Code for slide deck on penalised regression and cross-validation

Benjamin Skov Kaas-Hansen

11/02/2020

Contents

Setup

```
packages <- c("plyr", "tidyr", "glmnet", "selectiveInference", "MASS", "tidyverse")
for (p in packages)
    library(p, character.only = TRUE)
knitr::opts_chunk$set(fig.align = "center")

theme_set(theme_minimal())

# Little helper to get the glmnet coefficients in nice tidy format
pretty_coefs <- function(coefs) { # coefs: the output from coef(fit_object, s = [value])
    enframe(coefs[, 1], "predictor", "coefficient") %>%
        filter(coefficient != 0) %>%
        arrange(desc(abs(coefficient)))
}
```

Lasso regression example: biopsies from breast cancer patients

Lasso regression

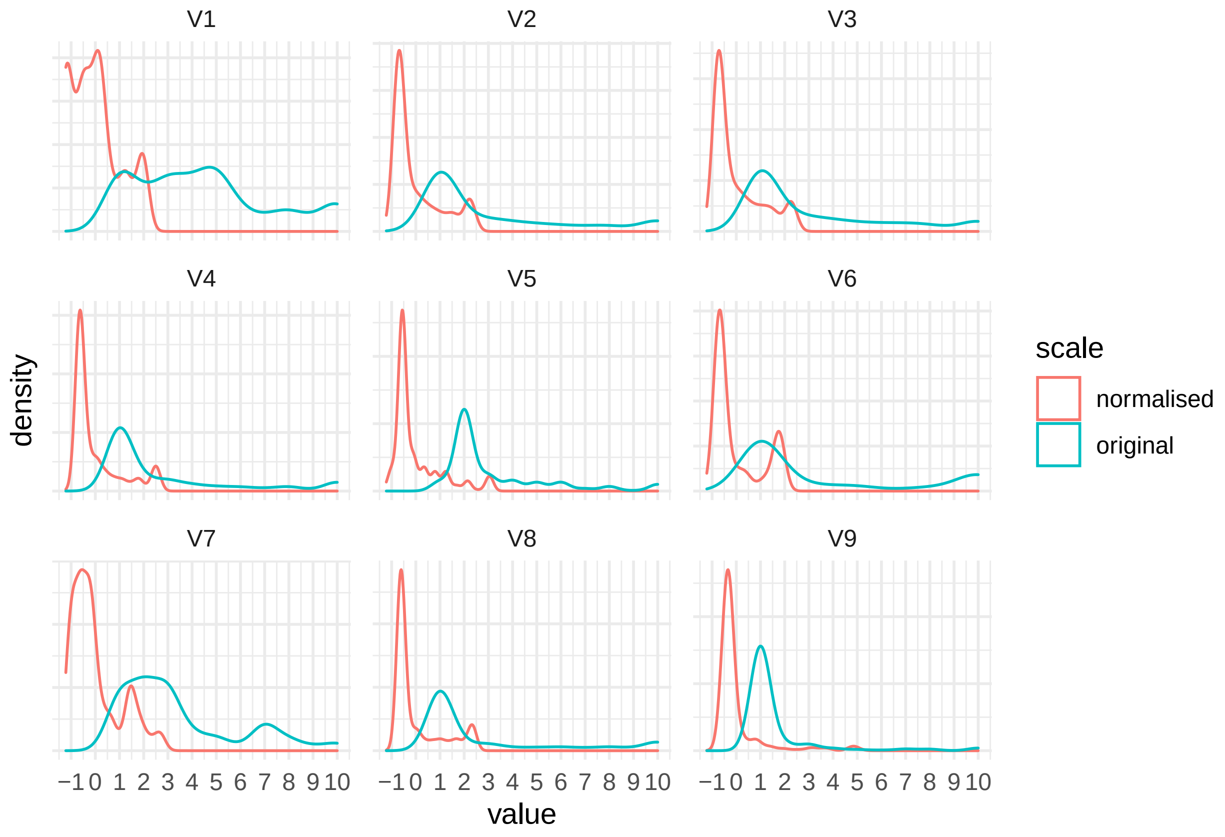
Let's look at the data

```
data(biopsy)
summary(biopsy) # NA's in V6; mean varies across variables but anyway somewhere around 2 and 4
```

