Week 11 - Exercises-Solutions

Exercise solutions

Week 12

Investigation

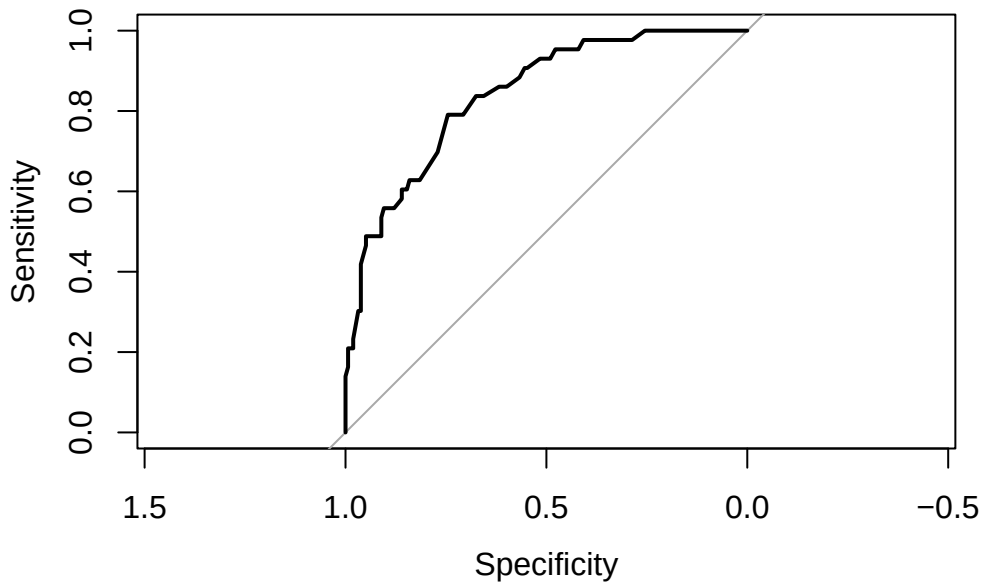
R code and output

1) Fitting the model, naive AUC and ROC

Clearly all covariates (*age*, *oral* and *smoke*) are important predictors of *mi*. A naive ROC curve can be obtained by computing the predicted probablity of MI for each observation in the dataset and calling the *roc* function from *pROC*

```
infarct <- read.csv("infarct.csv")
infarct<-data.frame(infarct)
require(pROC)
## Loading required package: pROC
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
out0 <- glm(mi ~ oral + age + smoke, family=binomial(link="logit"),data=infarct)
summary(out0)
##
## Call:
## glm(formula = mi ~ oral + age + smoke, family = binomial(link = "logit"),
##     data = infarct)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7019  -0.6045  -0.3215  -0.1562   2.5689
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -9.11405    1.75703  -5.187 2.13e-07 ***
## oral         1.97990    0.46968   4.215 2.49e-05 ***
## age          0.16256    0.04454   3.650 0.000262 ***
## smoke        1.81224    0.42938   4.221 2.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 208.20  on 199  degrees of freedom
## Residual deviance: 150.37  on 196  degrees of freedom
## AIC: 158.37
##
## Number of Fisher Scoring iterations: 5
infarct$pred=predict(out0,infarct,type="response")
out.roc<- roc(infarct$mi, infarct$pred,plot=TRUE,ci=TRUE)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
out.roc
##
## Call:
## roc.default(response = infarct$mi, predictor = infarct$pred,     ci = TRUE, plot = TRUE
##
## Data: infarct$pred in 157 controls (infarct$mi 0) < 43 cases (infarct$mi 1).
## Area under the curve: 0.8403
## 95% CI: 0.7771-0.9036 (DeLong)
```

