9: Discrimination and Classification

John H Maindonald

April 3, 2018

Ideas and issues illustrated by the graphs in this vignette

The methods illustrated here have the character of regression models where the outcome is categorical, one of g classes. For example, the `fgl` dataset has measurements of each on nine physical properties, for 214 samples of glass that are classified into six different glass types.

Note: The versions of Figures 9.9 and 9.10 that are shown in Section 2 are for a substantially reduced number of points, relative to the text *Statistically Informed Data Mining*.

1 Code for the Figures

```r
fig9.1 <- function(plotit=TRUE){
    fgl.lda <- lda(type ~ ., data=fgl)
    scores <- predict(fgl.lda)$x
    library(lattice)
    gph <- xyplot(scores[,2] ~ scores[,1], groups=fgl$type,
                  xlab="Discriminant 1", ylab="Discriminant 2",
                  aspect=1, scales=list(tck=0.4),
                  auto.key=list(columns=3),
                  par.settings=simpleTheme(alpha=0.6, pch=1:6))
    gph
}
```

```r
fig9.2 <- function(){
    gph <- xyplot(length ~ breadth, groups=species, data=cuckoos,
                  type=c("p"), auto.key=list(space="right"), aspect=1,
                  scales=list(tck=0.5), par.settings=simpleTheme(pch=16))
    LDmat <- cuckoos.lda$scaling
    ld1 <- LDmat[,1]
    ld2 <- LDmat[,2]
    gm <- sapply(cuckoos[, c("length", "breadth")], mean)
    av1 <- gm[1] + ld1[2]/ld1[1]*gm[2]
    av2 <- gm[1] + ld2[2]/ld2[1]*gm[2]
    assign('av1', av1, pos=1)
    assign('av2', av2, pos=1)
    assign('ld1', ld1, pos=1)
    assign('ld2', ld2, pos=1)
    addlayer <- latticeExtra::layer(panel.abline(av1, -ld1[2]/ld1[1], lty=1),
                                    panel.abline(av2, -ld2[2]/ld2[1], lty=2))
    gph + addlayer
}
```

```
fig9.3 <- function(){
    ## This will show decision boundaries
    gph <- xyplot(length ~ breadth, groups=species, data=cuckoos,
                  type=c("p"), auto.key=list(space="right"), aspect=1,
                  scales=list(tck=0.5), par.settings=simpleTheme(pch=16))
    x <- pretty(cuckoos$breadth, 20)
    y <- pretty(cuckoos$length, 20)
    Xcon <- expand.grid(breadth=x, length=y)
    cucklda.pr <- predict(cuckoos.lda, Xcon)$posterior
    cuckqda.pr <- predict(cuckoos.qda, Xcon)$posterior
    m <- match("wren", colnames(cucklda.pr))
    ldadiff <- apply(cucklda.pr, 1, function(x)x[m]-max(x[-m]))
    qdadiff <- apply(cuckqda.pr, 1, function(x)x[m]-max(x[-m]))
    addlayer1 <- latticeExtra::as.layer(contourplot(ldadiff ~ breadth*length,
                                         at=c(-1,0,1), labels=c("", "lda",""),
                                         label.style="flat",
                                         data=Xcon), axes=FALSE)
    addlayer2 <- latticeExtra::as.layer(contourplot(qdadiff ~ breadth*length,
                                        at=c(-1,0,1), labels=c("", "qda",""),
                                        label.style="flat",
                                        data=Xcon), axes=FALSE)
    gph + addlayer1 + addlayer2
}
```

```
fig9.4 <- function(seed=47){
  opar <- par(xpd=TRUE)
  ## xpd=TRUE allows labels to extend outside of figure region
    b.rpart <- rpart(rfac ~ cig+poll, data=bronchit)
    plot(b.rpart, uniform=TRUE)
    text(b.rpart)
  par(opar)
}
```

```
fig9.5 <- function(){
    b001.rpart <- rpart(rfac ~ cig+poll, cp=0.001, minsplit=15,
                        data=bronchit)
    plotcp(b001.rpart)
}
```

```
fig9.6 <-
function () {
plot.root <- function(text='Reduction in "error" (Gini) = 20.55',
                      cutoff="cig<4.375", left="138/11", rt="28/35",
                      xlef=0.15, xrt=0.85,
                      treetop=0.85, treebot=0.1){
    par(mar=rep(0,4))