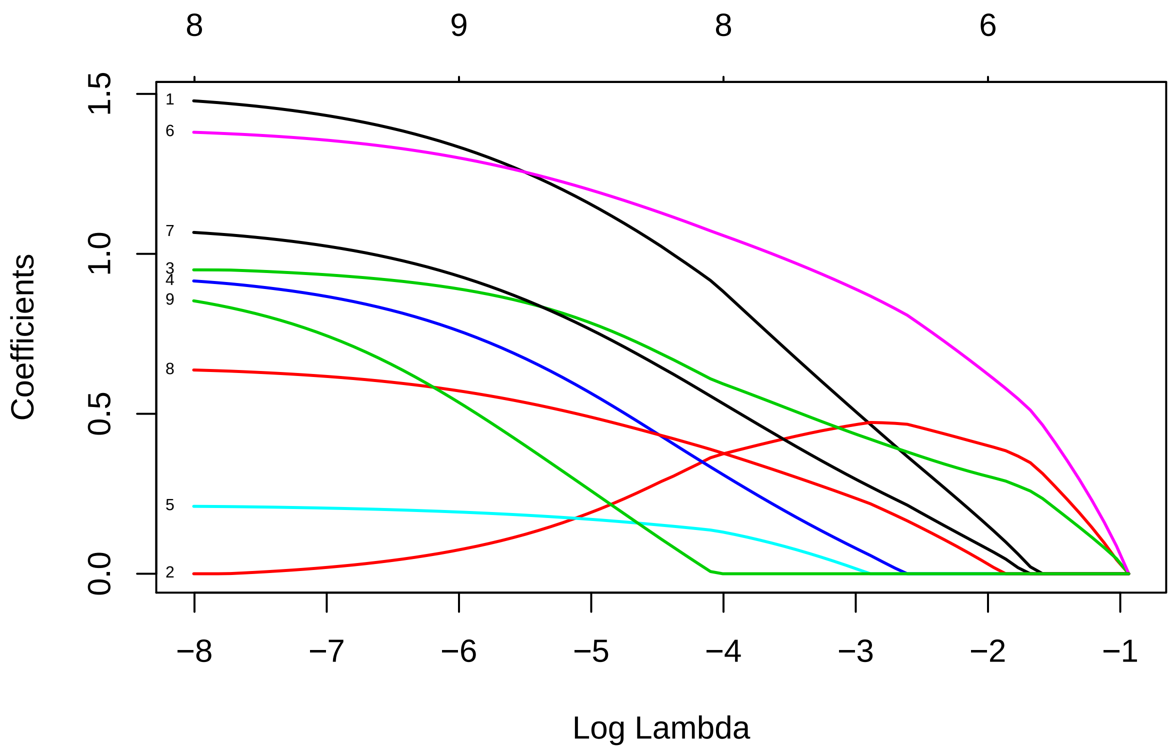
```
##       ID                  V1               V2               V3
##  Length:699         Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
##  Class :character   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 1.000
##  Mode  :character   Median : 4.000   Median : 1.000   Median : 1.000
##                     Mean   : 4.418   Mean   : 3.134   Mean   : 3.207
##                     3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 5.000
##                     Max.   :10.000   Max.   :10.000   Max.   :10.000
##
```

```
##        V4               V5               V6               V7        
##  Min.   : 1.000   Min.   : 1.000   Min.   : 1.000   Min.   : 1.000  
##  1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 2.000  
##  Median : 1.000   Median : 2.000   Median : 1.000   Median : 3.000  
##  Mean   : 2.807   Mean   : 3.216   Mean   : 3.545   Mean   : 3.438  
##  3rd Qu.: 4.000   3rd Qu.: 4.000   3rd Qu.: 6.000   3rd Qu.: 5.000  
##  Max.   :10.000   Max.   :10.000   Max.   :10.000   Max.   :10.000  
##                                    NA's   :16                       
##        V8               V9              class    
##  Min.   : 1.000   Min.   : 1.000   benign   :458  
##  1st Qu.: 1.000   1st Qu.: 1.000   malignant:241  
##  Median : 1.000   Median : 1.000                  
##  Mean   : 2.867   Mean   : 1.589                  
##  3rd Qu.: 4.000   3rd Qu.: 1.000                  
##  Max.   :10.000   Max.   :10.000                  
## 
```

```
biopsy_complete <- na.exclude(biopsy) # remove rows with any missing value
biopsy_predictors <- select(biopsy_complete, -ID, -class) %>%
    scale() # note attributes "remember" normlisation factors; useful for transforming test set

bind_rows(gather(as_tibble(biopsy_predictors), var, value) %>%
            mutate(scale = "normalised"),
          gather(select(biopsy_complete, -ID, -class), var, value) %>%
            mutate(scale = "original")) %>%
    ggplot(aes(x = value, colour = scale)) +
        geom_density(position = "identity") +
        scale_x_continuous(breaks = -2:10) +
        facet_wrap(~ var, scales = "free_y") +
        theme(axis.text.y = element_blank())
```



```
lasso_logreg <- glmnet(biopsy_predictors, biopsy_complete$class, family = "binomial")

# Coefficient profile plot (built-in: ugly but easy)
plot(lasso_logreg, xvar = "lambda", label = TRUE, lwd = 1.5)
```



