HW 4: Clustering and Regression

Kaori Hirano

2023-07-11

Packages

```
suppressPackageStartupMessages(library(tidyverse))
library(ISLR2)
library(broom) # for tidy function
library(patchwork) # for plot placement
library(ggdendro) # for dendrograms
library(mdsr) # for later examples
library(ggplot2)
suppressPackageStartupMessages(library(openintro))
suppressPackageStartupMessages(library(boot))
library(FNN)
suppressPackageStartupMessages(library(glmnet))
suppressPackageStartupMessages(library(factoextra))
suppressPackageStartupMessages(library(cluster))
```

Data

```
data('evals')
```

Exercises

Data Prep

Q1

```
# assigns num times a prof teaches a class to total courses taught into new dataframe
#evals <- evals %>% group_by(prof_id) %>% mutate(prof_num_cls = (count = n()))
evals <- add_count(evals, prof_id)
evals$prof_num_cls <- evals$n
d <- evals
d <- d %>% select(-n)
```

Q2

```
# removes course id, prof id, and cls did eval
d <- d %>% select(-one_of('course_id', 'prof_id', 'cls_did_eval')) %>% ungroup()
# removes bty flower through btymupper
d <- d %>% select(-(bty_f1lower:bty_m2upper))

# splits data in training and test set by 70/30
set.seed(280)
train <- sample(c(TRUE, FALSE), nrow(d),
     replace = TRUE, prob=c(.7,.3))
test <- (!train)

# making it into dataframes instead of subsets
set.seed(280)
sample1 <- sample(c(TRUE, FALSE), nrow(d), replace=TRUE, prob=c(0.7,0.3))
cv <- d[sample1, ]
val <- d[!sample1, ]
```

