

spamizer

Marc Sànchez, Francesc Xavier Bullich, Gil Gassó

5/8/2019

```
# x és el nom del fitxer que volem carregar
loadFormattedData <- function(x){

  tmp = read.csv(x)
  names(tmp) <- c("id", "phi", "k", "tp", "tn", "fp", "fn", "nham", "nspam")

  #Calculem els tcr dels valors
  # BASE : (NSPAM) / (50 * NHAM + NSPAM)
  base <- tmp$nspam / (50 * tmp$nham + tmp$nspam)
  # WERR: (50 * FP + FN)/(50 * NHAM + NSPAM) + 0.000001 -> per que no sigui 0
  werr <- (50 * tmp$fp + tmp$fn) / (50 * tmp$nham + tmp$nspam) + 0.000001
  # TCR : BASE / WERR
  tcr <- base/werr

  # Calculem l'accuracy
  accuracy <- (tmp$nspam + tmp$nham - tmp$fp - tmp$fn)/(tmp$nspam + tmp$nham) * 100

  # Generar una matriu que permeti representar els resultats en funció de k i phi
  values <- data.frame(accuracy, tcr)
  names(values) <- c("accuracy", "tcr")
  head(values)

  tmp <- cbind(tmp, values)

  # Ordenem els valors
  tmp <- tmp[order(-tmp$tcr), ]

  return(tmp)
}
```

Treballem amb la variable K

Mesurem la mitjana que en aquest cas hauria de caure sobre entre 0 i 3 ja que els valors aleatòris estan generats a partir d'aquest màxim i mínim :

```
#median(results$K)
```

A partir d'aquí mirem la desviació de la variable i la variança.

```
#var(results$K)
#sd(results$K)
```

Vull mirar si les variables phi i k estan correlacionades.

```
// TODO : Posar el projecte al github.
```

Naive Bayes

En aquest apartat s'especifica com s'adapta el mètode de naive bayes al filtratge de correu.

Naive Bayes.

// TODO : S'ha de parlar de tot el que es fa a dins del mètode que tenim implementat al codi.

Assumpcions.

// TODO : Comentar tot el que es dona per sentat al utilitzar aquest mètode, com per exemple que la màquina està ben entrenada ...

Punts forts i febles del mètode de Naive Bayes.

// TODO :

Aplicació.

En aquest apartat s'es

Tecnologies escollides.

Comentar també la comparació de l'ús de base de dades en memòria.

Manual de l'aplicació.

Utilització.

Implementació.

Lectura de fitxers.

Mètode de selecció.

Adaptació del mètode K-fold cross-validation.

En comptes de realitzar la divisió ...

Filtre i abstracció del filtratge.

Stanford Core NLP.

Custom Filter.

Entrenament.

Validació.

// TODO : Explicar la nostre adaptació del mètode hill climbing utilitzat.

Compute (Application, adaptació del mètode Hill Climbing).

Fase Experimental.

// TODO : ... Pensar l'estructura encara.

Evaluació dels FP i dels FN en funció de K i PHI

// TODO :

Càlcul del TCR (Total Cost Ratio)

Amb el Total cost ratio podem extreure un valor que pondera amb més força el valor de les aparicions dels falços positius. El que es busca amb el Tcr és el valor màxim possibles. Per fer-ho hem realitzat varíes execucions i hem preparat una sèrie de conclusions per intentar esbrinar les funcions phi i k que millor s'acosten al nostre problema mitjançant el càlcul del tcr.

Veiem com es genera la columna TCR

```
results = read.csv("/Users/marc Sanchez/Projects/spamizer/analisis/20000m-500n-SF.csv")
#Calculem els tcr dels valors
# BASE : (NSPAM) / (50 * NHAM + NSPAM)
base <- results$NSPAM / (50 * results$NHAM + results$NSPAM)
# WERR: (50 * FP + FN)/(50 * NHAM + NSPAM) + 0.000001 -> per que no sigui 0
werr <- (50 * results$FP + results$FN) / (50 * results$NHAM + results$NSPAM) + 0.000001
# TCR : BASE / WERR
tcr <- base/werr
```

Carreguem les dades des de la funció loadFormattedData que ja incorpora le càlcul del tcr i l'accuracy

Anàlisi de la PHI i la K.

