Lesson 1: Review of topics

Objectives

By the end of this lesson you will have had the opportunity to:

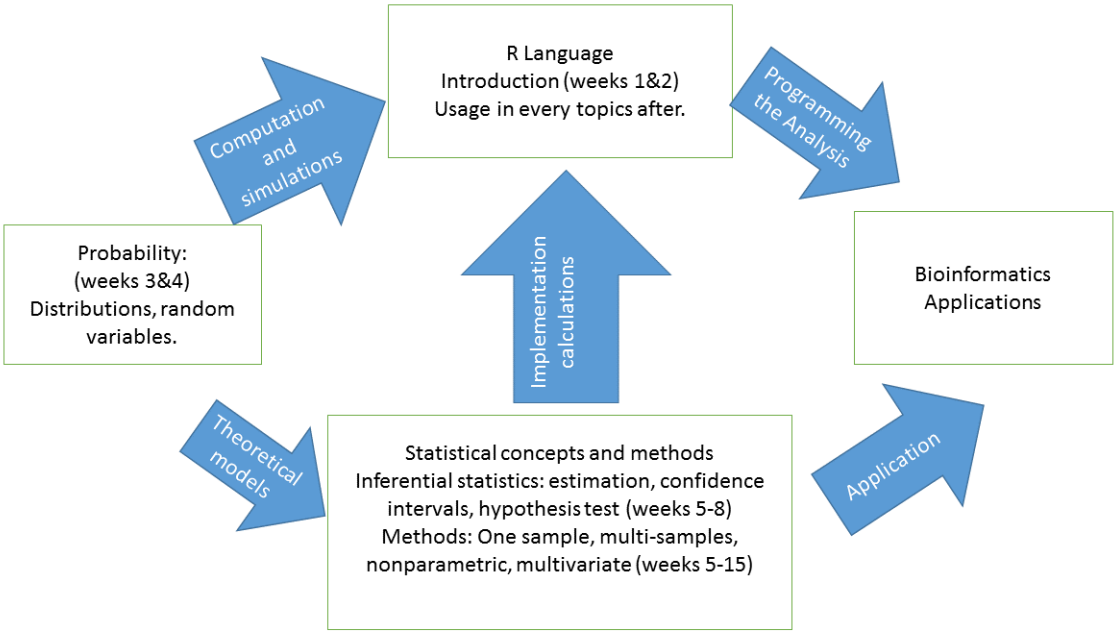
- Examine the relationship among topics covered in the course
- Apply statistical analysis on microarray data

Overview

In this lesson, we will review the topics taught in the previous modules, and provide some explanation on the relationship among the topics.

Then we use the Chiaretti, et al. (2004) analysis as an example to illustrate how to apply the methods covered in this course.

Course Review



This outline was provided at beginning of the course. The introduction of R and probability theory at the beginning was to provide foundations for understanding and using the statistical methods in the rest of the course. For bioinformatics applications, we focused on the microarray data analysis. Module 10 provided basics on R processing of the microarray data. Then we used those examples throughout the remainder of the class to illustrate the statistical methods covered.

Statistical Concepts and Probability Theory

We covered the three basic statistical inferences: point estimator, confidence interval and hypothesis test. Those inferences appear in every statistical model. The probability theory enables us to describe these concepts. We only require that you know enough probability theory to do corresponding statistical inferences.

(1) On the point estimator, we taught two common approaches:

- *maximum likelihood,* and
- *method of moments*

You should be able to write out the likelihood function (from the pdf in probability theory), and maximize it. You can often use the preprogrammed function in R to do this. For example, for the distribution $N(mean = \theta, sd = \theta)$ in the midterm exam, the pdf for one observation y when $\theta = x$ is simply dnorm(y, mean=x, sd=x) . For a random sample of more than one observation, y is a vector and you can get the likelihood (joint pdf) by taking the product of that expression. To find MLE, we numerically minimize the negative log-likelihood. The reason to do log-likelihood is that it will be sum instead of product. So the negative log-likelihood in this case is -sum(log(dnorm(y, mean=x, sd=x))). Numerical minimization gives the MLE. For the method of moments, it is necessary to match the moments. You should be able to calculate the moments (mean and variance) for common distributions and for linear combinations of random variables.