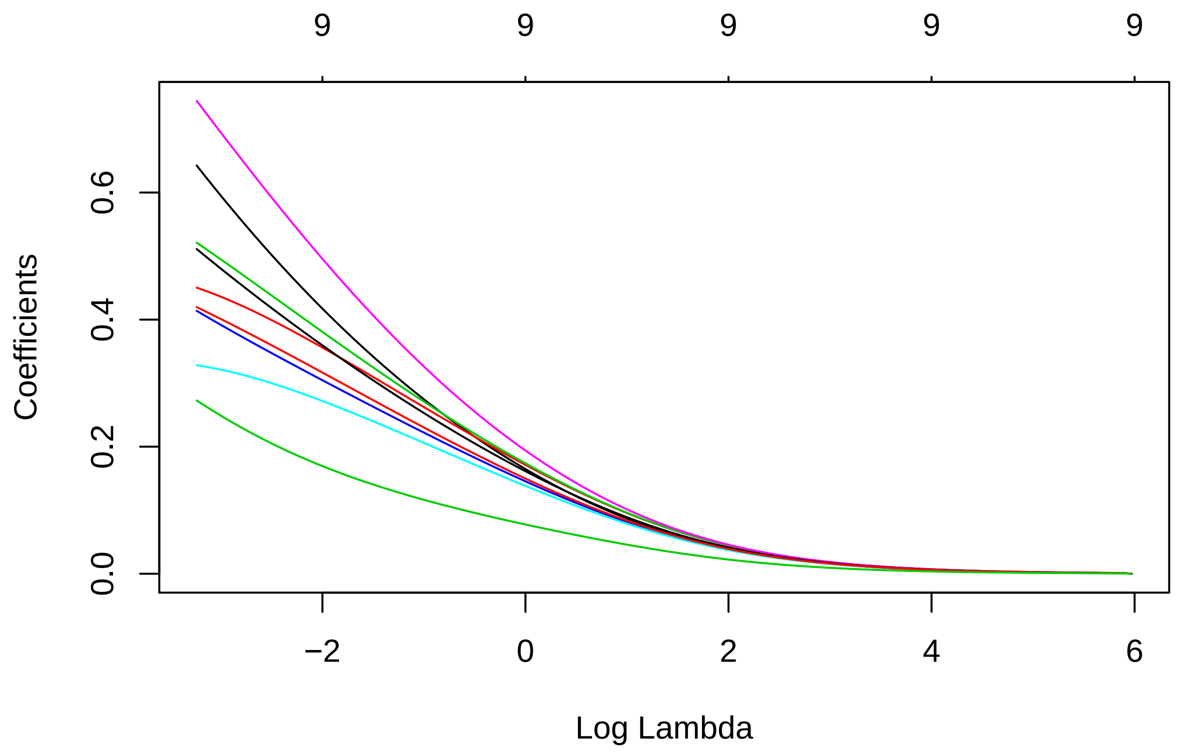
```
pretty_coefs(coef(lasso_logreg, s = exp(-1.2)))
```

```
## # A tibble: 4 x 2
##   predictor   coefficient
##   <chr>             <dbl>
## 1 (Intercept)      -0.642
## 2 V6                0.219
## 3 V2                0.138
## 4 V3                0.109
```

