Xiner Ning Assignment 3-Part II

Xiner Ning

4/6/2020

Question 3 (Word2Vec) Emeddings

1. Unzip the files and take a look at some of the recipes. Now given that the provided data set has not been munged in any way, how would you improve the quality of the embedding?

Get rid of any extra space and punctuations.

Remove words that are less meaningful, such as a, the, of

Apply stemming and lemmatisation in cookbooks.txt to reduce duplicated words in the vocabulary such as eggs and egg

2. Run through the starter code set and try different ingredients, list your ingredients and the top 5 ones closest to them. They cannot be the same as presented in the starter code. Mark anything that is interesting. (This approach can be used for selecting paragraphs in a text with similar content but different words.)

```
library(wordVectors)
library(Rtsne)
library(tidytext)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------
--------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------------
--------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
model = read.vectors("cookbook_vectors.bin")
```

```
## Filename ends with .bin, so reading in binary format
```

```
## Reading a word2vec binary file of 25375 rows and 100 columns
```

```
##
  |                                                                      
  |                                                                |   0%
  |                                                                      
  |                                                                |   1%
  |                                                                      
  |=                                                               |   1%
  |                                                                      
  |=                                                               |   2%
  |                                                                      
  |==                                                              |   2%
  |                                                                      
  |==                                                              |   3%
  |                                                                      
  |==                                                              |   4%
  |                                                                      
  |===                                                             |   4%
  |                                                                      
  |===                                                             |   5%
  |                                                                      
  |====                                                            |   5%
  |                                                                      
  |====                                                            |   6%
  |                                                                      
  |=====                                                           |   6%
  |                                                                      
  |=====                                                           |   7%
  |                                                                      
  |=====                                                           |   8%
  |                                                                      
  |======                                                          |   8%
  |                                                                      
  |======                                                          |   9%
  |                                                                      
  |=======                                                         |   9%
  |                                                                      
  |=======                                                         |  10%
  |                                                                      
  |=======                                                         |  11%
  |                                                                      
  |========                                                        |  11%
  |                                                                      
  |========                                                        |  12%
  |                                                                      
  |=========                                                       |  12%
  |                                                                      
  |=========                                                       |  13%
  |                                                                      
  |=========                                                       |  14%
  |                                                                      
  |==========                                                      |  14%
  |                                                                      
  |==========                                                      |  15%
```

```
|
|===========                                                            |  15%
|
|===========                                                            |  16%
|
|============                                                           |  16%
|
|============                                                           |  17%
|
|============                                                           |  18%
|
|=============                                                          |  18%
|
|=============                                                          |  19%
|
|==============                                                         |  19%
|
|==============                                                         |  20%
|
|==============                                                         |  21%
|
|===============                                                        |  21%
|
|===============                                                        |  22%
|
|================                                                       |  22%
|
|================                                                       |  23%
|
|================                                                       |  24%
|
|=================                                                      |  24%
|
|=================                                                      |  25%
|
|==================                                                     |  25%
|
|==================                                                     |  26%
|
|===================                                                    |  26%
|
|===================                                                    |  27%
|
|===================                                                    |  28%
|
|====================                                                   |  28%
|
|====================                                                   |  29%
|
|=====================                                                  |  29%
|
|=====================                                                  |  30%
|
|=====================                                                  |  31%
```

```
|
|======================                                                 |  31%
|
|======================                                                 |  32%
|
|=======================                                                |  32%
|
|=======================                                                |  33%
|
|=======================                                                |  34%
|
|========================                                               |  34%
|
|========================                                               |  35%
|
|=========================                                              |  35%
|
|=========================                                              |  36%
|
|==========================                                             |  36%
|
|==========================                                             |  37%
|
|==========================                                             |  38%
|
|===========================                                            |  38%
|
|===========================                                            |  39%
|
|============================                                           |  39%
|
|============================                                           |  40%
|
|============================                                           |  41%
|
|=============================                                          |  41%
|
|=============================                                          |  42%
|
|==============================                                         |  42%
|
|==============================                                         |  43%
|
|==============================                                         |  44%
|
|===============================                                        |  44%
|
|===============================                                        |  45%
|
|================================                                       |  45%
|
|================================                                       |  46%
|
|=================================                                      |  46%
```

```
  |
  |=================================                                     |  47%
  |
  |=================================                                     |  48%
  |
  |==================================                                    |  48%
  |
  |==================================                                    |  49%
  |
  |===================================                                   |  49%
  |
  |===================================                                   |  50%
  |
  |===================================                                   |  51%
  |
  |====================================                                  |  51%
  |
  |====================================                                  |  52%
  |
  |=====================================                                 |  52%
  |
  |=====================================                                 |  53%
  |
  |=====================================                                 |  54%
  |
  |======================================                                |  54%
  |
  |======================================                                |  55%
  |
  |=======================================                               |  55%
  |
  |=======================================                               |  56%
  |
  |========================================                              |  56%
  |
  |========================================                              |  57%
  |
  |========================================                              |  58%
  |
  |=========================================                             |  58%
  |
  |=========================================                             |  59%
  |
  |==========================================                            |  59%
  |
  |==========================================                            |  60%
  |
  |==========================================                            |  61%
  |
  |===========================================                           |  61%
  |
  |===========================================                           |  62%
  |
  |============================================                          |  62%
```

```
|
|============================================                          |  63%
|
|============================================                          |  64%
|
|=============================================                         |  64%
|
|=============================================                         |  65%
|
|==============================================                        |  65%
|
|==============================================                        |  66%
|
|===============================================                       |  66%
|
|===============================================                       |  67%
|
|===============================================                       |  68%
|
|================================================                      |  68%
|
|================================================                      |  69%
|
|=================================================                     |  69%
|
|=================================================                     |  70%
|
|=================================================                     |  71%
|
|==================================================                    |  71%
|
|==================================================                    |  72%
|
|===================================================                   |  72%
|
|===================================================                   |  73%
|
|===================================================                   |  74%
|
|====================================================                  |  74%
|
|====================================================                  |  75%
|
|=====================================================                 |  75%
|
|=====================================================                 |  76%
|
|======================================================                |  76%
|
|======================================================                |  77%
|
|======================================================                |  78%
|
|=======================================================               |  78%
```

```
|
|=======================================================               |  79%
|
|========================================================              |  79%
|
|========================================================              |  80%
|
|========================================================              |  81%
|
|=========================================================             |  81%
|
|=========================================================             |  82%
|
|==========================================================            |  82%
|
|==========================================================            |  83%
|
|==========================================================            |  84%
|
|===========================================================           |  84%
|
|===========================================================           |  85%
|
|============================================================          |  85%
|
|============================================================          |  86%
|
|=============================================================         |  86%
|
|=============================================================         |  87%
|
|=============================================================         |  88%
|
|==============================================================        |  88%
|
|==============================================================        |  89%
|
|===============================================================       |  89%
|
|===============================================================       |  90%
|
|===============================================================       |  91%
|
|================================================================      |  91%
|
|================================================================      |  92%
|
|=================================================================     |  92%
|
|=================================================================     |  93%
|
|=================================================================     |  94%
|
|==================================================================    |  94%
```

```
  |
  |=================================================================     |  95%
  |
  |==================================================================    |  95%
  |
  |==================================================================    |  96%
  |
  |===================================================================   |  96%
  |
  |===================================================================   |  97%
  |
  |===================================================================   |  98%
  |
  |====================================================================  |  98%
  |
  |====================================================================  |  99%
  |
  |=====================================================================|  99%
  |
  |=====================================================================| 100%
```

```
# -- Select ingredient and cuisine --
ingredient = 'cheese'
ingredient_2 = 'onion'
ingredient_3 = 'garlic'
list_of_ingredients = c(ingredient, ingredient_2, ingredient_3)
cuisine = 'french'
```

```
# Searching 5 closest words to the ingredirnts
model %>% closest_to(model[[ingredient]],5) #<- set of closest ingredients to "sage"
```

```
##             word similarity to model[[ingredient]]
## 1         cheese                         1.0000000
## 2 anatto_arsenic                         0.7409412
## 3           edam                         0.7315099
## 4        stilton                         0.7278010
## 5      roquefort                         0.7064436
```

```
model %>% closest_to(model[[ingredient_2]], 5) #<- set of closest cuisines to "italian"
```

```
##           word similarity to model[[ingredient_2]]
## 1        onion                           1.0000000
## 2       carrot                           0.8131161
## 3 minced_onion                           0.7697297
## 4     bay_leaf                           0.7560707
## 5       garlic                           0.7514345
```

```
model %>% closest_to(model[[ingredient_3]], 5)
```

