What’s cooking?

박찬엽

음식 아이디 별로 제공된 음식의 재료(`ingredients`)로 종류(`cuisine`)를 맞추기

데이터 불러오기

```
cuis <- fromJSON("data/train.json") %>%
  as_tibble() %>%
  rename(ingre = ingredients)

cuis %>%
  unnest %>%
  head(10) %>%
  gt() %>%
  tab_options(
    table.width = pct(100)
  )
```

id	cuisine	ingre
10259	greek	romaine lettuce
10259	greek	black olives
10259	greek	grape tomatoes
10259	greek	garlic
10259	greek	pepper
10259	greek	purple onion
10259	greek	seasoning
10259	greek	garbanzo beans
10259	greek	feta cheese crumbles
25693	southern_us	plain flour

전처리

띄어쓰기 단위 분리, 소문자화, 특수문자 제거 진행

```
cuis_pre <-
  cuis %>%
  unnest() %>%
  # 띄어쓰기 단위 분리
  unnest_tokens(word, ingre) %>%
  # 소문자 화
  mutate(word = tolower(word)) %>%
  # 언더스코어로 통일
  mutate(word = gsub("-", "_", word)) %>%
  # 특수문자 제거
  mutate(word = gsub("[^a-z0-9_ ]", "", word)) %>%
  # 원문 복원
  # mutate(word = lemmatize_words(word)) %>%
  # id당 한 문장으로 결합
  group_by(id, cuisine) %>%
  summarise(ingre = paste0(word, collapse = " ")) %>%
  ungroup()
```

데이터 나누기

데이터 셋을 20/80으로 나누어야 함. 카테고리가 있어 각 카테고리별로 균등 분할함.

```
set.seed(2019)
split <- createDataPartition(cuis_pre$cuisine, p = 0.8)

train <- cuis_pre[split$Resample1, ]
test <- cuis_pre[-split$Resample1, ]

train %>%
  group_by(cuisine) %>%
  summarise(n = n()) %>%
  mutate(per = n/sum(n)) %>%
  left_join(
    test %>%
      group_by(cuisine) %>%
      summarise(n = n()) %>%
      mutate(per = n/sum(n)),
    by = "cuisine"
  ) %>%
  mutate(diff = per.x - per.y) %>%
  gt() %>%
  tab_options(
    table.width = pct(100)
  ) %>%
  cols_label(
    n.x = "cuisine별 갯수",
    per.x = "cuisine별 비율",
    n.y = "cuisine별 갯수",
    per.y = "cuisine별 비율",
    diff = "비율의 차이"
  ) %>%
  tab_spanner(
    label = "Train 데이터 셋",
    columns = vars(n.x, per.x)
  ) %>%
  tab_spanner(
    label = "Test 데이터 셋",
    columns = vars(n.y, per.y)
  ) %>%
  fmt_percent(
    columns = vars(per.x, per.y),
    decimals = 2
  )
```

cuisine	Train 데이터 셋		Test 데이터 셋		비율의 차이
	cuisine별 갯수	cuisine별 비율	cuisine별 갯수	cuisine별 비율	
brazilian	374	1.18%	93	1.17%	4.849974e-05
british	644	2.02%	160	2.01%	1.010085e-04
cajun_creole	1237	3.89%	309	3.89%	-1.622589e-05
chinese	2139	6.72%	534	6.72%	1.192033e-05
filipino	604	1.90%	151	1.90%	-2.328320e-05
french	2117	6.65%	529	6.66%	-5.014843e-05
greek	940	2.95%	235	2.96%	-3.623544e-05
indian	2403	7.55%	600	7.55%	1.743571e-06
irish	534	1.68%	133	1.67%	4.233201e-05
italian	6271	19.70%	1567	19.72%	-1.473614e-04
jamaican	421	1.32%	105	1.32%	1.522956e-05
japanese	1139	3.58%	284	3.57%	5.046867e-05
korean	664	2.09%	166	2.09%	-2.559610e-05
mexican	5151	16.18%	1287	16.19%	-1.041873e-04
moroccan	657	2.06%	164	2.06%	6.132152e-06
russian	392	1.23%	97	1.22%	1.107227e-04
southern_us	3456	10.86%	864	10.87%	-1.332231e-04
spanish	792	2.49%	197	2.48%	9.530336e-05
thai	1232	3.87%	307	3.86%	7.834209e-05
vietnamese	660	2.07%	165	2.08%	-2.544191e-05

