

1. (70 points) Analysis of the ALL data set.

(a) Define an indicator variable `IsB` such that `IsB=TRUE` for B-cell patients and `IsB=FALSE` for T-cell patients.

(b) Use two genes `"39317_at"` and `"38018_g_at"` to fit a classification tree for `IsB`. Print out the confusion matrix. Plot ROC curve for the tree.

(c) Find its empirical misclassification rate (`mcr`), false negative rate (`fnr`) and specificity. Find the area under curve (`AUC`) for the ROC curve.

(d) Use 10-fold cross-validation to estimate its real false negative rate (`fnr`). What is your estimated `fnr`?

(e) Do a logistic regression, using genes `"39317_at"` and `"38018_g_at"` to predict `IsB`. Find an 80% confidence interval for the coefficient of gene `"39317_at"`.

(f) Use n-fold cross-validation to estimate misclassification rate (`mcr`) of the logistic regression classifier. What is your estimated `mcr`?

(g) Conduct a PCA on the scaled variables of the whole ALL data set (NOT just the two genes used above). We do this to reduce the dimension in term of genes (so this PCA should be done on the transpose of the matrix of expression values). To simplify our future analysis, we use only the first `K` principal components (`PC`) to represent the data. How many `PCs` should be used? Explain how you arrived at your conclusion. Provide graphs or other R outputs to support your choice.

(h) Do a SVM classifier of `IsB` using only the first five `PCs`. (The number `K=5` is fixed so that we all use the same classifier. You do not need to choose this number in the previous part (g).) What is the sensitivity of this

(i) Use leave-one-out cross-validation to estimate misclassification rate (`mcr`) of the SVM classifier. Report your estimate.

(j) If you had to choose between classifiers in part (e) and in part (h), which one would you choose? Why?

You should put answers to the questions in the PDF file. That means, for (a), provide the R command; for (b), provide the printout and the plot; for (c) provide the numerical answers; et al. Remember to answer each question directly. The grader should not have to pick out the numerical answers from the R outputs.

The R commands that you used to get those printout/plots et al. should be submitted in the separate R script file.

A)

Rscript:

```
library(ALL);data(ALL)
```

```
IsB <- factor(ALL$BT %in% c("B","B1","B2","B3","B4"))
IsB
```

Answer:

```
[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE
[14] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[27] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[40] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[53] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[66] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[79] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE
[92] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE
[105] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE
[118] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE
Levels: FALSE TRUE
```

B)

Rscript:

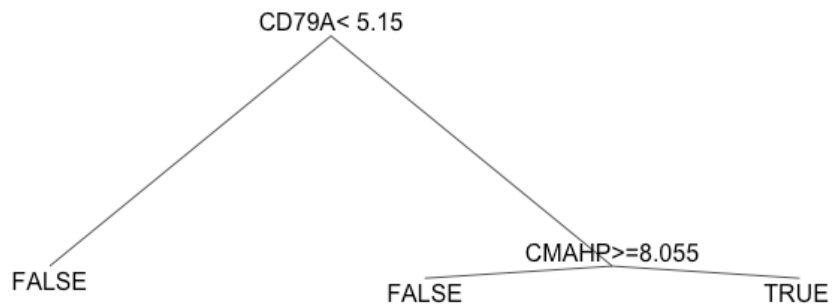
```
install.packages('rpart')
library("hgu95av2.db")
library(ALL);data(ALL)

names <- c("39317_at", "38018_g_at")
expr.data <- exprs(ALL)[names,]
symb <- mget(names, env = hgu95av2SYMBOL)
ALLBTnames <- ALL[names, ]
probedat <- as.matrix(exprs(ALLBTnames))
row.names(probedat) <- unlist(symb)

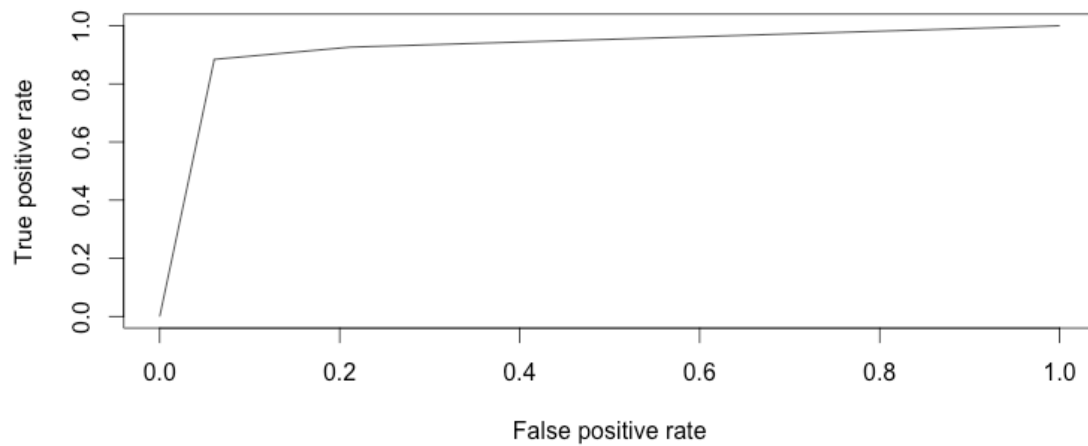
require(rpart)
B.stage <- factor(IsB)
c.tr <- rpart(B.stage~., data = data.frame(t(probedat)))
plot(c.tr, branch=0,margin=0.1)
text(c.tr, digits=3,)
rpartpred <- predict(c.tr, type="class")
table(rpartpred, B.stage)
```

```
install.packages('ROCR')
library("ROCR")
pred.prob <- predict(c.tr, type="prob")[,2]
pred <- prediction(pred.prob, lsB=="TRUE")
perf <- performance(pred,"tpr","fpr")
plot(perf)
```

Answer:



```
B.stage
rpartpred FALSE TRUE
FALSE 31 11
TRUE 2 84
```



C)

Rscript:

```
mcr<-(11+2)/(31+11+2+84)
mcr
fnr<-11/(11+84)
fnr
spe<-31/(2+31)
spe
performance(pred,"auc")
```

Answer:

```
empirical misclassification rate (mcr) = 0.1015625
false negative rate (fnr) = 0.1157895
specificity = 0.9393939
area under curve (AUC) for the ROC curve = 0.922807
Slot "y.values":
[[1]]
[1] 0.922807
```

D)

Rscript:

```
require(caret)
n <- length(rpartpred)
index <- 1:n
K <- 10
folds <- createFolds(index, k=K)
fnr.cv.raw <- rep(NA, K)
for (i in 1:K) {
  testID <- folds[[i]]
  c.tr <- rpart(IsB[-testID]~., data=data.frame(t(expr.data)[-testID,]))
  tr.pred <- predict(c.tr, newdata=data.frame(t(expr.data)[-testID,]), type="class")
  fnr.cv.raw[i] <- mean(tr.pred[testID] == 'FALSE' & IsB[testID] ==
'TRUE')/mean(IsB[testID] == 'TRUE')
}
fnr.cv <- mean(fnr.cv.raw)
fnr.cv
```

Answer:

```
estimated fnr: 0.1630556
```

E)

Rscript:

```
prob.name <- c("39317_at", "38018_g_at")
expr.data <- exprs(ALL)[prob.name,]
data.lgr <- data.frame(IsB, t(expr.data))
fit.lgr <- glm(IsB~., family=binomial(link='logit'), data=data.lgr)
```

```

pred.prob <- predict(fit.lgr, data=data.lgr$expr.data, type="response")
pred.B1 <- factor(pred.prob> 0.5, levels=c(TRUE,FALSE), labels=c("Bcell","not
Bcell"))
IsBcell <- factor(IsB, levels=c(TRUE,FALSE), labels=c("Bcell","not Bcell"))
table(pred.B1, IsBcell)

