**Midterm exam DATA 590**

**Fall 2017**

**Exam Instruction**

*Answer the following questions in organized answers. Work on your own. Do NOT do web searches to find information or answers directly related to the questions. Include R code used and all relevant output including plots in your Word document.*

*Be sure to read each question carefully and write at least one complete sentence in answering each question.*

*If you do not understand a question or think there's no way to answer it, take a deep breath or two and relax. I probably did not explain well enough. Email me for clarification and work around the question. I will check my email regularly.*

**Part 1 - Theory**

**A. We started the semester with LDFA, with its explicit assumptions and useful visualization methods. What are they? By the time we got to neural Networks, we talked about a "black box" or boîte noire method. Briefly summarize each method we have learned, including assumptions (requirements), any necessary data pre-processing, and advantages and disadvantages of each method. The table on the last page should help. For Naïve Bayes you can restrict it to nominal data.**

From LDFA to Neural Network we have discussed 9 major machine learning concepts. We discuss below the Assumption, useful data visualization technique if applicable, Data processing required, advantages and disadvantage for each method.

**Linear Discriminant Function Analysis(LDFA)**

Assumption

* The Data must be normal
* The data should have equal variance co variance matrix

Useful Visualization Method

* Plot

Data Pre-processing

* It uses input which are continuous
* The output is nominal

Advantages

* It’s good at classification
* Plots groups which are greater than two

Disadvantages

* Disadvantage is overfitting

**Quadratic discriminant analysis(QDA)**

Assumption

* The Data must be normal

Useful Visualization Method

* Plot

Data Pre-processing

* It uses input which are continuous
* The output is nominal

Advantages

* It run on data with unequal variance co variance matrix

Disadvantages

* Disadvantage is overfitting

**Regularized Discriminant Analysis(RDA)**

Assumption

* The Data must be normal

Useful Visualization Method

* Plot

Data Pre-processing

* It uses input which are continuous
* The output is nominal

Advantages

* It is a compromise between LDFA and QDA

Disadvantages

* Disadvantage is overfitting

**Logistic regression**

Assumption

* Logs odds linear in predictors
* Assumes a linear relationship between continuous predictors and logit
* Need larger samples than DFA for best estimates
* Model fit must be evaluated without pure R2
* Multicollinearity can cause problems
* Outliers cause problems
* MLE won't work when groups completely separated
* Compares likelihood ratios of different models

Useful Visualization Method

* Sigmoid curve

Data Pre-processing

Advantages

* Predictors can be continuous, binary, nominal ordinal, or a mix
* no normality requirement - can use OR mix continuous, discrete, ordinal
* no VCVM homogeneity requirement
* Most useful when assessing predictors in a binary outcome
* but can predict more categories

Disadvantages

* larger sample sizes recommended
* most research is in goodness of fit, not classification performance
* larger sample sizes recommended - most research is in goodness of fit, not classification performance
* Accuracy falls with 2+ groups

**k-Nearest Neighbour (done)**

Assumption

* Need to normalize all variables

Data Pre-processing

* The default in R is to use the Euclidean distance. So, the variance in each variable is important.
* Scaling or Transformation is often needed

Advantages

* Group distance unimportant
* Can use mixed data
* less overfitting. NL bounds

Disadvantages

* MUST transform
* Need to choose number of neighbours
* There is no model (relationship of inputs to output)

**Kernel Probability Density**

Assumption

* Need to normalize all variables

Useful Visualization Method

Data Pre-processing

* type of kernel
* Bandwidth/radius

Advantages

* Group distance unimportant
* Can use mixed data
* less overfitting. NL bounds

Disadvantages

* MUST transform
* Choose distance
* Its kernel types
* Uses radius for classification
* There is no model (relationship of inputs to output)

**SVM**

Assumption

* There is no major assumption

Data Pre-processing

Advantages

* Non- Linear boundaries
* Regularizes/transforms data
* less overfit
* Uses kernels;

Disadvantages

* Choose kernel, gamma, cost, etc.
* There is no model (relationship of inputs to output)

**Neural Network**

Assumption

* Normalize all vars

Useful Visualization Method

Advantages

* Model fitting very sophisticated

Disadvantages

* Very complex models
* Make some choices;

**Naïve Bayes**

Assumption

Useful Visualization Method

Data Pre-processing

Advantages

Disadvantages

**B. I showed you the quote attributed to the great American philosopher and baseball player Yogi Berra: "It’s tough to make predictions, especially about the future." and explained its relevance for Machine Learning. From the following list of other Berra quotations corresponding to your name, interpret and explain how the quote could be applied to Machine Learning and/or statistical methods. Use a MAXIMUM of 8 sentences. You only have to write about ONE quote.**

**“**Little things are big**”**

Mzxn vmnzx vmnzx vmn zxvmn zmnc

**Part 2 – DATA**

**Include all of your relevant coding and output.**

**C. Work with the data in file: http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv. Make sure you are using at least 10 predictors (inputs, from i1 to i59) you can that are present in at least 50 individuals from each group (Grp). What classification methods are theoretically most appropriate, and why?**

**Run analyses, making sure to check assumptions and perform any pre-processing necessary, to find the method that produces the greatest median accuracy in repeated holdout samples.**

**Use methods: LDFA, FDA, multinom(nnet), SVM Run at least 1,000 times for fast methods and 100 for slower methods.**

**Using caret will help with some of the methods. Present results in a table. Which method is best?**

|  |  |  |  |
| --- | --- | --- | --- |
| Sl No | Method | Accuracy | No of Runs |
| 1 | LDA | 0.7 | 1000 |
| 2 | FDA | 0.8606 | 1000 |
| 3 | Multinomial | 0.7987 | 1000 |
| 4 | SVM | 0.7985 | 1000 |
| 5 | Decision Tree | 0.77193 | 1000 |
| 6 | Naive Bayes | 0.6045 | 1000 |
| 7 | KNN | 0.7463 | 1000 |

The 10 predictors That I have used for analysis are i2+i3+i6+i10+i22+i36+i37+i9+i24+i29 as they classify group with group count as below.

1 2 3

79 272 101

The classification which work for this data set are LDA, FDA, Multinomial using neural net, SVM, Decision tree, Naïve Bayes and KNN.

The method that best predict the accuracy is Multinomial using neural network with accuracy of 0.8606 when we ran the code for 1000 times.