    plot(0:1, 0:1, axes=F, xlab="",ylab="", type="n")
    lines(c(xlef,xlef, xrt,xrt), c(.1,treetop,treetop,.1))
    lines(c(.5,.5),c(-0.01,0.01)+treetop)
    chh <- strheight("O")
    text(.5, treetop+chh, cutoff)
    text(c(xlef,xrt), rep(.1-chh,2), c(left,rt))
    legend(x=0.5, y=1, xjust=0.5, yjust=1, xpd=TRUE,
       legend=text, bg='gray')
}
    par(fig=c(0,0.5,0,1))
    plot.root(text='Decrease in "error" = 20.55',
              cutoff="cig<4.375", left="138/11", rt="28/35",
              treetop=0.6, treebot=0.1)
    par(fig=c(0.5,1,0,1), new=TRUE)
    plot.root(text='Decrease in "error" = 2.90',
              cutoff="poll<58.55", left="98/16", rt="68/30",
              treetop=0.6, treebot=0.1)
}
```

```
fig9.7 <-
function ()
{
    set.seed(31)   # Reproduce the trees shown
    opar <- par(mfrow=c(3,3), xpd=TRUE)
    num <- 1:nrow(bronchit)
    for(i in 1:9){
        useobs <- sample(num, replace=TRUE)
        dset <- bronchit[useobs, ]
        b.rpart <- rpart(rfac ~ cig+poll, data=dset,
                         control=rpart.control(maxdepth=2))
        plot(b.rpart, uniform=TRUE)
        text(b.rpart, xpd=TRUE, cex=1.2)
    }
    par(mfrow=c(1,1))
    par(opar)
}
```

```
fig9.8 <- function(){
    bronchit <-
      within(bronchit,
             rfac <- factor(r, labels=c("abs","pres")))
    parset <- simpleTheme(pch=1:2)
    bronchit.rf <- randomForest(rfac ~ cig+poll, proximity=TRUE,
                                data=bronchit)
    points <- cmdscale(1-bronchit.rf$proximity)
    gph <- xyplot(points[,2] ~ points[,1], groups=bronchit$rfac,
                  xlab="Axis 1", ylab="Axis 2",
                  par.settings=parset, aspect=1,
                  auto.key=list(columns=2))
    gph
    }
```

```
fig9.9 <- function(nn0 = c(3596, 900, 600, 400, 270, 180, 120, 90, 60, 40),
                   repeats=5, seed=NULL, testlong=NULL, plotit=TRUE){
if(!is.null(seed))set.seed(seed)
if(is.null(testlong)){
## ---- vary-noninsure ----
testInsure <- matrix(0, ncol=repeats, nrow=length(nn0))
for(i in 1:repeats){
  j<-0
  for(n0 in nn0){
    j<-j+1
    testInsure[j, i] <- bestsize(n0)
  }
}
attr(testInsure, "dimnames") <- list(n0=nn0, Repeat=1:repeats)
## Long version of data frame
testlong <- data.frame(test=as.vector(testInsure),
                       n0=rep(nn0, repeats),
                       gp=rep(1:repeats, rep(length(nn0),repeats)))
```

```r
}
if(!plotit)return(invisible(testlong))
## Plot data
nn0 <- unique(testlong[,"n0"])
ndistinct <- length(nn0)
if(ndistinct >= 4){
test.gam <- gam(test ~ s(log(n0), k=min(ndistinct,3)), data=testlong)
plot(test.gam, se=T, residuals=T, pch=1, xaxt="n",
     xlab="n0, in 'sampsize=c(n0, 226)'",
     ylab="# insurances, best 400 test",
     shift=mean(fitted(test.gam)))
axis(1, at=log(nn0), labels=paste(nn0), las=3)
} else
{
    plot(test ~ log(n0), data=testlong, pch=1, xaxt="n",
    xlab="n0, in 'sampsize=c(n0, 226)'",
    ylab="# insurances, best 400 test")
    mtext(side=3, line=0.5, expression(
          "Curve is fitted only if there are " >= " 4 distinct values of n0"))
}
invisible(testlong)
}
```

```r
fig9.10 <- function(nn0 = c(3596, 900, 600, 400, 270, 180, 120, 90, 60, 40),
                    repeats=5, seed=NULL, heldlong=NULL, plotit=TRUE){
if(!is.null(seed))set.seed(seed)
if(is.null(heldlong)){
## ---- vary-held ----
heldInsure <- matrix(0, ncol=repeats, nrow=length(nn0))
for(i in 1:repeats){
  j<-0
  for(n0 in nn0){
    j<-j+1
    heldInsure[j, i] <- bestsize(n0, nselect=800,
                                 x=ticShown[, -c(1,86)],
                                 y=ticShown[, 86],
                                 xtest=ticHeld[, -c(1,86)],
                                 ytest=ticHeld[, 86])
  }
}
attr(heldInsure, "dimnames") <- list(n0=nn0, Repeat=1:repeats)