```

```

y <- as.numeric(IsB==TRUE)
data.CI <- exprs(ALL)["39317_at", ]
data <- glm(data.frame(y, data.CI))
confint(data, level=0.8)

```

Answer:

Logistic regression:

	IsBcell	
pred.B1	Bcell	not Bcell
Bcell	90	6
not Bcell	5	27

80% confidence interval for "39317_at"

	10 %	90 %
(Intercept)	1.7157421	2.1038847
data.CI	-0.2039277	-0.1469204

80% CI for $B_0 = (1.72, 2.01)$

80% CI for $B_1 = (-0.2, -0.15)$

F)

Rscript:

```

install.packages('caret');
require(caret);
data.lgr <- data.frame(IsB,t(expr.data))
n <- dim(data.lgr)[1]
index <- 1:n
K <- 10
flds <- createFolds(index, k=K)
mcr.cv.raw <- rep(NA, K)
for (i in 1:K) {
  testID <- flds[[i]]
  data.tr <- data.lgr[-testID,]
  data.test <- data.lgr[testID,]
  fit.lgr <- glm(IsB~., family=binomial(link='logit'), data=data.tr)
  pred.prob <- predict(fit.lgr, newdata=data.test, type="response")
  pred <- (pred.prob> 0.5)
}

```

```

mcr.cv.raw[i] <- sum(pred!=data.test$IsB)/length(pred)
}
mcr.cv <- mean(mcr.cv.raw)
mcr.cv

```

Answer:

estimated mcr = .09386447 = 9.4%

G)

Rscript:

```

pca.all <- prcomp(t(exprs(ALL)), scale=TRUE)
summary(pca.all)
PropVar <- summary(pca.all)$importance[2,]
plot(1:length(PropVar), PropVar, xlab='number of principal components',
ylab='proportion of variance explained',cex=0.3)

```

Answer:

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	47.8103	36.9157	27.73208	24.0204	21.29449	19.64675	18.00937
Proportion of Variance	0.1811	0.1079	0.06092	0.0457	0.03592	0.03057	0.02569
Cumulative Proportion	0.1811	0.2890	0.34991	0.3956	0.43153	0.46211	0.48780

	PC8	PC9	PC10	PC11	PC12	PC13	PC14
Standard deviation	16.52815	16.08110	15.68492	14.73970	13.49120	13.46128	13.1528
Proportion of Variance	0.02164	0.02048	0.01949	0.01721	0.01442	0.01435	0.0137
Cumulative Proportion	0.50943	0.52992	0.54940	0.56661	0.58103	0.59538	0.6091

	PC15	PC16	PC17	PC18	PC19	PC20	PC21
Standard deviation	12.50326	11.62651	11.33948	10.95969	10.56977	10.27269	9.98280
Proportion of Variance	0.01238	0.01071	0.01018	0.00951	0.00885	0.00836	0.00789
Cumulative Proportion	0.62147	0.63217	0.64236	0.65187	0.66072	0.66908	0.67698

	PC22	PC23	PC24	PC25	PC26	PC27	PC28	PC29
Standard deviation	9.76071	9.69351	9.35307	9.07879	8.97473	8.85997	8.72421	8.59119
Proportion of Variance	0.00755	0.00744	0.00693	0.00653	0.00638	0.00622	0.00603	0.00585
Cumulative Proportion	0.68452	0.69196	0.69889	0.70542	0.71180	0.71802	0.72405	0.72989

	PC30	PC31	PC32	PC33	PC34	PC35	PC36	PC37
--	------	------	------	------	------	------	------	------

Standard deviation	8.53111	8.27069	8.23309	8.08897	8.07028	7.83775	7.80235
	7.7043						
Proportion of Variance	0.00576	0.00542	0.00537	0.00518	0.00516	0.00487	0.00482
	0.0047						
Cumulative Proportion	0.73566	0.74108	0.74645	0.75163	0.75679	0.76165	
	0.76648	0.7712					