text 벡터화

빠르고 유의미하게 수행할 수 있는 tf-idf로 벡터화 진행

```
# iterator 생성
it_train = itoken(train$ingre,
                  tokenizer = word_tokenizer,
                  ids = train$id,
                  progressbar = FALSE)

# 단어 사전을 학습 데이터 셋에서 구축
vocab = create_vocabulary(it_train)
vectorizer = vocab_vectorizer(vocab)
# dtMatrix 생성
dtm_train = create_dtm(it_train, vectorizer)

# tfidf 값 계산 적용
tfidf = TfIdf$new()
dtm_train_tfidf = fit_transform(dtm_train, tfidf)



it_test = itoken(test$ingre,
                 tokenizer = word_tokenizer,
                 ids = test$id,
                 progressbar = FALSE)

## 기 학습한 tfidf 모델을 활용하여 벡터화
dtm_test_tfidf <- create_dtm(it_test, vectorizer) %>%
  transform(tfidf)
```

모델 학습

다중 회귀의 10 fold cross validation 진행

```
glmnet_classifier <- cv.glmnet(x = dtm_train_tfidf,
                               y = train[['cuisine']],
                               family = "multinomial",
                               type.measure = "class",
                               nfolds = 10,
                               thresh = 1e-3,
                               maxit = 1e3)
```

