Assignment 1
Assigned 2018-01-11  Due 2018-01-21 5p

This is a individual assignment except for problem 8 for which students may work together. For problems 1–7 your work must be completely independent. Your work must be a finished html file that includes all code.

1. Problem 1 in Chapter 2 of Regression Modeling Strategies
2. Problem 2
3. Problem 3
4. Problem 4
5. Problem 5. The SAT dataset may be created by using the sat.r code in an attachment on the course web page.
6. Derive the formulas for the restricted cubic spline component variables without cubing or squaring any terms.
7. Prove that each component variable is linear in $X$ when $X \geq t_k$, the last knot, using general principles and not algebra or calculus. Derive an expression for the restricted spline regression function when $X \geq t_k$.
8. Consider a 3–dimensional surface relating $X_1$ and $X_2$ to $C(Y|X_1,X_2)$ defined by a patch–wise cubic polynomial. The patches are formed by a grid of knots for $X_1 (u_1,\ldots,u_k)$ and for $X_2 (v_1,\ldots,v_k)$. Each polynomial is of the form
   \[ f(X_1,X_2) = \sum_{i=j=0}^{3} \beta_{ij}X_1^iX_2^j, \]
   but is written in terms of an offset from polynomials below and to the left to facilitate continuity restrictions in a) below. For example, if $k = 1$, define four polynomials for four quadrants
For bivariate knots \((u_1, v_1)\) let \(f_2 = f_1 + \) terms in \((X_1 - u_1)\), \(f_3 = f_1 + \) terms in \((X_2 - v_1)\), and \(f_4\) involves both “knot crossings.” Derive equations for \(f(X_1, X_2)\) or for its component terms under the following two conditions.

(a) Restrict this 3–dimensional spline function so that \(f(\cdot)\) is continuous and has continuous first and second derivatives — \(f(\cdot)\) agrees at the rectangle boundaries and so does \(\frac{\partial f}{\partial X_1}, \frac{\partial f}{\partial X_2}\), and \(\frac{\partial^2 f}{\partial X_1 \partial X_2}\).

(b) Further restrict \(f(\cdot)\) so that \(f\) is of the form \(aX_1 + bX_2 + cX_1X_2\) if \(X_1 \leq u_1\) and \(X_2 \leq v_1\) or if \(X_1 \geq u_k\) and \(X_2 \geq v_k\).
Assignment 2  Assigned 2018-01-24  Due 2018-01-30

This is a group assignment. Groups are defined in the following table. One solution should be turned in per group, and the work should list all members who contributed meaningfully to the work. Groups must not help each other except for general questions posted to slack.

<table>
<thead>
<tr>
<th>Student</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elizabeth Sigworth</td>
<td>1</td>
</tr>
<tr>
<td>Mary Lauren Benton</td>
<td>1</td>
</tr>
<tr>
<td>Michael Pritchard</td>
<td>1</td>
</tr>
<tr>
<td>Sarah Lotspeich</td>
<td>1</td>
</tr>
<tr>
<td>Elizabeth McNeer</td>
<td>2</td>
</tr>
<tr>
<td>Josh DeClercq</td>
<td>2</td>
</tr>
<tr>
<td>Rui Wang</td>
<td>2</td>
</tr>
<tr>
<td>Tin Nguyen</td>
<td>2</td>
</tr>
<tr>
<td>Woo-yeol Lee</td>
<td>2</td>
</tr>
<tr>
<td>Dinesh Karki</td>
<td>3</td>
</tr>
<tr>
<td>Kim Kondratieff</td>
<td>3</td>
</tr>
<tr>
<td>Nathan James</td>
<td>3</td>
</tr>
<tr>
<td>Valerie Welty</td>
<td>3</td>
</tr>
</tbody>
</table>

Consider a two–stage procedure in which one tests for linearity of the effect of a predictor $X$ on a property of the response $C(Y | X)$ against a quadratic alternative. If the two–tailed test of linearity is significant at the $\alpha$ level, a two d.f. test of association between $X$ and $Y$ is done. If the test for linearity is not significant, the square term is dropped and a linear model is fitted. The test of association between $X$ and $Y$ is then (apparently) a one d.f. test.

1. Write a formal expression for the test statistic for association.

2. Write an expression for the nominal $P$–value for testing association using this strategy.

3. Write an expression for the actual $P$–value or alternatively for the type–I error if using a fixed critical value for the test of association.

