自然言語処理入門

岸山 健 (31-187002)

Oct. 29, 2018

1　課題 1

まず与えられた文書 1–6 に含まれる単語の単語頻度 tf をデータフレーム df に与える[*1]．そして各文書ごとの類似度を tf 値を用いて求める．

```
df <- data.frame(
#    word =c( 'ウィルス', 'エイズ ', '肝炎', '感染', 'コンピュータ',
#             '被害','ファイル','米国','メール'),
    tf.d1=c(4,0,0,3,1,1,0,1,2),
    tf.d2=c(3,0,0,3,1,1,1,0,2),
    tf.d3=c(2,0,0,1,2,1,2,0,0),
    tf.d4=c(2,1,1,2,0,1,0,3,0),
    tf.d5=c(1,0,2,2,0,1,0,0,0),
    tf.d6=c(1,3,0,3,0,1,0,0,0))
```

類似度を求めるために 各文書から 2 つ選択する組み合わせを docs.C.2 リストとして作成する． そして docs.C.2 の V1 と V2 が示す文書のベクトルを先ほどの文書ベクトルから取得し，前回の課題で作成したコサイン類似度を求める関数に適用して類似度を求め，sim という列に格納する．

```
library(dplyr)
library(purrr)

# 組み合わせの行列を作る
docs.C.2 <- as.data.frame(t(combn(colnames(df), 2)))
docs.C.2$sim = 1:nrow(docs.C.2) %>% map_dbl(function(i)
             Sim.c(unlist(select(df,as.character(docs.C.2[i,]$V1))) ,
                   unlist(select(df,as.character(docs.C.2[i,]$V2)))))
docs.C.2
##       V1    V2       sim
## 1  tf.d1 tf.d2 0.9545942
## 2  tf.d1 tf.d3 0.6614378
## 3  tf.d1 tf.d4 0.7115125
```

[*1] 回答の一部には GNU R を用いた．

```
## 4  tf.d1 tf.d5 0.6149187
## 5  tf.d1 tf.d6 0.5533986
## 6  tf.d2 tf.d3 0.7483315
## 7  tf.d2 tf.d4 0.5813777
## 8  tf.d2 tf.d5 0.6324555
## 9  tf.d2 tf.d6 0.5813777
## 10 tf.d3 tf.d4 0.4183300
## 11 tf.d3 tf.d5 0.4225771
## 12 tf.d3 tf.d6 0.3585686
## 13 tf.d4 tf.d5 0.6363961
## 14 tf.d4 tf.d6 0.6000000
## 15 tf.d5 tf.d6 0.5656854
```

```
docs.C.2[docs.C.2$sim==(docs.C.2$sim %>% max),]
```

上のデータフレームを表にすると以下のとおりとなる．最も似ているのは文書 1 と文書 2 である．

	tf.d2	tf.d3	tf.d4	tf.d5	tf.d6
tf.d1	0.954	0.661	0.711	0.614	0.553
tf.d2		0.748	0.581	0.632	0.581
tf.d3			0.418	0.422	0.358
tf.d4				0.636	0.600
tf.d5					0.565

```
docs.C.2[docs.C.2$sim!=(docs.C.2$sim %>% max),]
```

1.1　前回つくったコサイン類似度を求める関数

```
# 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))

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

以上の関数の動作を確認する．講義内で扱われた検索質問<2,1,3,0>を q0，対象となった文書のベクトル<4,3,0,5>を q0 とする．それらを関数に入れることで余弦尺度が求まる．なお，単語の情報は各列の index からアクセスできる．

```
# 文書 1 = <2,1,3,0> と 質問=<4,3,0,5> で動作確認
d0 <- c(0,0,3,3)
q0 <- c(4,3,0,5)

Sim.c(q0)(d0)
## [1] 0.4157609
```

最初の検索質問は<薬:1.0, 風邪:3.0, 熱:2.0>であった． まずはこの質問を q1 に格納し，Sim.c.q1 にこの質問との類似度を返す関数を格納する．

```
# 検索質問: 薬，風邪，熱
q1 = c(1,3,2,0,0,0)

# q1 との類似度を返す関数
Sim.c.q1 <- Sim.c(q1)

# 1 薬，2 風邪，3 熱，4 のど，5 胃，6 消化のベクトル
df $ tf.idf.1
## [1] 2.088624 6.521776 0.000000 3.568636 0.000000 0.000000
```

データフレーム df の tf.idf.1 には文書 1 のベクトルが入っているので，これを Sim.c.q1 に与えれば q1 と文書 1 の類似度が求まる．それを df.sim という表に格納する

```
df.sim <- data.frame(
    d1 = Sim.c.q1(df $ tf.idf.1) ,
    d2 = Sim.c.q1(df $ tf.idf.2) ,
    d3 = Sim.c.q1(df $ tf.idf.3) ,
    d4 = Sim.c.q1(df $ tf.idf.4) ,
    d5 = Sim.c.q1(df $ tf.idf.5) )

df.sim
#          d1       d2        d3        d4          d5
# 1 0.7494399 0.955446 0.4074928 0.1164894 0.02758927

df.sim[order(df.sim[1,],decreasing=T)]
#         d2        d1        d3        d4          d5
# 1 0.955446 0.7494399 0.4074928 0.1164894 0.02758927
```

表に格納した後は降順にして出力する． その結果，文書 2, 文書 1, 文書 3, そして文書 4,5 の順で 似ていたことが分かる．

次の検索質問は<薬:1.0, 胃:2.0>であったので，同様の手順を繰り返す．

```
# 検索質問: 薬 (index:1)，胃 (index:5)
```

```
q2 = c(1,0,0,0,2,0)
# q1 との類似度を返す関数
Sim.c.q2 <- Sim.c(q2)
```

そして各文書ベクトルを Sim.c.q2 に与えれば q2 との類似度が求まる．それを df.sim という表に格納する

```
df.sim <- data.frame(
    d1 = Sim.c.q2(df $ tf.idf.1) ,
    d2 = Sim.c.q2(df $ tf.idf.2) ,
    d3 = Sim.c.q2(df $ tf.idf.3) ,
    d4 = Sim.c.q2(df $ tf.idf.4) ,
    d5 = Sim.c.q2(df $ tf.idf.5) )

df.sim
#          d1        d2        d3        d4        d5
# 1 0.1209592 0.1280726 0.1884064 0.8580712 0.2555781

df.sim[order(df.sim[1,],decreasing=T)]
#          d4        d5        d3        d2        d1
# 1 0.8580712 0.2555781 0.1884064 0.1280726 0.1209592
```

ソートの結果，文書 4, 文書 5, 文書 3, 文書 2, 文書 1 の順で類似度が検索質問 2 に近いことが分かった．