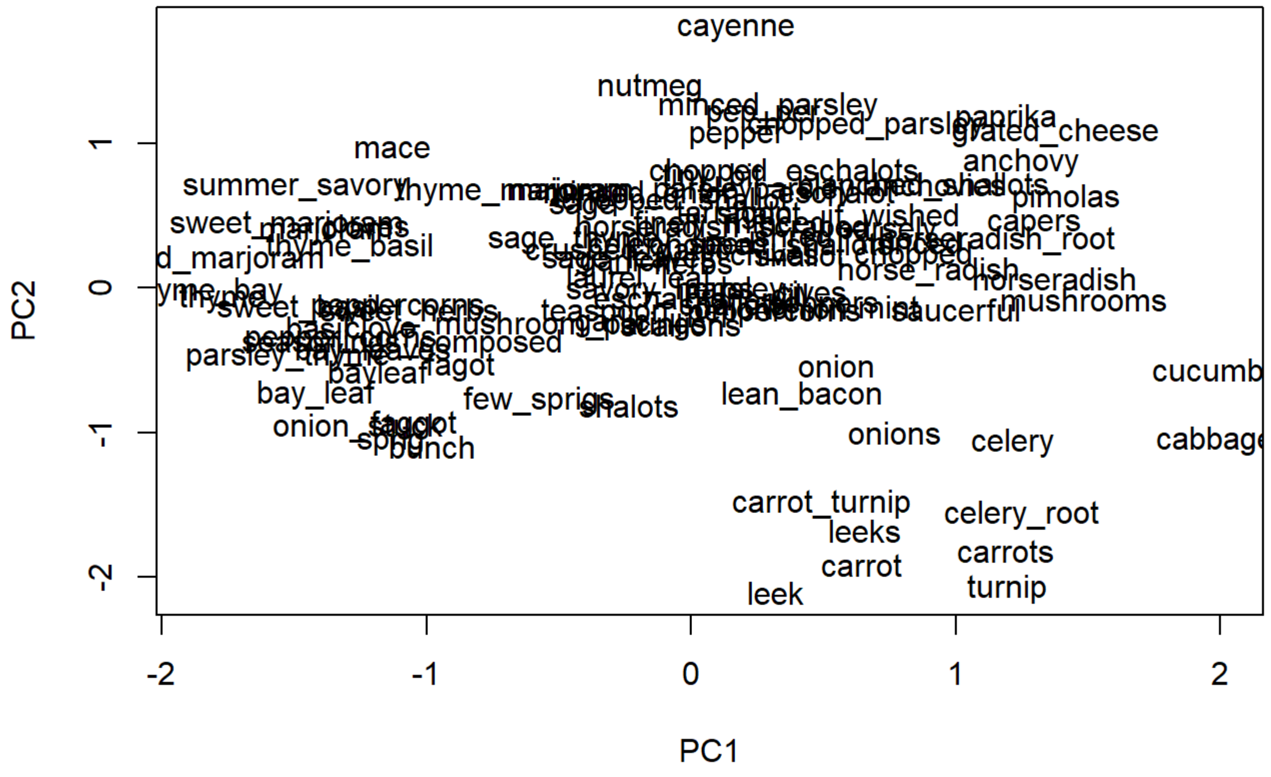
```
##        word similarity to model[[ingredient_3]]
## 1    garlic                           1.0000000
## 2     clove                           0.8197101
## 3   garlick                           0.8143136
## 4 eschalots                           0.7915964
## 5  bay_leaf                           0.7801213
```

