Cancer de mama

Por definir

Jose Felix Rojas Cabeza

2021 / August / 26

Table of Contents

# Data

## Data collection

> bc\_full <- read.csv(params$data1, stringsAsFactors = FALSE)  
> # dim(bc\_full)  
> # head(bc\_full[1:4])  
>   
> bc\_10 <- read.csv(params$data2, stringsAsFactors = FALSE)  
> # dim(bc\_10)  
> # head(bc\_10[1:4])  
>   
> names(bc\_10)[1]<-paste("ID") # Numero de codigo de muestra #ID  
> names(bc\_10)[2]<-paste("Clump\_thickness") # Espesor del grupo 1 - 10  
> names(bc\_10)[3]<-paste("Uniformty\_c\_size") # Uniformidad del tamano de la célula 1 - 10  
> names(bc\_10)[4]<-paste("Uniformty\_c\_shape") # Uniformidad de la forma de la célula 1 - 10   
> names(bc\_10)[5]<-paste("Marginal\_adhesion") # Adherencia marginal 1 - 10   
> names(bc\_10)[6]<-paste("Sing\_epith\_c\_size") # Tamano de la célula epitelial unica 1 - 10  
> names(bc\_10)[7]<-paste("Bare\_nuclei") # Nucleos desnudos 1 - 10  
> names(bc\_10)[8]<-paste("bland\_Chromatina") # Cromatina blanda 1 - 10   
> names(bc\_10)[9]<-paste("Normal\_nucleoli") # Nucleos normales 1 - 10   
> names(bc\_10)[10]<-paste("Mitosis") # Mitosis 1 - 10   
> names(bc\_10)[11]<-paste("Class") # Class (2 for Benign, 4 for malignant)

> dim(bc\_full)

[1] 569 32

> dim(bc\_10)

[1] 698 11

> head(bc\_full)

id radius\_mean texture\_mean perimeter\_mean area\_mean smoothness\_mean  
1 842302 17.99 10.38 122.80 1001.0 0.11840  
2 842517 20.57 17.77 132.90 1326.0 0.08474  
3 84300903 19.69 21.25 130.00 1203.0 0.10960  
4 84348301 11.42 20.38 77.58 386.1 0.14250  
5 84358402 20.29 14.34 135.10 1297.0 0.10030  
6 843786 12.45 15.70 82.57 477.1 0.12780  
 compactness\_mean concavity\_mean concave.points\_mean symmetry\_mean  
1 0.27760 0.3001 0.14710 0.2419  
2 0.07864 0.0869 0.07017 0.1812  
3 0.15990 0.1974 0.12790 0.2069  
4 0.28390 0.2414 0.10520 0.2597  
5 0.13280 0.1980 0.10430 0.1809  
6 0.17000 0.1578 0.08089 0.2087  
 fractal\_dimension\_mean radius\_se texture\_se perimeter\_se area\_se  
1 0.07871 1.0950 0.9053 8.589 153.40  
2 0.05667 0.5435 0.7339 3.398 74.08  
3 0.05999 0.7456 0.7869 4.585 94.03  
4 0.09744 0.4956 1.1560 3.445 27.23  
5 0.05883 0.7572 0.7813 5.438 94.44  
6 0.07613 0.3345 0.8902 2.217 27.19  
 smoothness\_se compactness\_se concavity\_se concave.points\_se symmetry\_se  
1 0.006399 0.04904 0.05373 0.01587 0.03003  
2 0.005225 0.01308 0.01860 0.01340 0.01389  
3 0.006150 0.04006 0.03832 0.02058 0.02250  
4 0.009110 0.07458 0.05661 0.01867 0.05963  
5 0.011490 0.02461 0.05688 0.01885 0.01756  
6 0.007510 0.03345 0.03672 0.01137 0.02165  
 fractal\_dimension\_se radius\_worst texture\_worst perimeter\_worst area\_worst  
1 0.006193 25.38 17.33 184.60 2019.0  
2 0.003532 24.99 23.41 158.80 1956.0  
3 0.004571 23.57 25.53 152.50 1709.0  
4 0.009208 14.91 26.50 98.87 567.7  
5 0.005115 22.54 16.67 152.20 1575.0  
6 0.005082 15.47 23.75 103.40 741.6  
 smoothness\_worst compactness\_worst concavity\_worst concave.points\_worst  
1 0.1622 0.6656 0.7119 0.2654  
2 0.1238 0.1866 0.2416 0.1860  
3 0.1444 0.4245 0.4504 0.2430  
4 0.2098 0.8663 0.6869 0.2575  
5 0.1374 0.2050 0.4000 0.1625  
6 0.1791 0.5249 0.5355 0.1741  
 symmetry\_worst fractal\_dimension\_worst diagnosis  
1 0.4601 0.11890 M  
2 0.2750 0.08902 M  
3 0.3613 0.08758 M  
4 0.6638 0.17300 M  
5 0.2364 0.07678 M  
6 0.3985 0.12440 M

> head(bc\_10)

ID Clump\_thickness Uniformty\_c\_size Uniformty\_c\_shape Marginal\_adhesion  
1 1002945 5 4 4 5  
2 1015425 3 1 1 1  
3 1016277 6 8 8 1  
4 1017023 4 1 1 3  
5 1017122 8 10 10 8  
6 1018099 1 1 1 1  
 Sing\_epith\_c\_size Bare\_nuclei bland\_Chromatina Normal\_nucleoli Mitosis Class  
1 7 10 3 2 1 2  
2 2 2 3 1 1 2  
3 3 4 3 7 1 2  
4 2 1 3 1 1 2  
5 7 10 9 7 1 4  
6 2 10 3 1 1 2

## Data exploration and preparation

> # 1234567890123456789012345678901234567890123456789012345678901234567890  
> bc <- bc\_full[,-1]   
>   
> bc\_s <- bc\_10[,-1]   
>   
> # Figura 1. Izq  
> #str(bc)  
>   
> table(is.na(bc\_s))

FALSE   
 6980

> for (i in 1:length(bc\_s$Class)){  
+ if (bc\_s$Class[i]==2) {  
+ bc\_s$Class[i]<-"B"}  
+ else { bc\_s$Class[i]<- "M"  
+ }}  
>   
> table(is.na(bc\_s))

FALSE   
 6980

> # EMPIEZAN A SALIR NA's

> bc\_s[,1] <- as.numeric(bc\_s[,1])  
> bc\_s[,2] <- as.numeric(bc\_s[,2])  
> bc\_s[,3] <- as.numeric(bc\_s[,3])  
> bc\_s[,4] <- as.numeric(bc\_s[,4])  
> bc\_s[,5] <- as.numeric(bc\_s[,5])  
> bc\_s[,6] <- as.numeric(bc\_s[,6])  
> bc\_s[,7] <- as.numeric(bc\_s[,7])  
> bc\_s[,8] <- as.numeric(bc\_s[,8])  
> bc\_s[,9] <- as.numeric(bc\_s[,9])  
> bc\_s[,10] <- as.character(bc\_s[,10])  
>   
> #str(bc\_s)  
> table(is.na(bc\_s))

FALSE TRUE   
 6964 16

> bc\_s <- na.omit(bc\_s)  
>   
> table(is.na(bc\_s))

FALSE   
 6820

> # se pierden 160 datos

> # Benign / Malignant  
> bc$diagnosis <- factor(bc$diagnosis, levels= c("B", "M"), labels=c("Benign", "Malignant"))  
> table(bc$diagnosis)

Benign Malignant   
 357 212

> bc\_s$Class <- factor(bc\_s$Class, levels= c("B", "M"), labels=c("Benign", "Malignant"))  
> table(bc\_s$Class)

Benign Malignant   
 443 239

> table(is.na(bc))

FALSE   
17639

> table(is.na(bc\_s))

FALSE   
 6820

> dim(bc\_s)

[1] 682 10

> table(is.na(bc\_s))

FALSE   
 6820

> dim(bc\_s)

[1] 682 10

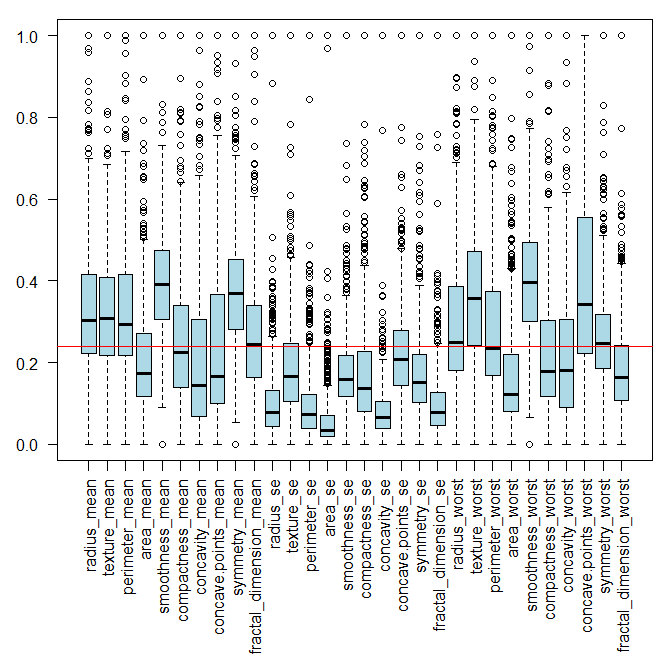
> normalize <- function(x) {  
+ return ((x - min(x)) / (max(x) - min(x)))  
+ }  
>   
> bc\_n <- as.data.frame(lapply(bc[1:(round(dim[2],0)-1)], normalize))  
>   
> #bc\_z <- as.data.frame(scale(bc[1:(round(dim[2],0)-1)]))  
>   
> bc\_s\_n <- as.data.frame(lapply(bc\_s[1:(round(dim\_s[2],0)-1)], normalize))  
>   
> #str(bc\_n)  
> str(bc\_s)

'data.frame': 682 obs. of 10 variables:  
 $ Clump\_thickness : num 5 3 6 4 8 1 2 2 4 1 ...  
 $ Uniformty\_c\_size : num 4 1 8 1 10 1 1 1 2 1 ...  
 $ Uniformty\_c\_shape: num 4 1 8 1 10 1 2 1 1 1 ...  
 $ Marginal\_adhesion: num 5 1 1 3 8 1 1 1 1 1 ...  
 $ Sing\_epith\_c\_size: num 7 2 3 2 7 2 2 2 2 1 ...  
 $ Bare\_nuclei : num 10 2 4 1 10 10 1 1 1 1 ...  
 $ bland\_Chromatina : num 3 3 3 3 9 3 3 1 2 3 ...  
 $ Normal\_nucleoli : num 2 1 7 1 7 1 1 1 1 1 ...  
 $ Mitosis : num 1 1 1 1 1 1 1 5 1 1 ...  
 $ Class : Factor w/ 2 levels "Benign","Malignant": 1 1 1 1 2 1 1 1 1 1 ...  
 - attr(\*, "na.action")= 'omit' Named int 23 40 139 145 158 164 235 249 275 292 ...  
 ..- attr(\*, "names")= chr "23" "40" "139" "145" ...

