Fuzzy Logic Induced Content-Based Recommendation

Nguyen Quy Khoi

June 01, 2023

Contents

1 Introduction to content-based recommendation

When a friend comes to you for a movie recommendation, you don't arbitrarily start shooting movie names. You try to suggest movies while keeping in mind your friend's tastes. Content-based recommendation systems, also known as content-based filtering methods, try to mimic the exact same process.

Consider a scenario in which a user is browsing through a list of products. Given a set of products and the associated product properties, when a person views a particular product, content-based recommendation systems can generate a subset of products with similar properties to the one currently being viewed by the user.

The following example demonstrates how content-based recommendation systems work.

1.1 Introduction to wine suggestion system

An example of using content-based recommendation is wine suggestion. We will use the wine data set from UCI Machine Learning Repository. This data set is the result of the chemical analysis of wine grown in the same region in Italy. We have data from three different cultivars (from an assemblage of plants selected for desirable characters).

In the data set, rows represent each wine brand, while the columns represent the properties of the wines. The table is extracted from UCI machine learning repository as follows:

```
# data.table provides a high-performance version of base R's data.frame with
# syntax and feature enhancements for ease of use, convenience and
# programming speed.
library(data.table)
library(here); library(skimr); library(janitor)
# differences between "<-" and "="
# "<-" is for assignment, standard everywhere
# "=" is for passing arguments, (allowed only for top level env)
#wine.data <- fread('https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wir
wine.data <- fread('data\\wine.data')
skim_without_charts(as.data.frame(wine.data))
```

Table 1: Data summary

Name	as.data.frame(wine.data)
Number of rows	178
Number of columns	14
_____	
Column type frequency:	
numeric	14
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
V1	0	1	1.94	0.78	1.00	1.00	2.00	3.00	3.00
V2	0	1	13.00	0.81	11.03	12.36	13.05	13.68	14.83
V3	0	1	2.34	1.12	0.74	1.60	1.87	3.08	5.80
V4	0	1	2.37	0.27	1.36	2.21	2.36	2.56	3.23
V5	0	1	19.49	3.34	10.60	17.20	19.50	21.50	30.00
V6	0	1	99.74	14.28	70.00	88.00	98.00	107.00	162.00
V7	0	1	2.30	0.63	0.98	1.74	2.36	2.80	3.88
V8	0	1	2.03	1.00	0.34	1.20	2.13	2.88	5.08
V9	0	1	0.36	0.12	0.13	0.27	0.34	0.44	0.66
V10	0	1	1.59	0.57	0.41	1.25	1.56	1.95	3.58
V11	0	1	5.06	2.32	1.28	3.22	4.69	6.20	13.00
V12	0	1	0.96	0.23	0.48	0.78	0.96	1.12	1.71
V13	0	1	2.61	0.71	1.27	1.94	2.78	3.17	4.00
V14	0	1	746.89	314.91	278.00	500.50	673.50	985.00	1680.00

1.2 Chemical properties similarity between two wines

We have a total of 14 columns. The first column `V1` represents the cultivar, which is assigned in `wine.type`. The remaining columns are the chemical properties of the wine, which are stored in `wine.features`:

```
wine.type <- wine.data[,1]
wine.features <- wine.data[,-1]
# scale feature properties such that each column has mean=0 and std=1
wine.features.scaled <- scale(wine.features)
# convert feature properties into matrix to perform linear algebra operations
wine.mat <- data.matrix(wine.features.scaled)
# assign row names for matrix wine.mat (name = sequence from 1 to 178)
rownames(wine.mat) <- seq(1:dim(wine.features.scaled)[1])
# transpose the matrix, feature properties are row, wines are column
# to perform pearson coefficient calculation between 2 columns
wine.mat <- t(wine.mat)
```

The pairwise correlation in the numerical matrix is measured using Pearson coefficient after scaling. In this problem, the Pearson coefficient matrix returns *the similarity between 2 wines* (i.e., compare 2 columns).

The output is the similarity matrix, which shows how closely related items are. The values range from -1 for perfect negative correlation, when two items have attributes that move in opposite directions, and +1 for perfect positive correlation, when attributes for the two items move in the same direction. The diagonal values will be +1, as we are comparing a wine to itself.

```
# perform pearson correlation calculation, which returns an 178x178 matrix
cor.matrix <- cor(wine.mat, use="pairwise.complete.obs", method="pearson")
```

