Solutions Week 5

Marjolein Fokkema

Exercise 1: Gene expression

```r
library("TH.data")
dim(Westbc$assay) ## contains the predictors
```

```
## [1] 7129   49
```

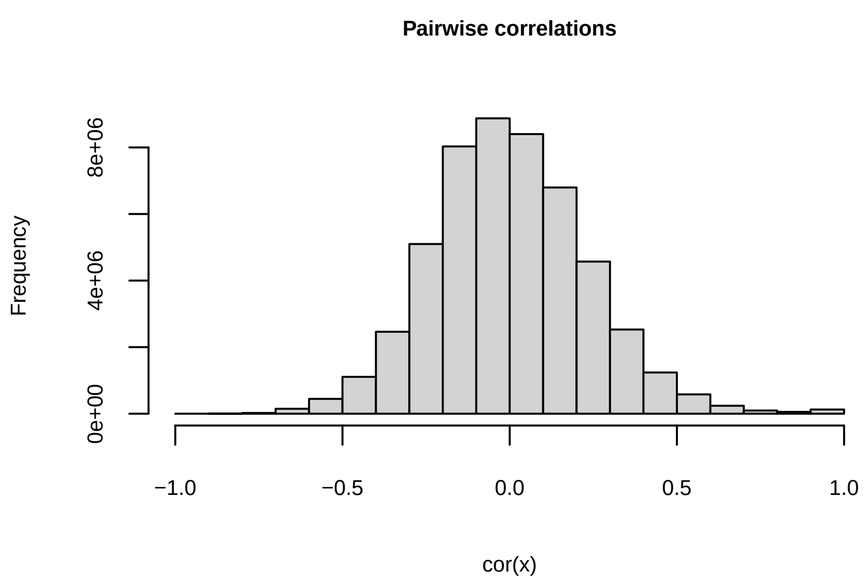
```r
x <- t(Westbc$assay)
table(Westbc$pheno$nodal.y) ## contains the response
```

```
##
## negative positive
##       25       24
```

```r
y <- Westbc$pheno$nodal.y
```

a)

```r
hist(cor(x), main = "Pairwise correlations", cex = .7,
     cex.lab = .7, cex.axis = .7, cex.main = .7)
```

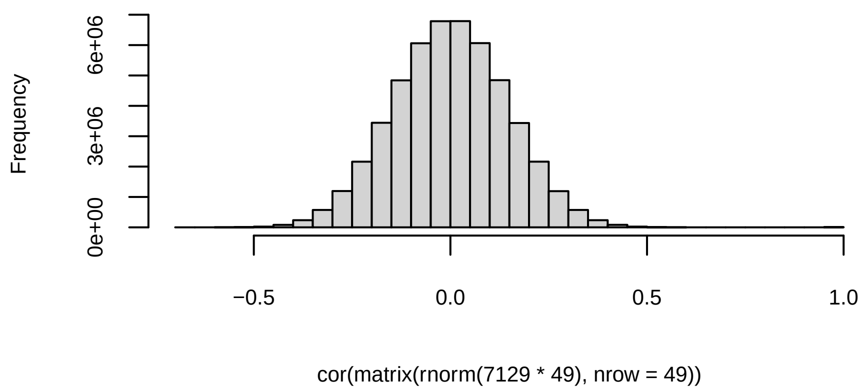


The pairwise correlations follow a normal distribution around 0. On the right, we see a somewhat elevated bar of correlations very close or equal to 1, but this is the diagonal of the correlation matrix, which we can ignore. The majority is close to 0, suggesting low multicollinearity.

Thus, lasso will probably not perform worse than elastic net or ridge, and may provide a sparse (i.e., few predictors selected) and stable model.

Even though some of the observed sample correlations are substantial (e.g., around 0.5 or -0.5), this is to be expected with a large values for p and small n, even if multicollinearity would be 0 in the population. For comparison, this is what sample correlations from a population with 0 multicollinearity look like:

```r
hist(cor(matrix(rnorm(7129*49), nrow = 49)), main = "")
```



Next, make a 80/20% train/test split:

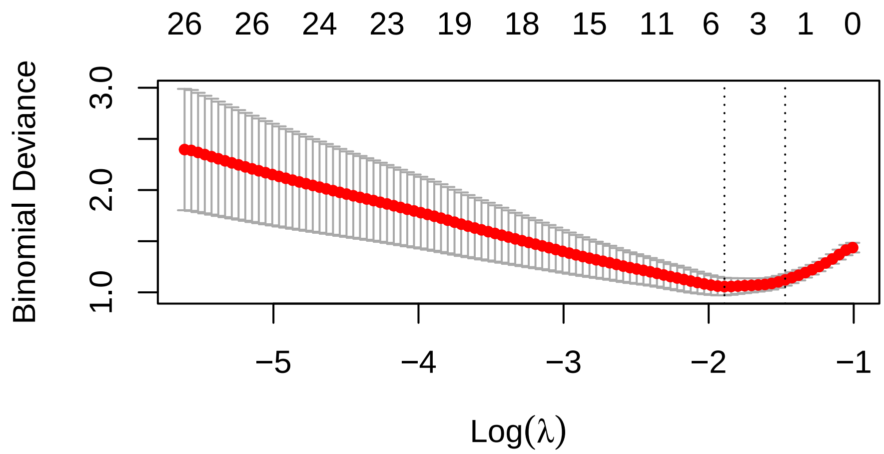
```r
set.seed(42)
n <- length(y)
train <- sample(1:n, size = n*.8)
```

b)

```r
library("glmnet")
## Lasso
set.seed(42)
mod_l <- cv.glmnet(x = x[train, ], y = y[train], family = "binomial")
mod_l
```

```
##
## Call:  cv.glmnet(x = x[train, ], y = y[train], family = "binomial")
##
## Measure: Binomial Deviance
##
##     Lambda Index Measure      SE Nonzero
## min 0.1509    20   1.056 0.08583       6
## 1se 0.2293    11   1.122 0.05600       1
```

```r
plot(mod_l)
```



The plot shows a convex curve for the CV error, which is reassuring. The minimum cross-validated binomial deviance with a λ value close to e^{-2}, which yield a model of between 3 and 6 predictors (the printed results show the exact values). The `lambda.1se` criterion yields an even sparser solution, with only 1 predictor.

c) Compute predictions and MCR for both criteria:

```r
preds_l_1se <- predict(mod_l, newx = x[-train, ], type = "response")
preds_l_min <- predict(mod_l, newx = x[-train, ], type = "response",
                       s = "lambda.min")
tab_l_1se <- prop.table(table(preds_l_1se > .5, y[-train]))
tab_l_min <- prop.table(table(preds_l_min > .5, y[-train]))
tab_l_1se
```

```
##
##          negative positive
##   FALSE       0.3      0.2
##   TRUE        0.0      0.5
```

```r
tab_l_min
```

```
##
##          negative positive
##   FALSE       0.3      0.2
##   TRUE        0.0      0.5
```

```r
sum(diag(tab_l_1se))
```

```
## [1] 0.8
```

```r
sum(diag(tab_l_min))
```

```
## [1] 0.8
```

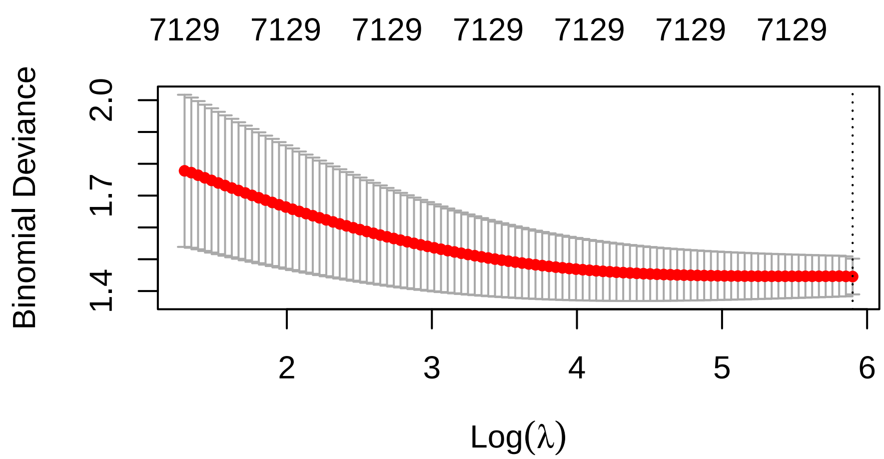
According to test MCR, there is no difference in performance of the two criteria for lasso regression.