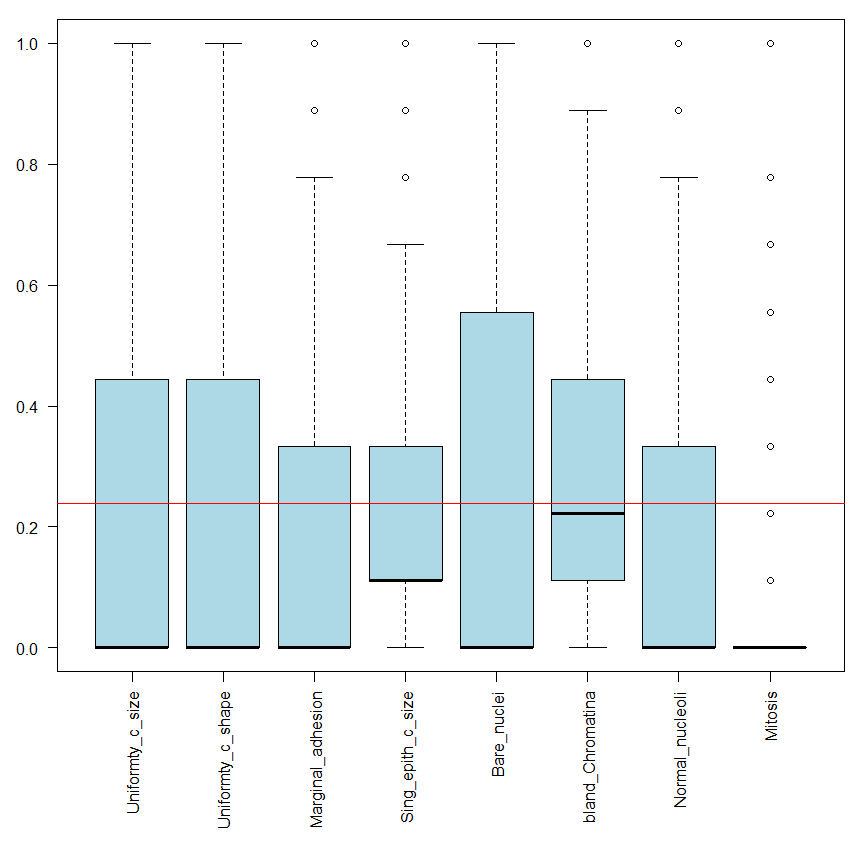
> head(bc\_s, 10)

Clump\_thickness Uniformty\_c\_size Uniformty\_c\_shape Marginal\_adhesion  
1 5 4 4 5  
2 3 1 1 1  
3 6 8 8 1  
4 4 1 1 3  
5 8 10 10 8  
6 1 1 1 1  
7 2 1 2 1  
8 2 1 1 1  
9 4 2 1 1  
10 1 1 1 1  
 Sing\_epith\_c\_size Bare\_nuclei bland\_Chromatina Normal\_nucleoli Mitosis  
1 7 10 3 2 1  
2 2 2 3 1 1  
3 3 4 3 7 1  
4 2 1 3 1 1  
5 7 10 9 7 1  
6 2 10 3 1 1  
7 2 1 3 1 1  
8 2 1 1 1 5  
9 2 1 2 1 1  
10 1 1 3 1 1  
 Class  
1 Benign  
2 Benign  
3 Benign  
4 Benign  
5 Malignant  
6 Benign  
7 Benign  
8 Benign  
9 Benign  
10 Benign

> # Figura 1. der  
> par(mar=c(11,3,1,1))  
>   
> means\_n <- as.numeric((lapply(bc\_n, mean)))  
>   
> boxplot(bc\_n[1:30], las=2, col="lightblue")  
> # , main="Figura 2. Distribucion de las variables del dataset normalizadas"  
> abline(h=mean(means\_n), col="red")

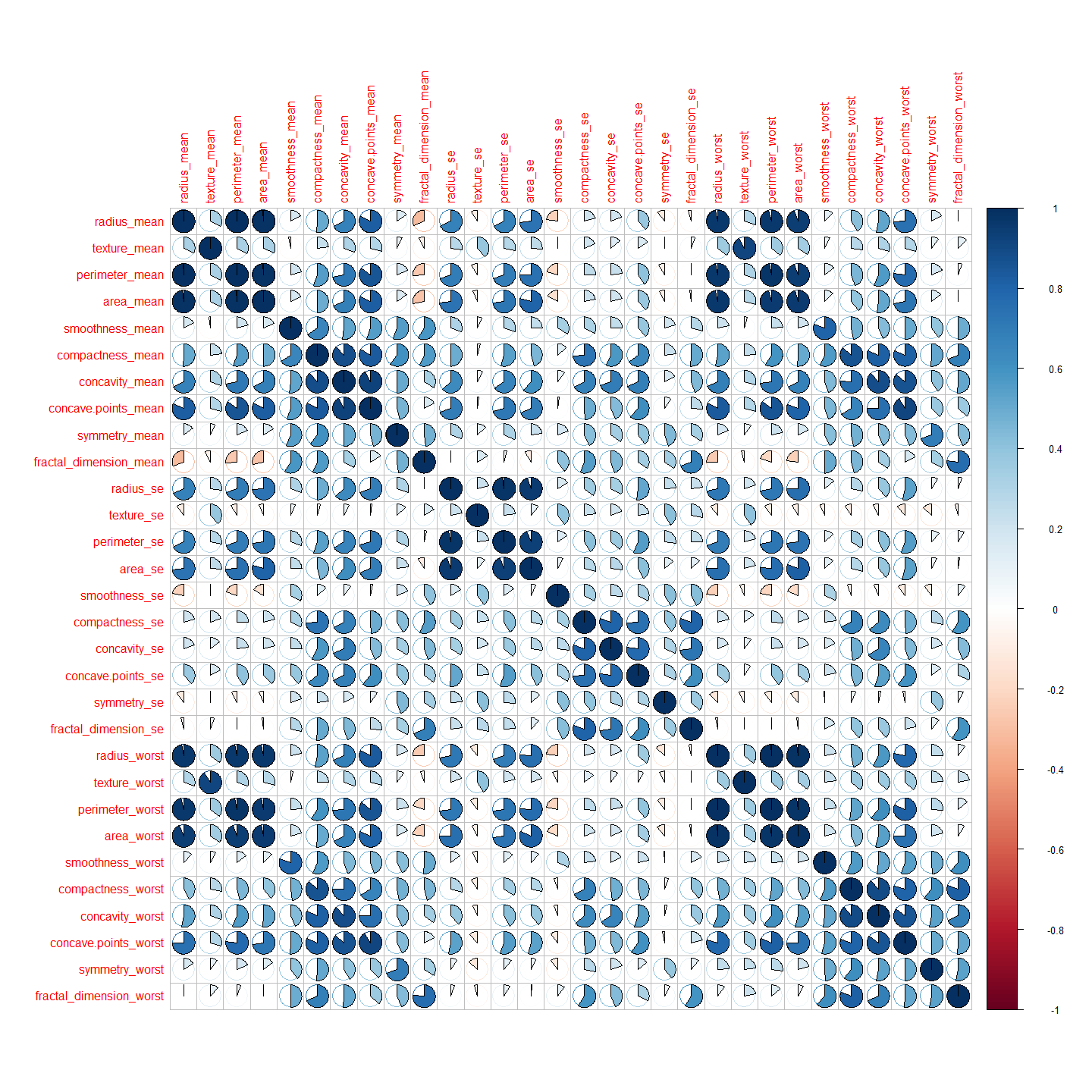


> par(mar=c(10,3,1,1))  
>   
> means\_n\_s <- as.numeric((lapply(bc\_s\_n, mean)))  
>   
> boxplot(bc\_s\_n[,-1], las=2, col="lightblue")  
> # , main="Figura 2. Distribucion de las variables del dataset normalizadas"  
> abline(h=mean(means\_n\_s), col="red")

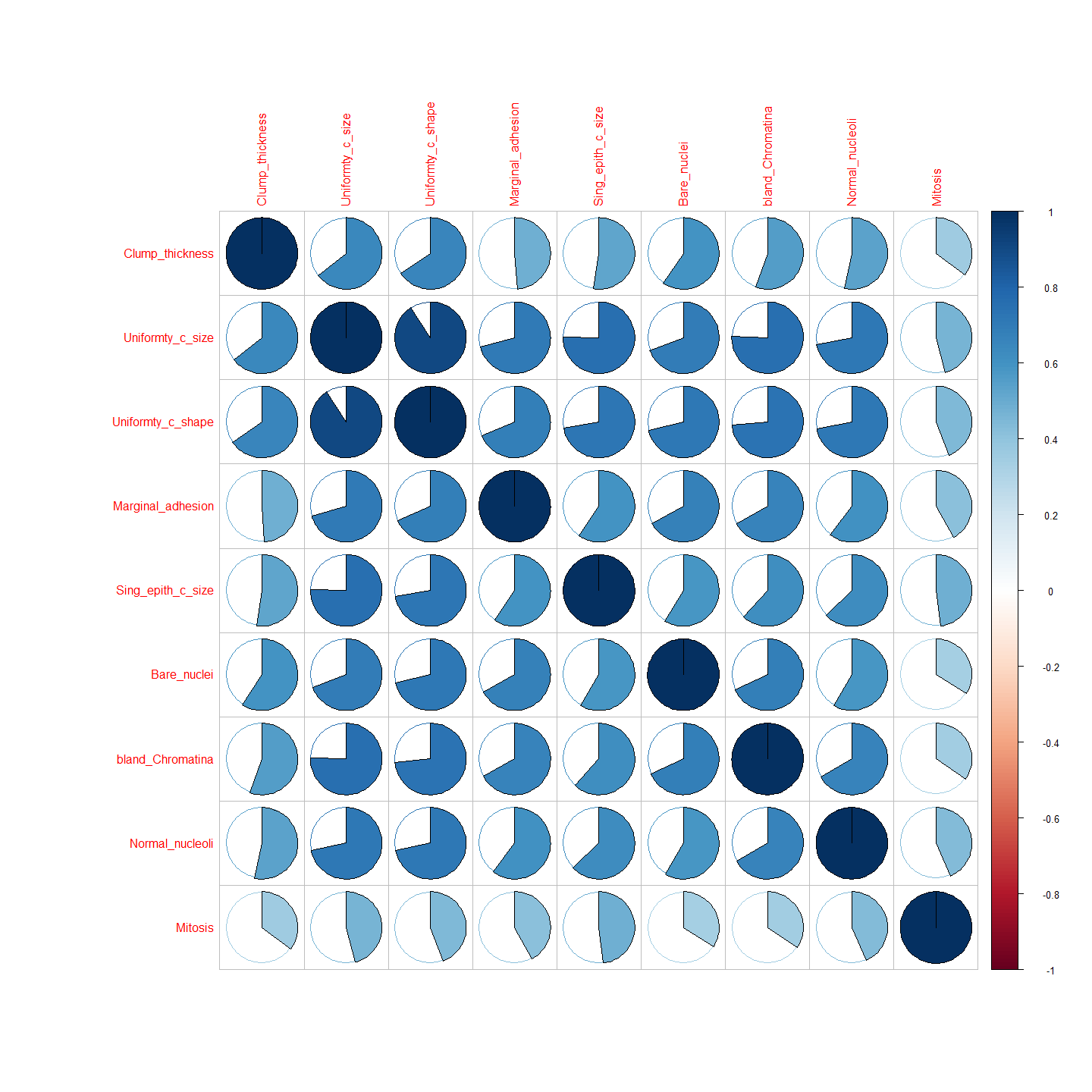


# Correlogram

> library(corrplot)  
> par(mar=c(11,3,1,1))  
>   
> C <- cor(round(as.matrix(bc\_n[1:30]), 4))  
>   
> corrplot(C, method = "pie")



> C2 <- cor(round(as.matrix(bc\_s\_n[1:9]),4))  
>   
> corrplot(C2, method = "pie")



# Creation training and test datasets for 10-fold crossvalidation

> # 2021-07-19  
> library(caret)  
>   
> bc\_n\_full <- cbind(bc\_n, bc$diagnosis)  
>   
> names(bc\_n\_full)[31] <- "diagnosis"  
>   
> dim(bc\_n\_full)

[1] 569 31

> set.seed(params$seed\_train)  
>   
> inTrain <- createDataPartition( y = 1:nrow(bc\_n\_full),  
+ p = params$ptrain,  
+ list = FALSE )  
>   
> # KNN, SVM, RF  
> train <- bc\_n\_full[inTrain, ]  
> test <- bc\_n\_full[-inTrain, ]  
>   
> table(train[,31])