4. For the same two–stage strategy consider an estimate of the effect on $C(Y | X)$ of increasing $X$ from $a$ to $b$. Write a brief symbolic algorithm for deriving a true two–sided $1 – \alpha$ confidence interval for the $b : a$ effect (difference in $C(Y)$) using the bootstrap.
Assignment 3  Assigned 2018-01-31  Due 2018-02-07

This is a group assignment. Groups are defined in the following table. One solution should be turned in per group, and the work should list all members who contributed meaningfully to the work. Groups must not help each other.

<table>
<thead>
<tr>
<th>Student</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Elizabeth McNeer</td>
<td>1</td>
</tr>
<tr>
<td>2 Josh DeClercq</td>
<td>1</td>
</tr>
<tr>
<td>3 Kim Kondratieff</td>
<td>1</td>
</tr>
<tr>
<td>4 Woo-yeol Lee</td>
<td>1</td>
</tr>
<tr>
<td>5 Dinesh Karki</td>
<td>2</td>
</tr>
<tr>
<td>6 Elizabeth Sigworth</td>
<td>2</td>
</tr>
<tr>
<td>7 Rui Wang</td>
<td>2</td>
</tr>
<tr>
<td>8 Sarah Lotspeich</td>
<td>2</td>
</tr>
<tr>
<td>9 Tin Nguyen</td>
<td>2</td>
</tr>
<tr>
<td>10 Mary Lauren Benton</td>
<td>3</td>
</tr>
<tr>
<td>11 Michael Pritchard</td>
<td>3</td>
</tr>
<tr>
<td>12 Nathan James</td>
<td>3</td>
</tr>
<tr>
<td>13 Valerie Welty</td>
<td>3</td>
</tr>
</tbody>
</table>

To access the support dataset you can use the command getHdata(support) once you have access to the Hisc package (which is automatic if you access the rms package).

1. Chapter 3 Problem 1
2. Chapter 3 Problem 2
3. State briefly why single conditional median\(^1\) imputation is OK here.
4. Use transcan to develop single imputations for total cost, commenting on the strength of the model fitted by transcan as well as how strongly each variable can be predicted from all the others.
5. Use predictive mean matching to multiply impute cost 10 times per missing observation. Describe graphically the distributions of imputed values and briefly compare these to distributions of non-imputed values. State in a simple way what the sample variance of multiple imputations for a single observation of a continuous predictor is approximating.
6. Using the multiple imputed values, develop an overall least squares model for total cost (using the log transformation) making optimal use of partial information, with variances computed so as to take imputation (except for cost) into account. The model should use the predictors in Problem 1 and should not assume linearity in any predictor but should assume additivity. Interpret one of the resulting ratios of imputation-corrected variance to apparent variance and explain why ratios greater than one do not mean that imputation is inefficient.

\(^1\)We are anti-logging predicted log costs and we assume log cost has a symmetric distribution
Assignment 4  Assigned 2018-02-08  Due 2018-02-18

The goal is to understand the performance of various internal validation methods for binary logistic models using Monte Carlo simulation. Your assignment is to modify the simulation in at least two meaningful ways with regard to covariate distribution or number, sample size, true regression coefficients, and number of times certain strategies are averaged. You should interpret your findings and give recommendations for best practice for the type of configuration you studied. Store the simulation summary in an object named \texttt{valSimresult} just as is done below so that your results can later be combined with results of other simulations. Turn in the summary object \texttt{valSimresult.rda} file electronically along with your report. This file contains, for each simulation, the difference between the estimated and the independently validated statistical index. The R code from this assignment may be downloaded from the course wiki page.

This is an \textbf{individual} assignment.

Simulation Method  For each of 200 simulations generate a training sample of 200 observations with \( p \) predictors \((p = 15 \text{ or } 30)\) and a binary response. The predictors are independently \( U(-0.5, 0.5) \). The response is sampled so as to follow a logistic model where the intercept is zero and all regression coefficients equal 0.5 (which is admittedly not very realistic). The “gold standard” is the predictive ability of the fitted model on a test sample containing 50,000 observations generated from the same population model.

Validation Methods  For each of the 200 training and validation samples several validation methods were employed to estimate how the training sample model predicts responses in the 50,000 observations. These validation methods involving fitting 40 or 200 models per training sample.