	PC38	PC39	PC40	PC41	PC42	PC43	PC44	PC45
Standard deviation	7.56167	7.54920	7.47852	7.40003	7.33338	7.21207	7.18165	
	7.07957							
Proportion of Variance	0.00453	0.00451	0.00443	0.00434	0.00426	0.00412	0.00409	
	0.00397							
Cumulative Proportion	0.77571	0.78022	0.78465	0.78899	0.79325	0.79737		
	0.80145	0.80542						

	PC46	PC47	PC48	PC49	PC50	PC51	PC52	PC53
Standard deviation	7.00761	6.88692	6.86142	6.81489	6.79102	6.70541	6.68737	
	6.61719							
Proportion of Variance	0.00389	0.00376	0.00373	0.00368	0.00365	0.00356	0.00354	
	0.00347							
Cumulative Proportion	0.80931	0.81307	0.81680	0.82048	0.82413	0.82769		
	0.83123	0.83470						

	PC54	PC55	PC56	PC57	PC58	PC59	PC60	PC61
Standard deviation	6.58283	6.51855	6.4543	6.42488	6.40876	6.33493	6.2549	
	6.22671							
Proportion of Variance	0.00343	0.00337	0.0033	0.00327	0.00325	0.00318	0.0031	
	0.00307							
Cumulative Proportion	0.83813	0.84150	0.8448	0.84807	0.85132	0.85450	0.8576	
	0.86067							

	PC62	PC63	PC64	PC65	PC66	PC67	PC68	PC69
Standard deviation	6.19791	6.17728	6.13009	6.07863	6.0457	6.00987	5.98143	
	5.95744							
Proportion of Variance	0.00304	0.00302	0.00298	0.00293	0.0029	0.00286	0.00283	
	0.00281							
Cumulative Proportion	0.86371	0.86674	0.86971	0.87264	0.8755	0.87839	0.88123	
	0.88404							

	PC70	PC71	PC72	PC73	PC74	PC75	PC76	PC77
Standard deviation	5.87113	5.84515	5.82817	5.76546	5.74950	5.69443	5.67019	
	5.66516							
Proportion of Variance	0.00273	0.00271	0.00269	0.00263	0.00262	0.00257	0.00255	
	0.00254							
Cumulative Proportion	0.88677	0.88948	0.89217	0.89480	0.89742	0.89999		
	0.90253	0.90508						

	PC78	PC79	PC80	PC81	PC82	PC83	PC84	PC85
Standard deviation	5.64871	5.60202	5.56279	5.53503	5.5092	5.48405	5.44800	
	5.42787							
Proportion of Variance	0.00253	0.00249	0.00245	0.00243	0.0024	0.00238	0.00235	
	0.00233							

Cumulative Proportion 0.90760 0.91009 0.91254 0.91497 0.9174 0.91975 0.92210
0.92444

PC86 PC87 PC88 PC89 PC90 PC91 PC92 PC93
Standard deviation 5.36975 5.33498 5.29293 5.28715 5.25939 5.22066 5.18953
5.17319
Proportion of Variance 0.00228 0.00225 0.00222 0.00221 0.00219 0.00216 0.00213
0.00212

Cumulative Proportion 0.92672 0.92898 0.93119 0.93341 0.93560 0.93776
0.93989 0.94201

PC94 PC95 PC96 PC97 PC98 PC99 PC100 PC101
Standard deviation 5.1529 5.10942 5.09202 5.07675 5.03399 5.0260 4.99505
4.96053
Proportion of Variance 0.0021 0.00207 0.00205 0.00204 0.00201 0.0020 0.00198
0.00195

Cumulative Proportion 0.9441 0.94618 0.94824 0.95028 0.95228 0.9543 0.95626
0.95821

PC102 PC103 PC104 PC105 PC106 PC107 PC108 PC109
Standard deviation 4.92004 4.8988 4.86395 4.85237 4.81879 4.80002 4.72967
4.69107
Proportion of Variance 0.00192 0.0019 0.00187 0.00186 0.00184 0.00182 0.00177
0.00174

