13주차:
텍스트 마이닝

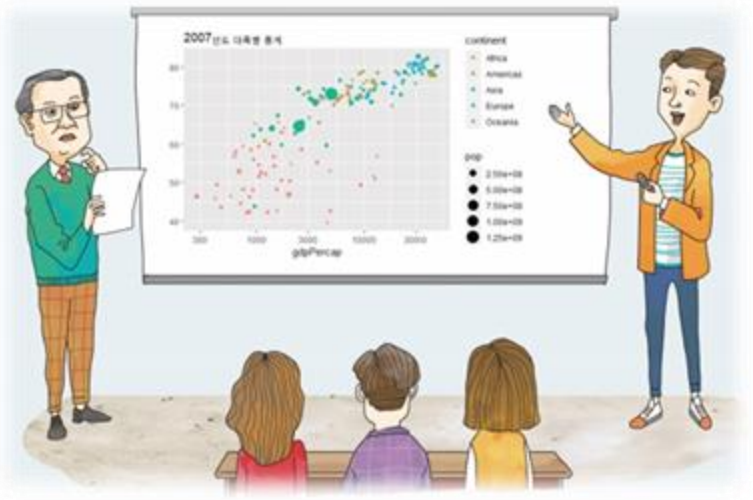
ChulSoo Park

School of Computer Engineering & Information Technology

Korea National University of Transportation

11 CHAPTER

텍스트 마이닝



CONTENTS

11.3 단어 구름(word cloud)

- DTM을 구성하는 단어를 가시화
  - 단어마다 중요성이 다르고, 단어 사이에 연관관계 정보가 있음.
  - 예) data는 field보다 science와 연관성이 더 높음
  - 단어 구름은 이런 정보를 2차원 공간에 표시하는 가시화 기법
  - 중요도가 높은 단어는 큰 폰트를 써서 중앙에 배치. 연관성이 높은 단어는 가까이 배치

Terms Docs	big	data	field	information	knowledge	learning	name	new	science	statistics
10	0	2	0	0	0	0	0	1	0	
12	4	6	0	1	0	0	0	1	1	0
13	0	2	0	0	0	1	0	0	2	0
2	1	6	1	0	2	1	0	0	2	0
3	0	5	0	2	1	0	0	0	6	2
4	1	10	2	2	2	1	0	0	5	2
5	0	11	2	0	0	0	1	1	8	8
6	0	4	1	0	0	0	1	1	2	3
7	0	5	0	0	0	0	2	2	4	2
9	0	8	1	0	0	2	2	1	7	1

11.3 단어 구름(wordcloud)

단어 구름 예제

- 앞 절에서 구한 DTM을 활용(readLines('https://en.wikipedia.org/wiki/Data_science',n=577))

```
Console C:/RSources/
> library(wordcloud)
필요한 패키지를 로딩중입니다: RColorBrewer
경고메시지(들):
패키지 'wordcloud'는 R 버전 4.0.5에서 작성되었습니다
> m = as.matrix(dtm)
> v = sort(colSums(m), decreasing = TRUE)
> d = data.frame(word = names(v), freq = v)
> wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.wo
rds = 100, random.order = FALSE, rot.per = 0.35)
```

as.matrix 함수는 DTM을 행렬 표현으로 변환

sort 함수는 빈도(중요도)가 높은 순서로 단어를 정렬

데이터 프레임으로 변환

중요도가 높은 상위 100개만 그리라는 옵션

세로로 배치할 단어의 비율을 35%로 하라는 옵션

가장 중요한 단어 : data
그리고 science, statistics, fild.....