\( g \)-fold cross-validation is done using the command \texttt{validate(f, method='cross', B=4 \text{ or } B=10)} using the rms package. This was repeated and averaged using an extra loop, shown below.

For bootstrap methods \texttt{validate(f, method='boot' \text{ or } '632', B=40 \text{ or } B=200)} was used. \texttt{method='632'} does Efron’s “.632” method, labeled \texttt{632a} in the output. An ad-hoc modification of the .632 method, \texttt{632b} was also done. Here a “bias-corrected” index of accuracy is simply the index evaluated in the observation omitted from the bootstrap re-sample.

The “gold standard” external validations were done using the \texttt{val.prob} function in the rms package.

Indexes of Predictive Accuracy

\( D_{xy} \): Somers’ rank correlation between predicted probability that \( Y = 1 \) vs. the binary \( Y \) values. This equals \( 2(C - 0.5) \) where \( C \) is the “ROC Area” or concordance probability.

\( D \): Discrimination index — likelihood ratio \( \chi^2 \) divided by the sample size

\( U \): Unreliability index — unitless index of how far the logit calibration curve intercept and slope are from \((0, 1)\)

\( Q \): Logarithmic accuracy score — a scaled version of the log-likelihood achieved by the predictive model

Intercept: Calibration intercept on logit scale

Slope: Calibration slope (slope of predicted log odds vs. true log odds)

Measure of Accuracy of Validation Estimates  Root mean squared error of estimates (e.g., of \( D_{xy} \) from the bootstrap on the 200 observations) against the “gold standard” (e.g., Dxy for the fitted 200-observation model achieved in the 50,000 observations).

```
require(rms)
dosims ← TRUE
set.seed(1)  # so can reproduce results
```
n ← 200                # Size of training sample
reps ← 200               # Simulations
npop ← 50000             # Size of validation gold standard sample
methods ← c('Boot 40', 'Boot 200', '632a 40', '632a 200',
             '632b 40', '632b 200', '10-fold x 4', '4-fold x 10',
             '10-fold x 20', '4-fold x 50')
R ← expand.grid(sim = 1:reps,
                p = c(15, 30),
                method = methods)
R$Dxy ← R$ Intercept ← R$ Slope ← R$ D ← R$ U ← R$ Q ← R$ repmeth ← R$ B
R$n ← n

## Function to do r overall reps of B resamples, averaging to get
## estimates similar to as if r*B resamples were done
val ← function(fit, method, B, r) {
  contains ← function(m) length(grep(m, method)) > 0
  meth ← if(contains('Boot')) 'boot' else
         if(contains('fold')) 'crossvalidation' else
         if(contains('632')) '.632'
  z ← 0
  for(i in 1:r) z ← z + validate(fit, method=meth, B=B)[
    c("Dxy", "Intercept", "Slope", "D", "U", "Q"), 'index.corrected'
  ]
  z/r
}

for(p in c(15, 30)) {

  ## For each p create the true betas, the design matrix, 
  ## and realizations of binary y in the gold standard large sample
  Beta ← rep(.5, p)             # True betas
  X ← matrix(runif(npop*p), nrow=npop) - 0.5
  LX ← matxv(X, Beta)
  Y ← ifelse(runif(npop) <= plogis(LX), 1, 0)

  ## For each simulation create the data matrix and realizations of y
  for(j in 1:reps) {
    cat(j, file='/tmp/z.txt ')

    ## Make training sample
    x ← matrix(runif(n*p), nrow=n) - 0.5
    L ← matxv(x, Beta)
    y ← ifelse(runif(n) <= plogis(L), 1, 0)
    f ← lrm(y ~ x, x=TRUE, y=TRUE)
    beta ← f$coef
    forecast ← matxv(X, beta)