Clustering and Modeling

```
cluster_eval <- function(col, data, cluster_num) {
  evals_kmeans <- kmeans(data %>% select(-one_of('rank', 'ethnicity', 'gender',
                'language', 'cls_level', 'cls_profs', 'cls_credits', 'pic_outfit',
```

```
                'pic_color')), centers = cluster_num, nstart = 50)
  # cluster assignment
  data <- data %>% mutate(col = evals_kmeans$cluster)
}
```

```
# adding clusters to dataset
# because my function didn't work :(
# adds 2
# gets cluster assignments
evals_kmeans2 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 2, nstart = 50)

# takes the training cases and assigns them a cluster num in new df to make the next step
# retrospectively, I didn't need a new data frame and could have assigned
# it to a col, but with the way I was doing the cv at the time with a
# function this made more sense
d2 <- d[train,] %>%
mutate(c2 = factor(evals_kmeans2$cluster))

# adds 3
evals_kmeans3 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 3, nstart = 50)
# cluster assignment
d3 <- d[train,] %>%
mutate(c3 = factor(evals_kmeans3$cluster))

# adds 4
evals_kmeans4 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 4, nstart = 50)
# cluster assignment
d4 <- d[train,] %>%
mutate(c4 = factor(evals_kmeans4$cluster))
```

```r
# adds 5
evals_kmeans5 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 5, nstart = 50)
# cluster assignment
d5 <- d[train,] %>%
mutate(c5 = factor(evals_kmeans5$cluster))

# adds 6
evals_kmeans6 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 6, nstart = 50)
# cluster assignment
d6 <- d[train,] %>%
mutate(c6 = factor(evals_kmeans6$cluster))

# adds 7
evals_kmeans7 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 7, nstart = 50)
# cluster assignment
d7 <- d[train,] %>%
mutate(c7 = factor(evals_kmeans7$cluster))

# adds 8
evals_kmeans8 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 8, nstart = 50)
# cluster assignment
d8 <- d[train,] %>%
mutate(c8 = factor(evals_kmeans8$cluster))

# adds 9
```

```
evals_kmeans9 <- kmeans(d[train,]
                        %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                           'cls_level', 'cls_profs', 'cls_credits',
                                           'pic_outfit', 'pic_color')),
                        centers = 9, nstart = 50)
# cluster assignment
d9 <- d[train,] %>%
mutate(c9 = factor(evals_kmeans9$cluster))

# adds 10
evals_kmeans10 <- kmeans(d[train,]
                         %>% select(-one_of('rank', 'ethnicity', 'gender','language',
                                            'cls_level', 'cls_profs', 'cls_credits',
                                            'pic_outfit', 'pic_color')),
                         centers = 10, nstart = 50)
# cluster assignment
d10 <- d[train,] %>%
mutate(c10 = factor(evals_kmeans10$cluster))
```

```
# cv for 2
# basic model is all categorical and cluster
glm2 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c2, data = d2)
# cross validation
set.seed(281)
cv_err2 <- cv.glm(data = d2, glmfit = glm2, K = 10)
# gives MSE
cv_err2$delta
```

```
[1] 0.2810875 0.2800867
```

```
# cv for 3
# basic model is all categorical and cluster
glm3 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c3, data = d3)
# cross validation
set.seed(281)
cv_err3 <- cv.glm(data = d3, glmfit = glm3, K = 10)
# gives MSE
cv_err3$delta
```

```
[1] 0.2728877 0.2720360
```

```r
# cv for 4
# basic model is all categorical and cluster
glm4 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c4, data = d4)
# cross validation
set.seed(281)
cv_err4 <- cv.glm(data = d4, glmfit = glm4, K = 10)
# gives MSE
cv_err4$delta
```

```
[1] 0.2714726 0.2705120
```

```r
# cv for 5
# basic model is all categorical and cluster
glm5 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c5, data = d5)
# cross validation
set.seed(281)
cv_err5 <- cv.glm(data = d5, glmfit = glm5, K = 10)
# gives MSE
cv_err5$delta
```

```
[1] 0.2760367 0.2749764
```

```r
# cv for 6
# basic model is all categorical and cluster
glm6 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c6, data = d6)
# cross validation
set.seed(281)
cv_err6 <- cv.glm(data = d6, glmfit = glm6, K = 10)
# gives MSE
cv_err6$delta
```

```
[1] 0.2723326 0.2711995
```

```r
# cv for 7
# basic model is all categorical and cluster
glm7 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c7, data = d7)
# cross validation
set.seed(281)
cv_err7 <- cv.glm(data = d7, glmfit = glm7, K = 10)
# gives MSE
cv_err7$delta
```

```
[1] 0.2758902 0.2746527
```

```r
# cv for 8
# basic model is all categorical and cluster
glm8 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c8, data = d8)
# cross validation
set.seed(281)
cv_err8 <- cv.glm(data = d8, glmfit = glm8, K = 10)
# gives MSE
cv_err8$delta
```

```
[1] 0.2823946 0.2809210
```

```r
# cv for 9
# basic model is all categorical and cluster
glm9 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
            + cls_credits + pic_outfit + pic_color + c9, data = d9)
# cross validation
set.seed(281)
cv_err9 <- cv.glm(data = d9, glmfit = glm9, K = 10)
# gives MSE
cv_err9$delta
```

```
[1] 0.2844476 0.2828824
```

```
# cv for 10
# basic model is all categorical and cluster
glm10 <- glm(score ~ rank + ethnicity + gender + language + cls_level + cls_profs
             + cls_credits + pic_outfit + pic_color + c10, data = d10)
# cross validation
set.seed(281)
cv_err10 <- cv.glm(data = d10, glmfit = glm10, K = 10)
# gives MSE
cv_err10$delta
```

```
[1] 0.2765239 0.2749649
```

