Stat 435 lecture notes 9: supplementary

Penalized logistic regression

```
load("sim_geno.rda")
load("sim_trait.rda")
# take a smaller par
sim_geno_a = sim_geno[1:500, 1:100]
sim_trait_a = sim_trait[1:100]
sim_trait_a1 = as.numeric(abs(sim_trait_a) > 0.5)

set.seed(123)

library(glmnet)

lga = cv.glmnet(t(sim_geno_a), sim_trait_a1, family = "binomial",
    alpha = 1, type.measure = "class")

Ma = glmnet(t(sim_geno_a), sim_trait_a1, alpha = 1, family = "binomial",
    lambda = lga$lambda.min)
# Display non-zero regression coefficients
coef(Ma)[coef(Ma) != 0]
```

```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient
```

```
##  [1]  0.906593001 -0.050484903 -0.142419280  0.198313448 -0.055539426
##  [6] -0.548750948  0.070020997  0.040914023  0.855089233  0.292877286
## [11]  0.565699508 -0.097575022  0.048652471  0.077754855 -0.142844461
## [16] -0.463245195 -0.236106215 -0.273159753 -0.569664748 -0.067262219
## [21] -0.332920025 -0.527457719 -1.008362016  0.053478316 -0.098952564
## [26]  0.311832251 -0.229191419 -0.611895254  0.048090837  0.506896889
## [31]  0.084019436  0.008013738  0.263433125 -0.114768149  0.685211115
## [36]  0.559790810 -0.754621693  0.018782008  0.460460521 -0.118559394
## [41]  0.508248393 -0.043992653
```

```
## obtain p-values
library(hdi)
```

```
## Warning: package 'hdi' was built under R version 3.5.3

## Loading required package: scalreg

## Warning: package 'scalreg' was built under R version 3.5.2

## Loading required package: lars

## Warning: package 'lars' was built under R version 3.5.2

## Loaded lars 1.2
```

```r
prj_est = lasso.proj(t(sim_geno_a), sim_trait_a1, family = "binomial")
```

```
## Nodewise regressions will be computed as no argument Z was provided.

## You can store Z to avoid the majority of the computation next time around.

## Z only depends on the design matrix x.

## Warning in warning.sigma.message(): Overriding the error variance estimate
## with your own value. The initial estimate implies an error variance
## estimate and if they don't correspond the testing might not be correct
## anymore.
```

```r
prj_est$pval[1:10]
```

```
##  [1] 0.2981953 0.7763949 0.5395813 0.8914452 0.5055726 0.5910186 0.9138428
##  [8] 0.8536747 0.9884205 0.4794335
```

```r
### Ridge regression

lgb = cv.glmnet(t(sim_geno_a), sim_trait_a1, family = "binomial",
    alpha = 0, type.measure = "class")

Mb = glmnet(t(sim_geno_a), sim_trait_a1, alpha = 0, family = "binomial",
    lambda = lga$lambda.min)


## obtain p-values
prj_estb = ridge.proj(t(sim_geno_a), sim_trait_a1, family = "binomial")
```

```
## Warning in warning.sigma.message(): Overriding the error variance estimate
## with your own value. The initial estimate implies an error variance
## estimate and if they don't correspond the testing might not be correct
## anymore.
```

```r
prj_estb$pval[1:10]
```

```
##  [1] 0.9594145 1.0000000 1.0000000 1.0000000 0.9233757 0.9354331 1.0000000
##  [8] 1.0000000 1.0000000 0.9020482
```

Multinomial logistic regression

Please go to webpage: https://stats.idre.ucla.edu/r/dae/multinomial-logistic-regression/

