Loaded glmnet 4.0

# load response variables  
  
### Read data for specific gene expression  
dat1 <- read.csv(file="ENSG115944.csv" )  
  
# STR data for locus of interest  
dat2 <- read.csv(file="ENS115944STRup2.csv",header=FALSE )  
  
  
dat3 <- merge(dat1,dat2,by.x="Gene",by.y="V1")  
  
  
#We only want the one locus. Drop all other levels from factor variables.  
dat4 <- droplevels(dat3)  
  
  
varit <- 3:67  
numcols <- ncol(dat4)-2  
resultsout <- NULL  
# bring response and STR information together.  
for (i in varit) {  
#### Want the STR value to numeric not factor  
dat4[,i] <- as.numeric(as.character(dat4[,i]))  
dat4[,i+numcols] <- dat4[,i]\*\*2  
  
}  
  
  
dat5 <- aggregate(dat4[,3:ncol(dat4)],by=list(dat4$Gene,dat4$Response),FUN=sum,na.rm=TRUE)  
  
names(dat5)[1:2] <- c("Gene","y")  
  
dat6 <- dat5  
  
varit <- 3:67  
numcols <- ncol(dat5)-2  
# bring response and STR information together.  
for (i in varit) {  
#### Want the STR value to numeric not factor  
dat6[,i+numcols] <- dat6[,i]\*dat6[,i+65]  
  
}  
  
  
agglm <- lm(y ~ .,dat=dat5[,2:ncol(dat5)])  
  
agglm2 <- lm(y ~ V30+V31+V47+V96+V114,dat=dat5[,2:ncol(dat5)])  
  
agglmf <- lm(y ~ V30+V31\*V96+V47\*V114,dat=dat5[,2:ncol(dat5)])  
  
  
  
  
  
SSTotal <- var( dat5$y ) \* (nrow(dat5)-1)  
 SSE <- sum( agglm2$resid^2 )  
SSreg <- SSTotal - SSE  
BICreg <- BIC(agglm2)  
  
 SSEf <- sum( agglmf$resid^2 )  
SSregf <- SSTotal - SSEf  
BICregf <- BIC(agglmf)  
  
#resultsout <- rbind(resultsout,cbind(i,BICreg,BICregf,SSTotal,SSreg,SSE,SSregf,SSEf))

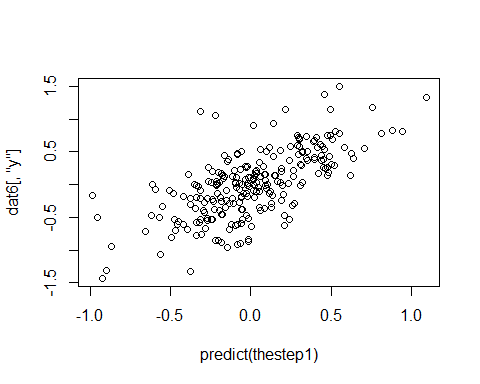
full.model <- lm(y ~ .,dat=dat6[,2:ncol(dat6)])  
# Stepwise regression model  
step.model <- stepAIC(full.model, direction = "both",   
 trace = FALSE)  
  
BIC(step.model)

## [1] 382.1433

thestep <- lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V125 + V135 + V153 + V154 + V159 + V162 + V169 +   
 V174 + V179 + V190 + V142, data = dat6[, 2:ncol(dat6)])  
  
summary(thestep)

##   
## Call:  
## lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
## V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
## V117 + V118 + V125 + V135 + V153 + V154 + V159 + V162 + V169 +   
## V174 + V179 + V190 + V142, data = dat6[, 2:ncol(dat6)])  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.90914 -0.20946 -0.00221 0.22220 1.43389   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.378808 0.064004 5.919 1.28e-08 \*\*\*  
## V4 -0.525092 0.171874 -3.055 0.002536 \*\*   
## V18 -0.128741 0.058972 -2.183 0.030117 \*   
## V20 0.805926 0.248962 3.237 0.001399 \*\*   
## V30 -0.108599 0.026758 -4.059 6.92e-05 \*\*\*  
## V31 -0.212672 0.117736 -1.806 0.072270 .   
## V38 0.092832 0.050327 1.845 0.066482 .   
## V48 -0.201828 0.088610 -2.278 0.023732 \*   
## V51 0.228535 0.155551 1.469 0.143247   
## V59 0.154097 0.091413 1.686 0.093307 .   
## V73 -0.205572 0.087537 -2.348 0.019766 \*   
## V83 0.049659 0.020898 2.376 0.018371 \*   
## V84 -0.049348 0.020825 -2.370 0.018692 \*   
## V88 0.177177 0.077951 2.273 0.024023 \*   
## V95 0.032839 0.014321 2.293 0.022817 \*   
## V96 0.009865 0.002503 3.940 0.000110 \*\*\*  
## V97 0.164085 0.077348 2.121 0.035039 \*   
## V101 0.092816 0.053853 1.723 0.086243 .   
## V109 -0.054985 0.020805 -2.643 0.008828 \*\*   
## V117 0.281840 0.159830 1.763 0.079265 .   
## V118 -0.009544 0.006687 -1.427 0.154956   
## V125 0.093701 0.063997 1.464 0.144621   
## V135 0.232253 0.089491 2.595 0.010106 \*   
## V153 0.080775 0.031630 2.554 0.011353 \*   
## V154 0.004307 0.002882 1.494 0.136561   
## V159 -0.015398 0.004269 -3.607 0.000386 \*\*\*  
## V162 -0.020138 0.010823 -1.861 0.064162 .   
## V169 -0.004385 0.002405 -1.823 0.069670 .   
## V174 -0.012147 0.005975 -2.033 0.043297 \*   
## V179 0.144379 0.041647 3.467 0.000637 \*\*\*  
## V190 0.016194 0.010847 1.493 0.136930   
## V142 -0.014435 0.004176 -3.457 0.000660 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3899 on 214 degrees of freedom  
## Multiple R-squared: 0.4856, Adjusted R-squared: 0.411   
## F-statistic: 6.516 on 31 and 214 DF, p-value: < 2.2e-16

SSEstep <- sum( thestep$resid^2 )  
SSregstep <- SSTotal - SSEf  
BICregstep <- BIC(thestep)  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V125 + V135 + V153 + V154 + V159 + V162 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V125 + V135 + V153 + V159 + V162 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V135 + V153 + V159 + V162 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V31 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V135 + V153 + V159 + V162 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V135 + V153 + V159 + V162 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V97 + V101 + V109 +   
 V117 + V118 + V135 + V153 + V159 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V38 + V48 + V51 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V101 + V109 +   
 V117 + V118 + V135 + V153 + V159 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V38 + V48 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V101 + V109 +   
 V117 + V118 + V135 + V153 + V159 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
thestep1 <- lm(formula = y ~ V4 + V18 + V20 + V30 + V38 + V48 +   
 V59 + V73 + V83 + V84 + V88 + V95 + V96 + V101 + V109 +   
 V117 + V135 + V153 + V159 + V169 +   
 V174 + V179 + V142, data = dat6[, 2:ncol(dat6)])  
  
plot(predict(thestep1),dat6[,"y"])



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.