HarvardX: PH125.9x Capstone - Chose Your Own Project

Friederike David

2020-12-22

Contents

This project is part of the HarvardX's Data Science Capstone course, which is the last out of nine courses within the HarvardX's Data Science Professional Certificate.

Introduction

In medical and epidemiological research, the identification of biomarkers is a relevant task in many different contexts. For example, by determining disease states or environmental exposures from blood samples, valuable and reliable information on potentially confounding variables can be recovered for cohort-based studies. Knowledge on biomarkers of pathology and environment is also suitable to be translated into clinical applications and thus to contribute to diagnostic approaches and guidance of treatment.

Epigenetic modifications, such as DNA methylation at so called CpG sites, are promising biomarkers since they represent dynamic molecular signatures that regulate gene expression in response to internal and external stimuli. High-thoughput technologies such as DNA methylation microarrays allow measurement of the methylation status of CpG sites across the genome using large sets of corresponding oligonucleotide probes.

In this project, smoking status - an example for a common environmental factor that affects epigenetic profiles - will be predicted from DNA methylation array data of blood cells using machine learning methods. Briefly, after retrieval and short description of the dataset, it will be split into a training and a test set. Using only the training set, feature selection will then be performed on the available probes and different classification models will be trained. Finally, performance of these models will be evaluated in the hold-out test set.

Methods

Data input

For this project, the publicly available NCBI GEO dataset `GSE53045` contributed by Robert Philibert is used. It contains preprocessed DNA methylation data as well as corresponding sample-level group information on smoking status (`Smoker` or `Control`). The dataset is automatically retrieved from GEO via the Bioconductor package `GEOquery`.

```
#### Data input ####

# load data from GEO
geo_id <- "GSE53045"
geo <- getGEO(geo_id, destdir = here("data"))
```

```
#### Data preparation ####

# extract methylation data
methyl <- geo[[1]]@assayData$exprs %>% t()
glimpse(methyl)
```

```
##  num [1:111, 1:485577] 0.595 0.538 0.626 0.599 0.7 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:111] "GSM1280914" "GSM1280915" "GSM1280916" "GSM1280917" ...
##   ..$ : chr [1:485577] "cg00000029" "cg00000108" "cg00000109" "cg00000165" ...
```

```
# extract relevant phenotypic data
pheno <- pData(geo[[1]]) %>%
  select(geo_accession, disease_state = `disease state:ch1`) %>%
  mutate(disease_state = factor(disease_state, levels = c("Smoker", "Control")))
str(pheno)
```

```
## 'data.frame':    111 obs. of  2 variables:
##  $ geo_accession: chr  "GSM1280914" "GSM1280915" "GSM1280916" "GSM1280917" ...
##  $ disease_state: Factor w/ 2 levels "Smoker","Control": 1 1 1 1 1 1 1 1 1 1 ...
```

The dataset contains DNA methylation data of 111 PBMC (peripheral blood mononuclear cell) samples from 50 smokers and 61 non-smokers, measured with the Illumina Infinium 450k Human Methylation Beadchip across 485577 CpG sites/probes.

```
# remove probes with missing values
probes_remove <- apply(methyl, 2, function(x) any(is.na(x)))
methyl <- methyl[, !probes_remove]
```

After removal of probes that contain missing values, 428651 probes are left as input variables to distinguish smokers from non-smokers.

For model training and subsequent testing, the complete dataset is split into a training (80%) and a test set (20%). During model development, only the training set will be used, while the test set will be exclusively used to determine model performance. The split sizes are chosen in a way that maximizes the size of the training set to achieve a good model performance while keeping about 20 samples, i.e. about 10 smokers and 10 non-smokers in the test set to allow for meaningful performance metrics.

```r
# split sample into distinct sets for model development and testing
if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
test_index <- createDataPartition(y = pheno$disease_state, p = .2, list = F)

methyl_train <- methyl[-test_index, ]
methyl_test <- methyl[test_index, ]

pheno_train <- pheno[-test_index, ]
pheno_test <- pheno[test_index, ]
```

Data exploration

For a brief overview of this high-dimensional dataset, dimensionality reduction via principal component analysis (PCA) is performed on the training set. The first three principal components are shown below.

