Before moving to cross-validation, it was natural to say “I will burn 50% (say) of my data to train a model, and then use the remaining to fit the model”. For instance, we can use training data for variable selection (e.g. using some stepwise procedure in a logistic regression), and then, once variable have been selected, fit the model on the remaining set of observations. A natural question is usually “does it really matter ?”.

In order to visualize this problem, consider my (simple) dataset

|  |  |
| --- | --- |
| 1  2  3 | MYOCARDE=**read.table**(  "http://freakonometrics.free.fr/saporta.csv",  **head**=TRUE,sep=";") |

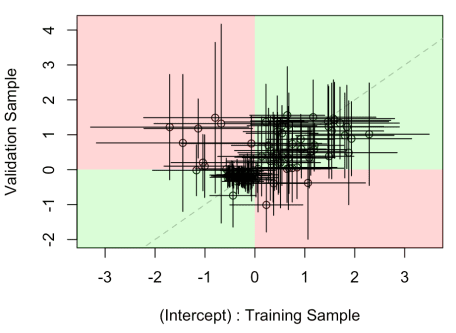
Let us generate 100 training samples (where we keep about 50% of the observations). On each of them, we use a stepwise procedure, and we keep the estimates of the remaining variables (and their standard deviation actually)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | n=**nrow**(MYOCARDE)  M=**matrix**(NA,100,**ncol**(MYOCARDE))  **colnames**(M)=**c**("(Intercept)",**names**(MYOCARDE)[1:7])  S1=S2=M1=M2=M  **for**(i **in** 1:100){  idx = **which**(**sample**(0:1,size=n, **replace**=TRUE)==1)  reg=**step**(**glm**(PRONO=="DECES"~.,**data**=MYOCARDE[idx,]))  nm=**names**(reg$coefficients)  M1[i,nm]=reg$coefficients  S1[i,nm]=**summary**(reg)$coefficients[,2]  f=**paste**("PRONO=='DECES'~",**paste**(nm[-1],collapse="+"),sep="")  reg=**glm**(f,**data**=MYOCARDE[-idx,])  M2[i,nm]=reg$coefficients  S2[i,nm]=**summary**(reg)$coefficients[,2]  } |

Then, for the 7 covariates (and the constant) we can look at the value of the coefficient in the model fitted on the training sample, and the value on the model fitted on the validation sample (of course, only when they were remaining)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **for**(j **in** 1:8){  idx=**which**(!**is.na**(M1[,j]))  **plot**(M1[idx,j],M2[idx,j])  **abline**(a=0,b=1,lty=2,**col**="gray")  **segments**(M1[idx,j]-2\*S1[idx,j],M2[idx,j],M1[idx,j]+2\*S1[idx,j],M2[idx,j])  **segments**(M1[idx,j],M2[idx,j]-2\*S2[idx,j],M1[idx,j],M2[idx,j]+2\*S2[idx,j])  } |

For instance, with the intercept, we have the following

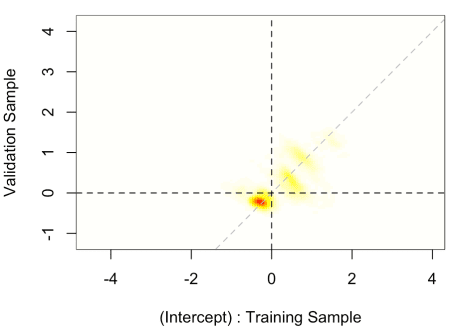


where horizontal segments are confidence intervals of the parameter on the model fitted on the training sample, the vertical on the validation sample. The green part means some sort of consistency, while the red one means that actually, the coefficient was negative with one model, positive with the other one. Which is odd (but in that case, observe that coefficients are rarely significant).

