CS624 Quiz2

Duc Le

11/19/2020

Problem 1

```
library(faraway)
library(ROSE)
```

```
## Loaded ROSE 0.0-3
```

```
library(AER)
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
## Registered S3 methods overwritten by 'car':
##   method                          from
##   influence.merMod                lme4
##   cooks.distance.influence.merMod lme4
##   dfbeta.influence.merMod         lme4
##   dfbetas.influence.merMod        lme4
```

```
##
## Attaching package: 'car'
```

```
## The following objects are masked from 'package:faraway':
##
##     logit, vif
```

```
## Loading required package: lmtest
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following objects are masked from 'package:faraway':
##
##     rats, solder
```

```r
library(plyr)
```

```
##
## Attaching package: 'plyr'

## The following object is masked from 'package:faraway':
##
##     ozone
```

```r
library(MASS)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:car':
##
##     recode

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(foreign)

data(wcgs)
```

```r

new.wcgs = na.omit(wcgs)
attach(new.wcgs)
new.wcgs$arcus = as.factor(ifelse(new.wcgs$arcus == "absent", 0, 1))
new.wcgs$typechd = as.factor(ifelse(new.wcgs$typechd == "none", 0, 1))
new.wcgs$chd = as.factor(ifelse(new.wcgs$chd == "no", 0, 1))
new.wcgs$dibep = as.factor(ifelse(new.wcgs$dibep == "A", 0, 1))

values = levels(behave)
new.wcgs$behave = mapvalues(factor(new.wcgs$behave), from = values, to = seq(length(values)))

new.wcgs = new.wcgs %>% select(-c(timechd, typechd, behave))
```

1a)

```r

base = glm(dibep~., data = new.wcgs, family = "binomial")
final1 = stepAIC(base, trace = 0)
```

```r
summary(final1)
```

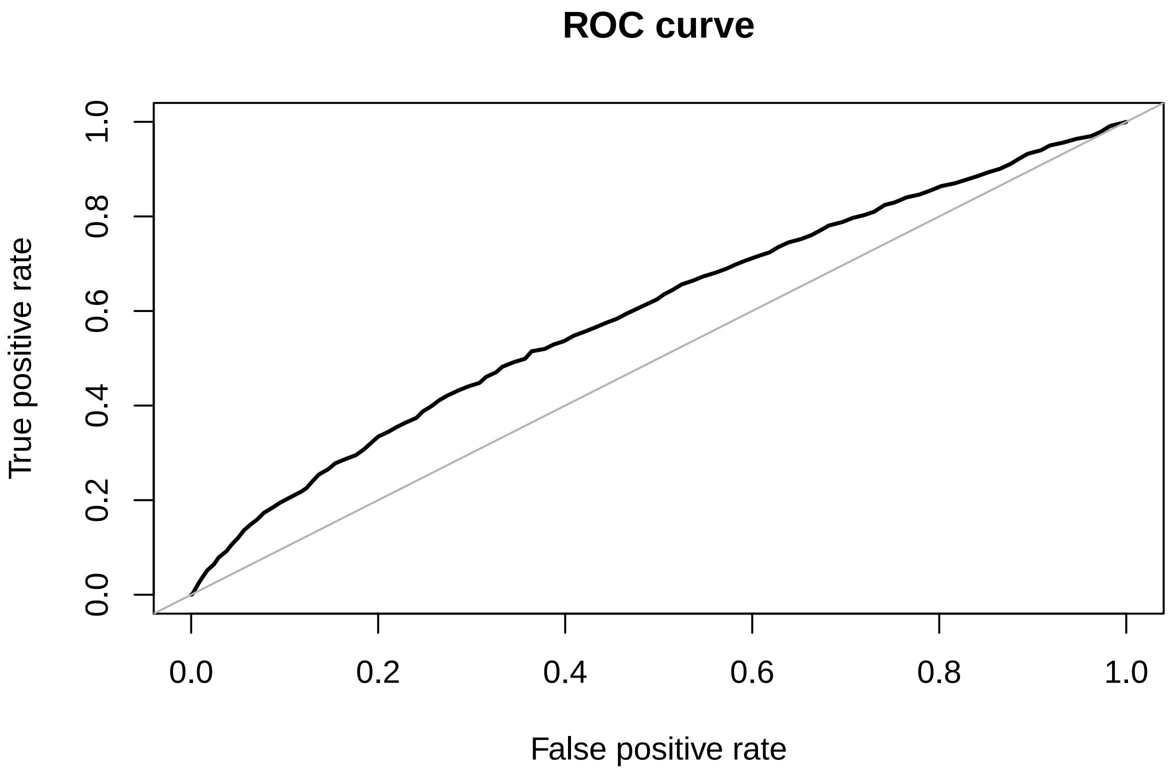
1b)

```
##
## Call:
## glm(formula = dibep ~ age + height + sdp + chol + cigs + chd,
##     family = "binomial", data = new.wcgs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8747  -1.1392   0.7039   1.1674   1.4753
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.8436316  1.1383946  -4.255 2.09e-05 ***
## age          0.0275033  0.0067612   4.068 4.75e-05 ***
## height       0.0325907  0.0145338   2.242  0.02493 *
## sdp          0.0065838  0.0024929   2.641  0.00826 **
## chol         0.0012634  0.0008613   1.467  0.14244
## cigs         0.0112907  0.0025458   4.435 9.21e-06 ***
## chd1         0.6695975  0.1450775   4.615 3.92e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4352.7  on 3139  degrees of freedom
## Residual deviance: 4255.9  on 3133  degrees of freedom
## AIC: 4269.9
##
## Number of Fisher Scoring iterations: 4
```