d) Fit a ridge model and inspect the result:

```r
set.seed(42)
mod_r <- cv.glmnet(x = x[train, ], y = y[train], alpha = 0, family = "binomial")
mod_r
```

```
##
## Call:  cv.glmnet(x = x[train, ], y = y[train], alpha = 0, family = "binomial")
##
## Measure: Binomial Deviance
##
##     Lambda Index Measure      SE Nonzero
## min  365.2     1   1.446 0.05613    7129
## 1se  365.2     1   1.446 0.05613    7129
```

```r
plot(mod_r)
```



Ridge, as expected, selects all predictor variables. The two criteria yield the same value of λ. Note that the curve is not convex and continues to go down with increasing values of λ, suggesting that with ridge regression, the intercept-only model might simply be the best.

It may be interesting to inspect the distribution of the estimated coefficients. With so many predictor variables, one could e.g., use a histogram or do:

```r
table(cut(coef(mod_r)[,1], breaks = 10))
```

```
##
##     (-0.258,-0.232]     (-0.232,-0.206]      (-0.206,-0.18]      (-0.18,-0.155]
##                   1                   0                   0                   0
##     (-0.155,-0.129]     (-0.129,-0.103]    (-0.103,-0.0773]  (-0.0773,-0.0516]
##                   0                   0                   0                   0
##   (-0.0516,-0.0258] (-0.0258,0.000258]
##                   0                7129
```

All but one (negative) coefficient are very close to zero.

Compute MCR:

```r
preds_r_1se <- predict(mod_r, newx = x[-train, ], type = "response")
preds_r_min <- predict(mod_r, newx = x[-train, ], type = "response",
                       s = "lambda.min")
tab_r_1se <- prop.table(table(preds_r_1se > .5, y[-train]))
tab_r_min <- prop.table(table(preds_r_min > .5, y[-train]))
tab_r_1se
```

```
##
##         negative positive
##   FALSE      0.3      0.7
```

```r
tab_r_min
```

```
##
##         negative positive
##   FALSE      0.3      0.7
```

Ridge is outperformed by the lasso.

Note that ridge predicts the same label for all test observations here. It even predicts the same probability for all test observations:

```r
cbind(preds_r_min, preds_r_1se)
```

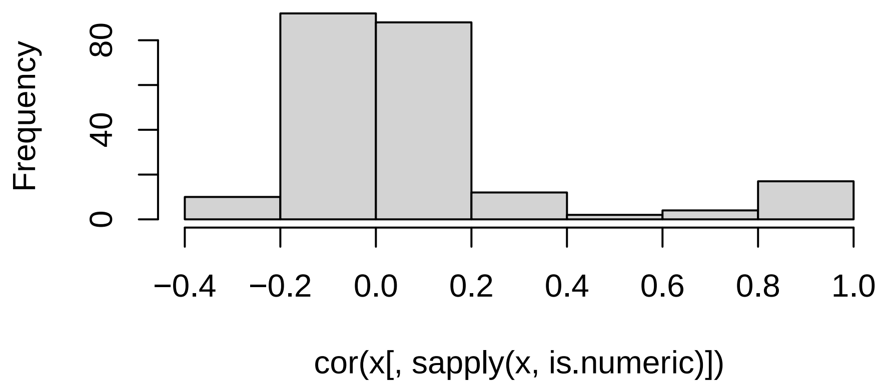
```
##       lambda.min lambda.1se
##  [1,]  0.4358974  0.4358974
##  [2,]  0.4358974  0.4358974
##  [3,]  0.4358974  0.4358974
##  [4,]  0.4358974  0.4358974
##  [5,]  0.4358974  0.4358974
##  [6,]  0.4358974  0.4358974
##  [7,]  0.4358974  0.4358974
##  [8,]  0.4358974  0.4358974
##  [9,]  0.4358974  0.4358974
## [10,]  0.4358974  0.4358974
```

Exercise 2: Predicting math grades

```r
student_full <- read.csv2("student-mat.csv")
```

a)

```r
x <- student_full[ , -which(names(student_full) == "G3")]
y <- student_full$G3
hist(cor(x[ , sapply(x, is.numeric)]), main = "")
```



- Given the not-too-large subset of predictors, best subset selection may work well.
- There is not too much multicollinearity, so lasso may do well. Some predictors do show strong positive correlations (.6 - .8). So possibly elastic net might do better than lasso.