## ---- plot-held ----
heldlong <- data.frame(insure=as.vector(heldInsure),
                       n0=rep(nn0, repeats),
                        gp=rep(1:repeats, rep(length(nn0),repeats)))
}
if(!plotit)return(invisible(heldlong))
## Plot data
nn0 <- unique(heldlong[,"n0"])
ndistinct <- length(nn0)
if(ndistinct>=4){
held.gam <- gam(insure ~ s(log(n0), k=min(ndistinct-1,3)), data=heldlong)
plot(held.gam, se=T, residuals=T, pch=1, xaxt="n",
     xlab="n0, in 'sampsize=c(n0, 348)'",
     ylab="# insurances, best 800 prospects",
     shift=mean(fitted(held.gam)))
} else
{
    plot(insure ~ log(n0), data=heldlong, pch=1, xaxt="n",
    xlab="n0, in 'sampsize=c(n0, 226)'",
    ylab="# insurances, best 400 test")
    mtext(side=3, line=0.5, expression(
          "Curve is fitted only if there are " >= " 4 distinct values of n0"))
}
axis(1, at=log(nn0), labels=paste(nn0), las=3)
invisible(heldlong)
}
```

```
compareTargets <-
function(rfobj, prior1, prior2){
    nam1 <- deparse(substitute(prior1))
    nam2 <- deparse(substitute(prior2))
    print(c(nam1,nam2))
    err <- rfobj$confusion[,3]
    err1 <- sum(err*prior1)/sum(prior1)
    err2 <- sum(err*prior2)/sum(prior2)
    errvec <- c(err, err1,err2)
    names(errvec) <- c("error-good", "error-bad", nam1, nam2)
    errvec
  }
```

```
bestsize <- function(n0=226, mtry=9, nselect=400,
                     x=tictrain[, -c(1,86)], y=tictrain[, 86],
                     xtest=tictest[, -c(1,86)], ytest=tictest[, 86])
{
  tic.rf <- randomForest(x=x, y=y,
                         xtest=xtest, ytest=ytest,
```

```r
                         sampsize=c(n0,226),
                         mtry=mtry, data=data)
  nrbest <- order(tic.rf$test$votes[,2],
                  decreasing=TRUE)[1:nselect]
  buy <- sum(ytest[nrbest]=="insurance")
  buy
}
```

```r
ldaErr <- function(train.lda=spam01.lda, train=spam01, test=spam2,
                   traingp=spam01[,'type'], testgp=spam2[,'type']){
    trainCV.lda <- update(train.lda, CV=TRUE)
    prior01 <- train.lda$prior
    ldaRates <- c(loo=1-confusion(traingp,
                            trainCV.lda$class,
                            printit=NULL)$overall,
              trainerr=1-confusion(traingp,
                                 predict(train.lda)$class,
                                 printit=NULL)$overall,
              testerr=1-confusion(testgp,
                                predict(train.lda,
                                        newdata=test)$class,
                                prior=prior01, printit=NULL)$overall)
     ldaRates
}
```

```r
rpartErr <- function(train.rp=spam01.rp, train=spam01, test=spam2,
                     outcome='type'){
    cptab <- train.rp$cptable
    nbest <- which.min(cptab[,"xerror"])
    rnprop <- prop.table(table(train.rp$y))
    xcv <- cptab[nbest,"xerror"] * min(rnprop)
    trainerr <- cptab[nbest,"rel error"] * min(rnprop)
    class2 <- predict(train.rp, newdata=test, type="class")
    testerr <- 1-confusion(test[, outcome], class2, printit=FALSE,
                         prior=rnprop)$overall
    c(cverror=xcv, trainerror=trainerr, testerror=testerr)
}
```

```r
rfErr <- function(train.rf=spam01.rf, train=spam01, test=spam2,
                  outcome='type'){
    trainClass <- predict(train.rf, newdata=spam01, type="class")
    testClass <- predict(train.rf, newdata=test, type="class")
    rnprop <- prop.table(table(train[, outcome]))
    rfRates <- c(OOBerr=train.rf$err.rate[train.rf$ntree, "OOB"],
             trainerr=1-confusion(train$type, trainClass,
                                    printit=FALSE)$overall,
             testerr=1-confusion(spam2$type, testClass, printit=FALSE,
                                   prior=rnprop)$overall)
    rfRates
}
```