PHI

Si tenim en compte el què representa els valors de phi, el que ens trobem és que la phi és el factor d'increment de la probabilitat per que un correu sigui considerat SPAM. És a dir un valor de phi = 2, provoca que per que

un correu sigui considerat spam ha de ser 2 cops superior a la probabilitat de que sigui ham. Un valor de phi = 1 fa que no hi hagi increment obligatori per a la comparació.

El valor mínim que té sentit assignar-li a phi és 1 i el màxim el podríem limitar a 5 com a molt o inclús a 6 si el que volem és no tenir cap correu que sigui Ham i que el consideri com Spam.

K

Quan apliquem el suavitzat hem de tenir en compte que donats el bag of words de ham i el de spam, què passa si la paraula no existeix? doncs que el valor de les multiplicacions serà 0 i farà que si una paraula no existeix aquesta paraula ens determini si un correu és ham o és spam.

Per tant la k estipula el valor que se li assigna a una paraula quan aquesta no és present. Aquest valor no pot ser 0 però pot ser proper a zero. Si fos zero es provocaria el mateix cas que l'esmentat anteriorment. Tanmateix no té sentit aplicar un valor molt gran a la k ja que si ho fem aquest valor provocaria que les paraules que no existeixen fossin puntuades molt altes i que les aparicions no computessin tant.

Limitarem els valors de k en un rang de (0 - 3].

Coportament de phi i k en simulació.

La idea d'aquest apartat és acotar el rang d'actuació de phi i de k en funció dels resultats trobats sobre el tcr. Per fer-ho s'ha fet una simulació d'aproximadament 22000 valors de tcr per un nombre de 20000 missatges. Per cada execució de la simulació, d'aquests 20000 missatges processats, s'ha utilitzat entre un 5% i un 15% dels mateixos per validació i la resta s'ha utilitat per entrenament. A cada execució de la simulació els missatges utilitzatzs per validació s'han seleccionat de manera aleatòria dins del total de missatges (els 20000).