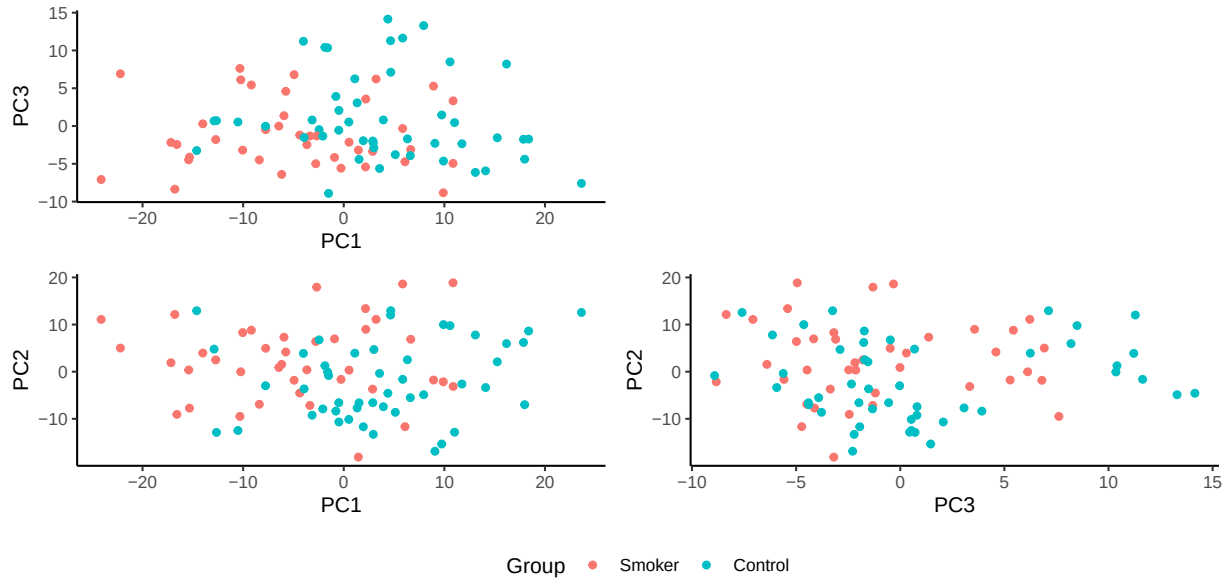

Figure 1: Sample distribution in principal component space

While there is no perfect separation of smokers from non-smokers within the first three principal components, it is clearly visible that the distribution of samples differs between groups (Figure 1). This is a solid basis for classification via machine learning.

Feature selection

With methylation data for 428651 probes, the number of variables that are available for model development by far exceeds the number of samples. Since model fitting with all available variables as input is not feasible with standard computing resources, feature selection is performed by testing for differential methylation between smokers and non-smokers. For this, the dedicated Bioconductor package `minfi` is used. The top 100 probes with most significant methylation differences between groups are then used as features in the machine learning models.

To avoid overfitting, an additional subset is split from the training subset, using 40% of the training set for differential methylation testing and the remaining 60% of the training set for model fitting.

```r
# split training set into distinct sets for feature selection and model fitting
if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
feat_index <- createDataPartition(y = pheno_train$disease_state,
                                  p = .4, list = FALSE)

methyl_feat <- methyl_train[feat_index, ]
methyl_dev <- methyl_train[-feat_index, ]

pheno_feat <- pheno_train[feat_index, ]
pheno_dev <- pheno_train[-feat_index, ]

# perform differential methylation testing on the feature selection subset
methyl_diff <- dmpFinder(dat = t(methyl_feat), pheno = pheno_feat$disease_state,
                         qCutoff = 0.05, type = "categorical")

# selected features: 100 probes with most significant differential methylation
feat_sel <- slice_min(methyl_diff, order_by = qval, n = 100) %>% rownames()
feat_sel_index <- colnames(methyl_feat) %in% feat_sel

# filter training and test set for selected features
methyl_dev <- methyl_dev[, feat_sel_index]
methyl_test <- methyl_test[, feat_sel_index]
```

Model development

For the given classification task, a number of different modeling approaches is employed. As most basic model, logistic regression (`glm`) is used. Considering that the number of selected features still exceeds the number of samples, penalized regression is further used to model smoking status from DNA methylation. In particular, lasso regression (`lasso`), ridge regression (`ridge`), and elastic net regression (`elasticnet`) as mixture of both is used. While in lasso regression the coefficients for some variables are forced to be 0, thus excluding these variables from the model, in ridge regression the coefficients cannot be shrunken to 0, which means that all variables are included in the model.