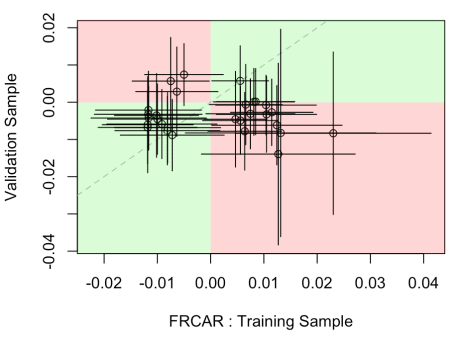
We can also visualize the joint distribution of the two estimators,

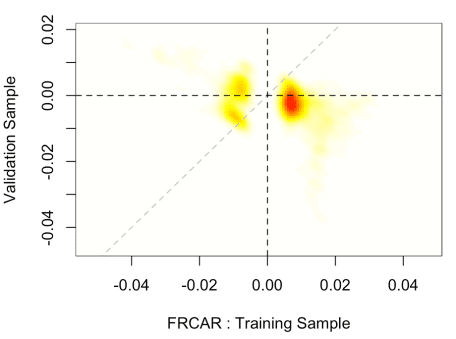
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **for**(j **in** 1:8){  **library**(ks)  idx = **which**(!**is.na**(M1[,j]))  Z = **cbind**(M1[idx,j],M2[idx,j])  H = Hpi(x=Z)  fhat = kde(x=Z, H=H)  **image**(fhat$eval.points[[1]],  fhat$eval.points[[2]],fhat$estimate)  **abline**(a=0,b=1,lty=2,**col**="gray")  **abline**(v=0,lty=2)  **abline**(h=0,lty=2)  } |

which are here, almost on the diagonal,



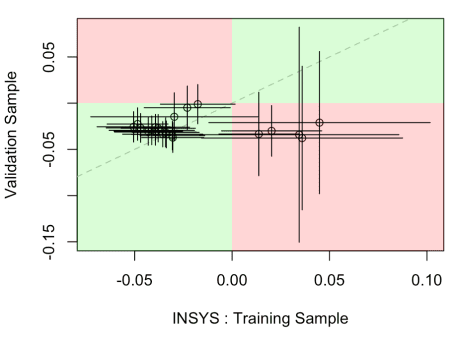
meaning that the intercept on the two samples is (more or less) the same. We can then look at other parameters (which is actually more interesting).

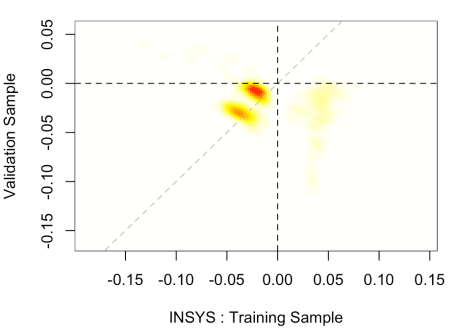


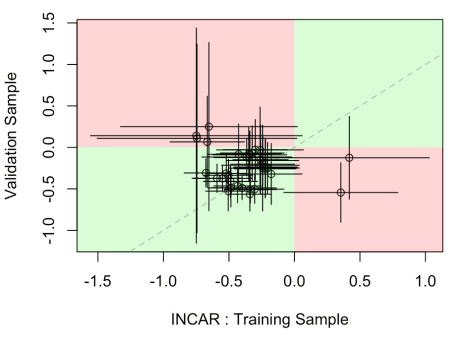


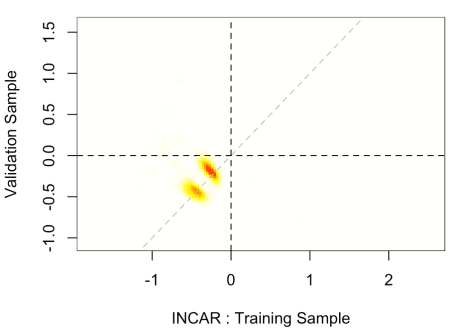
On that variable, it seems that it is significant on the training dataset (somehow, it is consistent with the fact that it is remaining in the model after the stepwise procedure) but not on the validation sample (or hardly significant).