The model was able to predict dibep best by using age + height + sdp + chol + cigs and chd. Quick overview of the coefficients: All of the features chosen are statistiscally significant with the exception of chol.

The coefficients mean for example: a unit increase in age (a continuous var) will increase the log(odds of being passive) by 0.027. Interpreation for a binary feature: If the person has coronary heart disease (1) then his log(odds of being passive) will increase by 0.00126.

```r
pp1 = predict(final1, type = "response")
roc1 = roc.curve(dibep, pp1)
```



1c)

```
roc1
```

```
## Area under the curve (AUC): 0.596
```

The AUC obtained under the ROC is 0.596. This is obviously not an ideal AUC score we're striving for. However, given the limited time of the exam, I cannot afford to further investigate to why the model yielded an undesireable AUC score.

```
best_threshold_index = which.max(roc1$true.positive.rate - roc1$false.positive.rate)
roc1$thresholds[best_threshold_index]
```

1d)

```
## [1] 0.5045996
```

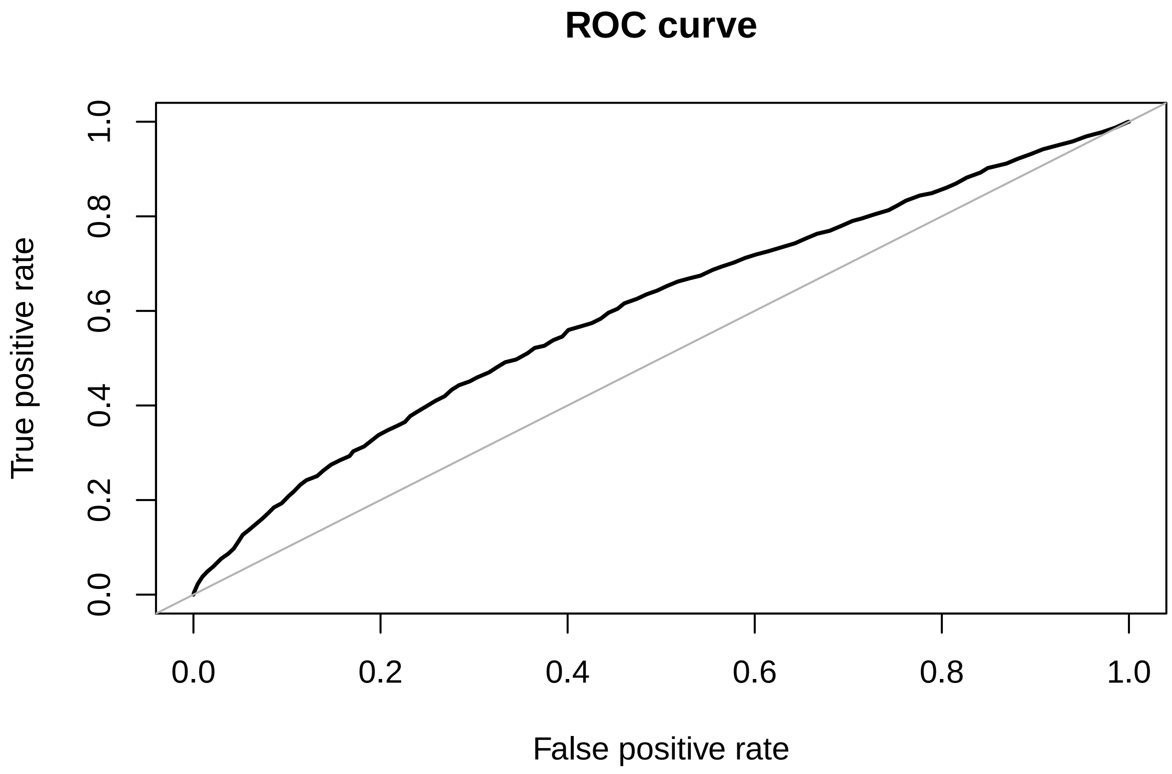
```
final2 = stepAIC(base, ~.^2, trace = 0)
final3 = stepAIC(base, ~.^3, trace = 0)

pp2 = predict(final2, type = "response")
roc2 = roc.curve(dibep, pp2)
roc2
```