테스트 진행

```
pred <- predict(object = glmnet_classifier, newx = dtm_test_tfidf, type = 'class')
ref <- factor(test$cuisine)

matric <-
  confusionMatrix(ref,
                  factor(pred, levels = levels(ref)))
matric
```

```
## Confusion Matrix and Statistics
##
##               Reference
## Prediction     brazilian british cajun_creole chinese filipino french
##   brazilian           21       0            1       2        0      2
##   british              0      14            0       1        0     37
##   cajun_creole         0       0          190       1        0     20
##   chinese              0       1            0     464        0      3
##   filipino             0       0            1      40       29      5
##   french               0       1            5       3        0    265
##   greek                0       0            0       0        0      8
##   indian               0       3            0       1        0      4
##   irish                0       2            0       0        0     15
##   italian              0       2            3       1        0     56
##   jamaican             0       1            2       5        0      6
##   japanese             0       0            0      46        0      8
##   korean               0       0            0      41        0      3
##   mexican              0       2            3       1        0     16
##   moroccan             0       0            0       0        0      3
##   russian              0       0            1       1        0     22
##   southern_us          0       1           48       2        1     40
##   spanish              0       0            1       1        1     19
##   thai                 1       0            0      30        0      4
##   vietnamese           1       0            0      37        2      4
##               Reference
## Prediction     greek indian irish italian jamaican japanese korean mexican
##   brazilian        0      0     0      22        1        0      0      15
##   british          2      5     3      22        1        0      0       4
##   cajun_creole     0      0     0      28        2        0      0      12
##   chinese          0      0     1      11        0       10      7       3
##   filipino         0      1     1      13        1        1      3       7
##   french           0      1     3     171        2        1      0       3
##   greek          124      6     0      77        0        0      0       3
##   indian           4    520     0      14        0        0      0      28
##   irish            0      1    18      19        4        0      0       2
##   italian          8      2     2    1413        1        0      1      12
##   jamaican         0     14     0       2       47        0      0       4
##   japanese         1     28     0       9        0      159      8       2
##   korean           0      0     0       1        0       11     89       3
##   mexican          2      3     0      38        1        1      0    1171
##   moroccan         5     23     0      25        0        0      0      19
##   russian          0      1     3      19        0        0      0       3
##   southern_us      1      5     3      78        3        0      0      29
##   spanish          0      0     2      91        0        0      0      29
##   thai             2     16     0       6        0        3      2      21
##   vietnamese       0      2     0       3        0        1      3      11
##               Reference
## Prediction     moroccan russian southern_us spanish thai vietnamese
##   brazilian           0       0          17       2    9          1
##   british             0       0          64       3    1          3
##   cajun_creole        0       0          54       1    1          0
##   chinese             0       0          21       0    9          4
##   filipino            0       0          26       1    5         17
##   french              5       1          59       5    2          2
##   greek               4       1          12       0    0          0
##   indian              5       0          12       1    8          0
##   irish               2       1          69       0    0          0
##   italian             2       1          60       2    0          1
##   jamaican            0       0          20       0    2          2
##   japanese            1       1          16       0    3          2
##   korean              0       0           7       0    2          9
##   mexican             1       0          44       3    1          0
##   moroccan           73       0          11       4    0          1
##   russian             0      20          25       1    0          1
##   southern_us         1       1         644       0    6          1
##   spanish             2       1          14      35    0          1
##   thai                1       0           3       0  202         16
##   vietnamese          0       0           4       0   56         41
##
## Overall Statistics
##
##                Accuracy : 0.697
##                  95% CI : (0.6868, 0.7071)
##     No Information Rate : 0.2595
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6568
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: brazilian Class: british Class: cajun_creole
## Sensitivity                  0.913043       0.518519             0.74510
## Specificity                  0.990914       0.981566             0.98453
## Pos Pred Value               0.225806       0.087500             0.61489
## Neg Pred Value               0.999745       0.998331             0.99149
## Prevalence                   0.002894       0.003398             0.03209
## Detection Rate               0.002643       0.001762             0.02391
## Detection Prevalence         0.011703       0.020133             0.03888
## Balanced Accuracy            0.951979       0.750042             0.86481
##                      Class: chinese Class: filipino Class: french
## Sensitivity                 0.68538        0.878788       0.49074
## Specificity                 0.99037        0.984584       0.96436
## Pos Pred Value              0.86891        0.192053       0.50095
## Neg Pred Value              0.97127        0.999487       0.96293
## Prevalence                  0.08519        0.004153       0.06795
## Detection Rate              0.05839        0.003649       0.03335
## Detection Prevalence        0.06720        0.019001       0.06657
## Balanced Accuracy           0.83787        0.931686       0.72755
##                      Class: greek Class: indian Class: irish
## Sensitivity               0.83221       0.82803     0.500000
## Specificity               0.98577       0.98907     0.985463
## Pos Pred Value            0.52766       0.86667     0.135338
## Neg Pred Value            0.99676       0.98530     0.997696
## Prevalence                0.01875       0.07902     0.004530
## Detection Rate            0.01560       0.06543     0.002265
## Detection Prevalence      0.02957       0.07550     0.016736
## Balanced Accuracy         0.90899       0.90855     0.742732
##                      Class: italian Class: jamaican Class: japanese
## Sensitivity                  0.6853        0.746032         0.85027
## Specificity                  0.9738        0.992643         0.98389
## Pos Pred Value               0.9017        0.447619         0.55986
## Neg Pred Value               0.8983        0.997960         0.99635
## Prevalence                   0.2595        0.007928         0.02353
## Detection Rate               0.1778        0.005914         0.02001
## Detection Prevalence         0.1972        0.013213         0.03574
## Balanced Accuracy            0.8295        0.869338         0.91708
##                      Class: korean Class: mexican Class: moroccan
## Sensitivity                0.78761         0.8479        0.752577
## Specificity                0.99017         0.9823        0.988408
## Pos Pred Value             0.53614         0.9099        0.445122
## Neg Pred Value             0.99692         0.9685        0.996916
## Prevalence                 0.01422         0.1738        0.012206
## Detection Rate             0.01120         0.1474        0.009186
## Detection Prevalence       0.02089         0.1619        0.020637
## Balanced Accuracy          0.88889         0.9151        0.870492
##                      Class: russian Class: southern_us Class: spanish
## Sensitivity                0.740741            0.54484       0.603448
## Specificity                0.990278            0.96748       0.979465
## Pos Pred Value             0.206186            0.74537       0.177665
## Neg Pred Value             0.999108            0.92404       0.997032
## Prevalence                 0.003398            0.14874       0.007298
## Detection Rate             0.002517            0.08104       0.004404
## Detection Prevalence       0.012206            0.10872       0.024789
## Balanced Accuracy          0.865509            0.75616       0.791457
##                      Class: thai Class: vietnamese
## Sensitivity              0.65798          0.401961
## Specificity              0.98626          0.984194
## Pos Pred Value           0.65798          0.248485
## Neg Pred Value           0.98626          0.992161
## Prevalence               0.03863          0.012835
## Detection Rate           0.02542          0.005159
## Detection Prevalence     0.03863          0.020763
## Balanced Accuracy        0.82212          0.693077
```