(2) For confidence intervals and hypothesis tests, you should know the definitions in terms of probability. You should be able to explain how the z-test and z-interval, t-test and t-interval formulas come from the probability theory. You should be able to convert between confidence intervals and hypothesis tests, which is a simple probability statement. $\alpha = P[\theta_0 \notin (1-\alpha)CI \mid H_0 : \theta = \theta_0] = P[reject \quad H_0 \mid H_0 : \theta = \theta_0]$. Binomial distribution is often useful to understand the empirical coverage probability (of CI). Also, you should know how to setup the hypothesis test: usually *the alternative hypothesis is what you want to prove*.

(3) Another important concept is the Monte Carlo simulation. This is the basis of many nonparametric procedures, including bootstrap and permutation tests. It is also used in some cross-validation procedures. You should be able to do each of these.

Statistical Procedures

We have covered many statistical methods. You need to know how, and *when* to use them.

(4) Nonparametric methods versus parametric methods. Exact tests versus asymptotic tests.

- Common nonparametric tests:
- Wilcoxon tests
- binomial tests
- Fisher's exact test, often by bootstrap
- permutation tests, and
- cross-validation

(5) An important concept in bioinformatics is the adjustment for multiple testing: false discovery rate. You should be able to explain why we need it, and be able to do it in R.

(6) Inferences on means and proportions:

- one sample
- two samples
- more than two samples (ANOVA)

(7) Linear Regression, correlation. Linear model (regression and ANOVA both in this form).

(8) Model checking in regression, ANOVA.

(9) Unsupervised learning: clustering and PCA. You should be able to state how and when to use them.

(10) Classification methods:

- logistic regression
- SVM, and
- classification tree

Other Skills

(11) Microarray data analysis in R. You should be able to preprocess the raw data in R. You should also be able to filter genes, and correctly use the linear model in R (use contrasts to build appropriate design matrix). You should be able to carry out a gene oncology search.

(12) Various graphic displays in R, particularly of multivariate data.

(13) Concepts like Type I/II error, power of a test. These are related to, in classification, the false positive rate, false negative rate, misclassification rates, etc. You should be able to state what they are and be able to calculate them. Often, we need to estimate them through simulation and/or cross-validation.

You should be familiar with the above topics listed. The final exam will be on those topics.

Next, we will use illustrate how to combine various statistical methods together for bioinformatics research. We shall use the ALL data set example frequently used in the textbook, and look at the statistical analysis done in the original paper by Chiaretti, et al. (2004) in the journal *Blood*.

Data Analysis in Chiaretti, et al. (2004)

We will review the data analysis in the paper using the ALL data set from the Bioconductor library ALL. The analysis is focused on the T-cell acute lymphocytic leukemia (T-ALL) patients, a subset of the data. Inspection of ALL$BT review shows that the first 95 patients are B-cell patients and the last 33 are T-cell patients. We will use the last 33 patients data set for analysis.

```
library("ALL"); data("ALL")
datmy<-exprs(ALL)     #Get expression data
datmy<-datmy[,96:128] #Get the last 33 patients (T,..., T4)
clmy<-as.character(ALL$BT[96:128]) #The class (T-T4) indicator
```

First, the authors applied a cluster analysis. The purpose of the cluster analysis is to explore the structure in the gene expression data, finding the naturally forming groups in the patients. Of course, we would hope that those natural groups correspond to the desired study outcome: predicting (classifying) the clinical outcome. While the clustering reveals some structure within the data, the structure in fact reflects the T-cell differentiation, not the clinical outcomes.

Second, the authors studied how to predict clinical outcomes from the gene expression data. This is a classification problem. The patients belong to two groups: those who responded to induction chemotherapy and those who did not. 25 patients responded, achieving complete remission (CR). The others were refractory or died during induction chemotherapy. The authors used classification tools to classify the patients into the two groups: CR or refractory.

Third, the authors further studied the 25 CR patients' outcomes: relapse or continuous complete remission (CCR). This is also a classification problem.

We look at the analyses one by one in next pages.

Gene Filter before Cluster Analysis in Chiaretti, et al. (2004)

There are 12625 genes in this data set. To avoid computation problems, and for interpretability, we generally wish to vastly reduce the number of genes used.

Gene filtering generally requires some consideration. We could filter for genes that are differentially expressed in the CR group versus the other group. However, the cluster analysis on those genes does not reveal the nature of structure in the data, since it was biased to the CR versus no-CR division. We need some neutral criterion to filter the genes.

