Tenth post of our series on [classification from scratch](https://freakonometrics.hypotheses.org/52731). Today, we’ll see the heuristics of the algorithm inside bagging techniques.

Often, bagging is associated with trees, to generate forests. But actually, it is possible using bagging for any kind of model. Recall that [bagging](https://en.wikipedia.org/wiki/Bootstrap_aggregating) means “boostrap aggregation”. So, consider a model . Let denote the estimator of obtained from sample with .

Consider now some boostrap sample, with is randomly drawn from (with replacement). Based on that sample, estimate . Then draw many samples, and consider the agregation of the estimators obtained, using either a majority rule, or using the average of probabilities (if a probabilist model was considered). Hence

**Bagging logistic regression #1**

Consider the case of the logistic regression. To generate a bootstrap sample, it is natural to use the technique describe above. I.e. draw pairs randomly, uniformly (with probability ) with replacement. Consider here the small dataset, just to visualize. For the **b** part of **b**agging, use the following code

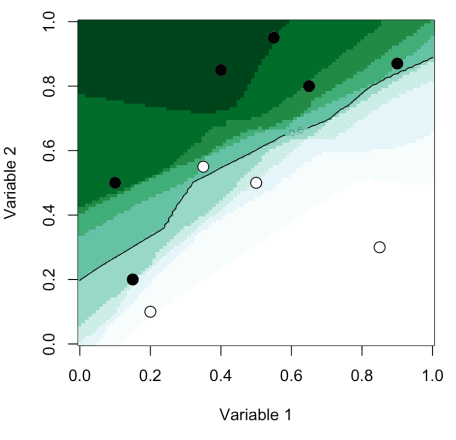
|  |  |
| --- | --- |
| 1  2  3  4  5 | L\_logit = **list**()  n = **nrow**(**df**)  **for**(s **in** 1:1000){  df\_s = **df**[**sample**(1:n,size=n,**replace**=TRUE),]  L\_logit[[s]] = **glm**(y~., df\_s, **family**=**binomial**)} |

Then we should aggregate over the 1000 models, to get the **agg** part of b**agg**ing,

|  |  |
| --- | --- |
| 1  2  3 | p = **function**(x){  nd=**data.frame**(x1=x[1], x2=x[2])  **unlist**(**lapply**(1:1000,**function**(z) **predict**(L\_logit[[z]],newdata=nd,type="response")))} |

We now have a prediction for any new observation

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | vu = **seq**(0,1,**length**=101)  vv = **outer**(vu,vu,**Vectorize**(**function**(x,y) **mean**(p(**c**(x,y)))))  **image**(vu,vu,vv,xlab="Variable 1",ylab="Variable 2",**col**=clr10,breaks=(0:10)/10)  **points**(**df**$x1,**df**$x2,pch=19,cex=1.5,**col**="white")  **points**(**df**$x1,**df**$x2,pch=**c**(1,19)[1+(**df**$y=="1")],cex=1.5)  **contour**(vu,vu,vv,**levels** = .5,add=TRUE) |



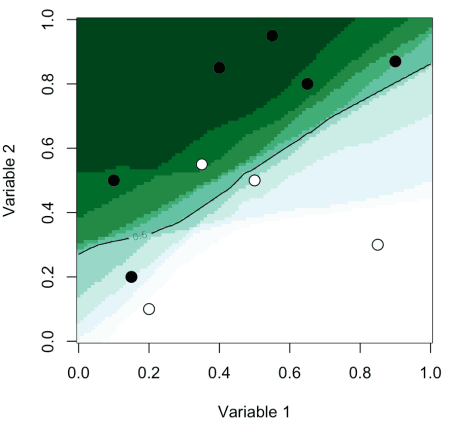
**Bagging logistic regression #2**

Another technique that can be used to generate a bootstrap sample is to keep all ‘s, but for each of them, to draw (randomly) a value for , withsinceThus, the code for the **b** part of **b**agging algorithm is now

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | L\_logit = **list**()  n = **nrow**(**df**)  reg = **glm**(y~x1+x2, **df**, **family**=**binomial**)  **for**(s **in** 1:100){  df\_s = **df**  df\_s$y = **factor**(**rbinom**(n,size=1,prob=**predict**(reg,type="response")),**labels**=0:1)  L\_logit[[s]] = **glm**(y~., df\_s, **family**=**binomial**)  } |

