Data leakage due to feature selection using the information of the classes.

Create simulated data

I create a dataset with 2000 rows - samples and 20.000 features. All data are drown from the normal distribution with `mean = 0` and `standard deviation = 1` using `rnorm()` function. Then, I assigned half of the samples to class 0 and the other half to class 1.

```
set.seed(42)
library(glmnet)
```

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

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

```
mydata_all <- matrix(
    data = rnorm(n = 2000 * 20000),
    nrow = 2000, ncol = 20000
)
rownames(mydata_all) <- paste0("s", 1:nrow(mydata_all))
colnames(mydata_all) <- paste0("g", 1:ncol(mydata_all))
classes <- rep(c(0, 1), times = nrow(mydata_all) / 2)
mydata_all[1:5, 1:5]
```

```
##            g1         g2         g3         g4          g5
## s1  1.3709584  0.2505781 -0.1418087  0.1728323 -0.05745257
## s2 -0.5646982 -0.2779240 -0.8138981 -1.2729637 -0.24903540
## s3  0.3631284 -1.7247357 -0.3255406 -0.8678954 -1.52416211
## s4  0.6328626 -2.0067049  0.3781574  0.6263211  0.46359103
## s5  0.4042683 -1.2918083 -1.9944854 -0.1056306 -1.18762073
```

Spit the data

I split the data to `test_data_external`, `test_y_external` and `mydata_x`, `mydata_y`. The first two correspond to an external dataset that will never been used in the model building. Thus we will be able to evaluate the **true** performance of our model. The latter will be used in model building.

```
test_data_external <- mydata_all[1501:2000, ]
test_y_external <- classes[1501:2000]
mydata_x <- mydata_all[1:1500, ]
mydata_y <- classes[1:1500]
```

Perform feature selection using the class information

I perform a `t.test()` to identify statistical significant features using the `mydata_x` data and keep the features that have a `p.value < 0.05`.

```
pvalues <- sapply(
    X = 1:ncol(mydata_x), function(i) {
        t.test(
            x = mydata_x[mydata_y == 0, i],
            y = mydata_x[mydata_y == 1, i]
        )$p.value
    }
)
```

```
mydata_x <- mydata_x[, pvalues < 0.05]
cat("Number of genes kept:", ncol(mydata_x), "\n")
```

```
## Number of genes kept: 1019
```

Split train - test

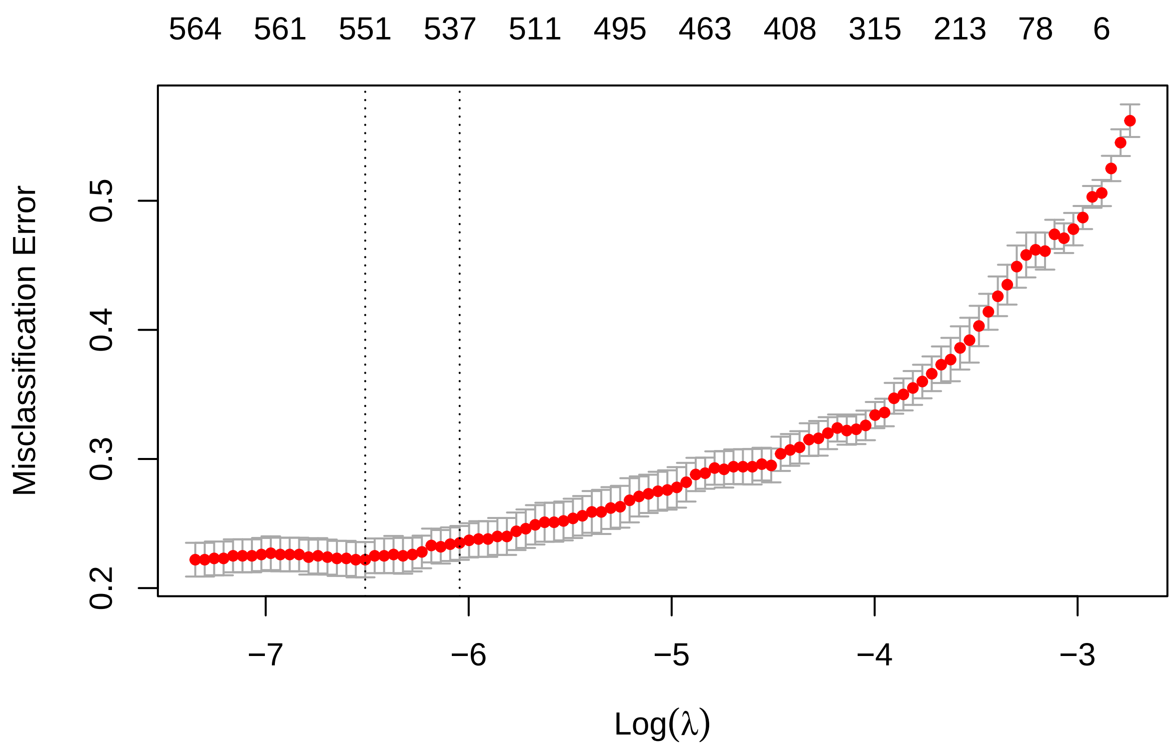
I split the data `mydata_x` in train `train_data` and test `test_data` sets to build a model and evaluate its performance.

```
train_data <- mydata_x[1:1000, ]
train_y <- mydata_y[1:1000]
test_data <- mydata_x[1001:1500, ]
test_y <- mydata_y[1001:1500]
```

Build ElasticNet model

I build the ElasticNet model and plot its cross validation error

```
fit_glmnet <- cv.glmnet(
    x = train_data, y = train_y, family = "binomial",
    type.measure = "class"
)
plot(fit_glmnet)
```



Note that the model has a misclassification accuracy of ~0.2 while we know that all the data were drown from the normal distribution with the same mean and standard deviation and thus they contain no signal!!