11.3 단어 구름(wordcloud)

단어 구름 예제

```
> dtm
<<DocumentTermMatrix (documents: 13, terms: 363)>>
Non-/sparse entries: 482/4237
Sparsity           : 90%
Maximal term length: 17
Weighting          : term frequency (tf)
> m = as.matrix(dtm)
> m
    Terms
Docs across actionable algorithms application apply big broad data domains extract
  1      0          0          0           0     0   0     0    0       0       0
  2      1          1          1           1     1   1     1    6       1       1
    Terms
Docs field insights interdisciplinary knowledge learning machine methods mining
  1     0        0                 0         0        0       0       0      0
  2     1        2                 1         2        1       1       1      1

    Terms
Docs endend fullfeatured heavily platforms recently technologies variety
  1      0            0       0         0        0            0       0
  2      0            0       0         0        0            0       0
 [ getOption("max.print") 에 도달했습니다 -- 11 행들을 생략합니다 ]
```

as.matrix 함수는 DTM을 행렬 표현으로 변환

11.3 단어 구름(wordcloud)

단어 구름 예제

Console Jobs

C:/RSources/

```
> v = sort(colSums(m), decreasing = TRUE)
> v
```

sort 함수는 빈도(중요도)가 높은 순서로 단어를 정렬

data	science	statistics	field
61	42	18	7
big	information	name	new
6	6	6	6
knowledge	learning	used	application
5	5	5	4
analysis	computer	statistical	“data
4	4	4	4
domains	insights	interdisciplinary	machine
3	3	3	3
range	systems	datadriven	many
3	3	3	3
techniques	problems	described	digital
3	3	3	3
university	jeff	actionable	algorithms
3	3	2	2
broad	methods	mining	related
2	2	2	2
uses	concept	different	however
2	2	2	2

endend	fullfeatured	heavily	platforms
1	1	1	1
recently	technologies	variety	
1	1	1	

11.3 단어 구름(word cloud)

- 단어 구름의 여러 가지 옵션
  - RColorBrewer 라이브러리 (6.3.2절 참조)를 이용해 색상 입히기

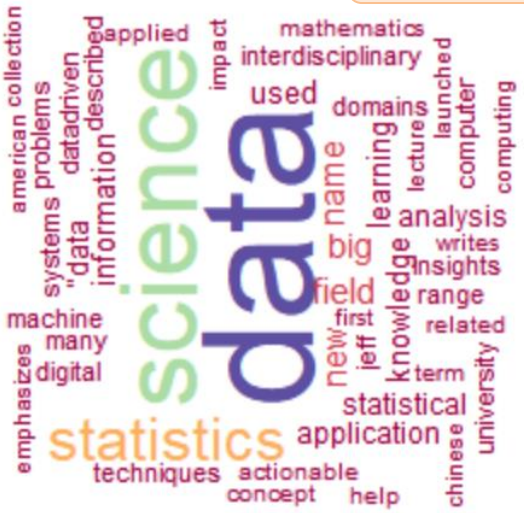
```
> library(RColorBrewer)
> pal = brewer.pal(11,"Spectral")
> wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.words =
  50, random.order = FALSE, rot.per = 0.50, colors = pal)
> wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.words =
  50, random.order = FALSE, rot.per = 0.50, colors = pal, family = "mono",
  font = 2)
```

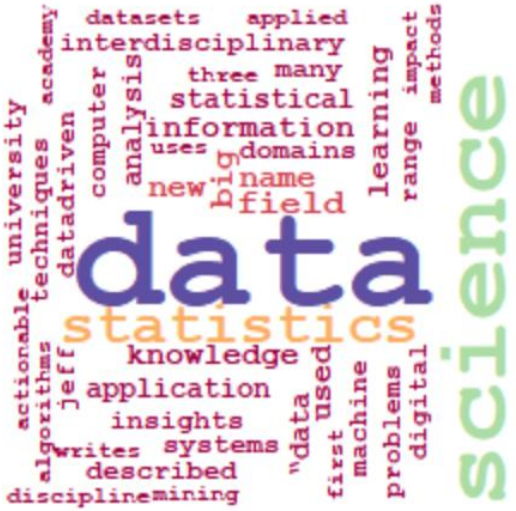
가로 세로 비율

색상

폰트

단어 개수





11.3 단어 구름(word cloud)