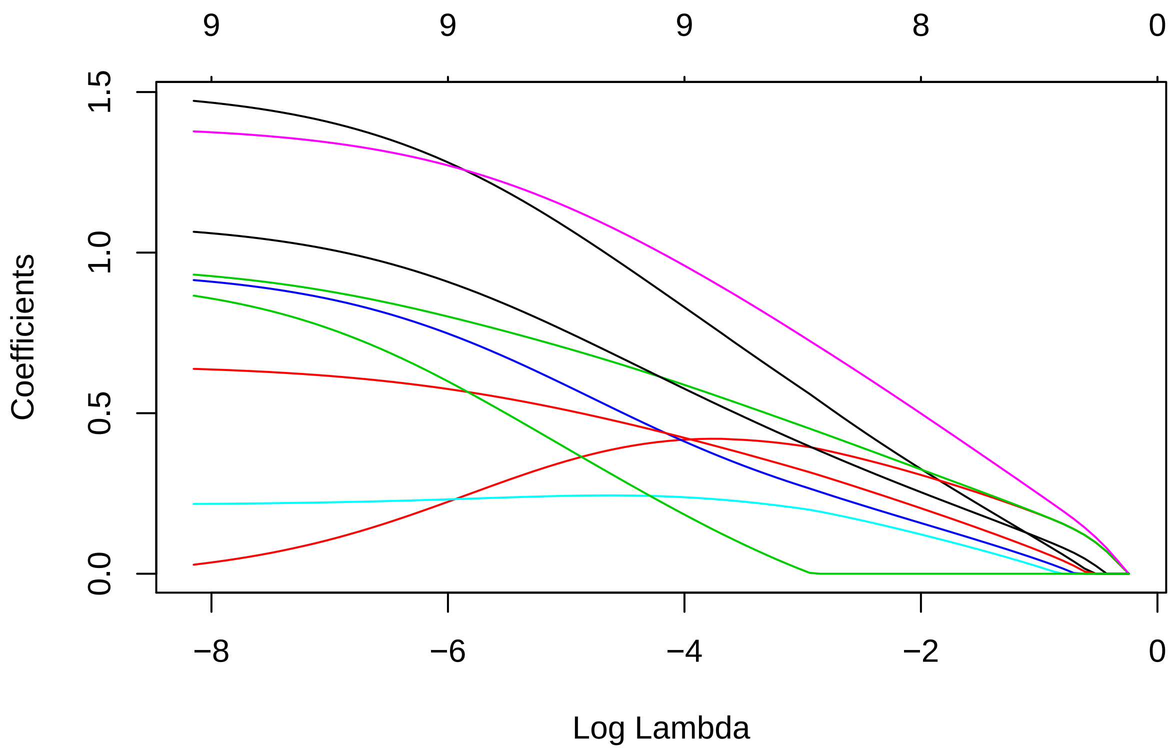
BACK TO PRESENTATION

Ridge and elastic net models

```
ridge_logreg <- update(lasso_logreg, alpha = 0)
plot(ridge_logreg, xvar = "lambda")
```



