Trabajo Práctico Final

Técnicas y Algoritmos de Aprendizaje Automático

Especialización en Ciencia de Datos

Luciana Gattuso

Ejercicio 1 – Aprendizaje Supervisado

En este ejercicio se pide comparar 3 modelos para predecir qué *fonema* se está diciendo. Primero se realiza un Análisis Exploratorio de los Datos para entender la base, luego se particiona la base en un conjunto de entrenamiento y uno de testeo, después con estos conjuntos se realizan dos modelos de SVM y un modelo de Naive Bayes (con la librería *e1071*). Finalmente se comparan las mediciones de la performance de los modelos.

Parte A – Análisis Exploratorio de los Datos

1)Considere el archivo phoneme.data de la web del libro The Elements of Statistical Learning (Hastie, Tibshirani, Friedman) https://web.stanford.edu/~hastie/ElemStatLearn/ En la página web -> Busque **"Data"** en el menú -> Busque "Phoneme: Info Data" -> en Data se encuentra el archivo *phoneme.data.*
Abra Info y busque cuáles son los 5 fonemas a predecir. Copie los ejemplos de palabras en inglés que el autor usa para cada fonema.

> Los datos que se analizarán a lo largo de este trabajo práctico final fueron extraídos del TIMIT (Corpus acústico-fonético de habla continua, NTIS, Departamento de Comercio de EE. UU.). Los 5 fonemas a predecir son:
>
> - *sh* como "she", del inglés;
> - *dcl* como "dark";
> - *iy* como la vocal de "she";
> - *aa* como la vocal de "dark" y
> - *ao* como la primera vocal de "water".

2) ¿Para qué podría servir predecir qué fonema se está diciendo? (pista: piense en sistemas como el asistente del celular).

> Predecir el fonema que esta diciendo una persona podría servir para predecir una instrucción a un asistente virtual como Siri.
>
> Abra el archivo "phoneme.data" en R de la siguiente manera
> base=read.table("phoneme.data",sep=",",header=TRUE).

Haga un head de la base, pero no lo incluya aquí, solamente vea las variables del dataset.
Borre las variables "speaker" y "row.names" base$speaker=NULL
base$row.names=NULL.
Renombre la variable a predecir "g" como fonemas y luego transfórmela a factor.
names(base)[names(base)=="g"]="fonemas" base$fonemas=factor(base$fonemas)
Muestre un head(base). Muestre aquí algunas de las variables del comienzo y del final del head (un head "truncado").

El dataset cuenta con 256 columnas etiquetadas a partir del x.1 al x.256 y una columna llamada g, que utilizaremos como variable a predecir, con los fonemas.

```
> head(base)
      x.1      x.2      x.3      x.4      x.5      x.6      x.7
1  9.85770  9.20711  9.81689  9.01692  9.05675  8.92518 11.28308
2 13.23079 14.19189 15.34428 18.11737 19.53875 18.32726 17.34169
3 10.81889  9.07615  9.77940 12.20135 12.59005 10.53364  8.54693
      x.8      x.9     x.10     x.11     x.12     x.13     x.14
1 11.52980 10.79713  9.04747 11.38065 11.61545 10.31166 12.14300
2 17.16861 19.63557 20.15212 18.84739 19.35436 19.61419 20.32628
3  9.46049 11.96755 12.05282  8.92333  7.23825  6.98416  7.50434
     x.15     x.16     x.17     x.18     x.19     x.20     x.21
1 10.65254 11.19395 11.27878 11.13465 11.82706 12.81048 12.32239
2 21.41752 20.14227 17.97866 16.32943 17.63076 16.51955 15.95348
3  7.21911  6.10589  5.37987  5.69972  6.15451  8.33365  8.61714
     x.22     x.23     x.24     x.25     x.26     x.27     x.28
1 10.27542  9.57760 10.62308 11.34983 11.58342 12.19465 10.67933
2 16.77282 16.98819 18.32396 17.59234 15.94519 13.31152 15.36856
3  6.58075  5.63212  4.57192  7.18243  7.64013  6.48067  6.51159
     x.29     x.30     x.31     x.32     x.33     x.34     x.35
1  9.95300 11.47047 11.74805 11.28167 11.51648 12.50765 12.10918
2 16.10275 14.76571 14.85925 11.89751 14.90796 15.49864 15.13160
3  4.77806  6.32551  5.61790  5.93619  3.93011  5.45621  5.22023
     x.225    x.226    x.227    x.228    x.229    x.230    x.231
1 13.23555 13.76750 13.53994 12.28767 13.48164 13.43467 11.43232
2 11.27136 11.29874 10.02574 10.11055  7.53453  8.71152  9.58630
3  5.99069  2.37876  5.10594  5.88512  5.65892  6.57863  4.45805
     x.232    x.233    x.234    x.235    x.236    x.237    x.238
1 11.01716 13.02529 13.37422  7.72797 12.32757 13.37041 12.68903
2  9.71517  9.29711  7.74722 10.07586  6.80730  7.99336  8.70087
3  4.86107  5.52299  6.17580  6.20610  6.49488  6.79143  5.87053
     x.239    x.240    x.241    x.242    x.243    x.244    x.245
1 13.41566 13.16755 13.62844 13.32554 12.46720 13.84955 10.83688
2  6.50054  9.11663  4.95653  8.98125  7.78602  8.06256  7.46077
3  6.65905  6.37389  4.16703  5.47719  5.43269  4.10159  3.91116
     x.246    x.247    x.248    x.249    x.250    x.251    x.252
1 11.23451 13.19359 12.94519 12.68076 11.20767 13.69394 13.72055
2  7.03316  7.93572  8.91567  8.45714  8.77266  9.59717  8.45336
3  3.88583  5.83309  6.49345  5.00824  5.51019  5.95725  7.04992
     x.253    x.254    x.255    x.256 fonemas
1 12.16628 12.92489 12.51195 9.75527      sh
2  7.57730  5.38504  9.43063 8.59328      iy
3  7.02469  6.58416  6.27058 3.85042     dcl
```

3) Con table(base$fonemas) indique cuántos fonemas de cada tipo de fonema tiene el dataset.