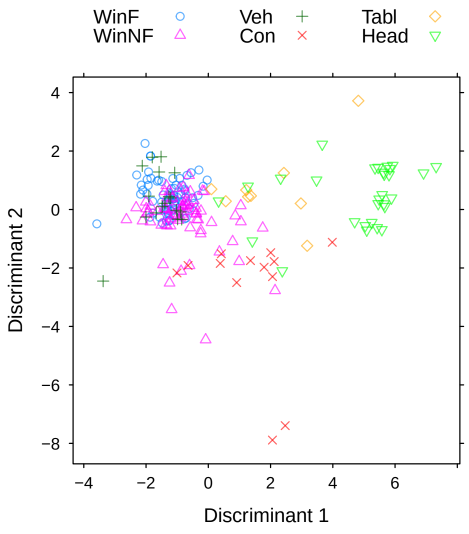
```
fig9.11 <- function(){
if(!exists("Vowel"))
  return("Dataset 'Vowel' (from mlbench) is not available")
form <- paste("~", paste(paste("V", 2:10, sep= ""),
                     collapse="+"))
gph <- bwplot(formula(paste("Class", form)),
              scales=list(x="free"),
              data=Vowel, outer=TRUE, layout=c(3,3))
gph
}
```

2 Show the Figures

```
pkgs <- c("DAAG","rpart","randomForest","MASS","mgcv","kernlab","mlbench")
z <- sapply(pkgs, require, character.only=TRUE, warn.conflicts=FALSE)
if(any(!z)){
  notAvail <- paste(names(z)[!z], collapse=", ")
  print(paste("The following packages need to be installed:", notAvail))
}
```

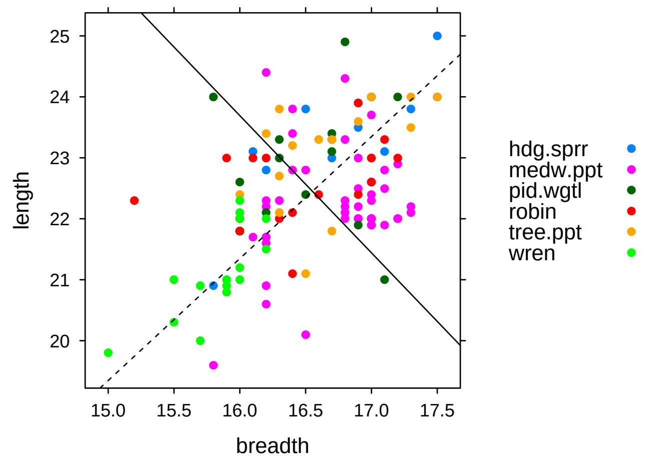
```
getbronchit <- function(){
if(!exists("bronchit")){
  if(require("SMIR")) data("bronchit", package="SMIR") else
    print("Dataset 'bronchit' is not available")
}
if(!exists("bronchit"))
  return("Dataset 'bronchit' is not available") else {
  bronchit <-
     within(bronchit,
         rfac <- factor(r, labels=c("abs","pres")))
}
bronchit
}

bronchit <- getbronchit()
```

```
fig9.1()
```