- 보다 뛰어난 wordcloud2 라이브러리
  - wordcloud2는 wordcloud 이후에 나온 새로운 버전
  - 자동으로 색상 입혀주고 단어를 다양한 방향으로 배치해 줌

```
Console  C:/RSources/
> library(wordcloud2)
> wordcloud2(d)
```

11.3 단어 구름(word cloud)

- wordcloud2의 여러 가지 옵션들

```
Console  C:/RSources/
> d1 = d[1:200, ]          # 단어 200개
> wordcloud2(d1, shape = 'star')
> wordcloud2(d1, minRotation = pi/4, maxRotation = pi/4, rotateR
atio = 1.0)
```

wordcloud2에는 단어 개수 지정하는 옵션이 없어 미리 추출

배경 모양

단어 방향 범위 지정

11.3 단어 구름(word cloud)

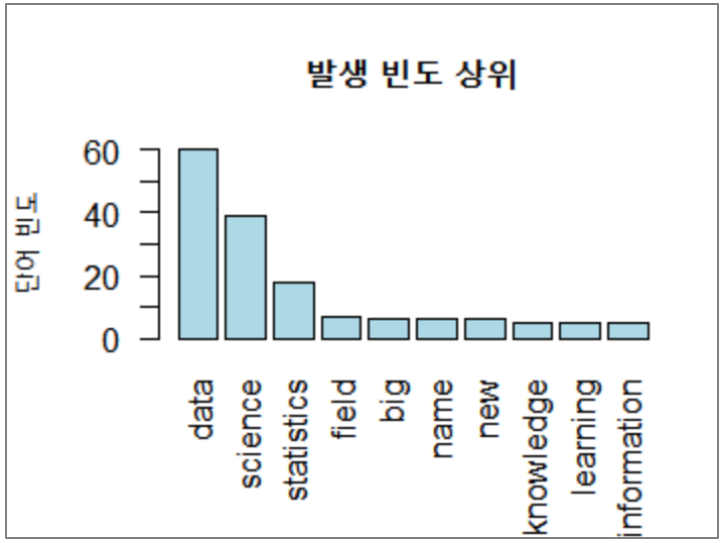
- 빈도 표시하기 : findFreqTerms와 findAssocs 함수

```
Console  C:/RSources/
> findFreqTerms(dtm, lowfreq = 12)
[1] "data"       "science"    "statistics"
>
> findAssocs(dtm, terms = 'data', corlimit = 0.6)
$data
     field    science statistics   problems   columbia
      0.83       0.83       0.67       0.66       0.60

>
> barplot(d[1:10, ]$freq, las = 2, names.arg = d[1:10,]$word, col = 'lightblue', ma
in = '발생 빈도 상위', ylab = '단어 빈도')
```

상위 12개 단어만 표시하라는 옵션

'dara'와 상관관계가 0.6이상인 단어를 표시하라는 옵션



11.3 단어 구름(word cloud)

- wordcloud는 데이터 프레임을 사용
  - 첫번째 열은 단어, 두번째 열은 해당 단어의 빈도수인 데이터 프레임
  - 텍스트 이외에도 이런 형식을 갖추면 단어 구름 가능함
  - 예) gapminder에서 첫번째 열은 대륙, 두번째 열은 해당 대륙의 인구가 되도록 데이터 추출

```
Console C:/RSources/
> library(gapminder)
> library(dplyr)
>
> pop_siz = gapminder%>%filter(year==2007)%>%group_by(continent)%>%summarize(sum(as.nu
meric(pop)))
> d = data.frame(word = pop_siz[, 1], freq = pop_siz[, 2])
> wordcloud(words = d[, 1], freq = d[, 2], min.freq = 1, max.words = 100, random.order
 = FALSE, rot.per = 0.35)
> wordcloud2(d)
```

