Math 377 Project

Professor Bradley Warner

Wednesday, August 19, 2015

Introduction

This project will guide you through a small research project. We will be building a simple probability based spell checker in R. The objectives of this project are:
1. Read and summarize a research paper
2. Find and experiment with existing functions in R
3. Find existing code and pseudo code
4. Acquire appropriate materials
5. Implement in R
6. Test and validate

To complete this project you will need to run the 32-bit version of R in RStudio. That is because the `qdap` package has the capability to open interactive windows, widgets. This relies on RJava and thus our, USAFA, 32-bit java. We will not use the interactive windows in the package, which require `Rjava`, but the package will not load if our versions of java does not match our version of R.

Authorized Resources: Anyone and anything.

Points: 75

Due: Lesson 38 at close of business

Deliverables

You must use reproducible research by creating an RMarkdown file where your compiled code and data is visible to the reader. There is an RMarkdown information sheet on the course website under reference materials to help get you started. You should add the following elements to your file to show at the top of the compiled document:

Title
Name
Section
Documentation

You will complete each of the sections below. You will turn in an html file with your section and name as the title. So for example, I would turn in T2Warner.html. You will submit the document to Assignment Dropbox folder on our course website. The are some suggested completion dates as well to help keep you on track.

Components

`1.` (5 pts) (September 8) Research the history of spell checkers using Wikipedia. Briefly, one paragraph, summarize your reading.

`2.` (10pts) (September 16) The package `qdap` in R has a spell checker. Load the package and use it in your RMarkdown file to get the spelling of the following using the function `check_spelling` and the default options.

```
c("Robots are evl creatres and derv exterimanitation.","tes")
```

Notice that word `desr` is probably `deserve` but it did not appear in the list of suggestions. This is because `deserve` is too far away from `derv`. Run the following command:

```
adist("derv","deserve")
```

Now change the appropriate option in `check_spelling` to get `deserve` as a suggestion.

`3.` (10pts) (October 15) Read Peter Norvig's article. Yes, the code is in Python but it gives us the ideas we need. Reading the article, we are going to use Bayes Theorem to find a probability for the suggestion given the typed word. You should spend some time thinking about Professor Norvig's claim that $P(c|w)$ is difficult to find empirically. Instead we need to find $P(c)$, the probability of correctly spelled word, and $P(w|c)$. To understand his code, we will work with a smaller data set. Go the the Gutenburg Project website and download the book the *Journal of a Soldier* as a text file. We need to read this data into R. It is a text file with line breaks so we need to use the `readLines` command. Here is my command for reading the first 10 lines from the file both from the website and my local hard drive.

```r
readLines("http://www.gutenberg.org/files/49163/49163-0.txt",n=10)
```

```
##  [1] "ï»¿The Project Gutenberg EBook of Journal of a Soldier of the Seventy-First"
##  [2] "or Glasgow Regiment Highland Light Infant, by Anonymous"
##  [3] ""
##  [4] "This eBook is for the use of anyone anywhere at no cost and with"
##  [5] "almost no restrictions whatsoever.  You may copy it, give it away or"
##  [6] "re-use it under the terms of the Project Gutenberg License included"
##  [7] "with this eBook or online at www.gutenberg.org/license"
##  [8] ""
##  [9] ""
## [10] "Title: Journal of a Soldier"
```

```r
readLines("~/Classes/Math 377/Fall 2015/Project/Journal of a Soldier.txt",n=10)
```

```
##  [1] "ï»¿The Project Gutenberg EBook of Journal of a Soldier of the Seventy-First"
##  [2] "or Glasgow Regiment Highland Light Infant, by Anonymous"
##  [3] ""
##  [4] "This eBook is for the use of anyone anywhere at no cost and with"
##  [5] "almost no restrictions whatsoever.  You may copy it, give it away or"
##  [6] "re-use it under the terms of the Project Gutenberg License included"
##  [7] "with this eBook or online at www.gutenberg.org/license"
##  [8] ""
##  [9] ""
## [10] "Title: Journal of a Soldier"
```