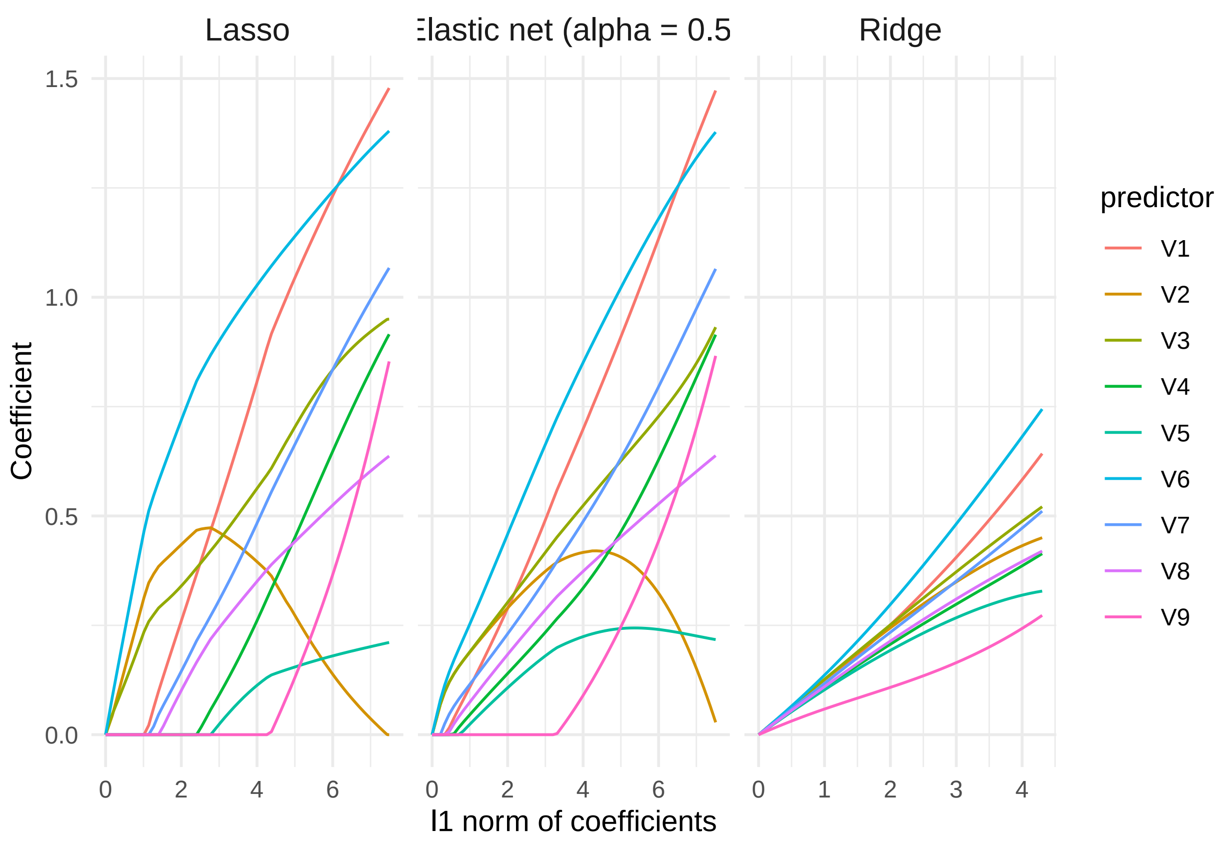
```
pretty_coefs(coef(ridge_logreg, s = exp(2))) # all shrunk but no real ranking or anything

## # A tibble: 10 x 2
##    predictor   coefficient
##    <chr>             <dbl>
##  1 (Intercept)     -0.630
##  2 V6               0.0457
##  3 V3               0.0448
##  4 V2               0.0447
##  5 V7               0.0414
##  6 V1               0.0395
##  7 V8               0.0391
##  8 V4               0.0383
##  9 V5               0.0373
## 10 V9               0.0224

elastic_logreg <- update(lasso_logreg, alpha = 0.5)
plot(elastic_logreg, xvar = "lambda")
```



```
pretty_coefs(coef(elastic_logreg, s = exp(-1.2))) # all shrunk AND "ranking"
```

```
## # A tibble: 9 x 2
##   predictor   coefficient
##   <chr>             <dbl>
## 1 (Intercept)     -0.719
## 2 V6               0.298
## 3 V3               0.215
## 4 V2               0.213
## 5 V1               0.148
## 6 V7               0.141
## 7 V8               0.0996
## 8 V4               0.0675
## 9 V5               0.0435
```

```
ldply(list(Lasso = lasso_logreg, Ridge = ridge_logreg, Elastic_net = elastic_logreg),
      function(.) with(., data.frame(as.matrix(t(beta)), x = apply(abs(beta), 2, sum))), .id = "mod") %
    gather(predictor, value, -x, -mod) %>%
    mutate(mod = factor(mod, levels = c("Lasso", "Elastic_net", "Ridge"),
                        labels = c("Lasso", "Elastic net (alpha = 0.5)", "Ridge"))) %>%
    ggplot(aes(x, value, colour = predictor)) +
        geom_line() +
        labs(x = "l1 norm of coefficients", y = "Coefficient") +
        facet_wrap(~ mod, scales = "free_x") +
        theme(strip.text = element_text(size = 12))
```