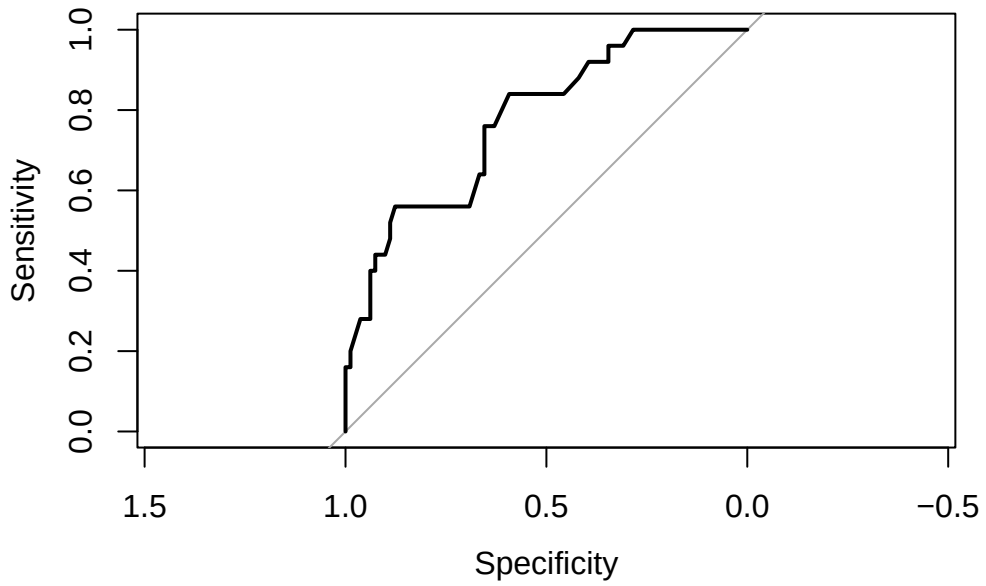
The naive AUC is 0.84, 95% CI=(0.78 ; 0.90), which can be classified as *good*. The model predicts well but this estimate is affected by optimism bias. The true AUC is probably a bit lower, how much lower, we are going to discover.

2) split sample AUC

An AUC free of optimism bias can be obtained using the split sample approach whereby the data is split in two, the model built on the training dataset and the AUC computed on the validation dataset

```
n=dim(infarct)[1]
set.seed(1002)
# set.seed(2002)
infarct$val=rbinom(n,1,0.5) # val =0 for dvelopment and val=1 for validation
table(infarct$val)
##
##   0   1
##  94 106

# NB: we choose here to split the data in two samples of similar sizes but
#     we could decide to put more patients in the development dataset
#     (increase the probability 0.5 to 0.6 etc)

# development dataset

infarct.dev=infarct[infarct$val==0,]
fit.dev<-glm(mi ~  oral + age + smoke, family=binomial, data=infarct.dev)
summary(fit.dev)
##
## Call:
## glm(formula = mi ~ oral + age + smoke, family = binomial, data = infarct.dev)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.8572  -0.4278  -0.2356  -0.1007   2.4841
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -12.00945    3.12672  -3.841 0.000123 ***
## oral          1.84803    0.77714   2.378 0.017407 *
## age           0.22427    0.07734   2.900 0.003734 **
## smoke         2.71876    0.76007   3.577 0.000348 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 91.815  on 93  degrees of freedom
## Residual deviance: 55.118  on 90  degrees of freedom
## AIC: 63.118
##
## Number of Fisher Scoring iterations: 6

# validation dataset + ROC/AUC

infarct.val=infarct[infarct$val==1,]
infarct.val$pred<- predict(fit.dev, infarct.val,type="response")
# predictions on the validation dataset using the previous fit
head(infarct.val)
##    id oral age smoke mi        pred val
## 2   2    0  32     0  0 0.007901981   1
## 3   3    1  37     0  1 0.134317344   1
## 6   6    1  40     0  0 0.233170631   1
## 7   7    0  35     0  0 0.015369383   1
## 8   8    1  33     0  0 0.059501663   1
## 10 10    0  31     0  0 0.006324488   1
require(pROC)
# ROC + AUC on the validation dataset (suffix .val)
out<- roc(infarct.val$mi, infarct.val$pred,plot=TRUE,ci=TRUE)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
out
##
## Call:
## roc.default(response = infarct.val$mi, predictor = infarct.val$pred,     ci = TRUE, plo
##
## Data: infarct.val$pred in 81 controls (infarct.val$mi 0) < 25 cases (infarct.val$mi 1).
## Area under the curve: 0.7751
## 95% CI: 0.6725-0.8776 (DeLong)
```

The AUC is lower with a larger CI, i.e. AUC=0.775 with a wider 95% CI=(0.67 ; 0.88). You may get a different result if you used another the seed or did not specify it. Setting the seed allows you to reproduce your results. The naive AUC=84% was a bit optimistic. Note that this method is suboptimal and it shows here with a small sample, yielding a large 95% CI