Benign Malignant   
 240 141

> table(test[,31])

Benign Malignant   
 117 71

> #ANN  
> #Data Partition  
> set.seed(params$seed\_alg)  
>   
>   
> inTrain\_ann <- createDataPartition(y=bc\_n\_full$diagnosis, p=round(params$ptrain,4), list=FALSE)  
>   
>   
> # Normalized dataset  
> ann\_data\_n <- cbind(bc\_n, bc$diagnosis)  
> names(ann\_data\_n)[31] <- "diagnosis"  
>   
> train\_ann <- ann\_data\_n[inTrain\_ann, ]  
> test\_ann <- ann\_data\_n[-inTrain\_ann, ]  
>   
> #train.set <- dataset[inTrain,]  
> #test.set <- dataset[-inTrain,]  
>   
> ( c(nrow(train)/nrow(test), nrow(train\_ann)/nrow(test\_ann)) )

[1] 2.026596 2.026596

> # should be around 2  
>   
> # SMALL DATASET  
> # bc\_s\_n refers to Breast Cancer Small dataset Normalized  
>   
> bc\_s\_n\_full <- cbind(bc\_s\_n, bc\_s$Class)  
>   
> names(bc\_s\_n\_full)[10] <- "diagnosis"  
>   
>   
> dim(bc\_s\_n\_full)

[1] 682 10

> inTrain\_s <- createDataPartition( y = 1:nrow(bc\_s\_n\_full),  
+ p = params$ptrain,  
+ list = FALSE )  
>   
> train\_s <- bc\_s\_n\_full[inTrain, ]  
> test\_s <- bc\_s\_n\_full[-inTrain, ]  
>   
>   
>   
> table(train\_s[,10])

Benign Malignant   
 236 145

> table(test\_s[,10])

Benign Malignant   
 207 94

> #ANN  
> #Data Partition  
> inTrain\_s\_ann <- createDataPartition(y=1:nrow(bc\_s\_n\_full),   
+ p=params$ptrain,   
+ list=FALSE)  
>   
>   
> # Normalized dataset  
> ann\_data\_s\_n <- bc\_s\_n\_full   
>   
> train\_s\_ann <- ann\_data\_s\_n[inTrain\_ann,]  
> test\_s\_ann <- ann\_data\_s\_n[-inTrain\_ann,]  
>   
> #train.set <- dataset[inTrain,]  
> #test.set <- dataset[-inTrain,]  
>   
> ( c(nrow(train\_s)/nrow(test\_s), nrow(train\_s\_ann)/nrow(test\_s\_ann)) )

[1] 1.265781 1.265781

> table(train\_s$diagnosis)

Benign Malignant   
 236 145

> table(test\_s$diagnosis)

Benign Malignant   
 207 94

# KNN

> set.seed(params$seed\_alg)  
>   
> start\_time <- Sys.time()  
>   
> ctrl <- trainControl(method="repeatedcv",repeats = 10, number = 10,   
+ classProbs = T, savePredictions = T)   
> model\_knn <- train(diagnosis ~ ., data = train, method = "knn",   
+ trControl = ctrl, preProcess = c("center","scale"),  
+ tuneLength = 20)  
>   
> end\_time <- Sys.time()  
>   
> (knn\_time <- end\_time - start\_time)

Time difference of 17.21593 secs

## Confusion matrix KNN

> prediction <- predict(model\_knn, test)   
> ct <- table(prediction, test$diagnosis)   
> (cm\_knn <- confusionMatrix(ct, positive="Malignant"))

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 115 6  
 Malignant 2 65  
   
 Accuracy : 0.9574   
 95% CI : (0.9179, 0.9815)  
 No Information Rate : 0.6223   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9085   
   
 Mcnemar's Test P-Value : 0.2888   
   
 Sensitivity : 0.9155   
 Specificity : 0.9829   
 Pos Pred Value : 0.9701   
 Neg Pred Value : 0.9504   
 Prevalence : 0.3777   
 Detection Rate : 0.3457   
 Detection Prevalence : 0.3564   
 Balanced Accuracy : 0.9492   
   
 'Positive' Class : Malignant

# SVM

> set.seed(params$seed\_alg)  
>   
> start\_time <- Sys.time()  
>   
> model\_svm <- train(diagnosis ~ ., train, method='svmLinear',  
+ trControl= trainControl(method='cv',number=10,repeats = 10,  
+ classProbs = T,  
+ savePredictions = T),  
+ tuneGrid= NULL, trace = FALSE)  
>   
> end\_time <- Sys.time()  
>   
> (svm\_time <- end\_time - start\_time)

Time difference of 1.563604 secs

## Confusion matrix SVM

> prediction <- predict(model\_svm, test)   
> ct <- table(prediction, test$diagnosis)   
> (cm\_svm <- confusionMatrix(ct, positive="Malignant"))

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 115 2  
 Malignant 2 69  
   
 Accuracy : 0.9787   
 95% CI : (0.9464, 0.9942)  
 No Information Rate : 0.6223   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9547   
   
 Mcnemar's Test P-Value : 1   
   
 Sensitivity : 0.9718   
 Specificity : 0.9829   
 Pos Pred Value : 0.9718   
 Neg Pred Value : 0.9829   
 Prevalence : 0.3777   
 Detection Rate : 0.3670   
 Detection Prevalence : 0.3777   
 Balanced Accuracy : 0.9774   
   
 'Positive' Class : Malignant

# RF

> set.seed(params$seed\_alg)  
>   
> ctrl <- trainControl( method = "repeatedcv",  
+ number = 10,  
+ repeats = 10,  
+ summaryFunction = defaultSummary,  
+ verboseIter = FALSE,  
+ classProbs = TRUE,  
+ savePredictions = TRUE)  
> # repeats=3 )  
>   
> ## Tunegrid for Random Forest  
> # mtry define cuantas variables se seleccionan al azar en cada split. Por  
> # defecto sqrt(n.variables)  
> grid\_rf <- expand.grid(.mtry = c(2,4,8,16))  
>   
> start\_time <- Sys.time()  
>   
> model\_rf <- train (diagnosis ~ .,  
+ data = train,  
+ method ="rf",  
+ trControl=ctrl,  
+ #tuneLength = 9,  
+ tuneGrid = grid\_rf,  
+ metric="Accuracy",  
+ prePoc = c("center", "scale"),  
+ verbose =FALSE,  
+ trace = FALSE  
+ )  
>   
> end\_time <- Sys.time()  
>   
> (rf\_time <- end\_time - start\_time)

Time difference of 1.033657 mins

## Confusion matrix RF

> pred\_rf <- predict (model\_rf, newdata = test)  
>   
> (cm\_rf <- confusionMatrix(data=pred\_rf, test$diagnosis, positive = "Malignant"))

Confusion Matrix and Statistics  
  
 Reference  
Prediction Benign Malignant  
 Benign 114 5  
 Malignant 3 66  
   
 Accuracy : 0.9574   
 95% CI : (0.9179, 0.9815)  
 No Information Rate : 0.6223   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.909   
   
 Mcnemar's Test P-Value : 0.7237   
   
 Sensitivity : 0.9296   
 Specificity : 0.9744   
 Pos Pred Value : 0.9565   
 Neg Pred Value : 0.9580   
 Prevalence : 0.3777   
 Detection Rate : 0.3511   
 Detection Prevalence : 0.3670   
 Balanced Accuracy : 0.9520   
   
 'Positive' Class : Malignant

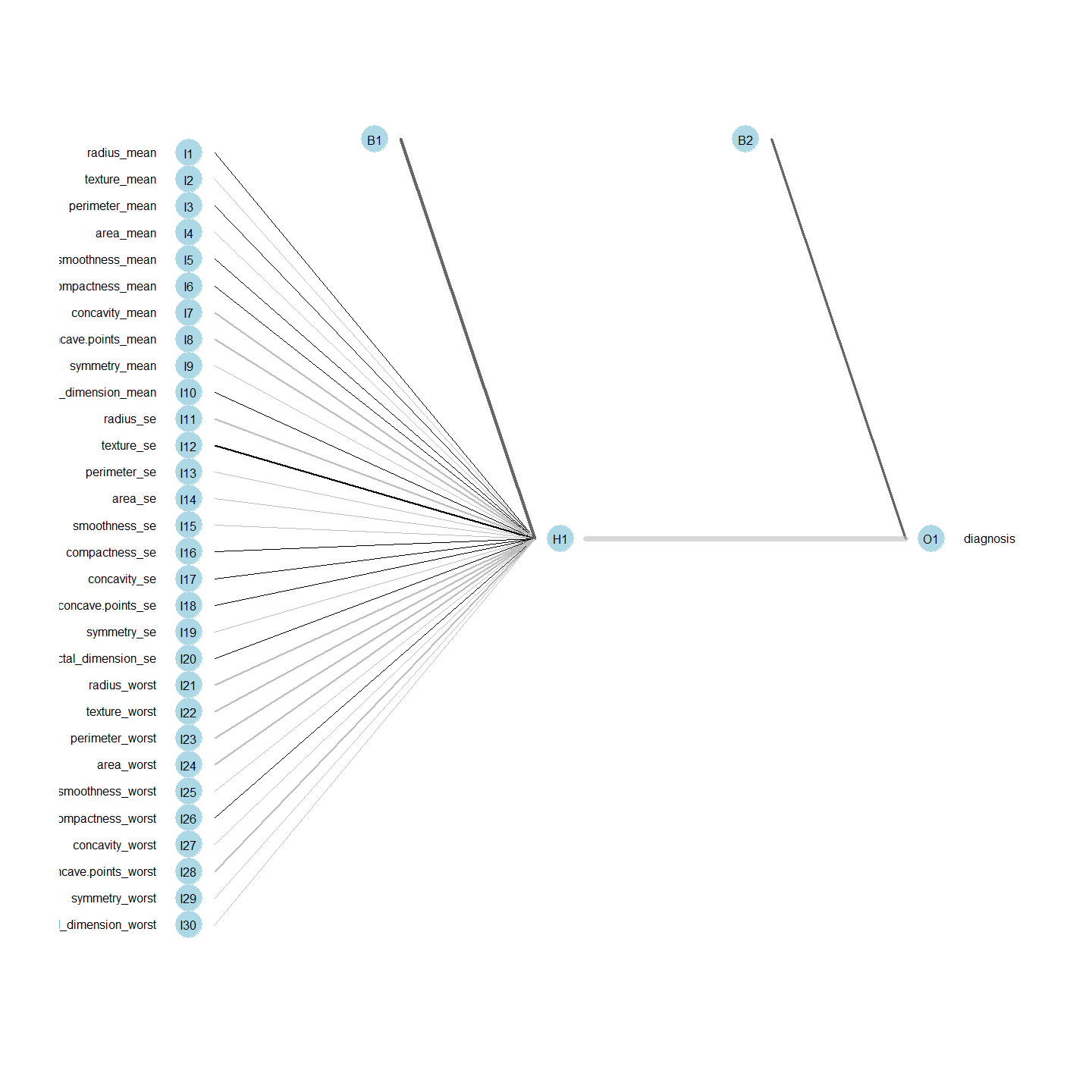
# ANN

> library(nnet)  
> library(NeuralNetTools)

> # 10 fold-crossvalidation model  
>   
> start\_time <- Sys.time()  
>   
> model\_ann <- train(diagnosis ~ ., train\_ann, method='nnet',   
+ trControl= trainControl(method='cv', number=10),   
+ tuneGrid= NULL, tuneLength=10 ,trace = FALSE)  
>   
>   
> end\_time <- Sys.time()  
>   
> (ann\_time <- end\_time - start\_time)

Time difference of 2.008861 mins

> plotnet(model\_ann, alpha=0.6)