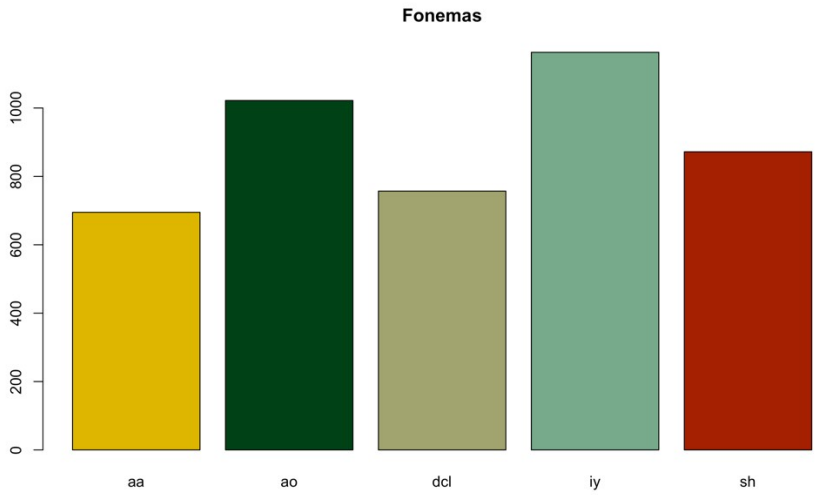
   El dataset tiene cuenta con:

   - 695 registros del fonema *aa*;
   - 1022 de *ao*;
   - 757 de *dcl*;
   - 1163 de *iy* y, - 872 de *sh*.

4) ¿Cuál es el fonema que tiene más elementos en el dataset?

   El fonema con más elementos del dataset es *iy*.

5) Realice un gráfico de barras de la variable a predecir. Elija un color y un título para el gráfico. plot(base$fonemas,col="color",main="Título")



6) Con la instrucción fonema=base[num,] podemos ver los valores del elemento num del dataset.

Considere los últimos 3 números de su DNI y muestre los valores del fonema de la base en esa fila. Fonema=base[3numDNI,] ¿Qué fonema tiene?

Utilizando el número 945, Rstudio devuelve los siguientes valores de la base en esa fila y el fonema *ao*.

```
        x.1     x.2      x.3      x.4      x.5      x.6      x.7
945 14.15126 9.94577 18.14784 16.97855 14.25021 19.35191 19.98799
        x.8      x.9     x.10     x.11     x.12     x.13     x.14
945 17.92598 18.40413 20.51355 19.52211 15.5835 20.71385 20.84173
       x.15     x.16     x.17     x.18     x.19     x.20     x.21
945 17.73291 21.4687 22.92391 21.03872 21.25671 24.53404 24.20224
       x.22     x.23     x.24     x.25     x.26     x.27     x.28
945 20.33126 22.68526 23.35476 20.91341 20.84777 22.9695 21.78424
       x.29     x.30     x.31     x.32     x.33     x.34     x.35
945 17.53015 21.51763 21.52603 15.10694 21.15718 23.29664 22.26498
       x.36     x.37     x.38     x.39     x.40     x.41     x.42
945 18.81369 21.84058 22.05515 19.48275 18.40444 19.001 17.04658
       x.43     x.44     x.45     x.46     x.47     x.48     x.49
945 16.0101 17.60558 15.68662 13.41928 16.6127 16.08884 12.87893
       x.50     x.51     x.52     x.53     x.54     x.55     x.56
945 14.70911 15.49392 13.84778 13.35219 15.53112 14.70515 12.17087
       x.57     x.58     x.59     x.60     x.61     x.62     x.63
945 14.41336 14.06188 11.77925 13.49212 14.65199 13.66895 12.72979
       x.64     x.65     x.66     x.67     x.68     x.69     x.70
945 15.31567 14.84975 11.25388 15.04458 15.34967 12.81471 15.63146
       x.71     x.72     x.73     x.74     x.75     x.76     x.77
945 17.02302 15.40491 11.84721 14.73062 13.42217 10.65343 12.2638
       x.78     x.79     x.80     x.81     x.82  x.83     x.84
945 13.86462 11.98266 12.08584 14.68975 13.42138 6.217 13.26155
       x.85     x.86     x.87     x.88     x.89     x.90     x.91
945 13.11418 7.32355 11.56797 13.77701 11.66337 7.8632 13.30582
```

```
       x.92     x.93     x.94     x.95     x.96     x.97     x.98
945 12.84767 9.27691 14.0393 14.61017 12.27945 12.97576 15.79818
       x.99     x.100    x.101    x.102    x.103    x.104    x.105
945 14.79045 11.62551 14.91891 15.3706 11.75269 14.55258 16.95427
      x.106    x.107    x.108    x.109    x.110    x.111    x.112
945 16.01695 13.62002 16.45277 16.3798 12.73747 15.40108 16.41552
      x.113    x.114    x.115    x.116    x.117    x.118    x.119
945 14.68286 14.04333 16.54526 15.89194 12.8731 14.75026 14.82682
      x.120    x.121    x.122    x.123    x.124    x.125    x.126
945 12.22191 12.14969 14.04238 12.57388 10.08612 11.45955 9.31454
      x.127    x.128    x.129    x.130    x.131    x.132    x.133    x.134
945 9.56333 10.52359 12.12283 10.6184 9.58044 10.80744 6.50962 9.70973
      x.135    x.136    x.137    x.138    x.139    x.140    x.141    x.142
945 9.21033 10.63837 8.58139 9.60904 11.41216 8.48858 8.82579 9.47174
      x.143    x.144    x.145    x.146    x.147    x.148    x.149    x.150
945 9.94772 9.91076 11.36224 12.22679 9.73152 9.59327 11.77451 10.12375
      x.151    x.152    x.153    x.154    x.155  x.156    x.157
945 11.32766 13.76013 14.14647 11.99507 12.35424 14.122 12.9739
      x.158    x.159    x.160    x.161    x.162    x.163    x.164
945 11.14477 10.15319 9.79919 7.32873 13.03745 13.39813 12.64267
      x.165    x.166    x.167    x.168    x.169    x.170    x.171
945 7.20328 12.39422 12.57811 12.73134 13.36419 12.24811 11.86978
      x.172    x.173    x.174    x.175    x.176    x.177    x.178    x.179
945 12.88031 13.5624 12.54246 11.51606 9.24976 7.26836 9.51058 11.50864
      x.180    x.181    x.182    x.183    x.184    x.185    x.186
945 12.02846 10.78334 10.56042 12.04583 10.97968 11.58665 10.64254
```

```
     x.187 x.188    x.189    x.190    x.191    x.192    x.193    x.194
945 9.40472 9.432 10.76544 12.48496 11.10743 10.95703 13.10375 12.02923
     x.195    x.196    x.197    x.198    x.199    x.200    x.201    x.202
945 9.9649 9.60672 9.94987 9.16207 11.07861 11.89464 10.61511 9.64999
      x.203    x.204    x.205    x.206    x.207    x.208    x.209    x.210
945 11.61999 10.71877 7.98601 7.74421 9.43884 8.68538 10.06143 10.71383
      x.211    x.212    x.213    x.214    x.215    x.216    x.217    x.218
945 9.56825 7.93195 10.19897 9.90051 8.53786 7.88787 3.63093 8.97197
      x.219    x.220    x.221    x.222    x.223    x.224    x.225    x.226
945 8.90102 8.30711 8.74632 5.92358 7.18563 7.47615 8.15128 8.86988
      x.227    x.228    x.229    x.230  x.231    x.232    x.233    x.234
945 6.12671 8.55881 6.68718 4.00085 7.4151 8.48025 8.15725 7.60547
     x.235    x.236    x.237    x.238    x.239    x.240    x.241    x.242
945 8.1558 8.38459 6.53047 8.38545 7.45107 7.84105 7.32526 8.82224
      x.243  x.244    x.245    x.246    x.247    x.248    x.249    x.250
945 8.55615 6.0537 8.13869 8.29486 7.25854 8.67484 6.86264 8.05217
      x.251    x.252    x.253    x.254    x.255    x.256 fonemas
945 7.67228 8.48727 8.46554 8.78923 9.43575 4.95243      ao
```