Additional models are fitted based on linear discriminant analysis (`lda`), k-nearest neighbors (`knn`), random forest (`rf`), and support vector machines with linear kernel (`svmLinear`). Finally, an ensemble prediction (`ensemble`) is generated from all models that have a training accuracy above 0.8.

In model development, a five-fold cross validation with 80:20 splits is used to optimize tunable parameters.

```r
# train control: 5-fold cross validation
cv <- trainControl(method = "cv", number = 5, p = .2,
                   classProbs = TRUE, savePredictions = TRUE)

# ridge regression (alpha = 0)
if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
fit.ridge <- train(x = methyl_dev, y = pheno_dev$disease_state,
                   method = "glmnet", trControl = cv,
                   tuneGrid = data.frame(alpha = 0,
                                         lambda = seq(0.0001, 1, length = 100)))

# lasso regression (alpha = 1)
if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
fit.lasso <- train(x = methyl_dev, y = pheno_dev$disease_state,
```

```
                  method = "glmnet", trControl = cv,
                  tuneGrid = data.frame(alpha = 1,
                                        lambda = seq(0.0001, 1, length = 100)))

# elastic net regression (0 < alpha < 1)
if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
fit.elasticnet <- train(x = methyl_dev, y = pheno_dev$disease_state,
                        method = "glmnet", trControl = cv,
                        tuneGrid = expand.grid(alpha = seq(0.1, 0.9, 0.1),
                                               lambda = seq(0.0001, 1, length = 100)))

# additional models
models.add <- c("glm", "lda", "knn", "rf", "svmLinear")

tunegrids.add <- list(glm = NULL,
                      lda = NULL,
                      svmLinear = data.frame(C = seq(0, 2, length = 20)),
                      rf = data.frame(mtry = 1:7),
                      knn = data.frame(k = seq(3, 51, 2)))

fits.add <- lapply(models.add, function(model){
  if (rver < 360) {set.seed(42)} else {set.seed(42, sample.kind = "Rounding")}
  train(x = methyl_dev, y = pheno_dev$disease_state, method = model,
        trControl = cv, tuneGrid = tunegrids.add[[model]])
})

# combine all models into single object
fits <- c(list(fit.ridge, fit.lasso, fit.elasticnet), fits.add) %>%
  set_names(c("ridge", "lasso", "elasticnet", models.add))

# select models for ensemble prediction
training_accuracies <- sapply(fits, function(fit) mean(fit$resample$Accuracy))
models_keep <- training_accuracies >= 0.8
```

Results and discussion

Model performance is evaluated within the hold-out test set based on prediction accuracy, recall, precision, and F1 score. The F1 score as harmonic mean of recall and precision is considered to be the most important metric.

```
# function to calculate model metrics: accuracy, recall, precision, F1 score
get_metrics <- function(prediction, truth = pheno_test$disease_state) {
  prediction <- factor(prediction, levels = c("Smoker", "Control"))
  data.frame(F1_score   = F_meas(prediction, truth),
             Accuracy  = mean(prediction == truth),
             Recall    = recall(prediction, truth),
             Precision = precision(prediction, truth))
}

# predictions on hold-out testing set for all individual models
pred <- sapply(fits, predict, methyl_test)
```

```
# ensemble prediction only with models that have a training accuracy > 0.8
ensemble <- apply(pred[, models_keep], 1,
                  function(i) names(sort(table(i), decreasing = T))[1])

# combine single-model predictions with ensemble prediction
pred <- cbind(pred, "ensemble" = ensemble)

# results table ordered by F1 score
res <- apply(pred, 2, get_metrics) %>% bind_rows(.id = "Model") %>% arrange(-F1_score)
```

