Assignment 4 Heart

Ben Lu

2023-10-06

Question 1

(a)

```
Insecticide.df <- read.csv("insecticide.txt", sep = " ")
Insecticide.df$pip <- as.factor(Insecticide.df$pip)
```

(b)

```
Insecticide.fit1 <- glm(cbind(y, n - y) ~ pip + pyr, family = "binomial", data = Insecticide.df)

Insecticide.fit2 <- glm(cbind(y, n - y) ~ pip * pyr, family = "binomial", data = Insecticide.df)

summary(Insecticide.fit1)
```

```
##
## Call:
## glm(formula = cbind(y, n - y) ~ pip + pyr, family = "binomial",
##     data = Insecticide.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2033  -1.2047   0.3311   1.8968   2.9032
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -7.0768     0.3805 -18.597  < 2e-16 ***
## pip0.25       1.3796     0.1904   7.245 4.33e-13 ***
## pip2.5        3.8921     0.2411  16.143  < 2e-16 ***
## pip10         4.8150     0.2679  17.975  < 2e-16 ***
## pyr           6.8753     0.3541  19.416  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1038.053  on 12  degrees of freedom
```

```
## Residual deviance:   50.818  on  8  degrees of freedom
## AIC: 120.91
##
## Number of Fisher Scoring iterations: 4
```

```
summary(Insecticide.fit2 )
```

```
##
## Call:
## glm(formula = cbind(y, n - y) ~ pip * pyr, family = "binomial",
##     data = Insecticide.df)
##
## Deviance Residuals:
##       1        2        3        4        5        6        7        8
##  0.4459  -0.6734   0.5557  -0.1789   0.3026  -0.2582  -0.7639   0.7167
##       9       10       11       12       13
## -0.4244  -0.8848   1.0511  -0.5348  -0.1523
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.1444     0.4379 -11.747  < 2e-16 ***
## pip0.25      -0.4016     0.6650  -0.604   0.5460
## pip2.5       -0.2474     0.7276  -0.340   0.7338
## pip10         1.3458     0.6584   2.044   0.0409 *
## pyr           4.9700     0.4147  11.983  < 2e-16 ***
## pip0.25:pyr   1.7174     0.7327   2.344   0.0191 *
## pip2.5:pyr    6.1521     1.1893   5.173 2.30e-07 ***
## pip10:pyr     5.5972     1.2448   4.496 6.91e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1038.0533  on 12  degrees of freedom
## Residual deviance:    4.6256  on  5  degrees of freedom
## AIC: 80.72
##
## Number of Fisher Scoring iterations: 4
```

```
anova(Insecticide.fit1, Insecticide.fit2, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(y, n - y) ~ pip + pyr
## Model 2: cbind(y, n - y) ~ pip * pyr
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1         8     50.818
## 2         5      4.626  3   46.192 5.162e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As we obtained a very small p-value of 5.162e-10 in our ANOVA, we have extremely strong evidence against the null hypothesis that the additive effects of pip and pyr are appropriate. This implies that the interaction effects of pip and pyr were statistically significant, and hence should be kept for our analysis.

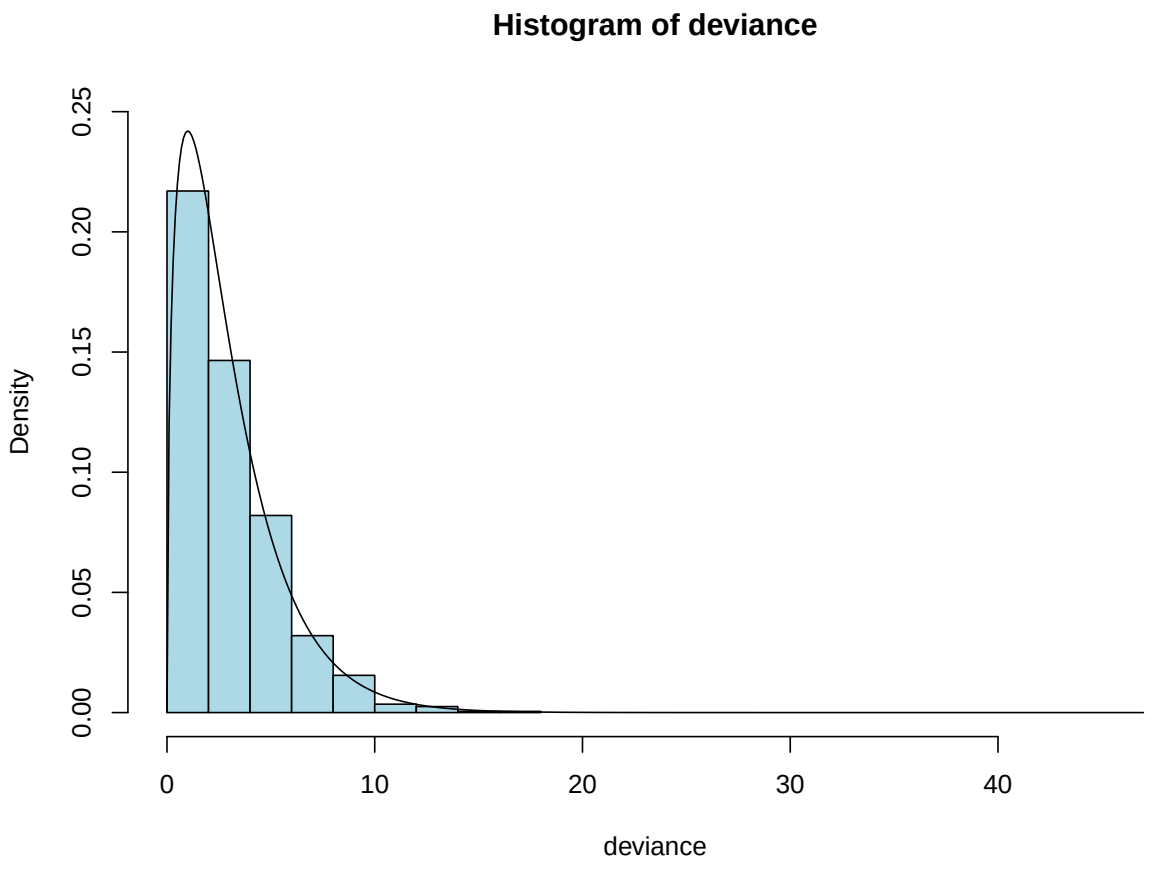
(c)