**Decision Tree**

> library(caret)

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

>

> attach(eddat);

>

> sapply(eddat, function(x) sum(is.na(x))/length(x))

>

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

Rec RecT Grp i1 i2 i3

>

> colSums(is.na(eddat))

>

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

>

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

>

> for (i in seq(1000))

+ {

+ inTest<-createDataPartition(eddat$Grp, p = .25, list = FALSE)

+ require(rpart)

+ edata1<-rpart(Grp~ ., data=eddat,method = "class",subset = inTest,

+ parms= list(split = "gini",prior = c(1/3,1/3,1/3)),

+ control = rpart.control(usesurrogate= 0, maxsurrogate= 0))

+

+ Accuracies[i] <- confusionMatrix(eddat[inTest,"Grp"],predict(edata1,newdata= eddat[inTest,],type = "class"))$overall["Accuracy"]

+

+ }

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.6315789 0.7456140 0.7719298 0.7680965 0.7894737 0.8771930

**Naive Bayes**

> library(caret)

>

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

>

> attach(eddat);

>

> sapply(eddat, function(x) sum(is.na(x))/length(x))

>

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

>

> colSums(is.na(eddat))

>

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

>

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

> for (i in seq(1000))

+ {

+ inTrain <- createDataPartition(y = as.factor(eddat$Grp), p = .70, list = FALSE)

+ train <- eddat[inTrain,]

+ test <- eddat[-inTrain,]

+ nb1 <- train(as.factor(Grp) ~ i2+i3+i6+i10+i22+i36+i37+i9+i24+i29, data = train, method = "nb",

+ trControl = trainControl(method = "cv"),

+ tuneGrid = data.frame(usekernel = TRUE, fL = 0.5, adjust = 5))

+ bps <- predict(nb1, newdata = test)

+ Accuracies[i] <- confusionMatrix(test$Grp,bps)$overall["Accuracy"]

+ }

There were 50 or more warnings (use warnings() to see the first 50)

>

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.5820896 0.5970149 0.6044776 0.6054627 0.6044776 0.7537313

**LDA**

> library(caret)

>

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

>

> attach(eddat);

>

>

> sapply(eddat, function(x) sum(is.na(x))/length(x))

>

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

>

> colSums(is.na(eddat))

>

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

>

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

>

> require(asbio)

> Accuracies <- c(0.00)

> for (i in seq(1000))

+ {

+ inTrain <- createDataPartition(y = eddat$Grp, p = .70, list = FALSE)

+ training <- eddat[inTrain,]

+ edata2 <- lda(as.matrix(eddat[,c(2:11)]), eddat[,"Grp"], data = eddat, prior = c(1/3,1/3,1/3),

+ subset = -inTrain, CV = T)

+ Accuracies[i] <- confusionMatrix(eddat[-inTrain,"Grp"], edata2$class)$overall["Accuracy"]

+ }

Error in lda.default(x, grouping, ...) :

variable 4 appears to be constant within groups

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.6888889 0.6944444 0.7000000 0.7000000 0.7055556 0.7111111

**KNN**

> library(caret)

>

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

>

> attach(eddat);

>

>

> sapply(eddat, function(x) sum(is.na(x))/length(x))

>

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

>

> colSums(is.na(eddat))

>

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

>

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

>

> for (i in seq(1000))

+ {

+ inTrain <- createDataPartition(y = as.factor(eddat$Grp), p = .70, list = FALSE)

+ training <- eddat[inTrain,]

+ testing <- eddat[-inTrain,]

+ knn4 <- train(as.factor(Grp) ~ i2+i3+i6+i10+i22+i36+i37+i9+i24+i29, data = training, method = "knn",

+ preProcess = c("center", "scale"), tuneLength = 10,

+ trControl = trainControl(method = "cv"))

+ update(knn4, list(.k = 3))

+ knn4\_pred <- predict(knn4,newdata = testing)

+ Accuracies[i] <- confusionMatrix(knn4\_pred,as.factor(testing$Grp))$overall["Accuracy"]

+ }

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.6492537 0.7238806 0.7462687 0.7452463 0.7611940 0.8283582

**FDA**

> library(mda)

> library(caret)

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

> attach(eddat);

> sapply(eddat, function(x) sum(is.na(x))/length(x))

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

> colSums(is.na(eddat))

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

> for (i in seq(1000))

+ {

+

+ edat1 <- fda(Grp ~ i2+i3+i6+i10+i22+i36+i37+i9+i24+i29, data = eddat, prior = c(1/3,1/3,1/3), CV = TRUE, method =

+ polyreg, degree = 3)

+

+ Accuracies[i] <- confusionMatrix(eddat$Grp, predict(edat1))$overall["Accuracy"]

+ }

There were 50 or more warnings (use warnings() to see the first 50)

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.8606195 0.8606195 0.8606195 0.8606195 0.8606195 0.8606195

**Multinomial**

> library(nnet)

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

> attach(eddat);

> sapply(eddat, function(x) sum(is.na(x))/length(x))

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

> colSums(is.na(eddat))

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

> for (i in seq(1000))

+ {

+

+ mnmfit <- multinom(Grp ~ i2+i3+i6+i10+i22+i36+i37+i9+i24+i29,data=eddat, trace=FALSE)

+ #summary(mnmfit)

+

+ # get predictions on a new data set

+ round(predict(mnmfit, newdata=eddat, "probs"),4)

+

+ predict(mnmfit, newdata=eddat, "class")

+

+ #Multinominal

+

+

+ Accuracies[i] <- confusionMatrix(predict(mnmfit, newdata=eddat, "class"), eddat$Grp)$overall["Accuracy"]

+ }

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.7986726 0.7986726 0.7986726 0.7986726 0.7986726 0.7986726

**SVM**

> library(nnet)

>

> eddat <- read.csv('http://math.mercyhurst.edu/~sousley/STAT\_139/data/EDDat3.csv', as.is = T);

> attach(eddat);

> sapply(eddat, function(x) sum(is.na(x))/length(x))

> sapply(eddat, function(x) sum(!is.na(x))/length(x))

> colSums(is.na(eddat))

> table(na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])$Grp)

1 2 3

79 272 101

> eddat <- na.omit(eddat[,c(3,5,6,9,13,25,39,40,12,27,32)])

>

> Accuracies <- c(0.00)

> for (i in seq(1000))

+ {

+

+ inTrain <- createDataPartition(y = as.factor(eddat$Grp), p = .70, list = FALSE)

+ train <- eddat[inTrain,]

+ test <- eddat[-inTrain,]

+ # book code, demo code from e1071

+ linear.tune = tune.svm(as.factor(Grp) ~ i2+i3+i6+i10+i22+i36+i37+i9+i24+i29, data=eddat, kernel="linear",

+ cost=c(0.001, 0.01, 0.1, 1,5,10))

+ poly.tune = tune.svm(as.factor(Grp)~i2+i3+i6+i10+i22+i36+i37+i9+i24+i29, data=eddat, kernel="polynomial",

+ degree=c(3,4,5), coef0=c(0.1,0.5,1,2,3,4))

+

+ summary(linear.tune)

+

+ summary(poly.tune)

+

+ best.linear = linear.tune$best.model

+ tune.test = predict(best.linear, newdata=test)

+

+ Accuracies[i] <- confusionMatrix(test$Grp, tune.test)$overall["Accuracy"]

+ }

> summary(Accuracies)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.7238806 0.7835821 0.7985075 0.8024179 0.8208955 0.8880597

**D. This will be a chance for some of you to make up for low scores on homework 1. Work with the Asian fusion data again, which are skull measurements.**

**Extract the first 48 columns. Then use na.omit() and you should be left with 526 complete individuals and 45 predictors. The first 3 are record, catkey, and sex.**