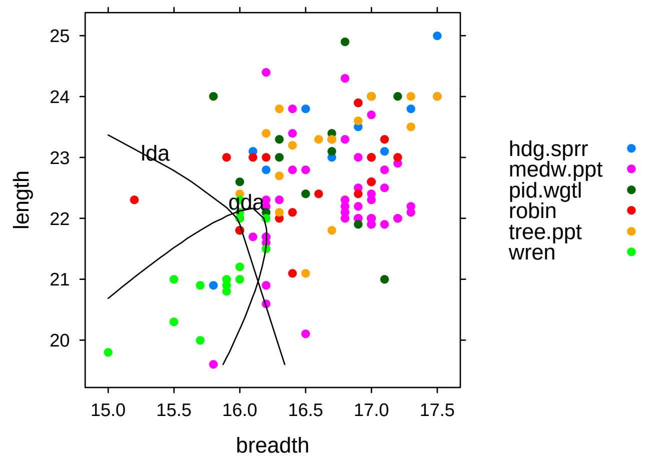


```
    if(!exists('cuckoos.lda')){
        cuckoos <- within(cuckoos,
                          levels(species) <- abbreviate(levels(species), 8))
        cuckoos.lda <- lda(species ~ length + breadth, data=cuckoos)
        cuckoos.qda <- qda(species ~ length + breadth,
                           data=cuckoos)
    }
```

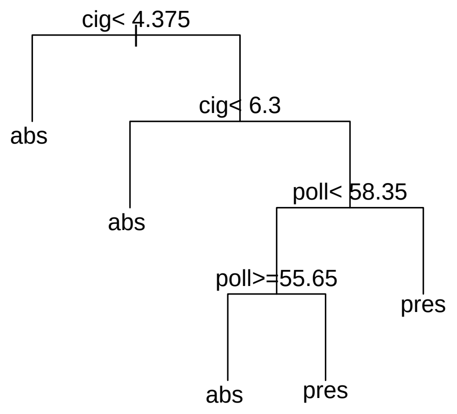
```
fig9.2()
```



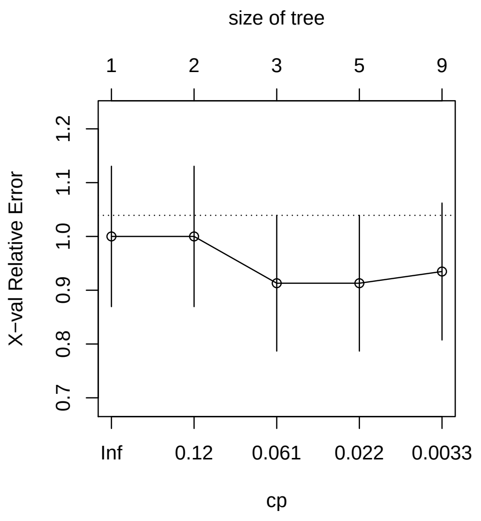
```
fig9.3()
```



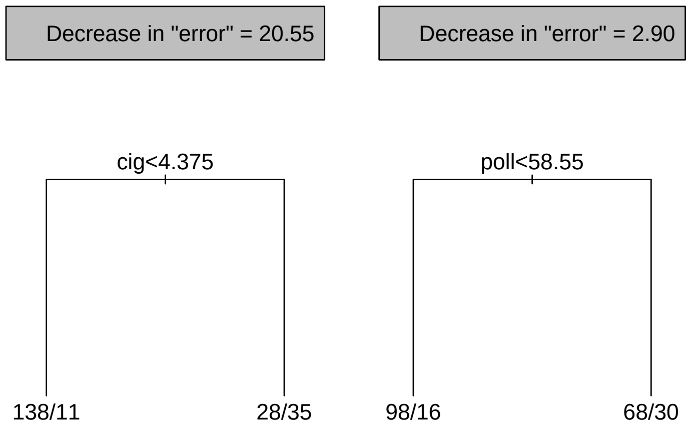
```
if(exists("bronchit")) fig9.4() else
  print("Dataset 'bronchit' was not found; look in SMIR::bronchit")
```



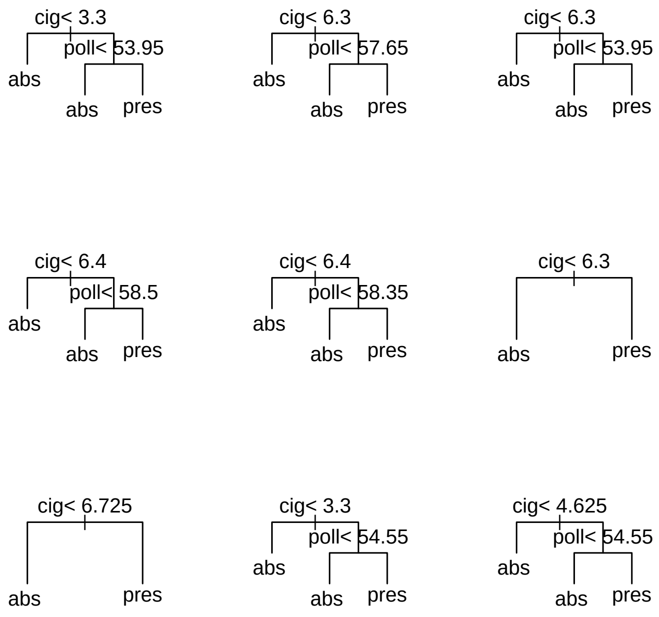
```
if(exists("bronchit")) fig9.5() else
  print("Dataset 'bronchit' was not found; look in SMIR::bronchit")
```



