

## College Data Example

We will look at a dataset containing data on colleges in the year 2000. We have the following 10 variables:

- (1) Name of the college
- (2) Academic reputation (0-5 scale)
- (3) 1998 actual graduation rate
- (4) Student/faculty ratio (# of students per faculty)
- (5) 25th percentile of SAT scores
- (6) 75th percentile of SAT scores
- (7) % of freshmen in the top 10% of their high school class
- (8) Acceptance rate (ratio of students admitted to applicants)
- (9) Financial resources rank
- (10) Type (1=Private, 2=Public)

Read the data file "college2000.txt" in to R. This is accomplished with the following command (assuming this file is in the R working directory).

```
T = read.table(file="college2000.txt", head=TRUE, row.name=1)
```

1. Begin by creating a function that takes a sample correlation coefficient and a sample size and returns an approximate 95% CI:

```
get.CI = function(rhat, n)
{
  logterm = 0.5*log(1+rhat)-0.5*log(1-rhat)
  CL = logterm-2/sqrt(n-3)
  CU = logterm+2/sqrt(n-3)
  LB = (exp(2*CL)-1)/(exp(2*CL)+1)
  UB = (exp(2*CU)-1)/(exp(2*CU)+1)
  return(c(LB,UB))
}
```

2. Calculate the sample mean acceptance ratio for both private colleges and public colleges.

```
## Calculate the mean acceptance ratio for type=Private
mean(T$Accept.Ratio[T$Type==1])
```

```
## Calculate the mean acceptance ratio for type=Public
mean(T$Accept.Ratio[T$Type==2])
```

3. Make a scatter plot of graduation rate versus student to faculty ratio:

```
plot(T$Grad.rate, T$S.F.Ratio, type="p", pch=1,
      xlab="Graduation Rate", ylab="Students/Faculty")
```

4. Compute an approximate 95% CI for the correlation coefficient between variables graduation rate and student to faculty ratio.

```
## Calculate the sample correlation coef. between Grad.rate
## and S.F.Ratio
rhat=cor(T$Grad.rate, T$S.F.Ratio)
n=length(T$Grad.rate)
CI=get.CI(rhat,n)

## Approximate 95% CI for r
print(CI)
```

5. Repeat the last two exercises but do it separately for public and private schools.

```
## Do this separately for type=Public, type=Private
## Make Scatterplots
par(mfrow=c(1,2))
plot(T$Grad.rate[T$Type==1], T$S.F.Ratio[T$Type==1], type="p", pch=1,
      xlab="Graduation Rate", ylab="Students/Faculty", main="Private")
plot(T$Grad.rate[T$Type==2], T$S.F.Ratio[T$Type==2], type="p", pch=1,
      xlab="Graduation Rate", ylab="Students/Faculty", main="Public")

## Private
rhat=cor(T$Grad.rate[T$Type==1], T$S.F.Ratio[T$Type==1])
n=length(T$Grad.rate[T$Type==1])
CI=get.CI(rhat,n)

## Private Approximate 95% CI for r
print(CI)

## Public
rhat=cor(T$Grad.rate[T$Type==2], T$S.F.Ratio[T$Type==2])
n=length(T$Grad.rate[T$Type==2])
CI=get.CI(rhat,n)

## Public Approximate 95% CI for r
print(CI)
```

## Gene-Microarray Example

Colon adenocarcinoma tissue samples were collected, 40 of which were tumor tissues and 22 non-tumor tissues. Tissue samples were analyzed using an Affymetrix oligonucleotide array. We have 2,001 variables where the first 2,000 are gene expression values and the last variable is the tissue class (1=tumor, 2=non-tumor).

1. Begin by reading the file "genes.txt" in to R.

```
G=read.table("genes.txt")
```

2. A common approach for selecting significant genes is to compute a 2-sample t-statistic for each gene, and keep the genes with the largest t-statistic magnitude. Let's do this here:

- (a) Create a 2,000 length vector of t-statistics for the 2,000 genes.

```
## Get rows for tumor/non-tumor classes
c1=which(G[,2001] == 1)
c2=which(G[,2001] == 2)

## Get Sample Sizes
n1=length(c1)
n2=length(c2)

## Make a table with the class variable removed
GG=G[,-2001]

## Get Sample Variances
var1=apply(GG[c1,],2,var)
var2=apply(GG[c2,],2,var)

## Get Sample Means
mean1=apply(GG[c1,],2,mean)
mean2=apply(GG[c2,],2,mean)

## Compute the t-statistics
t.den=sqrt(var1/n1+var2/n2)
t.num=mean1-mean2
t.stats=t.num/t.den
```