> summary(model\_ann)

a 30-1-1 network with 33 weights  
options were - entropy fitting decay=0.04216965  
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1 i5->h1 i6->h1 i7->h1 i8->h1 i9->h1   
 6.23 0.41 -1.03 0.24 -0.70 0.58 1.49 -2.45 -2.83 -0.39   
i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1 i18->h1 i19->h1   
 1.69 -2.78 2.21 -1.14 -1.79 -1.61 1.67 0.66 0.38 0.00   
i20->h1 i21->h1 i22->h1 i23->h1 i24->h1 i25->h1 i26->h1 i27->h1 i28->h1 i29->h1   
 0.92 -3.46 -3.20 -2.36 -2.93 -1.51 0.45 -0.95 -2.53 -1.62   
i30->h1   
 -1.66   
 b->o h1->o   
 5.68 -11.97

> prediction\_ann <- predict(model\_ann, test\_ann[-31]) # predict  
> table(prediction, test\_ann$diagnosis) # compare

prediction Benign Malignant  
 Benign 75 42  
 Malignant 43 28

> # predict can also return the probability for each class:  
> prediction\_ann\_prob <- predict(model\_ann, test\_ann[-31], type="prob")   
>   
> head(prediction)

[1] Malignant Malignant Malignant Malignant Malignant Malignant  
Levels: Benign Malignant

# Confusion Matrix ANN

> (cm\_ann <- confusionMatrix(predict(model\_ann, test\_ann[,-31],type="raw"),  
+ test\_ann[,31], positive = "Malignant"))

Confusion Matrix and Statistics  
  
 Reference  
Prediction Benign Malignant  
 Benign 116 2  
 Malignant 2 68  
   
 Accuracy : 0.9787   
 95% CI : (0.9464, 0.9942)  
 No Information Rate : 0.6277   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9545   
   
 Mcnemar's Test P-Value : 1   
   
 Sensitivity : 0.9714   
 Specificity : 0.9831   
 Pos Pred Value : 0.9714   
 Neg Pred Value : 0.9831   
 Prevalence : 0.3723   
 Detection Rate : 0.3617   
 Detection Prevalence : 0.3723   
 Balanced Accuracy : 0.9772   
   
 'Positive' Class : Malignant

> #predict(model, test.set[,31],type="raw")  
> #test.set[,31]

# KNN Small Dataset

> set.seed(params$seed\_alg)  
>   
> start\_time <- Sys.time()  
>   
> ctrl <- trainControl(method="repeatedcv",repeats = 10, number = 10,   
+ classProbs = T, savePredictions = T)   
> model\_knn\_s <- train(diagnosis ~ ., data = train\_s, method = "knn",   
+ trControl = ctrl, preProcess = c("center","scale"),  
+ tuneLength = 20)  
>   
> end\_time <- Sys.time()  
>   
> (knn\_time\_s <- end\_time - start\_time)

Time difference of 12.8608 secs

## Confusion matrix KNN Small Dataset

> prediction <- predict(model\_knn\_s, test\_s)   
> ct <- table(prediction, test\_s$diagnosis)   
> (cm\_knn\_s <- confusionMatrix(ct, positive="Malignant"))

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 201 2  
 Malignant 6 92  
   
 Accuracy : 0.9734   
 95% CI : (0.9483, 0.9885)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9388   
   
 Mcnemar's Test P-Value : 0.2888   
   
 Sensitivity : 0.9787   
 Specificity : 0.9710   
 Pos Pred Value : 0.9388   
 Neg Pred Value : 0.9901   
 Prevalence : 0.3123   
 Detection Rate : 0.3056   
 Detection Prevalence : 0.3256   
 Balanced Accuracy : 0.9749   
   
 'Positive' Class : Malignant

# SVM Small Dataset

> set.seed(params$seed\_alg)  
>   
> start\_time <- Sys.time()  
>   
> model\_svm\_s <- train(diagnosis ~ ., train\_s, method='svmLinear',  
+ trControl= trainControl(method='cv',number=10,repeats = 10,  
+ classProbs = T,  
+ savePredictions = T),  
+ tuneGrid= NULL, trace = FALSE)  
>   
> end\_time <- Sys.time()  
>   
> (svm\_time\_s <- end\_time - start\_time)

Time difference of 0.6280019 secs

## Confusion matrix SVM Small Dataset

> prediction <- predict(model\_svm\_s, test\_s)   
> ct <- table(prediction, test\_s$diagnosis)   
> (cm\_svm\_s <- confusionMatrix(ct, positive="Malignant"))

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 200 3  
 Malignant 7 91  
   
 Accuracy : 0.9668   
 95% CI : (0.9398, 0.984)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9235   
   
 Mcnemar's Test P-Value : 0.3428   
   
 Sensitivity : 0.9681   
 Specificity : 0.9662   
 Pos Pred Value : 0.9286   
 Neg Pred Value : 0.9852   
 Prevalence : 0.3123   
 Detection Rate : 0.3023   
 Detection Prevalence : 0.3256   
 Balanced Accuracy : 0.9671   
   
 'Positive' Class : Malignant

# RF Small Dataset

> set.seed(params$seed\_alg)  
>   
> ctrl <- trainControl( method = "repeatedcv",  
+ number = 10,  
+ repeats = 10,  
+ summaryFunction = defaultSummary,  
+ verboseIter = FALSE,  
+ classProbs = TRUE,  
+ savePredictions = TRUE)  
> # repeats=3 )  
>   
> ## Tunegrid for Random Forest  
> # mtry define cuantas variables se seleccionan al azar en cada split. Por  
> # defecto sqrt(n.variables)  
> grid\_rf <- expand.grid(.mtry = c(2,4,8,16))  
>   
> start\_time <- Sys.time()  
>   
> model\_rf\_s <- train (diagnosis ~ .,  
+ data = train\_s,  
+ method ="rf",  
+ trControl=ctrl,  
+ #tuneLength = 9,  
+ tuneGrid = grid\_rf,  
+ metric="Accuracy",  
+ prePoc = c("center", "scale"),  
+ verbose =FALSE,  
+ trace = FALSE  
+ )  
>   
> end\_time <- Sys.time()  
>   
> (rf\_time\_s <- end\_time - start\_time)

Time difference of 29.40851 secs

## Confusion matrix RF Small Dataset

> pred\_rf\_s <- predict (model\_rf\_s, newdata = test\_s)  
>   
> (cm\_rf\_s <- confusionMatrix(data=pred\_rf\_s, test\_s$diagnosis, positive = "Malignant"))

Confusion Matrix and Statistics  
  
 Reference  
Prediction Benign Malignant  
 Benign 200 0  
 Malignant 7 94  
   
 Accuracy : 0.9767   
 95% CI : (0.9527, 0.9906)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : < 2e-16   
   
 Kappa : 0.9469   
   
 Mcnemar's Test P-Value : 0.02334   
   
 Sensitivity : 1.0000   
 Specificity : 0.9662   
 Pos Pred Value : 0.9307   
 Neg Pred Value : 1.0000   
 Prevalence : 0.3123   
 Detection Rate : 0.3123   
 Detection Prevalence : 0.3355   
 Balanced Accuracy : 0.9831   
   
 'Positive' Class : Malignant

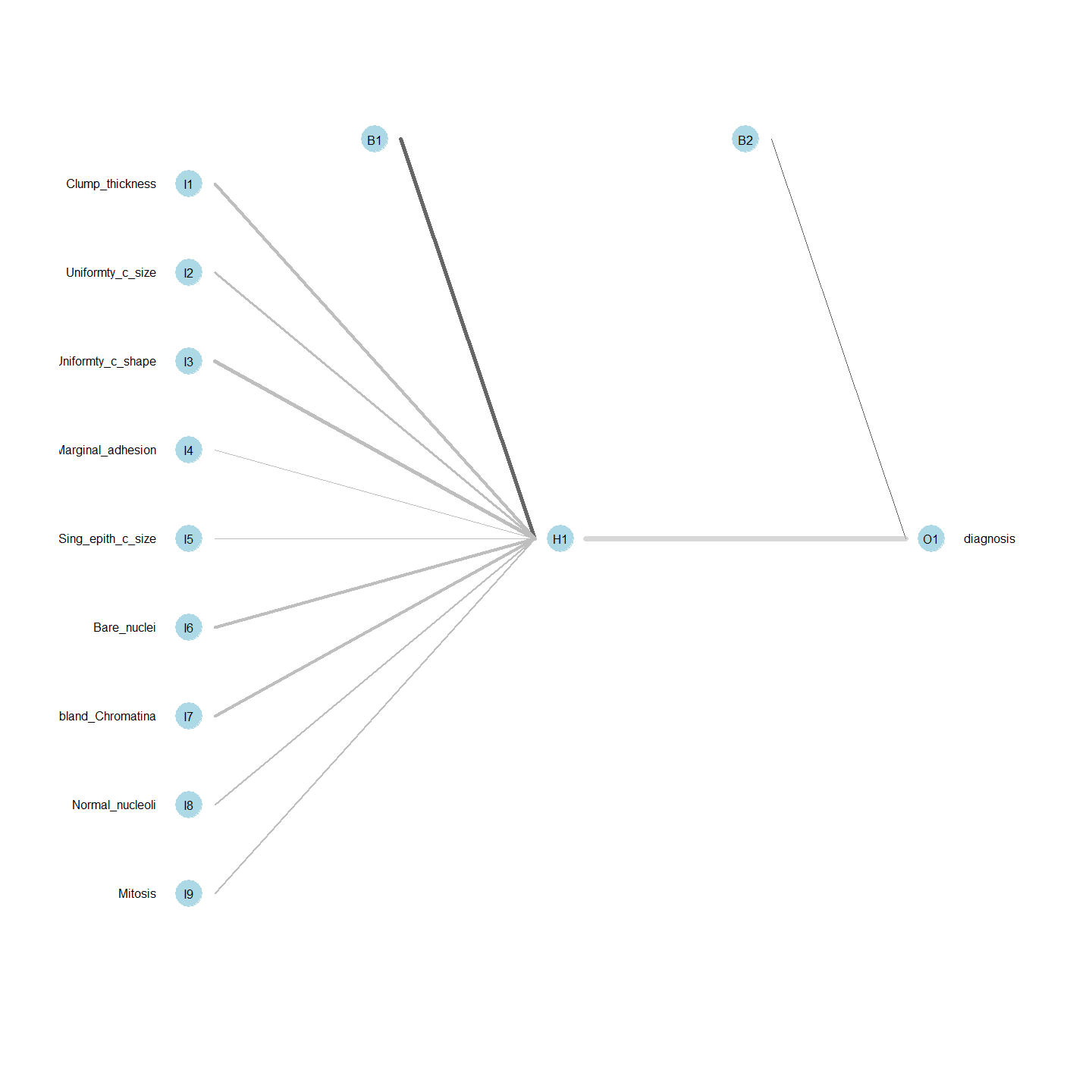
# ANN Small Dataset

> library(nnet)  
> library(NeuralNetTools)

> # 10 fold-crossvalidation model  
>   
> start\_time <- Sys.time()  
>   
> model\_ann\_s <- train(diagnosis ~ ., train\_s\_ann, method='nnet',   
+ trControl= trainControl(method='cv', number=10),   
+ tuneGrid= NULL, tuneLength=10 ,trace = FALSE)  
>   
> end\_time <- Sys.time()  
>   
> (ann\_time\_s <- end\_time - start\_time)

Time difference of 1.05702 mins

> plotnet(model\_ann\_s, alpha=0.6)



> summary(model\_ann)

a 30-1-1 network with 33 weights  
options were - entropy fitting decay=0.04216965  
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1 i5->h1 i6->h1 i7->h1 i8->h1 i9->h1   
 6.23 0.41 -1.03 0.24 -0.70 0.58 1.49 -2.45 -2.83 -0.39   
i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1 i18->h1 i19->h1   
 1.69 -2.78 2.21 -1.14 -1.79 -1.61 1.67 0.66 0.38 0.00   
i20->h1 i21->h1 i22->h1 i23->h1 i24->h1 i25->h1 i26->h1 i27->h1 i28->h1 i29->h1   
 0.92 -3.46 -3.20 -2.36 -2.93 -1.51 0.45 -0.95 -2.53 -1.62   
i30->h1   
 -1.66   
 b->o h1->o   
 5.68 -11.97