글씨 표시할 때
35%는 세로로 표시

11.4 문서 분류

- 지도 학습(supervised learning)에 속하는 분류 문제
  - 단어 구름은 깔끔하게 시각화하여 직관적으로 이해는 데 매우 유리함
  - 레이블링을 하지 않은 데이터를 사용하여 비지도학습임.
  - 예) Preview에서 제기한 시의 분류 문제(시를 보고 누구의 시인지 분류)
  - 다양한 응용
    - ✓ 영화 관람평을 모델링하면 흥행 예측 가능
    - ✓ 상품에 대한 댓글을 분석하여 마케팅 전략 세움
    - ✓ 트윗을 분석하여 대선이나 총선 결과 예측
    - ✓ 주식 관련 댓글을 보고 주가 예측
    - ✓ …..

11.4 문서 분류

- 영화평 분류 : movie_review 데이터 모델링
  - text2vec 라이브러리가 제공
  - tm과 SnowballC 라이브러리는 텍스트마이니을 위해 사용
  - caret은 모델의 성능 평가를 위해 사용
  - 5000개의 샘플, 3개의 변수(id, sentiment, review)
    - id는 일련번호이므로 무시, sentiment는 반응 변수에 해당(1은 긍정 평가, 0은 부정 평가)
    - Review 변수는 영화를 평가하는 텍스트

```
Console C:/RSources/
> library(text2vec)
> library(caret)
> str(movie_review)
'data.frame':   5000 obs. of  3 variables:
 $ id       : chr  "5814_8" "2381_9" "7759_3" "3630_4" ...
 $ sentiment: int  1 1 0 0 1 1 0 0 0 1 ...
 $ review   : chr  "With all this stuff going down at the moment with MJ i've start
ed listening to his music, watching the odd docu"| __truncated__ "\\\"The Classic W
ar of the Worlds\\\" by Timothy Hines is a very entertaining film that obviously go
es to great"| __truncated__ "The film starts with a manager (Nicholas Bell) giving
```

11.4 문서 분류

- 첫 번째 샘플의 review 변수 내용

```
> head(movie_review)
       id sentiment
1 5814_8         1
2 2381_9         1
3 7759_3         0
4 3630_4         0
5 9495_8         1
6 8196_8         1
```

Console C:/RSources/

review

1

With all this stuff going down at the moment with MJ i've started listening to his music, watching the odd documentary here and there, watched The Wiz and watched Moonwalker again. Maybe i just want to get a certain insight into this guy who i thought was really cool in the eighties just to maybe make up my mind whether he is guilty or innocent. Moonwalker is part biography, part feature film which i remember going to see at the cinema when it was originally released. Some of it has subtle messages about MJ's feeling towards the press and also the obvious message of drugs are bad m'kay.

Visually impressive but of course this is all about Michael Jackson so unless you remotely like MJ in anyway then you are going to hate this and find it boring. Some may call MJ an egotist for consenting to the making of this movie BUT MJ and most of his fans would say that he made it for the fans which if true is really nice of

3 The film starts with a manager (Nicholas Bell) giving welcome investors (Robert Carradine) to Primal Park . A secret project mutating a primal animal using fossilized DNA, like Jurassik Park, and some scientists resurrect one of nature's most fearsome predators, the Sabretooth tiger or Smilodon . Scientific ambition turns deadly, however, and when the high voltage fence is opened the creature escape and begins savagely stalking its prey - the human visitors , tourists and scientific.Meanwhile some youngsters enter in the restricted area of the security center and are attacked by a pack of large pre-historical animals which are deadlier and bigger . In addition , a security agent (Stacy Haiduk) and her mate (Brian Wimmer) fight hardly against the carnivorous Smilodons. The Sabretooths, themselves , of course, are the real star stars and they are astounding terrifyingly though not convincing. The giant animals savagely are stalking its prey and the group run afoul and fight against one nature's most fearsome predators. Furthermore a third Sabretooth more dangerous and slow stalks its victims.