**Using your new or corrected or improved or working outlier function code from before, generate a master list of all univariate outliers for each predictor using car's Boxplot function, and be sure to display by CATKEY and predictor label. Some individuals will be outliers in several predictors. These are measurements; interpret any patterns in the repeated outliers you see. In other words, how many kinds of repeated outliers do you see?**

**Compare your univariate list to a list generated using the Penny method on the original data using all available measurements and display those with multivariate p values < 0.01. Be sure to display CATKEY again. How many individuals were common to each list? How can you interpret any great differences in outliers found by each method?**

ljnavnjnv

|  |  |  |
| --- | --- | --- |
| Univariate > 2 | Multivariate | Univariate |
| 024-CRA | 019-CRA | 024-CRA |
| 029-CRA | 035-CRA | 029-CRA |
| 031-CRA | 052-CRA | 031-CRA |
| 049-CRA | 052-CRA | 034-CRA |
| 083-CRA | 081-CRA | 037-CRA |
| 11-058 | 087-CRA | 049-CRA |
| A2SC031 | 089-CRA | 051-CRA |
| A2SC043 | 089-CRA | 083-CRA |
| A3SC78 | 090-CRA | 09-079 |
| B1SC153 | 11-058 | 11-058 |
| B2SC184 | 76-720 | A1SC025 |
| B2SC189 | A2SC031 | A2SC031 |
| B3SC235 | A2SC056 | A2SC043 |
| B5SC297 | A3SC72 | A2SC59 |
| C3SC367 | A4SC091 | A3SC066 |
| C3SC390 | A4SC102 | A3SC089 |
| C5SC423 | A4SC118 | A3SC72 |
| C5SC438 | B1SC153 | A3SC78 |
| D4SC551 | B1SC153 | A4SC118 |
| E2SC634 | B2SC181 | A5SC149 |
| JI-178 | B2SC184 | B1SC153 |
| JI-4883 | B2SC188 | B2SC183 |
| JI-4985 |  | B2SC184 |
| JI-5022 | B3SC231 | B2SC186 |
| JIKEI-47F | B4SC243 | B2SC189 |
| KU-1116 | B4SC253 | B3SC216 |
| KU-1388 | B5SC281 | B3SC221 |
| KU-2181 | B5SC281 | B3SC235 |
| KU-2958 | C1SC0318 | B4SC520 |
| T75-353 | C1SC0318 | B5SC297 |
| T75-387 | C1SC305 | C1SC0324 |
| T79-288 | C1SC305 | C2SC338 |
| T79-92 | C3SC367 | C3SC367 |
| Tohoku-1063 |  | C3SC390 |
| Tohoku-1199 | C5SC422 | C5SC423 |
| Tohoku-1203 | C5SC427 | C5SC434 |
| Tohoku-1299 | C5SC434 | C5SC435 |
| Tohoku-2480 | C5SC435 | C5SC438 |
| Tohoku-2544 | D3SC529 | D1SC459 |
| Tohoku-2748 | D5SC580 | D1SC461 |
| Tohoku-3345 | JI-13 | D3SC525 |
| Tohoku-395 | JI-178 | D3SC532 |
| Tohoku-478 | JI-4923 | D4SC551 |
| Tohoku-508 | JI-4923 | E2SC634 |
| Tohoku-553 | JI-4946 | JI-133 |
| TU-1367 | JI-4947 | JI-178 |
| TU-1369 | JI-5211 | JI-21 |
| TU-1443 | JI-5232 | JI-3562 |
| TU-1457 | JI-5737 | JI-4883 |
| TU-1471 | KU-1562 | JI-4892 |
| TU-1849 | KU-2405 | JI-4946 |
| TU-1868 | KU-3139 | JI-4985 |
| TU-1871 | T75-351 | JI-5022 |
| TU-220 | T75-353 | JI-6476 |
| TU-220 | T75-365 | JIKEI-47F |
| TU-2218 | T76-615 | KU-1116 |
| TU-2440 | T76-671 | KU-1118 |
| TU-49 | T76-677 | KU-1120 |
|  | T77-198 | KU-1265 |
|  | T77-400 | KU-1301 |
|  | T77-450 | KU-1388 |
|  | T77-94 | KU-1415 |
|  | T78-205 | KU-1421 |
|  | T78-246 | KU-1446 |
|  | T78-251 | KU-1562 |
|  | T78-258 | KU-2133 |
|  | T78-337 | KU-2181 |
|  | T79-107 | KU-2594 |
|  | T79-144 | KU-2627 |
|  | T79-198 | KU-2636 |
|  | T79-228 | KU-2941 |
|  | T79-28 | KU-2945 |
|  | T79-92 | KU-2958 |
|  | T80-061 | KU-907 |
|  | Tohoku-1137 | T74-221 |
|  | Tohoku-1139 | T75-353 |
|  | TU-1334 | T75-362 |
|  | TU-1342 | T75-387 |
|  | TU-1343 | T76-677 |
|  | TU-1471 | T76-714 |
|  | TU-1479 | T77-438 |
|  | TU-2099 | T78-220 |
|  | TU-2137 | T78-264 |
|  | TU-2389 | T78-280 |
|  | TU-2539 | T78-48 |
|  |  | T79-112 |
|  |  | T79-164 |
|  |  | T79-27 |
|  |  | T79-28 |
|  |  | T79-288 |
|  |  | T79-311 |
|  |  | T79-92 |
|  |  | T81-293 |
|  |  | Tohoku-1063 |
|  |  | Tohoku-1082 |
|  |  | Tohoku-1199 |
|  |  | Tohoku-1203 |
|  |  | Tohoku-1299 |
|  |  | Tohoku-1475 |
|  |  | Tohoku-2088 |
|  |  | Tohoku-235 |
|  |  | Tohoku-2480 |
|  |  | Tohoku-2544 |
|  |  | Tohoku-2748 |
|  |  | Tohoku-302 |
|  |  | Tohoku-3264 |
|  |  | Tohoku-3278 |
|  |  | Tohoku-3290 |
|  |  | Tohoku-3296 |
|  |  | Tohoku-3345 |
|  |  | Tohoku-3375 |
|  |  | Tohoku-3466 |
|  |  | Tohoku-395 |
|  |  | Tohoku-453 |
|  |  | Tohoku-478 |
|  |  | Tohoku-481 |
|  |  | Tohoku-484 |
|  |  | Tohoku-508 |
|  |  | Tohoku-522 |
|  |  | Tohoku-553 |
|  |  | Tohoku-569 |
|  |  | Tohoku-965 |
|  |  | TU-1334 |
|  |  | TU-1342 |
|  |  | TU-1360 |
|  |  | TU-1367 |
|  |  | TU-1369 |
|  |  | TU-1432 |
|  |  | TU-1443 |
|  |  | TU-1457 |
|  |  | TU-1471 |
|  |  | TU-1831 |
|  |  | TU-1849 |
|  |  | TU-1868 |
|  |  | TU-1871 |
|  |  | TU-2063 |
|  |  | TU-207 |
|  |  | TU-2081 |
|  |  | TU-2084 |
|  |  | TU-2101 |
|  |  | TU-2139 |
|  |  | TU-2188 |
|  |  | TU-220 |
|  |  | TU-2218 |
|  |  | TU-227 |
|  |  | TU-2409 |
|  |  | TU-2440 |
|  |  | TU-2544 |
|  |  | TU-2551 |
|  |  | TU-2556 |
|  |  | TU-2557 |
|  |  | TU-271 |
|  |  | TU-37 |
|  |  | TU-39 |
|  |  | TU-49 |

