Validating predictions on the Chang data

Kevin R. Coombes, Jing Wang, and Keith A. Baggerly

13 March 2007

1 Description of the problem

We want to test the predictions from five models, built on training data from the NCI60 cell lines:

1. The cell lines we chose, with features based on Novartis A data
2. The cell lines we chose, with features based on averaged Novartis data
3. The cell lines Potti chose, with features based on Novartis A data
4. The cell lines Potti chose, with features based on averaged Novartis data
5. The cell lines Potti chose, with features reported by Potti in their online supplement, corrected for the off-by-one error.

We constructed the first four of these models ourselves. The fifth model, however, relies on a set of features reported by Potti et al that we are unable to reproduce.

We will use the Chang breast cancer data as the validation set. Note, however, that the Chang data has also been processed using two different dChip algorithms: the $PM-MM$ model (in `ChangSoftNL`, which was taken from the GEO source files) and the PM-only model (in `changDChipNL`, which we computed from the CEL files).

All told, then, we have to look at ten possible validations.

2 Load the existing data

Start with the individual data from Novartis and the data on 10 drugs from the DTP.

```
> prep <- file.path("Tangled", "prepareData.R")
> Stangle(file.path("RNowebSource", "prepareData.Rnw"),
+     output = prep)

Writing to file Tangled/prepareData.R

> source(prep)
> rm(prep)
```

Use this function to load cached data, if it exists, and produce it from scratch otherwise.

```
> getCached <- function(rda, r) {
+     rfile <- file.path("Tangled", paste(r, "R", sep = "."))
+     rdafile <- file.path("RDataObjects", paste(rda, "Rda",
+         sep = "."))
+     if (file.exists(rdafile)) {
+         cat("loading from cache\n")
+         load(rdafile, .GlobalEnv)
+     }
+     else {
+         Stangle(file.path("RNowebSource", paste(r, "Rnw",
+             sep = ".")), output = rfile)
+         source(rfile)
+     }
+ }
```

See how easy it is to use?

```
> getCached("chemoPredictors", "predCellLines")

loading from cache

> getCached("doceGI50", "gi50ValuesOverlap")

loading from cache

> getCached("features", "wobblingFeatures")

loading from cache

> getCached("pcaModels", "pca")

loading from cache

> getCached("changData", "prepareChang")

loading from cache

> library(ClassComparison)
> library(ClassDiscovery)
```

3 Making the predictions

First, we define a set of display colors to use in later PCA plots. Note that orange is used to denote non-responders (i.e., resistant patients) and blue is used to denote responders (i.e., sensitive patients).

```
> changColors <- c("orange", "blue")[as.numeric(changInfo[,
+     "Response"])]
> data.frame(changInfo, changColors)
```

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

In order to make predictions, we first pick out the subset of the validation data defined by the features selected for the model. We then project the validation data into the principal component space defined by those features from the training set. Using the probit regression model built using the principal components as predictors, we can see what the model says about the validation set. The following function carries out these computations.

```
> valPredictions <- function(model, valdata, valinfo) {
+     makePredictions <- function(data, model, info) {
```

```
+         train <- predict(model, type = "response") >
+             0.5
+         tab <- table(train, info$Response)
+         preds <- colnames(tab)[order(tab[2, ])]
+         preds[1 + as.numeric(predict(model, newdata = temp,
+             type = "response") > 0.5)]
+     }
+     valsubset <- valdata[model$features, ]
+     projection <- predict(model$spca, newdata = valsubset)
+     temp <- data.frame(projection)
+     N <- ncol(temp)
+     dimnames(temp) <- list(colnames(valdata), paste("PC",
+         1:N, sep = ""))
+     psens1 <- makePredictions(temp, model$justOne, model$info)
+     psens <- makePredictions(temp, model$better, model$info)
+     list(projection = projection, psens = psens, psens1 = psens1,
+         spca = model$spca)
+ }
```

We now use this function in the ten cases of interest:

```
> vOAS <- valPredictions(ourModelA, changSoftNL, changInfo)
> vOMS <- valPredictions(ourModelAvg, changSoftNL, changInfo)
> vPAS <- valPredictions(pottiModelA, changSoftNL, changInfo)
> vPMS <- valPredictions(pottiModelAvg, changSoftNL, changInfo)
> vPRS <- valPredictions(pottiModelRpt, changSoftNL, changInfo)
> vOAD <- valPredictions(ourModelA, changDChipNL, changInfo)
> vOMD <- valPredictions(ourModelAvg, changDChipNL, changInfo)
> vPAD <- valPredictions(pottiModelA, changDChipNL, changInfo)
> vPMD <- valPredictions(pottiModelAvg, changDChipNL, changInfo)
> vPRD <- valPredictions(pottiModelRpt, changDChipNL, changInfo)
```