3) cross-validated AUC

A better for to use the data is to perform cross-validation. Given the data is rather, you may run into trouble with 10-fold crossvalidation depending on the seed. In that case, you can either change the seed or even use a smaller number of groups (e.g. carry out 5-fold CV). I used seed=2002, which seems to be ok.

```r
require(cvAUC)
## Loading required package: cvAUC
# reload the data - we don't want the indicator we used before
infarct <- read.csv("infarct.csv")
infarct<-data.frame(infarct)
infarct=data.frame(infarct)

# reformatting the dataset
colnames(infarct)=c("id", "oral","age", "smoke","Y")
# remove id (only the variables of interest MUST BE KEPT)
#            ========================================
infarct=infarct[,-c(1)]
# only outcome (Y) and covariates in dataset
head(infarct)
##   oral age smoke Y
## 1    1  33     1 0
## 2    0  32     0 0
## 3    1  37     0 1
## 4    0  36     0 0
## 5    1  50     1 1
## 6    1  40     0 0

# function doing the CV
cv_eval <- function(data, V=10){
  f.cvFolds <- function(Y, V){ #Create CV folds (stratify by outcome)
    Y0 <- split(sample(which(Y==0)), rep(1:V, length=length(which(Y==0))))
    Y1 <- split(sample(which(Y==1)), rep(1:V, length=length(which(Y==1))))
    folds <- vector("list", length=V)
    for (v in seq(V)) {folds[[v]] <- c(Y0[[v]], Y1[[v]])}
    return(folds)
  }
  f.doFit <- function(v, folds, data){ #Train/test glm for each fold
    fit <- glm(Y~., data=data[-folds[[v]],], family=binomial)
    pred <- predict(fit, newdata=data[folds[[v]],], type="response")
    return(pred)
  }
  folds <- f.cvFolds(Y=data$Y, V=V) #Create folds
  predictions <- unlist(sapply(seq(V), f.doFit, folds=folds, data=data)) #CV train/predict
  predictions[unlist(folds)] <- predictions #Re-order pred values
  ci.pooled.cvAUC
  # Get CV AUC and confidence interval
```

```
  out <- ci.cvAUC(predictions=predictions, labels=data$Y, folds=folds, confidence=0.95)
  return(out)
}

set.seed(2002)
out.cv <- cv_eval(data=infarct, V=10)
out.cv
## $cvAUC
## [1] 0.8276562
##
## $se
## [1] 0.03388826
##
## $ci
## [1] 0.7612365 0.8940760
##
## $confidence
## [1] 0.95
```

The 10-fold cross-validated AUC is 0.827, 95%CI=(0.76 ; 0.89) with seed=2002. It's a bit lower than the naive AUC but not by much, This is indicative of a small optimism bias (a common finding in practice). A slightly larger 95% CI than the one found with the naive analysis is observed. You may get different results with the seed(s) of your choice and it's perfectly fine. You could also repeat the process a few times and average all the AUC's.

Stata code and output

Once again the figures will appear when you run the code but are not displayed by Markdown.

1) Fitting the model, naive AUC and ROC

Clearly all covariates (*age*, *oral* and *smoke*) are important predictors of *mi*. A naive ROC curve can be obtained by computing the predicted probablity of MI for each observation in the dataset and calling the function *roctab* and using the option *graph*.

```
use infarct.dta
logistic mi oral age smoke, coef
predict fitted, pr
roctab mi fitted
roctab mi fitted,  graph title("Naive ROC")
## . use infarct.. logistic mi oral age smoke, coef
##
```

```
## Logistic regression                                      Number of obs =    200
##                                                          LR chi2(3)    =  57.83
##                                                          Prob > chi2   = 0.0000
## Log likelihood = -75.187408                              Pseudo R2     = 0.2777
##
## ------------------------------------------------------------------------------
##           mi | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
## -------------+----------------------------------------------------------------
##         oral |   1.979896   .4696992     4.22   0.000     1.059302    2.900489
##          age |   .1625556   .0445372     3.65   0.000     .0752643    .2498469
##        smoke |   1.812237   .4293955     4.22   0.000     .9706372    2.653836
##        _cons |  -9.114046   1.757112    -5.19   0.000    -12.55792   -5.670169
## ------------------------------------------------------------------------------
##
## . predict fitted, pr
##
## . roctab mi fitted
##
##                        ROC                    Asymptotic normal
##            Obs       area     Std. err.      [95% conf. interval]
##      ----------------------------------------------------------
##            200     0.8403       0.0323        0.77708     0.90356
##
## . roctab mi fitted,  graph title("Naive ROC")
```

The naive AUC is 0.84, 95% CI=(0.78 ; 0.90), which can be classified as *good*. The model predicts well but this estimate is affected by optimism bias. The true AUC is probably a bit lower, how much lower, we are going to discover.