The **agg** part of b**agg**ing algorithm remains unchanged. Here we obtain

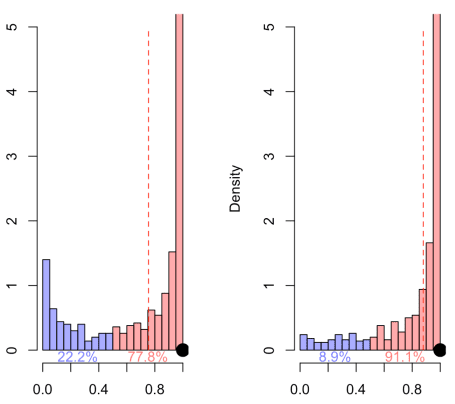
|  |  |
| --- | --- |
| 1  2  3  4  5  6 | vu = **seq**(0,1,**length**=101)  vv = **outer**(vu,vu,**Vectorize**(**function**(x,y) **mean**(p(**c**(x,y)))))  **image**(vu,vu,vv,xlab="Variable 1",ylab="Variable 2",**col**=clr10,breaks=(0:10)/10)  **points**(**df**$x1,**df**$x2,pch=19,cex=1.5,**col**="white")  **points**(**df**$x1,**df**$x2,pch=**c**(1,19)[1+(**df**$y=="1")],cex=1.5)  **contour**(vu,vu,vv,**levels** = .5,add=TRUE) |

  
Of course, we can use that code we check the prediction obtain on the observations we have in our sample. Just to change, consider here the myocarde data. The entiere code is here

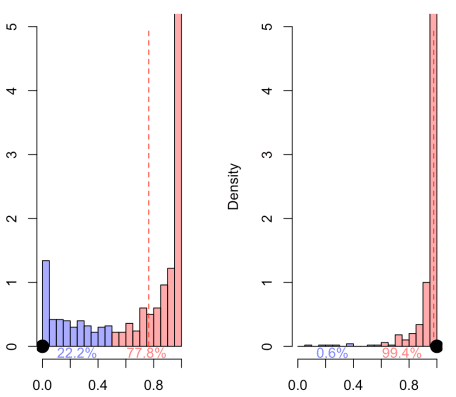
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | L\_logit = **list**()  reg = **glm**(**as.factor**(PRONO)~., myocarde, **family**=**binomial**)  **for**(s **in** 1:1000){  myocarde\_s = myocarde  myocarde\_s$PRONO = 1\***rbinom**(n,size=1,prob=**predict**(reg,type="response"))  L\_logit[[s]] = **glm**(**as.factor**(PRONO)~., myocarde\_s, **family**=**binomial**)  }  p = **function**(x){  nd=**data.frame**(FRCAR=x[1], INCAR=x[2], INSYS=x[3], PRDIA=x[4],  PAPUL=x[4], PVENT=x[5], REPUL=x[6])  **unlist**(**lapply**(1:1000,**function**(z) **predict**(L\_logit[[z]],newdata=nd,type="response")))} |

For the first observation, with our 1000 simulated datasets, and our 1000 models, we obtained the following estimation for the probability to die.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | histo = **function**(i){  x = **as.numeric**(myocarde[i,1:7])  v\_x = p(x)  **hist**(v\_x,proba=TRUE,breaks=**seq**(0,1,**by**=.05),xlab="",main="",  **col**=**rep**(**c**(**rgb**(0,0,1,.4),**rgb**(1,0,0,.4)),each=10),ylim=**c**(0,5))  **segments**(**mean**(v\_x),0,**mean**(v\_x),5,**col**="red",lty=2)  **points**(myocarde$PRONO[i],0,pch=19,cex=2)  xi = **round**(**mean**(v\_x.5)\*1000)/10  **text**(.75,-.1,**paste**(xi,"%",sep=""),**col**=**rgb**(1,0,0,.6))}  histo(1)  histo(4) |

Hence, for the first observation, in 77.8% of the models, the predicted probability was higher than 50%, and the average probability was actually close to 75%.  
  