We also define a plotting function to produce colored PCA plots.

```
> plotval <- function(val, ...) {
+     xl <- ceiling(max(abs(c(val$spca@scores[, 1], val$projection[,
+         1])))/3) * 3
+     yl <- ceiling(max(abs(c(val$spca@scores[, 2], val$projection[,
+         2])))/3) * 3
+     plot(val$spca, col = c("red", "green"), xlim = c(-xl,
+         xl), ylim = c(-yl, yl), ...)
+     points(val$projection[, 1], val$projection[, 2],
+         col = changColors, pch = 16, cex = 1.2)
```

```
+     invisible(val)
+ }
> legval <- function(x, y) {
+     legend(x, y, c("NCI60 Sensitive", "NCI60 Resistant",
+         "Chang Sensitive (Resp)", "Chang Resistant (NR)"),
+         col = c("green", "red", "blue", "orange"), pch = c(15,
+             15, 16, 16))
+ }
```

4 Assessing the predictions on Chang SOFT data

4.1 Trying to validate our cell lines, Novartis A features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vOAS$psens1, changInfo$Response)
```

```
           NR Resp
  Resistant  3    1
  Sensitive 10   10
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vOAS$psens, changInfo$Response)
```

```
           NR Resp
  Resistant  3    1
  Sensitive 10   10
```

4.2 Trying to validate our cell lines, average features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vOMS$psens1, changInfo$Response)
```