The data set contains variables on 200 students. The outcome variable is `prog`, program type. The predictor variables are social economic status, `ses`, a three-level categorical variable and writing score, `write`, a continuous variable.

```r
require(foreign)
require(nnet)
require(ggplot2)
require(reshape2)

ml <- read.dta("hsbdemo.dta")

with(ml, table(ses, prog))
```

```
##         prog
## ses      general academic vocation
##   low         16       19       12
##   middle      20       44       31
##   high         9       42        7
```

```r
# set baseline level for 'prog'
ml$prog2 <- relevel(ml$prog, ref = "academic")
test <- multinom(prog2 ~ ses + write, data = ml)
```

```
## # weights:  15 (8 variable)
## initial  value 219.722458
## iter  10 value 179.982880
## final  value 179.981726
## converged
```

```r
# 'ses = low' is the baseline and absorbed into the intercept
summary(test)
```

```
## Call:
## multinom(formula = prog2 ~ ses + write, data = ml)
##
## Coefficients:
##          (Intercept)  sesmiddle    seshigh      write
## general     2.852198 -0.5332810 -1.1628226 -0.0579287
## vocation    5.218260  0.2913859 -0.9826649 -0.1136037
##
## Std. Errors:
##          (Intercept) sesmiddle   seshigh      write
## general     1.166441 0.4437323 0.5142196 0.02141097
## vocation    1.163552 0.4763739 0.5955665 0.02221996
##
## Residual Deviance: 359.9635
## AIC: 375.9635
```

```r
# calculate z-test value
z <- summary(test)$coefficients/summary(test)$standard.errors
z
```

```
##          (Intercept)  sesmiddle   seshigh     write
## general     2.445214 -1.2018081 -2.261334 -2.705562
## vocation    4.484769  0.6116747 -1.649967 -5.112689
```

```
# calculate two-sided p-values using z-tests
p <- (1 - pnorm(abs(z), 0, 1)) * 2
p
```

```
##           (Intercept) sesmiddle    seshigh        write
## general  0.0144766100 0.2294379 0.02373856 6.818902e-03
## vocation 0.0000072993 0.5407530 0.09894976 3.176045e-07
```

```
# use fitted model to predict probabilities
head(pp <- fitted(test))
```

```
##    academic   general  vocation
## 1 0.1482764 0.3382454 0.5134781
## 2 0.1202017 0.1806283 0.6991700
## 3 0.4186747 0.2368082 0.3445171
## 4 0.1726885 0.3508384 0.4764731
## 5 0.1001231 0.1689374 0.7309395
## 6 0.3533566 0.2377976 0.4088458
```

Poisson regression

The following contents are adopted from: https://stats.idre.ucla.edu/r/dae/poisson-regression/. In this example, `num_awards` is the outcome variable and indicates the number of awards earned by students at a high school in a year, `math` is a continuous predictor variable and represents students' scores on their math final exam, and `prog` is a categorical predictor variable with three levels indicating the type of program in which the students were enrolled. It is coded as 1 = "General", 2 = "Academic" and 3 = "Vocational".

```
require(ggplot2)
require(sandwich)
```

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

```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'sandwich'
```

```
require(msm)
```