or, for observation 22, predictions very close to the first one (except that the first one died, while the 22nd survived)

|  |  |
| --- | --- |
| 1  2 | histo(23)  histo(11) |

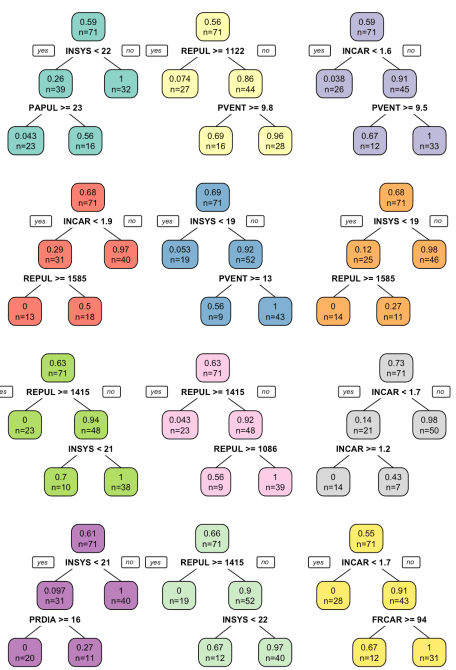
and, we observe here  


**Bagging trees**

Let’s now get back on our trees. Bagging was introduced in 1994 by Leo Breiman in [Bagging Predictors](https://www.stat.berkeley.edu/~breiman/bagging.pdf). If the first section describes the procedure, the second one introduces “Bagging Classification Trees”. Trees are nice for interpretation, but most of the time, they are rather poor predictors. The idea of bagging was to improve the accuracy of classification trees.

The idea of **b**agging to to generate a lot of trees

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | clr12 = **c**("#8dd3c7","#ffffb3","#bebada","#fb8072","#80b1d3","#fdb462","#b3de69","#fccde5","#d9d9d9","#bc80bd","#ccebc5","#ffed6f")  n = **nrow**(myocarde)  **par**(mfrow=**c**(4,3))  sed=**c**(1,2,4,5,6,10,11,21,22,24,27,28,30)  **for**(i **in** 1:12){  **set.seed**(sed[i])  idx = **sample**(1:n, size=n, **replace**=TRUE)  cart = rpart(PRONO~., myocarde[idx,])  prp(cart,type=2,extra=1,box.col=clr12[i])} |

  
The strategie is actually the same as before. For the **b**ootstrap part, store the tree in a list

|  |  |
| --- | --- |
| 1  2  3  4  5 | L\_tree = **list**()  **for**(s **in** 1:1000){  idx = **sample**(1:n, size=n, **replace**=TRUE)  L\_tree[[s]] = rpart(**as.factor**(PRONO)~., myocarde[idx,])  } |

and for the **agg**regation part, just take the average of predicted probabilities

|  |  |
| --- | --- |
| 1  2  3  4 | p = **function**(x){  nd=**data.frame**(FRCAR=x[1], INCAR=x[2], INSYS=x[3], PRDIA=x[4],  PAPUL=x[4], PVENT=x[5], REPUL=x[6])  **unlist**(**lapply**(1:1000,**function**(z) **predict**(L\_tree[[z]],newdata=nd,type="prob")[,2]))} |

Because with this example, we cannot visualize predictions, let us run the same code on the smaller dataset

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | L\_tree = **list**()  n = **nrow**(**df**)  **for**(s **in** 1:1000){  idx = **sample**(1:n, size=n, **replace**=TRUE)  L\_tree[[s]] = rpart(y~x1+x2, **df**[idx,],control = rpart.control(cp = 0.25,  minsplit = 2))  }  p = **function**(x){  nd=**data.frame**(x1=x[1], x2=x[2])  **unlist**(**lapply**(1:1000,**function**(z) **predict**(L\_tree[[z]],newdata=nd,type="prob")[,2]))}  vu=**seq**(0,1,**length**=101)  vv=**outer**(vu,vu,**Vectorize**(**function**(x,y) **mean**(p(**c**(x,y)))))  **image**(vu,vu,vv,xlab="Variable 1",ylab="Variable 2",**col**=clr10,breaks=(0:10)/10)  **points**(**df**$x1,**df**$x2,pch=19,cex=1.5,**col**="white")  **points**(**df**$x1,**df**$x2,pch=**c**(1,19)[1+(**df**$y=="1")],cex=1.5)  **contour**(vu,vu,vv,**levels** = .5,add=TRUE) |