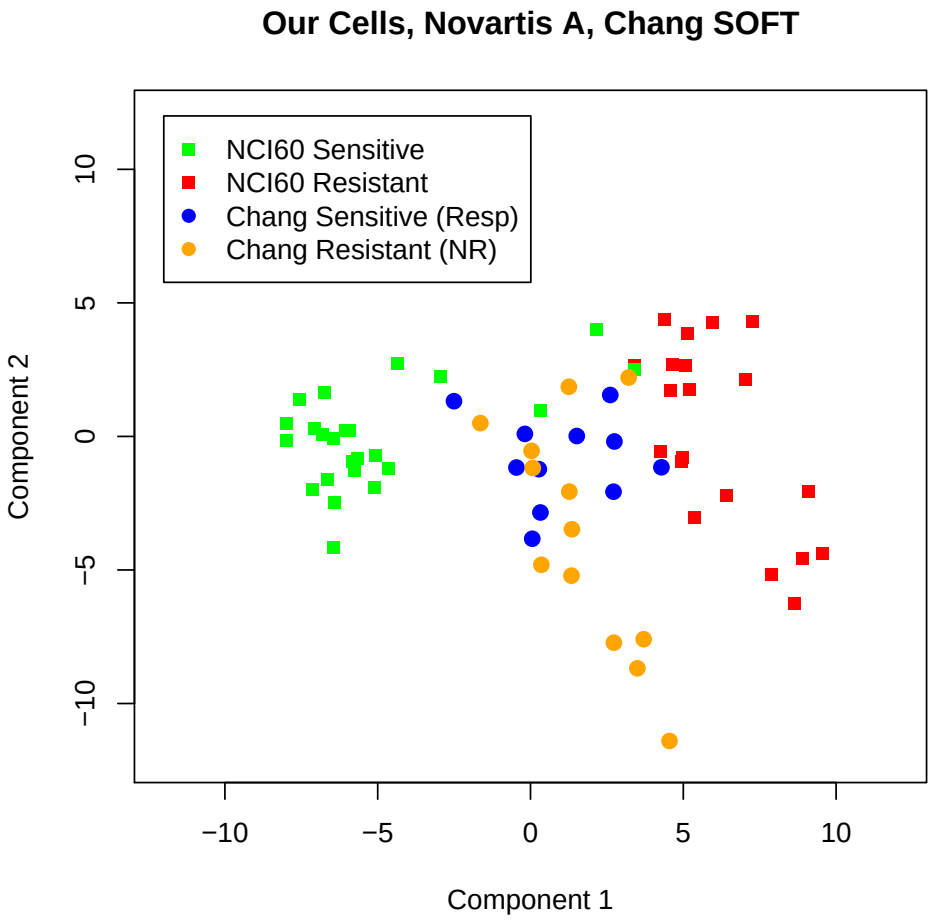

Figure 1: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

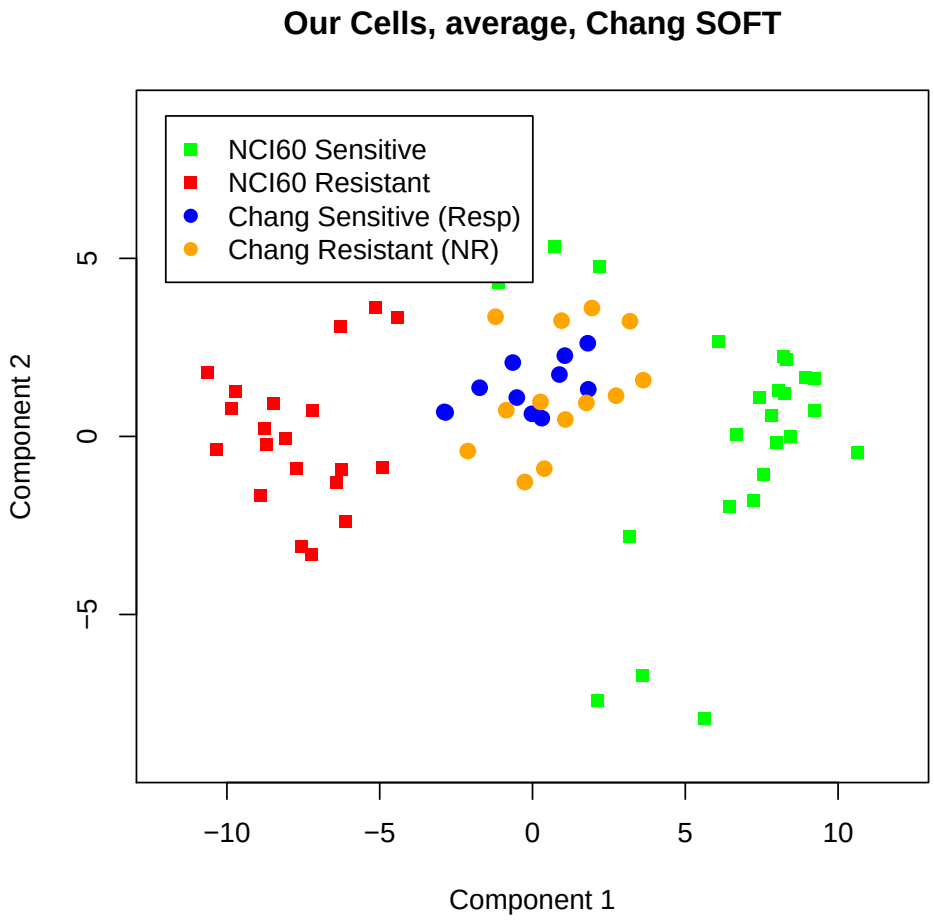

Figure 2: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

```
           NR Resp
  Resistant  0    2
  Sensitive 13    9
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vOMS$psens, changInfo$Response)

           NR Resp
  Resistant  0    2
  Sensitive 13    9
```

4.3 Trying to validate Potti cell lines, Novartis A features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vPAS$psens1, changInfo$Response)

           NR Resp
  Resistant  1    4
  Sensitive 12    7
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPAS$psens, changInfo$Response)