It is interesting to see that although onion and garlic are related (Garlic is a species in the onion genus), more ingredients in the oninion genus such as eschalots and shalots are found close to garlic. As for onion, we find ingredients such as carrot, parsely are more likely to appear nearby.

3. Use TSNE to find interesting relationships amongst the different set of ingredients that you have selected.

```
#############################################
#####   Using TSNE to see similarity    #####
#############################################

n_words = 100
closest_ingredients = closest_to(model,model[[list_of_ingredients]], n_words)$word
surrounding_ingredients = model[[closest_ingredients,average=F]]
plot(surrounding_ingredients,method="pca")
```



```
embedding = Rtsne(X = surrounding_ingredients, dims = 2,
                  perplexity = 4,
                  theta = 0.5,
                  eta = 10,
                  pca = TRUE, verbose = TRUE,
                  max_iter = 2000)
```

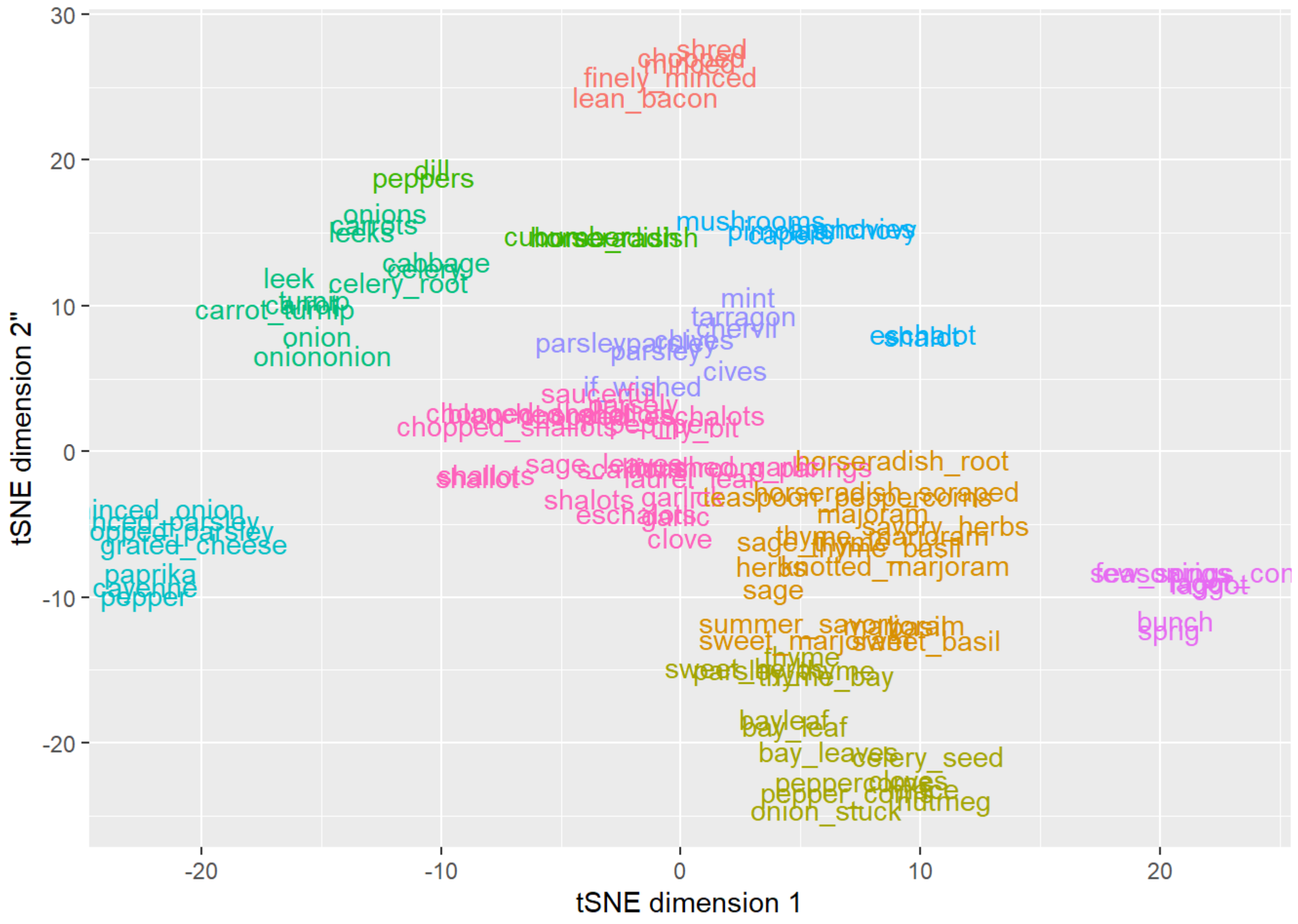
```
## Performing PCA
## Read the 100 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 4.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.02 seconds (sparsity = 0.179400)!
## Learning embedding...
## Iteration 50: error is 65.077509 (50 iterations in 1.78 seconds)
## Iteration 100: error is 65.055531 (50 iterations in 1.11 seconds)
## Iteration 150: error is 63.798792 (50 iterations in 0.24 seconds)
## Iteration 200: error is 63.729035 (50 iterations in 1.28 seconds)
## Iteration 250: error is 63.729058 (50 iterations in 1.26 seconds)
## Iteration 300: error is 1.031264 (50 iterations in 0.37 seconds)
## Iteration 350: error is 0.853733 (50 iterations in 0.19 seconds)
## Iteration 400: error is 0.799741 (50 iterations in 0.17 seconds)
## Iteration 450: error is 0.771162 (50 iterations in 0.32 seconds)
## Iteration 500: error is 0.763693 (50 iterations in 0.41 seconds)
## Iteration 550: error is 0.752403 (50 iterations in 8.18 seconds)
## Iteration 600: error is 0.747638 (50 iterations in 1.26 seconds)
## Iteration 650: error is 0.744448 (50 iterations in 0.49 seconds)
## Iteration 700: error is 0.735965 (50 iterations in 0.58 seconds)
## Iteration 750: error is 0.730766 (50 iterations in 0.13 seconds)
## Iteration 800: error is 0.728383 (50 iterations in 0.17 seconds)
## Iteration 850: error is 0.724239 (50 iterations in 0.32 seconds)
## Iteration 900: error is 0.725665 (50 iterations in 0.17 seconds)
## Iteration 950: error is 0.726563 (50 iterations in 0.17 seconds)
## Iteration 1000: error is 0.727817 (50 iterations in 0.14 seconds)
## Iteration 