```
set.seed(123) #for reproducibility of results
num.obs = nrow(Insecticide.df) #number of observations
num.sim = 1e3 #number of simulations
n = Insecticide.df$n #vector for sample size

deviance = numeric(1000)

#calculate simulated values
for (i in 1:num.sim) {
  ysim = rbinom(num.obs, size = n, prob = fitted.values(Insecticide.fit1))
  Insecticide.sim1 = glm(cbind(ysim, n -ysim) ~ pip + pyr, family = "binomial", data = Insecticide.df)
  Insecticide.sim2 = glm(cbind(ysim, n -ysim) ~ pip * pyr, family = "binomial", data = Insecticide.df)
  diff = deviance(Insecticide.sim1) - deviance(Insecticide.sim2)
  deviance[i] = diff
}

#histogram plot of deviance
hist(deviance, freq = FALSE, col = "lightblue", xlim = c(0, 47), ylim = c(0, 0.25))
#superimpose chi-squared distribution with df = 3
lines(x = seq(0, 47, 0.1), y = dchisq(seq(0, 47, 0.1), df = 3))
```



The null hypothesis is the assumption which the additive effects of pip and pyr were sufficient, and that there is no difference between our additive model and interactive model.

Under the assumption that the null hypothesis is true, there is only a 5.16e-10 probability of observing a difference in residual deviance between 50.8 in our additive model and 4.6 in our interactive model. The difference is very unlikely to occur in a theory based chi-squared distribution with 3 degrees of freedom.

We are examining the differences in our models' residual deviance using a chi-squared distribution with 3 degrees of freedom because the change in degrees of freedom from our additive model to the interactive model is 3.

In our sample of 1000 simulations, we reproduced both of the additive and interactive models based on the original additive model. This allows us to recreate both models under the assumption which they are the same, since they are based on the results of the additive model. Then, the residual deviance of both models was used to calculate the deviance. After which, we created a histogram plot of the deviance from the two models and superimposed a chi-squared distribution with 3 degrees of freedom.

It appears that the plotted values of our simulated deviance from our models follow closely to the theory based chi-squared distribution with 3 degrees of freedom, this indicates that both of the additive and interactive models are good fits using the binomial family.

Based on our plot, it is clear that a deviance of 46.192 is very unlikely to occur by chance on the assumption that both models are the same. This reinforces our earlier conclusion that there is very strong evidence against the null hypothesis which both models are equivalent, and that our interactive model was statistically significant. Thus, we prefer to keep our interaction model as it is better in modelling our data with a lower residual deviance.

(d)