           NR Resp
  Resistant 12   11
  Sensitive  1    0
```

4.4 Trying to validate Potti cell lines, average features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

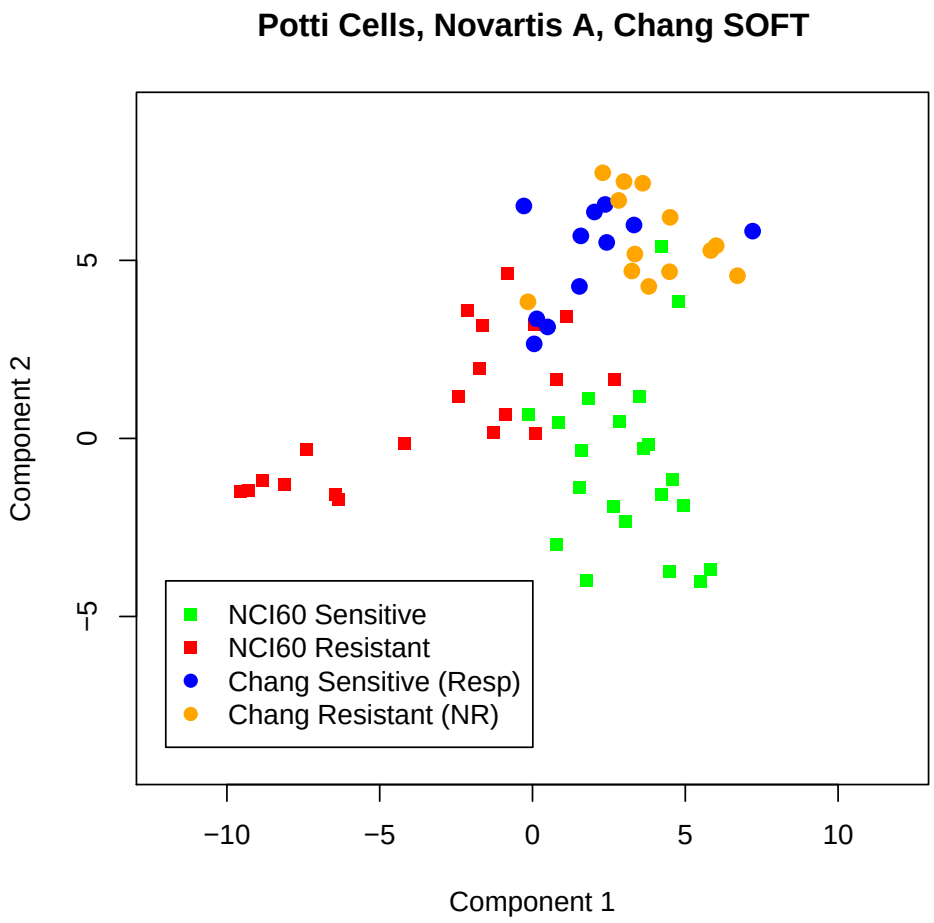

Figure 3: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

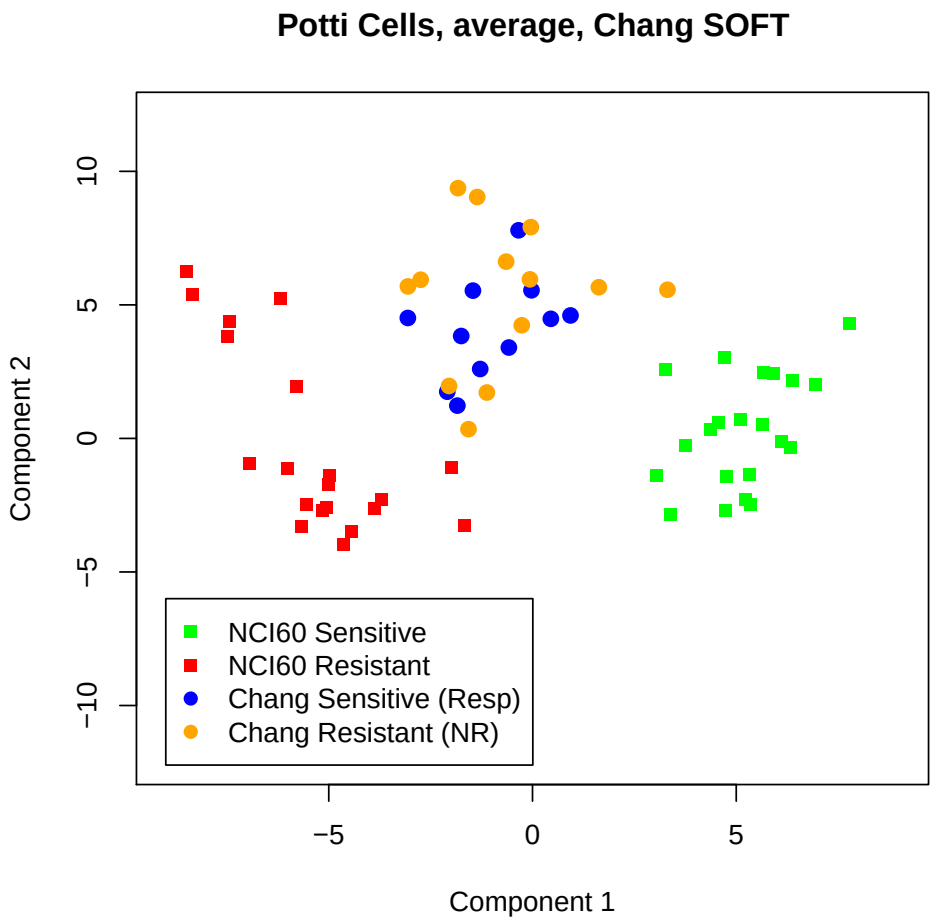

Figure 4: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

```
> table(vPMS$psens1, changInfo$Response)
```

```
           NR Resp
  Resistant 11   10
  Sensitive  2    1
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPMS$psens, changInfo$Response)
```

```
           NR Resp
  Resistant 11   10
  Sensitive  2    1
```

4.5 Trying to validate Potti cell lines, mystery features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vPRS$psens1, changInfo$Response)
```

```
           NR Resp
  Resistant  6    8
  Sensitive  7    3
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPRS$psens, changInfo$Response)
```

```
           NR Resp
  Resistant 13   10
  Sensitive  0    1
```