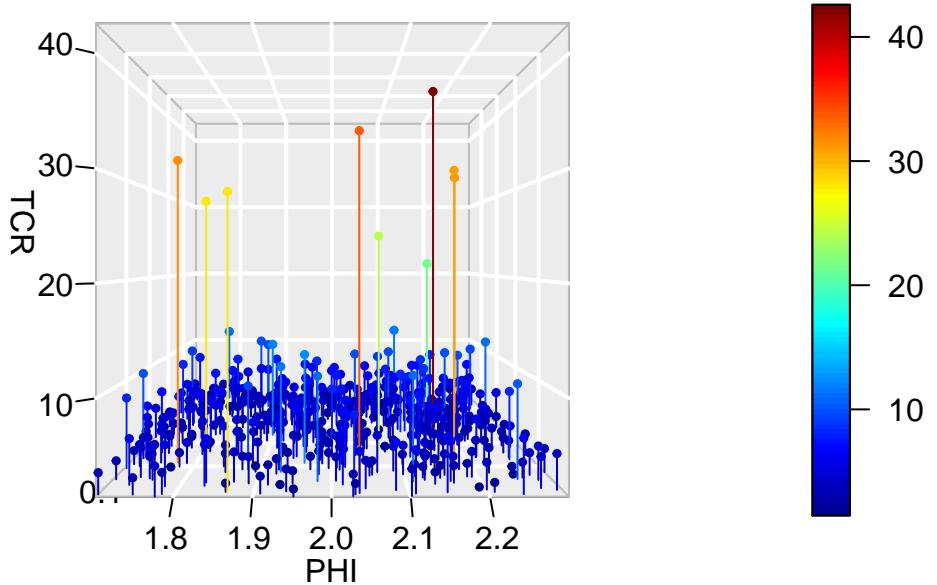
Carreguem les diferents simulacions i les carreguem en un dataframe per poder processar les dades, utilitzem la funció loadFormattedData declarada a l'aratat de funcions del document.

```
b1 = loadFormattedData("/Users/marc Sanchez/Projects/spamizer/analisis/m20000-n9500-SF-P-16-k-03.csv")
b2 = loadFormattedData("/Users/marc Sanchez/Projects/spamizer/analisis/m20000-n10000-P-15-K-03.csv")
b3 = loadFormattedData("/Users/marc Sanchez/Projects/spamizer/analisis/20000m-500n-SF.csv")
b4 = loadFormattedData("/Users/marc Sanchez/Projects/spamizer/analisis/2000m-1000n-phi-1.7-2.3-k-0-0.5.csv")

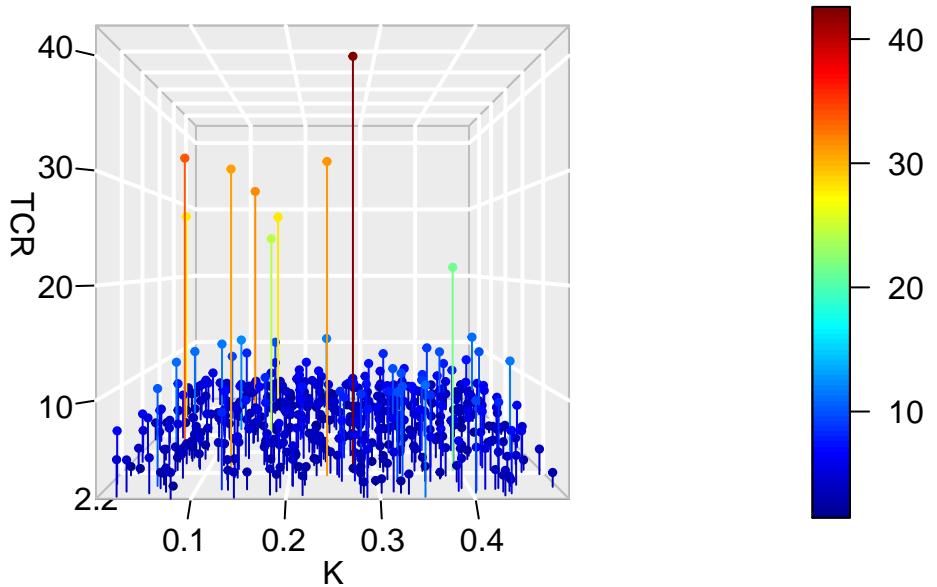
v <- rbind(b1, b2, b3, b4)
v <- v[order(-v$tcr), ]
head(v)

##          id      phi       k   tp   tn fp fn nham nspam accuracy      tcr
## 66031 8637 1.778913 0.5925286 695 625  0 11  695   636 99.17355 57.63278
## 26510 2299 1.362548 0.6460839 688 644  0 15  688   659 98.88641 43.83089
## 14114 1064 2.179125 0.2739705 607 542  0 13  607   555 98.88124 42.59106
## 38981 5932 1.734514 1.0386465 694 666  0 16  694   682 98.83721 42.53095
## 20121 4046 2.723349 0.4087232 836 740  0 18  836   758 98.87077 42.01178
## 59141 7948 4.616297 0.4507945 1075 948  0 25 1075   973 98.77930 38.83499

scatter3D(b4$phi, b4$k, b4$tcr, phi = 0, theta=0, bty = "g", type = "h", ticktype = "detailed", pch = 1)
```