    ## Validate in population
    v ← val.prob(logit=forecast, y=Y, pl=FALSE)[
      c("Dxy", "Intercept", "Slope", "D", "U", "Q")]

    for(method in methods) {
      repmeth ← 1
      if(method %in% c('Boot 40', '632a 40', '632b 40')) B ← 40
  }
if(method %in% c('Boot 200', '632a 200', '632b 200')) B ← 200
if(method == '10-fold x 4') {
  B ← 10
  repmeth ← 4
}
if(method == '4-fold x 10') {
  B ← 4
  repmeth ← 10
}
if(method == '10-fold x 20') {
  B ← 10
  repmeth ← 20
}
if(method == '4-fold x 50') {
  B ← 4
  repmeth ← 50
}

z ← val(f, method, B, repmeth)
k ← which(R$sim == j & R$p == p & R$method == method)
if(length(k) != 1) stop('program logic error')
R[k, names(z)] ← z - v
R[k, c('B', 'repmeth')] ← c(B=B, repmeth=repmeth)
}
# end over methods
}
# end over reps
valSimresult ← R
Save(valSimresult)

Results are best summarized in a multi-way dot chart. Bootstrap nonparametric percentile 0.95 confidence limits are included.

statnames ← names(R)[6:11]
w ← reshape(R, direction='long', varying=list(statnames),
v.names='x', timevar='stat', times=statnames)
w$p ← paste('p', w$p, sep='=')
s ← with(w, summarize(abs(x), llist(p, method, stat), smean.cl.boot,
stat.name='mae'))
require(ggplot2)
ggplot(s, aes(x=mae, y=method)) + geom_point() + facet_grid(stat ~ p) +
  geom_errorbarh(aes(xmin=Lower, xmax=Upper, y=method), height=0) +
xlab('Mean |error|')
s ← with(w, summarize(x^2, llist(p, method, stat), smean.cl.boot,
stat.name='mse'))
ggplot(s, aes(x=sqrt(mse), y=method)) + geom_point() + facet_grid(stat ~ p ) +
  geom_errorbarh(aes(xmin=sqrt(Lower), xmax=sqrt(Upper), y=method), height =0) +
xlab(expression(sqrt(MSE)))
Assignment 5  Assigned 2018-02-19  Due 2018-02-27

Do the problems at the end of Chapter 8 in the second edition. Consider stage as linear for transcan because its excessive ties prevent knot identification.

This is an individual project, counting 1.5 times each of the previous homework assignments. Your interpretations will be key.

Some R programming hints follow.

```r
# To subset a data frame and run varclus on all remaining variables:
require(Hmisc)
m ← subset(mydata, row.subsetting.expression,
     select=-c(x17,x19,x21))  # exclude 3 vars
plot(varclus(~., data=m))

# Function to compute first k PCs of a matrix of numeric variables
pcl ← function(x, k=1) {
   g ← prcomp(x, scale=TRUE)
   g$x[, 1:k]
}

# Correlate PC1 with some transformed individual variables
vars ← trans[,c('x1','x2','x3')]
cor(pcl(vars), vars)

# Cumulative proportion of variance explained by PCs: see
# addscree function in text

# Subset to complete cases for numeric variables
w ← subset(m, !is.na(x1 + x2 + x3))
# Cs is in Hmisc - allows one to omit quote marks
# Create a matrix containing numeric variables in data frame
a ← as.matrix(w[,Cs(x1,x3,x7)])
# Augment the matrix with binary variable translations of
# some categorical variables
a ← with(w, cbind(a,
   male=1*(sex == 'male'),
   ...))

# Get another set of variables
b ← as.matrix(w[,Cs(x2,x4)])
# Compute correlation of two PC1s for two sets of variables
cor(pcl(a), pcl(b))

# Function to compute transcan transformations
ttrans ← function(x) {
   # asis= tells transcan to leave some variables untransformed
   z ← transcan(x, transformed=TRUE, data=w, pr=FALSE, pl=FALSE,
   asis=Cs(male,...))
   z$transformed
}

# Function to compute first nonlinear principal component
npc1 ← function(x) pcl(ttrans(x))
cor(npc1(a), npc1(b))

# Function to compute first canonical correlation across 2 matrices
```

8
cancor1 ← function(X, Y) cancor(X, Y)$cor[1]
cancor1(a, b)
cancor1(ttrans(a), ttrans(b))  # Can. corr. on transformed vars

# Cox PH model
S ← with(mydata, Surv(followup.time, binarystatusindicator))
f ← cph(S ~ ..., x=TRUE)
X ← f$x  # save numeric design matrix for later