5 Assessing the predictions on Chang dChip data

5.1 Trying to validate our cell lines, Novartis A features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

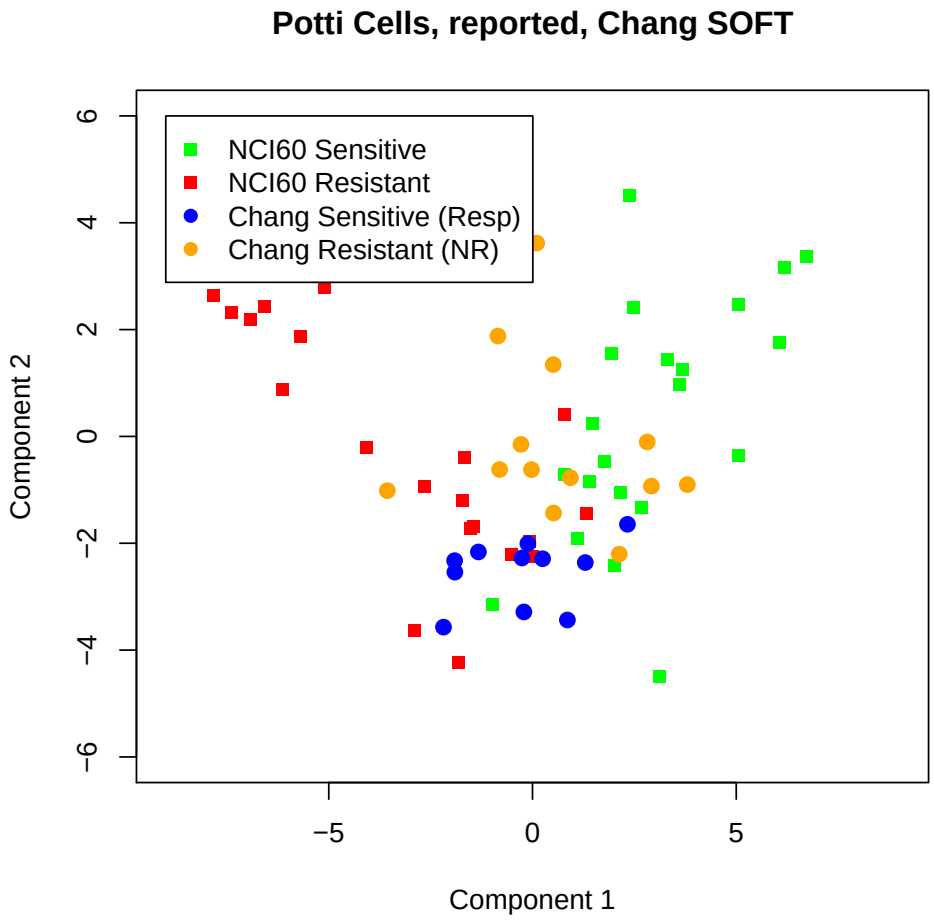

Figure 5: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

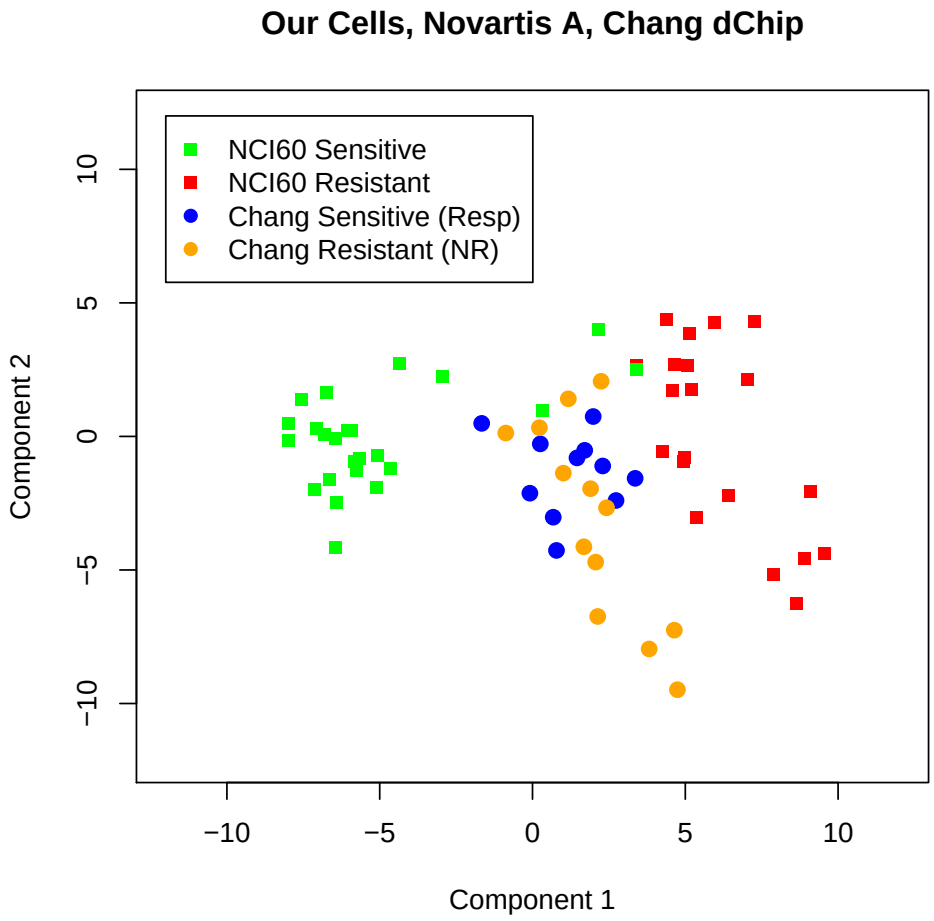

Figure 6: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vOAD$psens1, changInfo$Response)
```

```
           NR Resp
  Resistant  3    0
  Sensitive 10   11
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vOAD$psens, changInfo$Response)
```

```
           NR Resp
  Resistant  3    0
  Sensitive 10   11
```

5.2 Trying to validate our cell lines, average features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vOMD$psens1, changInfo$Response)
```

```
           NR Resp
  Resistant  1    1
  Sensitive 12   10
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vOMD$psens, changInfo$Response)
```

```
           NR Resp
  Resistant  1    1
  Sensitive 12   10
```