Table 1: Performance metrics of predictive models

Model	F1_score	Accuracy	Recall	Precision
ridge	0.947	0.957	0.9	1.0
elasticnet	0.947	0.957	0.9	1.0
knn	0.947	0.957	0.9	1.0
svmLinear	0.947	0.957	0.9	1.0
ensemble	0.947	0.957	0.9	1.0
rf	0.900	0.913	0.9	0.9
lasso	0.889	0.913	0.8	1.0
glm	0.889	0.913	0.8	1.0
lda	0.889	0.913	0.8	1.0

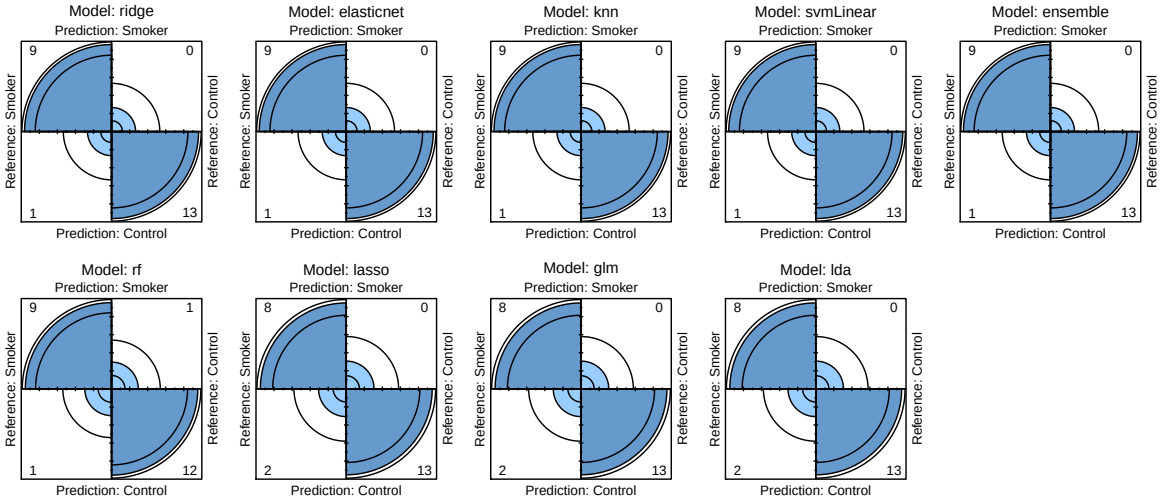

Figure 2: Visualization of confusion matrices for fitted models

Within this dataset of DNA methylation, smoking status can be almost perfectly modeled from the top 100 differentially methylated probes (Table 1). Both the `ridge` and `elasticnet` regularized regression models as well as the `knn`,`svmLinear`, and `ensemble` approach achieve an F1 score of 0.95 with one false negative classification. The remaining models each misclassify two samples, with one false positive and one false negative in the random forest (`rf`) approach and two false negatives in the three other approaches (Figure 2).