1e)

```
## Area under the curve (AUC): 0.601
```

```
pp3 = predict(final3, type = "response")
roc3 = roc.curve(dibep, pp3)
```



```
roc3
```

```
## Area under the curve (AUC): 0.601
```

The 2 new models yield the same results in terms of the AUC score. They both performed better than the original log model that did not have interactions.

1f) The first logistic model performed poorly, only having an AUC of 0.596. The following 2 interactive models performed slightly better. A guess on this improvement could be because some features are dependent on one another, thus 2 & 3 way interactions between them will yield better results. If given more time, an approach I could do to improve the accuracy + AUC of all these models could be to look more into the variables' meanings & do a better job of feature selecting.

Problem 2

```
d = read.table("https://data.princeton.edu/wws509/datasets/ships.dat")

poisson.base = glm(damage~., offset(log(months)), data = d, family = poisson)
poisson.final = stepAIC(poisson.base, trace = 0)
```

2a)

```r
1 - pchisq(poisson.final$deviance, poisson.final$df.residual)
```

2b)

```
## [1] 0
```

```r
dispersiontest(poisson.final)
```

2c)

```
##
##  Overdispersion test
##
## data:  poisson.final
## z = 2.6742, p-value = 0.003745
## alternative hypothesis: true dispersion is greater than 1
## sample estimates:
## dispersion
##   1.956256
```

The dispersion is 1.956 which is over 1.

```r
quasi.poisson =  glm(poisson.final$call, family = quasipoisson, data = d)
summary(quasi.poisson)
```

2d)

```
##
## Call:
## glm(formula = poisson.final$call, family = quasipoisson, data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5484  -1.3867  -0.4307   0.5222   3.1152
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.786e-01  4.587e-01   0.389 0.700447
## typeB                  6.701e-01  3.595e-01   1.864 0.074653 .
## typeC                 -1.192e+00  5.422e-01  -2.198 0.037863 *
## typeD                 -8.294e-01  4.763e-01  -1.741 0.094420 .
## typeE                 -1.493e-01  3.888e-01  -0.384 0.704284
## construction1965-69  1.087e+00  2.967e-01   3.665 0.001224 **
## construction1970-74  1.500e+00  3.721e-01   4.031 0.000487 ***
## construction1975-79  8.545e-01  4.568e-01   1.871 0.073628 .
## operation1975-79     7.284e-01  2.246e-01   3.243 0.003461 **
## months               6.697e-05  1.411e-05   4.746 7.92e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 2.740679)
##
##     Null deviance: 614.539  on 33  degrees of freedom
## Residual deviance:  70.498  on 24  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```
1 - pchisq(quasi.poisson$deviance, quasi.poisson$df.residual)
```

```
## [1] 1.837649e-06
```

After refitting the model with Quasi-Poisson, I can observe that there isn't any major changes when it comes to the coefficients. However the number of significant features has been massively reduced. With the Quasi-Poisson model, only 4 of the features are ruled statistically significant. Using Chi-Sq GoF test, the Quasi-Poisson model provided us better results.