Cumulative Proportion 0.96013 0.9620 0.96390 0.96577 0.96761 0.96943 0.97120
0.97295

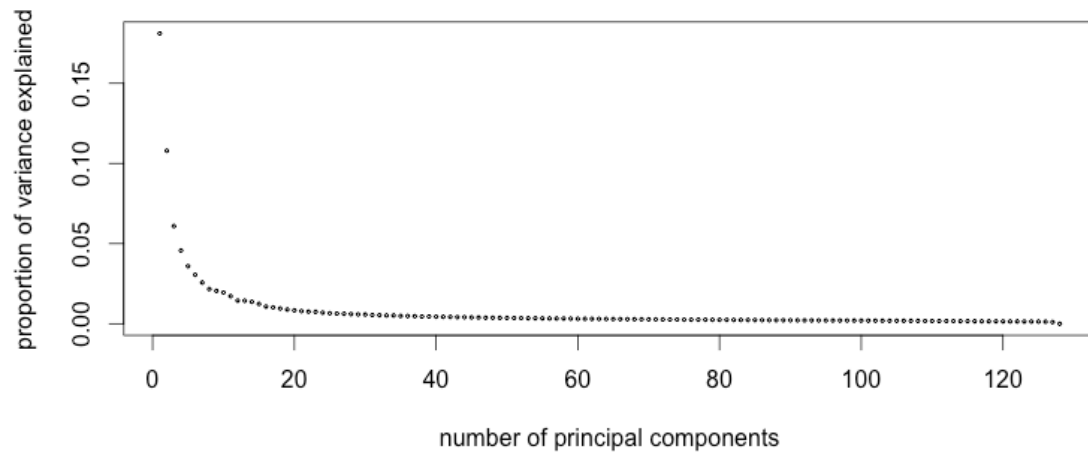
PC110 PC111 PC112 PC113 PC114 PC115 PC116 PC117
Standard deviation 4.68160 4.64703 4.61168 4.59981 4.56873 4.54519 4.45476
4.42230
Proportion of Variance 0.00174 0.00171 0.00168 0.00168 0.00165 0.00164 0.00157
0.00155

Cumulative Proportion 0.97468 0.97639 0.97808 0.97975 0.98141 0.98304
0.98462 0.98616

PC118 PC119 PC120 PC121 PC122 PC123 PC124 PC125
Standard deviation 4.38206 4.36821 4.30457 4.27980 4.25439 4.18120 4.16837
4.14619
Proportion of Variance 0.00152 0.00151 0.00147 0.00145 0.00143 0.00138 0.00138
0.00136

Cumulative Proportion 0.98769 0.98920 0.99066 0.99212 0.99355 0.99493
0.99631 0.99767

PC126 PC127 PC128
Standard deviation 4.00554 3.65401 1.028e-13
Proportion of Variance 0.00127 0.00106 0.000e+00
Cumulative Proportion 0.99894 1.00000 1.000e+00



I think 1 to 11 PCs should be used because there is a rapid drop of proportion of variance till PC11.

H)

Rscript:

```
install.packages('e1071');
library(e1071)
data.pca <- pca.all$x[,1:5]
svmest <- svm(IsB~data.pca,type="C-classification",kernel="linear")
svmpred <- predict(svmest,data.pca)
table(svmpred,IsB)
tpr.svm <- mean(svmpred==IsB &
IsB==TRUE)/(mean(svmpred==IsB&IsB==TRUE)+mean(svmpred!=IsB&IsB==TRUE))
tpr.svm
```

Answer:

```
IsB
svmpred FALSE TRUE
FALSE 30 1
TRUE 3 94
```

Sensitivity = 0.9894737

I)

Rscript:

```
n <- length(IsB)
mcr.cv.raw <- rep (NA,n)
```

```

for (i in 1:n) {
  svmest <- svm(data.pca[-i,],IsB[-i],type="C-classification",kernel="linear")
  svmpred <- predict(svmest,t(data.pca[i,]))
  mcr.cv.raw[i] <- mean(svmpred!=IsB[i])
}
mcr.cv <- mean(mcr.cv.raw)
mcr.cv

```

Answer:

estimate misclassification rate = $0.0390625 = 3.9\%$

J)

Answer:

MCR in E part is $11/128 = .0859 = 8.6\%$

MCR in H part is $4/128 = 0.0312 = 3.12\%$

As MCR for SVM method is less than logistic regression so I will choose SVM method
i.e. h part.