2) split sample AUC

An AUC free of optimism bias can be obtained using the split sample approach whereby the data is split in two, the model built on the training dataset and the AUC computed on the validation dataset

```
use infarct.dta
set seed 1001
gen val = runiform()<.5
** Derive a prediction model y-chd69 in the development cohort
logistic mi oral age smoke  if val==0
** Generate a new variable containing the predicted probabilities
predict fitted, pr
```

```
** AUC on the development data (training)
** roctab mi fitted if  val==0
** roctab mi fitted if  val==0, graph name(graph0) title("training")

** AUC on the validation data  - THE ONE WE NEED
roctab mi fitted if  val==1
roctab mi fitted if  val==1,  graph name(graph1) title("validation")

## . use infarct.. set seed 1001
##
## . gen val = runiform()<.5
##
## . ** Derive a prediction model y-chd69 in the development cohort
## . logistic mi oral age smoke  if val==0
##
## Logistic regression                                     Number of obs =    109
##                                                         LR chi2(3)    =  32.19
##                                                         Prob > chi2   = 0.0000
## Log likelihood = -46.00784                              Pseudo R2     = 0.2592
##
## ------------------------------------------------------------------------------
##           mi | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
## -------------+----------------------------------------------------------------
##         oral |   11.62421   7.317309     3.90   0.000     3.384865    39.91952
##          age |   1.089913   .0620405     1.51   0.130     .9748538    1.218553
##        smoke |   4.494196   2.405762     2.81   0.005     1.573983    12.83228
##        _cons |   .0015526   .0033645    -2.98   0.003     .0000222    .1085545
## ------------------------------------------------------------------------------
## Note: _cons estimates baseline odds.
##
## . ** Generate a new variable containing the predicted probabilities
## . predict fitted, pr
##
## .
## . ** AUC on the development data (training)
## . ** roctab mi fitted if  val==0
## . ** roctab mi fitted if  val==0, graph name(graph0) title("training")
## .
## . ** AUC on the validation data  - THE ONE WE NEED
## . roctab mi fitted if  val==1
##
```

```
##                         ROC                    Asymptotic normal
##            Obs       area     Std. err.      [95% conf. interval]
##      -------------------------------------------------------------
##             91     0.8101       0.0554        0.70147     0.91871
##
## . roctab mi fitted if  val==1,  graph name(graph1) title("validation")
##
## .
```

The AUC is lower with a larger CI, i.e. AUC=0.81 with a wider 95% CI=(0.70 ; 0.92). You may get a different result if you used another the seed or did not specify it. Setting the seed allows you to reproduce your results. The naive AUC=84% was indeed optimistic. Note that this method is suboptimal and it shows here with a small sample, yielding a larger 95% CI,

3) cross-validated AUC

A better for to use the data is to perform cross-validation. Given the data is rather, you may run into trouble with 10-fold crossvalidation depending on the seed. In that case, you can either change the seed or even use a smaller number of groups (e.g. carry out 5-fold CV). I used seed=2002, which seems to be ok.

```
use infarct.dta
set seed 2002
xtile group = uniform(), nq(10)
quietly gen cvfitted = .
forvalues i = 1/10 {

    * Step 2: estimate model omitting each subset
    quietly xi: logistic mi oral age smoke if group !=`i'
    quietly predict cvfittedi, pr

    * Step 3: save cross-validated statistic for each omitted subset
    quietly replace cvfitted = cvfittedi if group==`i'
    quietly drop cvfittedi
    }

* Step 4: calculate cross-validated area under ROC curve
roctab mi cvfitted
roctab mi cvfitted,  graph title("10-fold CV ROC")
## . use infarct.. set seed 2002
##
## . xtile group = uniform(), nq(10)
##
```

```
## . quietly gen cvfitted = .
##
## . forvalues i = 1/10 {
##   2.
## .         * Step 2: estimate model omitting each subset
## .         quietly xi: logistic mi oral age smoke if group !=`i'
##   3.          quietly predict cvfittedi, pr
##   4.
## .         * Step 3: save cross-validated statistic for each omitted subset
## .         quietly replace cvfitted = cvfittedi if group==`i'
##   5.          quietly drop cvfittedi
##   6.          }
##
## .
## . * Step 4: calculate cross-validated area under ROC curve
## . roctab mi cvfitted
##
##                       ROC                    Asymptotic normal
##            Obs       area     Std. err.      [95% conf. interval]
##      ------------------------------------------------------------
##            200     0.8110       0.0357        0.74095     0.88103
##
## . roctab mi cvfitted,  graph title("10-fold CV ROC")
```