The authors had two criteria:

(1) that the absolute expression levels were big enough;

(2) that the relative expression spread across the patients were big enough.

It is easy to understand the motivations for these criteria. There are measurement noises, so we need genes containing signals that are not likely overwhelmed by noise. There is a detection level below which measurements are unlikely to be very accurate, so criterion (1) is needed. The noise is often proportional to the measurement, and criterion (2) selects genes spread large enough in proportional terms. Precisely, the authors used (1) “the expression level to be higher than 100 in more than 20% of the samples”, and (2) “the ratio of the SD to the mean expression across samples to be between 0.7 and 10”. The ratio in criterion (2) is often called the coefficient of variance.

Let us try to repeat these filters. Unfortunately the data are not exactly the same. We will investigate this in more detail on next page.

Gene Filter, and Data Sets Difference?

Inspection of the expression data shows that the values are between 2.2 and 14.1

```
> range(datmy)
[1]  2.236158 14.126571
```

Clearly this is not the original data, as none is bigger than 100. The ALL dataset has gone through preprocessing (see Module 10), also it is likely to be on the logarithm scale. Since there is no detailed description, we cannot match the data to the raw data the authors used. Hence, we will not be able to reproduce the analysis results in the paper exactly. However, we will continue to apply the statistical procedures on our data set, which should have similar patterns of results. Comparing the CD2 expression values in Table 3 of the paper, with CD2 (probe 40738_at) expression values in our data set datmy["40738_at",], roughly the data in our data set is about the logarithm transformed with base 2.1. (The data still differ quite a bit, this just roughly match the magnitude.) Assuming this logarithm transformation, we apply the filters (1) and (2) from above.

```
library("genefilter")
f1 <- pOverA(.2, 100)  #select when at least .2 proportion >100
f2 <- cv(0.7, 10)  #select if coefficient of variance between 0.7 and 10.
sel1 <- genefilter(2.1^datmy, filterfun(f1)) #filter1, 2.1^ invert logarithm
transformation
sel2 <- genefilter(2.1^datmy, filterfun(f2)) #filter1, 2.1^ invert logarithm
transformation
dat0<-datmy[sel1 & sel2,] #gene data passing both filters. 338 selected
```

We can check that this selects 338 genes. Compared with the 313 genes listed in the supplemental materials of the paper, 209 genes are the same. So different preprocessing does change the filtered gene candidate set somewhat. But we can see the basic idea of selecting genes with (a) enough expression level and (b) enough expression spread still works on most genes. The 313 genes were copied from the PDF file into a text file “genes313.txt” attached below.

We continue to the next step of the analysis using the authors’ 313 genes list.

Cluster Analysis in Chiaretti, et al. (2004)

Now we apply the hierarchical clustering on the patients to look for nature groupings.

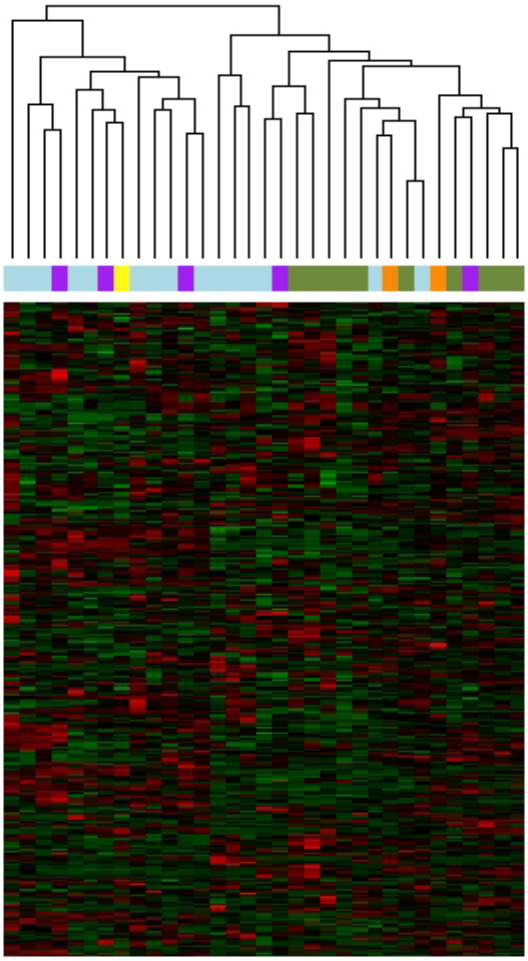
```
tmp0<-read.table("genes313.txt", sep=" ") #read the 313 genes
genedat<-tmp0[, 1:3] #Delete the empty 4th column (due to blank space at
end)
label313<-genedat[, 1] #probe names for the 313 genes
dat1=datmy[label313, ] #The 313 genes expression
hc.my <- hclust(dist(t(dat1)), method = "complete")
#The distance used in hcluster should be between data points
#  (rows/genes). dist() compute distance between columns. Hence
#  transpose t() to have the genes as rows in hclust(dist())
```