```
#store coefficients
intercept = coef(Insecticide.fit2)[1]
pyr = coef(Insecticide.fit2)[5]
pip10 = coef(Insecticide.fit2)[4]
pip10.pyr = coef(Insecticide.fit2)[8]

# 0% piperonyl butoxide
pyr.pip0 = (log(0.8/0.2) - intercept)/pyr
# 10% piperonyl butoxide
pyr.pip10 = (log(0.8/0.2) - intercept - pip10)/(pyr + pip10.pyr)

pyr.pip0
```

```
## (Intercept)
##    1.314023
```

```
pyr.pip10
```

```
## (Intercept)
##   0.4906598
```

It appears that we need concentration of 1.31% Pyrethrin to have a 0.80 probability of mortality in flour beetles when no Piperonyl Butoxide is present.

In comparison, we need to a concentration of 0.49% Pyrethrin to have a 0.8 probability of mortality in flour bettles when a concentration of 10% Piperonyl Butoxide is present.

(e)

```
set.seed(123)
#vectors to store calculated values
estimates.0pip = numeric(num.sim)
estimates.10pip = numeric(num.sim)

#calculate simulated values
for (i in 1:num.sim) {
  ysim = rbinom(num.obs, size = n, prob = fitted.values(Insecticide.fit2))

  Insecticide.sim2 = glm(cbind(ysim, n -ysim) ~ pip * pyr, family = "binomial", data = Insecticide.df)

  #store coefficients
  intercept.sim = coef(Insecticide.sim2)[1]
  pyr.sim = coef(Insecticide.sim2)[5]
  pip10.sim = coef(Insecticide.sim2)[4]
  pip10.pyr.sim = coef(Insecticide.sim2)[8]

  estimates.0pip[i] = (log(0.8/0.2) - intercept.sim)/pyr.sim
  estimates.10pip[i] = (log(0.8/0.2) - intercept.sim - pip10.sim)/(pyr.sim + pip10.pyr.sim)
}

#Confidence interval for 0% piperonyl butoxide
quantile(estimates.0pip, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 1.244814 1.376516
```

```
#Confidence interval for 10% piperonyl butoxide
quantile(estimates.10pip, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.4613924 0.5202572
```

In the absence of Piperonyl Butoxide, it requires an average concentration of between 1.245% and 1.377% Pyrethrin to have a 0.80 probability of mortality in flour beetles.

In the presence of 10% Piperonyl Butoxide, it requires an average concentration of between 0.461% and 0.520% Pyrethrin to have a 0.80 probability of mortality in flour beetles.

(f)

```
#percentage change before and after 10% Piperonyl Butoxide, LCI
((0.461 - 1.245)/1.245) * 100
```

```
## [1] -62.97189
```

```r
#percentage change before and after 10% Piperonyl Butoxide, UCI
((0.520 - 1.377)/1.377 ) * 100
```

```
## [1] -62.23675
```

It appears that in the presence of 10% Piperonyl Butoxide, it requires a lower concentration of Pyrethrin to get the same probability of mortality in flour beetles.

On average, in the presence of 10% Piperonyl Butoxide, it requires between 0.784% (0.461 - 1.245) and 0.857% (0.520 - 1.377) less concentrations of Pyrethrin to get a 0.80 probability of mortality in flour beetles.

In other words, by adding 10% Piperonyl Butoxide, it has resulted in between 62% and 63% decrease in the required concentrations of Pyrethrin to get a 0.80 probability of mortality in flour beetles. Therefore, Piperonyl Butoxide was an effective synergist and greatly enhanced the effectiveness of Pyrethrin.

Question 2

(a)