Parte B – Conjuntos

1) Para el seteo de semilla considere los 3 últimos dígitos de su DNI y particione la base en un conjunto de entrenamiento y uno de testeo, utilizando la instrucción createDataPartition de la librería caret.

Además, si su DNI termina en 0, 1, 2 ó 3

Setee p=0.70

Si su DNI termina en 4, 5, 6 ó 7

Setee p=0.75

Si su DNI termina en 8 ó 9

Setee p=0.80

set.seed(3numDNI);particion=createDataPartition(y=base$fonemas,p=num,list= FALSE)

entreno=base[particion,] testeo=base[-particion,] Indique cómo quedó el código R.

set.seed(945);particion=createDataPartition(y=base$fonemas,p=0.75,list= FALSE)
entreno=base[particion,] testeo=base[-particion,]

2) Muestre un head truncado y un summary truncado del conjunto de entrenamiento y del conjunto de testeo.

```
> head(entreno)
      x.1      x.2      x.3      x.4      x.5      x.6      x.7      x.8      x.9     x.10     x.11     x.12     x.13
1  9.85770  9.20711  9.81689  9.01692  9.05675  8.92518 11.28308 11.52980 10.79713  9.04747 11.38065 11.61545 10.31166
2 13.23079 14.19189 15.34428 18.11737 19.53875 18.32726 17.34169 17.16861 19.63557 20.15212 18.84739 19.35436 19.61419
3 10.81889  9.07615  9.77940 12.20135 12.59005 10.53364  8.54693  9.46049 11.96755 12.05282  8.92333  7.23825  6.98416
     x.14     x.15     x.16     x.17     x.18     x.19     x.20     x.21     x.22     x.23     x.24     x.25     x.26
1 12.14300 10.65254 11.19395 11.27878 11.13465 11.82706 12.81048 12.32239 10.27542  9.57760 10.62308 11.34983 11.58342
2 20.32628 21.41752 20.14227 17.97866 16.32943 17.63076 16.51955 15.95348 16.77282 16.98819 18.32396 17.59234 15.94519
3  7.50434  7.21911  6.10589  5.37987  5.69972  6.15451  8.33365  8.61714  6.58075  5.63212  4.57192  7.18243  7.64013
     x.235    x.236    x.237    x.238    x.239    x.240    x.241    x.242    x.243    x.244    x.245    x.246    x.247
1  7.72797 12.32757 13.37041 12.68903 13.41566 13.16755 13.62844 13.32554 12.46720 13.84955 10.83688 11.23451 13.19359
2 10.07586  6.80730  7.99336  8.70087  6.50054  9.11663  4.95653  8.98125  7.78602  8.06256  7.46077  7.03316  7.93572
3  6.20610  6.49488  6.79143  5.87053  6.65905  6.37389  4.16703  5.47719  5.43269  4.10159  3.91116  3.88583  5.83309
     x.248    x.249    x.250    x.251    x.252    x.253    x.254    x.255   x.256 fonemas
1 12.94519 12.68076 11.20767 13.69394 13.72055 12.16628 12.92489 12.51195 9.75527      sh
2  8.91567  8.45714  8.77266  9.59717  8.45336  7.57730  5.38504  9.43063 8.59328      iy
3  6.49345  5.00824  5.51019  5.95725  7.04992  7.02469  6.58416  6.27058 3.85042     dcl
```

```
> summary(entreno)
      x.1              x.2              x.3               x.4              x.5              x.6               x.7
 Min.   : 1.764   Min.   : 1.794   Min.   : 0.00562   Min.   : 2.552   Min.   : 4.262   Min.   : 0.8793   Min.   : 4.148
 1st Qu.: 9.636   1st Qu.:10.115   1st Qu.:12.69016   1st Qu.:13.513   1st Qu.:12.848   1st Qu.:12.5157   1st Qu.:13.279
 Median :10.868   Median :11.852   Median :15.51382   Median :16.657   Median :15.211   Median :14.9610   Median :16.381
      x.8              x.9              x.10             x.11             x.12             x.13              x.14
 Min.   : 4.562   Min.   : 4.80   Min.   : 2.731   Min.   : 3.691   Min.   : 2.961   Min.   :-0.6702   Min.   : 3.342
 1st Qu.:13.626   1st Qu.:13.43   1st Qu.:13.063   1st Qu.:13.009   1st Qu.:13.003   1st Qu.:13.0435   1st Qu.:13.004
 Median :16.775   Median :16.29   Median :16.261   Median :16.420   Median :16.438   Median :16.3222   Median :16.293
     x.249            x.250            x.251            x.252            x.253            x.254             x.255
 Min.   :-4.432   Min.   :-1.836   Min.   :-2.738   Min.   :-1.325   Min.   :-0.6303   Min.   :-0.0862   Min.   :-0.6502
 1st Qu.: 6.901   1st Qu.: 6.858   1st Qu.: 6.869   1st Qu.: 6.855   1st Qu.: 6.8129   1st Qu.: 6.8440   1st Qu.: 6.8350
 Median : 8.900   Median : 8.961   Median : 8.925   Median : 8.923   Median : 8.8549   Median : 8.9012   Median : 8.9112
     x.256         fonemas
 Min.   :-8.172   aa :522
 1st Qu.: 6.202   ao :767
 Median : 8.414   dcl:568
```

```
> head(testeo)
       x.1      x.2      x.3      x.4      x.5      x.6      x.7      x.8      x.9     x.10     x.11     x.12     x.13
4 10.53679  9.12147 10.84621 13.92331 13.52476 10.27831  8.97459 11.57109 12.35839 10.47826  8.56676  8.97603 10.67954
5 12.96705 13.69454 14.91182 18.22292 18.45390 17.25760 17.79614 17.76387 18.99632 17.40394 18.08701 17.78545 18.58141
7 10.95324 11.20585 16.17634 18.59300 17.50922 10.27798 16.00151 19.32894 18.84554 13.37857 14.16436 18.85797 19.03694
      x.14     x.15     x.16     x.17     x.18     x.19     x.20     x.21     x.22     x.23     x.24     x.25     x.26
4  9.67530  8.83348  8.43713  5.42489  6.46217  7.50640  3.71255  5.43962  7.34509  7.94968  7.08204  3.99006  4.51822
5 18.26168 18.72958 19.56382 18.81372 19.77800 18.92736 21.83878 21.83288 20.81927 21.14635 17.96529 21.40177 20.30099
7 14.76523 14.04992 19.83137 20.70606 17.13739 16.08513 20.46493 21.94998 19.92326 16.49355 18.59621 20.49528 19.36451
     x.234    x.235    x.236    x.237    x.238    x.239    x.240    x.241    x.242    x.243    x.244    x.245    x.246    x.247
4  6.55074 5.15614 4.18213 4.95448  5.61000  5.15144 5.44564 4.74798 5.55902 2.88668 4.56161 5.62022 5.51356  5.95977
5  9.71158 9.85747 8.77021 9.17960 10.49989  9.99960 8.33319 9.29810 4.60656 7.58772 7.82380 6.96809 6.97810  8.91020
7 11.71408 8.11747 7.18819 8.85977 10.66308 10.25736 9.00297 8.98193 8.84182 8.54476 8.20910 7.14580 9.24845 10.29726
    x.248    x.249    x.250    x.251    x.252    x.253    x.254    x.255    x.256 fonemas
4 5.36506 5.85688 5.40324 6.07126 5.30651 4.27412 3.63384 3.22823 4.63123     dcl
5 7.74256 8.00151 7.58624 6.65202 7.69109 6.93683 7.03600 7.01278 8.52197      aa
7 7.63297 8.76707 6.94130 8.79901 8.22345 7.63610 8.44448 8.28905 8.04018      aa
```

```
> summary(testeo)
      x.1              x.2              x.3              x.4              x.5              x.6              x.7
 Min.   : 4.642   Min.   : 4.058   Min.   : 5.703   Min.   : 5.368   Min.   : 2.965   Min.   : 4.543   Min.   : 4.884
 1st Qu.: 9.598   1st Qu.:10.030   1st Qu.:12.534   1st Qu.:13.672   1st Qu.:13.005   1st Qu.:12.409   1st Qu.:13.288
 Median :10.692   Median :11.707   Median :15.355   Median :16.587   Median :15.315   Median :14.900   Median :16.384
      x.8              x.9              x.10             x.11             x.12             x.13             x.14
 Min.   : 4.908   Min.   : 4.981   Min.   : 6.791   Min.   : 5.24   Min.   : 0.943   Min.   : 3.475   Min.   : 3.341
 1st Qu.:13.618   1st Qu.:13.426   1st Qu.:13.041   1st Qu.:13.14   1st Qu.:13.182   1st Qu.:13.002   1st Qu.:12.791
 Median :16.721   Median :16.308   Median :16.062   Median :16.58   Median :16.587   Median :16.164   Median :16.161
     x.247            x.248            x.249            x.250            x.251            x.252            x.253
 Min.   :-1.334   Min.   : 0.7379   Min.   : 0.7329   Min.   : 1.462   Min.   :-2.481   Min.   : 1.076   Min.   : 1.187
 1st Qu.: 6.839   1st Qu.: 6.7461   1st Qu.: 6.9475   1st Qu.: 6.886   1st Qu.: 6.779   1st Qu.: 6.729   1st Qu.: 6.678
 Median : 8.771   Median : 8.8496   Median : 8.9875   Median : 8.897   Median : 8.816   Median : 8.884   Median : 8.905
     x.254            x.255            x.256         fonemas
 Min.   :-0.7841   Min.   : 0.922   Min.   :-6.263   aa :173
 1st Qu.: 6.7196   1st Qu.: 6.887   1st Qu.: 6.246   ao :255
 Median : 8.7829   Median : 8.856   Median : 8.387   dcl:189
```