1.3 Test case: Find top 4 similar wines to wine 3

In this problem, the similarity matrix is useful for finding similar wines (target column) in the wine's catalog. For example, if a customer likes wine 3 and wants to know if the company's list of wines has something similar. Using the similarity matrix, we will look at the third row to find any wines having high correlation value (i.e., column value closer to 1). **The top 4 matches** of wine 3 in terms of chemical properties are wines 52, 51, 85, and 15. The chemical properties of the matching wines are shown below:

```
sim.items <- cor.matrix[3,]
# find the closest match by sorting row 3 in decreasing order
sim.items.sorted <- sort(sim.items, decreasing=TRUE)
# take out top 5 matches
sim.items.name <- names(sim.items.sorted[2:5])
rbind(wine.data[as.double(sim.items.name),])
```

```
##    V1    V2   V3   V4   V5  V6   V7   V8   V9  V10  V11  V12  V13  V14
## 1:  1 13.83 1.65 2.60 17.2  94 2.45 2.99 0.22 2.29 5.60 1.24 3.37 1265
## 2:  1 13.05 1.73 2.04 12.4  92 2.72 3.27 0.17 2.91 7.20 1.12 2.91 1150
## 3:  2 11.84 0.89 2.58 18.0  94 2.20 2.21 0.22 2.35 3.05 0.79 3.08  520
## 4:  1 14.38 1.87 2.38 12.0 102 3.30 3.64 0.29 2.96 7.50 1.20 3.00 1547
```

2 Case study: News aggregator

2.1 Problem statement

We have 1,000 news articles from different publishers in different categories: technical, entertainment, etc. The problem is building a recommendation system for anonymous customers. The customers either are first time visitors or we have not recorded their interaction with our products.

```
#
library(dplyr)
library(ggplot2)
library(sentimentr); library(sets); library(slam)
library(tidytext); library(tidyverse); library(tm)

data.population <- read_tsv('data\\NewsAggregatorDataset\\newsCorpora.csv',
                   col_names=c('ID','TITLE','URL','PUBLISHER','CATEGORY',
                               'STORY','HOSTNAME','TIMESTAMP'),
                   col_types=cols(
                     ID       =col_integer(),
                     TITLE    =col_character(),
                     URL      =col_character(),
                     PUBLISHER=col_character(),
                     CATEGORY =col_character(),
                     STORY    =col_character(),
                     HOSTNAME =col_character(),
```

```
                      TIMESTAMP=col_double()))

# sample without replacement 40,000 random data in the whole data set
index.sampled <- sample(1:nrow(data.population), 40000, replace=FALSE)
data.sample <- data.population[index.sampled,]
```

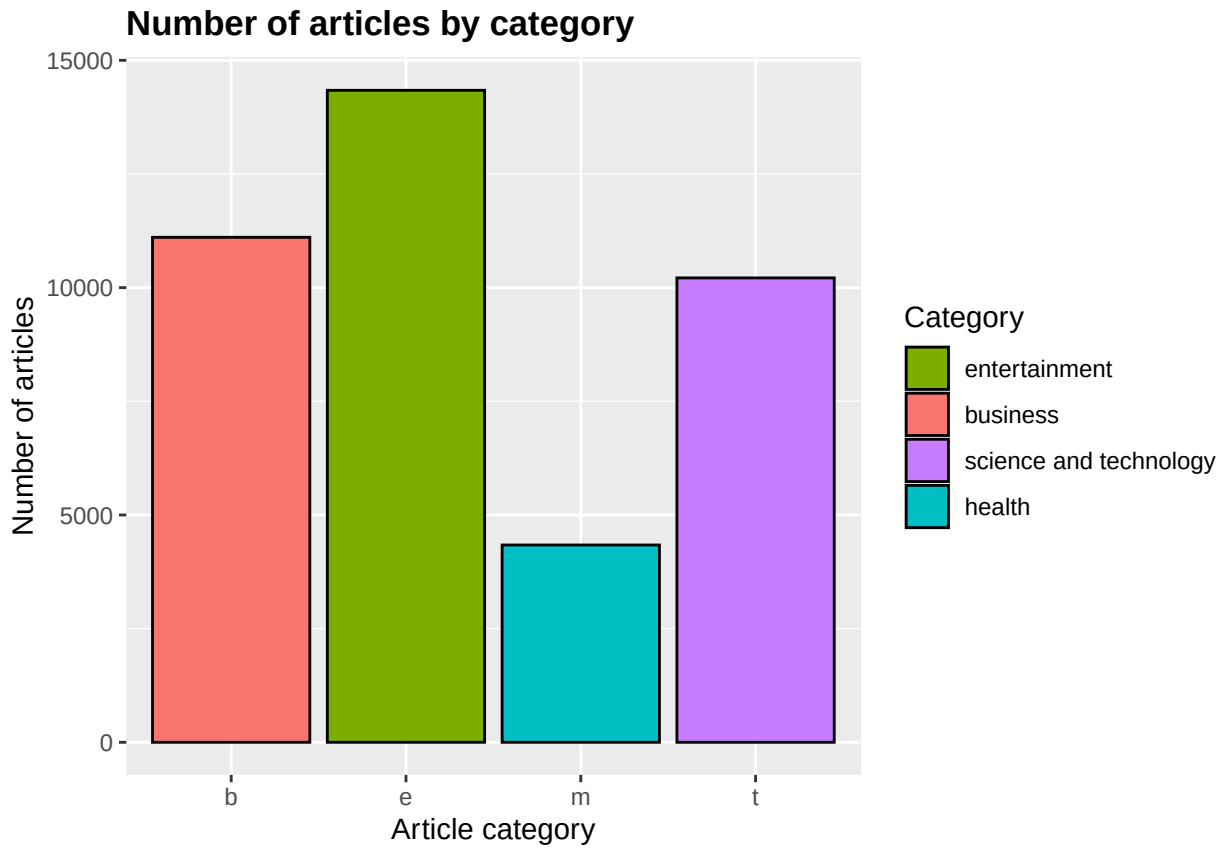
Every article has the following columns:

- `ID`: A unique identifier.
- `TITLE`: The title of the article (free text).
- `URL`: The article's URL.
- `PUBLISHER`: Publisher of the article.
- `CATEGORY`: Categorization under which the articles are grouped (`b` = business, `t` = science and technology, `e` = entertainment, `m` = health).
- `STORY`: An ID for the group of stories the article belongs to.
- `HOSTNAME`: Hostname of the URL.
- `TIMESTAMP`: Approximate time the news was published, as the number of milliseconds since the epoch 00:00:00 GMT, January 1, 1970.

2.2 Exploratory Data Analysis

Firstly, the articles are grouped by category (as specified by `CATEGORY`) and arranged in a descending order. From observation, entertainment has the highest number of published articles.

```
ggplot(data.sample, mapping=aes(x=CATEGORY, fill=factor(CATEGORY))) +
  geom_bar(colour='black',stat='count') +
  ggtitle("Number of articles by category") +
  xlab("Article category") +
  ylab("Number of articles") +
  scale_fill_discrete(name='Category',
                      breaks=c('e','b','t','m'),
                      labels=c("entertainment","business",
                               "science and technology","health")) +
  theme(plot.title=element_text(lineheight=1, face="bold"))
```



Secondly, we order the publishers in descending order by the number of published articles. `Reuters` publishes the most articles, followed by `Businessweek`. Then, all articles of the top 100 publishers are obtained from `newsCorpora.csv` to design content-based recommendation engine.

```
publisher.top <- data.sample |> group_by(PUBLISHER) |> summarise(count=n()) |>
                          arrange(desc(count), PUBLISHER)
publisher.top
```

```
## # A tibble: 5,450 x 2
##    PUBLISHER         count
##    <chr>             <int>
##  1 Reuters             351
##  2 Businessweek        236
##  3 Huffington Post     236
##  4 Daily Mail          230
....
##  9 GlobalPost          182
## 10 RTT News            180
## # i 5,440 more rows
```

```
# get top 100 publishers, join with article tables
data.subset <- inner_join(data.frame(publisher.top[1:100,]), data.sample)
```

2.3 Content-based recommendation engine design

The problem statement from customer requirements is: **When a customer browses a particular article, what other articles should we suggest to him?** From the statement, some properties of the news articles must be analyzed to provide a similar content to the one the user is reading. Those properties are:

- The article's content.
- The article's publisher.
- The article's category.

2.4 Design procedure

Cosine distance compares words between two documents.

2.4.1

Jaccard's distance measures the Jaccard's publishers and categories

1. Calculate **cosine distance**. The similarity matrix's dimension is $R^{N\times N}$, with N is the number of rows/columns. In this problem, the rows/columns are the 40,000 articles in the sampled data set. The similarity score is an entry in the matrix, ranging from 0 to 1. The score 0 indicates there is no relationship between two articles, while the score 1 means the two articles are an exact replica of each other.
2. Search for relevant news articles in the similarity index. Given an article,
3. Implement a fuzzy ranking engine. The ranking engine finds the top 10 matches in a ranked order using the cosine distance, Jaccard distance, and Manhattan distance.