Over-fitting biopsy

It's a little data set but let's try and split into training and test sets. There is quite some over-fitting on the relative scale, but the absolute prediction error difference is minor.

```
set.seed(42) # reproducible stochastic code
train_idx <- runif(nrow(biopsy_complete)) <= 0.8 # not exactly indices
mean(train_idx) # fraction in training set
```

```
## [1] 0.806735
```

```
biopsy_train <- filter(biopsy_complete, train_idx)
biopsy_test <- filter(biopsy_complete, !train_idx)
all_equal(biopsy_complete, bind_rows(biopsy_train, biopsy_test)) # sanity check
```

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

```
# Alternative with actual indices (mostly a matter of taste)
set.seed(42)
train_idx <- which(runif(nrow(biopsy_complete)) <= 0.8)
length(train_idx) / nrow(biopsy_complete)
```

```
## [1] 0.806735
```

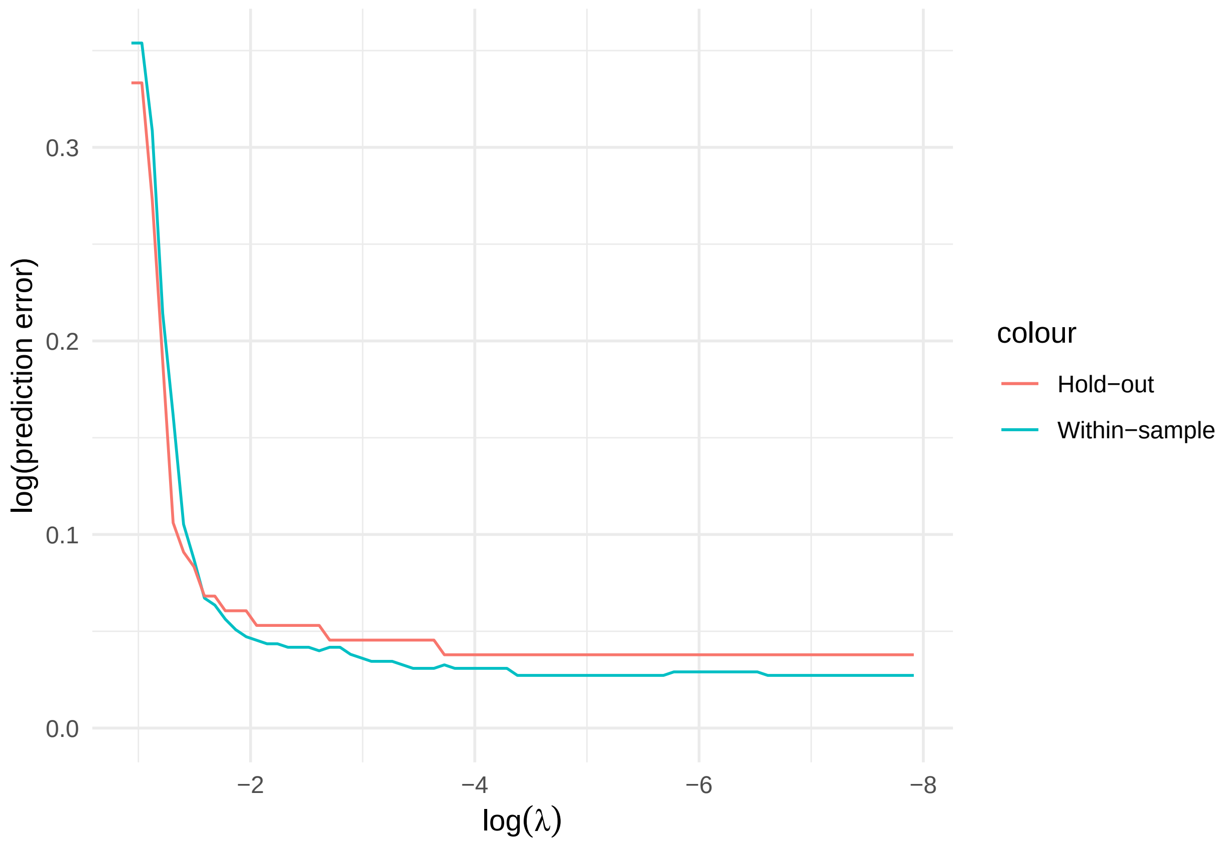
```
biopsy_train <- slice(biopsy_complete, train_idx)
biopsy_test <- slice(biopsy_complete, -train_idx)
all_equal(biopsy_complete, bind_rows(biopsy_train, biopsy_test)) # sanity check
```