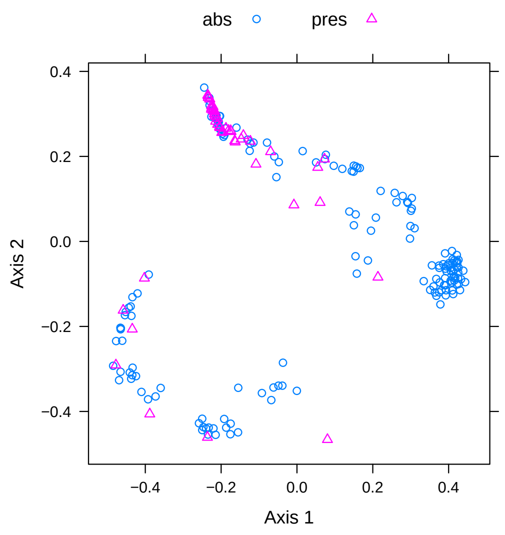
```r
fig9.6()
```



```r
if(exists("bronchit")) fig9.7() else
  print("Dataset 'bronchit' was not found; look in SMIR::bronchit")
```



```
set.seed(31)
if(exists("bronchit")) fig9.8() else
  print("Dataset 'bronchit' was not found; look in SMIR::bronchit")
```



```
if(!exists("spam")){
  if(require("kernlab")) data("spam", package="kernlab") else
    print("Dataset 'spam' is not available")
}
if(exists("spam")){
nr <- sample(1:nrow(spam))
spam0 <- spam[nr[1:2601],]      ## Training
spam1 <- spam[nr[2602:3601],]   ## Holdout
spam01 <- spam[nr[1:3601],]     ## Use for training,
                                ## if holdout not needed
spam2 <- spam[nr[3602:4601],]   ## Test
spam01.lda <- lda(type~., data=spam01)
ldaError <- ldaErr()
set.seed(29)     ## Make results precisely reproducible
spam01.rp <- rpart(type~., data=spam01, cp=0.0001)
rpartError <- rpartErr()
set.seed(29)
spam01.rf <- randomForest(type ~ ., data=spam01)
rfError <- rfErr()
}
```

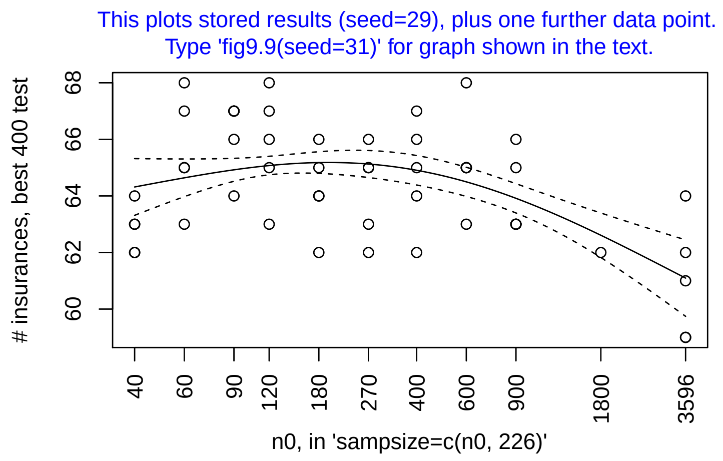
```r
if(!exists('ticShown') | !exists('ticHeld')){
        cat("Will try to load dataset 'ticdata' from package 'kernlab'")
        if(require("kernlab"))
          data("ticdata", package="kernlab") else
          print("Dataset 'ticdata' is not available; get from kernlab")
if(exists('ticdata')){
          ## Use first 5822 observations for prediction
          ticShown <- ticdata[1:5822, ]
          ticHeld <- ticdata[-(1:5822), ]
      }
}
```

```
Will try to load dataset 'ticdata' from package 'kernlab'
```

```r
if(!exists('tictrain') | !exists('tictest')){
tictrain <- ticShown[1:3822, ]
tictest <- ticShown[-(1:3822), ]
}
```

```r
## Generated with seed=29
testLong <-
structure(list(test = c(61, 63, 65, 66, 65, 65, 67, 67, 63, 62,
62, 63, 65, 62, 65, 64, 63, 67, 67, 62, 59, 66, 68, 65, 62, 66,
66, 64, 65, 63, 59, 63, 65, 64, 66, 62, 65, 67, 65, 64, 64, 65,
63, 67, 63, 64, 68, 66, 68, 63), n0 = c(3596, 900, 600, 400,
270, 180, 120, 90, 60, 40, 3596, 900, 600, 400, 270, 180, 120,
90, 60, 40, 3596, 900, 600, 400, 270, 180, 120, 90, 60, 40, 3596,
900, 600, 400, 270, 180, 120, 90, 60, 40, 3596, 900, 600, 400,
270, 180, 120, 90, 60, 40), gp = c(1L, 1L, 1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 3L, 3L, 3L,
3L, 3L, 3L, 3L, 3L, 3L, 3L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 4L,
4L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L, 5L)), .Names = c("test",
"n0", "gp"), row.names = c(NA, -50L), class = "data.frame")
```