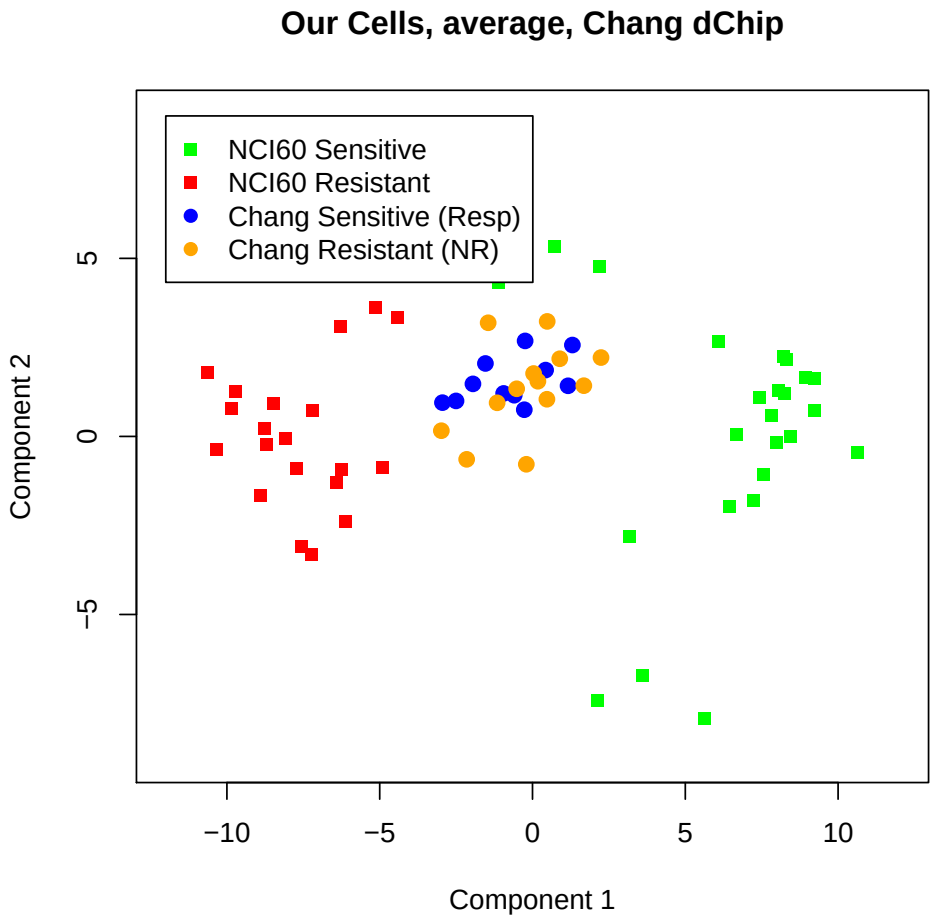

Figure 7: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

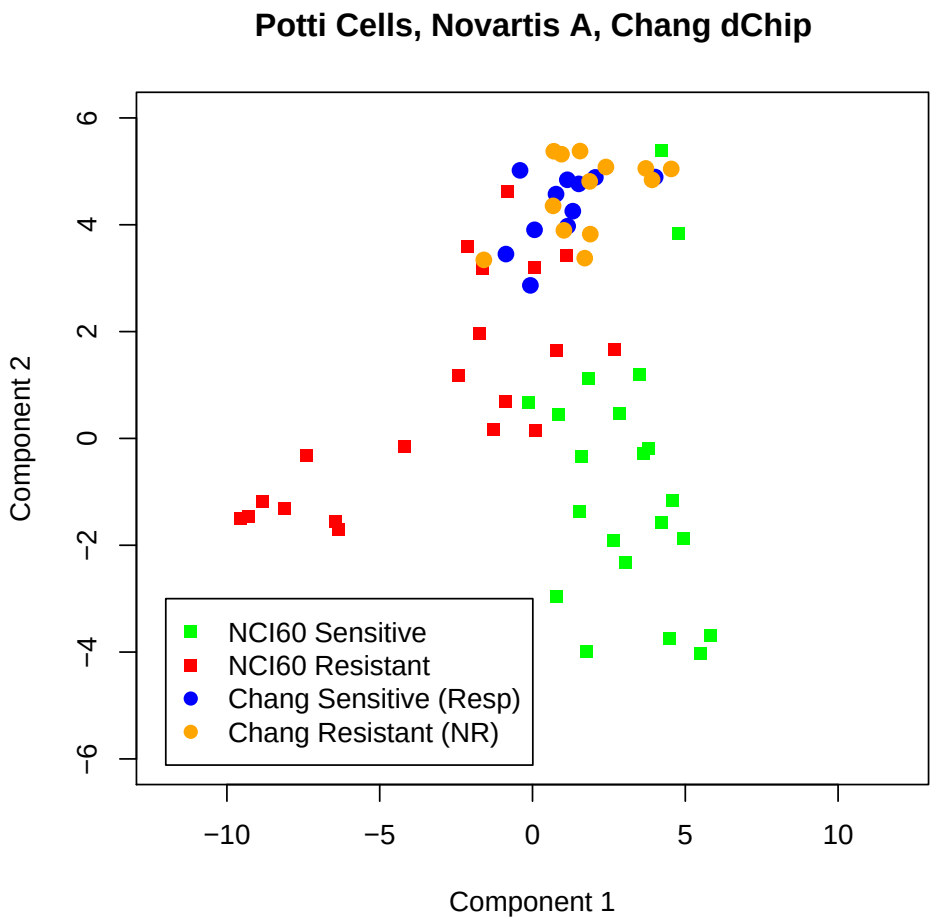

Figure 8: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

5.3 Trying to validate Potti cell lines, Novartis A features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vPAD$psens1, changInfo$Response)

           NR Resp
  Resistant  3    5
  Sensitive 10    6
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPAD$psens, changInfo$Response)

           NR Resp
  Resistant 13   11
```

5.4 Trying to validate Potti cell lines, average features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vPMD$psens1, changInfo$Response)

           NR Resp
  Resistant 13   11
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPMD$psens, changInfo$Response)

           NR Resp
  Resistant 13   11
```