The articles list separated into two data frames: `title.df` and `others.df`. Data frame `title.df` stores article titles while `others.df` stores articles' ID, publisher, and category.

```
######### Cosine Similarity ######################
title.df  <- data.subset[,c('ID','TITLE')]
others.df <- data.subset[,c('ID','PUBLISHER','CATEGORY')]
corpus <- title.df |> rename(all_of(c(doc_id='ID',text='TITLE'))) |>
            DataframeSource() |> Corpus()
corpus <- corpus |> tm_map(removePunctuation) |> tm_map(removeNumbers) |>
       tm_map(stripWhitespace) |> tm_map(content_transformer(tolower)) |>
       tm_map(removeWords, stopwords("english"))
dtm <- DocumentTermMatrix(corpus, control=list(wordLength=c(3,10),
                                               weighting="weightTfIdf"))
dtm
```

```
## <<DocumentTermMatrix (documents: 10802, terms: 13501)>>
## Non-/sparse entries: 74979/145762823
## Sparsity           : 100%
## Maximal term length: 22
```

```
inspect(dtm[1:5,10:15])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 6)>>
## Non-/sparse entries: 7/23
## Sparsity           : 77%
## Maximal term length: 12
## Sample             :
##         Terms
## Docs     bnp china fine france paribas unreasonable
....
##   41643    0     0    0      0       0            0
##   91006    0     1    0      0       0            0
##   91477    0     0    0      0       0            0
```

```
sim.score <- tcrossprod_simple_triplet_matrix(dtm) /
  (sqrt(row_sums(dtm^2) %*% t(row_sums(dtm^2))))
match.docs <- sim.score[1,]
match.docs
```

```
##      91477     267061      91006     402213      41643      59934      47357
## 1.00000000 0.13363062 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##     399262     327934     332464     104978       4639     307736     124838
## 0.13363062 0.00000000 0.00000000 0.00000000 0.13363062 0.00000000 0.00000000
##     149249     297206      41315     217155     146927     105128      70545
## 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##     192682     139609     367054     308689     121799      41755     383958
....
## 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##     368205
## 0.00000000
```

```
match.df <- data.frame(ID = names(match.docs), cosine=match.docs,
                       stringsAsFactors=FALSE)
match.df$ID <- as.integer(match.df$ID)
match.df
```

```
##            ID     cosine
## 91477   91477 1.00000000
## 267061 267061 0.13363062
## 91006   91006 0.00000000
## 402213 402213 0.00000000
## 41643   41643 0.00000000
## 59934   59934 0.00000000
....
## 413426 413426 0.00000000
## 356093 356093 0.00000000
```

```
## 368205 368205 0.00000000
```

```
match.refined <- head(match.df[order(-match.df$cosine),],30)
match.refined
```

```
##            ID    cosine
## 91477   91477 1.0000000
## 226424 226424 0.5345225
## 226104 226104 0.4008919
## 188909 188909 0.3535534
## 192643 192643 0.3535534
## 255932 255932 0.2672612
....
## 180580 180580 0.2041241
## 51063   51063 0.2041241
## 405854 405854 0.1767767
```

```
################## Polarity score ###############

match.refined <- inner_join(match.refined, title.df)
```

```
## Joining with `by = join_by(ID)`
```

```
match.refined <- inner_join(match.refined, others.df)
```

```
## Joining with `by = join_by(ID)`
```

```
head(match.refined)
```

```
##       ID    cosine
## 1  91477 1.0000000
## 2 226424 0.5345225
## 3 226104 0.4008919
## 4 188909 0.3535534
## 5 192643 0.3535534
## 6 255932 0.2672612
##                                                                      TITLE
## 1              American Airlines says weather hurt first quarter results
## 2 First quarter results were 'slightly below our expectations,' TJX says
## 3                     Home Depot first quarter results miss expectations
## 4                                U.S. Productivity Falls in First Quarter
## 5                Emirates airlines says profit up 43% in 2013 to $887 mn
## 6        Home Price Rises Cool Further in Quarter, says S&P/Case-Shiller
##      PUBLISHER CATEGORY
## 1      Reuters        b
## 2 Boston Globe        b
## 3     CBS News        b
## 4       NASDAQ        b
```

```
## 5   GlobalPost        b
## 6       NASDAQ        b
```

```
sentiment.score <- sentiment(match.refined$TITLE)
head(sentiment.score)
```

```
##    element_id sentence_id word_count   sentiment
## 1:          1           1          8 -0.17677670
## 2:          2           1         10  0.07905694
## 3:          3           1          7 -0.28347335
## 4:          4           1          2  0.00000000
## 5:          4           2          5  0.33541020
## 6:          5           1          8  0.00000000
```

```
sentiment.score <- sentiment.score |> group_by(element_id) |>