In previous modules we display the results of the clustering using the tree dendrogram. A commonly used graphic tool is to display the dendrogram with a heatmap. We reproduce the heatmap in Figure 1 of the paper.

```
library(gplots)
color.map <- function(T) {#color code for each type as in paper
 if (T=="T1") "yellow"
 else if(T=="T2") "lightblue"
 else if(T=="T3") "darkolivegreen4"
 else if(T=="T4") "darkorange"
 else "purple"
}
patientcolors<- unlist(lapply(clmy, color.map)) #apply color code
heatmap.2(dat1, #heatmap on dat1
 col=greenred(75), #use greenred color scheme for expression value
 scale="row",  #scale values by row, to show relative pattern
 dendrogram="column", #dendrogram for columns only
 ColSideColors=patientcolors, #apply color codes for patients
 margin=c(3, 12), cexRow=0.5, #margin: 3 on bottom, 12 on right.
 #for a clean graph,  we turn off many default displays below
 key=FALSE, trace="none", labRow=NA, labCol=NA)
```

On the next page, we will compare the heatmap with Figure 1 of the paper.

Figure 1 Heatmap for Cluster Analysis in Chiaretti, et al. (2004)
Our heatmap is very similar to the Figure 1 in original paper. All T4 (orange) and T3 (green) patients, and 6 of 15 T2 (blue) patients are put into the second cluster. The T1 (yellow) and 9 of 15 T2 (blue) patients are grouped in the first cluster.



Notice that this data set has only 1 T1 (yellow) patient compared to the 2 T1 patients in the original paper. The authors later changed one patient label from T1 to T (unknown type, colored purple here).
Other than the dendrogram, the heatmap itself shows some patterns, with two blocks of genes in the middle expressed higher (red) in the first cluster, alternating with some groups of genes expressed lower (green) in the first cluster. A hierarchical clustering was also done for the rows (genes) in the heatmap so that we may observe those block patterns. The default dendrogram="both" option would display row dendrogram also.

Comments on Heatmap R Commands

The heatmap is a rather standard data exploration tool for microarray data. It has many options and allows users to specify whether to add various display features. We have turned off many features for a bare-bones cleaner graph. You can try yourself to see what those various feature does. The graphic display capability in R is very rich. We do not intend, and are not able, to review every option in any R function. You should be able to learn the syntax yourself. The help feature in R is where you can start. For example, use ?heatmap.2 to learn about various options available within it. For example, if we do not want the default complete linkage clustering and want to use Ward linkage, then we specify the option
hclustfun=function(d) hclust(d,method="ward.D2")
inside the heatmap.2().

In this example we used another option scale="row". This scales the expression values within each row (gene) when displaying with the red-green color scheme. Here, red indicates relative higher expression values within this gene among 33 patients, not the high absolute expression value across genes. This allows the differential expression pattern of each gene to show up in the heatmap. So we can observe the genes expressing higher in one cluster. Otherwise, it is hard to observe the patterns. You can try not specifying this option; the resulting heatmap will show genes highly expressed and lowly expressed, but the pattern across patients is hard to see.

Testing the Association between Clusters and T-cells Differentiation

We can compare the two groups in the hierarchical clustering with the T-cell differentiation stages, using the confusion matrix

```
> groups.my <- cutree(hc.my, k = 2)
> table(groups.my, clmy)
          clmy
groups.my  T T1 T2 T3 T4
        1  2  0  6  10  2
        2  3  1  9   0  0
```

This association is tested by a Fisher's exact test in Chiaretti et al. (2004). We should exclude the unknown cases (labeled with "T") from the test.

```
> fisher.test(groups.my[clmy!="T"], clmy[clmy!="T"])$p.value
[1] 0.001809327
```

The p-value is very small, as reported in the original paper. So the natural grouping in the data reflects the degree of differentiation of leukemic cells. (A higher number indicates a higher degree of T-cell differentiation in T1, T2, T3 and T4.)