1050: error is 0.727900 (50 iterations in 0.24 seconds)
## Iteration 1100: error is 0.725179 (50 iterations in 0.25 seconds)
## Iteration 1150: error is 0.720577 (50 iterations in 0.15 seconds)
## Iteration 1200: error is 0.721016 (50 iterations in 0.16 seconds)
## Iteration 1250: error is 0.718056 (50 iterations in 0.25 seconds)
## Iteration 1300: error is 0.717646 (50 iterations in 0.18 seconds)
## Iteration 1350: error is 0.715654 (50 iterations in 0.12 seconds)
## Iteration 1400: error is 0.718064 (50 iterations in 0.10 seconds)
## Iteration 1450: error is 0.715741 (50 iterations in 0.54 seconds)
## Iteration 1500: error is 0.718497 (50 iterations in 0.96 seconds)
## Iteration 1550: error is 0.718302 (50 iterations in 0.14 seconds)
## Iteration 1600: error is 0.717098 (50 iterations in 0.08 seconds)
## Iteration 1650: error is 0.714479 (50 iterations in 0.12 seconds)
## Iteration 1700: error is 0.714173 (50 iterations in 0.12 seconds)
## Iteration 1750: error is 0.714562 (50 iterations in 0.14 seconds)
## Iteration 1800: error is 0.714281 (50 iterations in 0.09 seconds)
## Iteration 1850: error is 0.713614 (50 iterations in 0.09 seconds)
## Iteration 1900: error is 0.713343 (50 iterations in 0.08 seconds)
## Iteration 1950: error is 0.712067 (50 iterations in 0.08 seconds)
## Iteration 2000: error is 0.710268 (50 iterations in 0.08 seconds)
## Fitting performed in 22.74 seconds.
```

```
embedding_vals = embedding$Y
rownames(embedding_vals) = rownames(surrounding_ingredients)

# Looking for clusters for embedding
set.seed(10)
n_centers = 10
clustering = kmeans(embedding_vals,centers=n_centers,
                    iter.max = 5)

# Setting up data for plotting
embedding_plot = tibble(x = embedding$Y[,1],
                        y = embedding$Y[,2],
                        labels = rownames(surrounding_ingredients)) %>%
  bind_cols(cluster = as.character(clustering$cluster))

# Visualizing TSNE output
ggplot(aes(x = x, y=y,label = labels, color = cluster), data = embedding_plot) +
  geom_text() +xlab('tSNE dimension 1') +ylab('tSNE dimension 2"')+theme(legend.position = 'non
e')
```



4. Replace the set of “tastes” by 3 other “orthogonal” dimensions. Before selecting your three words, explore the set of words in the data set. Generate the plot for the set of words. Do they make sense?

```
dim1 = 'baked'
model[[dim1]]
```

```
## A VectorSpaceModel object of  1  words and  100  vectors
##            [,1]       [,2]      [,3]       [,4]      [,5]       [,6]
## [1,] -0.4689357 -0.1241355 0.2844899 -0.1562047 0.1495488 -0.1199075
## attr(,".cache")
## <environment: 0x0000000022077078>
```

