Prediction with SEM: CERQ example

true

July 27, 2021

Intro

I made an R-function for the SEM based prediction rule and in this note I will analyze the `CERQ` data with this rule. In this data set we have 4 items for each of 9 scales that measure emotion regulation. As a response variable we have the SCL score (actually we have item scores also) measured at a later time point. At the first timepoint we also have a SCL score. For more detail see https://scholarlypublications.universiteitleiden.nl/access/item%3A2871874/view

Data set

```r
source('~/surfdrive/Predictive-Psychometrics/paper/SEM-Predictive Validity/versie2/Rcode/predict
load("~/surfdrive/Shared/pred_val_shared/Project_2/p2_application/CERQ Project/CERQdepr_12.Rdata
load("~/surfdrive/Shared/pred_val_shared/Project_2/p2_application/CERQ Project/wideCERQ_T1_T2_T3
# merge
CERQdepr_12 = merge(CERQdepr_12, wideCERQ_T1[ , c(1, 18)])

# depression scale at T = 1
SCLt1 = CERQdepr_12[, 76]

# response variable: scale score
SCLt2 = CERQdepr_12[, 72]

#response variable: item scores
SCLi = CERQdepr_12[, c(56:71)]
colnames(SCLi) = paste("d", c(1:16), sep = "")

# item data
CERQi = CERQdepr_12[ , c(2:5, 8:11, 14:17, 20:23, 26:29, 32:35, 38:41, 44:47, 50:53)]
colnames(CERQi) = c("sb1",
                    "sb2",
                    "sb3",
                    "sb4",
                    "ac1",
                   "ac2",
                   "ac3",
                   "ac4",
                   "ru1",
                   "ru2",
                   "ru3",
                   "ru4",
                   "rf1",
                   "rf2",
                   "rf3",
                   "rf4",
                   "rp1",
                   "rp2",
                   "rp3",
                   "rp4",
                   "ra1",
                   "ra2",
                   "ra3",
                   "ra4",
                   "pp1",
                   "pp2",
                   "pp3",
                   "pp4",
                   "ca1",
                   "ca2",
                   "ca3",
                   "ca4",
                   "bo1",
                   "bo2",
                   "bo3",
                   "bo4"
                   )
# scale data
CERQt = CERQdepr_12[ , c(6, 12, 18, 24, 30, 36, 42, 48, 54)]
colnames(CERQt) = c("sb",
                    "ac",
                    "ru",
                    "rf",
                    "rp",
                    "ra",
                    "pp",
                    "ca",
                    "bo"
)

# data set with item scores
mydat1 = cbind(CERQi, SCLt2)
```

```r
mydat1 = mydat1[complete.cases(mydat1), ]
x = as.matrix(mydat1[, 1:36]);
y = mydat1[, 37]

# data set with scale scores
mydat2 = cbind(CERQt, SCLt2)
mydat2 = mydat2[complete.cases(mydat2), ]
```