Classification of Clinical Outcomes in Chiaretti, et al. (2004)

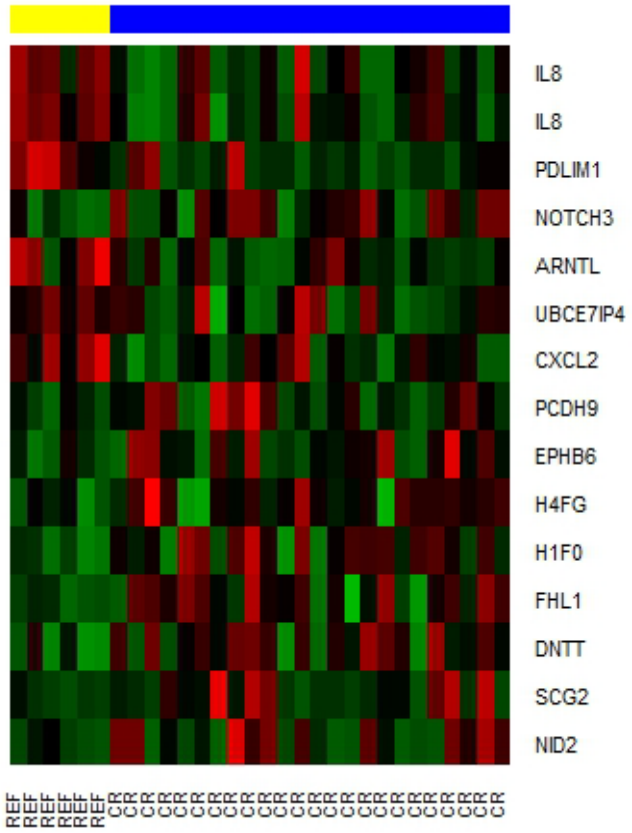
Now we will consider the second study in Chiaretti et al. (2004) which predicts the clinical outcomes from the gene expression data. Particularly, the authors are interested in using gene expression to predict response to chemotherapy: CR (complete remission) or refractory to induction therapy. While the paper reports 25 CR patients and 6 refractory patients, the data set in R has 24 CR patients and 6 refractory patients. Three patients belong to either category (paper reports 2 died during induction chemotherapy, the third case was likely changed at some point between the paper publication and data submission to R library). But let us follow their statistical analysis on the current available data.

The authors used two-sample t-tests (at 0.05 level) to filter genes expressed differently in the CR versus refractory groups. They added a twist, applying delete-one-cross validation, and kept the genes passing the t-test filter at least 75% of times. We repeat that on this data set.

```
crmy<-as.character(ALL$remission[96:128]) #CR/REF indicator on 33 patients
ind.TNA<-(!is.na(crmy)) #indicator for non-NA in crmy
crmy<-crmy[ind.TNA]     #(non-NA cases) 24CR + 6REF: 30 patients
datCR1<-dat1[, ind.TNA] #The 313 genes expression on 30 patients
###### Select genes with t-test
n.gene<-dim(datCR1)[1]     # number of predictors (313 genes)
n.sample<-dim(datCR1)[2] # number of patients (30)
##Create a matrix to store selected variable in delete-one cases
selected.M<-matrix(NA, nrow=n.gene, ncol=n.sample)
for(icv in 1:n.sample){ #iterate for delete-one-validation
  dat.tmp=datCR1[,-icv] # leave one out
  f2<-function(x) (t.test(x ~ crmy[-icv])$p.value < 0.05)
  selected<- genefilter(dat.tmp, filterfun(f2))  #pass t-tests
  selected.M[,icv]=selected #save selected gene in the iteration
}
# Percentage of selection
sel.freq<-apply(selected.M,1,function(x) sum(x))
selected1<-(sel.freq>n.sample*0.75) #genes selected >75% times
sum(selected1) #[1] 15
```

Using these genes we can produce a heatmap similar to Figure 2 in the paper.