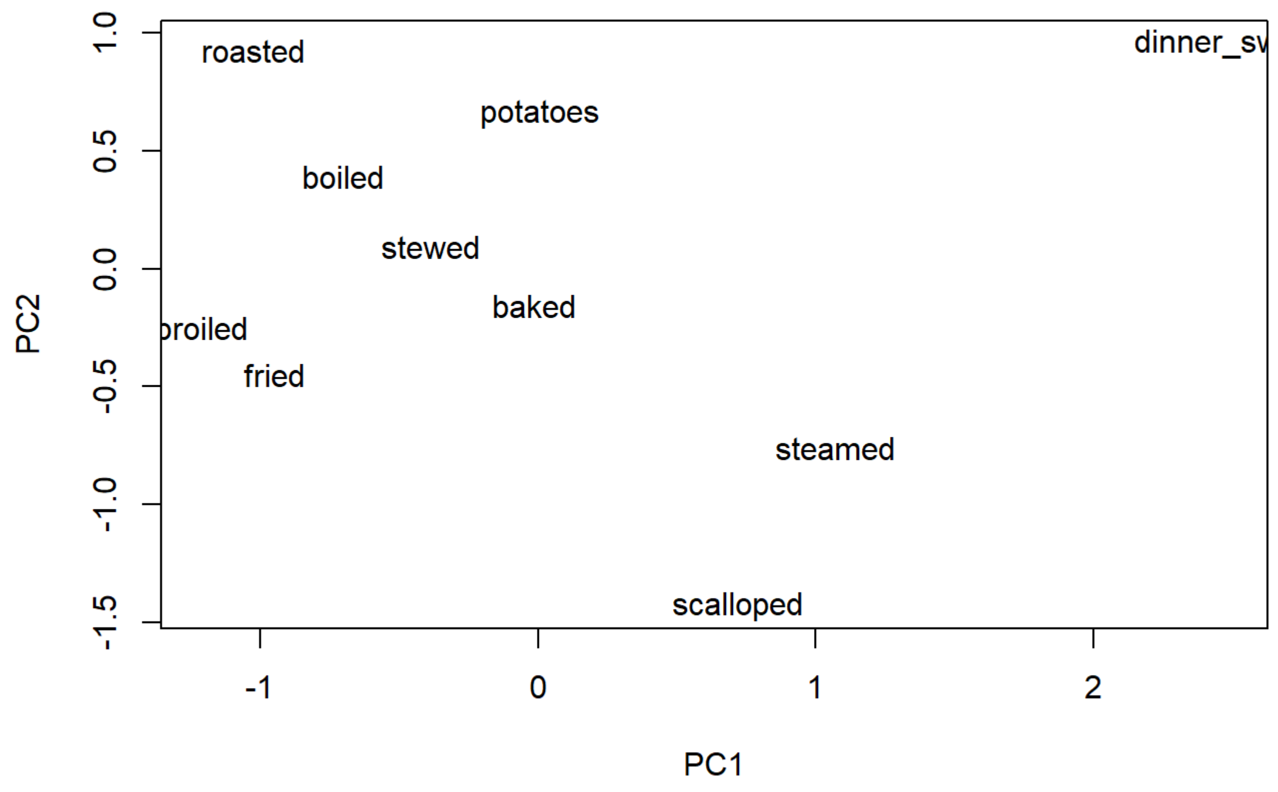
```
# Searching closest 10 words to baked
model %>% closest_to(model[[dim1]],10) #<- set of closest ingredients to "baked"
```

```
##             word similarity to model[[dim1]]
## 1          baked                   1.0000000
## 2        steamed                   0.7735006
## 3         stewed                   0.7644829
## 4      scalloped                   0.7491722
## 5         boiled                   0.7383854
## 6          fried                   0.7193403
## 7        broiled                   0.6867452
## 8        roasted                   0.6772377
## 9   dinner_swiss                   0.6643740
## 10      potatoes                   0.6619407
```

```
closest_baked = closest_to(model,model[[dim1]], 10)$word
closest_baked
```

```
##  [1] "baked"        "steamed"      "stewed"       "scalloped"    "boiled"
##  [6] "fried"        "broiled"      "roasted"      "dinner_swiss" "potatoes"
```

```
surrounding_baked = model[[closest_baked,average=F]]
plot(surrounding_baked,method="pca")
```



```
dim2 = 'fried'
model[[dim2]]
```

```
## A VectorSpaceModel object of  1  words and  100  vectors
##             [,1]       [,2]      [,3]        [,4]        [,5]       [,6]
## [1,] -0.03594827 0.08284434 0.5739112 -0.03868808 -0.01192625 0.03550383
## attr(,".cache")
## <environment: 0x0000000024dd71d8>
```