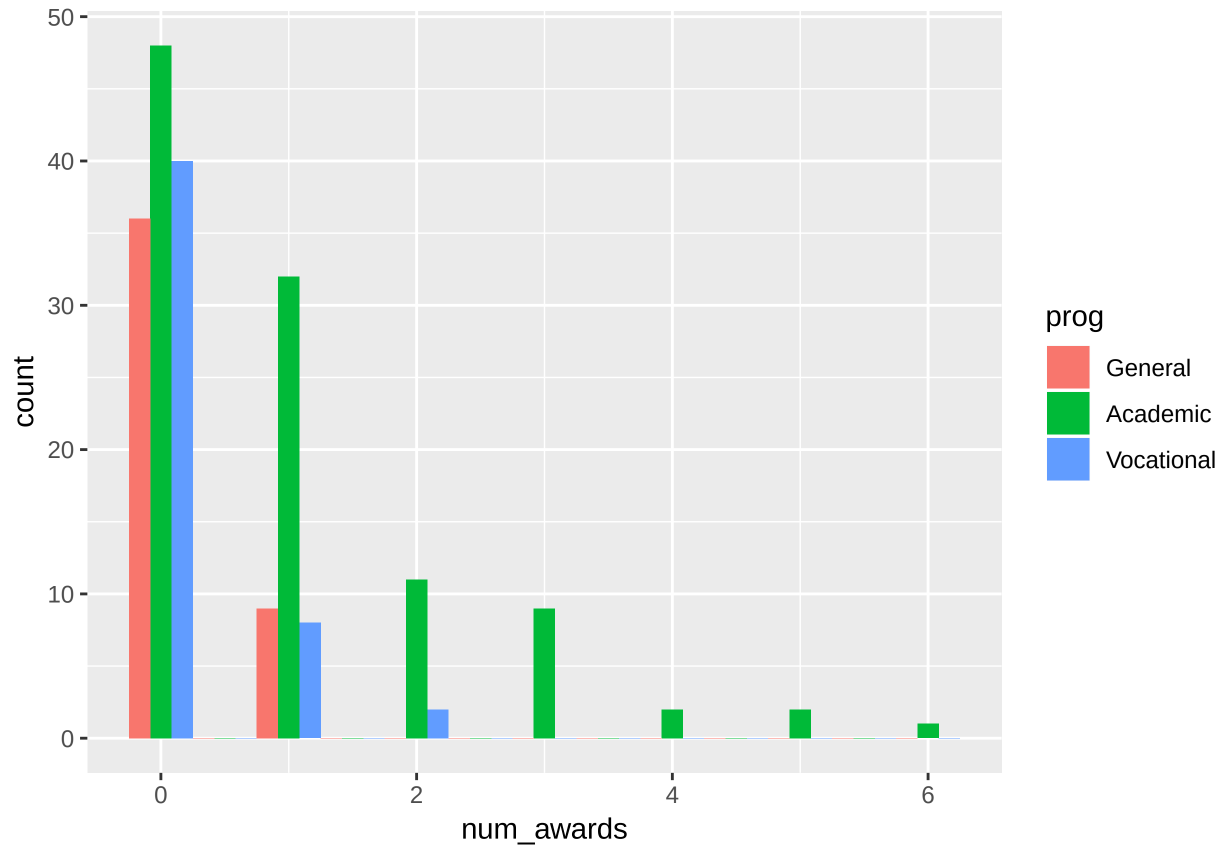
```
## Loading required package: msm
```

```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'msm'
```

```
p <- read.csv("poisson_sim.csv")
p <- within(p, {
    prog <- factor(prog, levels = 1:3, labels = c("General",
        "Academic", "Vocational"))
    id <- factor(id)
})
summary(p)
```

```
##        id         num_awards           prog          math
##  1      :  1   Min.   :0.00   General    : 45   Min.   :33.00
##  2      :  1   1st Qu.:0.00   Academic   :105   1st Qu.:45.00
##  3      :  1   Median :0.00   Vocational: 50   Median :52.00
##  4      :  1   Mean   :0.63                     Mean   :52.65
##  5      :  1   3rd Qu.:1.00                     3rd Qu.:59.00
##  6      :  1   Max.   :6.00                     Max.   :75.00
##  (Other):194
```

```
ggplot(p, aes(num_awards, fill = prog)) + geom_histogram(binwidth = 0.5,
    position = "dodge")
```



```
summary(m1 <- glm(num_awards ~ prog + math, family = "poisson",
    data = p))
```

