# Computer lab block 2

**Instructions**

* Create a report to the lab solutions in PDF.
* Be concise and do not include unnecessary printouts and figures produced by the software and not required in the assignments.
* **Include all your codes as an appendix into your report.**
* **Use set.seed(12345) for every piece of code that contains randomness**
* A typical lab report should 2-4 pages of text plus some amount of figures plus appendix with codes.
* The lab report should be submitted via LISAM before the deadline.

### Assignment 1. Ensemble methods

Your task is to learn some random forests using the function **randomForest** from the R package **randomForest**. The training data is produced by running the following R code:

x1<-runif(100)

x2<-runif(100)

trdata<-cbind(x1,x2)

y<-as.numeric(x1<x2)

trlabels<-as.factor(y)

The task is therefore classifying Y from X1 and X2, where Y is binary and X1 and X2 continuous. You should learn a random forest with 1, 10 and 100 trees, which you can do by setting the argument **ntree** to the appropriate value. Use **nodesize = 25** and **keep.forest = TRUE**. The latter saves the random forest learned. You need it because you should also compute the misclassification error in the following test dataset (use the function **predict** for this purpose):

set.seed(1234)

x1<-runif(1000)

x2<-runif(1000)

tedata<-cbind(x1,x2)

y<-as.numeric(x1<x2)

telabels<-as.factor(y)

plot(x1,x2,col=(y+1))

1. Repeat the procedure above for 1000 training datasets of size 100 and report the mean and variance of the misclassification errors. In other words, create 1000 training datasets of size 100, learn a random forest from each dataset, and compute the misclassification error in the **same** test dataset of size 1000. Report results for when the random forest has 1, 10 and 100 trees.
2. Repeat the exercise above but this time use the condition (x1<0.5) instead of (x1<x2) when producing the training **and** test datasets.
3. Repeat the exercise above but this time use the condition ((x1<0.5 & x2<0.5) | (x1>0.5 & x2>0.5)) instead of (x1<x2) when producing the training **and** test datasets. Unlike above, use **nodesize =** 12 for this exercise**.**
4. Answer the following questions:
   1. What happens with the mean and variance of the error rate when the number of trees in the random forest grows ?
   2. The third dataset represents a slightly more complicated classification problem than the first one. Still, you should get better performance for it when using sufficient trees in the random forest. Explain why you get better performance.
   3. Why is it desirable to have low error variance ?

### Assignment 2. Mixture models

Your task is to implement the EM algorithm for mixtures of multivariate Bernoulli distributions. Please use the R template below to solve the assignment. Then, use your implementation to show what happens when your mixture model has too few and too many components, i.e. set K=2,3,4 and compare results. Please provide a short explanation as well.

set.seed(1234567890)

max\_it <- 100 # max number of EM iterations

min\_change <- 0.1 # min change in log likelihood between two consecutive EM iterations

N=1000 # number of training points

D=10 # number of dimensions

x <- matrix(nrow=N, ncol=D) # training data

true\_pi <- vector(length = 3) # true mixing coefficients

true\_mu <- matrix(nrow=3, ncol=D) # true conditional distributions

true\_pi=c(1/3, 1/3, 1/3)

true\_mu[1,]=c(0.5,0.6,0.4,0.7,0.3,0.8,0.2,0.9,0.1,1)

true\_mu[2,]=c(0.5,0.4,0.6,0.3,0.7,0.2,0.8,0.1,0.9,0)

true\_mu[3,]=c(0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5)

plot(true\_mu[1,], type="o", col="blue", ylim=c(0,1))

points(true\_mu[2,], type="o", col="red")

points(true\_mu[3,], type="o", col="green")

# Producing the training data

for(n in 1:N) {

k <- sample(1:3,1,prob=true\_pi)

for(d in 1:D) {

x[n,d] <- rbinom(1,1,true\_mu[k,d])

}

}

K=3 # number of guessed components

z <- matrix(nrow=N, ncol=K) # fractional component assignments

pi <- vector(length = K) # mixing coefficients

mu <- matrix(nrow=K, ncol=D) # conditional distributions

llik <- vector(length = max\_it) # log likelihood of the EM iterations

# Random initialization of the paramters

pi <- runif(K,0.49,0.51)

pi <- pi / sum(pi)

for(k in 1:K) {

mu[k,] <- runif(D,0.49,0.51)

}

pi

mu

for(it in 1:max\_it) {

plot(mu[1,], type="o", col="blue", ylim=c(0,1))

points(mu[2,], type="o", col="red")

points(mu[3,], type="o", col="green")

#points(mu[4,], type="o", col="yellow")

Sys.sleep(0.5)

# E-step: Computation of the fractional component assignments

# Your code here

#Log likelihood computation.