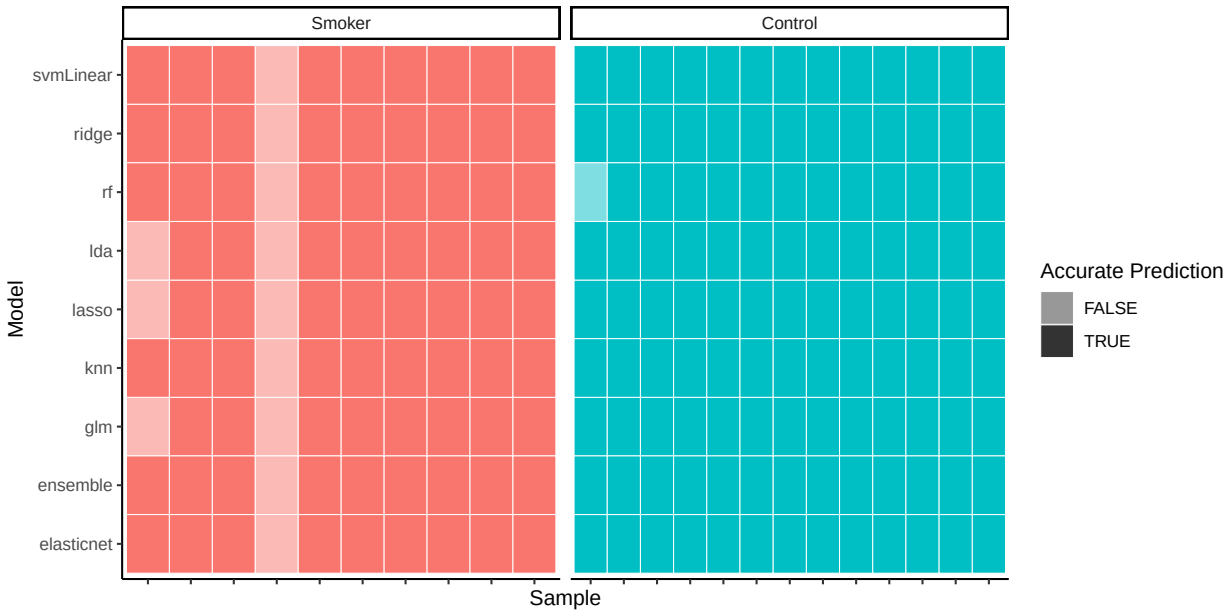

Figure 3: Concordance of predictions between models

Interestingly, all false negative misclassifications fall into the same samples, with one sample misclassified across all models and a second sample misclassified by three models (Figure 3). This could indicate a systematic problem with these samples, which may be linked to technical reasons (too small training set, too few probes as input variables) but could also be due to biological reasons or incorrect phenotypic data in case of the sample misclassified across all models. A biological reason for misclassification could be the binary setup of the classifier, since the impact of smoking on epigenetic profiles will most likely also depend on the amount and duration of smoke exposure. This could lead to a higher similarity of individuals that rarely smoke with non-smokers compared to regular smokers, resulting in the misclassification of such samples.

Conclusion

In this project, smoking status was modeled from DNA methylation profiles to show how epigenetic profiles from easily accessible biomaterial such as blood samples can be used as a biomarkers for e.g. environmental factors. After feature selection by differential methylation testing, 100 probes were used to build nine different machine learning models including an ensemble model. Model evaluation in the hold-out testing set showed a very good performance for all tested models, with only one or two misclassified samples per model.

Overall, this proof-of-concept project demonstrates that epigenetic profiles are suitable for predictive modeling of environmental factors such as smoking. Thus, they represent promising biomarkers for medical studies and eventually for clinical applications. Future studies on larger datasets with more extensive annotations are required to develop reliable and robust models for different environmental factors of interest.