3) En R ingrese table(base$fonemas); table(entreno$fonemas); table(testeo$fonemas) y haga una tabla comparando por fonema cuántos quedaron de cada uno en la base, en entrenamiento y en testeo (la tabla no es necesario hacerla en R sino tipo Word).

Conjuntos/Fonemas *aa*	*ao*	*dcl*	*iy*	*sh*
Base	695	1022	757	1163	872
Entrenamiento	522	767	568	873	654
Testeo	173	255	189	290	218

Optativo: Fijarse si el registro Fonema de los 3 últimos números de su DNI quedó en entrenamiento o en testeo.

El registro Fonema quedó dentro del conjunto de entrenamiento.

Parte C – SVM1

1) Cargue la librería e1071 y modele el problema planteado con una Support Vector Machine con un kernel = (asignado) y los parámetros restantes con los valores por defecto sin cambiar.
   Indique cómo quedó el código R utilizado.
   svmAsignado=svm(fonemas~.,entreno,kernel="sigmoid") svmAsignado

2) Escriba svm<enter> y muestre una captura de pantalla de la información que aparece. ¿Cuántos vectores soporte aparecen?

```
> svmAsignado

Call:
svm(formula = fonemas ~ ., data = entreno, kernel = "sigmoid")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
     coef.0:  0

Number of Support Vectors:  1021
```

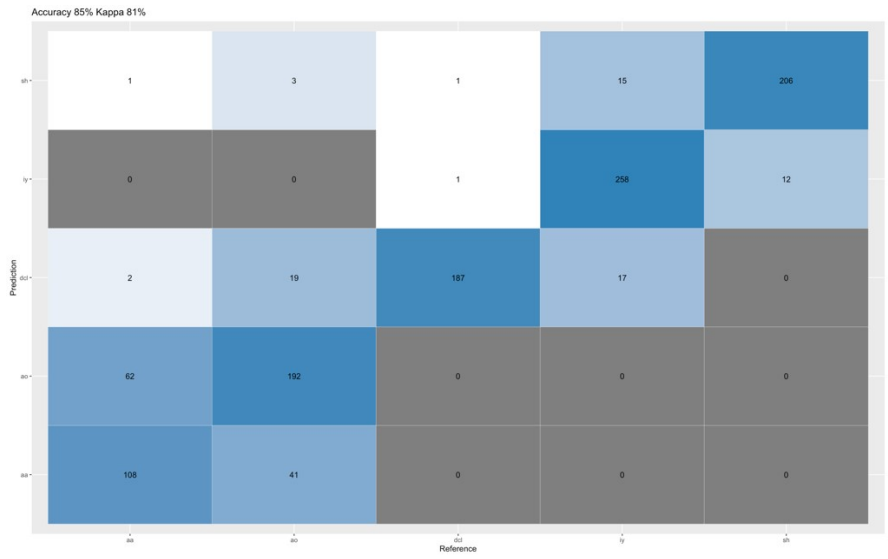
Aparecen 1021 vectores soporte.