```r
#read txt file and factor variables
Heart.df <- read.csv("HeartData.txt", sep = " ")

Heart.df$sex <- as.factor(Heart.df$sex)
Heart.df$cp <- as.factor(Heart.df$cp)
Heart.df$fbs <- as.factor(Heart.df$fbs)
Heart.df$restecg <- as.factor(Heart.df$restecg)
Heart.df$exang <- as.factor(Heart.df$exang)

#summary outputs
str(Heart.df)
```

```
## 'data.frame':    261 obs. of  11 variables:
##  $ age     : int  28 29 30 31 32 32 32 33 34 34 ...
##  $ sex     : Factor w/ 2 levels "0","1": 2 2 1 1 1 2 2 2 1 2 ...
##  $ cp      : Factor w/ 4 levels "1","2","3","4": 2 2 1 2 2 2 2 3 2 2 ...
##  $ trestbps: int  130 120 170 100 105 110 125 120 130 150 ...
##  $ chol    : int  132 243 237 219 198 225 254 298 161 214 ...
##  $ fbs     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ restecg : Factor w/ 3 levels "0","1","2": 3 1 2 2 1 1 1 1 1 2 ...
##  $ thalach : int  185 160 170 150 165 184 155 185 190 168 ...
##  $ exang   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ oldpeak : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num     : int  0 0 0 0 0 0 0 0 0 0 ...
```

```r
summary(Heart.df)
```

```
##       age        sex     cp         trestbps          chol       fbs
##  Min.   :28.00   0: 69   1: 10   Min.   : 92.0   Min.   : 85.0   0:242
##  1st Qu.:42.00   1:192   2: 92   1st Qu.:120.0   1st Qu.:208.0   1: 19
##  Median :49.00           3: 46   Median :130.0   Median :242.0
```

```
##  Mean   :47.77          4:113   Mean   :132.6   Mean   :248.8
##  3rd Qu.:54.00                  3rd Qu.:140.0   3rd Qu.:280.0
##  Max.   :65.00                  Max.   :200.0   Max.   :603.0
##  restecg    thalach      exang      oldpeak            num
##  0:208   Min.   : 82.0   0:178   Min.   :0.0000   Min.   :0.0000
##  1: 47   1st Qu.:122.0   1: 83   1st Qu.:0.0000   1st Qu.:0.0000
##  2:  6   Median :140.0           Median :0.0000   Median :0.0000
##          Mean   :139.2           Mean   :0.6123   Mean   :0.3755
##          3rd Qu.:155.0           3rd Qu.:1.0000   3rd Qu.:1.0000
##          Max.   :190.0           Max.   :5.0000   Max.   :1.0000
```

(b)

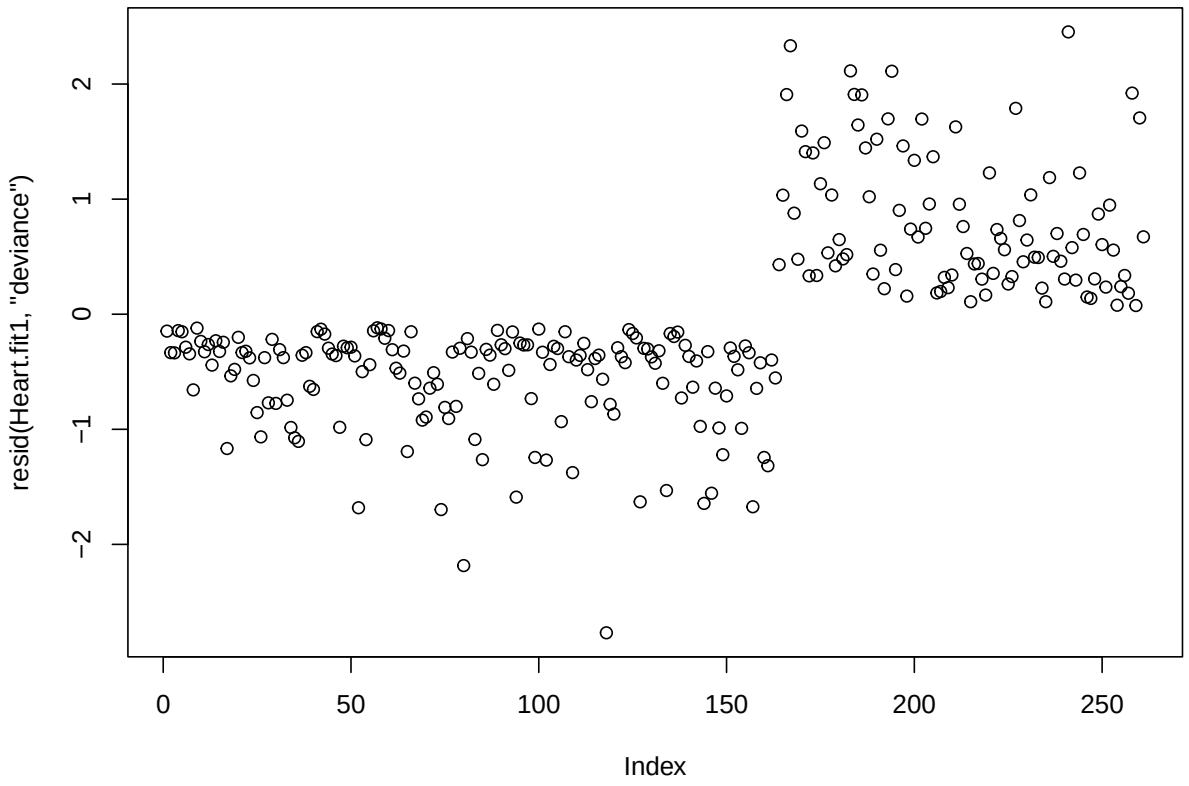
```
Heart.fit1 <- glm(num ~ age + sex + cp + trestbps + chol + fbs + restecg + thalach + exang + oldpeak, fa