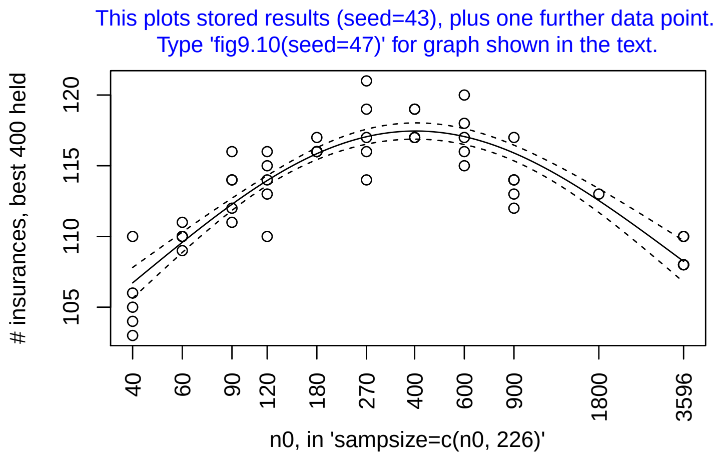
```r
opar <- par(mar=c(4.6,4.6,2.6, 0.6))
note <- paste("This plots stored results (seed=29), plus one further data point.",
              "\nType 'fig9.9(seed=31)' for graph shown in the text.")
oneExtra <- fig9.9(nn0 = 1800, repeats=1, plotit=FALSE)
df <- rbind(testLong, oneExtra)
nn0 <- unique(df$n0)
ndistinct <- length(unique(nn0))
test.gam <- gam(test ~ s(log(n0), k=min(ndistinct,3)), data=df)
plot(test.gam, se=T, residuals=T, pch=1, xaxt="n",
     xlab="n0, in 'sampsize=c(n0, 226)'",
     ylab="# insurances, best 400 test",
     shift=mean(fitted(test.gam)))
axis(1, at=log(nn0), labels=paste(nn0), las=3)
mtext(side=3, line=0.5, note, col="blue")
par(opar)
```



```
## Generated with seed=43
heldLong <-
structure(list(insure = c(108, 114, 120, 119, 121, 116, 114,
114, 110, 103, 110, 114, 116, 117, 117, 116, 110, 112, 110, 110,
110, 112, 118, 119, 119, 117, 113, 116, 110, 106, 108, 113, 115,
117, 114, 116, 116, 111, 109, 105, 108, 117, 117, 117, 116, 116,
115, 114, 111, 104), n0 = c(3596, 900, 600, 400, 270, 180, 120,
90, 60, 40, 3596, 900, 600, 400, 270, 180, 120, 90, 60, 40, 3596,
900, 600, 400, 270, 180, 120, 90, 60, 40, 3596, 900, 600, 400,
270, 180, 120, 90, 60, 40, 3596, 900, 600, 400, 270, 180, 120,
90, 60, 40), gp = c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 2L,
2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 2L, 3L, 3L, 3L, 3L, 3L, 3L, 3L,
3L, 3L, 3L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 4L, 5L, 5L, 5L,
5L, 5L, 5L, 5L, 5L, 5L, 5L)), .Names = c("insure", "n0", "gp"
), row.names = c(NA, -50L), class = "data.frame")
```