3) Calcule la matriz de confusión utilizando la instrucción confusionMatrix de la librería caret. Muestre una captura de pantalla de la matriz **y los resultados obtenidos.** pred=predict(svm,testeo) confusionMatrix(pred,testeo$fonemas)

```
> pred=predict(svmAsignado,testeo)
> confusionMatrix(pred,testeo$fonemas)
Confusion Matrix and Statistics

          Reference
Prediction  aa  ao dcl  iy  sh
       aa  108  41   0   0   0
       ao   62 192   0   0   0
       dcl   2  19 187  17   0
       iy    0   0   1 258  12
       sh    1   3   1  15 206

Overall Statistics

               Accuracy : 0.8453
                 95% CI : (0.8229, 0.866)
    No Information Rate : 0.2578
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8052

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: aa Class: ao Class: dcl Class: iy Class: sh
Sensitivity             0.6243    0.7529     0.9894    0.8897    0.9450
Specificity             0.9569    0.9287     0.9594    0.9844    0.9779
Pos Pred Value          0.7248    0.7559     0.8311    0.9520    0.9115
Neg Pred Value          0.9334    0.9277     0.9978    0.9625    0.9867
Prevalence              0.1538    0.2267     0.1680    0.2578    0.1938
Detection Rate          0.0960    0.1707     0.1662    0.2293    0.1831
Detection Prevalence    0.1324    0.2258     0.2000    0.2409    0.2009
Balanced Accuracy       0.7906    0.8408     0.9744    0.9370    0.9615
```

4) ¿Cuántos registros quedaron bien y mal clasificados?

El 84% de los registros quedaron bien clasificados. 951 registros quedaron bien clasificados y 174 mal de un total de 1125 registros del conjunto de testeo.



5) ¿Cuál fue el accuracy?

El accuracy fue 0.84.

6) ¿Cuál tipo de fonema tiene la mayor sensibilidad?

El fonema con mayor sensibilidad fue *dcl* con 0.98. En segundo lugar, *sh* con 0.94.

7) Prediga con la SVM el registro de los 3 últimos números de su DNI. predict(svm,Fonema) ¿Coincide la predicción con lo esperado?

Sí, predice correctamente.

```
> predict(svmAsignado,Fonema)
945
 ao
Levels: aa ao dcl iy sh
```

Parte D – SVM2

1) Cargue la libreríae 1071 y modele el problema planteado con una Support Vector Machine con alguna variación de la SVM del punto anterior. Por ejemplo cambie el kernel y/o el costo y/o algún otro parámetro.
No es necesario que los resultados de esta SVM sean mejores que los de la SVM anterior. Muestre el código R utilizado.

```
> svmLinear=svm(fonemas~.,entreno,kernel="linear")
> svmLinear

Call:
svm(formula = fonemas ~ ., data = entreno, kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  698
```

8) Calcule la matriz de confusión utilizando la instrucción confusionMatrix de la librería caret. Muestre una captura de pantalla de la matriz y los resultados obtenidos. pred=predict(svm,testeo) confusionMatrix(pred,testeo$fonemas)

```
> predL=predict(svmLinear,testeo)
> confusionMatrix(predL,testeo$fonemas)
Confusion Matrix and Statistics

          Reference
Prediction  aa  ao dcl  iy  sh
       aa  115  48   0   0   0
       ao   57 207   0   0   0
       dcl   0   0 188   3   0
       iy    1   0   1 287   0
       sh    0   0   0   0 218

Overall Statistics

               Accuracy : 0.9022
                 95% CI : (0.8834, 0.919)
    No Information Rate : 0.2578
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8766

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: aa Class: ao Class: dcl Class: iy Class: sh
Sensitivity             0.6647    0.8118     0.9947    0.9897    1.0000
Specificity             0.9496    0.9345     0.9968    0.9976    1.0000
Pos Pred Value          0.7055    0.7841     0.9843    0.9931    1.0000
Neg Pred Value          0.9397    0.9443     0.9989    0.9964    1.0000
Prevalence              0.1538    0.2267     0.1680    0.2578    0.1938
Detection Rate          0.1022    0.1840     0.1671    0.2551    0.1938
Detection Prevalence    0.1449    0.2347     0.1698    0.2569    0.1938
Balanced Accuracy       0.8072    0.8731     0.9958    0.9936    1.0000
```

9) Prediga con la SVM el registro de los 3 últimos números de su DNI. predict(svm,Fonema) ¿Coincide la predicción con lo esperado?

Sí, coincide con lo esperado.

```
> predict(svmLinear,Fonema)
945
 ao
Levels: aa ao dcl iy sh
```

Parte E – Naive Bayes

1) Cargue la librería e1071 y modele el problema planteado utilizando Naive Bayes nb=naiveBayes(fonemas~.,entreno)

2) Calcule la matriz de confusión utilizando la instrucción confusionMatrix de la librería caret. Muestre una captura de pantalla de la matriz **y los resultados obtenidos.** pred=predict(nb,testeo) confusionMatrix(pred,testeo$fonemas)

```
> pred=predict(nb,testeo)
> confusionMatrix(pred,testeo$fonemas)
Confusion Matrix and Statistics

          Reference
Prediction  aa  ao dcl  iy  sh
       aa  110  70   0   0   0
       ao   63 185   1   1   0
       dcl   0   0 176   1   0
       iy    0   0  12 285   3
       sh    0   0   0   3 215

Overall Statistics

               Accuracy : 0.8631
                 95% CI : (0.8416, 0.8827)
    No Information Rate : 0.2578
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8272

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: aa Class: ao Class: dcl Class: iy Class: sh
Sensitivity            0.63584    0.7255     0.9312    0.9828    0.9862
Specificity            0.92647    0.9253     0.9989    0.9820    0.9967
Pos Pred Value         0.61111    0.7400     0.9944    0.9500    0.9862
Neg Pred Value         0.93333    0.9200     0.9863    0.9939    0.9967
Prevalence             0.15378    0.2267     0.1680    0.2578    0.1938
Detection Rate         0.09778    0.1644     0.1564    0.2533    0.1911
Detection Prevalence   0.16000    0.2222     0.1573    0.2667    0.1938
Balanced Accuracy      0.78115    0.8254     0.9651    0.9824    0.9915
```

3) ¿Cuántos registros quedaron bien y mal clasificados?