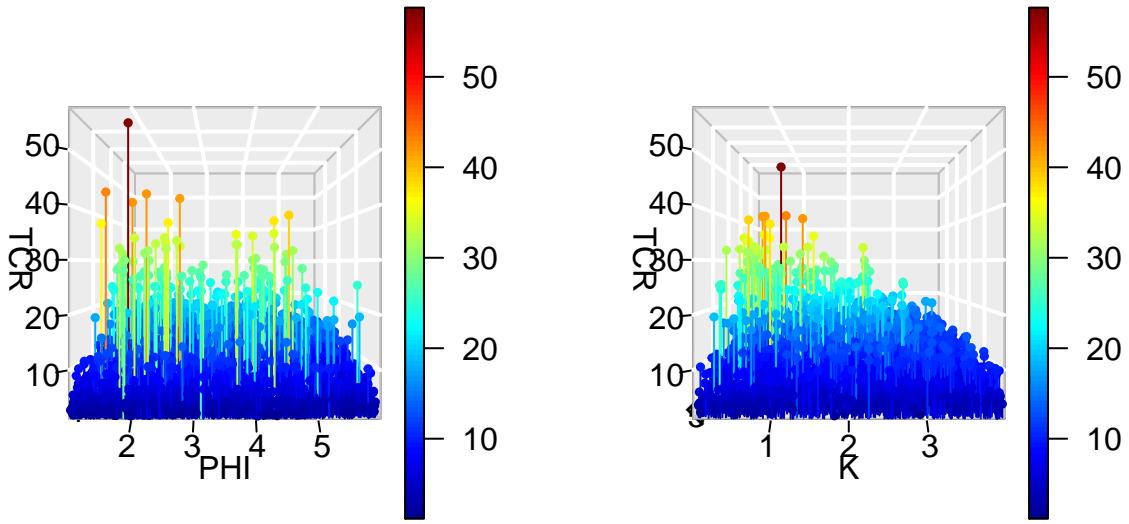


```
scatter3D(b4$phi, b4$k, b4$tcr, phi = 0, theta=90, bty = "g", type = "h", ticktype = "detailed", pch = 19)
```



Dins del dataset v ens han quedat els valors de les simulacions i el seu resultat de accuracy i de tcr calculats per cada fila. El que ens proposem a continuació és a realitzar un gràfic 3D per veure com es reparteixen els valors.

```
par(mfrow=c(1,2))
scatter3D(v$phi, v$k, v$tcr, phi = 0, theta=0, bty = "g", type = "h", ticktype = "detailed", pch = 19,
scatter3D(v$phi, v$k, v$tcr, phi = 0, theta=90, bty = "g", type = "h", ticktype = "detailed", pch = 19)
```

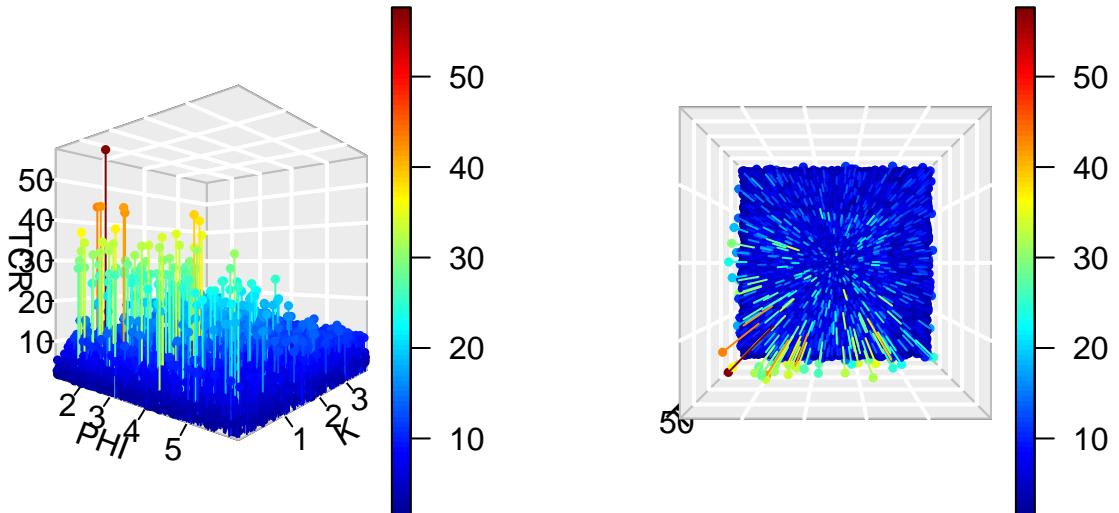


Al observar els gràfics ens podem adonar que tots els valors generats son molts que el nombre de punts està molt escampat si parlem del global de punts. Ara, si en comptes de fixar-nos en el normal funcionament de la funció ens fixem només en la seva maximització, és a dir estudiem els valors més bons i els seu comportament podem extreure les conclusions següents :

- Els valors de k més bons es concentren entre el 0 i el 1.5.
- Per el gràfic de phi podem observar 2 màxims locals (no podem afirmar que existeixi un màxim global o que l'haguem caçat).
- Dels dos màxims locals que s'intueixen un el podrem trobar en el rang de valors de 0 a 3 i l'altre entre 3.6 i 4.8 aproximadament.

per tenir una mica més de perspectiva mostrem els següents gràfics per veure el comportament del tcr i els valors de les dues variables a la vegada.