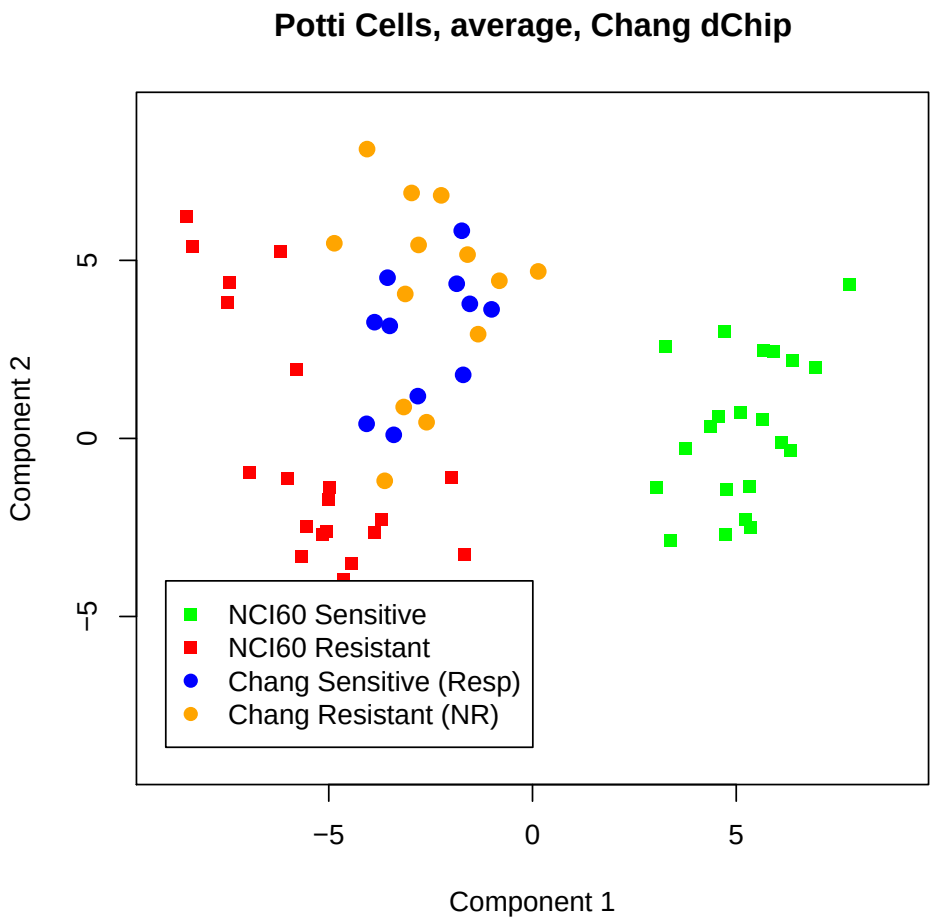

Figure 9: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.

5.5 Trying to validate Potti cell lines, mystery features

The main result is a PCA plot of the first two principal components, showing the training set and validation set on the same graph. Note that the first principal component mostly separates the training set, but does not separate the validation set.

We also built probit regression models on the training data. We now compute the predictions from those models on the validation data. The first model only uses the first principal component, PC1. Here is how the predictions compare with reality.

```