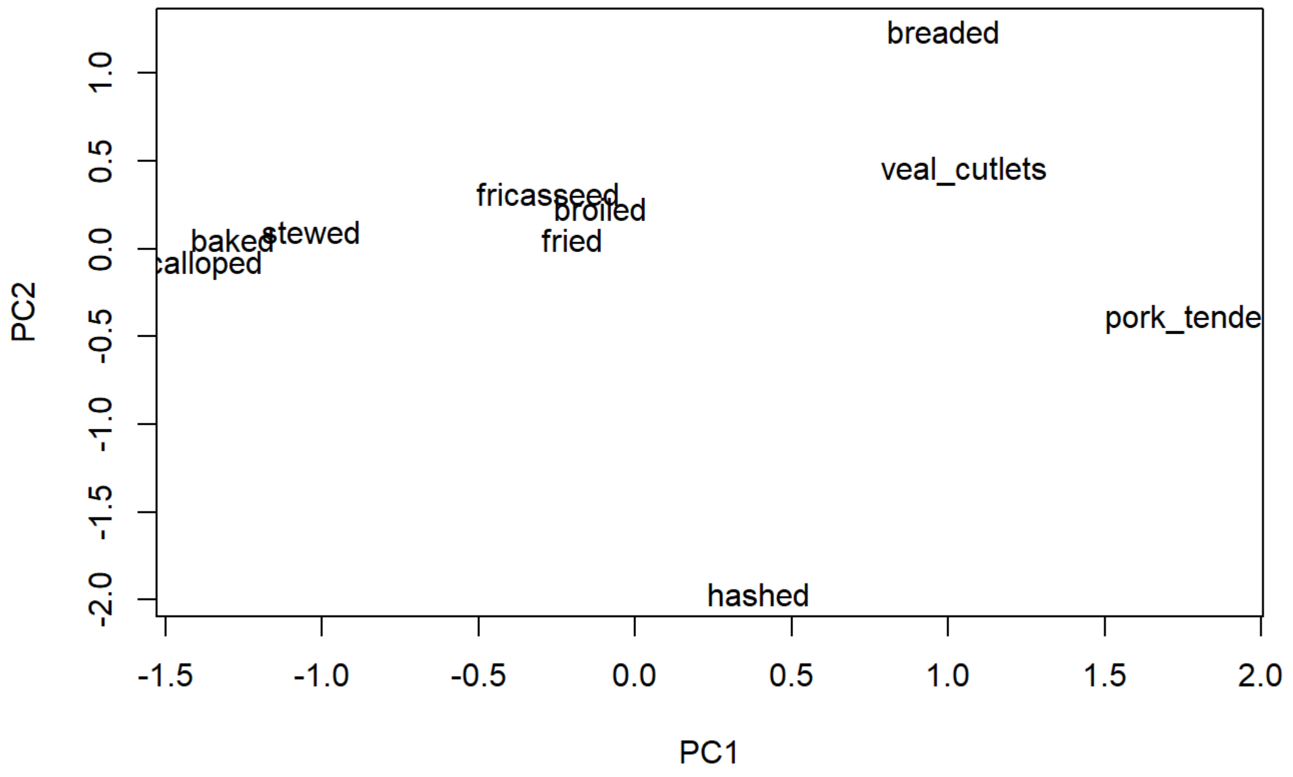
```
# Searching closest 10 words to fried
model %>% closest_to(model[[dim2]],10) #<- set of closest ingredients to "baked"
```

```
##                 word similarity to model[[dim2]]
## 1              fried                   1.0000000
## 2            broiled                   0.7897396
## 3             stewed                   0.7238477
## 4              baked                   0.7193403
## 5          fricasseed                  0.7132304
## 6            breaded                   0.7096002
## 7   pork_tenderloins                   0.7055917
## 8          scalloped                   0.6741050
## 9        veal_cutlets                  0.6730260
## 10            hashed                   0.6726537
```

```
closest_fried= closest_to(model,model[[dim2]], 10)$word
closest_fried
```

```
##  [1] "fried"          "broiled"          "stewed"           "baked"
##  [5] "fricasseed"     "breaded"          "pork_tenderloins" "scalloped"
##  [9] "veal_cutlets"   "hashed"
```

```
surrounding_fried = model[[closest_fried,average=F]]
plot(surrounding_fried,method="pca")
```



```
dim3 = 'roasted'
model[[dim3]]
```

```
## A VectorSpaceModel object of  1  words and  100  vectors
##             [,1]         [,2]      [,3]      [,4]      [,5]       [,6]
## [1,] -0.4685724 -0.009744201 0.8327693 0.0752188 0.1477062 0.03251315
## attr(,".cache")
## <environment: 0x0000000025abe130>
```

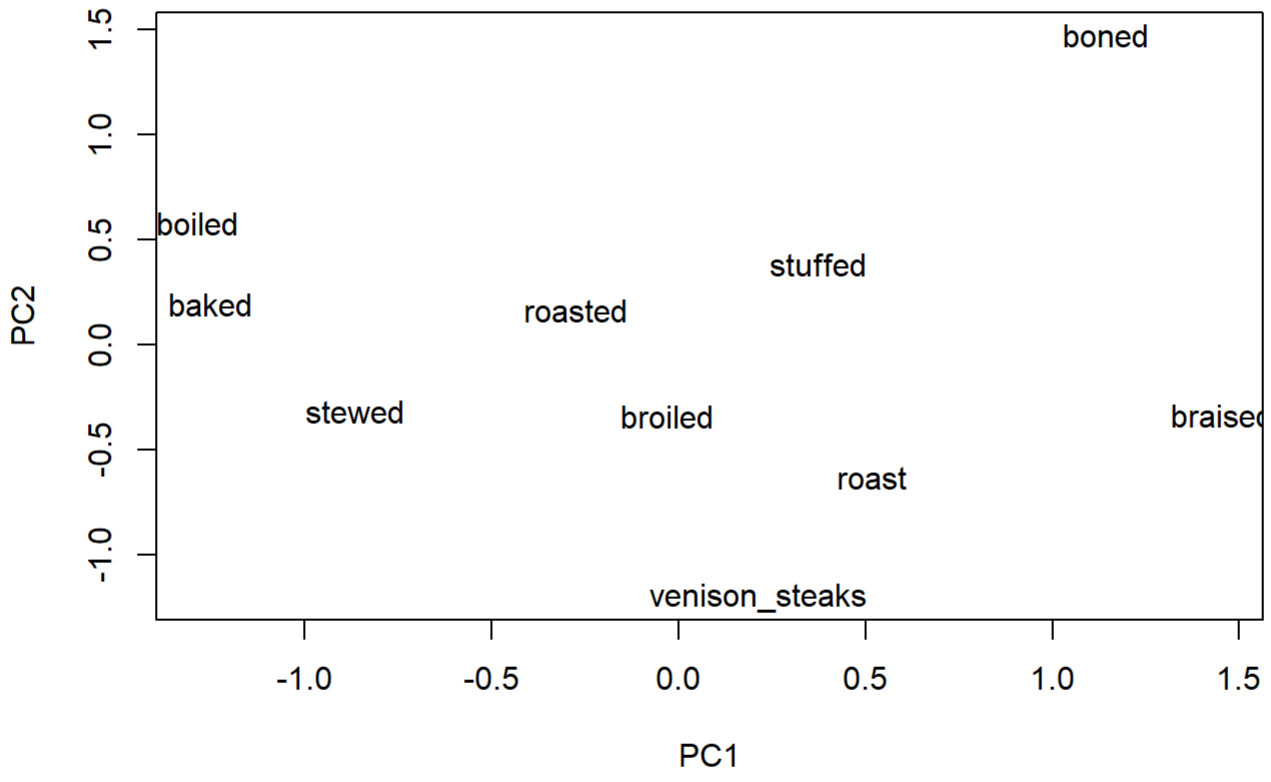
```
# Searching closest 10 words to baked
model %>% closest_to(model[[dim3]],10) #<- set of closest ingredients to "baked"
```

```
##               word similarity to model[[dim3]]
## 1          roasted                   1.0000000
## 2          broiled                   0.7322659
## 3           stewed                   0.7255370
## 4            roast                   0.7193496
## 5           boiled                   0.6853909
## 6            baked                   0.6772377
## 7          stuffed                   0.6700680
## 8   venison_steaks                   0.6695662
## 9            boned                   0.6530345
## 10         braised                   0.6418657
```

```
closest_roasted = closest_to(model,model[[dim3]], 10)$word
closest_roasted
```

```
##  [1] "roasted"        "broiled"        "stewed"         "roast"
##  [5] "boiled"         "baked"          "stuffed"        "venison_steaks"
##  [9] "boned"          "braised"
```