b)

```r
library("leaps")

train_dat <- student_full[train, ]
test_dat <- student_full[-train, ]

## Best subset
model_BSS <- regsubsets(G3 ~ . , data = student_full, nvmax = 12,
                        method = "exhaustive")
sum <- summary(model_BSS)
which.max(sum$adjr2); which.max(sum$cp); which.min(sum$bic)
```

```
## [1] 12
```

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

```
## [1] 5
```

BIC, adjusted R^2 and Mallow's Cp all select differently sized models (inspect the summary `sum` to see which).

```r
coef(model_BSS, id = 5)
```

```
## (Intercept)         age      famrel    absences          G1          G2
## -0.07765375 -0.20167083  0.35724740  0.04365321  0.15794465  0.97804334
```

c)

```r
## Forward stepwise
model_Fstep <- regsubsets(G3 ~ . , data = student_full,
                        nvmax = 33, method = "forward")
sum <- summary(model_Fstep)
which.max(sum$adjr2); which.max(sum$cp); which.min(sum$bic)
```

```
## [1] 13
```

```
## [1] 33
```

```
## [1] 5
```

```r
## Backward stepwise
model_Bstep <- regsubsets(G3 ~ . , data = student_full,
                        nvmax = 33, method = "backward")
sum <- summary(model_Bstep)
which.max(sum$adjr2); which.max(sum$cp); which.min(sum$bic)
```

```
## [1] 13
```

```
## [1] 33
```

```
## [1] 5
```

The same models are selected between forward and backward selection, but the three different criteria select differently sized models.

d)

```r
coef(model_BSS, id = 5)
```

```
## (Intercept)         age      famrel    absences          G1          G2
## -0.07765375 -0.20167083  0.35724740  0.04365321  0.15794465  0.97804334
```

```r
coef(model_Fstep, id = 5)
```

```
## (Intercept)         age      famrel    absences          G1          G2
## -0.07765375 -0.20167083  0.35724740  0.04365321  0.15794465  0.97804334
```

```r
coef(model_Bstep, id = 5)
```

```
## (Intercept)         age      famrel   absences          G1          G2
## -0.07765375 -0.20167083  0.35724740  0.04365321  0.15794465  0.97804334
```

With the BIC criterion, all three subset selection methods (best subset, forward stepwise, backward stepwise) selected the same variables, and thus yielded identical final models.

The strongest predictor of math achievement at moment 3 seems to be math achievement at moment 2 (which is not very surprising).

e) To generate predictions for the test observations, we can create a design matrix with intercept and the selected predictor variables, then multiply those by the coefficients extracted from one of the models:

```r
coefs <- coef(model_BSS, id = 5)
x_test <- as.matrix(cbind(1, test_dat[c("age", "famrel", "absences", "G1", "G2")]))
L0_preds <- x_test %*% coefs
```

f) Compute mean squared error (MSE) for the test observations:

```r
MSE_L0 <- mean((L0_preds - test_dat$G3)^2)
MSE_L0
```

```
## [1] 4.429321
```

```r
MSE_max <- var(test_dat$G3) ## serves as a benchmark
MSE_max
```

```
## [1] 25.65129
```

```r
1 - MSE_L0 / MSE_max ## cross-validated R2
```

```
## [1] 0.8273256
```

g) Fit a Lasso, Ridge and Elastic Net (with $\alpha = 0.5$) to the training data:

```r
library("glmnet")
par(mfrow = c(1, 3))

## Data preparation
x <- model.matrix(G3 ~ . -1, train_dat)
x_test <- model.matrix(G3 ~ . -1, test_dat)
y <- train_dat$G3

## Model fitting
set.seed(42)
l_mod <- cv.glmnet(x, y, alpha = 1) ## l is for lasso
l_mod
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min 0.1835    34   3.458 0.617       8
## 1se 0.7409    19   3.997 0.717       1
```

```r
plot(l_mod)

set.seed(42)
r_mod <- cv.glmnet(x, y, alpha = 0) ## r is for ridge
r_mod
```