2. (30 points) Choosing Classifiers and Number of Principal Components for PCA reduced iris data set.

In the last example of this module, we compared three classifiers on the iris data by working on the first three principal components. We choose the best classifiers based on cross-validated misclassification rate. We can also choose the number of principal components to use by cross-validation, instead of fixing it at $K=3$.

Use the leave-one-out cross-validation to choose the number of principal components together with the classifier. Please report the empirical misclassification rates (on whole data set) and the leave-one-out cross-validation misclassification rates for each value of $K=1, 2, 3, 4$ principal components and for each of the three classifiers: logistic regression, support vector machine and classification tree. Based on those rates, what is your choice?

Note: when you fit the logistic regression with $K=1$ principal component, then the PC1 becomes a vector instead of a matrix. You will need to modify the code for logistic regression for $K=1$ differently from the other values of $K=2, 3, 4$.

2)

Rscript:

```
# Answer 2
```

```
install.packages("VGAM")
```

```
library("VGAM")
```

```
pca.iris <- prcomp(iris[,1:4], scale=TRUE)
```

```
Species <- iris$Species
```

```
data.pca <- pca.iris$x[,1]
```

```
n <- length(Species)
```

```
iris2 <- data.frame(Species, data.pca)
```

```
iris2.lgr <- vglm(Species~., family=multinomial, data=iris2)
```

```
pred.prob <- predict(iris2.lgr, iris2[,2,drop=F], type="response")
```

```
pred.lgr <- apply(pred.prob, 1, which.max)
```

```
pred.lgr <- factor(pred.lgr, levels=c("1","2","3"), labels=levels(iris2$Species))
```

```
mcr.lgr <- mean(pred.lgr!=iris2$Species)
```

```
### leave-one-out cross validation
```

```
mcr.cv.raw<-rep(NA, n)
```

```
for (i in 1:n) {
```

```
  fit.lgr <- vglm(Species~., family=multinomial, data=iris2[-i,])
```

```
  pred.prob <- predict(fit.lgr, iris2[i,-1, drop=F], type="response")
```

```
  pred <- apply(pred.prob, 1, which.max)
```

```
  pred <- factor(pred, levels=c("1","2","3"), labels=levels(iris2$Species))
```

```

    mcr.cv.raw[i] <- mean(pred!=Species[i])
  }
  mcr.cv <- mean(mcr.cv.raw)
  c(mcr.lgr, mcr.cv)
  # SVM mcr
  svmest <- svm(data.pca, Species, type = "C-classification", kernel = "linear") #train
  SVM
  svmpred <- predict(svmest, data.pca)
  mcr.svm<- mean(svmpred!=Species)
  mcr.svm
  mat<-data.frame(data.pca)
  ### leave-one-out cross validation
  mcr.cv.raw<-rep(NA, n)
  for (i in 1:n) {
    svmest <- svm(mat[-i,], Species[-i], type = "C-classification", kernel ="linear")
    svmpred <- predict(svmest, t(mat[i,]))
    mcr.cv.raw[i]<- mean(svmpred!=Species[i])
  }
  mcr.cv<-mean(mcr.cv.raw)
  c(mcr.svm, mcr.cv)

#Classification tree
fit <- rpart(Species ~ ., data = iris2, method = "class")
pred.tr<-predict(fit, iris2, type = "class")
mcr.tr <- mean(pred.tr!=Species)
### leave-one-out cross validation
mcr.cv.raw <- rep(NA, n) #A vector to save mcr validation
for (i in 1:n) {
  fit.tr <- rpart(Species ~ ., data = iris2[-i,], method = "class") #train the tree without
  i-th observation
  pred <- predict(fit.tr, iris2[i,], type = "class")#use tree to predict i-th observation
  class
  mcr.cv.raw[i] <- mean(pred!=Species[i]) #check misclassification
}
mcr.cv<-mean(mcr.cv.raw) #average the mcr over all n rounds.
c(mcr.tr, mcr.cv)

get_mcr_values <- function(k){
  data.pca <- pca.iris$x[,1:k]
  n <- length(Species)
  iris2 <- data.frame(Species, data.pca)
  iris2.lgr <- vglm(Species~., family=multinomial, data=iris2)
  pred.prob <- predict(iris2.lgr, iris2[,-1,drop=F], type="response")
  pred.lgr <- apply(pred.prob, 1, which.max)
  pred.lgr <- factor(pred.lgr, levels=c("1","2","3"), labels=levels(iris2$Species))