Codes to get univariate outliers

> af <- read.csv("Asian\_Fusion\_Data.csv", as.is = TRUE);

>

> af <- af[,c(1:48)]

>

> mydata <- na.omit(af)

>

> flist <- c("a", "c")

> df <- data.frame()

>

> for (i in 4:48)

+ {

+ #flist <- c("","")

+ x <- mydata[,i]

+ y <- colnames(mydata[i])

+ OUT <- Boxplot(x)

+ list1 <- cbind(mydata[OUT,c("Catkey",y)])

+ a <- print(list1)

+ a

+ #flist <- rbind(flist, a)

+ #print(list(y))

+ #df <-

+

+ mydata[Boxplot(x),y] <- NA

+ #mydata <- na.omit(mydata)

+

+ #names(list1)[1] <- "Var"

+

+ #names(list1)[3] <- "Val"

+

+ OUT <- Boxplot(x)

+

+ if (is.null(OUT))

+ {

+ #Do Nothin

+ }

+

+ else

+ {

+ x <- mydata[,i]

+ y <- colnames(mydata[i])

+ OUT <- Boxplot(x)

+ list1 <- cbind(mydata[OUT,c("Catkey",y)])

+ a <- print(list1)

+ a

+ #flist <- rbind(flist, a)

+ #print(list(y))

+

+ mydata[Boxplot(x),y] <- NA

+

+ }

+ }

Catkey GOL

58 A2SC043 152

216 KU-1265 153

510 TU-49 199

[1] Catkey GOL

<0 rows> (or 0-length row.names)

Catkey NOL

58 A2SC043 148

510 TU-49 197

[1] Catkey NOL

<0 rows> (or 0-length row.names)

Catkey BNL

9 029-CRA 85

190 JI-5022 85

211 KU-1116 82

582 Tohoku-553 83

417 TU-1457 115

423 TU-1471 115

[1] Catkey BNL

<0 rows> (or 0-length row.names)

Catkey BBH

169 JI-133 113

211 KU-1116 117

429 TU-1831 119

[1] Catkey BBH

<0 rows> (or 0-length row.names)

Catkey XCB

139 C5SC438 156

375 T79-288 156

Catkey XCB

174 JI-4883 123

546 Tohoku-302 124

103 B3SC235 153

111 B4SC520 153

137 C5SC434 154

Catkey XFB

5 024-CRA 100

237 KU-2181 98

396 TU-1342 100

524 Tohoku-1299 100

537 Tohoku-2480 80

45 11-058 132

139 C5SC438 129

160 E2SC634 129

[1] Catkey XFB

<0 rows> (or 0-length row.names)

Catkey WFB

82 B1SC153 106

205 JIKEI-47F 108

[1] Catkey WFB

<0 rows> (or 0-length row.names)

Catkey ZYB

9 029-CRA 104

68 A3SC78 107

515 Tohoku-1063 111

570 Tohoku-395 111

580 Tohoku-508 106

582 Tohoku-553 110

[1] Catkey ZYB

<0 rows> (or 0-length row.names)

Catkey AUB

9 029-CRA 103

68 A3SC78 105

452 TU-2101 103

580 Tohoku-508 104

160 E2SC634 139

300 T75-387 140

[1] Catkey AUB

<0 rows> (or 0-length row.names)

Catkey ASB

237 KU-2181 92

372 T79-27 90

452 TU-2101 89

56 A2SC031 122

89 B2SC183 128

205 JIKEI-47F 125

378 T79-311 124

417 TU-1457 121

492 TU-2556 121

[1] Catkey ASB

<0 rows> (or 0-length row.names)

Catkey BPL

9 029-CRA 75

68 A3SC78 79

262 KU-2958 81

517 Tohoku-1082 80

536 Tohoku-235 80

580 Tohoku-508 80

582 Tohoku-553 78

90 B2SC184 111

151 D4SC551 110

402 TU-1360 113

404 TU-1367 111

410 TU-1432 110

417 TU-1457 116

423 TU-1471 113

482 TU-2409 110

Catkey BPL

121 C2SC338 83

211 KU-1116 83

576 Tohoku-478 83

314 T76-714 109

362 T79-112 109

391 TU-1334 109

405 TU-1369 109

Catkey NPH

9 029-CRA 54

68 A3SC78 51

536 Tohoku-235 53

580 Tohoku-508 50

582 Tohoku-553 55

41 09-079 81

[1] Catkey NPH

<0 rows> (or 0-length row.names)

Catkey NLH

68 A3SC78 38

211 KU-1116 40

536 Tohoku-235 40

580 Tohoku-508 39

[1] Catkey NLH

<0 rows> (or 0-length row.names)

Catkey JUB

9 029-CRA 91

515 Tohoku-1063 93

536 Tohoku-235 96

570 Tohoku-395 95

580 Tohoku-508 90

[1] Catkey JUB

<0 rows> (or 0-length row.names)

Catkey NLB

58 A2SC043 15

580 Tohoku-508 19

77 A4SC118 32

[1] Catkey NLB

<0 rows> (or 0-length row.names)

Catkey MAB

9 029-CRA 48

576 Tohoku-478 50

359 T78-48 76

[1] Catkey MAB

<0 rows> (or 0-length row.names)

Catkey MAL

9 029-CRA 37

25 049-CRA 42

68 A3SC78 39

412 TU-1443 36

449 TU-2084 38

468 TU-2139 36

536 Tohoku-235 40

580 Tohoku-508 40

582 Tohoku-553 41

[1] Catkey MAL

<0 rows> (or 0-length row.names)

Catkey MDH

9 029-CRA 15

68 A3SC78 16

580 Tohoku-508 17

582 Tohoku-553 17

[1] Catkey MDH

<0 rows> (or 0-length row.names)

Catkey OBH

583 Tohoku-569 25

281 KU-907 40

[1] Catkey OBH

<0 rows> (or 0-length row.names)

Catkey OBB

100 B3SC221 32

190 JI-5022 32

404 TU-1367 45

478 TU-227 45

[1] Catkey OBB

<0 rows> (or 0-length row.names)

Catkey DKB

9 029-CRA 15

25 049-CRA 15

432 TU-1849 15

537 Tohoku-2480 15

580 Tohoku-508 14

81 A5SC149 30

138 C5SC435 28

139 C5SC438 29

188 JI-4985 28

286 T74-221 28

325 T77-438 28

475 TU-220 28

493 TU-2557 28

[1] Catkey DKB

<0 rows> (or 0-length row.names)

Catkey NDS

541 Tohoku-2544 4

103 B3SC235 14

113 B5SC297 17

366 T79-164 14

476 TU-2218 16

[1] Catkey NDS

<0 rows> (or 0-length row.names)