> prediction\_ann\_s <- predict(model\_ann\_s, test\_s\_ann[-10]) # predict  
> table(prediction, test\_s\_ann$diagnosis) # compare

prediction Benign Malignant  
 Benign 163 40  
 Malignant 44 54

> # predict can also return the probability for each class:  
> prediction\_ann\_s\_prob <- predict(model\_ann\_s, test\_s\_ann[-10], type="prob")   
>   
> head(prediction)

[1] Malignant Malignant Benign Benign Benign Benign   
Levels: Benign Malignant

> (cm\_ann\_s <- confusionMatrix(predict(model\_ann\_s, test\_s\_ann[,-10],type="raw"),  
+ test\_s\_ann[,10], positive = "Malignant"))#predict(model, test.set[,31],type="raw")

Confusion Matrix and Statistics  
  
 Reference  
Prediction Benign Malignant  
 Benign 203 1  
 Malignant 4 93  
   
 Accuracy : 0.9834   
 95% CI : (0.9617, 0.9946)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9617   
   
 Mcnemar's Test P-Value : 0.3711   
   
 Sensitivity : 0.9894   
 Specificity : 0.9807   
 Pos Pred Value : 0.9588   
 Neg Pred Value : 0.9951   
 Prevalence : 0.3123   
 Detection Rate : 0.3090   
 Detection Prevalence : 0.3223   
 Balanced Accuracy : 0.9850   
   
 'Positive' Class : Malignant

> #test.set[,31]

# TABLA 1

> print("KNN")

[1] "KNN"

> cm\_knn$overall[1]

Accuracy   
0.9574468

> cm\_knn$overall[2]

Kappa   
0.9084601

> cm\_knn$byClass

Sensitivity Specificity Pos Pred Value   
 0.9154930 0.9829060 0.9701493   
 Neg Pred Value Precision Recall   
 0.9504132 0.9701493 0.9154930   
 F1 Prevalence Detection Rate   
 0.9420290 0.3776596 0.3457447   
Detection Prevalence Balanced Accuracy   
 0.3563830 0.9491995

> print("SVM")

[1] "SVM"

> cm\_svm$overall[1]

Accuracy   
0.9787234

> cm\_svm$overall[2]

Kappa   
0.954737

> cm\_svm$byClass

Sensitivity Specificity Pos Pred Value   
 0.9718310 0.9829060 0.9718310   
 Neg Pred Value Precision Recall   
 0.9829060 0.9718310 0.9718310   
 F1 Prevalence Detection Rate   
 0.9718310 0.3776596 0.3670213   
Detection Prevalence Balanced Accuracy   
 0.3776596 0.9773685

> print("RF")

[1] "RF"

> cm\_rf$overall[1]

Accuracy   
0.9574468

> cm\_rf$overall[2]

Kappa   
0.9089699

> cm\_rf$byClass

Sensitivity Specificity Pos Pred Value   
 0.9295775 0.9743590 0.9565217   
 Neg Pred Value Precision Recall   
 0.9579832 0.9565217 0.9295775   
 F1 Prevalence Detection Rate   
 0.9428571 0.3776596 0.3510638   
Detection Prevalence Balanced Accuracy   
 0.3670213 0.9519682

> print("ANN")

[1] "ANN"

> cm\_ann$overall[1]

Accuracy   
0.9787234

> cm\_ann$overall[2]

Kappa   
0.9544794

> cm\_ann$byClass

Sensitivity Specificity Pos Pred Value   
 0.9714286 0.9830508 0.9714286   
 Neg Pred Value Precision Recall   
 0.9830508 0.9714286 0.9714286   
 F1 Prevalence Detection Rate   
 0.9714286 0.3723404 0.3617021   
Detection Prevalence Balanced Accuracy   
 0.3723404 0.9772397

Tabla 1 Small

> print("KNN")

[1] "KNN"

> cm\_knn\_s$overall[1]

Accuracy   
0.9734219

> cm\_knn\_s$overall[2]

Kappa   
0.9388336

> cm\_knn\_s$byClass

Sensitivity Specificity Pos Pred Value   
 0.9787234 0.9710145 0.9387755   
 Neg Pred Value Precision Recall   
 0.9901478 0.9387755 0.9787234   
 F1 Prevalence Detection Rate   
 0.9583333 0.3122924 0.3056478   
Detection Prevalence Balanced Accuracy   
 0.3255814 0.9748689

> print("SVM")

[1] "SVM"

> cm\_svm\_s$overall[1]

Accuracy   
0.9667774

> cm\_svm\_s$overall[2]

Kappa   
0.923542

> cm\_svm\_s$byClass

Sensitivity Specificity Pos Pred Value   
 0.9680851 0.9661836 0.9285714   
 Neg Pred Value Precision Recall   
 0.9852217 0.9285714 0.9680851   
 F1 Prevalence Detection Rate   
 0.9479167 0.3122924 0.3023256   
Detection Prevalence Balanced Accuracy   
 0.3255814 0.9671343

> print("RF")

[1] "RF"

> cm\_rf\_s$overall[1]

Accuracy   
0.9767442

> cm\_rf\_s$overall[2]

Kappa   
0.9469363

> cm\_rf\_s$byClass

Sensitivity Specificity Pos Pred Value   
 1.0000000 0.9661836 0.9306931   
 Neg Pred Value Precision Recall   
 1.0000000 0.9306931 1.0000000   
 F1 Prevalence Detection Rate   
 0.9641026 0.3122924 0.3122924   
Detection Prevalence Balanced Accuracy   
 0.3355482 0.9830918

> print("ANN")

[1] "ANN"

> cm\_ann\_s$overall[1]

Accuracy   
0.9833887

> cm\_ann\_s$overall[2]

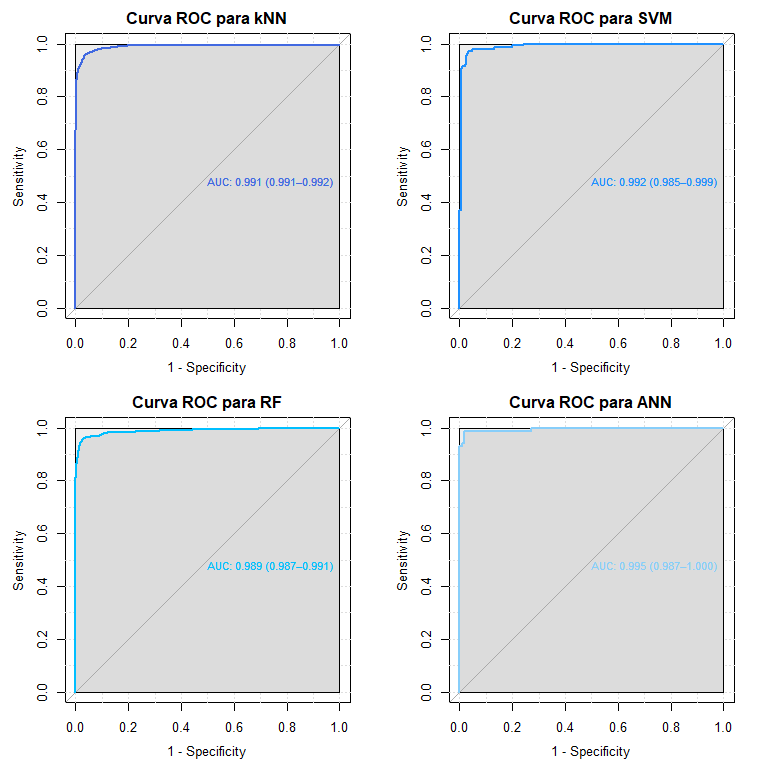
Kappa   
0.9616609

> cm\_ann\_s$byClass

Sensitivity Specificity Pos Pred Value   
 0.9893617 0.9806763 0.9587629   
 Neg Pred Value Precision Recall   
 0.9950980 0.9587629 0.9893617   
 F1 Prevalence Detection Rate   
 0.9738220 0.3122924 0.3089701   
Detection Prevalence Balanced Accuracy   
 0.3222591 0.9850190

# CURVAS ROC

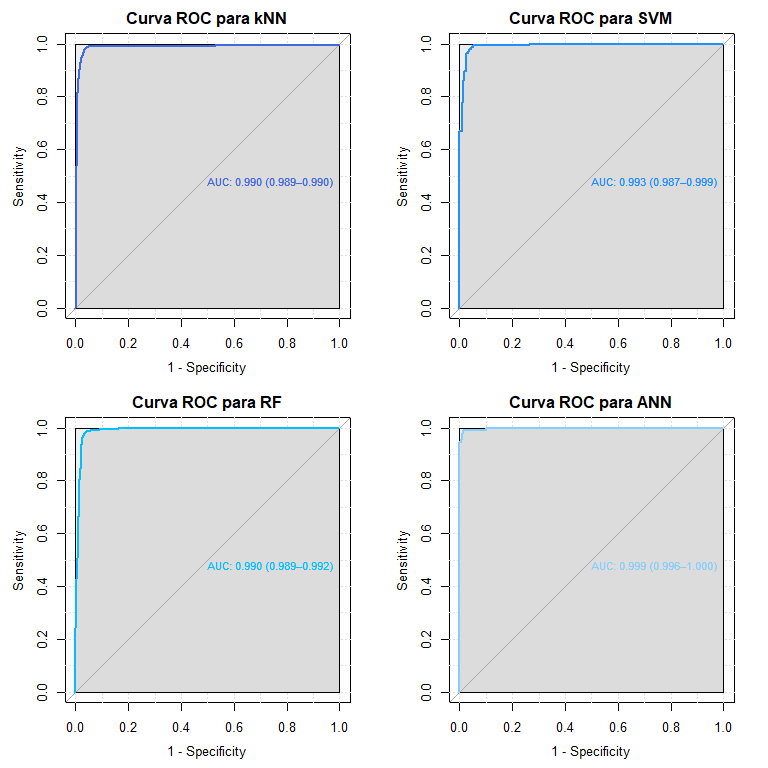
> library(pROC)  
>   
> # "royalblue","dodgerblue", "deepskyblue", "lightskyblue"  
>   
> par(mfrow=c(2,2))  
>   
> plot.roc(model\_knn$pred$obs,  
+ model\_knn$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,   
+ col="royalblue", main = "Curva ROC para kNN")  
>   
> plot.roc(model\_svm$pred$obs,  
+ model\_svm$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="dodgerblue", main = "Curva ROC para SVM")  
>   
> plot.roc(model\_rf$pred$obs,  
+ model\_rf$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="deepskyblue", main = "Curva ROC para RF")  
>   
> plot.roc(test\_ann$diagnosis,   
+ prediction\_ann\_prob$Malignant,   
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="lightskyblue", main = "Curva ROC para ANN")