The 10-fold cross-validated AUC is 0.81, 95%CI=(0.74 ; 0.88) with seed=2002. It's a bit lower than the naive AUC but not by much, This is indicative of a small optimism bias (a common finding in practice). A slightly larger 95% CI than the one found with the naive analysis is observed. This is however much better than the one obtained in 2). You may get different results with the seed(s) of your choice and it's perfectly fine. You could also repeat the process a few times and average all the AUC's.

Practice - calibration plot

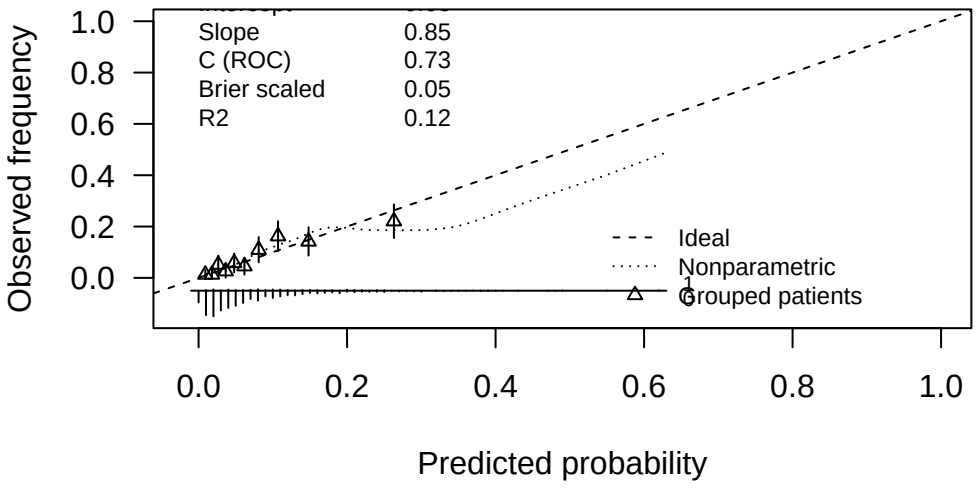
R code and output

1) Calibration plot for the final model (Table 5.18) without outlier in cholesterol. Use the split sample approach.

```
require(rms)
## Loading required package: rms
## Loading required package: Hmisc
##
```

```
## Attaching package: 'Hmisc'
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
## The following objects are masked from 'package:base':
##
##     format.pval, units
wcgs <- read.csv("wcgs.csv")
wcgs<-data.frame(wcgs)
wcgs1=cbind(wcgs$age,wcgs$chol,wcgs$sbp,wcgs$bmi,wcgs$smoke,wcgs$dibpat,wcgs$chd69)
colnames(wcgs1)=c("age", "chol", "sbp", "bmi", "smoke","dibpat","chd69")
wcgs1=data.frame(wcgs1)
wcgs1=na.omit(wcgs1)
# remove outlier chol=645
wcgs1=wcgs1[wcgs1$chol <645,]
wcgs1$age_10<-(wcgs1$age-mean(wcgs1$age))/10
wcgs1$bmi_10<-(wcgs1$bmi-mean(wcgs1$bmi))/10
wcgs1$sbp_50<-(wcgs1$sbp-mean(wcgs1$sbp))/50
wcgs1$chol_50<-(wcgs1$chol-mean(wcgs1$chol,na.rm=T))/50

wcgs1$bmichol<-wcgs1$bmi_10*wcgs1$chol_50
wcgs1$bmisbp<-wcgs1$bmi_10*wcgs1$sbp_50


dim(wcgs1)
## [1] 3141   13

# out<-glm(chd69 ~ age_10 + chol_50 + sbp_50 +  bmi_10 + smoke + dibpat + bmichol + bmisbp
# summary(out)

n=dim(wcgs1)[1]
set.seed(1001) # choose the same seed as in the notes
wcgs1$val=rbinom(n,1,0.5) # val =0 for training and val=1 for validation
table(wcgs1$val)
##
##    0    1
## 1596 1545

# training dataset
wcgs1.dev=wcgs1[wcgs1$val==0,]
fit.dev<-glm(chd69 ~ age_10 + chol_50 + sbp_50 +  bmi_10 + smoke + dibpat + bmichol + bmis
```

```r
summary(fit.dev)
