自然言語処理入門

岸山 健 (31-187002)

Dec. 3, 2018

課題

> 共起の強さの指標として自己相互情報量 (負の値は 0 とする) を用いて，果物名の (五次元！) を求めよ．

まず「イチゴ」と「切る」のペアから 自己相互情報量 (PMI) を考えてみる．そのためにはまずデータ全体と頻度の情報を読み込む．文書の数は 2000 万なので変数 N に格納し，その他の情報も適宜データフレームに入れる．

```
N <- 20000000
df <- data.frame(
    word =c( ' イチゴ', ' ナシ', ' ミカン', ' リンゴ'),
    切る=c(371,486,137,730),
    摘む=c(84,2,18,13),
    なる=c(59,75,29,154),
    むく=c(1,52,192,97),
    もぐ=c(1,10,2,14))

df.t <- t(df)
# the first row will be the header
colnames(df.t) <- df.t[1, ]; df.t <- df.t[-1 ,]
# このままだと数値の char 型なので int に変更
df.t <- as.data.frame(df.t)
df.t
#       イチゴ ナシ ミカン リンゴ
# 切る    371  486    137   730
# 摘む     84    2     18    13
# なる     59   75     29   154
# むく      1   52    192    97
# もぐ      1   10      2    14

freq <- data.frame(
    word= c("切る","摘む","なる","むく","もぐ",
```

```
            "イチゴ","ナシ","ミカン","リンゴ"),
    n = c(26600,5070,8090,3340,2160,8360,17440,12420,24500))
#     word     n
# 1   切る 26600
# 2   摘む  5070
# 3   なる  8090
# 4   むく  3340
# 5   もぐ  2160
# 6 イチゴ  8360
# 7   ナシ 17440
# 8 ミカン 12420
# 9 リンゴ 24500
```

まず P(イチゴ, 切る) を P(イチゴ) と P(切る) の積で除算し，その値に対し底が 2 の対数をとった値を I(イチゴ, 切る) とする．P(イチゴ) と P(切る) の積 (分母) は「イチゴ」と「切る」が独立であった場合の値である．他方，P(イチゴ, 切る) は実際に共起した割合である．相互に関連があるほど独立ではなくなり， P(イチゴ, 切る) の値は上がる．したがって，I(イチゴ, 切る) は 「イチゴ」と「切る」がお互いにどれほど独立していないか，の指標となる．

```
# N <- 20,000,000
i.v.w <- function(v,N,freq) function(w){
    # P(v,w): ややこしいが df.t の w 行 v 列を参照し，char 型を num 型に変換
    p.v.w <- as.numeric(as.character(df.t[w,v]))/N
    p.v <- subset(freq,word==v)$"n" / N
    p.w <- subset(freq,word==w)$"n" / N
    value <- log2(p.v.w/(p.v*p.w))
    ifelse(value>0,value,0)
    }
i.v.w("イチゴ",N,freq)("切る")
# [1] 5.060346
```

「イチゴ」と各動詞 "切る" "摘む" "なる" "むく" "もぐ" の自己相互情報量を求めるために，各動詞を上の関数 i.v.w にマップする．そしてその結果が果物名の分散表現 (五次元) となる．

```
rownames(df.t)
# [1] "切る" "摘む" "なる" "むく" "もぐ"
イチゴ <- sapply(rownames(df.t),i.v.w("イチゴ",N,freq))
#      切る      摘む      なる      むく      もぐ
# 5.0603462 5.3087568 4.1249285 0.0000000 0.1473938
ナシ   <- sapply(rownames(df.t),i.v.w("ナシ",N,freq))
#     切る     摘む     なる     むく     もぐ
# 4.389058 0.000000 3.410279 4.158192 2.408497
ミカン <- sapply(rownames(df.t),i.v.w("ミカン",N,freq))
#      切る      摘む      なる      むく      もぐ
# 3.0520126 2.5152741 2.5291761 6.5324492 0.5763035
リンゴ <- sapply(rownames(df.t),i.v.w("リンゴ",N,freq))
```

```
#      切る       摘む       なる       むく       もぐ
# 4.485617 1.065672 3.957865 4.567283 2.403542
```

最後に類似度を求める関数 Sim.c にマップするとリンゴ–他のフルーツ間の類似度が分かる．結果，順序はナシ (0.99)>ミカン (0.90)>イチゴ (0.67) となった．ナシもミカンもむくが，イチゴはむかず，イチゴを摘む，という表現はあるがその他のフルーツではあまり聞かない．こうした直感にも合う結果である．

```
sapply(list(イチゴ, ナシ, ミカン),Sim.c(リンゴ))
# [1] 0.6712413 0.9895714 0.8958742
```

```
# Sim.c 関数について
# l と r の内積を返す中置関数を定義
'%ip%' <- function (l,r) {
    # 入力のベクトル l,r を列に格納
    tmp <- data.frame(l=l,r=r)
    # l.r という列名の各列に l*r を格納
    tmp$l.r <- tmp$l * tmp$r
    # l.r の和を取る
    sum(tmp$l.r)}


# ベクトル (Ws) の長さを返す関数を定義
D <- function(Ws) sqrt(sum(Ws ** 2))

# 第一引数に q(uestion), 第二引数に d(ocument vector)
Sim.c <- function(q) function(d) q %ip% d / (D(d) * D(q))
```