The movie delivers the goods with lots of blood and gore as beheading, hair-raising chills,full of scares when the Sabretooths appear with mediocre special effects.The story provides exciting and stirring entertainment but it results to be quite boring .The giant animals are majority

이 샘플의 sentiment 변수 값은 1(긍정평가), 0(부정) : review(1), review(3)

11.4 문서 분류

영화평 분류: movie_review 데이터 모델링

- 6:4 비율로 훈련 집합과 테스트 집합으로 분할

```
Console  C:/RSources/
> # 데이터 나눔 훈련 집합(mtrain), 테스트 집합(mtest)
> train_list = createDataPartition(y= movie_review$sentiment, p = 0.6, list =  FALSE)
> mtrain = movie_review[train_list, ]
> mtest = movie_review[-train_list, ]
```

- 훈련 집합에 대해 DTM 구축

```
> # 데이터 나눔 훈련 집합(mtrain), 테스트 집합(mtest)
> train_list = createDataPartition(y= movie_review$sentiment, p = 0.6, list =  FALSE)
> mtrain = movie_review[train_list, ]
> mtest = movie_review[-train_list, ]
> doc = Corpus(VectorSource(mtrain$review))
> doc = tm_map(doc, content_transformer(tolower))
> doc = tm_map(doc, removeNumbers)
> doc = tm_map(doc, removeWords, stopwords('english'))
> doc = tm_map(doc, removePunctuation)
> doc = tm_map(doc, stripWhitespace)
> dtm = DocumentTermMatrix(doc)
> dim(dtm)
[1]  3000 36967
```

사전의 크기는 36871 (3000개 문서에서 36967개의 단어가 추출됨)

11.4 문서 분류

- inspect 함수로 DTM의 내용을 살펴보면

Console C:/RSources/

```
> inspect(dtm)
<<DocumentTermMatrix (documents: 3000, terms: 36967)>>
Non-/sparse entries: 298020/110602980
Sparsity           : 100% 99.7%
Maximal term length: 47
Weighting          : term frequency (tf)
Sample             :
      Terms
Docs   even film good just like movie one really see time
  127     3    7   11    2    3     5   3      1   0    2
  1329    3    6    5    6    9    14   4      1   6    1
  1714    2   14    1    5    2     1   5      3   2    5
  1803    2    0    2    0    2     0   6      0   2    2
  2108    0    0    1    4    3     0   2      2   0    0
  2164    2    9    1    1    2     6   7      1   1    2
  2232    1    9    4    1    1     0   6      0   2    1
  2558    3    1    2    2    4    14   3      1   1    3
  2649    0    3    1    1    0     3   2      0   2    0
  41      0    0    0    3    2     2   1      2   2    0
```

3000*36967=110901000개의 칸 중에
298020개만 0이 아니고 나머지 110602980개는 0

11.4 문서 분류

- 영화평 분류: movie_review 데이터 모델링
- DTM을 모델링 가능한 형태로 변환
  - 사전이 아주 커서 그대로 적용하면 메모리 오류 발생 → removeSparseTerms 함수로 줄임(빈도가 일정 정도 이하인 단어는 제거)
  - cbind 함수로 반응 변수 sentiment를 덧붙임

```
Console  C:/RSources/
> dtm_small = removeSparseTerms(dtm, 0.90)
> X = as.matrix(dtm_small)
> dataTrain = as.data.frame(cbind(mtrain$sentiment, X))
> dataTrain$V1 = as.factor(dataTrain$V1)
> colnames(dataTrain)[1] = 'y'
```

11.4 문서 분류

```
> inspect(dtm)
<<DocumentTermMatrix (documents: 3500, terms: 39984)>>
Non-/sparse entries: 346669/139597331
Sparsity           : 100%
Maximal term length:
Weighting          :
Sample             :
       Terms
Docs    can even film
  1547    3     3    6
  2004    1     2   14
  2239    2     2   11
  2317    1     1    9
  2447    1     2    0
  2526    0     2    9
  3071    7     0    3
  3455    4     3    0
  46      3     0    0
  542     1     0    6
```