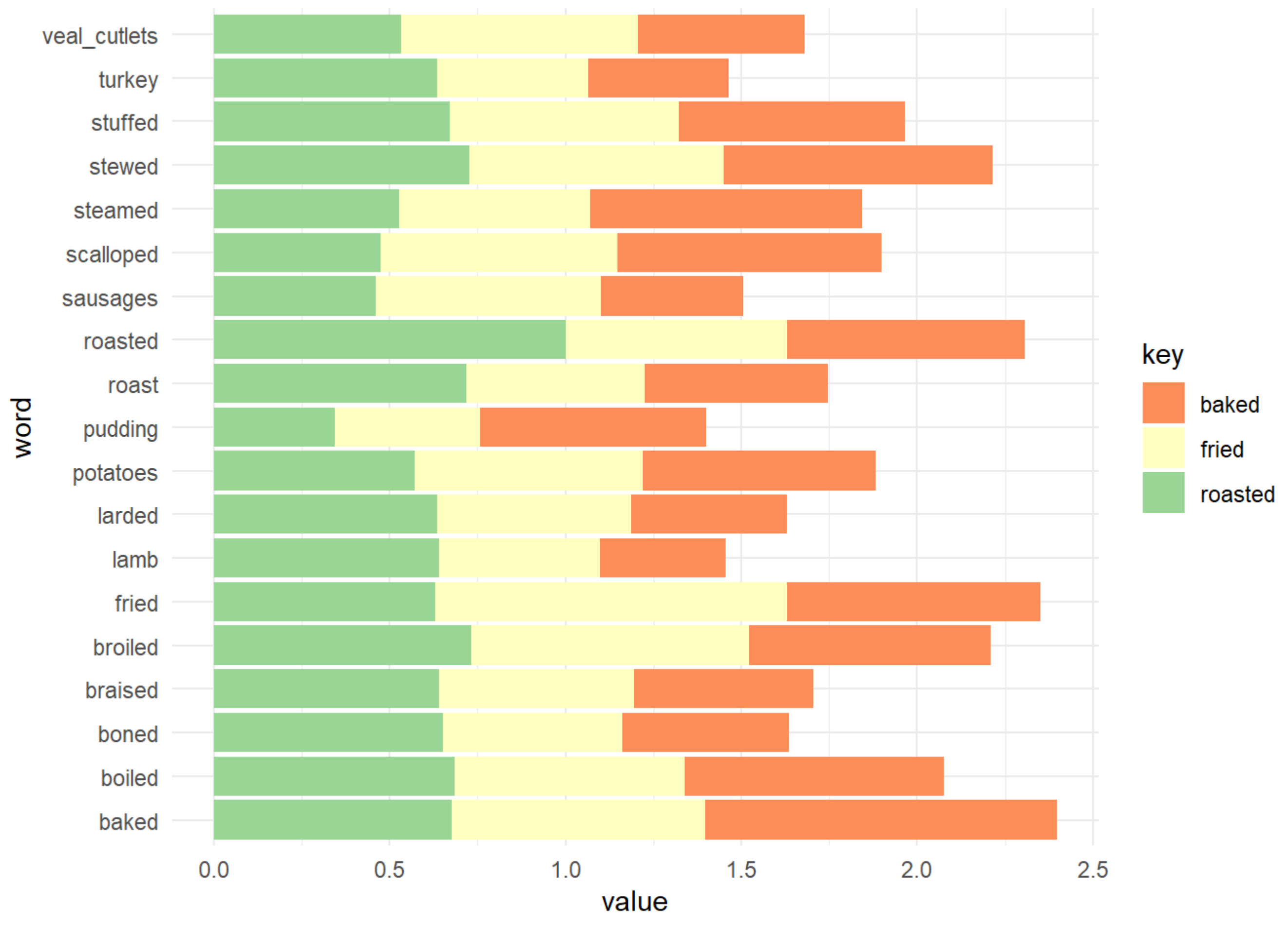
```
surrounding_roasted = model[[closest_roasted,average=F]]
plot(surrounding_roasted,method="pca")
```



