Statistical Learning Weekly Assigment 05

Joshua Damm

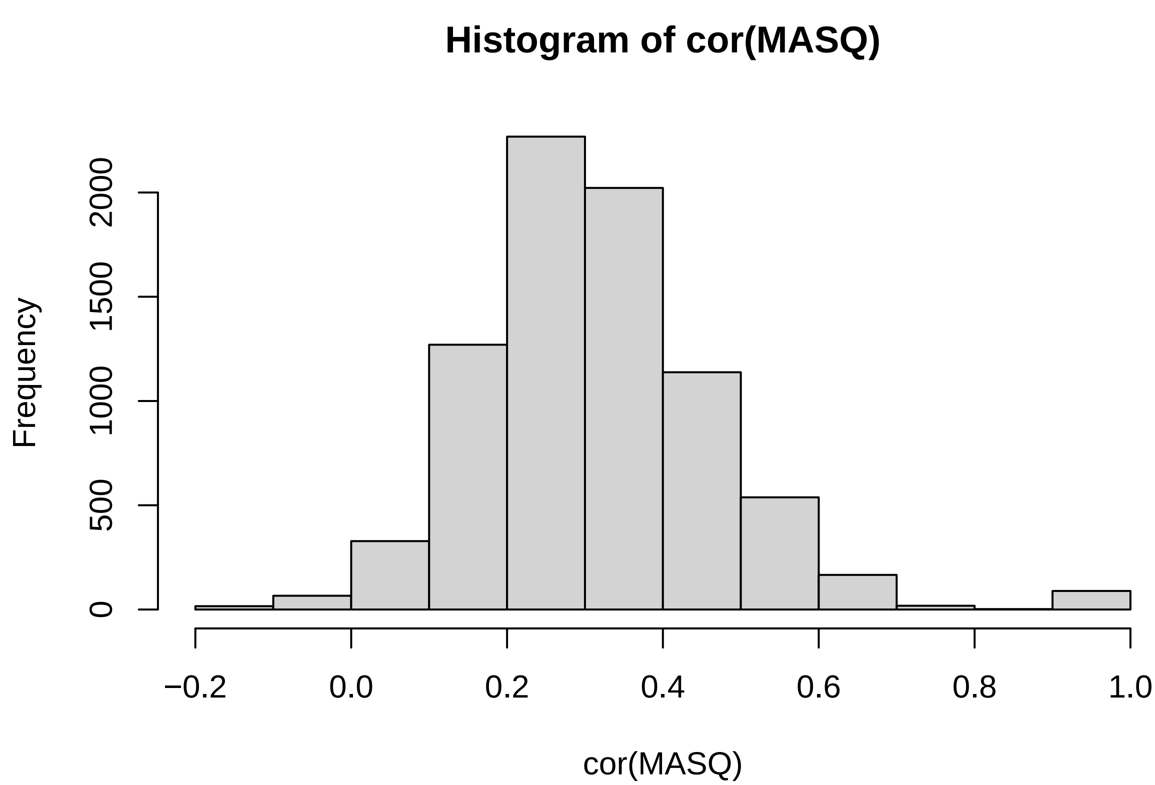
2024-03-14

```r
train <- readRDS("masq_train.Rda")
test <- readRDS("masq_test.Rda")
```

a) Inspect multicollinearity between the numeric MASQ items. What do you expect about relative performance of lasso, ridge and elastic net regression?

```r
# select MASQ columns
MASQ = train[, 12:100]

# inspect multicollinearity
hist(cor(MASQ))
```



The pairwise correlations / the correlation matrix follow a normal distribution with a mean around 0.3. There is a slight peak at 1, because of the diagonal entries of the matrix. All 3 techniques might be a feasible solution. Lasso might be an option, since it could remove redundant variables. Elastic net might be the best choice as it combines the benefits of lasso and ridge.

(b) Pick three candidate procedures from ridge, elastic net, lasso, relaxed lasso.

I will choose lasso, ridge and elastic net with an alpha of 0.5.