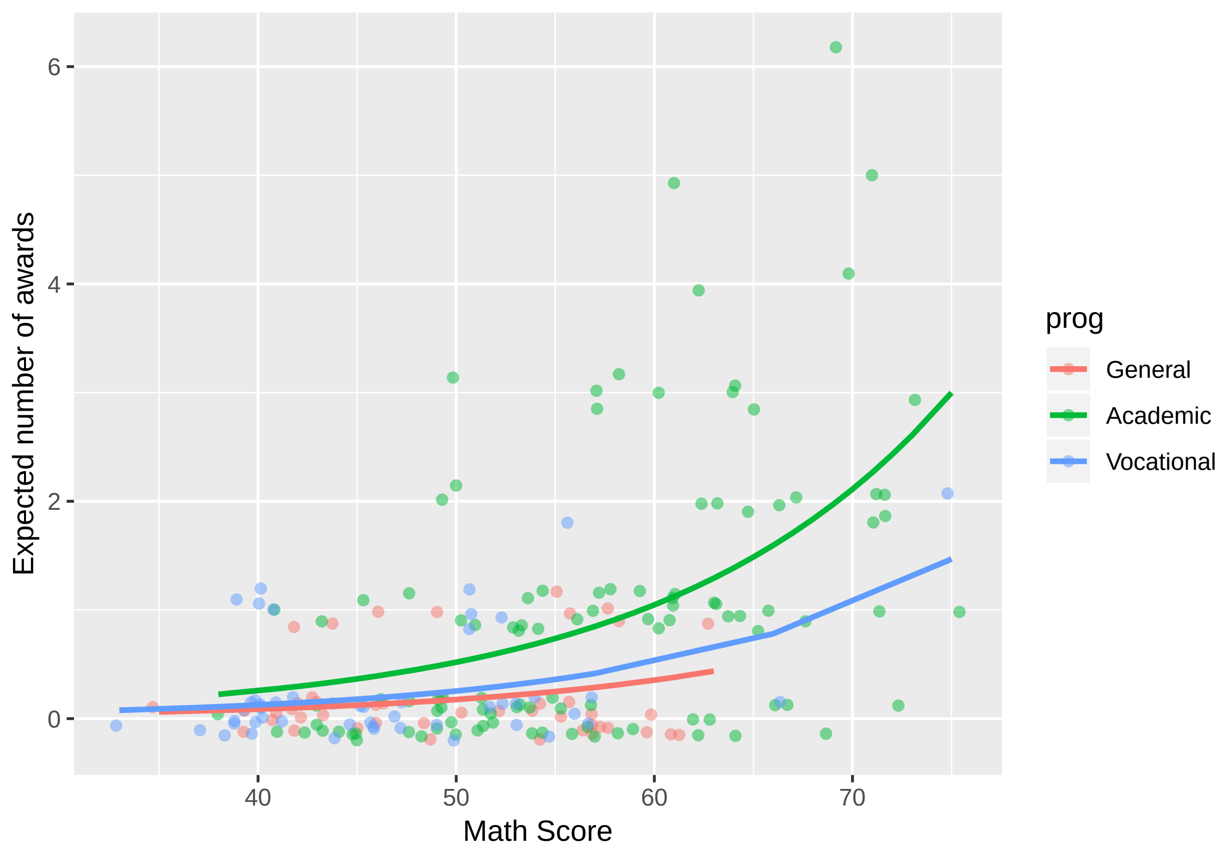
```
##
## Call:
## glm(formula = num_awards ~ prog + math, family = "poisson", data = p)
##
## Deviance Residuals:
##     Min       1Q    Median       3Q      Max
## -2.2043  -0.8436  -0.5106   0.2558   2.6796
##
## Coefficients:
```

```
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -5.24712    0.65845  -7.969 1.60e-15 ***
## progAcademic    1.08386    0.35825   3.025  0.00248 **
## progVocational  0.36981    0.44107   0.838  0.40179
## math            0.07015    0.01060   6.619 3.63e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
## Number of Fisher Scoring iterations: 6
```

```
## calculate and store predicted values
p$phat <- predict(m1, type = "response")

## order by program and then by math
p <- p[with(p, order(prog, math)), ]

## create the plot
ggplot(p, aes(x = math, y = phat, colour = prog)) + geom_point(aes(y = num_awards),
    alpha = 0.5, position = position_jitter(h = 0.2)) + geom_line(size = 1) +
    labs(x = "Math Score", y = "Expected number of awards")
```



Expected number of awards
6
4
2
0
40
50
60
70
Math Score
prog
General
Academic
Vocational