Checking Our Fit From Supplementary Report SR9

Keith A. Baggerly and Kevin R. Coombes

November 13, 2007

1 Introduction

In their reply to our correspondence, Potti and Nevins comment (with respect to predicting docetaxel) that

> Moreover, when Coombes et al. compared the results of models that create metagenes from training data alone to the more extensive model that creates metagenes with both training and test data, they obtained a very similar result to ours (Fig.8 in Supplementary Report 9). In short, they reproduce our result when they use our methods.

We don't agree with this interpretation of our findings. What the figure in question shows is that when the "more extensive model" is used, we place some samples in each of the sensitive and resistant categories, as opposed to simply placing them all in one group. However, applying both sets of labels does not mean that we are applying the *correct* labels, and we found our accuracy to be much worse than what they reported.

Here, we check our earlier results to see exactly how well we did. Since we saved all of the results from our earlier Matlab run, this is largely an exercise in the use of the R.matlab package.

2 Options and Libraries

```
> options(width = 80)
```

```
> library(R.matlab)
```

```
R.matlab v1.1.3 (2007-04-07) successfully loaded. See ?R.matlab for help.
```

3 Load Results

The numbers we need were stored as part of the "Silent" structure in Matlab, specifically part "f4" which contains the numbers needed to produce the test data plot.

```
> sr9Results <- readMat(file.path("MatlabFiles", "DoceTrainAndTestOldPlotData.mat"))
> names(sr9Results)
```

```
[1] "Silent"
```

```
> sr9Results$Silent
```

```
, , 1

    [,1]
f1  List,7
f2  List,5
f3  List,8
f4  List,7
f5  List,5
f11 List,5
```

```
> class(sr9Results)
```

```
[1] "list"
```

```
> sr9Results$Silent["f4", 1, 1]
```

```
[[1]]
, , 1

       [,1]
use    1
fit    Numeric,38
ivalid Numeric,24
Z      Integer,38
pfit   Numeric,38
sl     Numeric,38
su     Numeric,38
```

```
> class(sr9Results$Silent["f4", 1, 1])
```

```
[1] "list"
```

```
> pFit <- as.vector(sr9Results$Silent["f4", 1, 1][[1]]["pfit",
+     1, 1][[1]])
```

We now have the 38 fitted probability scores. In order to make sure we give the correct interpretation for values close to 0 or 1, let's also load the sample info that we saved when we ran the Matlab scripts.

```
> trainingInfo <- read.table(file.path("MatlabFiles", "DoceTrainAndTestSampleInfo.csv"),
+     sep = ",", header = TRUE, nrows = 14)
> testInfo <- read.table(file.path("MatlabFiles", "DoceTrainAndTestSampleInfo.csv"),
+     sep = ",", header = TRUE, nrows = 24, skip = 15)
> trainingInfo
```

```
    index drugName responseStatus    Source NovartisName
108   108     Doce      Resistant      EKVX       A.EKVX
109   109     Doce      Resistant    IGROV1     A.IGROV1
110   110     Doce      Resistant   OVCAR-4    A.OVCAR-4
111   111     Doce      Resistant     786-0      A.786-0
112   112     Doce      Resistant    CAKI-1     A.CAKI-1
```

```
113   113     Doce      Resistant     SN12C      A.SN12C
114   114     Doce      Resistant     TK-10      A.TK-10
115   115     Doce      Sensitive HL-60(TB)  A.HL-60(TB)
116   116     Doce      Sensitive    SF-539     A.SF-539
117   117     Doce      Sensitive      HT29       A.HT29
118   118     Doce      Sensitive    HOP-62     A.HOP-62
119   119     Doce      Sensitive  SK-MEL-2   A.SK-MEL-2
120   120     Doce      Sensitive  SK-MEL-5   A.SK-MEL-5
121   121     Doce      Sensitive  NCI-H522   A.NCI-H522
```

```
> testInfo
```

```
    GEO.ID Response
1  GSM4903     Resp
2  GSM4907     Resp
3  GSM4908     Resp
4  GSM4914     Resp
5  GSM4915     Resp
6  GSM4917     Resp
7  GSM4919     Resp
8  GSM4920     Resp
9  GSM4921     Resp
10 GSM4923     Resp
11 GSM4913     Resp
12 GSM4901       NR
13 GSM4902       NR
14 GSM4904       NR
15 GSM4905       NR
16 GSM4906       NR
17 GSM4909       NR
18 GSM4910       NR
19 GSM4911       NR
20 GSM4912       NR
21 GSM4916       NR
22 GSM4918       NR
23 GSM4922       NR
24 GSM4924       NR
```

The first 7 training values are from resistant samples, and the next 7 are from sensitive samples. The first 11 test samples are Resp, and the last 13 are NR.

```
> summary(pFit[1:14][trainingInfo$responseStatus == "Resistant"])
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02662 0.04303 0.06324 0.06025 0.07718 0.09148
```

```
> summary(pFit[1:14][trainingInfo$responseStatus == "Sensitive"])
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.8606  0.9265  0.9433  0.9345  0.9600  0.9643
```

Looking at the summary values for the training data, values close to 0 are Resistant, and those close to 1 are Sensitive.

So, how well do we do?

```
> table(pFit[15:38] > 0.5, testInfo$Response)

        NR Resp
  FALSE  8    2
  TRUE   5    9
```

With a cutoff of 0.5, we get 9 of the 11 responders right, and 8 of the 13 nonresponders, for an accuracy of 17/24. This may be slightly better than chance, though using the test data warps the independence assumption enough that we're not sure.

However, this degree of accuracy does not match the 22/24 level reported in Figure 1d of Potti et al, and this approach is still not one that we would recommend.

4 Appendix

4.1 Saves

4.2 SessionInfo

```
> sessionInfo()

R version 2.5.1 (2007-06-27)
i386-pc-mingw32

locale:
LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONETARY=English_United Sta

attached base packages:
[1] "stats"     "graphics"  "grDevices" "utils"     "datasets"  "methods"
[7] "base"

other attached packages:
R.matlab     R.oo
 "1.1.3"  "1.3.0"
```