Session info

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
```

```
## Matrix products: default
##
## Random number generation:
##  RNG:     Mersenne-Twister
##  Normal:  Inversion
##  Sample:  Rounding
##
## locale:
## [1] LC_COLLATE=German_Germany.1252  LC_CTYPE=German_Germany.1252
## [3] LC_MONETARY=German_Germany.1252 LC_NUMERIC=C
## [5] LC_TIME=German_Germany.1252
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] minfi_1.32.0                bumphunter_1.28.0
##  [3] locfit_1.5-9.4              iterators_1.0.12
##  [5] foreach_1.5.0               Biostrings_2.54.0
##  [7] XVector_0.26.0              SummarizedExperiment_1.16.1
##  [9] DelayedArray_0.12.3         BiocParallel_1.20.1
## [11] matrixStats_0.56.0          GenomicRanges_1.38.0
## [13] GenomeInfoDb_1.22.1         IRanges_2.20.2
## [15] S4Vectors_0.24.4            GEOquery_2.54.1
## [17] Biobase_2.46.0              BiocGenerics_0.32.0
## [19] patchwork_1.0.1             MLeval_0.3
## [21] randomForest_4.6-14         kernlab_0.9-29
## [23] glmnet_4.0-2                Matrix_1.2-17
## [25] caret_6.0-86                lattice_0.20-38
## [27] BiocManager_1.30.10         data.table_1.12.8
## [29] here_0.1                    forcats_0.5.0
## [31] stringr_1.4.0               dplyr_1.0.0
## [33] purrr_0.3.4                 readr_1.3.1
## [35] tidyr_1.1.0                 tibble_3.0.3
## [37] ggplot2_3.3.2               tidyverse_1.3.0
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1             backports_1.1.7          BiocFileCache_1.10.2
##   [4] plyr_1.8.6               splines_3.6.1            digest_0.6.25
##   [7] htmltools_0.5.0          fansi_0.4.1              magrittr_1.5
##  [10] memoise_1.1.0            limma_3.42.2             annotate_1.64.0
##  [13] recipes_0.1.13           modelr_0.1.8             gower_0.2.2
##  [16] siggenes_1.60.0          askpass_1.1              prettyunits_1.1.1
##  [19] colorspace_1.4-1         blob_1.2.1               rvest_0.3.6
##  [22] rappdirs_0.3.1           haven_2.3.1              xfun_0.15
##  [25] crayon_1.3.4             RCurl_1.98-1.2           jsonlite_1.7.1
##  [28] genefilter_1.68.0        survival_2.44-1.1        glue_1.4.1
##  [31] gtable_0.3.0             ipred_0.9-9              zlibbioc_1.32.0
##  [34] Rhdf5lib_1.8.0           shape_1.4.5              HDF5Array_1.14.4
##  [37] scales_1.1.1             DBI_1.1.0                rngtools_1.5
##  [40] Rcpp_1.0.5               xtable_1.8-4             progress_1.2.2
##  [43] mclust_5.4.7             bit_4.0.4                preprocessCore_1.48.0
##  [46] lava_1.6.7               prodlim_2019.11.13       httr_1.4.2
```

```
##  [49] RColorBrewer_1.1-2       ellipsis_0.3.1           farver_2.0.3
##  [52] pkgconfig_2.0.3          reshape_0.8.8            XML_3.99-0.3
##  [55] nnet_7.3-12              dbplyr_1.4.4             labeling_0.3
##  [58] tidyselect_1.1.0         rlang_0.4.7              reshape2_1.4.4
##  [61] AnnotationDbi_1.48.0     munsell_0.5.0            cellranger_1.1.0
##  [64] tools_3.6.1              cli_2.0.2                generics_0.0.2
##  [67] RSQLite_2.2.1            broom_0.7.0              evaluate_0.14
##  [70] yaml_2.2.1               ModelMetrics_1.2.2.2     knitr_1.29
##  [73] bit64_4.0.5              fs_1.4.2                 beanplot_1.2
##  [76] scrime_1.3.5             nlme_3.1-140             doRNG_1.8.2
##  [79] nor1mix_1.3-0            xml2_1.3.2               biomaRt_2.42.1
##  [82] compiler_3.6.1           rstudioapi_0.11          curl_4.3
##  [85] e1071_1.7-4              reprex_0.3.0             stringi_1.4.6
##  [88] highr_0.8                GenomicFeatures_1.38.2   multtest_2.42.0
##  [91] vctrs_0.3.2              pillar_1.4.6             lifecycle_0.2.0
##  [94] bitops_1.0-6             rtracklayer_1.46.0       R6_2.4.1
##  [97] codetools_0.2-16         MASS_7.3-51.4            assertthat_0.2.1
## [100] rhdf5_2.30.1             openssl_1.4.2            rprojroot_1.3-2
## [103] withr_2.2.0              GenomicAlignments_1.22.1 Rsamtools_2.2.3
## [106] GenomeInfoDbData_1.2.2   hms_0.5.3                quadprog_1.5-8
## [109] grid_3.6.1               rpart_4.1-15             timeDate_3043.102
## [112] base64_2.0               class_7.3-15             rmarkdown_2.3
## [115] DelayedMatrixStats_1.8.0 illuminaio_0.28.0        pROC_1.16.2
## [118] lubridate_1.7.9
```