# Your code here

cat("iteration: ", it, "log likelihood: ", llik[it], "\n")

flush.console()

# Stop if the lok likelihood has not changed significantly

# Your code here

#M-step: ML parameter estimation from the data and fractional component assignments

# Your code here

}

pi

mu

plot(llik[1:it], type="o")

### Assignment 3. High-dimensional methods

Data file **geneexp.csv** contains information about gene expression of three different cell types (column Cell Type). These cell types are CD4 and CD8 (two sorts of T cells) and CD19 (B cells). The aim of this assignment is to classify cells to the appropriate cell types using gene expressions and discover relevant genes for the given cell types.

1. Divide data into training and test sets (70/30) without scaling. Perform nearest shrunken centroid classification of training data in which the threshold is chosen by cross-validation. Provide a centroid plot and interpret it. How many genes were selected by the method? What meaning do positive and negative values have in the centroid plot? Can it happen that all values in the centroid plot are positive for some gene?
2. List the names of the 2 most contributing genes and find their alternative names in Google. Then, by checking this webpage <https://panglaodb.se/markers.html> find out whether these two genes are “marker genes” for given cell types. Report the test error of the model.
3. Compute the test error and the number of the contributing features for the following methods fitted to the training data:
   1. Elastic net with the binomial response and in which penalty is selected by the cross-validation
   2. Support vector machine with “vanilladot” kernel.

Compare the results of these models with the results of the nearest shrunken centroids (make a comparative table). Which model would you prefer and why?

1. Implement Benjamini-Hochberg method for the original data in which you test each cell type versus the remaining ones, and use t.test() for computing p-values. Present plots showing p-values and the rejection area for each cell type and interpret them. How many genes correspond to the rejected hypotheses for each cell type?

***Submission procedure***

***First read ‘Course Information.PDF’ at LISAM, folder ‘Course documents’***

**Assume that X is the current lab number, Y is your group number.**

**If you are neither speaker nor opponent for this lab,**

* Submit your report using *Lab X* item in the *Submissions* folder before the deadline.
* Make sure that you or some of your group members submits the group report using *Lab X group report*  in the *Submissions* folder before the deadline

**If you are a speaker for this lab,**

* Submit your report using *Lab X* item in the *Submissions* folder before the deadline.
* Make sure that you or some of your group members does the following before the deadline:
  + submits the group report using *Lab X group report*  in the *Submissions* folder before the deadline
  + Goes to *Sumissions* and opens item *Password X*. Then the student should put your group report into ZIP file *Lab X\_Group Y.zip* and protect it with a password you found in Password X
  + Uploads the file to *Collaborative workspace* 🡪*Lab X* folder

**If you are opponent for this lab,**

* Submit your report using *Lab X* item in the *Submissions* folder before the deadline.
* Make sure that you or some of your group members submits the group report using *Lab X group report*  in the *Submissions* folder before the deadline
* After the deadline for the lab has passed, go to Collaborative workspace🡪*Lab X* folder and download the appropriate ZIP file. Open the PDF in this ZIP file by using the password available in *Submission🡪Password X* item, read it carefully and prepare (in cooperation with other group members) **at least three questions/comments/improvement suggestions per lab assignment** in order to put them at the seminar.

Hi,

yesterday I was not satisfied with my explanation of why bagging is beneficial for classification. You can find the proper explanation below. Regards, Jose M.

We saw that bagging reduces the error, the variance of the error and the variance of the prediction compared with the average individual predictor. These results build on some assumptions that may not hold in practice. However, the assumptions almost hold in many domains and, thus, the mentioned reductions typically occur in practice, although they may be smaller than what theory says. For instance, the error may decrease by a factor smaller than B because the individual predictors' errors may be correlated. In any case, we can prove (using Jensen's inequality) that the bagging predictor never has greater error than the average individual predictor. Unfortunately, this is not true for classification, as the following example shows. Assume a set of predictors for binary classification. Each of them makes an error with probability 0.6. Then, the bagging classifier will make an error with probability 1, because we expect 60 % of the individual predictors to vote for the wrong label and, thus, the bagging classifier will always predict the wrong label, because it predicts the label in the majority. So, the bagging classifier performs worse than the average individual classifier.

The example above is quite unrealistic, because it builds on classifiers than perform worse than a random classifier (that has an error rate of 0.5). Luckily, when the error rate of the individual classifiers is below 0.5 (which is what typically happens in practice), bagging is beneficial. To see this, assume that the individual classifiers have all error rate e (<0.5). Then, each of them classifies an instance x correctly with probability 1-e. Then, the number S of individual classifiers that correctly classify x follow a Binomial(B,1-e) distribution. We are interested in the probability p(S>B/2), i.e. the probability that at least half of the individual classifiers classify x correctly. Why ? because the bagging classifier classifies according to majority voting. So, if the majority is right then the bagging classifier will be right. We can prove that p(S>B/2) goes to 1 when B grows. So, the probability of the bagging classifier classifying correctly goes to 1 as B grows. Again, this result builds on some assumptions (the individual classifiers are independent and all have the same error rate) which may only hold approximately in practice. So, we should not expect the bagging classifier to classify every instance correctly. It is more reasonable to expect the bagging classifier to simply outperform the average individual classifier.

Jose M. Peña

STIMA, IDA, LiU

|  |
| --- |
|  |