Test after feature selection

I test the model on the `test_data`. The set that I kept out after the feature selection.

```
preds <- predict(
    object = fit_glmnet,
    newx = test_data, s = "lambda.min",
    type = "class"
)
cat("Misclassification Error of test_data:", 1 - mean(preds == test_y), "\n")
```

```
## Misclassification Error of test_data: 0.238
```

The Misclassification Error in the this set that I selected is similar to the cross validation Misclassification Error, as expected.

Test in the TRUE external set

Finally we test the model on the external set that did not went through feature selection.

```
preds_external <- predict(
    object = fit_glmnet,
    newx = test_data_external[, pvalues < 0.05],
    s = "lambda.min",
    type = "class"
)
cat(
    "Misclassification Error of test_data_external:",
    1 - mean(preds_external == test_y_external), "\n"
)
```

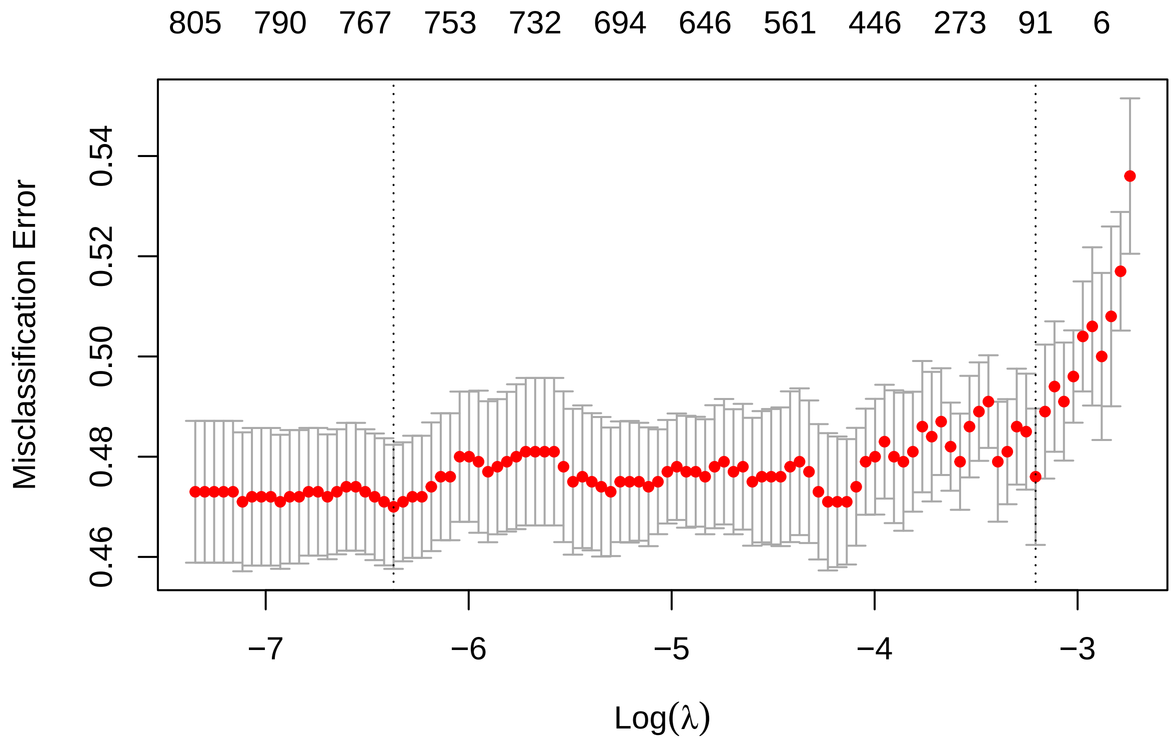
```
## Misclassification Error of test_data_external: 0.508
```

We observe that the **true** Misclassification Error is ~0.5 that is expected because the similated dataset has equal number of samples belonging to class 0 and class 1.

Rerun without feature selection

Rerunning without feature selection we observe that the three misclassification errors agree, the `cv error`, the error on the `test_data` and the error on the `external_data`.

```
# Split the data again
rm(train_data, test_data)
test_data_external <- mydata_all[1501:2000, ]
test_y_external <- classes[1501:2000]
mydata_x <- mydata_all[1:1500, ]
mydata_y <- classes[1:1500]
train_data <- mydata_x[1:1000, ]
train_y <- mydata_y[1:1000]
test_data <- mydata_x[1001:1500, ]
test_y <- mydata_y[1001:1500]
# Model fitting
fit_glmnet <- cv.glmnet(
    x = train_data, y = train_y, family = "binomial",
    type.measure = "class"
)
plot(fit_glmnet)
```




```r
# Prediction on test data
preds <- predict(
    object = fit_glmnet,
    newx = test_data, s = "lambda.min",
    type = "class"
)
cat("Misclassification Error of test_data:", 1 - mean(preds == test_y))
```

```
## Misclassification Error of test_data: 0.506
```

```r
# Prediction on external data
preds_external <- predict(
    object = fit_glmnet,
    newx = test_data_external,
    s = "lambda.min",
    type = "class"
)
cat(
    "Misclassification Error of test_data_external:",
    1 - mean(preds_external == test_y_external),
    "\n"
)
```

```
## Misclassification Error of test_data_external: 0.49
```