##
## Call:
## glm(formula = chd69 ~ age_10 + chol_50 + sbp_50 + bmi_10 + smoke +
##     dibpat + bmichol + bmisbp, family = binomial, data = wcgs1.dev)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2859  -0.4311  -0.2977  -0.1993   2.8899
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.4771     0.2179 -15.960  < 2e-16 ***
## age_10        0.5830     0.1719   3.391 0.000697 ***
## chol_50       0.7121     0.1127   6.317 2.67e-10 ***
## sbp_50        1.0343     0.2924   3.537 0.000405 ***
## bmi_10        0.9759     0.4221   2.312 0.020771 *
## smoke         0.5865     0.2054   2.856 0.004294 **
## dibpat        0.6821     0.2066   3.302 0.000959 ***
## bmichol      -0.9965     0.3760  -2.650 0.008038 **
## bmisbp       -0.9603     0.8799  -1.091 0.275105
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 881.60  on 1595  degrees of freedom
## Residual deviance: 769.12  on 1587  degrees of freedom
## AIC: 787.12
##
## Number of Fisher Scoring iterations: 6

# validation dataset
wcgs1.val=wcgs1[wcgs1$val==1,]
wcgs1.val$pred<- predict(fit.dev, wcgs1.val,type="response")

source("val.prob.ci.R")

val.prob.ci(wcgs1.val$pred,wcgs1.val$chd69, pl=T,smooth=T,logistic.cal=F,
            g=10)
```




```
##          Dxy      C (ROC)           R2            D     D:Chi-sq          D:p
## 4.688013e-01 7.344007e-01 1.199755e-01 5.342483e-02 8.354136e+01 0.000000e+00
##            U     U:Chi-sq          U:p            Q        Brier    Intercept
## 4.244369e-04 2.655755e+00 2.650392e-01 5.300039e-02 7.305119e-02 6.225161e-02
##        Slope         Emax Brier scaled         Eavg
## 8.487498e-01 8.825670e-02 5.205208e-02 1.498093e-02

#val.prob.ci(wcgs1.val$pred,wcgs1.val$chd69, pl=T,smooth=T,logistic.cal=F,
#            g=20)
```

2) Has the calibration plot improved compared with the simpler model?

The calibration plot still displays overestimation of CHD risk for the highest risk category. The plot may look different depending on the seed you used. This is somehow artificial due to the fact that we don't have new data to play with - see also 3).

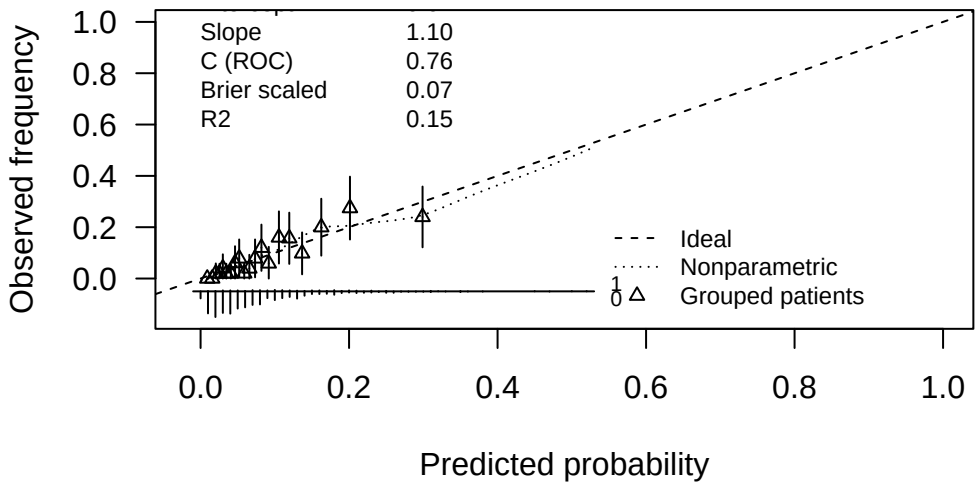
3) Repeat with 20 groups and 2/3 in the training dataset