Se calcularon correctamente 971 registros y, quedaron mal clasificados 154.

4) Prediga con Naive Bayes el registro de los 3 últimos números de su DNI. predict(nb,Fonema) ¿Coincide la predicción con lo esperado?

No, no predijo correctamente.

```
> predict(nb,Fonema)
[1] aa
Levels: aa ao dcl iy sh
```

Parte F – Comparación

1) Cree una tabla con el accuracy de cada modelo, y la sensibilidad y especificidad de cada modelo por categoría. La tabla esperada no es necesario hacerla con R, sino una tabla tipo Word.

	Modelo 1: SVM Sigmoidea	Modelo 2: SVM Lineal	Modelo 3: Naive Bayes
Accuracy	0.84	0.90	0.86

Sensibilidad

Modelo/Fonema	*aa*	*ao*	*dcl*	*iy*	*sh*
Modelo 1	0.62	0.75	0.98	0.88	0.94
Modelo 2	0.66	0.81	0.99	0.98	1.00
Modelo 3	0.63	0.72	0.93	0.98	0.98

Especificidad

Modelo/Fonema	*aa*	*ao*	*dcl*	*iy*	*sh*
Modelo 1	0.95	0.92	0.95	0.98	0.97
Modelo 2	0.94	0.93	0.99	0.99	1.00
Modelo 3	0.92	0.92	0.99	0.98	0.99

2) Compare los resultados obtenidos con las SVM y Naive Bayes. ¿Cuál modelo le parece que resultó mejor? ¿Según qué criterio? No promedie las sensibilidades ni especificidades.

El modelo 2, lineal con Support Vector Machine, obtuvo el mejor accuracy con 0.90. Por otro lado, resulta llamativo que el modelo 1 (SVM Sigmoidea) tuvo un accuracy menor que modelo que el modelo 3 (Naive Bayes) y, sin embargo, logró predecir correctamente el Fonema seleccionado de muestra.

Ejercicio 2 – Aprendizaje No Supervisado Parte A – Análisis Exploratorio de Datos

1) Abra la base *rock*. data(rock) ¿Cuántos registros tiene y cuántas variables?

La base cuenta con 4 variables: "area", "peri", "shape" y "perm" y, 48 registros.

2) Muestre un head y summary de la base. ¿Qué significa cada variable?

```
> head(rock)
  area    peri     shape perm
1 4990 2791.90 0.0903296  6.3
2 7002 3892.60 0.1486220  6.3
3 7558 3930.66 0.1833120  6.3
4 7352 3869.32 0.1170630  6.3
5 7943 3948.54 0.1224170 17.1
6 7979 4010.15 0.1670450 17.1
```

```
> summary(rock)
      area            peri           shape              perm
 Min.   : 1016   Min.   : 308.6   Min.   :0.09033   Min.   :  6.30
 1st Qu.: 5305   1st Qu.:1414.9   1st Qu.:0.16226   1st Qu.:  76.45
 Median : 7487   Median :2536.2   Median :0.19886   Median : 130.50
 Mean   : 7188   Mean   :2682.2   Mean   :0.21811   Mean   : 415.45
 3rd Qu.: 8870   3rd Qu.:3989.5   3rd Qu.:0.26267   3rd Qu.: 777.50
 Max.   :12212   Max.   :4864.2   Max.   :0.46413   Max.   :1300.00
```

El dataset contiene muestras de piedras de yacimientos de petróleo dónde la variable área refiere al espacio de los poros; peri significa perímetro en pixeles; shape es el perímetro sobre la raíz del área y, permeabilidad de las piedras.

Parte B – Red Neuronal Kohonen

1) Transforme la base en una matriz llamada “somRock” somRock=as.matrix(rock)

2) Para el seteo de semilla considere los 3 últimos dígitos de su DNI y realice un agrupamiento con una Red Neuronal Kohonen con somgrid(1,cantGrupos,”hexagonal”)

Deje el resto de los parámetros por defecto. Indique cómo quedó el código R utilizado.

```
set.seed(945);som=som(somRock,grid=somgrid(1,4,"hexagonal"))
```

3) Muestre som$unit.classif. ¿A qué grupo pertenece el elemento 8 de la base?

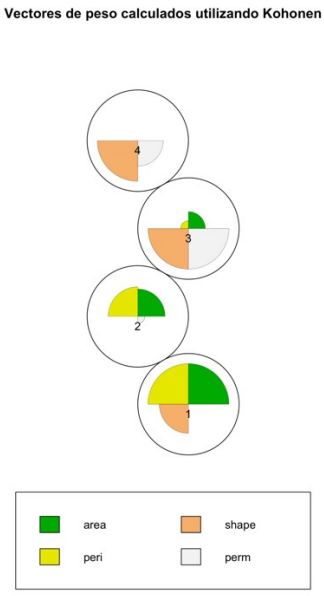
```
> som$unit.classif
 [1] 3 2 2 2 2 2 2 2 2 3 2 2 1 2 2 2 1 1 1 1 2 1 1 2 3 4 3 3 3 3 3 3 3 3 3 3 4 4 4 3 3 4 3 2 4 4 3 2
```

El elemento 8 pertenece al grupo 2.

4) Muestre una captura de pantalla de los vectores de pesos. Indique el código R utilizado.

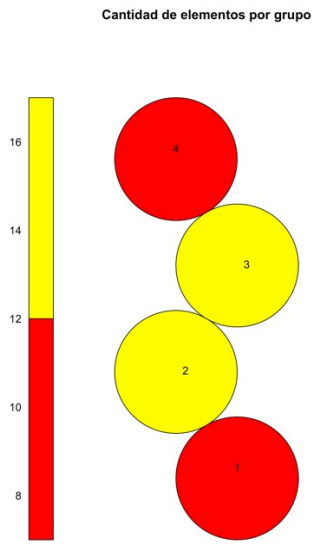
```
> som$codes
[[1]]
         area      peri     shape      perm
V1 11174.303 4562.7415 0.2231601  82.63817
V2  8405.724 3575.3466 0.1844857 204.42149
V3  6226.117 1629.7517 0.2382950 780.31212
V4  2705.212  928.0808 0.2384031 519.84374
```

5) Realice un gráfico con los vectores de peso. Indique el código R utilizado.



plot(som,type="codes", main="Vectores de peso calculados utilizando Kohonen")

6) Realice un gráfico con la cantidad de elementos que quedaron en cada grupo. Indique el código R utilizado.



cantidad=plot(som,type="count", main="Cantidad de elementos por grupo")
identify(som) cantidad