# Compute first 3 PCs for whole dataset
pc3orig ← pc1(X, 3)

# Simulate to get bootstrap percentile CLs
sim ← function(B, type=1) {
  # type = 2 to recompute PCs
  ...  
inversions ← 0
  for(i in 1:B) {  
    j ← set.of.subscripts.in.current.bootstrap.sample
    if(type == 2) {
      pc3 ← pc1(X[j,], 3)
      Sj ← S[j,]
      # PCs are not unique to within a sign inversion
      # Flip PCs to make them positively correlated with original PCs
      for(k in 1:3) {
        if(cor(pc3orig[j,k], pc3[,k]) < 0) {
          inversions ← inversions + 1
          pc3[,k] ← - pc3[,k]
        }
      }
      f ← cph(Sj ~ pc3)
    } else f ← cph(S ~ pc3, subset=j)
    pc1coef[i] ← coef(f)[1]
    orig.coef ← coef(lsfit(X, predict(f)))
  }
  if(inversions > 0) cat('inversions:', inversions, '\n')
  ...  # compute 3 quartiles of 2 estimates
}
Assignment 6  Assigned 2018-03-01  Due 2018-03-14

Do the problems at the end of Chapter 9 in the second edition.

This is an individual project, counting 1.5 times group assignments. Interpretations always receive significant weight. The last problem may be done in groups, with each student composing her own interpretation of the (common) results.

Some R code for the last problem is given below. See http://www.sumsar.net/blog/2015/07/easy-bayesian-bootstrap (reproduced at http://www.r-bloggers.com/easy-bayesian-bootstrap-in-r) for a nice introduction to the Bayesian bootstrap and its approximation, including some R code snippets below.

Extra credit will be given for any individual who adds the bootstrap-t confidence interval to this simulation study.

```r
require(rms)
require(ProfileLikelihood)
n ← 200  # sample size
m ← 1000  # no. simulations
B ← 1000  # no. bootstrap reps per simulation
n2 ← max(n, 1000)  # no. to sample with approx Bayes bootstrap
set.seed(13)

x1 ← exp(rnorm(n))
X ← cbind(x1, x1^2)
logit ← 1 + x1 / 2
P ← plogis(logit)
dd ← datadist(x1); options(datadist='dd')
trueLOR ← (5 - 1) / 2
lims ← c('Lower 0.95 ', 'Upper 0.95 ')
meths ← c('Wald ', 'Sandwich ', 'Bootstrap Percentile ', :
'Bootstrap BCa ', 'Bootstrap Basic ', 'Bayesian Bootstrap ', :
'Approx. Bayesian Bootstrap ', 'Profile Likelihood ')
r ← array(NA, dim=c(m, 8, 2), :
dimnames=list(NULL, meths, c('Lower ', 'Upper ')))
estLOR ← numeric(m)
options(showprogress=FALSE)

bayesboot ← function(type=c('bayes', 'approx')) {
  fit ← function(subs=1:n, weights=rep(1, n)) {
    tryCatch(lrm.fit(X[subs,], y[subs], weights=weights[subs],
                    normwt=TRUE)$coefficients,
       error=function(...) {cat('could not fit
'); c(NA,NA,NA)})
  type ← match.arg(type)
  lors ← numeric(B)
  for(j in 1:B) {
    wts ← rexp(n); wts ← wts / sum(wts)
    cof ← if(type == 'bayes') fit(weights=wts)
        else fit(subs=sample(1:n, size=n2, replace=TRUE, prob=wts))
  }
  quantile(lors, c(0.025, 0.975), na.rm=TRUE)
}
```

**Extra credit** will be given for any individual who adds the bootstrap-t confidence interval to this simulation study.

**## Compute design matrix that reparameterizes the model so that the last coefficient is the estimate of the log OR for x1=5:1, for
**## use with profile likelihood method
**## Original: a + bx + cax^2. Estimand: 24c + 4b = k; c = (k - 4b)/24**
\[ a + b \frac{x + (k - 4b) / 2}{2} x^2 = a + b \frac{x}{2} x^2 + b / 6 x^2 = \]
\[ a + b \frac{x - z^2 / 6}{2} + k \frac{x^2 / 2}{4} = \]
\[ X_p \leftarrow \text{cbind}(x_1 - x_1 * x_1 / 6, x_1 * x_1 / 24) \]