Console Jobs

C:/RSources/

```
> dtm_small = removeSparseTerms(dtm, 0.90)
> inspect(dtm_small)
<<DocumentTermMatrix (documents: 3500, terms: 111)>>
Non-/sparse entries: 70604/317896
Sparsity           : 82%
Maximal term length: 11
Weighting          : term frequency (tf)
Sample             :
       Terms
Docs    can even film good just like movie one really time
  1547    3     3    6    5    6    9    14   4       1    1
  1837    0     6   11    0    2    0     0   5       0    3
  2239    2     2   11    0    2    4     8   4       2    0
  2447    1     2    0    1    2    3     0   6       0    4
  2576    0     6   22    7    5    3     5   1       2    0
  2700    4     0   11    3    6    4     7   3       5    5
  2930    2     4   20    6    3    8     1   2       5    3
  3017    3     3   20    3    4    2     0   1       5    2
  3083    2     1    0    2    3    4     7  18       3    3
  3455    4     3    0    4    3    4     0   5       3    3
```

11.4 문서 분류

```
> head(dataTrain)
  V1 can every fact film found great look made movie never people things better director
1  1   1     1    1    2     1     1    1    1     2     1      1      1      0        0
2  0   0     0    0    1     0     0    0    1     1     0      0      0      1        1
3  0   1     1    0    1     1     0    0    1     1     0      0      1      0        0
4  1   0     0    0    1     0     0    0    0     2     0      0      0      0        0
5  1   0     0    0    3     0     0    0    0     0     0      0      0      0        0
6  0   0     0    0    0     0     0    0    0     2     0      0      0      0        0
  films give however like man many much one quite real scene scenes story though actors
1     0    0       0    0   0    0    0   0     0    0     0      0     0      0       0
2     2    1       1    1   1    2    1   2     1    1     1      1     1      2       0
3     0    1       1    0   0    0    1   0     0    0     1      0     0      0       1
4     0    0       0    0   0    0    0   2     0    0     1      0     0      0       0
5     0    0       0    0   0    0    0   0     0    0     0      0     0      0       0
6     0    0       0    1   0    1    1   1     0    0     0      0     0      0       0
  big even ever just least life little lot makes plot time times two watch well whole
1   0    0    0    0     0    0      0   0     0    0    0     0   0     0    0     0
2   0    0    0    0     0    0      0   0     0    0    0     0   0     0    0     0
3   2    3    1    1     1    2      1   1     1    3    1     1   1     1    1     1
4   0
5   2
6   0
  years
```

```
> colnames(dataTrain)[1] = 'y'
> head(dataTrain)
  y can every fact film found great look made movie never people things better director
1 1   1     1    1    2     1     1    1    1     2     1      1      1      0        0
2 0   0     0    0    1     0     0    0    1     1     0      0      0      1        1
3 0   1     1    0    1     1     0    0    1     1     0      0      1      0        0
4 1   0     0    0    1     0     0    0    0     2     0      0      0      0        0
5 1   0     0    0    3     0     0    0    0     0     0      0      0      0        0
```

11.4 문서 분류

- 영화평 분류: movie_review 데이터 모델링
- 결정 트리와 랜덤 포리스트를 학습

Console C:/RSources/

```
> library(rpart)
> r = rpart(y~., data = dataTrain)
> printcp(r)

Classification tree:
rpart(formula = y ~ ., data = dataTrain)

Variables actually used in tree construction:
[1] bad     great   love    nothing plot    thing

Root node error: 1493/3000 = 0.49767

n= 3000

        CP nsplit rel error  xerror     xstd
1 0.224380      0   1.00000 1.02411 0.018340
2 0.039518      1   0.77562 0.77562 0.017860
3 0.016410      3   0.69658 0.69190 0.017431
4 0.010000      6   0.64434 0.67046 0.017298
```