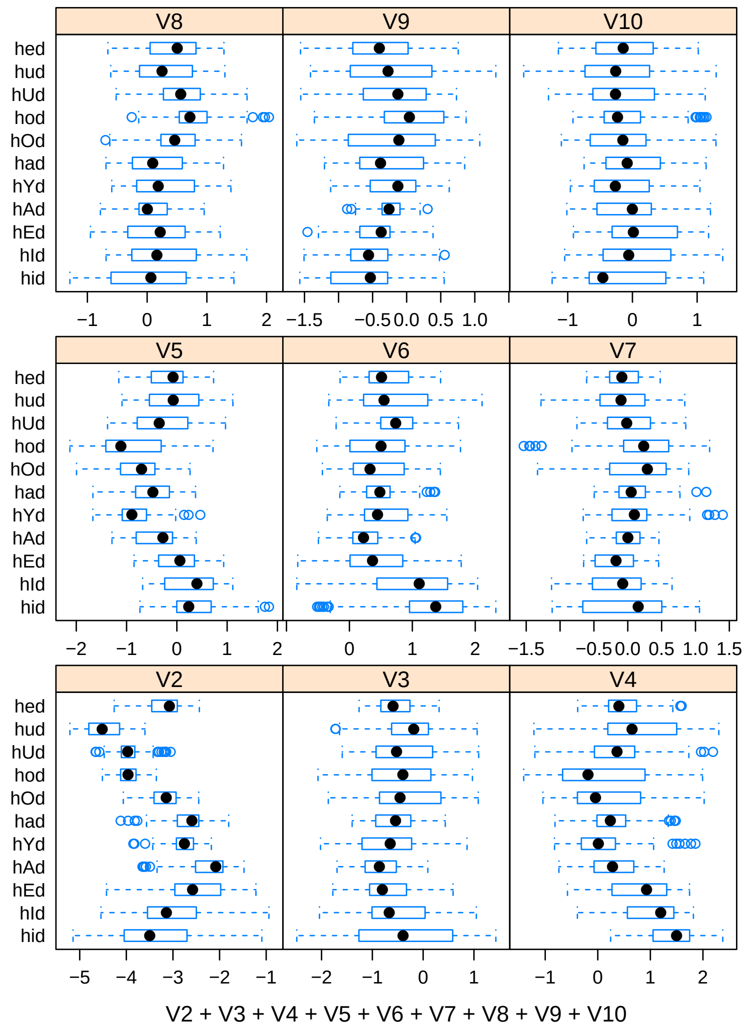
```
opar <- par(mar=c(4.6,4.6,2.6, 0.6))
note <- paste("This plots stored results (seed=43), plus one further data point.",
              "\nType 'fig9.10(seed=47)' for graph shown in the text.")
oneExtra <- fig9.10(nn0 = 1800, repeats=1, plotit=FALSE)
df <- rbind(heldLong, oneExtra)
nn0 <- unique(df$n0)
ndistinct <- length(unique(nn0))
```

```
held.gam <- gam(insure ~ s(log(n0), k=min(ndistinct,3)), data=df)
plot(held.gam, se=T, residuals=T, pch=1, xaxt="n",
     xlab="n0, in 'sampsize=c(n0, 226)'",
     ylab="# insurances, best 400 held",
     shift=mean(fitted(held.gam)))
axis(1, at=log(nn0), labels=paste(nn0), las=3)
mtext(side=3, line=0.5, note, col="blue")
par(opar)
```



```
    if(!exists('Vowel')){
        cat("Will try to load dataset 'Vowel' from package 'mlbench'")
        if(!requireNamespace("mlbench"))
          print("Package 'mlbench' is not installed") else
            data("Vowel", package="mlbench", envir=environment())
    }

Will try to load dataset 'Vowel' from package 'mlbench'
```

```
fig9.11()
```


V8
V9
V10
V5
V6
V7
V2
V3
V4
hed
hud
hUd
hod
hOd
had
hYd
hAd
hEd
hId
hid
V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10