for (i in 1:m) {
  cat(i, '\n', file='/tmp/progress.txt')
  y ← ifelse(runif(n) <= P, 1, 0)
  f ← lrm(y ~ pol(x1,2), x=TRUE, y=TRUE)
  s ← summary(f, x1=c(1,5))
  estLOR[i] ← s['x1', 'Effect']
  r[i, 'Wald',] ← s['x1', 'lms']
  rob ← robcov(f)
  r[i, 'Sandwich',] ← summary(rob, x1=c(1,5))['x1', 'lms']
  b ← bootcov(f, B=B)
  r[i, 'Bootstrap Percentile',] ← summary(b, x1=c(1,5))['x1', 'lms']
  r[i, 'Bootstrap Basic',] ← summary(b, x1=c(1,5), boot.type='basic')['x1', 'lms']
  r[i, 'Bootstrap BCa',] ← summary(b, x1=c(1,5), boot.type='bcap')['x1', 'lms']
  r[i, 'Approx. Bayesian Bootstrap',] ← bayesboot('approx')
  pdata ← data.frame(y=y, x1=Xp[,1], lor51=Xp[,2])
  fg ← glm(y ~ Xp, family=binomial)
  u ← profilelike.glm(y ~ x1, data=pdata,
    profile.theta='lor51', family=binomial(link="logit"),
    length=300, round=3)
  pl.ci ← profilelike.summary(k=8, theta=u$theta,
    profile.lik.norm=u$profile.lik.norm, round=3) $ LI.norm
  r[i, 'Profile Likelihood',] ← pl.ci

## Note: the following will not work because the R built-in
## profile likelihood method has false convergence
  r[i, 'Profile Likelihood',] ← confint(fg, 3)
}

truecl ← quantile(estLOR, c(.025, .975))
cat('True 0.95 CL: ', truecl[1], truecl[2], ' \nTail coverages: \n')
mn ← function(x) mean(x, na.rm=TRUE)
res ← cbind(Nlower=NA, Nupper=NA, r[i,], Overall=NA)
for (z in meths) res[z,] ← c(
  sum(! is.na(r[,z,'Lower'])),
  sum(! is.na(r[,z,'Upper'])),
  mn(r[,z,'Lower'] > trueLOR),
  mn(r[,z,'Upper'] < trueLOR),
  mn(trueLOR > r[,z,'Lower'] & trueLOR < r[,z,'Upper'])))
print(res)
Save(res)
Assignment 7  Assigned 2018-03-15  Due 2018-03-23

Do the problems at the end of Chapter 13 in the second edition. The last problem is for significant extra credit. This is an individual assignment, counting more than group assignments. For problem 4, also use $n$ groups where $n$ is the number of distinct cost values. The \texttt{rms orm} function efficiently handles large numbers of distinct $Y$ values.
Assignment 8  Assigned 2018-03-24  Due 2018-04-03

This is a group assignment. Groups are defined in the following table. One solution should be turned in per group, and the work should list all members who contributed meaningfully to the work. Groups must not help each other.

<table>
<thead>
<tr>
<th>Student</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Josh DeClercq</td>
<td>1</td>
</tr>
<tr>
<td>Kim Kondratieff</td>
<td>1</td>
</tr>
<tr>
<td>Mary Lauren Benton</td>
<td>1</td>
</tr>
<tr>
<td>Nathan James</td>
<td>1</td>
</tr>
<tr>
<td>Elizabeth Sigworth</td>
<td>2</td>
</tr>
<tr>
<td>Michael Pritchard</td>
<td>2</td>
</tr>
<tr>
<td>Sarah Lotspeich</td>
<td>2</td>
</tr>
<tr>
<td>Dinesh Karki</td>
<td>3</td>
</tr>
<tr>
<td>Elizabeth McNeer</td>
<td>3</td>
</tr>
<tr>
<td>Valerie Welty</td>
<td>3</td>
</tr>
</tbody>
</table>

Work the problems at the end Chapter 14 in the second edition, and in addition:

10. Derive the analytic solution of the MLE of $\beta$ in a two-group proportional odds model without covariates. Use this notation: response variable on $n$ observations is $Y_1, Y_2, \ldots, Y_n$, distinct values of $Y$ are $y_1, y_2, \ldots, y_k$, first $n_a$ observations are from group $A$ and second $n_b$ observations are on group $B$. You can use groups 1 and 2 instead of $A$ and $B$ if you prefer. If you are unable to derive an analytic solution, derive a one-step estimator (one iteration of the Newton-Raphson algorithm or simple Taylor series approximation). For Newton-Raphson note that the MLE of the intercepts when the first slope parameter is zero are very simple—this is what is used to start the iterations in the \texttt{lrm} and \texttt{orm} functions.