4 clusters has the lowest MSE (.2714, .2705), so it is the optimal number of clusters for this model.

```
# gets numeric only for k means output/predictions
numeric_cv <- cv %>% select(age, cls_perc_eval, cls_students, bty_avg, prof_num_cls)

#set seed
set.seed(282)

# do k means clustering on train again
kmeans_out <- kmeans(cv %>%
                       select(-one_of('rank', 'ethnicity', 'gender','language',
                                      'cls_level', 'cls_profs', 'cls_credits',
                                      'pic_outfit', 'pic_color')),
                     centers = 4, nstart = 50)

# creates dataset with the clusters stored for train
cdata <- cbind(d[train,], cluster = kmeans_out$cluster)

# build model on train again
glmfinal <- glm(score ~ rank + ethnicity + gender + language + cls_level +
                cls_profs + cls_credits + pic_outfit + pic_color + cluster,
                data = cdata)

# getting predictions on kmeans numerical only, test data
pred_clust  <- get.knnx(kmeans_out$center, val %>%
                          select(-one_of('rank', 'ethnicity', 'gender','language',
                                         'cls_level', 'cls_profs', 'cls_credits',
                                         'pic_outfit', 'pic_color')),
```

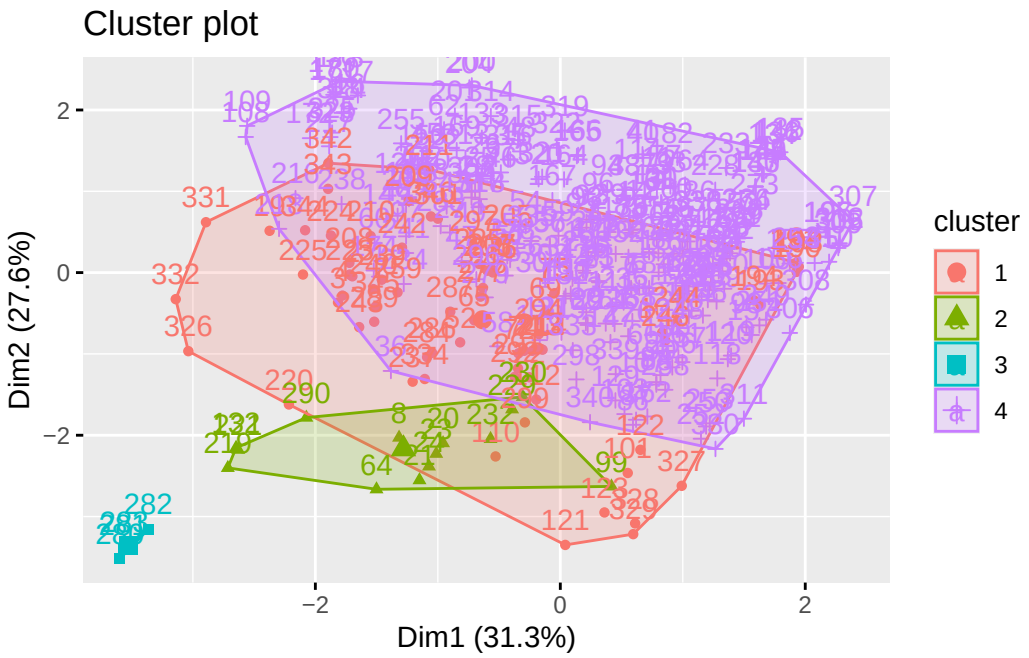
```
                        1)$nn.index[,1]

# assigning cluster data to test set
tdata <- cbind(d[test, ], cluster = pred_clust)

# build model on test again
glmfinal <- glm(score ~ rank + ethnicity + gender + language + cls_level +
                  cls_profs + cls_credits + pic_outfit + pic_color + cluster,
                data = tdata)

# cross validation to get the MSE
set.seed(281)
cv_err10 <- cv.glm(data = tdata, glmfit = glmfinal, K = 10)
# gives MSE
cv_err10$delta
```

```
[1] 0.3078647 0.3040451
```

The MSE for the test data with the optimal number of clusters is .30 (.3078, .3040).