Catkey WNB

173 JI-3562 2.4

432 TU-1849 1.7

521 Tohoku-1203 2.8

90 B2SC184 13.7

124 C3SC367 13.7

143 D1SC461 13.0

151 D4SC551 13.1

205 JIKEI-47F 14.3

475 TU-220 13.0

566 Tohoku-3375 13.7

Catkey WNB

212 KU-1118 3

541 Tohoku-2544 3

Catkey SIS

113 B5SC297 7.7

134 C5SC423 6.8

205 JIKEI-47F 6.3

476 TU-2218 6.1

487 TU-2544 6.1

505 TU-37 7.1

506 TU-39 6.5

Catkey SIS

91 B2SC186 5.9

381 T79-92 5.7

405 TU-1369 5.9

Catkey ZMB

9 029-CRA 71

68 A3SC78 79

262 KU-2958 79

580 Tohoku-508 74

417 TU-1457 118

474 TU-2188 114

[1] Catkey ZMB

<0 rows> (or 0-length row.names)

Catkey SSS

211 KU-1116 13

58 A2SC043 34

[1] Catkey SSS

<0 rows> (or 0-length row.names)

Catkey FMB

9 029-CRA 79

190 JI-5022 81

515 Tohoku-1063 81

536 Tohoku-235 80

580 Tohoku-508 78

103 B3SC235 112

Catkey FMB

68 A3SC78 83

452 TU-2101 83

476 TU-2218 109

Catkey NAS

188 JI-4985 7

476 TU-2218 26

103 B3SC235 24

82 B1SC153 22

412 TU-1443 22

90 B2SC184 21

113 B5SC297 21

313 T76-677 21

433 TU-1868 21

484 TU-2440 21

510 TU-49 21

Catkey NAS

567 Tohoku-3466 21

Catkey EKB

9 029-CRA 80

190 JI-5022 81

515 Tohoku-1063 82

580 Tohoku-508 79

[1] Catkey EKB

<0 rows> (or 0-length row.names)

Catkey DKS

188 JI-4985 5

205 JIKEI-47F 5

221 KU-1388 5

234 KU-2133 5

249 KU-2627 5

260 KU-2941 5

261 KU-2945 5

296 T75-362 5

387 T81-293 5

563 Tohoku-3345 5

182 JI-4946 16

213 KU-1120 16

228 KU-1562 15

250 KU-2636 15

375 T79-288 15

412 TU-1443 17

433 TU-1868 16

439 TU-2063 15

476 TU-2218 18

[1] Catkey DKS

<0 rows> (or 0-length row.names)

Catkey IML

134 C5SC423 18

176 JI-4892 22

223 KU-1415 18

432 TU-1849 19

434 TU-1871 21

550 Tohoku-3264 20

62 A2SC59 45

64 A3SC089 44

139 C5SC438 44

412 TU-1443 48

528 Tohoku-1475 45

[1] Catkey IML

<0 rows> (or 0-length row.names)

Catkey XML

134 C5SC423 40

434 TU-1871 39

139 C5SC438 73

412 TU-1443 66

[1] Catkey XML

<0 rows> (or 0-length row.names)

Catkey MLS

580 Tohoku-508 6

139 C5SC438 19

142 D1SC459 18

300 T75-387 17

520 Tohoku-1199 18

521 Tohoku-1203 17

556 Tohoku-3296 17

[1] Catkey MLS

<0 rows> (or 0-length row.names)

Catkey WMH

139 C5SC438 32

443 TU-207 34

[1] Catkey WMH

<0 rows> (or 0-length row.names)

Catkey GLS

118 C1SC0324 8

205 JIKEI-47F 8

[1] Catkey GLS

<0 rows> (or 0-length row.names)

Catkey STB

581 Tohoku-522 78

219 KU-1301 80

535 Tohoku-2088 81

553 Tohoku-3278 88

524 Tohoku-1299 90

589 Tohoku-965 90

226 KU-1446 92

543 Tohoku-2748 92

5 024-CRA 93

237 KU-2181 93

45 11-058 129

139 C5SC438 131

Catkey STB

563 Tohoku-3345 93

574 Tohoku-453 93

Catkey FRC

58 A2SC043 93

452 TU-2101 95

[1] Catkey FRC

<0 rows> (or 0-length row.names)

Catkey FRS

26 051-CRA 16

149 D3SC532 18

203 JI-6476 19

520 Tohoku-1199 19

36 083-CRA 83

56 A2SC031 40

58 A2SC043 38

63 A3SC066 35

139 C5SC438 32

171 JI-21 32

188 JI-4985 34

221 KU-1388 32

350 T78-264 32

448 TU-2081 39

[1] Catkey FRS

<0 rows> (or 0-length row.names)

Catkey FRF

58 A2SC043 13

68 A3SC78 37

237 KU-2181 34

373 T79-28 2

448 TU-2081 5

452 TU-2101 33

484 TU-2440 34

536 Tohoku-235 37

36 083-CRA 145

56 A2SC031 206

294 T75-353 63

[1] Catkey FRF

<0 rows> (or 0-length row.names)

Catkey PAC

174 JI-4883 92

237 KU-2181 92

[1] Catkey PAC

<0 rows> (or 0-length row.names)

Catkey PAS

170 JI-178 15

174 JI-4883 15

381 T79-92 15

499 TU-271 15

13 034-CRA 34

93 B2SC189 37

127 C3SC390 35

[1] Catkey PAS

<0 rows> (or 0-length row.names)

Catkey PAF

15 037-CRA 38

352 T78-280 74

[1] Catkey PAF

<0 rows> (or 0-length row.names)

Catkey OCC

93 B2SC189 80

127 C3SC390 79

11 031-CRA 112

99 B3SC216 113

170 JI-178 116

246 KU-2594 118

343 T78-220 116

491 TU-2551 113

577 Tohoku-481 113

[1] Catkey OCC

<0 rows> (or 0-length row.names)

Catkey OCS

93 B2SC189 16

11 031-CRA 40

555 Tohoku-3290 38

[1] Catkey OCS

<0 rows> (or 0-length row.names)

Catkey OCF

225 KU-1421 30

55 A1SC025 75

66 A3SC72 75

124 C3SC367 73

147 D3SC525 75

160 E2SC634 77

294 T75-353 74

578 Tohoku-484 73

[1] Catkey OCF

<0 rows> (or 0-length row.names)