> table(vPRD$psens1, changInfo$Response)

           NR Resp
  Resistant 11   10
  Sensitive  2    1
```

The second model uses AIC to select an optimal number of principal components. Here is how its predictions compare with reality.

```
> table(vPRD$psens, changInfo$Response)

           NR Resp
  Resistant 13   11
```

6 Wrapup

```
> save(valPredictions, plotval, legval, changColors, vOAS,
+     vOMS, vPAS, vPMS, vPRS, vOAD, vOMD, vPAD, vPMD, vPRD,
+     file = file.path("RDataObjects", "predict.Rda"))
```

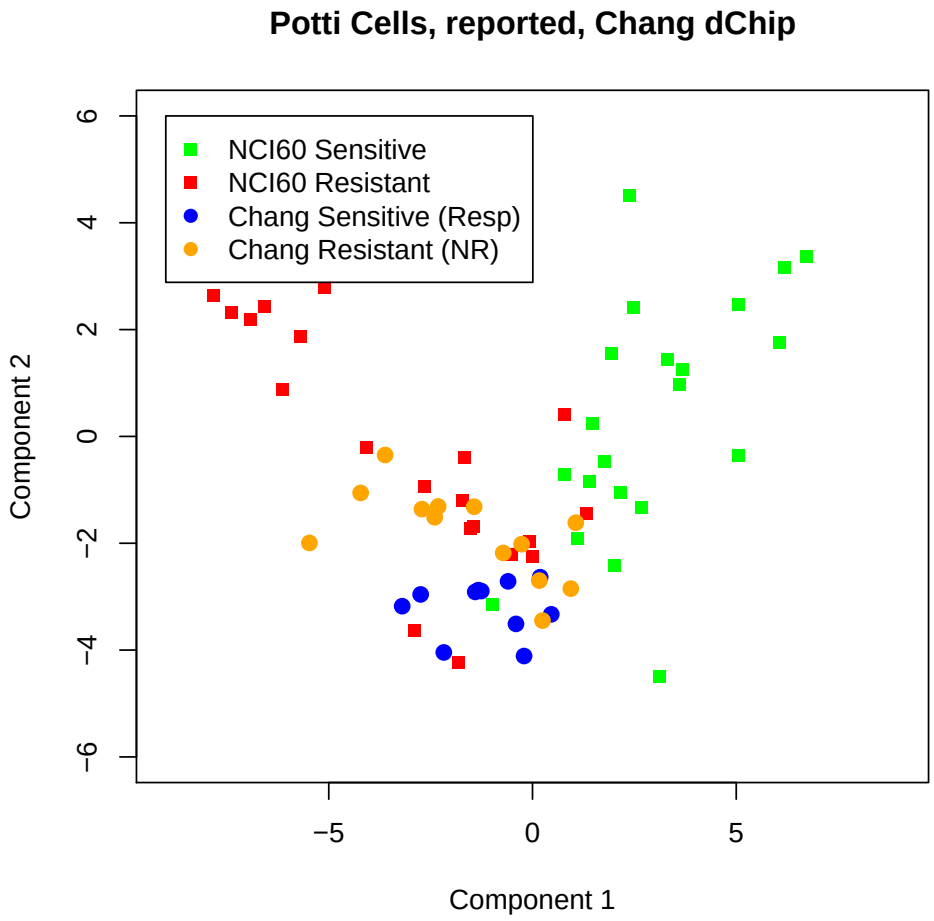

Figure 10: Plot of the first two principal components from the NCI60 training set, into which the Chang validation set has been projected.