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

```
# Normalise predictors and put them in matrix format
predictors_train <- select(biopsy_train, -ID, -class) %>%
    scale()
predictors_test <- select(biopsy_test, -ID, -class) %>%
    scale()

# Train model
lasso_biopsy_train <- glmnet(predictors_train, biopsy_train$class, family = "binomial")
D_train <- predict(lasso_biopsy_train, predictors_train, type = "class") %>%
    apply(2, function(.) mean(. != biopsy_train$class)) # s39 is the best-performing

D_test <- predict(lasso_biopsy_train, predictors_test, type = "class") %>%
    apply(2, function(.) mean(. != biopsy_test$class))

tibble(D_test = D_test[which.min(D_train)],
       D_train = D_train[which.min(D_train)],
       D_diff_abs = scales::percent(D_test - D_train),
       D_diff_rel = scales::percent((D_test - D_train) / D_train, big.mark = ","))
```

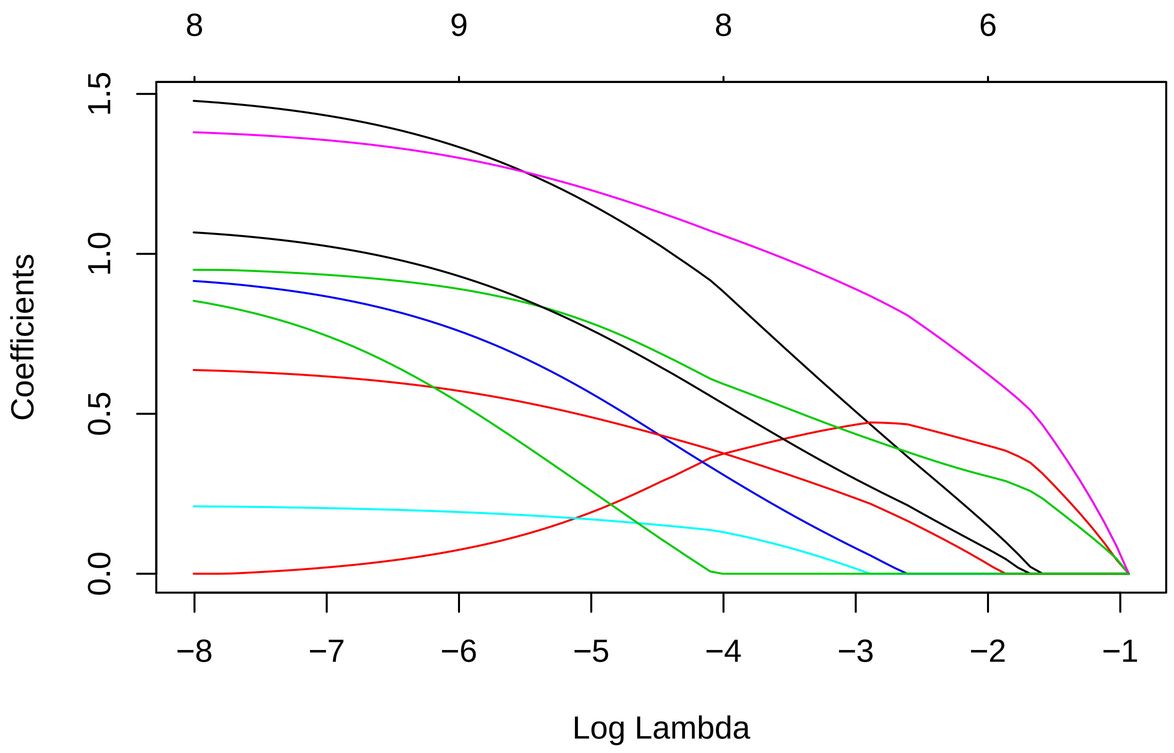
```
## # A tibble: 1 x 4
##   D_test D_train D_diff_abs D_diff_rel
##    <dbl>   <dbl> <chr>      <chr>
## 1 0.0379  0.0272 1%         39%
```

```
ggplot(mapping = aes(x = log(lasso_biopsy_train$lambda))) +
    coord_cartesian(ylim = c(0, NA)) +
    geom_line(aes(y = D_train, colour = "Within-sample")) +
    geom_line(aes(y = D_test, colour = "Hold-out")) +
    labs(y = "log(prediction error)", x = expression(log(lambda))) +
    scale_x_reverse()
```