**Cleaned and sorted list of all the univariate outlier**

|  |  |  |
| --- | --- | --- |
| Var | Catkey | Val |
| 5 | 024-CRA | 100 |
| 5 | 024-CRA | 93 |
| 9 | 029-CRA | 85 |
| 9 | 029-CRA | 104 |
| 9 | 029-CRA | 103 |
| 9 | 029-CRA | 75 |
| 9 | 029-CRA | 54 |
| 9 | 029-CRA | 91 |
| 9 | 029-CRA | 48 |
| 9 | 029-CRA | 37 |
| 9 | 029-CRA | 15 |
| 9 | 029-CRA | 15 |
| 9 | 029-CRA | 71 |
| 9 | 029-CRA | 79 |
| 9 | 029-CRA | 80 |
| 11 | 031-CRA | 112 |
| 11 | 031-CRA | 40 |
| 13 | 034-CRA | 34 |
| 15 | 037-CRA | 38 |
| 25 | 049-CRA | 42 |
| 25 | 049-CRA | 15 |
| 26 | 051-CRA | 16 |
| 36 | 083-CRA | 83 |
| 36 | 083-CRA | 145 |
| 41 | 09-079 | 81 |
| 45 | 11-058 | 132 |
| 45 | 11-058 | 129 |
| 55 | A1SC025 | 75 |
| 56 | A2SC031 | 122 |
| 56 | A2SC031 | 40 |
| 56 | A2SC031 | 206 |
| 58 | A2SC043 | 152 |
| 58 | A2SC043 | 148 |
| 58 | A2SC043 | 15 |
| 58 | A2SC043 | 34 |
| 58 | A2SC043 | 93 |
| 58 | A2SC043 | 38 |
| 58 | A2SC043 | 13 |
| 62 | A2SC59 | 45 |
| 63 | A3SC066 | 35 |
| 64 | A3SC089 | 44 |
| 66 | A3SC72 | 75 |
| 68 | A3SC78 | 107 |
| 68 | A3SC78 | 105 |
| 68 | A3SC78 | 79 |
| 68 | A3SC78 | 51 |
| 68 | A3SC78 | 38 |
| 68 | A3SC78 | 39 |
| 68 | A3SC78 | 16 |
| 68 | A3SC78 | 79 |
| 68 | A3SC78 | 83 |
| 68 | A3SC78 | 37 |
| 77 | A4SC118 | 32 |
| 81 | A5SC149 | 30 |
| 82 | B1SC153 | 106 |
| 82 | B1SC153 | 22 |
| 89 | B2SC183 | 128 |
| 90 | B2SC184 | 111 |
| 90 | B2SC184 | 13.7 |
| 90 | B2SC184 | 21 |
| 91 | B2SC186 | 5.9 |
| 93 | B2SC189 | 37 |
| 93 | B2SC189 | 80 |
| 93 | B2SC189 | 16 |
| 99 | B3SC216 | 113 |
| 100 | B3SC221 | 32 |
| 103 | B3SC235 | 153 |
| 103 | B3SC235 | 14 |
| 103 | B3SC235 | 112 |
| 103 | B3SC235 | 24 |
| 111 | B4SC520 | 153 |
| 113 | B5SC297 | 17 |
| 113 | B5SC297 | 7.7 |
| 113 | B5SC297 | 21 |
| 118 | C1SC0324 | 8 |
| 121 | C2SC338 | 83 |
| 124 | C3SC367 | 13.7 |
| 124 | C3SC367 | 73 |
| 127 | C3SC390 | 35 |
| 127 | C3SC390 | 79 |
| 134 | C5SC423 | 6.8 |
| 134 | C5SC423 | 18 |
| 134 | C5SC423 | 40 |
| 137 | C5SC434 | 154 |
| 138 | C5SC435 | 28 |
| 139 | C5SC438 | 156 |
| 139 | C5SC438 | 129 |
| 139 | C5SC438 | 29 |
| 139 | C5SC438 | 44 |
| 139 | C5SC438 | 73 |
| 139 | C5SC438 | 19 |
| 139 | C5SC438 | 32 |
| 139 | C5SC438 | 131 |
| 139 | C5SC438 | 32 |
| 142 | D1SC459 | 18 |
| 143 | D1SC461 | 13 |
| 147 | D3SC525 | 75 |
| 149 | D3SC532 | 18 |
| 151 | D4SC551 | 110 |
| 151 | D4SC551 | 13.1 |
| 160 | E2SC634 | 129 |
| 160 | E2SC634 | 139 |
| 160 | E2SC634 | 77 |
| 169 | JI-133 | 113 |
| 170 | JI-178 | 15 |
| 170 | JI-178 | 116 |
| 171 | JI-21 | 32 |
| 173 | JI-3562 | 2.4 |
| 174 | JI-4883 | 123 |
| 174 | JI-4883 | 92 |
| 174 | JI-4883 | 15 |
| 176 | JI-4892 | 22 |
| 182 | JI-4946 | 16 |
| 188 | JI-4985 | 28 |
| 188 | JI-4985 | 7 |
| 188 | JI-4985 | 5 |
| 188 | JI-4985 | 34 |
| 190 | JI-5022 | 85 |
| 190 | JI-5022 | 32 |
| 190 | JI-5022 | 81 |
| 190 | JI-5022 | 81 |
| 203 | JI-6476 | 19 |
| 205 | JIKEI-47F | 108 |
| 205 | JIKEI-47F | 125 |
| 205 | JIKEI-47F | 14.3 |
| 205 | JIKEI-47F | 6.3 |
| 205 | JIKEI-47F | 5 |
| 205 | JIKEI-47F | 8 |
| 211 | KU-1116 | 82 |
| 211 | KU-1116 | 117 |
| 211 | KU-1116 | 83 |
| 211 | KU-1116 | 40 |
| 211 | KU-1116 | 13 |
| 212 | KU-1118 | 3 |
| 213 | KU-1120 | 16 |
| 216 | KU-1265 | 153 |
| 219 | KU-1301 | 80 |
| 221 | KU-1388 | 5 |
| 221 | KU-1388 | 32 |
| 223 | KU-1415 | 18 |
| 225 | KU-1421 | 30 |
| 226 | KU-1446 | 92 |
| 228 | KU-1562 | 15 |
| 234 | KU-2133 | 5 |
| 237 | KU-2181 | 98 |
| 237 | KU-2181 | 92 |
| 237 | KU-2181 | 93 |
| 237 | KU-2181 | 34 |
| 237 | KU-2181 | 92 |
| 246 | KU-2594 | 118 |
| 249 | KU-2627 | 5 |
| 250 | KU-2636 | 15 |
| 260 | KU-2941 | 5 |
| 261 | KU-2945 | 5 |
| 262 | KU-2958 | 81 |
| 262 | KU-2958 | 79 |
| 281 | KU-907 | 40 |
| 286 | T74-221 | 28 |
| 294 | T75-353 | 63 |
| 294 | T75-353 | 74 |
| 296 | T75-362 | 5 |
| 300 | T75-387 | 140 |
| 300 | T75-387 | 17 |
| 313 | T76-677 | 21 |
| 314 | T76-714 | 109 |
| 325 | T77-438 | 28 |
| 343 | T78-220 | 116 |
| 350 | T78-264 | 32 |
| 352 | T78-280 | 74 |
| 359 | T78-48 | 76 |
| 362 | T79-112 | 109 |
| 366 | T79-164 | 14 |
| 372 | T79-27 | 90 |
| 373 | T79-28 | 2 |
| 375 | T79-288 | 156 |
| 375 | T79-288 | 15 |
| 378 | T79-311 | 124 |
| 381 | T79-92 | 5.7 |
| 381 | T79-92 | 15 |
| 387 | T81-293 | 5 |
| 515 | Tohoku-1063 | 111 |
| 515 | Tohoku-1063 | 93 |
| 515 | Tohoku-1063 | 81 |
| 515 | Tohoku-1063 | 82 |
| 517 | Tohoku-1082 | 80 |
| 520 | Tohoku-1199 | 18 |
| 520 | Tohoku-1199 | 19 |
| 521 | Tohoku-1203 | 2.8 |
| 521 | Tohoku-1203 | 17 |
| 524 | Tohoku-1299 | 100 |
| 524 | Tohoku-1299 | 90 |
| 528 | Tohoku-1475 | 45 |
| 535 | Tohoku-2088 | 81 |
| 536 | Tohoku-235 | 80 |
| 536 | Tohoku-235 | 53 |
| 536 | Tohoku-235 | 40 |
| 536 | Tohoku-235 | 96 |
| 536 | Tohoku-235 | 40 |
| 536 | Tohoku-235 | 80 |
| 536 | Tohoku-235 | 37 |
| 537 | Tohoku-2480 | 80 |
| 537 | Tohoku-2480 | 15 |
| 541 | Tohoku-2544 | 4 |
| 541 | Tohoku-2544 | 3 |
| 543 | Tohoku-2748 | 92 |
| 546 | Tohoku-302 | 124 |
| 550 | Tohoku-3264 | 20 |
| 553 | Tohoku-3278 | 88 |
| 555 | Tohoku-3290 | 38 |
| 556 | Tohoku-3296 | 17 |
| 563 | Tohoku-3345 | 5 |
| 563 | Tohoku-3345 | 93 |
| 566 | Tohoku-3375 | 13.7 |
| 567 | Tohoku-3466 | 21 |
| 570 | Tohoku-395 | 111 |
| 570 | Tohoku-395 | 95 |
| 574 | Tohoku-453 | 93 |
| 576 | Tohoku-478 | 83 |
| 576 | Tohoku-478 | 50 |
| 577 | Tohoku-481 | 113 |
| 578 | Tohoku-484 | 73 |
| 580 | Tohoku-508 | 106 |
| 580 | Tohoku-508 | 104 |
| 580 | Tohoku-508 | 80 |
| 580 | Tohoku-508 | 50 |
| 580 | Tohoku-508 | 39 |
| 580 | Tohoku-508 | 90 |
| 580 | Tohoku-508 | 19 |
| 580 | Tohoku-508 | 40 |
| 580 | Tohoku-508 | 17 |
| 580 | Tohoku-508 | 14 |
| 580 | Tohoku-508 | 74 |
| 580 | Tohoku-508 | 78 |
| 580 | Tohoku-508 | 79 |
| 580 | Tohoku-508 | 6 |
| 581 | Tohoku-522 | 78 |
| 582 | Tohoku-553 | 83 |
| 582 | Tohoku-553 | 110 |
| 582 | Tohoku-553 | 78 |
| 582 | Tohoku-553 | 55 |
| 582 | Tohoku-553 | 41 |
| 582 | Tohoku-553 | 17 |
| 583 | Tohoku-569 | 25 |
| 589 | Tohoku-965 | 90 |
| 391 | TU-1334 | 109 |
| 396 | TU-1342 | 100 |
| 402 | TU-1360 | 113 |
| 404 | TU-1367 | 111 |
| 404 | TU-1367 | 45 |
| 405 | TU-1369 | 109 |
| 405 | TU-1369 | 5.9 |
| 410 | TU-1432 | 110 |
| 412 | TU-1443 | 36 |
| 412 | TU-1443 | 22 |
| 412 | TU-1443 | 17 |
| 412 | TU-1443 | 48 |
| 412 | TU-1443 | 66 |
| 417 | TU-1457 | 115 |
| 417 | TU-1457 | 121 |
| 417 | TU-1457 | 116 |
| 417 | TU-1457 | 118 |
| 423 | TU-1471 | 115 |
| 423 | TU-1471 | 113 |
| 429 | TU-1831 | 119 |
| 432 | TU-1849 | 15 |
| 432 | TU-1849 | 1.7 |
| 432 | TU-1849 | 19 |
| 433 | TU-1868 | 21 |
| 433 | TU-1868 | 16 |
| 434 | TU-1871 | 21 |
| 434 | TU-1871 | 39 |
| 439 | TU-2063 | 15 |
| 443 | TU-207 | 34 |
| 448 | TU-2081 | 39 |
| 448 | TU-2081 | 5 |
| 449 | TU-2084 | 38 |
| 452 | TU-2101 | 103 |
| 452 | TU-2101 | 89 |
| 452 | TU-2101 | 83 |
| 452 | TU-2101 | 95 |
| 452 | TU-2101 | 33 |
| 468 | TU-2139 | 36 |
| 474 | TU-2188 | 114 |
| 475 | TU-220 | 28 |
| 475 | TU-220 | 13 |
| 476 | TU-2218 | 16 |
| 476 | TU-2218 | 6.1 |
| 476 | TU-2218 | 109 |
| 476 | TU-2218 | 26 |
| 476 | TU-2218 | 18 |
| 478 | TU-227 | 45 |
| 482 | TU-2409 | 110 |
| 484 | TU-2440 | 21 |
| 484 | TU-2440 | 34 |
| 487 | TU-2544 | 6.1 |
| 491 | TU-2551 | 113 |
| 492 | TU-2556 | 121 |
| 493 | TU-2557 | 28 |
| 499 | TU-271 | 15 |
| 505 | TU-37 | 7.1 |
| 506 | TU-39 | 6.5 |
| 510 | TU-49 | 199 |
| 510 | TU-49 | 197 |
| 510 | TU-49 | 21 |