  summarise(sentiment = mean(sentiment))
head(sentiment.score)
```

```
## # A tibble: 6 x 2
##   element_id sentiment
##        <int>     <dbl>
## 1          1   -0.177
## 2          2    0.0791
## 3          3   -0.283
## 4          4    0.168
## 5          5    0
## 6          6    0.217
```

```
match.refined$polarity <- sentiment.score$sentiment
head(match.refined)
```

```
##       ID    cosine
## 1  91477 1.0000000
## 2 226424 0.5345225
## 3 226104 0.4008919
## 4 188909 0.3535534
## 5 192643 0.3535534
## 6 255932 0.2672612
##                                                                    TITLE
## 1              American Airlines says weather hurt first quarter results
## 2 First quarter results were 'slightly below our expectations,' TJX says
## 3                     Home Depot first quarter results miss expectations
## 4                                U.S. Productivity Falls in First Quarter
## 5                Emirates airlines says profit up 43% in 2013 to $887 mn
## 6        Home Price Rises Cool Further in Quarter, says S&P/Case-Shiller
##      PUBLISHER CATEGORY    polarity
## 1      Reuters        b -0.17677670
```

```
## 2 Boston Globe        b  0.07905694
## 3     CBS News        b -0.28347335
## 4       NASDAQ        b  0.16770510
## 5   GlobalPost        b  0.00000000
## 6       NASDAQ        b  0.21650635
```

```
#help("sentiment")

target.publisher <- match.refined[1,]$PUBLISHER
target.category <- match.refined[1,]$CATEGORY
target.polarity <- match.refined[1,]$polarity
target.title <- match.refined[1,]$TITLE


#match.refined <- match.refined[-1,]
match.refined$is.publisher <- match.refined$PUBLISHER == target.publisher
match.refined$is.publisher <- as.numeric(match.refined$is.publisher)

match.refined$is.category <- match.refined$CATEGORY == target.category
match.refined$is.category <- as.numeric(match.refined$is.category)

# Calculate Jaccards
match.refined$jaccard <- (match.refined$is.publisher + match.refined$is.category)/2
match.refined$polaritydiff <- abs(target.polarity - match.refined$polarity)

range01 <- function(x){(x-min(x))/(max(x)-min(x))}
match.refined$polaritydiff <- range01(unlist(match.refined$polaritydiff))


head(match.refined)
```

```
##       ID    cosine
## 1  91477 1.0000000
## 2 226424 0.5345225
## 3 226104 0.4008919
## 4 188909 0.3535534
## 5 192643 0.3535534
## 6 255932 0.2672612
##                                                                  TITLE
## 1              American Airlines says weather hurt first quarter results
## 2 First quarter results were 'slightly below our expectations,' TJX says
## 3                     Home Depot first quarter results miss expectations
## 4                              U.S. Productivity Falls in First Quarter
## 5                Emirates airlines says profit up 43% in 2013 to $887 mn
## 6        Home Price Rises Cool Further in Quarter, says S&P/Case-Shiller
```

```
##      PUBLISHER CATEGORY    polarity is.publisher is.category jaccard
## 1      Reuters        b -0.17677670            1           1     1.0
## 2 Boston Globe        b  0.07905694            0           1     0.5
## 3     CBS News        b -0.28347335            0           1     0.5
## 4       NASDAQ        b  0.16770510            0           1     0.5
## 5   GlobalPost        b  0.00000000            0           1     0.5
## 6       NASDAQ        b  0.21650635            0           1     0.5
##   polaritydiff
## 1    0.0000000
## 2    0.3476975
## 3    0.1450089
## 4    0.4681772
## 5    0.2402531
## 6    0.5345018
```

```
## clean up
match.refined$is.publisher = NULL
match.refined$is.category = NULL
match.refined$polarity = NULL
match.refined$sentiment = NULL


head(match.refined)
```

```
##       ID    cosine
## 1  91477 1.0000000
## 2 226424 0.5345225
## 3 226104 0.4008919
## 4 188909 0.3535534
## 5 192643 0.3535534
## 6 255932 0.2672612
##                                                                    TITLE
## 1              American Airlines says weather hurt first quarter results
## 2 First quarter results were 'slightly below our expectations,' TJX says
## 3                     Home Depot first quarter results miss expectations
## 4                                U.S. Productivity Falls in First Quarter
## 5                Emirates airlines says profit up 43% in 2013 to $887 mn
## 6        Home Price Rises Cool Further in Quarter, says S&P/Case-Shiller
##      PUBLISHER CATEGORY jaccard polaritydiff
## 1      Reuters        b     1.0    0.0000000
## 2 Boston Globe        b     0.5    0.3476975
## 3     CBS News        b     0.5    0.1450089
## 4       NASDAQ        b     0.5    0.4681772
## 5   GlobalPost        b     0.5    0.2402531
## 6       NASDAQ        b     0.5    0.5345018
```