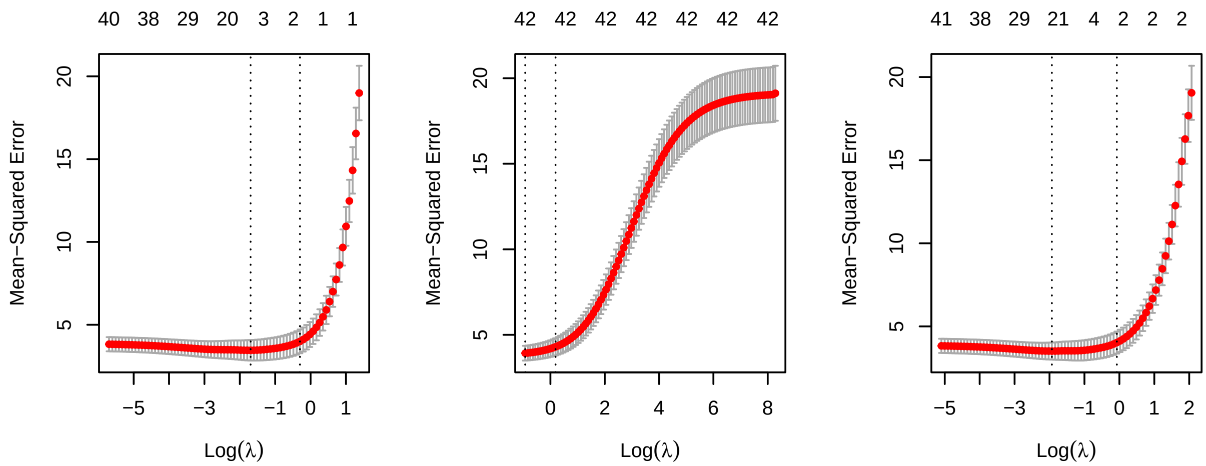
```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 0)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min 0.3954   100   3.926 0.4304      42
## 1se 1.2075    88   4.297 0.4998      42
```

```r
plot(r_mod)

set.seed(42)
e_mod <- cv.glmnet(x, y, alpha = .5) ## e is for elastic net
e_mod
```

```
##
## Call:  cv.glmnet(x = x, y = y, alpha = 0.5)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure     SE Nonzero
## min 0.1448    44   3.514 0.5132      22
## 1se 0.9307    24   4.025 0.6617       2
```

```r
plot(e_mod)
```



The curves for lasso and elastic net with $\alpha = .5$ (left and right plot) are convex, while that of ridge (middle plot) is not. The plot for ridge suggests that no penalization ($\lambda = 0$), might even do well.

How many variables were selected depended on the λ criterion used for selecting the final model:

With `lambda.min`, 8, 22 and 42 variables were retained with Lasso, Elastic Net and Ridge, respectively. (Note that `model.matrix` coded all factors as sets of dummy variables, thus increasing the number of predictor variables to 42 instead of 33).

With `lambda.1se`, 1, 2 and 42 variables were retained with Lasso, Elastic Net and Ridge, respectively.

h)

```
l_coefs <- coef(l_mod, s = "lambda.min")
l_coefs[l_coefs[ , 1] != 0, ]
```

```
##   (Intercept)           age  Fjobservices guardianmother     higheryes
##  -1.375061189  -0.012579948  -0.088060234   0.075355766   0.080350572
##        famrel      absences            G1             G2
##   0.151786387   0.004628783   0.069702792   0.986738604
```

```
e_coefs <- coef(e_mod, s = "lambda.min")
e_coefs[e_coefs[ , 1] != 0, ]
```

```
##   (Intercept)           age          Fedu  Mjobservices     Fjobother
##  -0.442981555  -0.129821244  -0.050720794   0.198921576   0.057010824
##  Fjobservices    reasonhome   reasonother guardianmother     failures
##  -0.254005929  -0.079120372   0.109566830   0.184854298  -0.097978405
##       paidyes activitiesyes    nurseryyes     higheryes   internetyes
##   0.094565940  -0.127442001  -0.113780592   0.404186683  -0.256272679
##   romanticyes        famrel         goout          Walc        health
##  -0.166071614   0.279118370   0.057398717   0.042904388   0.002637022
##      absences            G1            G2
##   0.024451458   0.149200666   0.930034024
```

In both models, `G2` is the strongest predictor, with much smaller effects for the other predictors.

