
```
 fichero = read.csv("distancia_universitarios.csv")
 fichero
```

```
##      Distancia
## 1         16.5
## 2         34.8
## 3         20.7
## 4          6.2
## 5          4.4
## 6          3.4
## 7         24.0
## 8         24.0
## 9         32.0
## 10        30.0
## 11        33.0
## 12        27.0
## 13        15.0
## 14         9.4
## 15         2.1
## 16        34.0
## 17        24.0
## 18        12.0
## 19         4.4
## 20        28.0
## 21        31.4
## 22        21.6
## 23         3.1
## 24         4.5
## 25         5.1
## 26         4.0
## 27         3.2
## 28        25.0
## 29         4.5
## 30        20.0
## 31        34.0
## 32        12.0
## 33        12.0
## 34        12.0
## 35        12.0
## 36         5.0
## 37        19.0
## 38        30.0
## 39         5.5
## 40        38.0
## 41        25.0
```

```
## 42      3.7
## 43      9.0
## 44     30.0
## 45     13.0
## 46     30.0
## 47     30.0
## 48     26.0
## 49     30.0
## 50     30.0
## 51      1.0
## 52     26.0
## 53     22.0
## 54     10.0
## 55      9.7
## 56     11.0
## 57     24.1
## 58     33.0
## 59     17.2
## 60     27.0
## 61     24.0
## 62     27.0
## 63     21.0
## 64     28.0
## 65     30.0
## 66      4.0
## 67     46.0
## 68     29.0
## 69      3.7
## 70      2.7
## 71      8.1
## 72     19.0
## 73     16.0

len = function(list){
  count = 0
  for (element in list){
    count = count + 1
  }
  count
}

distancias = fichero$Distancia

longitud = len(distancias)
longitud
```

```

## [1] 73

bubble = function(list, asc = TRUE){
  n = len(list)
  if(asc){
    for (i in 2:n){
      for (j in 1:(n-1)){
        if (list[j] > list[j+1]){
          temp = list[j]
          list[j] = list[j+1]
          list[j+1] = temp
        }
      }
    }
  }
  else {
    for (i in 2:n){
      for (j in 1:(n-1)){
        if (list[j] < list[j+1]){
          temp = list[j]
          list[j] = list[j+1]
          list[j+1] = temp
        }
      }
    }
  }
  list
}

distanciasordenadas = bubble(distancias, FALSE)
distanciasordenadas

## [1] 46.0 38.0 34.8 34.0 34.0 33.0 33.0 32.0 31.4 30.0 30.0 30.0 30.0 30.0 30.0
## [16] 30.0 30.0 29.0 28.0 28.0 27.0 27.0 27.0 26.0 26.0 25.0 25.0 24.1 24.0 24.0
## [31] 24.0 24.0 22.0 21.6 21.0 20.7 20.0 19.0 19.0 17.2 16.5 16.0 15.0 13.0 12.0
## [46] 12.0 12.0 12.0 12.0 11.0 10.0 9.7 9.4 9.0 8.1 6.2 5.5 5.1 5.0 4.5
## [61] 4.5 4.4 4.4 4.0 4.0 3.7 3.7 3.4 3.2 3.1 2.7 2.1 1.0

rank = function(list){
  ordered_list = bubble(list)
  ordered_list[len(ordered_list)] - ordered_list[1]
}

rango = rank(distanciasordenadas)
rango

## [1] 45

```

```

absolute_freq = function(list){
  ordered_list = bubble(list)
  n = len(ordered_list)
  elements = vector()
  frequencies = vector()
  i = 1
  while (i <= n){
    actual_element = ordered_list[i]
    elements = append(elements, actual_element)
    actual_freq = 0
    j = i
    while(j <= n & actual_element == ordered_list[j]){
      actual_freq = actual_freq + 1
      j = j+1
    }
    frequencies = append(frequencies, actual_freq)
    i = j
  }
  rbind(elements, frequencies)
}

```

PARTE 2

```

union = function(c1, c2){
  if (len(c1) == 0){
    c2
  }
  else if (is.element(c1[1], c2)){
    union(c1[-1], c2)
  }
  else{
    union(c1[-1], append(c2, c1[1]))
  }
}

unionp = union(c("P", "A", "L"), c("P", "A", "C", "N"))
unionp

intersect = function(c1, c2){
  if (len(c1) == 0){
    c()
  }
  else if (is.element(c1[1], c2)){

```

```

append(intersect(c1[-1], c2), c1[1])
}
else{
intersect(c1[-1], c2)
}
}

intersectp = intersect(c("P", "A", "L"), c("P", "A", "C", "N"))
intersectp

dif = function(c1, c2) {
res = c()
for (element in c1) {
if (!(element in c2)) {
res = append(res, element)
}
}
res
}

tabla <- matrix(c(1,1,0,1,1, 1,1,1,1,0, 1,1,0,1,0, 1,0,1,1,0, 1,1,0,0,0, 0,0,0,1,0),6,5,

tabla

support = function(table, elements){
count_support = 0
for (i in 1:len(table[,1])){
acum = 1
for (element in elements){
acum = (table[i,element]) & acum
}
count_support = count_support + acum
}
count_support/len(table[,1])
}

soporte = support(tabla, c("P", "A"))
soporte

support_clasif = function(table, ocurrences, s){
valid_ocurrences = c()
for (ocurrence in ocurrences){
support_oc = support(table, ocurrence)

```

```

if (support_oc >= s){
valid_ocurrences = append(valid_ocurrences, ocurrence)
}
}
valid_ocurrences
}

create_comb = function(table, clasif, s) {
lista = c()
for (i in 2:len(clasif)) {
comb = unlist(lapply(i, function(m) {combn(clasif, m=m, simplify=TRUE)}), recursive=FALSE)

for (j in seq(1, len(comb), by=i)) {
add = c()
for (k in j:(j+i-1)) {
add = append(add, comb[k])
}
if (support(table, add) >= s) {
lista = append(lista, list(add))
}
}
}
lista
}

confidence = function(table, comb) {
for (i in 2:len(table[1,])) { #itera dimensiones k
separar <- Filter(function(x) length(x)==i, comb)

for (j in 1:len(separar)) { #itera elementos de dim k

for (k in 1:len(separar[j])) { #itera posibilidades que haya en el lado izquierdo
#len(separar[j]) == i ?????

lado = lapply(k, function(m) {combn(separar[j], m=m, simplify=TRUE)})

for (n in 1:len(lado[,1])) { #

otro_lado = dif(separar[j], lado[,n])
hola = c(lado[n], otro_lado)
print(hola)
}
}
}
}

```

```

}
}
}

apriori = function(table, s, c) {
  soporte_clasif = support_clasif(tabla, c(c("P"), c("A") ,c("L"), c("C"), c("N")), s)
  print(soporte_clasif)
  combinaciones = create_comb(tabla, soporte_clasif, s)
  print(combinaciones)
  conf = check_conf()
}

apriori(tabla, 0.5, 0.8)

## Error: <text>:35:31: inesperado 'in'
## 34:         for (element in c1) {
## 35:             if (!(element in
##                      ^

```

40 de soporte y 90 de confianza