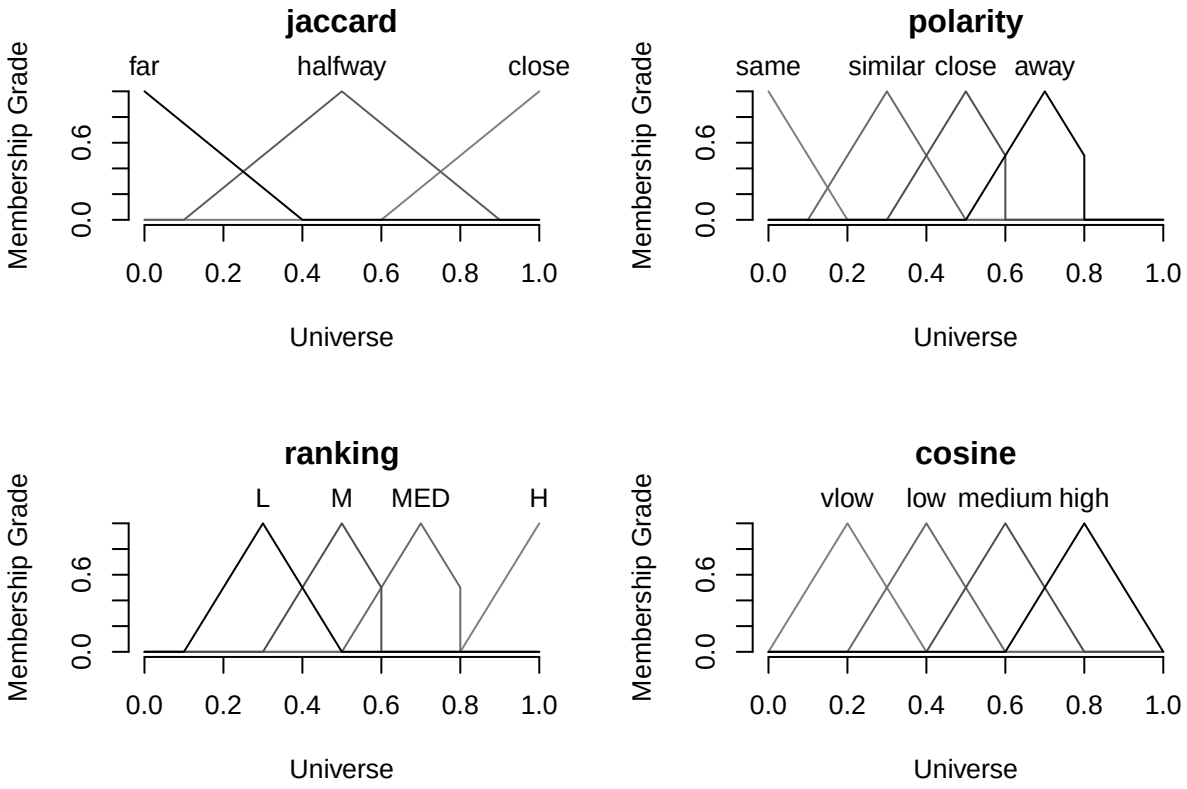
```
### Fuzzy Logic ########

sets_options("universe", seq(from=0, to=1, by=0.1))
variables <-
  set(cosine=
      fuzzy_partition(varnames=c(vlow=0.2, low=0.4, medium=0.6, high=0.8),
                          FUN=fuzzy_cone, radius=0.2),
      jaccard=
      fuzzy_partition(varnames=c(close=1.0, halfway=0.5, far=0.0),
                          FUN=fuzzy_cone , radius=0.4),
      polarity=
      fuzzy_partition(varnames=c(same=0.0, similar=0.3, close=0.5, away=0.7),
                          FUN = fuzzy_cone , radius = 0.2),
      ranking =
      fuzzy_partition(varnames=c(H=1.0, MED=0.7, M=0.5, L=0.3),
                          FUN = fuzzy_cone , radius = 0.2)
  )

rules <-
  set(
    ######### Low Ranking Rules ###################

  fuzzy_rule(cosine %is% vlow,
               ranking %is% L),
  fuzzy_rule(cosine %is% low || jaccard %is% far || polarity %is% away,
               ranking %is% L),
  fuzzy_rule(cosine %is% low || jaccard %is% halfway || polarity %is% away,
               ranking %is% L),
  fuzzy_rule(cosine %is% low || jaccard %is% halfway || polarity %is% close,
               ranking %is% L),
  fuzzy_rule(cosine %is% low || jaccard %is% halfway || polarity %is% similar,
               ranking %is% L),
  fuzzy_rule(cosine %is% low || jaccard %is% halfway || polarity %is% same,
               ranking %is% L),
  fuzzy_rule(cosine %is% medium || jaccard %is% far || polarity %is% away,
               ranking %is% L),

    ############### Medium Ranking Rules #################

  fuzzy_rule(cosine %is% low || jaccard %is% close || polarity %is% same,
               ranking %is% M),
  fuzzy_rule(cosine %is% low && jaccard %is% close && polarity %is% similar,
               ranking %is% M),
```

```
    ############### Median Ranking Rule ##################

  fuzzy_rule(cosine %is% medium && jaccard %is% close && polarity %is% same,
             ranking %is% MED),
  fuzzy_rule(cosine %is% medium && jaccard %is% halfway && polarity %is% same,
             ranking %is% MED),
  fuzzy_rule(cosine %is% medium && jaccard %is% close && polarity %is% similar,
             ranking %is% MED),
  fuzzy_rule(cosine%is% medium && jaccard %is% halfway && polarity %is% similar,
             ranking %is% MED),

    ############## High Ranking Rule ####################

  fuzzy_rule(cosine %is% high,
             ranking %is% H)
  )

ranking.system <- fuzzy_system(variables, rules)
print(ranking.system)
```

```
## A fuzzy system consisting of 4 variables and 14 rules.
##
## Variables:
##
## jaccard(close, halfway, far)
## polarity(same, similar, close, away)
## ranking(H, MED, M, L)
## cosine(vlow, low, medium, high)
##
## Rules:
##
## cosine %is% low && jaccard %is% close && polarity %is% similar => ranking %is% M
## cosine %is% medium && jaccard %is% close && polarity %is% same => ranking %is% MED
## cosine %is% medium && jaccard %is% close && polarity %is% similar => ranking %is% M
## cosine %is% medium && jaccard %is% halfway && polarity %is% same => ranking %is% ME
## cosine %is% medium && jaccard %is% halfway && polarity %is% similar => ranking %is%
## cosine %is% low || jaccard %is% far || polarity %is% away => ranking %is% L
## cosine %is% low || jaccard %is% close || polarity %is% same => ranking %is% M
## cosine %is% low || jaccard %is% halfway || polarity %is% away => ranking %is% L
## cosine %is% low || jaccard %is% halfway || polarity %is% same => ranking %is% L
## cosine %is% low || jaccard %is% halfway || polarity %is% close => ranking %is% L
## cosine %is% low || jaccard %is% halfway || polarity %is% similar => ranking %is% L
## cosine %is% medium || jaccard %is% far || polarity %is% away => ranking %is% L
## cosine %is% high => ranking %is% H
```

```
## cosine %is% vlow => ranking %is% L
```