Analysis

With the following code I define the SEM model. As methods of comparison I use a linear regression on the item scores estimated by various forms of elastic net (including lasso and ridge) as well as ordinary least squares. Also we use a linear regression on the scale scores, which is equivalent to a SEM model with an equality constraint on the factor loadings.

```r
model <- '
  # latent variable definitions
  sb =~ sb1 + sb2 + sb3 + sb4
  ac =~ ac1 + ac2 + ac3 + ac4
  ru =~ ru1 + ru2 + ru3 + ru4
  rf =~ rf1 + rf2 + rf3 + rf4
  rp =~ rp1 + rp2 + rp3 + rp4
  ra =~ ra1 + ra2 + ra3 + ra4
  pp =~ pp1 + pp2 + pp3 + pp4
  ca =~ ca1 + ca2 + ca3 + ca4
  bo =~ bo1 + bo2 + bo3 + bo4
  # regressions
  SCLt2 ~ sb + ac + ru + rf + rp + ra + pp + ca + bo
'

xnames = colnames(CERQi)
ynames = "SCLt2"
```

```r
set.seed(1234)
repeats = 100
PE = data.frame(repetition = rep(1:repeats, each = 14), model = rep(1:14, repeats), pe = rep(0,

folds = rep(1:10, length.out = 240)
for (r in 1:repeats){
  yhat1 = yhat2 = yhat3 = yhat4 = yhat5 = yhat6 = yhat7 = yhat8 = yhat9 = yhat10 = yhat11 = yhat
  folds = sample(folds)
  for(k in 1:10){
    idx = which(folds == k)
    # sem model
    fit.sem <- sem(model, data = mydat1[-idx, ], std.lv = TRUE, meanstructure = TRUE, warn = FAL
    yhat1[idx] = predicty.lavaan(fit.sem, newdata = mydat1[idx, ], xnames = xnames, ynames = yna
    # linear regession scale scores
    fit.lms = lm(SCLt2 ~ sb + ac + ru + rf + rp + ra + pp + ca + bo, data = mydat2[-idx, ])
    yhat2[idx] = predict(fit.lms, newdata = mydat2[idx, ])

    # elastic net on items
    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 1.0)
    out = glmnet(x[-idx, ],y[-idx], alpha = 1.0)
    yhat3[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.9)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.9)
    yhat4[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.8)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.8)
    yhat5[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.7)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.7)
    yhat6[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.6)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.6)
    yhat7[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.5)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.5)
    yhat8[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.4)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.4)
    yhat9[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.3)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.3)
    yhat10[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.2)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.2)
    yhat11[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.1)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.1)
    yhat12[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx], alpha = 0.0)
    out = glmnet(x[-idx, ],y[-idx], alpha = 0.0)
```

```
    yhat13[idx] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    out = lm(SCLt2 ~ ., data = mydat1[-idx, ])
    yhat14[idx] = predict(out, newdata = mydat1[idx, ])

  }# end folds

  pe1 = sqrt(mean((mydat1[, ynames] - yhat1)^2))
  pe2 = sqrt(mean((mydat1[, ynames] - yhat2)^2))
  pe3 = sqrt(mean((mydat1[, ynames] - yhat3)^2))
  pe4 = sqrt(mean((mydat1[, ynames] - yhat4)^2))
  pe5 = sqrt(mean((mydat1[, ynames] - yhat5)^2))
  pe6 = sqrt(mean((mydat1[, ynames] - yhat6)^2))
  pe7 = sqrt(mean((mydat1[, ynames] - yhat7)^2))
  pe8 = sqrt(mean((mydat1[, ynames] - yhat8)^2))
  pe9 = sqrt(mean((mydat1[, ynames] - yhat9)^2))
  pe10 = sqrt(mean((mydat1[, ynames] - yhat10)^2))
  pe11 = sqrt(mean((mydat1[, ynames] - yhat11)^2))
  pe12 = sqrt(mean((mydat1[, ynames] - yhat12)^2))
  pe13 = sqrt(mean((mydat1[, ynames] - yhat13)^2))
  pe14 = sqrt(mean((mydat1[, ynames] - yhat14)^2))

  PE$pe[((r-1)*14 + 1): (r*14)] = c(pe1, pe2, pe3, pe4, pe5, pe6, pe7, pe8, pe9, pe10, pe11, pe1
} # end repetitions
save(PE, file = "xvalcerq.Rdata")
```