Q4

```
# plots by first two principal components
fviz_cluster(kmeans_out, numeric_cv)
```




```
kmeans_out$centers
```

```
     score      age cls_perc_eval cls_students  bty_avg prof_num_cls
1 4.072603 46.72603      64.57871     96.82192 4.479411     5.602740
2 3.871429 48.64286      51.57302    245.28571 4.250000     6.214286
3 4.800000 51.00000      63.53843    567.75000 6.167000     5.000000
4 4.192188 48.91016      78.93377     25.61328 4.312387     6.742188
```

Based on this plot, which plots the two principal components, we can see that there is a lot of overlap for clusters 1 and 4, with less overlap in 1 and 2, with no overlap with the smallest cluster 3. By looking at the output itself, we can see that ratings are highest in cluster 3, which is the smallest, by .7 points, higher class sizes by 200, but also notably higher beauty averages (2 points higher) on average. Clusters 1 and 4 are also similar in score, beauty average, and num classes, and differ most notably by the class percent eval and class size. Clusters 1, 2, and 4 have similar beauty avg scores. Clusters 2 and 4 are similar in age, while 1 is slightly younger. Overall, these data show us that there are many similarities between clusters 1, 2, and 4, while cluster 3 is more different/shares less similarities than the other three, most notably in score, beauty average, class size, and class size.

Lasso

```
# set up lasso
# sets up matrix
set.seed(283)
x <- model.matrix(score ~ rank + ethnicity + gender + language + cls_level +
                    cls_profs + cls_credits + pic_outfit + pic_color + prof_num_cls
                  + age + cls_perc_eval + cls_students + bty_avg,
                  d)[, -1]
y <- d$score
y_val <- y[test] #gets y of test set

grid <- 10^seq(10, -2, length = 100) # sets lam

set.seed(283)
lasso_mod <- glmnet(x[train,], y[train], alpha = 1,
    lambda = grid)


# get l
set.seed(283)
cv_out_lasso <- cv.glmnet(x[train, ], y[train], alpha = 1)
bestlaml <- cv_out_lasso$lambda.min
bestlaml
```

```
[1] 0.008590297
```

```
# gets MSE
lasso_pred <- predict(lasso_mod, s = bestlaml,
    newx = x[test, ])
mean((lasso_pred - y_val)^2)
```

```
[1] 0.2195321
```

```
# refits to full data, uses best l to get coefs/important vars
out <- glmnet(x, y, alpha = 1, lambda = grid)
lasso_coef <- predict(out, type = 'coefficient', s = bestlaml)
lasso_coef
```

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                    s1
(Intercept)            3.780665e+00
ranktenure track      -5.721904e-02
ranktenured            .
ethnicitynot minority  1.804029e-01
gendermale             1.745965e-01
languagenon-english   -1.262305e-01
cls_levelupper         1.819702e-02
cls_profssingle        .
cls_creditsone credit  4.529527e-01
pic_outfitnot formal  -1.316095e-01
pic_colorcolor        -1.326121e-01
prof_num_cls           3.723984e-02
age                   -8.660003e-03
cls_perc_eval          3.302079e-03
cls_students           6.644968e-05
bty_avg                6.112340e-02
```

The best lambda was .008, and the mse was .219. The important variables were tenure track, ethnicity, gender, language, class level, credit level, picture outfit formality, pic color, number of classes, age, class % eval, students, and beauty average (all except tenured rank and single prof).

Q6

The approach that did better in terms of test MSE was k-means, which had an MSE of .30 while the lass had an MSE of .219 for the test set.

A way to combine the two methods could be to use lasso first to limit the variables so that only the most significant ones are included, then use k-means. This could help get more accurate clusters because the variables being used will be more significant.

Lasso is able to tell us which predictors are more important than other ones when predicting data, while k-means is able to tell us about how the data are related to each other. K-means is more focused on those relationships and similarities between clusters/data while lasso is very good at showing the overall what is important in the model in terms of predictors. The advantages and disadvantages will be dependent on if you are looking more for a model that is going to tell you what is important in prediction or how the data are grouped together or classified.