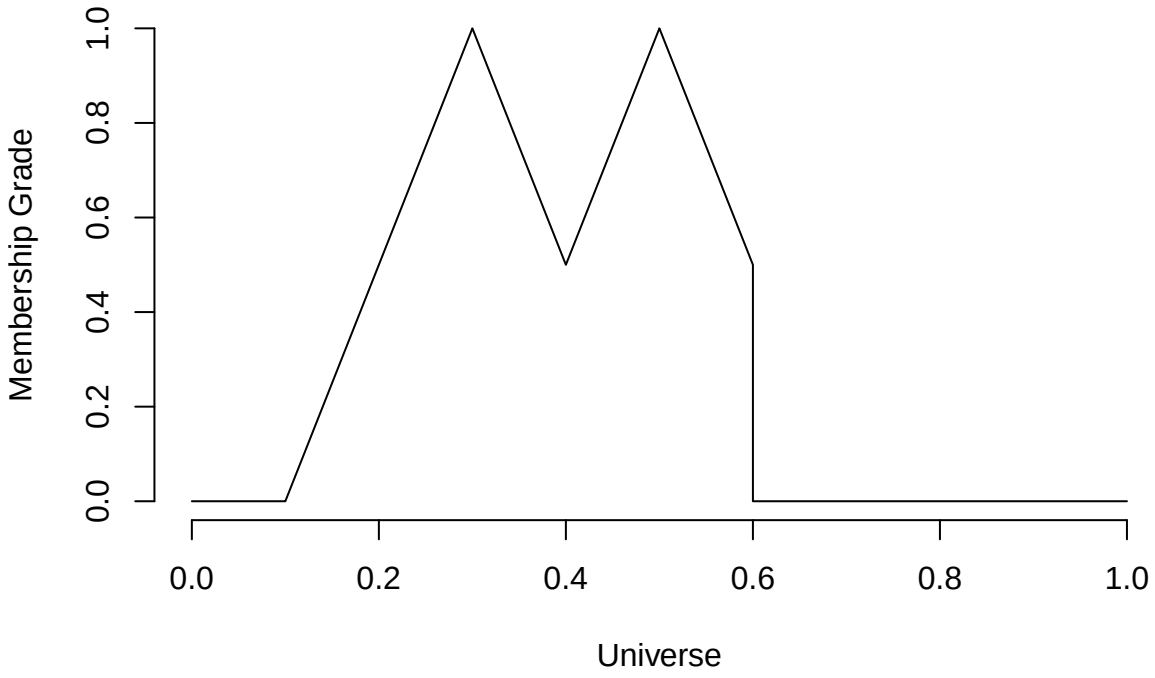
```
plot(ranking.system)
```




```
fi <- fuzzy_inference(ranking.system, list(cosine=0.5, jaccard=0, polarity=0.0))
gset_defuzzify(fi, "centroid")
```

```
## [1] 0.4
```

```
plot(fi)
```




```
get.ranks <- function(dataframe){
  cosine   = as.numeric(dataframe['cosine'])
  jaccard  = as.numeric(dataframe['jaccard'])
  polarity = as.numeric(dataframe['polaritydiff'])
  fi <- fuzzy_inference(ranking.system, list(cosine=cosine, jaccard=jaccard,
                                             polarity=polarity))
  return(gset_defuzzify(fi, "centroid"))
}

match.refined$ranking <- apply(match.refined, 1, get.ranks)
match.refined <- match.refined[order(-match.refined$ranking),]
as_tibble(match.refined)
```

```
## # A tibble: 30 x 8
##        ID cosine TITLE           PUBLISHER CATEGORY jaccard polaritydiff ranking
##     <int>  <dbl> <chr>           <chr>     <chr>      <dbl>        <dbl>   <dbl>
##  1  25151  0.236 Oracle says ne~ Reuters   b              1        0.756     0.5
##  2  91477  1     American Airli~ Reuters   b              1        0         0.4
##  3 226424  0.535 First quarter ~ Boston G~ b              0.5      0.348     0.3
##  4 226104  0.401 Home Depot fir~ CBS News  b              0.5      0.145     0.3
##  5 188909  0.354 U.S. Productiv~ NASDAQ    b              0.5      0.468     0.3
##  6 192643  0.354 Emirates airli~ GlobalPo~ b              0.5      0.240     0.3
```

```
##  7 255932  0.267 Home Price Ris~ NASDAQ    b           0.5       0.535     0.3
##  8 171393  0.267 eBay reports f~ Times of~ b           0.5       0.240     0.3
##  9 208112  0.267 Chrysler repor~ Business~ b           0.5       0.145     0.3
## 10 155774  0.267 Unilever says ~ Economic~ b           0.5       0.120     0.3
## # i 20 more rows
```