> finaldata <- read.csv("finaldata.csv", as.is = TRUE)

> head(finaldata)

ï..Var Catkey Val

1 5 024-CRA 100

2 5 024-CRA 93

3 9 029-CRA 85

4 9 029-CRA 104

5 9 029-CRA 103

6 9 029-CRA 75

> table(finaldata$Catkey)

024-CRA 029-CRA 031-CRA 034-CRA 037-CRA 049-CRA 051-CRA 083-CRA 09-079 11-058 A1SC025

2 13 2 1 1 2 1 2 1 2 1

A2SC031 A2SC043 A2SC59 A3SC066 A3SC089 A3SC72 A3SC78 A4SC118 A5SC149 B1SC153 B2SC183

3 7 1 1 1 1 10 1 1 2 1

B2SC184 B2SC186 B2SC189 B3SC216 B3SC221 B3SC235 B4SC520 B5SC297 C1SC0324 C2SC338 C3SC367

3 1 3 1 1 4 1 3 1 1 2

C3SC390 C5SC423 C5SC434 C5SC435 C5SC438 D1SC459 D1SC461 D3SC525 D3SC532 D4SC551 E2SC634

2 3 1 1 9 1 1 1 1 2 3

JI-133 JI-178 JI-21 JI-3562 JI-4883 JI-4892 JI-4946 JI-4985 JI-5022 JI-6476 JIKEI-47F

1 2 1 1 3 1 1 4 4 1 6

KU-1116 KU-1118 KU-1120 KU-1265 KU-1301 KU-1388 KU-1415 KU-1421 KU-1446 KU-1562 KU-2133

5 1 1 1 1 2 1 1 1 1 1

KU-2181 KU-2594 KU-2627 KU-2636 KU-2941 KU-2945 KU-2958 KU-907 T74-221 T75-353 T75-362

5 1 1 1 1 1 2 1 1 2 1

T75-387 T76-677 T76-714 T77-438 T78-220 T78-264 T78-280 T78-48 T79-112 T79-164 T79-27

2 1 1 1 1 1 1 1 1 1 1

T79-28 T79-288 T79-311 T79-92 T81-293 Tohoku-1063 Tohoku-1082 Tohoku-1199 Tohoku-1203 Tohoku-1299 Tohoku-1475