Classification of Clinical Outcomes in Chiaretti, et al. (2004)



We see the similar patterns as Figure 2 of the original paper, with the two IL8 gene probes showing a higher expression in refractory patients.

Due to the differences in the data sets (one less CR patient, different preprocessing of the expression values), we do not get the same gene set as in the original paper, but 9 out of our 15 genes do appear in Figure 2 of the paper.

The R commands for producing this heatmap are provided on the next page.

R Commands for Producing the Heatmap

```
########## Commands to draw the heatmap#############
sel.probe <- genedat[selected1,1] #15 selected genes probe names
sel.gene <- genedat[selected1,3]  #15 selected genes symbols
datFig2 <- datCR1[selected1,]     #15 selected genes expression
row.names(datFig2)<- sel.gene  #gene names as row names
colnames(datFig2) <-crmy  #class (REF/CR) labels as column names

col.ord<-order(crmy, decreasing=TRUE) #Order REF/CR cases
datFig2<-datFig2[, col.ord]  #Reorder columns, 6 REF cases first
crmy1 <- crmy[col.ord]       #Reorder class labels

row.ord<-order(datFig2[,1], decreasing=TRUE)
datFig2<-datFig2[row.ord,] #reorder rows by 1st patient's values

color.map <- function(x) {#color map to match Figure 2.
  if (x=="REF") "yellow"
  else "blue"
}
patientcolor1<- unlist(lapply(crmy1, color.map)) #apply color scheme

heatmap.2(datFig2, col=greenred(75), margin=c(3, 12), scale="row",
Rowv=FALSE, Colv=FALSE, key=FALSE, ColSideColors=patientcolor1,
trace="none", dendrogram="none")

heatmap.2(datFig2, #heatmap on datFig2
  col=greenred(75), #use greenred color scheme for expression value
  scale="row",  #scale values by row, to show relative pattern
  dendrogram="none", #no dendrogram
  ColSideColors=patientcolor1, #apply color codes for patients
  margin=c(3, 12), #margin: 3 on bottom, 12 on right.
  key=FALSE, trace="none", #turn off extra displays
  Rowv=FALSE, Colv=FALSE) #no reordering, we already ordered them
```

Classification of Clinical Outcomes in Chiaretti, et al. (2004)

Next, the authors used delete-one-validation to select the number of predictor genes similar to what we did in module 13, except that the extra step of t-test filtering was also included in the cross-validation process. They used the linear discriminant analysis (LDA), which is a tool similar to SVM, mentioned in textbook Statistics Using R with Biological Examples pages 304-306. Since we did not do LDA in module 13, we will use SVM here instead. You can try to use lda() in the R library MASS yourself, and see the results are similar to SVM results.

```
library(e1071)  #load library for SVM
n.k<-dim(datFig2)[1]  #number of selected predictors
n.cv<-dim(datFig2)[2] #number of CV iteration=sample size
###Create a data frame, y: CR/REF indicator, x: gene expresions
dat.svm<-data.frame(x=t(datFig2),y=crmy1)
#Matrices to save cross-validated misclassification rates
mcr.svm.M<-matrix(NA, nrow=n.k, ncol=n.cv)
for(icv in 1:n.cv){#iterate for delete-one-validation
  dat.tmp<-dat.svm[-icv,] #Leave one out
  p.t <-function(x) t.test(x~dat.tmp$y)$p.value #function calculate t-test p-values
  p.value<-apply(dat.tmp[,1:n.k], 2, p.t) #p-value (delete i-th)
  for(k in 1:n.k){
    topknames<-names(dat.tmp[,1:n.k])    #get (all) gene names
    topknames<-topknames[order(p.value)] #order by p-value
    topknames<-topknames[1:k]      #select top k gene names
    #get the model formula using top k genes
    fml<-as.formula(paste("y~",paste(topknames,collapse="+")))
    svm.fit<-svm(fml, data=dat.tmp, type = "C-classification", kernel = "linear")  #fit SVM
    svm.pred<-predict(svm.fit,dat.svm[icv,]) #SVM prediction
    mcr.svm.M[k,icv]<- (svm.pred !=dat.svm$y[icv]) #mcr for this iteration
  }
}
mcr.svm <- apply(mcr.svm.M,1,mean) #mcr by k=number of predictors
plot(1:n.k, mcr.svm, xlab="K, number of predictors")
k.best<-which.min(mcr.svm) #find k with minimal mcr
k.best
```

We find that the best predictive model in our data set uses five genes.