3 Appendix 1: Self-noted terminologies

There are two different ways to collect samples: **Sampling with replacement** and **sampling without replacement**.

- **Sampling with replacement** is the method where the items in the samples are *independent* because the outcome of one random draw is not affected by the previous draw. Sampling with replacement is useful in many different scenarios in statistics and machine learning like *bootstrapping*, *bagging*, *boosting*, *random forests*, etc. The sampling method allows generating different models with the same data set. This is less time-consuming and expensive than acquiring new data every time a new model is built.
- **Sampling without replacement** is the method where the items in the samples are *dependent* because the outcome of one random draw is affected by the previous draw. It is typically useful when we want to select a random sample from a population.

3.1 Feature engineering

Polarity of the document. A subjective content about a topic tends to have a *positive*, *negative*, or *neutral* perspective. Polarity identification algorithms quantify the perspective using text mining. Manhattan distance is a simple algorithm to measure the polarity value.

In data analysis, **cosine similarity** is a *measure of similarity* between two non-zero vectors defined in an inner product space. Cosine similarity is the cosine of the angle between the vectors; that is, it is the dot product of the vectors divided by the product of their lengths. It follows that the cosine similarity does not depend on the magnitudes of the vectors, but only on their angle. The cosine similarity always belongs to the interval $[-1, 1]$. The term cosine distance is commonly used for the complement of cosine similarity in positive space, that is

$$\text{cosine distance} = D_C(A, B) := 1 - S_C(A, B) = 1 - \frac{\sum_{i=1} n(A_i B_i)}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

The **Jaccard index**, also known as the **Jaccard similarity coefficient**, is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets. The **Jaccard distance**, which measures *dis*similarity between sample sets, is complementary to the Jaccard

coefficient and is obtained by subtracting the Jaccard coefficient from 1, or, equivalently, by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union:

$$d_J(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

4 Bibliography

Sampling With Replacement vs. Without Replacement