Here we simply need to recreat *val* to have 2/3 of observations in the dataset and use *g=20* when running *val.prob.ci*

```
n=dim(wcgs1)[1]
set.seed(1001) # choose the same seed as in the notes
```

```r
wcgs1$val=rbinom(n,1,1/3) # val =0 for dvelopment and val=1 for validation
table(wcgs1$val)
##
##    0    1
## 2129 1012

# development dataset
wcgs1.dev=wcgs1[wcgs1$val==0,]
fit.dev<-glm(chd69 ~ age_10 + chol_50 + sbp_50 +  bmi_10 + smoke + dibpat + bmichol + bmis
summary(fit.dev)
##
## Call:
## glm(formula = chd69 ~ age_10 + chol_50 + sbp_50 + bmi_10 + smoke +
##     dibpat + bmichol + bmisbp, family = binomial, data = wcgs1.dev)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1568  -0.4431  -0.3145  -0.2196   2.9405
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.29479    0.17816 -18.494  < 2e-16 ***
## age_10       0.56804    0.14649   3.878 0.000105 ***
## chol_50      0.61069    0.09528   6.409 1.46e-10 ***
## sbp_50       1.00208    0.24942   4.018 5.88e-05 ***
## bmi_10       0.93776    0.35769   2.622 0.008749 **
## smoke        0.43660    0.17136   2.548 0.010838 *
## dibpat       0.66386    0.17534   3.786 0.000153 ***
## bmichol     -0.85931    0.33072  -2.598 0.009369 **
## bmisbp      -1.18985    0.75416  -1.578 0.114632
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1185.4  on 2128  degrees of freedom
## Residual deviance: 1059.2  on 2120  degrees of freedom
## AIC: 1077.2
##
## Number of Fisher Scoring iterations: 6
```

```
# validation dataset + ROC/AUC
wcgs1.val=wcgs1[wcgs1$val==1,]
wcgs1.val$pred<- predict(fit.dev, wcgs1.val,type="response")

source("val.prob.ci.R")

val.prob.ci(wcgs1.val$pred,wcgs1.val$chd69, pl=T,smooth=T,logistic.cal=F,
            g=20)
```




```
##           Dxy       C (ROC)            R2             D      D:Chi-sq
##  5.200286e-01  7.600143e-01  1.517699e-01  6.828102e-02  7.010040e+01
##           D:p             U      U:Chi-sq          U:p             Q
##  1.110223e-16 -1.481565e-03  5.006565e-01  7.785452e-01  6.976259e-02
##         Brier     Intercept         Slope          Emax  Brier scaled
##  7.201299e-02  2.208180e-02  1.096972e+00  6.160309e-02  7.389028e-02
##          Eavg
##  1.043200e-02
```

We see more points and more variability. Still some overestimation of the CHD risk for the highest category (although now the wider 95% CI now covers the corresponding point on the 45 degree line0. We have not really fixed the problem with this more complex model. The plot

depends on the seed you chose here. In reality this uncertainty does not exist when carrying out external validation (which is normally the way to go). We use the the data at hand to build the model and the new external data to draw the calibration plot. No need of a seed at any stage. You can also notice that the plot gives the AUC (on the validation sample) albeit without the 95% CI.

Stata code and output

Once again the figures will appear when you run the code but are not displayed by Markdown.

1) Calibration plot for the final model (Table 5.18) without outlier in cholesterol. Use the split sample approach.

```
use wcgs.dta
summarize age
gen age10=(age-r(mean))/10
summarize bmi
gen bmi10=(bmi-r(mean))/10
summarize sbp
gen sbp50=(sbp-r(mean))/50
summarize chol
gen chol50=(chol-r(mean))/50
gen bmichol=bmi10*chol50
gen bmisbp=bmi10*sbp50
drop ih chol > 645

set seed 1001
# set.seed 1030
gen val = runiform()<.5
logistic chd69 age10 chol50 sbp50 bmi10 smoke dibpat bmichol  bmisbp if val==0
predict proba, pr
pmcalplot proba chd69 if val==1, ci
** alternatively
drop if val==0
pmcalplot proba chd69, ci
## . use wcgs.. summarize age
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          age |      3,154    46.27869    5.524045         39         59
##
## . gen age10=(age-r(mean))/10
```

```
##
## . summarize bmi
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          bmi |      3,154    24.51837    2.567496   11.19061   38.94737
##
## . gen bmi10=(bmi-r(mean))/10
##
## . summarize sbp
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          sbp |      3,154    128.6328    15.11773         98        230
##
## . gen sbp50=(sbp-r(mean))/50
##
## . summarize chol
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##         chol |      3,142    226.3724    43.42043        103        645
##
## . gen chol50=(chol-r(mean))/50
## (12 missing values generated)
##
## . gen bmichol=bmi10*chol50
## (12 missing values generated)
##
## . gen bmisbp=bmi10*sbp50
##
## . drop ih chol > 645
## variable ih not found
## r(111);
##
## end of do-file
## r(111);
```

2) Has the calibration plot improved compared with the simpler model?