summary(Heart.fit1)
```

```
##
## Call:
## glm(formula = num ~ age + sex + cp + trestbps + chol + fbs +
##     restecg + thalach + exang + oldpeak, family = "binomial",
##     data = Heart.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7682  -0.4879  -0.2655   0.4366   2.4531
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.669215   2.975271  -0.561  0.57478
## age         -0.009255   0.029526  -0.313  0.75394
## sex1         1.307881   0.481892   2.714  0.00665 **
## cp2         -1.919972   1.022129  -1.878  0.06033 .
## cp3         -0.515823   1.008135  -0.512  0.60889
## cp4          0.368482   0.967228   0.381  0.70323
## trestbps    -0.001603   0.011514  -0.139  0.88925
## chol         0.005218   0.002788   1.872  0.06127 .
## fbs1         1.639892   0.779925   2.103  0.03550 *
## restecg1    -0.391962   0.535431  -0.732  0.46414
## restecg2    -0.862764   1.715431  -0.503  0.61500
## thalach     -0.008658   0.010189  -0.850  0.39549
## exang1       0.943270   0.487140   1.936  0.05283 .
## oldpeak      1.184473   0.280460   4.223 2.41e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 345.46  on 260  degrees of freedom
## Residual deviance: 184.70  on 247  degrees of freedom
## AIC: 212.7
##
```

```
## Number of Fisher Scoring iterations: 6
```

```
#residual plot
plot(resid(Heart.fit1, "deviance"))
```




```
#chi-squared test
1 - pchisq(deviance(Heart.fit1), df.residual(Heart.fit1))
```

```
## [1] 0.998857
```

Our plot of deviance residuals indicates no concerns about our initial model as most of the values fall within the acceptable range of -2 to 2 which follows the normal distribution.

We obtained a very large p-value of 0.999 in our chi-squared test, this means if our model was correct there is a very high probability that a deviance of 184.7 comes from a chi-squared distribution with 247 degrees of freedom. Therefore, our initial model was an appropriate fit as it satisfies the goodness-of-fit test.

(c)