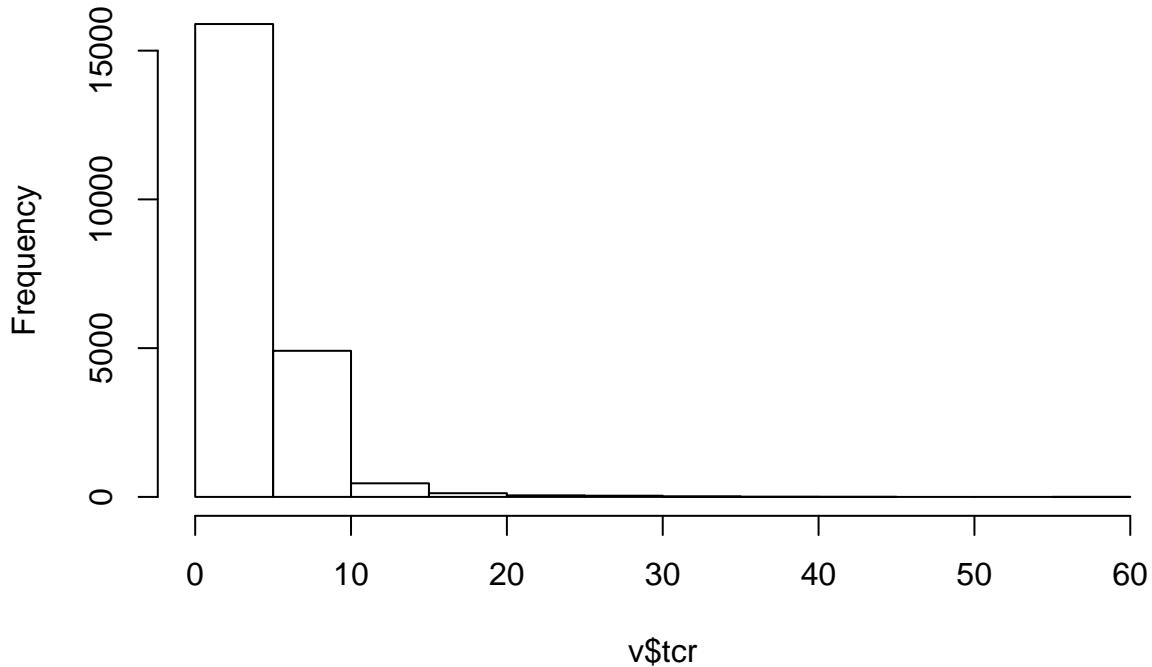
```
par(mfrow=c(1,2))
scatter3D(v$phi, v$k, v$tcr, phi = 0, bty = "g", type = "h", ticktype = "detailed", pch = 19, cex = 0.5)
scatter3D(v$phi, v$k, v$tcr, phi = 90, theta=0, bty = "g", type = "h", ticktype = "detailed", pch = 19, cex = 0.5)
```



Veiem l'histograma del valor del tcr en general abans de seguir estudiant.

```
hist(v$tcr)
```

Histogram of v\$tcr



Degut a que tenim una mostra molt gran i que els resultats més bons son molt pocs en funció del total no podem esperar molta millora ni concessió en els valors de phi i de k.