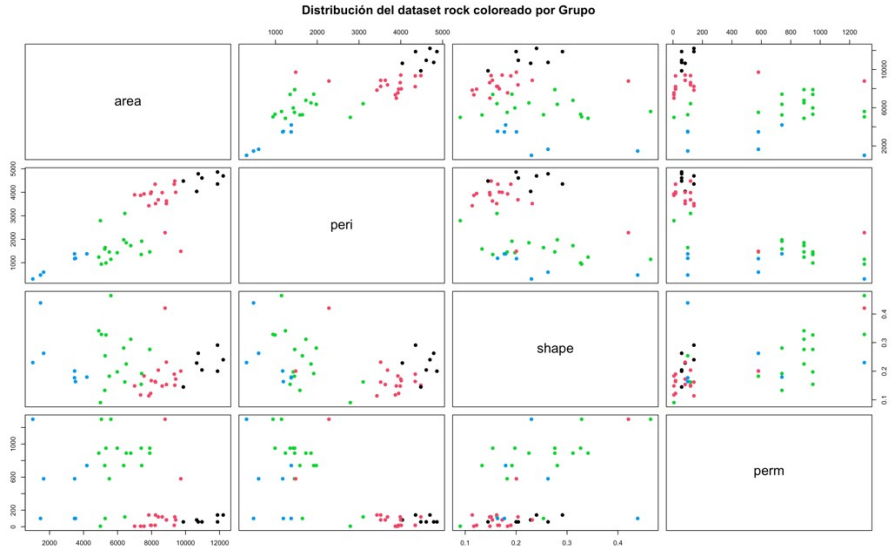
7) ¿Cuántos elementos quedaron en cada grupo? Indique el código R utilizado.

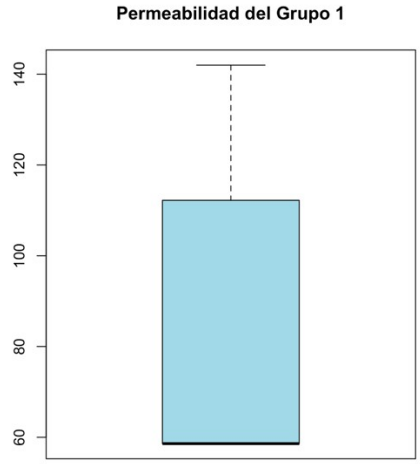
Los grupos se distribuyen de la siguiente manera:

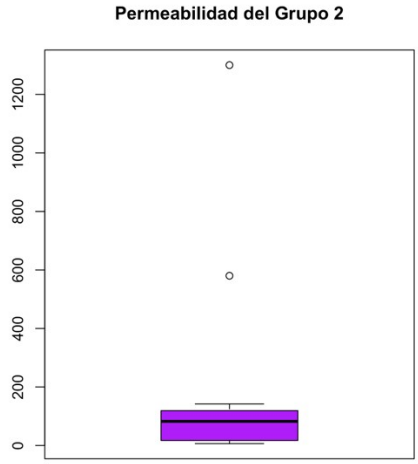
- Grupo 1: 7 elementos
- Grupo 2: 17 elementos
- Grupo 3: 17 elementos
- Grupo 4: 7 elementos

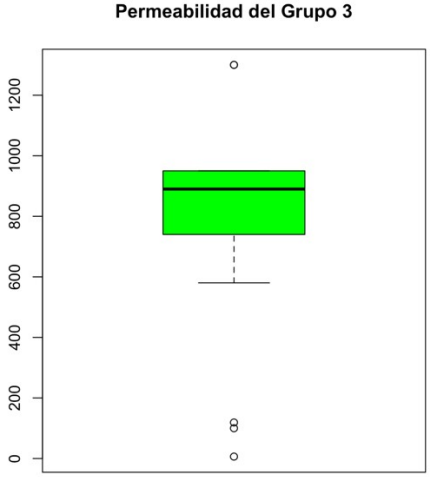
cantidad=plot(som,type="count", main="Cantidad de elementos por grupo")
identify(som) cantidad

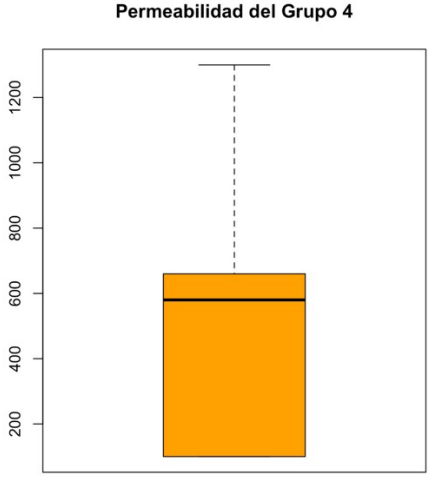
8) Determine alguna característica de alguno de los grupos con respecto al resto de los grupos.



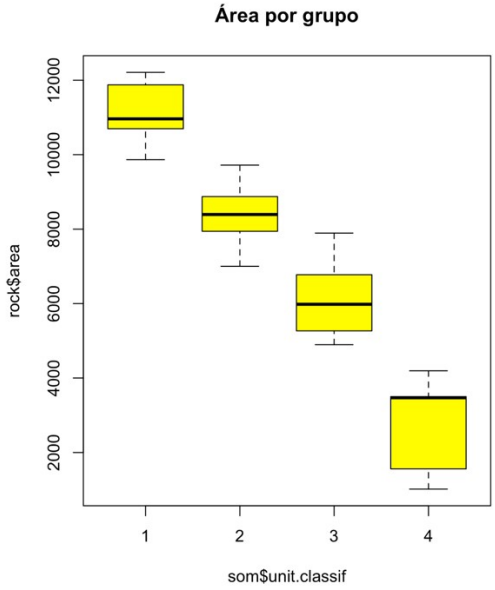






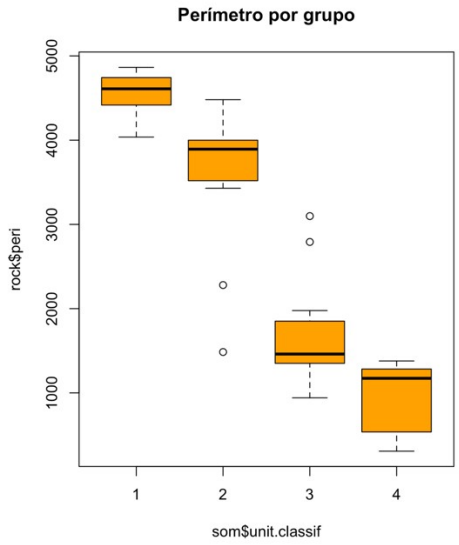


Los boxplots por grupo permiten observar que la permeabilidad del grupo 1 tiene valores menores que el resto de los grupos, siendo estás un grupo de piedras menos permeables.