```

```

mcr.lgr <- mean(pred.lgr!=iris2$Species)
### leave-one-out cross validation
mcr.cv.raw<-rep(NA, n)
for (i in 1:n) {
  fit.lgr <- vglm(Species~., family=multinomial, data=iris2[-i,])
  pred.prob <- predict(fit.lgr, iris2[i,-1, drop=F], type="response")
  pred <- apply(pred.prob, 1, which.max)
  pred <- factor(pred, levels=c("1","2","3"), labels=levels(iris2$Species))
  mcr.cv.raw[i] <- mean(pred!=Species[i])
}
mcr.cv <- mean(mcr.cv.raw)
print(c("Logistic regression",mcr.lgr, mcr.cv))

# SVM mcr
svmest <- svm(data.pca, Species, type = "C-classification", kernel = "linear") #train
SVM
svmpred <- predict(svmest , data.pca)
mcr.svm<- mean(svmpred!=Species)
### leave-one-out cross validation
mcr.cv.raw<-rep(NA, n)
for (i in 1:n) {
  svmest <- svm(data.pca[-i,], Species[-i], type = "C-classification", kernel ="linear")
  svmpred <- predict(svmest, t(data.pca[i,]))
  mcr.cv.raw[i]<- mean(svmpred!=Species[i])
}
mcr.cv<-mean(mcr.cv.raw)
print(c("SVM",mcr.svm, mcr.cv))

#Classification tree
fit <- rpart(Species ~ ., data = iris2, method = "class")
pred.tr<-predict(fit, iris2, type = "class")
mcr.tr <- mean(pred.tr!=Species)
### leave-one-out cross validation
mcr.cv.raw <- rep(NA, n) #A vector to save mcr validation
for (i in 1:n) {
  fit.tr <- rpart(Species ~ ., data = iris2[-i,], method = "class") #train the tree without
i-th observation
  pred <- predict(fit.tr, iris2[i,], type = "class")#use tree to predict i-th observation
class
  mcr.cv.raw[i] <- mean(pred!=Species[i]) #check misclassification
}
mcr.cv<-mean(mcr.cv.raw) #average the mcr over all n rounds.
print(c("Classification tree",mcr.tr, mcr.cv))

}
get_mcr_values(2)

```

```
get_mcr_values(3)
get_mcr_values(4)
```

Answer:

	Empirical	Leave one out
K=1		
Logistic Regression =	0.07333333	0.07333333
SVM =	0.07333333	0.08000000
Classification tree =	0.06666667	0.10666667
K=2		
Logistic Regression =	0.08	0.08
SVM =	0.08667	0.08667
Classification tree =	0.0667	0.10667
K=3		
Logistic Regression =	0.0133	0.02667
SVM =	0.02667	0.04667
Classification tree =	0.0667	0.14
K=4		
Logistic Regression =	0.0133	0.02
SVM =	0.02	0.02667
Classification tree =	0.0667	0.14

Based on the above rates I would choose K=4 as the values for empirical and leave one out is really close for K=4.

I would also consider SVM as the best classifier method as both the values are really close