Delassoing

```
plot(lasso_logreg, xvar = "lambda")
```



```
lambda <- exp(-4.5)
beta <- coef(lasso_logreg, x = biopsy_predictors, y = biopsy_complete$class,
             s = lambda/nrow(biopsy_complete), exact = TRUE)
delasso_fit <- fixedLassoInf(biopsy_predictors, as.numeric(biopsy_complete$class)-1, beta, lambda, "bin
                             alpha = 0.05)
```

```
## Warning in fixedLogitLassoInf(x, y, beta, lambda, alpha = alpha, type =
## type, : Solution beta does not satisfy the KKT conditions (to within specified
## tolerances)
```

```
delasso_fit
```

```
##
## Call:
## fixedLassoInf(x = biopsy_predictors, y = as.numeric(biopsy_complete$class) -
##     1, beta = beta, lambda = lambda, family = "binomial", alpha = 0.05)
##
## Testing results at lambda = 0.011, with alpha = 0.050
##
##  Var   Coef Z-score P-value LowConfPt UpConfPt LowTailArea UpTailArea
##    1  1.509   3.770   0.596    -9.749    2.005       0.025      0.024
##    2 -0.019  -0.030   0.989     0.857      Inf       0.025      0.000
##    3  0.964   1.399   0.962      -Inf    0.570       0.000      0.025
##    4  0.947   2.680   0.808   -17.165    1.237       0.025      0.024
##    5  0.215   0.617   0.863   -10.079    0.635       0.025      0.024
##    6  1.396   4.084   0.000     0.743    3.114       0.025      0.025
##    7  1.095   2.611   0.749   -14.329    1.537       0.025      0.025
##    8  0.650   1.889   0.805   -12.461    1.008       0.025      0.025
##    9  0.927   1.629   0.705   -10.952    1.724       0.025      0.025
```

```
##
## Note: coefficients shown are full regression coefficients
```

```
knitr::opts_chunk$set(include = FALSE)
```

Cross-validation Pima

Homegrown

Using packages

The `modelr`-package has some powerful functionalities for CV, LOOCV and bootstrapping.

Cross-validation biopsy

Use cross-validation to find best λ value. We found quite clear over-fitting above. Let's try to remedy this with cross-validation.

Test out-of-sample performance of the CV model

Resampling methods: jackknife and bootstrap

Sampling distribution

Jacknife

Exercise: cross-validation

1. LOO-CV error rate

2. Use proper cost function

3. Difference between error rates, and their interpretation

4. 10-fold CV

Exercise: penalised regression

1. + 2. Lasso regression

3. Why does it normally make sense to normalise predictors

4. Using CV to get reasonable estimate of λ

5. Obtain coefficients for "best" λ

6. Re-fit with correct family

Quite some difference between the sets of predictors kept:

7. As ridge regression

8. Get idea about sparse solution using ridge results?

9. Elastic net ($\alpha = 0.5$)

Ideally, one should do CV over α as well.

10. Delasso the results

11. Selective inference