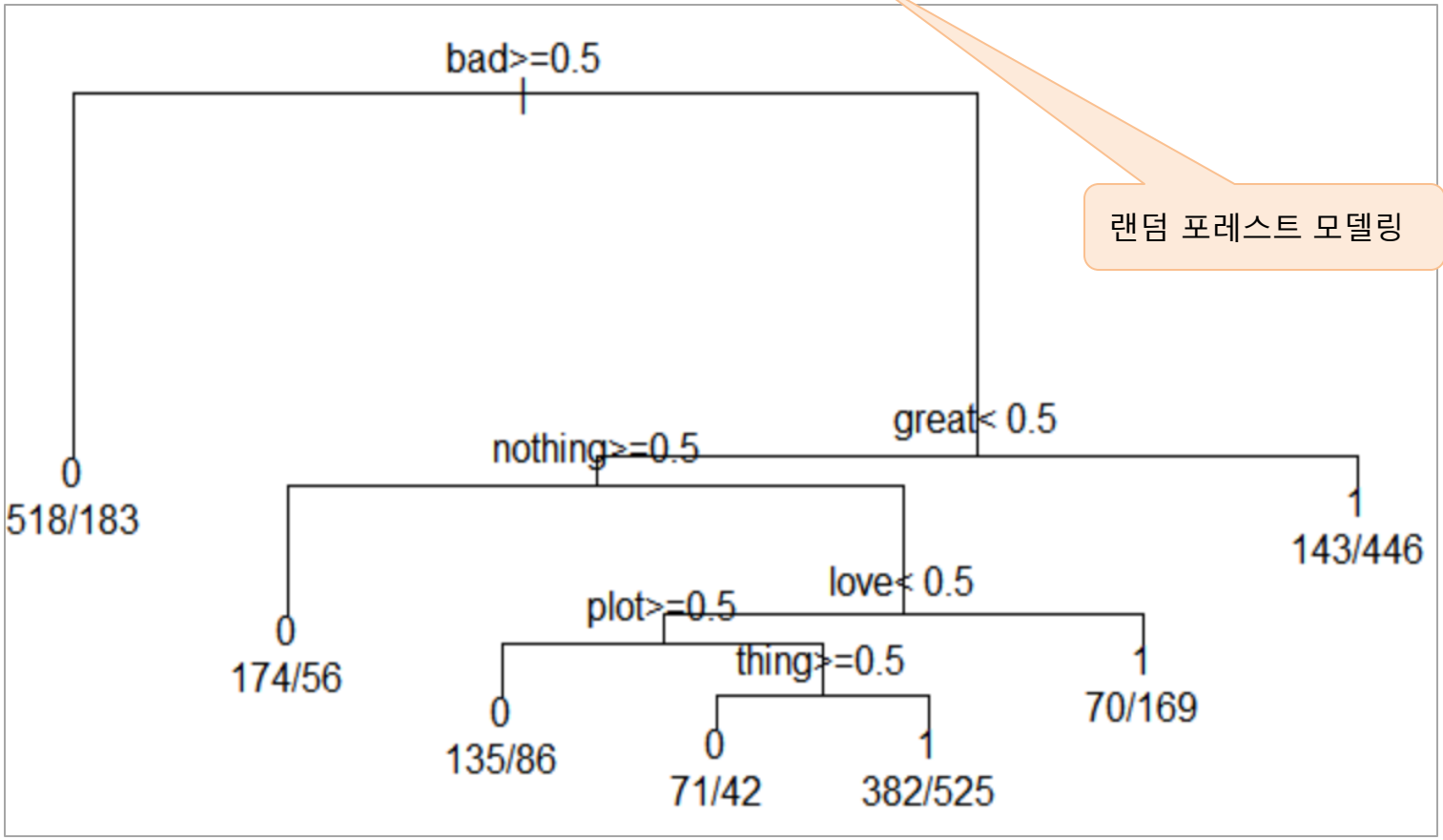
의사 결정 모델링

11.4 문서 분류

- 영화평 분류: movie_review 데이터 모델링
- 결정 트리와 랜덤 포리스트를 학습

```
Console C:/RSources/
> par(mfrow = c(1, 1), xpd = NA)
> plot(r)
> text(r, use.n = TRUE)
>
> library(randomForest)
> f = randomForest(y~., data = dataTrain)
```