Estudiem ara les mitjanes agrupant les execucions per rangs de k.

```
k1 <- v[v$k <= 1, ]
k2 <- v[v$k > 1 & v$k <=2, ]
k3 <- v[v$k > 2 & v$k <=3, ]
k4 <- v[v$k > 3 & v$k <=4, ]
```

La conclusió d'aquest apartat és que creiem que per els valors de k tenim un marge de millora ampli a l'hora d'e 0'intentar acotar el rang d'actuació d'aquests valors.

```
#hist(head(d1$tcr, 20))

# Mitjana per el valor del tcr per phi i per k amb rang entre [1,5] i (0-3].
#median(d1$tcr)
```

El que es pretén és realitzar un estudi de quan les variables phi i k considerades com a constants en l'execució del programa es comporten de manera adient per el filtratge. En una primera iteració amb prop de 1500 execucions utilitzant sense lemmatitzar amb un nombre total de 20000 correus i generant una discriminació d'entre un 5% i un 15% per a la validació ens trobem el següent gràfic.

```
#scatter3D(d1$phi, d1$k, d1$tcr, phi = 0, bty = "g", type = "h", ticktype = "detailed", pch = 19, cex = 1.5)
```

Per tenir una idea de on queden encabits exactament els valors podem rotar el gràfic fins a aconseguir que es vegin on cauen més o menys.

```
#scatter3D(d1$phi, d1$k, d1$tcr, phi = 90, theta = 0.5, bty = "g", type = "h", ticktype = "detailed", pch = 19, cex = 1.5)
```

Veiem que amb uns valors aproximats d'entre 0 i 1 per k i entre 1.7 i 3.5 de phi, s'hi concentren els que tenen el tcr més alt. Per tant generem una altre iteració d'uns 1000 valors restringint aquest rang per phi i k en el quadrant on apareixen més aparicions dels valors de phi i de k concretament phi entre [1.7,2.3] i k entre

(0,0.30] i tornant a visualitzar el resultat.

```
# Generem el tcr per els nous valors.  
# BASE : (NSPAM) / (50 * NHAM + NSPAM)  
#basel <- resultsP1723K005$NSPAM / (50 * resultsP1723K005$NHAM + resultsP1723K005$NSPAM)  
# WERR: (50 * FP + FN)/(50 * NHAM + NSPAM) + 0.000001 -> per que no sigui 0  
#werrl <- (50 * resultsP1723K005$FP + resultsP1723K005$FN) / (50 * resultsP1723K005$NHAM + #resultsP1723K005$NSPAM)  
# TCR : BASE / WERR  
#tcrl <- basel/werrl  
  
#valueslimited <- data.frame(resultsP1723K005$PHI, resultsP1723K005$K, tcrl)  
#names(valueslimited) <- c("phi", "k", "tcr")  
  
# Ordenem els valors  
#valueslimited <- valueslimited[order(-valueslimited$tcr), ]  
#head(valueslimited,20)  
  
#median(tcrl)
```

Veiem que la mitjana ha augmentat respecte l'anterior però poc, altres indicadors que podem fer servir son per exemple el valor màxim trobat o fins i tot la mitjana dels 50 valors més alts.

A continuació comparem els valors per les dos rangs i observerem els histogrames dels 50 valors millors per les dues distribucions de resultats per veure si ens han aparegut valors més bons restringit el rang.

```
#median(head(values$tcr, 10))  
#median(head(valueslimited$tcr, 10))
```

Tenint en compte que les dades limitades son de 500 elements i les dades que tenen el rang més ampli son de 1500 i veient com es mantenen els valors més alts mirem de concretar més els resultats i si es pot ajustar més el rang dels valors generats per phi i k mitjançant els gràfics següents :

```
#scatter3D(resultsP1723K005$PHI, resultsP1723K005$K, tcrl, phi = 0, bty = "g", type = "h", ticktype = "detailed")  
#scatter3D(resultsP1723K005$PHI, resultsP1723K005$K, tcrl, phi = 90, theta = 0.5, bty = "g", type = "h")
```

Recol·lecció dels millors valors

Per ara sabem que ajustant el rang una mostra de 500 valors es comporta de manera similar que una mostra de 1500 valors ambdós generats aleatoriament tant per phi com per k, això ens fa pensar que aquest ajustament s'està comportant millor que el rang més ampli i que en part pot ser que haguem trobat indicis d'un màxim local de la relació de les dues variables.