(c) Use library glmnet to fit the models, and select the most accurate model through 10-fold cross-validation on the training set.

```r
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```r
x = model.matrix(D_DEPDYS ~ ., data = train)
x_test = model.matrix(D_DEPDYS ~ ., data = test)
y = train$D_DEPDYS

# setup
lasso = cv.glmnet(x, y, alpha = 1); lasso
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 1)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure       SE Nonzero
## min 0.01140    34  0.1721 0.005380      37
## 1se 0.04195    20  0.1773 0.004925      17
```

```r
ridge = cv.glmnet(x, y, alpha = 0); ridge
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure       SE Nonzero
## min  0.276    74  0.1721 0.004654     132
## 1se  1.774    54  0.1763 0.003817     132
```

```r
elastic = cv.glmnet(x, y, alpha = 0.5); elastic
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 0.5)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure       SE Nonzero
## min 0.01725    37  0.1712 0.005562      49
## 1se 0.08389    20  0.1767 0.004946      19
```

```
# predict models
predict_L = predict(lasso, s = "lambda.min", newx = x_test)
predict_R = predict(ridge, s = "lambda.min", newx = x_test)
predict_E = predict(elastic, s = "lambda.min", newx = x_test)

# get mean squared errors
MSE_L = mean((predict_L - test$D_DEPDYS)^2)
MSE_R = mean((predict_R - test$D_DEPDYS)^2)
MSE_E = mean((predict_E - test$D_DEPDYS)^2)

# benchmark
MSE_MAX = var(test$D_DEPDYS)

# cross-validated R2
CVR2_L = 1 - MSE_L/MSE_MAX; CVR2_L
```

```
## [1] 0.3331148
```

```
CVR2_R = 1 - MSE_R/MSE_MAX; CVR2_R
```

```
## [1] 0.3308643
```

```
CVR2_E = 1 - MSE_E/MSE_MAX; CVR2_E
```

```
## [1] 0.334857
```

Lasso shows the best performance with a cross-validated R2 of 0.3337474.

(d) Compute the misclassification rate (MCR) on the test set.

```
preds_L_1se = predict(lasso, newx = x[x_test, ], type = "response")
preds_L_min = predict(lasso, newx = x[x_test, ], type = "response",
                      s = "lambda.min")

tab_L_1se = prop.table(table(preds_L_1se > .5, y[x_test]))
tab_L_min = prop.table(table(preds_L_min > .5, y[x_test]))
tab_L_1se; tab_L_min
```

```
##
##                   0           1
##   FALSE 0.141621916 0.001009058
##   TRUE  0.001994920 0.855374105
```

```
##
##                       0            1
##   FALSE 0.1416857073 0.0007538942
##   TRUE  0.0019311289 0.8556292697
```

```
sum(diag(tab_L_1se)); sum(diag(tab_L_min))
```

```
## [1] 0.996996
```

```
## [1] 0.997315
```

For the se1_lambda criteria the MCR is a little higher.

(e) Use the coef method to extract the selected variables and their coefficients from the best-performing model.

```
L_coefs = coef(lasso, s = "lambda.min")
L_coefs[L_coefs[,1] != 0,]
```

```
##   (Intercept)       GENDERv      Leeftijd       DEMOG26       DEMOG32
## -4.814769e-01  8.170714e-03  1.149901e-03 -7.256811e-02  1.328822e-02
##       DEMOG34      DEMOG3NA       DEMOG53       DEMOG55       DEMOG62
## -6.421318e-05 -3.518655e-02 -3.346312e-02  1.993973e-02  5.647645e-02
##        MASQ01        MASQ02        MASQ03        MASQ05        MASQ13
##  3.481755e-02 -1.548544e-02 -6.228674e-03  7.338993e-04  8.940010e-03
##        MASQ14        MASQ16        MASQ18        MASQ21        MASQ22
##  2.865788e-03  6.948313e-02  3.406095e-03  3.920310e-03  2.403443e-02
##        MASQ24        MASQ29        MASQ30        MASQ31        MASQ33
##  5.460323e-03  7.899327e-04  2.265849e-02  8.409185e-03  2.287192e-03
##        MASQ37        MASQ38        MASQ41        MASQ43        MASQ54
##  1.893545e-02  5.152496e-03  2.533319e-02  5.592111e-03  1.655632e-03
##        MASQ59        MASQ60        MASQ62        MASQ70        MASQ76
## -1.041407e-02  9.627503e-03  1.395614e-02  4.299627e-03  9.622140e-03
##        MASQ78        MASQ89        MASQ90
##  9.813473e-03  2.667092e-02  1.438215e-02
```

```
Anhedonic_Depression = c(1, 14, 18, 21, 23, 26, 27, 30, 33, 35, 36, 39, 40, 44, 49, 53, 58, 66, 72, 78,
Anxious_Arousal = c(3, 19, 25, 45, 48, 52, 55, 57, 61, 67, 69, 73, 75, 79, 85, 87, 88)
General_Distress_Depression = c(6, 8, 10, 13, 16, 22, 24, 42, 47, 56, 64, 74)
General_Distress_Anxiety = c(2, 9, 12, 15, 20, 59, 63, 65, 77, 81, 82)
General_Distress_Mixed = c(4, 5, 17, 29, 31, 34, 37, 50, 51, 70, 76, 80, 83, 84, 90)
```