```
options(na.action = "na.fail")
Heart.fits <- dredge(Heart.fit1)
```

```
## Fixed term is "(Intercept)"
```

```
head(Heart.fits)
```

```
## Global model call: glm(formula = num ~ age + sex + cp + trestbps + chol + fbs +
##     restecg + thalach + exang + oldpeak, family = "binomial",
##     data = Heart.df)
## ---
## Model selection table
##     (Intrc)       age     chol cp exang fbs oldpk sex     thlch     trstb df
## 191  -3.629           0.005109  +     +   + 1.199   +                     9
## 189  -2.422                     +     +   + 1.182   +                     8
## 447  -2.790           0.005104  +     +   + 1.195   + -0.005812           10
## 183  -3.780           0.005256  +         + 1.502   +                     8
## 703  -3.445           0.005155  +     +   + 1.201   +           -0.001534 10
## 192  -3.469 -0.003379 0.005091  +     +   + 1.200   +                     10
##      logLik  AICc delta weight
## 191 -93.081 204.9  0.00  0.341
## 189 -94.905 206.4  1.50  0.161
## 447 -92.871 206.6  1.74  0.143
## 183 -95.189 206.9  2.07  0.121
## 703 -93.072 207.0  2.14  0.117
## 192 -93.073 207.0  2.15  0.117
## Models ranked by AICc(x)
```

(d)

```
# pred function for 1st model
predfun1.glm <- function(train.x, train.y, test.x, test.y) {
  glm.fit <- glm(train.y ~ chol +cp + exang + fbs + oldpeak + sex , data = train.x)
  ynew <- predict(glm.fit, test.x, type = "response")
  roc = roc(test.y, ynew)$auc
}

# pred function for 2nd model
predfun2.glm <- function(train.x, train.y, test.x, test.y) {
  glm.fit <- glm(train.y ~  cp + exang + fbs + oldpeak + sex , data = train.x)
  ynew <- predict(glm.fit, test.x, type = "response")
  roc = roc(test.y, ynew)$auc
}

# pred function for 3rd model
predfun3.glm <- function(train.x, train.y, test.x, test.y) {
  glm.fit <- glm(train.y ~  chol + cp + exang + fbs + oldpeak + sex + thalach, data = train.x)
  ynew <- predict(glm.fit, test.x, type = "response")
  roc = roc(test.y, ynew)$auc
}

# find AUC values [suppressMessages is used to hide messages]
cv1 = suppressMessages(crossval(predfun1.glm, X = Heart.df[, 1:10], Y = Heart.df[, 11], K = 10, B = 100
```

```
cv2 = suppressMessages(crossval(predfun2.glm, X = Heart.df[, 1:10], Y = Heart.df[, 11], K = 10, B = 100

cv3 = suppressMessages(crossval(predfun3.glm, X = Heart.df[, 1:10], Y = Heart.df[, 11], K = 10, B = 100

#AUC with standard errors
c(cv1$stat, cv1$stat.se)
```

```
## [1] 0.899677617 0.001832389
```

```
c(cv2$stat, cv2$stat.se)
```

```
## [1] 0.89663527 0.00182911
```

```
c(cv3$stat, cv3$stat.se)
```

```
## [1] 0.897472175 0.001896526
```

```
#AUC range
c(cv1$stat - cv1$stat.se, cv1$stat + cv1$stat.se)
```

```
## [1] 0.8978452 0.9015100
```

```
c(cv2$stat - cv2$stat.se, cv2$stat + cv2$stat.se)
```

```
## [1] 0.8948062 0.8984644
```

```
c(cv3$stat - cv3$stat.se, cv3$stat + cv3$stat.se)
```

```
## [1] 0.8955756 0.8993687
```

We used our top three models to train on 900 rows of data and predicted results for 100 estimates, and retrieved their respective AUC values. We prefer model 191 (first model) as it has the highest AUC value of 0.899, which indicates this model's predictive ability are slightly better than the other two models.

However, after taking into consideration of the models' AUC values and its standard errors, it appears that the lower and upper bounds of each models' AUC values overlap. This suggests that despite model 191 has best predictive abilities, we are not too confident in this claim.

(e)