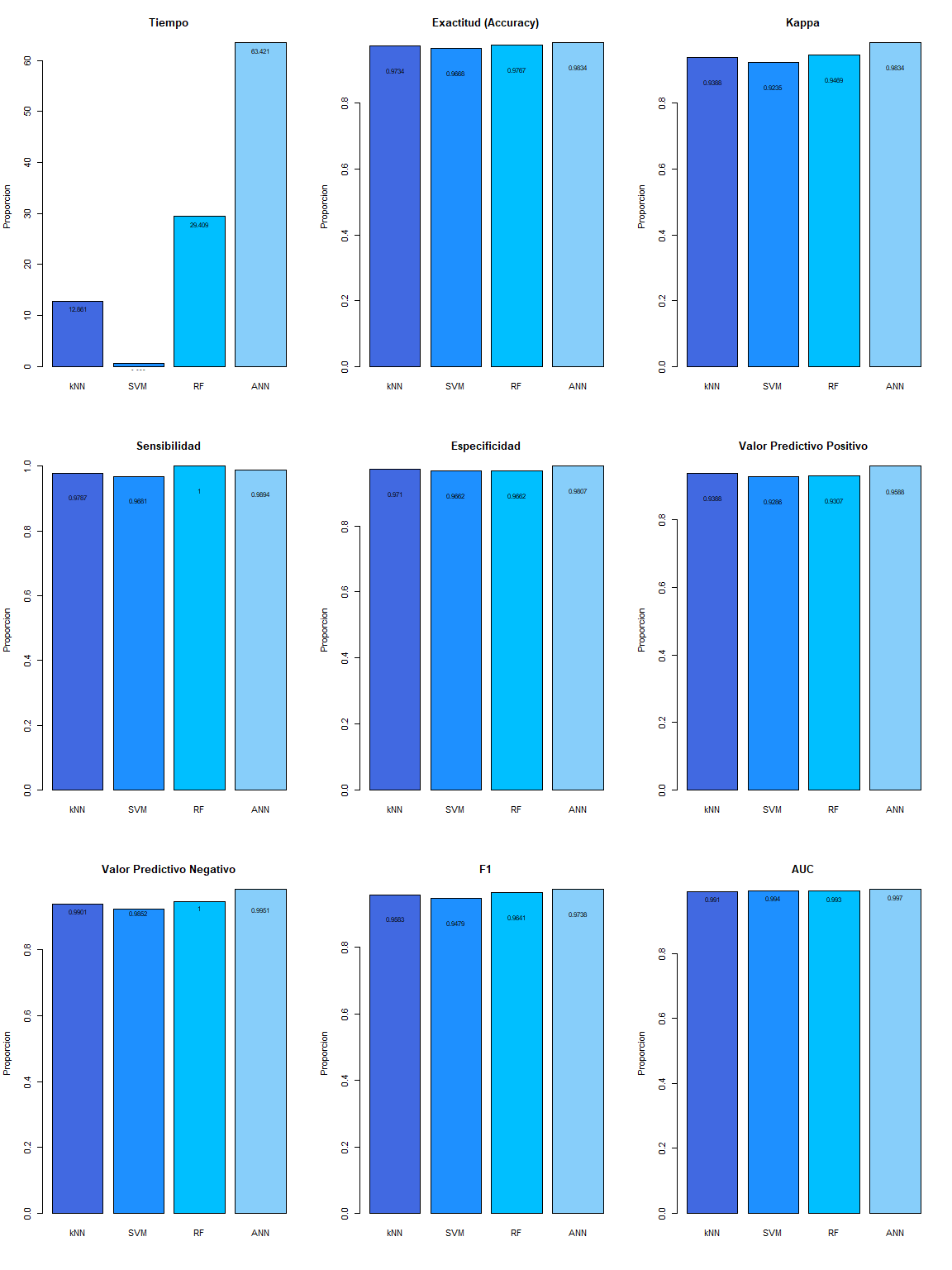
> # https://stackoverflow.com/questions/46891681/calculating-auc-from-nnet-model   
> # https://www.r-bloggers.com/2016/11/calculating-auc-the-area-under-a-roc-curve/   
>   
> #plot.roc(test\_ann$diagnosis,   
> # prediction\_ann\_prob$Malignant,   
> # smoothed = TRUE,  
> # legacy.axes=TRUE,  
> # ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
> # auc.polygon=TRUE,   
> # max.auc.polygon=TRUE, grid=TRUE,  
> # print.auc=TRUE, show.thres=TRUE,  
> # col="blue", main = "Curva ROC para RF")

# Small dataset

> library(pROC)  
>   
> # "royalblue","dodgerblue", "deepskyblue", "lightskyblue"  
>   
> par(mfrow=c(2,2))  
>   
> plot.roc(model\_knn\_s$pred$obs,  
+ model\_knn\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,   
+ col="royalblue", main = "Curva ROC para kNN")  
>   
> plot.roc(model\_svm\_s$pred$obs,  
+ model\_svm\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="dodgerblue", main = "Curva ROC para SVM")  
>   
> plot.roc(model\_rf\_s$pred$obs,  
+ model\_rf\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="deepskyblue", main = "Curva ROC para RF")  
>   
> plot.roc(test\_s\_ann$diagnosis,   
+ prediction\_ann\_s\_prob$Malignant,   
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="lightskyblue", main = "Curva ROC para ANN")



> #cm\_knn  
> #cm\_svm  
> #cm\_rf  
> #cm\_ann  
>   
> par(mfrow=c(3,3), pty="m")  
>   
> # 12345678901234567890123456789012345678901234567890123456789012345678901234567890  
>   
> color <- c("royalblue","dodgerblue", "deepskyblue", "lightskyblue")  
>   
> Acc <- c(cm\_knn\_s$overall[1], cm\_svm\_s$overall[1],   
+ cm\_rf\_s$overall[1], cm\_ann\_s$overall[1])  
>   
> Kappa <- c(cm\_knn\_s$overall[2], cm\_svm\_s$overall[2],   
+ cm\_rf\_s$overall[2], cm\_ann\_s$overall[1])  
>   
> Sensib <- c(cm\_knn\_s$byClass[1], cm\_svm\_s$byClass[1],   
+ cm\_rf\_s$byClass[1], cm\_ann\_s$byClass[1])  
>   
> Espec <- c(cm\_knn\_s$byClass[2], cm\_svm\_s$byClass[2],   
+ cm\_rf\_s$byClass[2], cm\_ann\_s$byClass[2])  
>   
> PPV <- c(cm\_knn\_s$byClass[3], cm\_svm\_s$byClass[3],   
+ cm\_rf\_s$byClass[3], cm\_ann\_s$byClass[3])  
>   
> NPV <- c(cm\_knn\_s$byClass[4], cm\_svm\_s$byClass[4],   
+ cm\_rf\_s$byClass[4], cm\_ann\_s$byClass[4])  
>   
> F1 <- c(cm\_knn\_s$byClass[7], cm\_svm\_s$byClass[7],   
+ cm\_rf\_s$byClass[7], cm\_ann\_s$byClass[7])  
>   
> AUC <- c(0.991, 0.994, 0.993, 0.997)  
>   
> times <- c(knn\_time\_s, svm\_time\_s, rf\_time\_s, ann\_time\_s)  
>   
> time\_p <- barplot(as.vector(times), names.arg=c("kNN", "SVM", "RF", "ANN"),   
+ col = color, ylab=c("Proporcion"), main="Tiempo")  
> text(x = time\_p, y = times, label = round(times,3), pos = 1, cex = 0.8, col =   
+ "black")  
>   
> Acc\_p <- barplot(Acc, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Exactitud (Accuracy)")  
> text(x = Acc\_p, y = Acc-0.05, label = round(Acc,4), pos = 1, cex = 0.8, col =   
+ "black")  
>   
> K\_p <- barplot(Kappa, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Kappa")  
> text(x = K\_p, y = Kappa-0.05, label = round(Kappa,4), pos = 1, cex = 0.8, col =   
+ "black")  
>   
> Sens\_p <- barplot(Sensib, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Sensibilidad")  
> text(x = Sens\_p, y = Sensib-0.05, label = round(Sensib,4), pos = 1, cex = 0.8, col =   
+ "black")  
>   
> Espe\_p <- barplot(Espec, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Especificidad")  
> text(x = Espe\_p, y = Espec-0.05, label = round(Espec,4), pos = 1, cex = 0.8, col =   
+ "black")  
>   
> PPV\_p <- barplot(PPV, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Valor Predictivo Positivo")  
> text(x = PPV\_p, y = PPV-0.05, label = round(PPV,4), pos = 1, cex = 0.8, col = "black")  
>   
> NPV\_p <- barplot(Kappa, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="Valor Predictivo Negativo")  
> text(x = NPV\_p, y = NPV-0.05, label = round(NPV,4), pos = 1, cex = 0.8, col = "black")  
>   
> F1\_p <- barplot(F1, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="F1")  
> text(x = F1\_p, y = F1-0.05, label = round(F1,4), pos = 1, cex = 0.8, col = "black")  
>   
> AUC\_p <- barplot(AUC, names.arg=c("kNN", "SVM", "RF", "ANN"), col = color,  
+ ylab=c("Proporcion"), main="AUC")  
> text(x = AUC\_p, y = AUC, label = round(AUC,4), pos = 1, cex = 0.8, col = "black")



> print(sessionInfo())

R version 3.6.3 (2020-02-29)  
Platform: x86\_64-w64-mingw32/x64 (64-bit)  
Running under: Windows 10 x64 (build 19043)  
  
Matrix products: default  
  
locale:  
[1] LC\_COLLATE=English\_United States.1252   
[2] LC\_CTYPE=English\_United States.1252   
[3] LC\_MONETARY=English\_United States.1252  
[4] LC\_NUMERIC=C   
[5] LC\_TIME=C   
  
attached base packages:  
[1] stats graphics grDevices utils datasets methods base   
  
other attached packages:  
[1] pROC\_1.17.0.1 NeuralNetTools\_1.5.2 nnet\_7.3-12   
[4] caret\_6.0-86 ggplot2\_3.3.3 lattice\_0.20-38   
[7] corrplot\_0.84 knitr\_1.31   
  
loaded via a namespace (and not attached):  
 [1] Rcpp\_1.0.6 lubridate\_1.7.10 tidyr\_1.1.3   
 [4] class\_7.3-15 assertthat\_0.2.1 digest\_0.6.27   
 [7] ipred\_0.9-11 foreach\_1.5.1 utf8\_1.2.1   
[10] R6\_2.5.0 plyr\_1.8.6 stats4\_3.6.3   
[13] evaluate\_0.14 e1071\_1.7-7 pillar\_1.5.1   
[16] rlang\_0.4.10 data.table\_1.14.0 kernlab\_0.9-29   
[19] rpart\_4.1-15 Matrix\_1.2-18 rmarkdown\_2.7   
[22] splines\_3.6.3 gower\_0.2.2 stringr\_1.4.0   
[25] munsell\_0.5.0 proxy\_0.4-26 compiler\_3.6.3   
[28] xfun\_0.22 pkgconfig\_2.0.3 htmltools\_0.5.1.1   
[31] tidyselect\_1.1.0 tibble\_3.1.0 prodlim\_2019.11.13   
[34] codetools\_0.2-16 randomForest\_4.6-14 fansi\_0.4.2   
[37] crayon\_1.4.1 dplyr\_1.0.5 withr\_2.4.1   
[40] MASS\_7.3-51.5 recipes\_0.1.15 ModelMetrics\_1.2.2.2  
[43] grid\_3.6.3 nlme\_3.1-144 gtable\_0.3.0   
[46] lifecycle\_1.0.0 DBI\_1.1.1 magrittr\_2.0.1   
[49] scales\_1.1.1 stringi\_1.5.3 farver\_2.1.0   
[52] reshape2\_1.4.4 timeDate\_3043.102 ellipsis\_0.3.1   
[55] generics\_0.1.0 vctrs\_0.3.7 lava\_1.6.9   
[58] iterators\_1.0.13 tools\_3.6.3 glue\_1.4.2   
[61] purrr\_0.3.4 survival\_3.1-8 yaml\_2.2.1   
[64] colorspace\_2.0-0