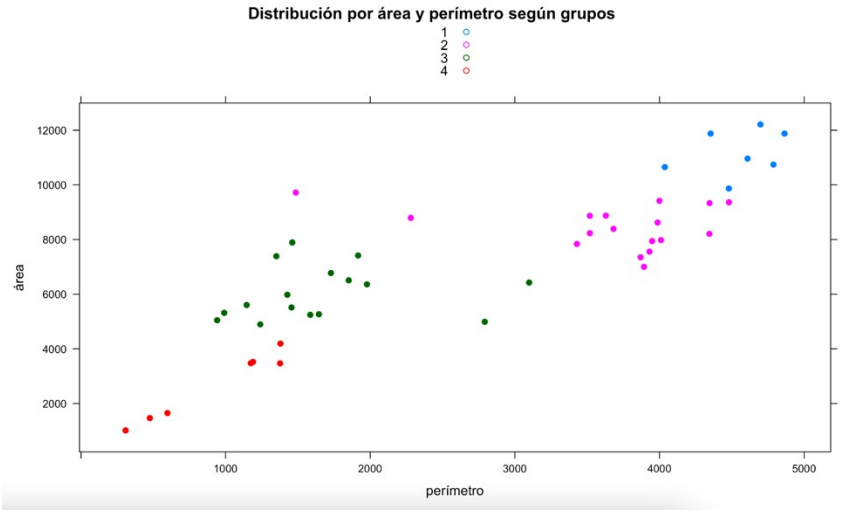


Asimismo, el grupo 1 concentra las piedras con más área del dataset. El área se encuentra distribuida entre los grupos de manera decreciente.

En cambio si se analiza el perímetro, la presencia de outliers indica que la clasificación de algunos registros entre los grupos 2 y 3 no es exacta.



Optativo: Realice un gráfico con dos variables coloreado por los grupos formados. Indique el código R utilizado.




```
xyplot(area~peri,rock,groups=som$unit.classif,auto.key=TRUE,pch=19, ylab="área",
xlab="perímetro", main= "Distribución por área y perímetro según grupos")
```

Trabajo Práctico Final

Anexo

```
#TRABAJO PRACTICO FINAL #PARTE

A

library(wesanderson)

base=read.table("phoneme.data",sep=",",header=TRUE)

base=phoneme.data head(base)

tail(base) dim(base) base$speaker=NULL

base$row.names=NULL

names(base)[names(base)=="g"]="fonemas"

base$fonemas=factor(base$fonemas)



table(base$fonemas)  plot(base$fonemas,main="Fonemas",

col=wes_palette(name="Cavalcanti1"))



Fonema=base[945,]

Fonema



#PARTE B

library(caret)

set.seed(945);particion=createDataPartition(y=base$fonemas,p=0.75,list= FALSE)

entreno=base[particion,] testeo=base[-particion,]



head(entreno) summary(entreno)

head(testeo) summary(testeo)



dim(entreno) dim(testeo)
```

```
table(base$fonemas) table(entreno$fonemas)

table(testeo$fonemas)



#Opcional: dónde quedó el registro FONEMA

PruebaBase=base[ "945",]

PruebaBase



PruebaE=entreno[ "945",]

PruebaE



PruebaT=testeo["945",]

PruebaT

#Quedó en entreno



#PARTE C

library(e1071)

svmAsignado=svm(fonemas~.,entreno,kernel="sigmoid")

svmAsignado pred=predict(svmAsignado,testeo)  cfm<-

confusionMatrix(pred,testeo$fonemas)

#Bien=(108+192+187+258+206)



#Bien

#Mal=(41+62+19+2+1+3+1+1+15+17+12)

#Mal

#Bien+Mal #ggplot confusion matrix library(ggplot2)

library(scales) ggplotConfusionMatrix <- function(m){   mytitle <-

paste("Accuracy", percent_format()(m$overall[1]),
```

```
"Kappa", percent_format()(m$overall[2]))   p <-     ggplot(data =

as.data.frame(m$table) ,            aes(x = Reference, y =

Prediction)) +     geom_tile(aes(fill = log(Freq)), colour = "white")

+     scale_fill_gradient(low = "white", high = "steelblue") +

geom_text(aes(x = Reference, y = Prediction, label = Freq)) +

theme(legend.position = "none") +     ggtitle(mytitle)   return(p)

}

ggplotConfusionMatrix(cfm)

predict(svmAsignado,Fonema)



#PARTE D

svmLinear=svm(fonemas~.,entreno,kernel="linear")

svmLinear predL=predict(svmLinear,testeo)

confusionMatrix(predL,testeo$fonemas)

predict(svmLinear,Fonema) #PARTE E

nb=naiveBayes(fonemas~.,entreno)

pred=predict(nb,testeo)

confusionMatrix(pred,testeo$fonemas)

predict(nb,Fonema)

#Bien=110+185+176+285+215

#Bien

#Mal=63+70+12+1+1+1+3+3

#Mal

#Bien+Mal

#ejercicio 2

data(rock) dim(rock)

names(rock)
```

```
head(rock)

summary(rock)



somRock=as.matrix(rock)  library(kohonen)

set.seed(945);som=som(somRock,grid=somgrid(1,4,"hexagonal"))

som$unit.classif



som$codes

library(caret) plot(som) plot(som,type="codes", main="Vectores de peso

calculados utilizando Kohonen") identify(som)



cantidad=plot(som,type="count", main="Cantidad de elementos por grupo") identify(som) cantidad

boxplot(rock) plot(rock,col=som$unit.classif,pch=19, main= "Distribución del dataset rock

coloreado por Grupo")

xyplot(area~peri,rock,groups=som$unit.classif,auto.key=TRUE,pch=19, ylab="área", xlab="perímetro", main= "Distribución por área y
perímetro según grupos")

Grupo1<-rock[som$unit.classif==1,]

Grupo2<-rock[som$unit.classif==2,]

Grupo3<-rock[som$unit.classif==3,] Grupo4<-

rock[som$unit.classif==4,]



par(mfrow = c(1, 2)) boxplot(Grupo1$perm, main= "Permeabilidad del

Grupo 1", col="lightblue") boxplot(Grupo2$perm, main= "Permeabilidad del

Grupo 2", col= "purple") boxplot(Grupo3$perm, main= "Permeabilidad del

Grupo 3", col= "green") boxplot(Grupo4$perm, main= "Permeabilidad del

Grupo 4", col= "orange")
```

```
boxplot(rock$area~som$unit.classif, col="yellow", main="Área por grupo") boxplot(rock$peri~som$unit.classif,
col="orange", main="Perímetro por grupo") boxplot(rock$shape~som$unit.classif, col="yellow", main="shape
por grupo")
```