랜덤 포레스트 모델링



11.4 문서 분류

- 예측과 성능 평가(테스트 목적으로 남겨둔 mtest로 성능을 평가)
  - 먼저 mtest에 전처리 적용 (학습 과정과 동일한 과정을 적용해야 함)
  - 훈련 집합이 사용한 사전을 테스트 과정에도 그대로 사용해야 함

```
# test 집합으로 DTM 구축
> docTest = Corpus(VectorSource(mtest$review))
> docTest = tm_map(docTest, content_transformer(tolower))
> docTest = tm_map(docTest, removeNumbers)
> docTest = tm_map(docTest, removeWords, stopwords('englisl
> docTest = tm_map(docTest, removePunctuation)
> docTest = tm_map(docTest, stripWhitespace)

> dtmTest = DocumentTermMatrix(docTest,
    control=list(dictionary=dtm_small$dimnames$Terms))
```

전처리 작업

control 옵션으로 훈련 집합으로 만든 사전을 test 동일하게 사용

11.4 문서 분류

- 예측과 성능 평가(테스트 목적으로 남겨둔 mtest로 성능을 평가)
  - 먼저 mtest에 전처리 적용 (학습 과정과 동일한 과정을 적용해야 함)
  - 훈련 집합이 사용한 사전을 테스트 과정에도 그대로 사용해야 함

```
Console C:/RSources/
> dim(dtmTest)
[1] 2000  110
> str(dtmTest)
List of 6
 $ i       : int [1:39602] 1 1 1 1 1 1 1 1 1 1 ...
 $ j       : int [1:39602] 1 2 3 4 5 6 7 8 9 10 ...
 $ v       : num [1:39602] 1 1 1 2 1 1 1 2 1 1 ...
 $ nrow    : int 2000
 $ ncol    : int 110
 $ dimnames:List of 2
  ..$ Docs : chr [1:2000] "1" "2" "3" "4" ...
  ..$ Terms: chr [1:110] "can" "every" "fact" "film" ...
> inspect(dtmTest)
<<DocumentTermMatrix (documents: 2000, terms: 110)>>
Non-/sparse entries: 39602/180398
Sparsity           : 82%
Maximal term length: 11
Weighting          : term frequency (tf)
Sample             :
      Terms
Docs   even film good just like movie one really see time
  1400    2    0    1    2    3     0   6      0   1    4
  1472    6   22    7    5    3     5   1      2   0    0
  1557    0   11    3    6    4     7   3      5   5    5
  1685    4   20    6    3    8     1   2      5   0    3
  1734    3   20    3    4    2     0   1      5   5    2
```

11.4 문서 분류

- 결정 트리와 랜덤 포리스트로 예측 수행

```
> X = as.matrix(dtmTest)
> dataTest = as.data.frame(cbind(mtest$sentiment, X))
> dataTest$V1 = as.factor(dataTest$V1)
> colnames(dataTest)[1] = 'y'
> pr = predict(r, newdata = dataTest, type = 'class')
> table(pr, dataTest$y)

pr    0   1
  0 554 241
  1 436 769
> pf = predict(f, newdata = dataTest)
> table(pf, dataTest$y)

pf    0   1
  0 671 266
  1 319 744
```

의사 결정 모델링

랜덤 포레스트 모델링

- 결정 트리는 (554+769)/2000=66.2%, 랜덤 포리스트는 (671+744)/2000=70.8%의 정확률

Thank you


Africa
Americas
Asia
Europe
Oceania
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09
gdpPercap