Problema 7: Reconeixement de lletres amb la xarxa MLP [R,G]

Gerard Barrachina Josep de Cid Albert Ribes
Kerstin Winter

20 de diciembre de 2017

1. Dissenyeu una funció que generi versions corruptes d'una lletra, a còpia de canviar un cert número de bits de manera aleatòria. Una manera senzilla és generar primer el número de bits corruptes p.e. amb una Poisson ($\lambda = 1{,}01$) seleccionar els bits concrets (uniformement) i després invertir-los.

```
        corrupt = function(bit_vector, nchanges) {
changes = sample(length(bit_vector), nchanges)
g = rep(0, length(bit_vector))
for (n in changes) {
g[n] = 1
}
new = as.numeric(xor(bit_vector, g))
return(new)
}
```

2. Dissenyeu una funció que, partint de les lletres netes (arxiu corruptes que usarem com a mostra de training, de mida letters.txt), generi unes dades N.

```
generate_corrupt = function(df, n) {
  nm = c(1:35, "letter")
  g = sample(1:nrow(df), n, replace = TRUE)
  ch = rpois(n, 1.01)
  new_data = data.frame()
  for (i in 1:length(g)) {
    elem = g[i]
    new_vector = corrupt(df[elem, -36], ch[i])
    newrow = c(new_vector, df[elem, 36])
    new_data = rbind(new_data, newrow)
    new_data[i,36] = as.character(df[elem, 36])
  }
  colnames(new_data) = nm
  new_data$letter = as.factor(new_data$letter)
  return(new_data)
}
```

3. Entreneu una xarxa MLP per aprendre la tasca. Caldrà que estimeu la millor arquitectura, cosa que podeu fer per cross-validation, usant regularització.

```
        train_data = generate_corrupt(mydata, 1000)
test_data = generate_corrupt(mydata, 300)

trc <- trainControl (method="repeatedcv", number=10, repeats=5)
## WARNING: this takes maaaaaany minutes
caret.nnet.model <- train (
  letter ~.,
  data = train_data,
  method='nnet',
  metric = "Accuracy",
  trControl=trc)

test_results = predict(caret.nnet.model, test_data)

confusionMatrix(test_data$letter, test_results)
```

4. Reporteu els resultats de predicció en una mostra de test gran -també generada per vosaltres, i de manera anàloga a la de training.

   Usando repeatedcv hemos visto que la mejor red tine 5 neuronas, haciendo regulación con $decay = 0{,}1$.

   Hemos conseguido una accuracy de 0,93