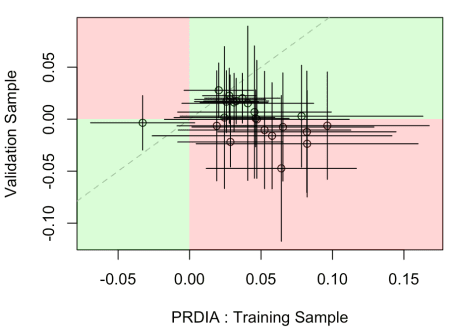
Others are much more consistent (with some possible outliers)

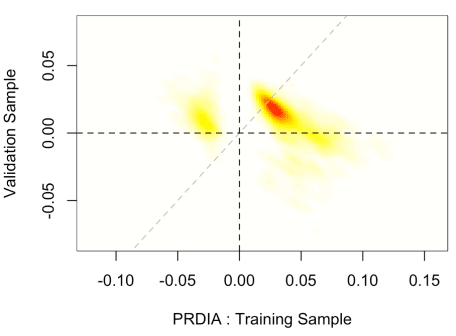




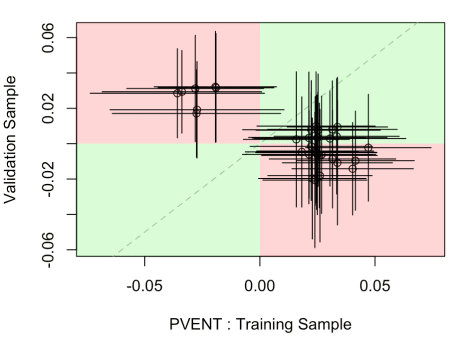


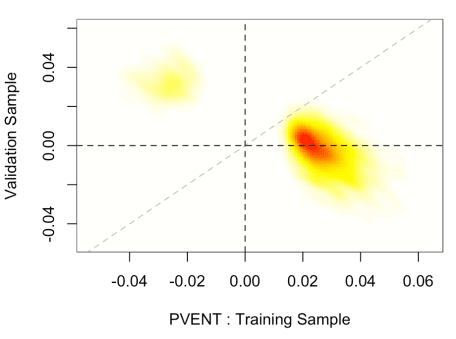


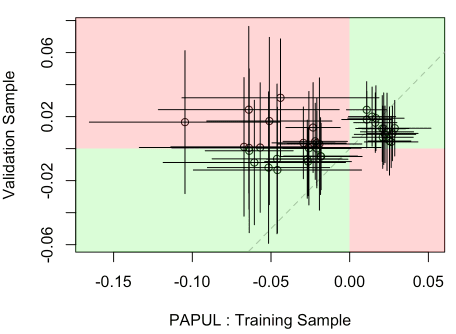


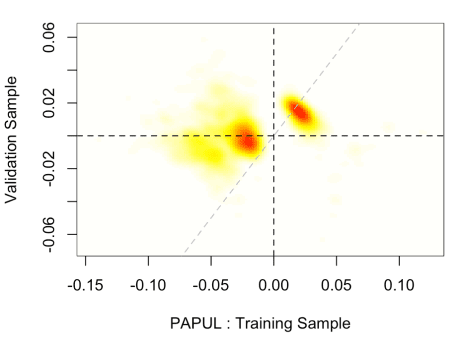


On the next one, we have again significance on the training sample, but not on the validation sample,

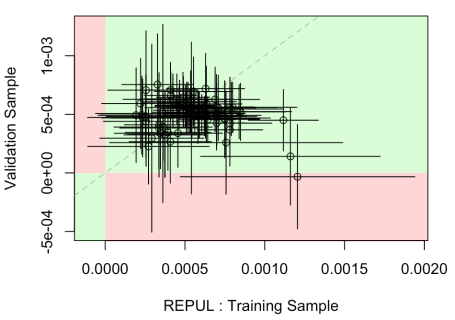


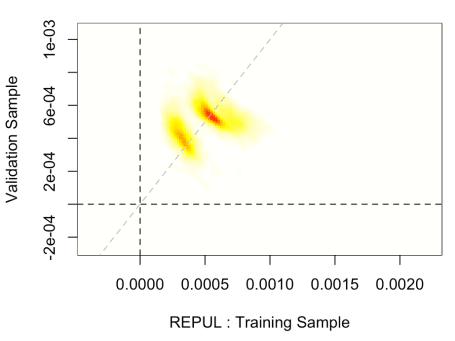






and probably more interesting





where the two are very consistent.