i) Compute test MSE:

```
l_preds <- predict(l_mod, s = "lambda.min", newx = x_test)
r_preds <- predict(r_mod, s = "lambda.min", newx = x_test)
e_preds <- predict(e_mod, s = "lambda.min", newx = x_test)
MSE_l <- mean((l_preds - test_dat$G3)^2)
MSE_l
```

```
## [1] 4.955671
```

```
MSE_r <- mean((r_preds - test_dat$G3)^2)
MSE_r
```

```
## [1] 5.525793
```

```
MSE_e <- mean((e_preds - test_dat$G3)^2)
MSE_e
```

```
## [1] 4.93186
```

```
1 - MSE_l / MSE_max ## cross-validated R2
```

```
## [1] 0.8068062
```

```
1 - MSE_r / MSE_max ## cross-validated R2
```

```
## [1] 0.7845803
```

```
1 - MSE_e / MSE_max ## cross-validated R2
```

```
## [1] 0.8077344
```

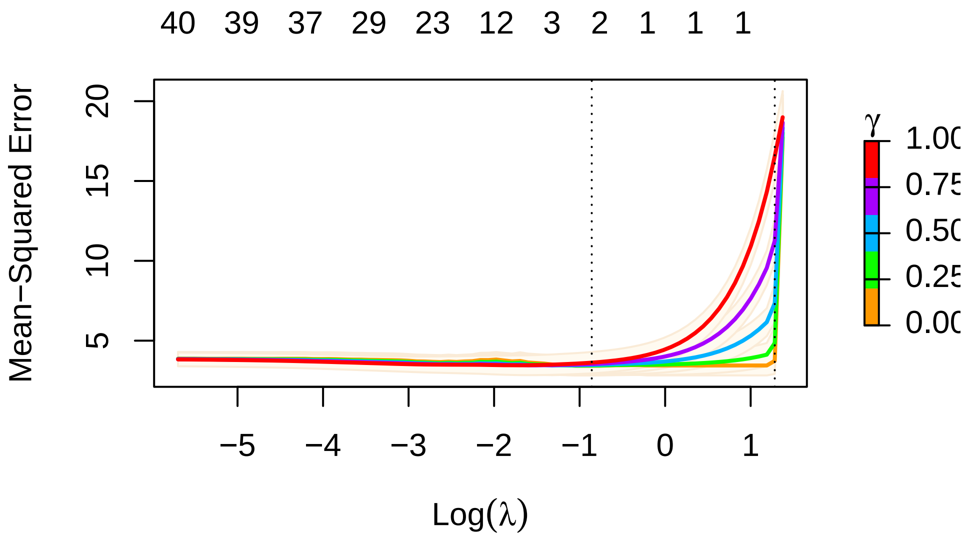
Among the shrinkage methods, elastic net performed best. But all three shrinkage methods were outperformed by the subset selection methods.

k) Fit the relaxed lasso:

```
set.seed(42)
rl_mod <- cv.glmnet(x, y, relax = TRUE)
rl_mod
```

```
##
## Call:  cv.glmnet(x = x, y = y, relax = TRUE)
##
## Measure: Mean-Squared Error
##
##     Gamma Index Lambda Index Measure     SE Nonzero
## min     0     1  0.424    25   3.429 0.6036       2
## 1se     0     1  3.603     2   3.736 0.8216       1
```

```r
plot(rl_mod)
```



The relaxed Lasso retained only 2 predictors, depending on the criterion used. Both optimal λ values had an optimal γ value of 0, indicating that variables are selected using the Lasso penalty, but no shrinkage should be performed when estimating the coefficients.

```r
rl_coefs <- coef(rl_mod, s = "lambda.min")
rl_coefs[rl_coefs[ , 1] != 0, ]
```

```
## (Intercept)          G1          G2
##   -1.510367    0.105165    1.010467
```

The relaxed lasso retained the G2 variable as the strongest predictor, followed by the G1 variable. It appears that past math performance is the best predictor of future math performance.

```r
rl_preds <- predict(rl_mod, s = "lambda.min", newx = x_test)
MSE_rl <- mean((rl_preds - test_dat$G3)^2)
MSE_rl
```

```
## [1] 4.86109
```

```r
1 - MSE_rl / MSE_max ## cross-validated R2
```

```
## [1] 0.8104933
```

The relaxed Lasso outperformed the other three shrinkage methods on the test data. It was, however, outperformed by the subset selection methods. Note that the relaxed Lasso could also be termed a selection (not shrinkage) method with $\gamma = 0$: It is then forward stepwise selection with variable entry determined by the Lasso.

Conclusion: Best subset selection provided best predictive accuracy, using 5 predictors, The relaxed Lasso shows a very good accuracy-complexity trade-off, using only 2 predictors and providing test set performance very close to best subset selection.