1 2 1 2 1 4 1 2 2 2 1

Tohoku-2088 Tohoku-235 Tohoku-2480 Tohoku-2544 Tohoku-2748 Tohoku-302 Tohoku-3264 Tohoku-3278 Tohoku-3290 Tohoku-3296 Tohoku-3345

1 7 2 2 1 1 1 1 1 1 2

Tohoku-3375 Tohoku-3466 Tohoku-395 Tohoku-453 Tohoku-478 Tohoku-481 Tohoku-484 Tohoku-508 Tohoku-522 Tohoku-553 Tohoku-569

1 1 2 1 2 1 1 14 1 6 1

Tohoku-965 TU-1334 TU-1342 TU-1360 TU-1367 TU-1369 TU-1432 TU-1443 TU-1457 TU-1471 TU-1831

1 1 1 1 2 2 1 5 4 2 1

TU-1849 TU-1868 TU-1871 TU-2063 TU-207 TU-2081 TU-2084 TU-2101 TU-2139 TU-2188 TU-220

3 2 2 1 1 2 1 5 1 1 2

TU-2218 TU-227 TU-2409 TU-2440 TU-2544 TU-2551 TU-2556 TU-2557 TU-271 TU-37 TU-39

5 1 1 2 1 1 1 1 1 1 1

TU-49

3

**Multivariate**

> af <- read.csv("Asian\_Fusion\_Data.csv", as.is = TRUE);

>

> af <- af[,c(1:48)]

>

> mydata <- na.omit(af)

>

> {

+ y<- c(1:length(mydata$Catkey))

+ z <- mydata$Catkey

+ for (i in y)

+ {

+ mdist <- mahalanobis(na.omit(mydata[i,c(4:48)]),center=colMeans(na.omit(mydata[-(i),c(4:48)])),cov(na.omit(mydata[-(i),c(4:48)])) )

+

+ if (length(mdist) > 0)

+ {

+ val2 <- 1-pchisq(mdist,46)

+ if(val2 < 0.01)

+ {

+ x <- c(x, z[which(mydata$Record==i)])

+ }

+

+ val2 <- NA

+ }

+

+ else

+ {

+ #Do Nothin

+ }

+ }

+

+ x

+

+ }

3 4 9 16 26

"0.000187945559159242" "0.00968309613871976" "1.04244997567449e-06" "2.4242110740702e-05" "0.00646468711494563"

32 36 46 47 55

"0.00235339120201439" "0" "0" "0" "4.73692152466176e-05"

56 58 60 63 65

"0" "0" "0.00614035796003598" "0.00775828264307843" "4.44089209850063e-16"

72 73 77 81 87

"0.00166914900664561" "0.0018166263866134" "0.00309185105223375" "0.00341216072102313" "0.000224327026795534"

91 93 98 103 108

"0.00574481083224798" "2.30382126820139e-07" "0.00171771048816061" "1.74527059471075e-12" "0.00665041236541386"

111 113 124 127 129

"0.00265742463278962" "0" "0.00166552090523098" "0.00956826472862948" "0.00104530732095853"

134 139 142 147 157

"1.19415043742244e-07" "7.2295537918432e-06" "8.7365535250683e-08" "0.0022308847016147" "0"

160 161 163 174 182

"0.00117109345593214" "5.23855847589516e-10" "0.00587219936833572" "0.00579203247592985" "0.00244423469783961"

188 194 196 205 241

"3.85457564470171e-05" "0.00383514117426498" "0.00481143115796756" "0" "0.000907627964022217"

259 294 297 300 307

"1.44656984168989e-08" "0.000832597515074363" "0.00728272612233938" "1.15581910709039e-09" "0.000529854530455176"

313 326 337 343 350

"0.00301061398526237" "0.00303126070184589" "1.95543107692764e-05" "0.00329429356133137" "0.000130168010095799"

359 372 373 381 387

"0.00156402235598996" "7.64189116675418e-05" "0" "0.00101234021022922" "0.00403314383114173"

412 417 430 432 434

"1.10834674771354e-11" "1.59813717814927e-10" "0.00170988582917209" "3.70019207213135e-05" "6.13554879497213e-05"

443 448 449 452 466

"2.69781358364085e-08" "0" "0.00150058877804327" "2.8657070805238e-07" "0.000504952895748034"

468 475 476 478 487

"8.6929946936376e-06" "0.00819966960148832" "0" "0.000838093204879042" "0.00269349968452892"

497 505 511 520 521

"3.88146494443298e-05" "1.68548198189455e-05" "0.00269705350888882" "1.15180785329461e-05" "1.96084259940221e-11"

537 539 541 544 552

"0" "0.000176608503194342" "1.68242086928672e-09" "0.00106540324245596" "5.77315972805081e-15"

553 565 570 580 581

"0.000239377794233775" "0.00513598436753793" "1.0988722931593e-08" "0" "5.91335869160048e-11"

583

"1.4364733323724e-05" "019-CRA" "021-CRA" "027-CRA" "035-CRA"

"049-CRA" "052-CRA" "081-CRA" "087-CRA" "089-CRA"

"090-CRA" "11-058" "76-720" "A2SC031" "A2SC056"

"A3SC72" "A4SC091" "A4SC102" "A4SC118" "B1SC153"

"B2SC181" "B2SC184" "B2SC188" "B3SC231" "B4SC243"

"B4SC253" "B5SC281" "C1SC0318" "C1SC305" "C3SC367"

"C5SC422" "C5SC427" "C5SC434" "C5SC435" "D3SC529"

"D5SC580" "JI-13" "JI-178" "JI-4923" "JI-4946"

"JI-4947" "JI-5211" "JI-5232" "JI-5737" "KU-2405"

"T75-353" "T75-365" "T76-615" "T76-677" "T77-198"

"T77-400" "T77-450" "T78-205" "T78-246" "T78-251"

"T78-258" "T78-337" "T79-107" "T79-198" "T79-28"

"T79-92" "T80-061" "019-CRA" "035-CRA" "052-CRA"

"089-CRA" "B1SC153" "B2SC188" "B5SC281" "C1SC0318"

"C1SC305" "C3SC367" "JI-4923" "KU-1562" "KU-3139"

"T75-351" "T76-671" "T77-94" "T79-144" "T79-228"

"TU-1334" "TU-1342" "TU-1343" "TU-1471" "TU-1479"

"TU-2099" "TU-2137" "TU-2389" "TU-2539" "Tohoku-1137"

"Tohoku-1139"