Per seguir recollirem els millors valors de les dues execucions anteriors i mirarem de centrar-los en un sol dataset per treballar-lo. Considerarem que els millors valors per nosaltres son a partir del tcr 25.

```
#bestvals <- rbind(values$values$tcr > 25, ], valueslimited$values$tcr > 25, ])  
#bestvals <- bestvals[order(-bestvals$tcr), ]
```

Ara presentem l'histograma de les aparicions per phi i per k, intentant cercar encara aquest màxim local que creiem que existeix en aquesta franja.

```
#hist(bestvals$k)  
#hist(bestvals$phi)
```

Veient els resultats obtinguts i observant les aparicions i els valors de phi i k sobre el dataset bestvals s'ha decidit llençar una 3a tanda focalitzant els rangs per k i phi als valors d'entre [0.20,0.30] per k i de [1.8,2.8] per phi.

Focalització del valor de k.

Carreguem els resultats.

```
#resultsP1828K0203 = read.csv("/home/marc/projects/spamizer/analisys/20000m-500n-phi-18-28-k-020-030.csv")
```

Realitzem el mateix procediment que amb els valors anteriors. Cerquem el TCR i recollim els millors valors dins del dataframe de bestvalues.

```
# Generem el tcr per els nous valors.
# BASE : (NSPAM) / (50 * NHAM + NSPAM)
#base3 <- resultsP1828K0203$NSPAM / (50 * resultsP1828K0203$NHAM + resultsP1828K0203$NSPAM)
# WERR: (50 * FP + FN)/(50 * NHAM + NSPAM) + 0.000001 -> per que no sigui 0
#werr3 <- (50 * resultsP1828K0203$FP + resultsP1828K0203$FN) / (50 * resultsP1828K0203$NHAM + resultsP1828K0203$NSPAM)
# TCR : BASE / WERR
#tcr3 <- base3/werr3

# Observem la mitjana dels nous valors
#median(tcr3)

#valuesfocalk <- data.frame(resultsP1828K0203$PHI, resultsP1828K0203$K, tcr3)
#names(valuesfocalk) <- c("phi", "k", "tcr")

# Ordenem els valors
#valuesfocalk <- valuesfocalk[order(-valuesfocalk$tcr), ]
#head(valuesfocalk,20)

#bestvals <- rbind(bestvals, valuesfocalk[valuesfocalk$tcr > 25, ])
#bestvals <- bestvals[order(-bestvals$tcr), ]

#scatter3D(bestvals$phi, bestvals$k, bestvals$tcr, phi = 0, bty = "g", type = "h", ticktype = "detailed")
#scatter3D(bestvals$phi, bestvals$k, bestvals$tcr, phi = 90, theta = 0.5, bty = "g", type = "h", ticktype = "detailed")
```

Exemple de funció K

En el següent gràfic la grandària dels punts estipula quant de gran és l'error no desitjat, és a dir, quan un correu considerat **HAM es filtra com SPAM**. Als eixos hi podem veure els valors de phi i k utilitzats per a la validació. El percentatge de correus utilitzats sobre els 200 correus totals és d'entre 5% i 15% i la selecció d'aquest valor és aleatòria.

```
#head(results)

# De moment la millor opció.

#scatter3D(valores$phi, valores$k, valores$tcr, phi = 0, bty = "g", type = "h", ticktype = "detailed",
#d = ggplot(valores,aes(phi, k, fill=tcr)) + ggtile("Plot of 100 values") + xlab("PHI") + ylab("K") + theme_minimal()
#firstValues <- head(valores, 30)

#ggplot(data = valores, aes(x = phi, y = k)) + geom_tile(aes(fill = tcr))

#grid.arrange(p1,p2,ncol=2)
#heatmap(data.matrix(valores))
```

```
#radius <- sqrt(valores$tcr/pi)
#symbols(valores$phi, valores$k, circles = radius, inches = 0.1, fg = "white", bg = "red", main = "Size")
#plot(valores$tcr~sort(valores$k), type="l")
#line(valores$tcr~sort(valores$phi), col="red")
```

Referències

- R graphics