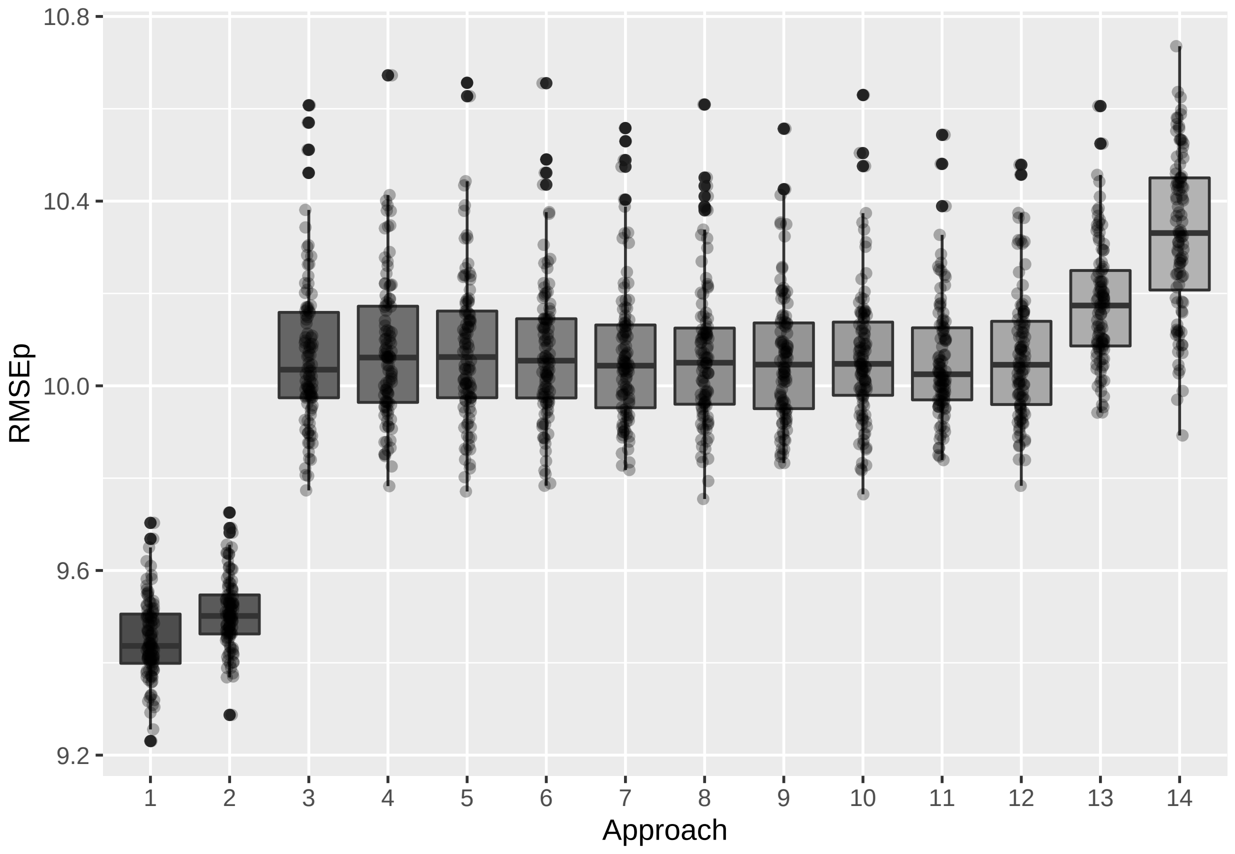
```
pe = cbind(PE[PE$model == 1, 3], PE[PE$model == 2, 3], PE[PE$model == 3, 3], PE[PE$model == 4, 3
table(apply(pe, 1, which.min))
```

```
##
##  1  2
## 94  6
```

```
library(ggplot2)
PE$model = as.factor(PE$model)

p <- ggplot(PE, aes(x=model, y=pe, fill=factor(model)))
    p <- p + geom_boxplot(aes(group = factor(model))) +
      geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
      xlab("Approach") + ylab("RMSEp") + theme(legend.position="none") +
      scale_fill_grey(start=.3,end=.7)

p
```



```
ggsave('~/surfdrive/Predictive-Psychometrics/paper/SEM-Predictive Validity/versie2/Figures/cerqm
```

```
## Saving 6.5 x 4.5 in image
```