Set of words that are close to baked, fried and roasted make sense. For example, one of the words close to fried is fricassee, whihc is a dish of fried pieces of meat served in a thick white sauce. One of the words close to roasted is vension, which is normally cooked by roasting.

```
############################################
#####   Plotting 3 cooking method  Dimensions  #####
############################################
# We can plot along mltiple dimensions:
tastes = c('baked','fried','roasted')
common_similarities_tastes = model[1:3000,]%>% cosineSimilarity( model[[tastes,average=F]])
high_similarities_to_tastes = common_similarities_tastes[rank(-apply(common_similarities_tastes,
1,max)) < 20,]

# - Plotting
high_similarities_to_tastes %>%
  as_tibble(rownames='word') %>%
  #filter( ! (is.element(word,set_of_tastes))) %>%
  #mutate(total = salty+sweet+savory+bitter+sour) %>%
  #mutate( sweet=sweet/total,salty=salty/total,savory=savory/total,bitter=bitter/total, sour = s
our/total) %>%
  #select(-total) %>%
  gather(key = 'key', value = 'value',-word) %>%
  ggplot(aes(x = word,
             y = value,
             fill = key)) + geom_bar(stat='identity') +
  coord_flip() + theme_minimal() + scale_fill_brewer(palette='Spectral')
```



5. Try different combinations for the analogic reasoning. (Given the data is not munged I expect that some of the relations will be odd).

```
model %>% closest_to("bread") # words associated with haelthy living (if not a bit outdated)
```

```
##            word similarity to "bread"
## 1         bread             1.0000000
## 2          loaf             0.6856392
## 3           rye             0.6676849
## 4      zwieback             0.6506997
## 5       biscuit             0.6423614
## 6          buns             0.6239088
## 7   whole_wheat             0.6233483
## 8         cakes             0.6224941
## 9  graham_bread             0.6166332
## 10   plain_buns             0.6029271
```

```
model %>% closest_to(~("bread" - "flour" ),15) # number 7 is cravings
```

```
##                  word similarity to ("bread" - "flour")
## 1               bread                         0.4996818
## 2            symbolic                         0.3951312
## 3                 616                         0.3630503
## 4  beliefs_concerning                         0.3579539
## 5                 604                         0.3505320
## 6        639_imitated                         0.3497197
## 7          sandwiches                         0.3410409
## 8              crusts                         0.3311744
## 9                 131                         0.3271351
## 10       attend_zuÃ±i                         0.3268602
## 11     ceremonies_608                         0.3216516
## 12           imitated                         0.3196894
## 13          by_clowns                         0.3183408
## 14         fireplaces                         0.3153471
## 15      pueblo_origin                         0.3152224
```

```
model %>% closest_to(~"chocolate" + ("cake"- "sugar"),15)
```

```
##           word similarity to "chocolate" + ("cake" - "sugar")
## 1          cake                                      0.7827496
## 2     chocolate                                      0.7762970
## 3        almond                                      0.7337314
## 4   sponge_cake                                      0.7047372
## 5  burnt_almond                                      0.6943996
## 6       quaking                                      0.6866486
## 7      cocoanut                                      0.6854037
## 8      streusel                                      0.6825046
## 9  devil's_food                                      0.6817650
## 10        dover                                      0.6784392
## 11      eclairs                                      0.6745960
## 12    ice_cream                                      0.6675368
## 13    frostings                                      0.6587910
## 14 lady_fingers                                      0.6495710
## 15    maccaroon                                      0.6486554
```

```
model %>% closest_to(~"milk" + ("wine" - "sweet"),15)
```

```
##             word similarity to "milk" + ("wine" - "sweet")
## 1           milk                                 0.6963920
## 2           wine                                 0.6488929
## 3  rich_unskimmed                                0.6112445
## 4      white_wine                                0.5897353
## 5    brandybrandy                                0.5743165
## 6       port_wine                                0.5729987
## 7  unskimmed_milk                                0.5560637
## 8         sherry                                 0.5541857
## 9        winewine                                0.5530507
## 10    mulled_wine                                0.5528987
## 11        brandy                                 0.5466458
## 12    pale_sherry                                0.5465539
## 13          whey                                 0.5444640
## 14           rum                                 0.5423299
## 15       whiskey                                 0.5356587
```

6. Given the set of analogic words and the values that are close to them (from 6), what are some interesting relations you find? List 2-3 of them - explain why they are interetsing to you.

- I am interested to see what is bread minus flour since bread is made of flour. The analogic reasoning returns stale. Staling, or "going stale", is a chemical and physical process in bread and similar foods that reduces their palatability - stale bread is dry and hard.

In plain English, chocolate" + ("cake"- "sugar") means chocolate cake without sugar. It is interesting to see the analogic reasoning returns sponge, eclairs, almond and eggless.

7. The starter code sets bundle_ngrams=1 which means we are interested in single word embeddings and not common pairs, if we set bundle_ngrams=2 what common pairs of words do you get - list 10.

```r
if (!file.exists("cookbooks.zip")) {
  download.file("http://archive.lib.msu.edu/dinfo/feedingamerica/cookbook_text.zip","cookbooks.z
ip")
}
unzip("cookbooks.zip",exdir="cookbooks")
if (!file.exists("cookbooks.txt")) prep_word2vec(origin="cookbooks",destination="cookbooks.txt",
lowercase=T,bundle_ngrams=2)
```

In cookbooks.txt, we fid many word pairs are combined by _
Some examples are: american_woman's, american_woman's, sleeping_rooms, clam_chowder, smppth_paste, wooden_spoon, chopping_bowl, sour_milk, buckwheat_cakes, china_closet, lemon_juice