Lab 4 complementary assignment

Anton Persson antpe404

21 januari 2017

Complimentary assignment for lab 4 732A95

The task I was about to do as a complimentary assignment was formulated as *Use NIRspectra.xls and Viscosity as a target variable in order to implement a linear regression model in which features are first M components in ICA analysis. Provide a plot showing a mean square prediction error of this model (cross-validation error) versus number of ICA features selected. Compare with a corresponding ´plot in lab 4 and make conclusions.*

I start off by answering the first question, by implementing a linear regression model in which features are first M component in ICA analysis. The code is shown below. Note that I'm using the fastICA() function from the package with the same name to do the ICA analysis.

```
library(ggplot2)
library(fastICA)
```

```
## Warning: package 'fastICA' was built under R version 3.3.2
```

```
nir<-read.csv2("data/NIRspectra.csv", sep=";", header=T)

part_one<-function(data=nir, y=Viscosity, M=3){
  #data=nir
  #y="Viscosity"
  #M=3

  new_x<-data.frame(fastICA(data, n.comp=M)$S) #Extraherar M ICA components.
  newdata<-cbind(Viscosity=data$Viscosity, new_x)
  formula<-as.formula(paste("Viscosity~", paste(colnames(newdata)[-1], collapse="+")))

  des.mat<-model.matrix(formula, newdata)
  dep.var <- all.vars(formula)[1]
  dep.var <- as.matrix(nir[dep.var])
  beta.hat <- solve( t(des.mat) %*% des.mat )  %*% t(des.mat) %*% dep.var
  y.hat <- des.mat %*% beta.hat
  res.err <- dep.var - y.hat
  degree.free <- nrow(des.mat) - ncol(des.mat)
  res.var2 <-( t(res.err) %*% res.err ) / degree.free
  var.hat.bhat <-diag( as.vector(res.var2) *  solve( t(des.mat) %*% des.mat )  )
  t.beta <- beta.hat / sqrt( var.hat.bhat )
  my.pvalues<- (1 - pt( abs( t.beta ) ,df = degree.free) ) * 2

  l<-list(coefficients = t(beta.hat) , degree.free = degree.free, res.var2 = res.var2,
          var.hat.bhat = var.hat.bhat, t.beta = t.beta, my.pvalues = my.pvalues,
          formula = formula, dataset=deparse(substitute(data)),
           data=cbind(des.mat,dep.var),y.hat=y.hat,res.err=res.err)
  return(l$coefficients)
}

part_one()
```

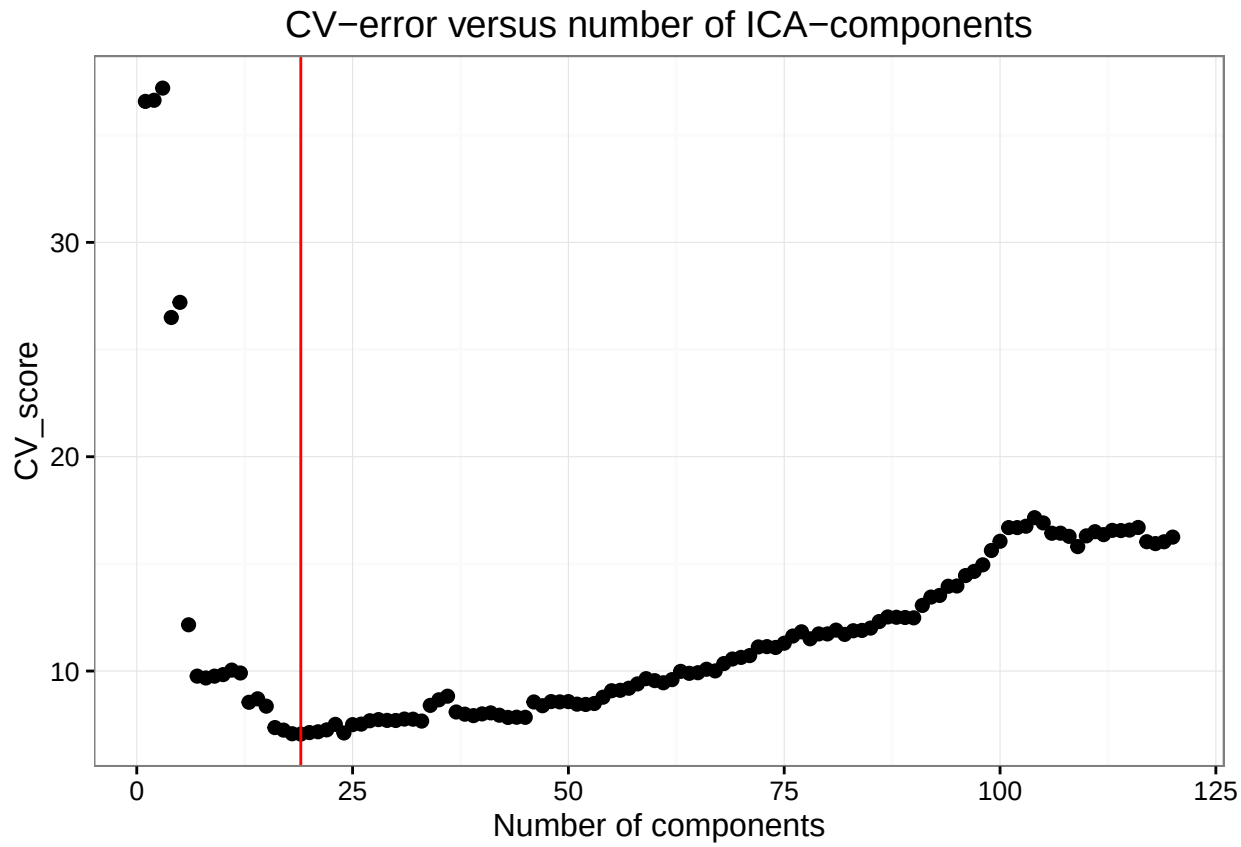
```
##           (Intercept)        X1         X2         X3
## Viscosity    2.516911 0.0510186 0.02516641 -0.5246745
```