Check your analytic result against the following dataset.

```r
require(rms)

w ← rbind(data.frame(x='a', y=c(0,3,5,5,10,11)),
           data.frame(x='b', y=c(1,3,5,16,17,20))

do coef(orm(y ~ x, data=w, eps=0.00001))
```

<table>
<thead>
<tr>
<th>$y \geq 1$</th>
<th>$y \geq 3$</th>
<th>$y \geq 5$</th>
<th>$y \geq 10$</th>
<th>$y \geq 11$</th>
<th>$y \geq 16$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.9690146</td>
<td>1.1780923</td>
<td>0.2884028</td>
<td>-0.7347160</td>
<td>-1.1056198</td>
<td>-1.5630033</td>
</tr>
<tr>
<td>$y \geq 17$</td>
<td>$y \geq 20$</td>
<td>$x=b$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2.1454948</td>
<td>-2.9889933</td>
<td>1.0199787</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Derive an empirical solution to the problem by simulating a variety of sample sizes and integer-valued $y$ and binary $x$. Use the $c$-index translation of the Wilcoxon statistic to predict $\hat{\beta}$, so as to be independent of sample size. You will be comparing $c$ to $\hat{\beta}$ across simulations, possibly making a sensible transformation of $c$. To compute $c$ you can do something like \texttt{somers2(y,x)["C"]}.
Assignment 9  Assigned 2018-04-05  Due 2018-04-16

This is a group assignment. Groups are defined in the following table. One solution should be turned in per group, and the work should list all members who contributed meaningfully to the work. Groups must not help each other except for the last problem.

<table>
<thead>
<tr>
<th>Student</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kim Kondratieff</td>
<td>1</td>
</tr>
<tr>
<td>Mary Lauren Benton</td>
<td>1</td>
</tr>
<tr>
<td>Michael Pritchard</td>
<td>1</td>
</tr>
<tr>
<td>Sarah Lotspeich</td>
<td>1</td>
</tr>
<tr>
<td>Dinesh Karki</td>
<td>2</td>
</tr>
<tr>
<td>Elizabeth McNeer</td>
<td>2</td>
</tr>
<tr>
<td>Nathan James</td>
<td>2</td>
</tr>
<tr>
<td>Elizabeth Sigworth</td>
<td>3</td>
</tr>
<tr>
<td>Josh DeClercq</td>
<td>3</td>
</tr>
<tr>
<td>Valerie Welty</td>
<td>3</td>
</tr>
</tbody>
</table>

1. Fit a binary logistic model to predict hospital death in the support2 dataset. Use the union of predictors that were used in other assignments in the text, modeling them flexibly. For missing continuous variables, fill in their values using “most normal” values as done in the text. For categorical missing values use the mode.

2. Set up for re-simulation. Save the coefficients from the fitted model and pretend they are population coefficients. Get the predicted probability of hospital death for each patient. For the simulations to follow, simulate Bernoulli random variables having true probability that Y = 1 given by the probabilities of hospital death your just estimated.

3. Run an adequate number of Monte Carlo simulations to study properties of the model derivation methods below. For each simulation measure the predictive discrimination of the full model fit using c, Brier score, and generalized $R^2$, and the accuracy of the model by estimating the mean absolute difference and mean squared difference between predicted logit and population (true) logit. Decide upon a model approximation strategy that yields approximate models that are 0.95 as good as the full model. But start with a strategy that checks that the method works by only removing the single least important predictor. If this method suffers on root mean squared error, something is amiss. Run the approximation procedures for each of the Monte Carlo simulations. Compute the same statistical measures for the approximate model as you ran for the full model.

4. Summarize the various indexes over all simulations.

5. Summarize the volatility of the model approximation step by depicting the variation in the entire list of variables selected by the approximation.

6. Use the R brms package to do a Bayesian binary logistic regression analysis for the age-sex-response example at the front of the binary logistic regression chapter. Use some sort of diffuse prior for the intercept, and slightly more constrained Gaussian priors for the age and sex parameters. Use slack to discuss this problem, allowing each group to help each other as well as the instructor and teaching assistant to help as you go.