The calibration plot seems a bit better with this seed but it's also sensitive to the choice of seed. Try seed=1030, you will have a different looking plot, where the situation is the same

as before. This is somehow articial due to the fact that we don't have new data to play with - see also 3).

3) Repeat with 20 groups and 2/3 in the training dataset

```
use wcgs.dta
summarize age
gen age10=(age-r(mean))/10
summarize bmi
gen bmi10=(bmi-r(mean))/10
summarize sbp
gen sbp50=(sbp-r(mean))/50
summarize chol
gen chol50=(chol-r(mean))/50
gen bmichol=bmi10*chol50
gen bmisbp=bmi10*sbp50
drop if chol > 645

** set seed 1001
 set seed 2002
** set seed 1030
gen val = runiform()<.33
logistic chd69 age10 chol50 sbp50 bmi10 smoke dibpat bmichol  bmisbp if val==0
predict proba, pr
pmcalplot proba chd69 if val==1, ci bin(20)
** alternatively
drop if val==0
pmcalplot proba chd69, ci bin(20)
## . use wcgs.. summarize age
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          age |      3,154    46.27869    5.524045         39         59
##
## . gen age10=(age-r(mean))/10
##
## . summarize bmi
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          bmi |      3,154    24.51837    2.567496   11.19061   38.94737
##
```

```
## . gen bmi10=(bmi-r(mean))/10
##
## . summarize sbp
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##          sbp |      3,154    128.6328    15.11773         98        230
##
## . gen sbp50=(sbp-r(mean))/50
##
## . summarize chol
##
##     Variable |        Obs        Mean    Std. dev.       Min        Max
## -------------+---------------------------------------------------------
##         chol |      3,142    226.3724    43.42043        103        645
##
## . gen chol50=(chol-r(mean))/50
## (12 missing values generated)
##
## . gen bmichol=bmi10*chol50
## (12 missing values generated)
##
## . gen bmisbp=bmi10*sbp50
##
## . drop if chol > 645
## (12 observations deleted)
##
## .
## . ** set seed 1001
## .  set seed 2002
##
## . ** set seed 1030
## . gen val = runiform()<.33
##
## . logistic chd69 age10 chol50 sbp50 bmi10 smoke dibpat bmichol  bmisbp if val==0
##
## Logistic regression                                     Number of obs =  2,094
##                                                         LR chi2(8)    = 132.43
##                                                         Prob > chi2   = 0.0000
## Log likelihood = -549.87382                             Pseudo R2     = 0.1075
##
```

```
## ------------------------------------------------------------------------------
##        chd69 | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
## -------------+----------------------------------------------------------------
##        age10 |   1.808118   .2602319     4.12   0.000     1.363698    2.397371
##       chol50 |   1.757084    .160149     6.18   0.000     1.469637    2.100754
##        sbp50 |   2.743119   .6621937     4.18   0.000     1.709083    4.402772
##        bmi10 |   2.928239   1.038346     3.03   0.002     1.461413    5.867327
##        smoke |   1.944388   .3258757     3.97   0.000     1.399985    2.700491
##       dibpat |   1.821471   .3122496     3.50   0.000     1.301677    2.548833
##      bmichol |   .4433118   .1251311    -2.88   0.004     .2549444    .7708557
##       bmisbp |   .2280157   .1648987    -2.04   0.041     .0552566    .9409036
##        _cons |   .0363761   .0064614   -18.66   0.000     .0256814    .0515245
## ------------------------------------------------------------------------------
## Note: _cons estimates baseline odds.
##
## . predict proba, pr
##
## . pmcalplot proba chd69 if val==1, ci bin(20)
## command pmcalplot is unrecognized
## r(199);
##
## end of do-file
## r(199);
```

We see more points and more variability. Still some overestimation of the CHD risk for the highest category (although now the wider 95% CI now covers the corresponding point on the 45 degree line). The plot depends on the seed you chose, see plot with seed 1001 or 1030 (pretty good). However, in reality, this uncertainty does not exist since typically external calibration is carried out. We use the the data at hand to build the model and the new external data to draw the calibration plot. No need of a seed at any stage.