```
#fit for model 191 and use roc function to find sensitivity, specificity and threshold
best.fit <- glm(num ~  chol + cp + exang + fbs + oldpeak + sex , data = Heart.df)
heart.roc <- roc(response = Heart.df$num, predictor = fitted.values(best.fit))
```

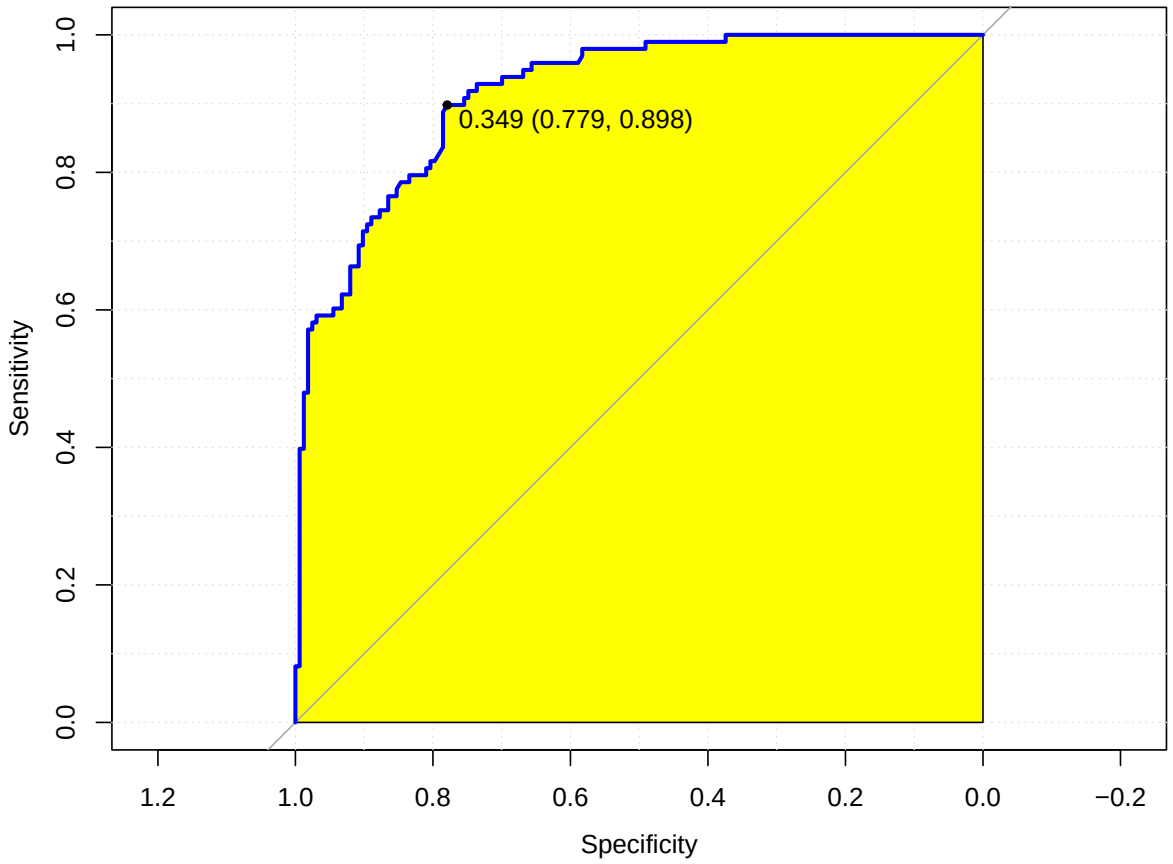
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
#threshold which maximizes sensitivity and specificity
c.index <- which.max(heart.roc$sensitivities + heart.roc$specificities)
heart.roc$thresholds[c.index]
```

```
## [1] 0.3485991
```

```r
#plot ROC curve
plot(heart.roc, print.thres = "best", col = "blue", grid = TRUE, lwd=2.5, auc.polygon = TRUE, auc.polyg
```



The threshold which maximizes specificity and sensitivity is 0.349, where the corresponding values of specificity and sensitivity are 0.779 and 0.898. This means that our best model can successfully predict a heart disease diagnosis 89.8% of the time, and also predict which patients who do not have heart disease 77.9% of the time.

The specificity and sensitivity values maximizes our model's ability to diagnose heart disease and the ability to identify patients who do not have heart diseases while keeping the estimate prediction error at the lowest.