# Congreso

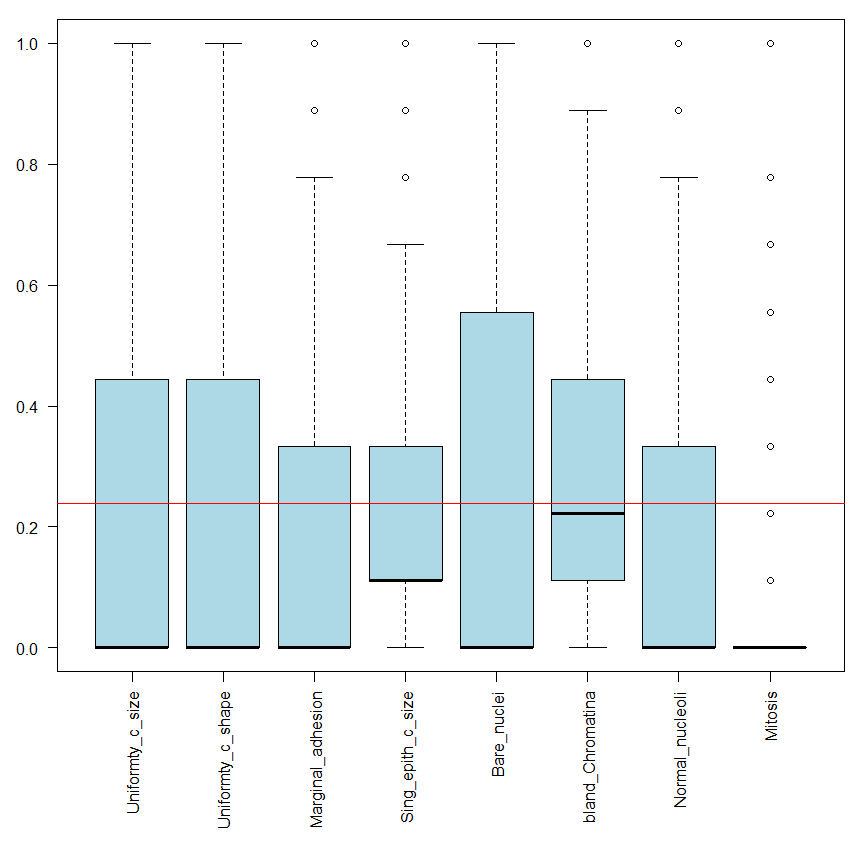
> str(bc\_s)

'data.frame': 682 obs. of 10 variables:  
 $ Clump\_thickness : num 5 3 6 4 8 1 2 2 4 1 ...  
 $ Uniformty\_c\_size : num 4 1 8 1 10 1 1 1 2 1 ...  
 $ Uniformty\_c\_shape: num 4 1 8 1 10 1 2 1 1 1 ...  
 $ Marginal\_adhesion: num 5 1 1 3 8 1 1 1 1 1 ...  
 $ Sing\_epith\_c\_size: num 7 2 3 2 7 2 2 2 2 1 ...  
 $ Bare\_nuclei : num 10 2 4 1 10 10 1 1 1 1 ...  
 $ bland\_Chromatina : num 3 3 3 3 9 3 3 1 2 3 ...  
 $ Normal\_nucleoli : num 2 1 7 1 7 1 1 1 1 1 ...  
 $ Mitosis : num 1 1 1 1 1 1 1 5 1 1 ...  
 $ Class : Factor w/ 2 levels "Benign","Malignant": 1 1 1 1 2 1 1 1 1 1 ...  
 - attr(\*, "na.action")= 'omit' Named int 23 40 139 145 158 164 235 249 275 292 ...  
 ..- attr(\*, "names")= chr "23" "40" "139" "145" ...

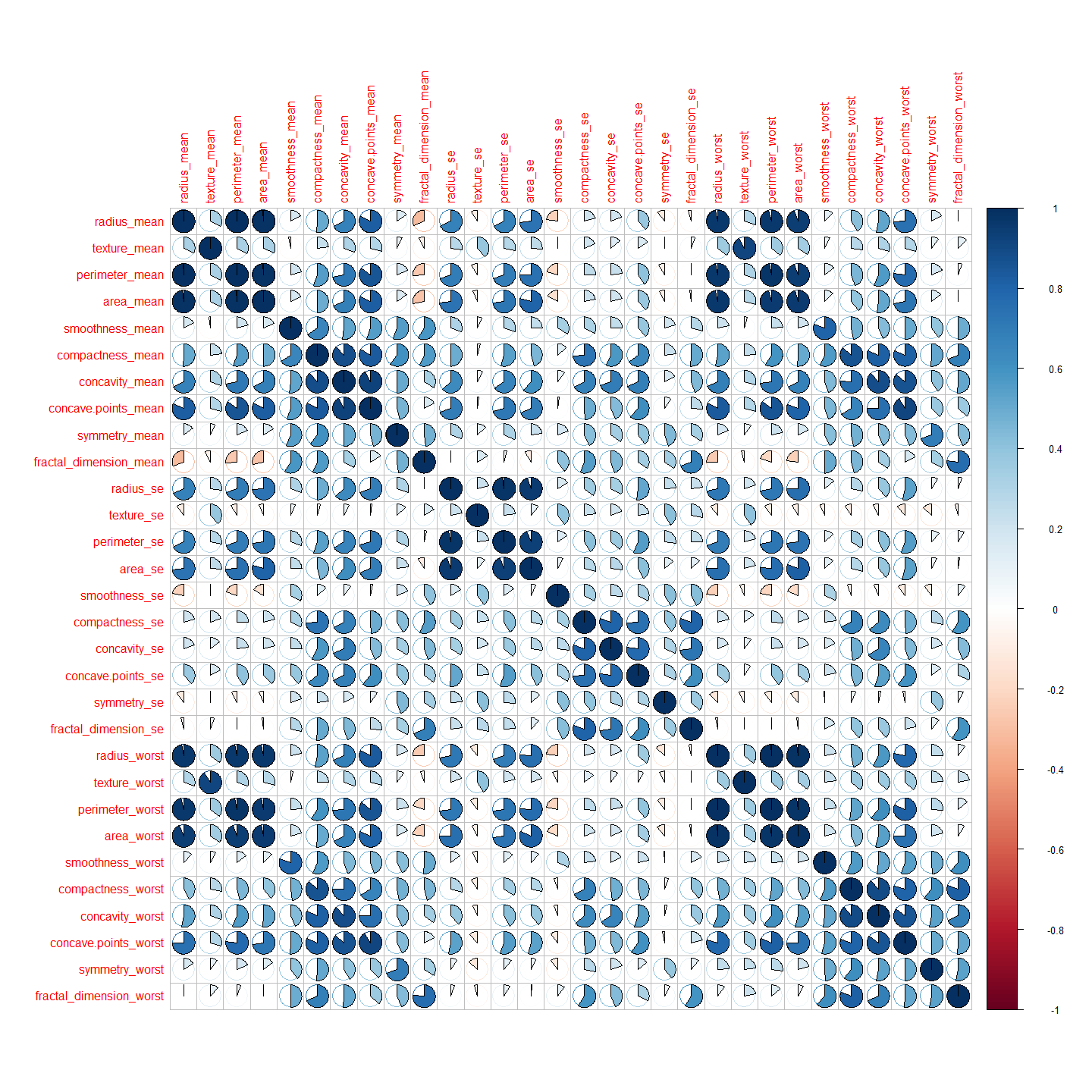
> head(bc\_s, 10)

Clump\_thickness Uniformty\_c\_size Uniformty\_c\_shape Marginal\_adhesion  
1 5 4 4 5  
2 3 1 1 1  
3 6 8 8 1  
4 4 1 1 3  
5 8 10 10 8  
6 1 1 1 1  
7 2 1 2 1  
8 2 1 1 1  
9 4 2 1 1  
10 1 1 1 1  
 Sing\_epith\_c\_size Bare\_nuclei bland\_Chromatina Normal\_nucleoli Mitosis  
1 7 10 3 2 1  
2 2 2 3 1 1  
3 3 4 3 7 1  
4 2 1 3 1 1  
5 7 10 9 7 1  
6 2 10 3 1 1  
7 2 1 3 1 1  
8 2 1 1 1 5  
9 2 1 2 1 1  
10 1 1 3 1 1  
 Class  
1 Benign  
2 Benign  
3 Benign  
4 Benign  
5 Malignant  
6 Benign  
7 Benign  
8 Benign  
9 Benign  
10 Benign

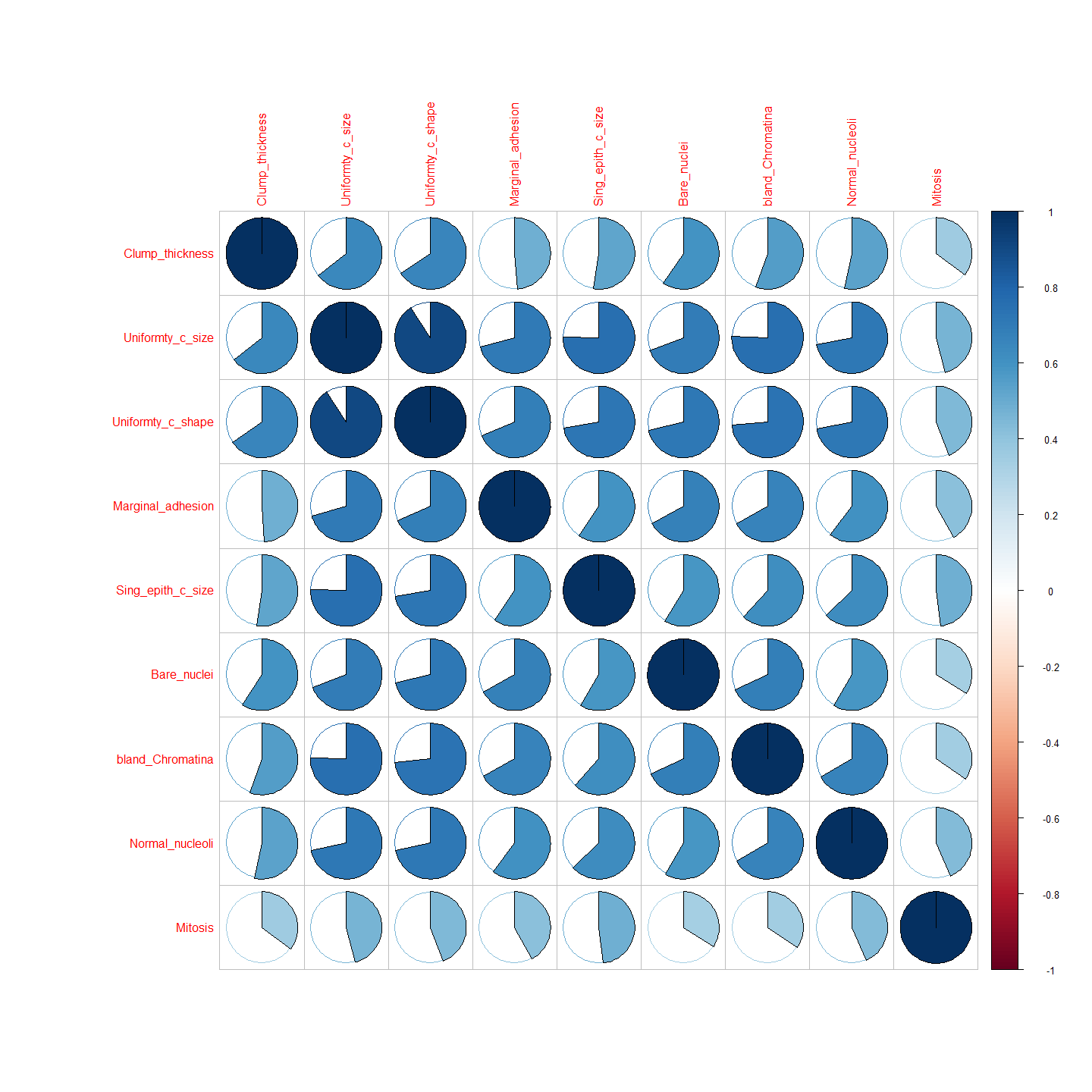
> par(mar=c(10,3,1,1))  
>   
> means\_n\_s <- as.numeric((lapply(bc\_s\_n, mean)))  
>   
> boxplot(bc\_s\_n[,-1], las=2, col="lightblue")  
> # , main="Figura 2. Distribucion de las variables del dataset normalizadas"  
> abline(h=mean(means\_n\_s), col="red")