- (b) Create a new data-table with the class variable in the first column and the 3 most significant genes based on t-statistic magnitude in columns 2-4.

```
## Get the variable ordering based on t-stat magnitude
v.order=order(abs(t.stats),decreasing=TRUE)
Gsmall=G[,c(2001,v.order[1:3])]

## Label the columns
colnames(Gsmall) = c("Class", "G1", "G2", "G3")
```

3. Using this small table, examine scatter plots between gene variables.

```
pairs(Gsmall)
```

4. Make a scatter-plot between the two most significant genes, where tumor-tissues are black circles and non-tumor-tissues are red triangles.

```
colors=Gsmall$class
points=Gsmall$class
plot(Gsmall$G1, Gsmall$G2, type="p", col=colors, pch=points,
      xlab="Gene 1", ylab="Gene 2")
legend(x=10,y=500,legend=c("Tumor", "non-Tumor"), col=c(1,2), pch=c(1,2))
```

5. Repeat the last exercise but take the log transform of the first two genes:

```
colors=Gsmall$class
points=Gsmall$class
plot(log(Gsmall$G1), log(Gsmall$G2), type="p", col=colors, pch=points,
      xlab="Gene 1", ylab="Gene 2")
legend(x=2.5,y=6,legend=c("Tumor", "non-Tumor"), col=c(1,2), pch=c(1,2))
```

## A Plotting Example

1. Read the data file "AnnArbor.txt" in to R. For the first 365 days, plot the recorded minimum, mean, and high temperatures.

```
Z=read.table("AnnArbor.txt",sep=",", row.names=NULL, header=T)

## Get min, mean, max temps for first 365 days
min.temp = Z[1:365,4]
mean.temp = Z[1:365,3]
max.temp = Z[1:365,2]

## Compute the y-min and y-max
y.min = min(min.temp)
y.max = max(max.temp)

plot(1:365, min.temp, type="l",lwd=2, col="blue", ylim=c(y.min,y.max),
      xlab="Day", ylab="Temperature F", main="Ann Arbor Temperature 2001",
      axes=FALSE)
lines(1:365, mean.temp, type="l", col="black")
lines(1:365, max.temp, type="l",lwd=2, col="red")
axis(at=seq(-20, 100, by=10), side=2, lwd=2, pos=1)
axis(at=c(1,seq(50,300,by=50),365), side=1,lwd=2)
legend(x=150,y=20,legend=c("Max", "Mean", "Min"),
      col=c("red","black", "blue"), lty=c(1,1,1), lwd=c(2,1,2), bty="n")
```

2. Repeat the above for the third set of 365 days.

```
## Get min, mean, max temps for the third year
inds = 731:1095
min.temp = Z[inds,4]
mean.temp = Z[inds,3]
max.temp = Z[inds,2]

## Compute the y-min and y-max
y.min = min(min.temp)
y.max = max(max.temp)

plot(1:365, min.temp, type="l",lwd=2, col="blue", ylim=c(y.min,y.max),
     xlab="Day", ylab="Temperature F", main="Ann Arbor Temperature Year 3",
     axes=FALSE)
lines(1:365, mean.temp, type="l", col="black")
lines(1:365, max.temp, type="l",lwd=2, col="red")
axis(at=seq(-20, 100, by=10), side=2,lwd=2, pos=1)
axis(at=c(1,seq(50,300,by=50),365), side=1,lwd=2)
legend(x=150,y=20,legend=c("Max", "Mean", "Min"),
      col=c("red","black", "blue"), lty=c(1,1,1), lwd=c(2,1,2), bty="n")
```

3. Save the above plot as a pdf file, make sure you close the plot window before running the following code:

```
## Save the plot as a pdf:
## make sure any open plots are closed before running this:

pdf(file="aatemp.pdf", width=10, height=5)
plot(1:365, min.temp, type="l",lwd=2, col="blue", ylim=c(y.min,y.max),
     xlab="Day", ylab="Temperature F", main="Ann Arbor Temperature Year 3",
     axes=FALSE)
lines(1:365, mean.temp, type="l", col="black")
lines(1:365, max.temp, type="l",lwd=2, col="red")
axis(at=seq(-20, 100, by=10), side=2,lwd=2, pos=1)
axis(at=c(1,seq(50,300,by=50),365), side=1,lwd=2)
legend(x=150,y=20,legend=c("Max", "Mean", "Min"),
      col=c("red","black", "blue"), lty=c(1,1,1), lwd=c(2,1,2), bty="n")
dev.off()
```

## Taking means, variances, etc with missing data.

In order to handle missing data, use the `na.rm=TRUE` argument to functions acting on these data.