In the code chunk above I chose to show only the coefficients for the model when the number of ICA components equals 3. As you can see, from this function I could show almost any values that might be of interest.

For the second part of the question, to run the function over several different values of M and plot their cross validation error versus the number of components, the following code is used.

```
CV_ica <- function(data=nir, folds, M) {
  Y<-data$Viscosity
  Y<-as.matrix(Y)
  CV<-integer(0)
  resmat<-matrix(ncol=M, nrow=folds)

  for (j in 1:M){

    X<-data.frame(fastICA(data[, 1:(ncol(data)-1)], n.comp=(j))$S) #Extraherar M ICA components.
    X<-as.matrix(X)

    n<-dim(X)[1]
    X<-cbind(rep(1, n), X)#Intercept-ettorna for X-matrisen
    p<-dim(X)[2]

    for (i in 1:folds){
      slumpat<-as.vector(unlist(suppressWarnings(split(1:n, f=1:folds)[i])))

      test<-X[slumpat, 1:(j+1)] #+1 eftersom jag har interceptcol
      train<-X[-slumpat, 1:(j+1)]
      Y_test<-Y[slumpat, ]
      Y_train<-Y[-slumpat, ]

      betahat<-solve(t(train)%*%train) %*% t(train) %*% Y_train #Skattar modellen pa traningsdatan
      yhat <- test %*% betahat  #Testar pa den nya datan, dvs skattar nya datan.
      resid_err <- Y_test - yhat #Och tar fram felen.

      CV[i]<-sum(resid_err**2)
    }
    resmat[,j]<-CV
  }
  resmat<-as.data.frame(resmat)
  CV_score<-apply(resmat, 2, mean)
  data<-as.data.frame(cbind(CV_score, M=1:M))
  #data<-as.data.frame(cbind(log_cv=log(CV_score), M=1:M))

  cv_plotten<-ggplot(data=data)+geom_point(aes(x=M, y=CV_score), size=2)+#y=log_cv
    theme_bw()+
    #ylim(c(0,.001))+
    xlab("Number of components")+
    ggtitle("CV-error versus number of ICA-components")+
    geom_vline(xintercept=data$M[data$CV_score==min(data$CV_score)], col="red")

  optimal_subset<-data[data$CV_score==min(data$CV_score),]
```

```
  plot(cv_plotten)
  #return(data)
  return(optimal_subset)
}

CV_ica(data=nir, folds=3, M=120)
```




```
##     CV_score  M
## V19 7.058292 19
```

The CV score gets minimized when 19 ICA components are used, which is clarified by the red vertical line in the plot. I'd say, according to the elbow rule, that it's reasonable to select either seven or about 16-19 components, depending on whether it's most important to minimize the number of components or the prediction error. If I choose seven, I'm minimizing the number of components and still has a prediction error below 10. If it's really important to have a low prediction error, I'd go with 16-19 components. However, there's no reason to use more than 19 components.

Note that all code used in this complimentary assignment is already visualized in the code chunks, why I have no appendix.