Classification of Clinical Outcomes in Chiaretti, et al. (2004)

```
###Find the k.best genes
mcr.best.V<-rep(NA, n.cv) #vector to save cross-validated mcrs
gene.best<-NULL
for(icv in 1:n.cv){#delete-one cross validation
  k<-k.best #fix k to the selected best number
  #then repeat previous procedure
  dat.tmp<-dat.svm[-icv,]
  p.t <-function(x) t.test(x~dat.tmp$y)$p.value
  p.value<-apply(dat.tmp[,1:n.k], 2, p.t)
  topknames<-names(dat.tmp[,1:n.k])
  topknames<-topknames[order(p.value)]
  topknames<-topknames[1:k]
  fml<-as.formula(paste("y~",paste(topknames,collapse="+")))
  svm.fit<-svm(fml, data=dat.tmp, type = "C-classification", kernel =
"linear")
  svm.pred<-predict(svm.fit,dat.svm[icv,])
  mcr.best.V[icv]= (svm.pred !=dat.svm$y[icv])
  gene.best<-topknames
}
mcr.best<-mean(mcr.best.V)
gene.best #[1] "x.H1F0"   "x.IL8.1"  "x.FHL1"   "x.SCG2"   "x.NOTCH3"
mcr.best #[1] 0.06666667
```

We find the best five genes model uses H1F0, IL8, FHL1, SCG2 and NOTCH3 genes, with a delete-one-validated misclassification rate of 6.7%.
We can then search literature on these five genes using methods taught in module 10.
All five genes except SCG2 were included in the 25 genes model of Chiaretti et al. (2004).

Classification of Clinical Outcomes in Chiaretti, et al. (2004)

Using our best five genes model on the data, set we can find the confusion matrix.

```
> fml<-as.formula(paste("y~",paste(gene.best,collapse="+")))
> svm.fit<-svm(fml, data=dat.tmp, type = "C-classification", kernel =
"linear")
> svm.pred<-predict(svm.fit,dat.svm)
> table(svm.pred, dat.svm$y)
svm.pred CR REF
    CR      24   1
    REF      0   5
```

This is much more accurate than the performance of the 25 genes model reported in the Chiaretti et al. (2004), "where 19 of the 25 responders and 4 of the 6 refractory patients were correctly predicted".

This is due to the difference in the data sets (the one we are using was preprocessed differently), mentioned earlier.

If we use LDA instead of SVM on these these genes, we get the same classification performance.

Further Analysis in Chiaretti, et al. (2004)

The authors further studied the relationship between gene expression and long-term clinical trials outcomes (whether patients experienced a relapse after CR). This is more complicated than classifying into two classes: relapse or continuous complete remission (CCR). The relapse times provide further information. Also, the CCR patients' duration of CR is different at the end of study (since their chemotherapy start time and time to remission are different.) It is possible that if observed for longer, the patients later could go into relapse. Those are called *censored* data: we only know that relapse has not occurred at certain duration time. Survival analysis is the statistical method to deal with this type of issues. The authors fit the data by the Cox proportional hazards model, which is a generalized regression model. As we did not cover survival analysis, we will not repeat this part of their analysis. Interested students may learn the Cox hazards model fit in R on their own.

Summary Comparison to Chiaretti, et al. (2004)
We will review the analysis by Chiaretti et al. This example shows how to apply statistical methods for microarray analysis. The t-tests, clustering, cross-validation, et al. are statistical procedures you need to become familiar in usage.

We did not get the same numerical results, mainly due to the differently preprocessed data. But we see that the general pattern still holds. The field is developing fast; the preprocessing is being standardized by a few methods in R. (Roughly speaking, the original data analysis seems to be done on the original scale without much normalization.) Also, the gene filter using a t-test at level 0.05 seems overly loose, without adjusting for multiple testing. The false discovery rate control started to become popular in such analysis around the time of this paper's publication. So, continuous learning of new advances in this field may well be needed in the future. This course provides you a foundation for understanding the basics of statistical inferences, so that you can continue on to learning more advanced methods.

Lesson Summary

This lesson reviewed the statistical topics taught in this class. We also illustrated the use of these methods on a microarray data set.