> library(corrplot)  
> par(mar=c(11,3,1,1))  
>   
> C <- cor(round(as.matrix(bc\_n[1:30]), 4))  
>   
> corrplot(C, method = "pie")



> C2 <- cor(round(as.matrix(bc\_s\_n[1:9]),4))  
>   
> corrplot(C2, method = "pie")



> (cm\_knn\_s)

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 201 2  
 Malignant 6 92  
   
 Accuracy : 0.9734   
 95% CI : (0.9483, 0.9885)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9388   
   
 Mcnemar's Test P-Value : 0.2888   
   
 Sensitivity : 0.9787   
 Specificity : 0.9710   
 Pos Pred Value : 0.9388   
 Neg Pred Value : 0.9901   
 Prevalence : 0.3123   
 Detection Rate : 0.3056   
 Detection Prevalence : 0.3256   
 Balanced Accuracy : 0.9749   
   
 'Positive' Class : Malignant

> (cm\_svm\_s)

Confusion Matrix and Statistics  
  
   
prediction Benign Malignant  
 Benign 200 3  
 Malignant 7 91  
   
 Accuracy : 0.9668   
 95% CI : (0.9398, 0.984)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : <2e-16   
   
 Kappa : 0.9235   
   
 Mcnemar's Test P-Value : 0.3428   
   
 Sensitivity : 0.9681   
 Specificity : 0.9662   
 Pos Pred Value : 0.9286   
 Neg Pred Value : 0.9852   
 Prevalence : 0.3123   
 Detection Rate : 0.3023   
 Detection Prevalence : 0.3256   
 Balanced Accuracy : 0.9671   
   
 'Positive' Class : Malignant

> (cm\_rf\_s)

Confusion Matrix and Statistics  
  
 Reference  
Prediction Benign Malignant  
 Benign 200 0  
 Malignant 7 94  
   
 Accuracy : 0.9767   
 95% CI : (0.9527, 0.9906)  
 No Information Rate : 0.6877   
 P-Value [Acc > NIR] : < 2e-16   
   
 Kappa : 0.9469   
   
 Mcnemar's Test P-Value : 0.02334   
   
 Sensitivity : 1.0000   
 Specificity : 0.9662   
 Pos Pred Value : 0.9307   
 Neg Pred Value : 1.0000   
 Prevalence : 0.3123   
 Detection Rate : 0.3123   
 Detection Prevalence : 0.3355   
 Balanced Accuracy : 0.9831   
   
 'Positive' Class : Malignant

> print("KNN")

[1] "KNN"

> cm\_knn\_s$overall[1]

Accuracy   
0.9734219

> cm\_knn\_s$overall[2]

Kappa   
0.9388336

> cm\_knn\_s$byClass

Sensitivity Specificity Pos Pred Value   
 0.9787234 0.9710145 0.9387755   
 Neg Pred Value Precision Recall   
 0.9901478 0.9387755 0.9787234   
 F1 Prevalence Detection Rate   
 0.9583333 0.3122924 0.3056478   
Detection Prevalence Balanced Accuracy   
 0.3255814 0.9748689

> print("SVM")

[1] "SVM"

> cm\_svm\_s$overall[1]

Accuracy   
0.9667774

> cm\_svm\_s$overall[2]

Kappa   
0.923542

> cm\_svm\_s$byClass

Sensitivity Specificity Pos Pred Value   
 0.9680851 0.9661836 0.9285714   
 Neg Pred Value Precision Recall   
 0.9852217 0.9285714 0.9680851   
 F1 Prevalence Detection Rate   
 0.9479167 0.3122924 0.3023256   
Detection Prevalence Balanced Accuracy   
 0.3255814 0.9671343

> print("RF")

[1] "RF"

> cm\_rf\_s$overall[1]

Accuracy   
0.9767442

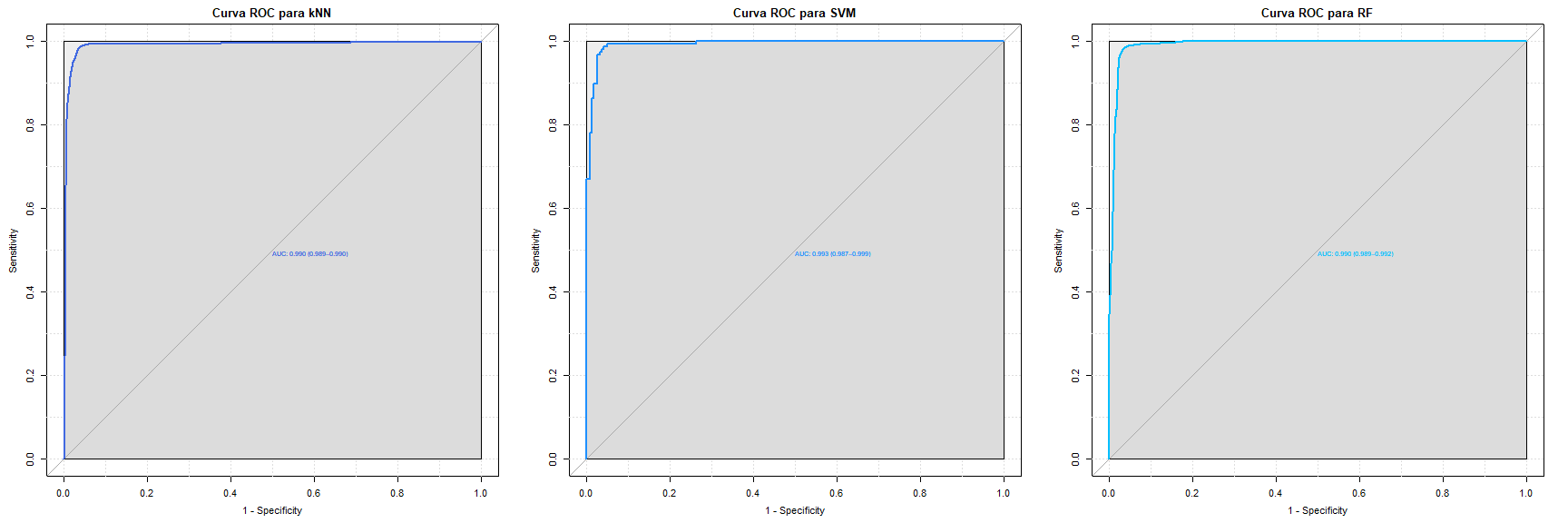
> cm\_rf\_s$overall[2]

Kappa   
0.9469363

> cm\_rf\_s$byClass

Sensitivity Specificity Pos Pred Value   
 1.0000000 0.9661836 0.9306931   
 Neg Pred Value Precision Recall   
 1.0000000 0.9306931 1.0000000   
 F1 Prevalence Detection Rate   
 0.9641026 0.3122924 0.3122924   
Detection Prevalence Balanced Accuracy   
 0.3355482 0.9830918

> library(pROC)  
>   
> # "royalblue","dodgerblue", "deepskyblue", "lightskyblue"  
>   
> par(mfrow=c(1,3), pty="m")  
>   
> plot.roc(model\_knn\_s$pred$obs,  
+ model\_knn\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,   
+ col="royalblue", main = "Curva ROC para kNN")  
>   
> plot.roc(model\_svm\_s$pred$obs,  
+ model\_svm\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="dodgerblue", main = "Curva ROC para SVM")  
>   
> plot.roc(model\_rf\_s$pred$obs,  
+ model\_rf\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="deepskyblue", main = "Curva ROC para RF")



> par(mfrow=c(4,3), pty="m")  
>   
> #Acc <- c(max(knn\_fit\_n$results$Accuracy),   
> # max(model$results$Accuracy),   
> # max(rf\_caret\_model$results$Accuracy))  
>   
> #Kappa <- c(max(knn\_fit\_n$results$Kappa),  
> # max(model$results$Kappa),   
> # max(rf\_caret\_model$results$Kappa))  
>   
> color <- c("royalblue", "dodgerblue", "deepskyblue")  
>   
> Acc <- c(cm\_knn\_s$overall[1], cm\_svm\_s$overall[1], cm\_rf\_s$overall[1])  
>   
> Kappa <- c(cm\_knn\_s$overall[2], cm\_svm\_s$overall[2], cm\_rf\_s$overall[2])  
>   
> Sensib <- c(cm\_knn\_s$byClass[1], cm\_svm\_s$byClass[1], cm\_rf\_s$byClass[1])  
>   
> Espec <- c(cm\_knn\_s$byClass[2], cm\_svm\_s$byClass[2], cm\_rf\_s$byClass[2])  
>   
> PPV <- c(cm\_knn\_s$byClass[3], cm\_svm\_s$byClass[3], cm\_rf\_s$byClass[3])  
>   
> NPV <- c(cm\_knn\_s$byClass[4], cm\_svm\_s$byClass[4], cm\_rf\_s$byClass[4])  
>   
> F1 <- c(cm\_knn\_s$byClass[7], cm\_svm\_s$byClass[7], cm\_rf\_s$byClass[7])  
>   
> AUC <- c(0.989, 0.996, 0.992)  
>   
> times\_3 <- times[1:3]  
>   
> time\_p <- barplot(as.vector(times\_3), names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Time")  
> text(x = time\_p, y = times, label = round(times,3), pos = 1, cex = 0.8, col = "black")  
>   
> Acc\_p <- barplot(Acc, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Exactitud (Accuracy)")  
> text(x = Acc\_p, y = Acc-0.05, label = round(Acc,4), pos = 1, cex = 0.8, col = "black")  
>   
> K\_p <- barplot(Kappa, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Kappa")  
> text(x = K\_p, y = Kappa-0.05, label = round(Kappa,4), pos = 1, cex = 0.8, col = "black")  
>   
> Sens\_p <- barplot(Sensib, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Sensititivy")  
> text(x = Sens\_p, y = Sensib-0.05, label = round(Sensib,4), pos = 1, cex = 0.8, col = "black")  
>   
> Espe\_p <- barplot(Espec, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Specificity")  
> text(x = Espe\_p, y = Espec-0.05, label = round(Espec,4), pos = 1, cex = 0.8, col = "black")  
>   
> PPV\_p <- barplot(PPV, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Positive Predictive Value")  
> text(x = PPV\_p, y = PPV-0.05, label = round(PPV,4), pos = 1, cex = 0.8, col = "black")  
>   
> NPV\_p <- barplot(Kappa, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="Negative Predictive Value")  
> text(x = NPV\_p, y = NPV-0.05, label = round(NPV,4), pos = 1, cex = 0.8, col = "black")  
>   
> F1\_p <- barplot(F1, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="F1")  
> text(x = F1\_p, y = F1-0.05, label = round(F1,4), pos = 1, cex = 0.8, col = "black")  
>   
> AUC\_p <- barplot(AUC, names.arg=c("kNN", "SVM", "RF"), col = color,  
+ ylab=c("Proportion"), main="AUC")  
> text(x = AUC\_p, y = AUC, label = round(AUC,4), pos = 1, cex = 0.8, col = "black")  
>   
> plot.roc(model\_knn\_s$pred$obs,  
+ model\_knn\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,   
+ col="royalblue", main = "ROC Curve for kNN")  
>   
> plot.roc(model\_svm\_s$pred$obs,  
+ model\_svm\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="dodgerblue", main = "ROC Curve for SVM")  
>   
> plot.roc(model\_rf\_s$pred$obs,  
+ model\_rf\_s$pred$Malignant,  
+ smoothed = TRUE,  
+ legacy.axes=TRUE,  
+ # arguments for ci  
+ ci=TRUE, ci.alpha=0.9, stratified=FALSE,  
+ # arguments for plot  
+ #plot=TRUE,   
+ auc.polygon=TRUE,   
+ max.auc.polygon=TRUE, grid=TRUE,  
+ print.auc=TRUE, show.thres=TRUE,  
+ col="deepskyblue", main = "ROC Curve for RF")