I am going to save the first 100 rows to an object and then see what I have to do to clean it up.

```r
test_sample<-readLines("~/Classes/Math 377/Fall 2015/Project/Journal of a Soldier.txt",n=100)
str(test_sample)
```

```
##  chr [1:100] "ï»¿The Project Gutenberg EBook of Journal of a Soldier of the Seventy-First" ...
```

This is a vector of characters that I need to collapse to one vector using paste.

```
test_sample<-paste(test_sample,collapse=" ")
str(test_sample)
```

```
##  chr "ï»¿The Project Gutenberg EBook of Journal of a Soldier of the Seventy-First or Glasgow Regimen
```

Next, in Professor Norvig's paper, he converts everything to lower case.

```
test_sample<-tolower(test_sample)
str(test_sample)
```

```
##  chr "ï»¿the project gutenberg ebook of journal of a soldier of the seventy-first or glasgow regimen
```

The next part is a little tricky. Professor Norvig is using a regular expression to parse the character string. Luckily, R has a function called `strsplit` that will do this for us. It returns a list so we need to make it a vector.

```
test_sample<-strsplit(test_sample, "[^a-z]+")
test_sample<-unlist(test_sample)
str(test_sample)
```

```
##  chr [1:363] "" "the" "project" "gutenberg" "ebook" "of" ...
```

Wow, that was powerful. Notice that there are several odd entries such as blank, www, or single letters. We could do more processing or simply hope that in a large corpus, these will be so rare as to not impact our answer. It appears that Professor Norvig assumes the later as he does no more data cleaning. Now let's table our data to get the frequencies and also the probabilities.

```
table(test_sample)
```

```
## test_sample
##                             a       account      accounts      accuracy
##             1             8             2             1             1
##          adam     adventure advertisement        almost    alteration
##             1             1             1             1             1
##       america      american           and     anecdotes     anonymous
##             1             1            14             1             2
##        anyone     anywhere       archive          army       arrival
##             1             1             1             1             4
##     ascertain            at        attack      attempts        author
##             1             6             1             1             1
##     available          away         ayres             b        battle
##             1             1             1             1             2
##          been     behaviour        belief        better         black
##             2             1             1             1             1
##         brash         brian        bridge        buenos           but
##             1             1             1             1             1
##            by          cape     character       charles         chuck
##             3             2             1             1             1
##            co           coe     confirmed      contents  conversation
##             1             1             1             1             1
##          copy    correction          cost         could          date
##             1             1             1             1             1
##     departure   distributed         ebook     edinburgh     education
##             1             1             5             2             2
##      encoding       english      expected    expressing         fails
##             1             2             1             1             1
##       farther           few          file         first     following
##             1             1             1             3             1
##           for          from             g       general      generous
##             6             4             1             1             1
##    generously          give       glasgow          good         greif
##             1             1             4             1             1
##     gutenberg           has          have      highland       himself
##             4             3             2             3             1
##           his          hope          http            ib        images
##             4             1             1             1             1
##            in  inaccuracies      included        infant      infantry
##             2             1             1             1             2
##   inhabitants      internet            is          isle            it
##             1             1             1             1             4
##         james         joins       journal          june            la
##             1             1             5             1             1
##      language     libraries       license         light        london
##             2             1             2             3             1
##          made       madeira         march           may         monte
##             2             1             1             1             1
##          more           net            no       nothing   observation
##             1             1             3             1             1
##       obvious            of        online            or           org
##             1            25             2             5             1
##           own          page         pains     parentage         party
##             1             1             1             1             1
##        passed          pgdp         plata        priest        prince
##             1             1             1             2             1
##       printed      prisoner     prisoners       private      produced
##             1             1             1             1             2
##       project  proofreading    publishers            re    recruiting
##             3             1             1             1             1
##      regiment       related       release  restrictions        result
##             3             1             1             1             1
##         river             s         sails           set       seventy
##             1             2             2             1             3
##     situation       soldier         south       spanish         stage
##             1             5             2             2             1
##         start    statements        street  sufficiently          tait
##             1             1             1             1             1
##         taken          team         terms            th          than
##             2             1             1             1             2
##          that           the          them         there          this
##             1            28             1             1             4
##         title            to          town         under          upon
##             1             2             2             2             1
##           use           utf        verbal         video             w
##             2             1             1             1             1
##           was          what    whatsoever     whitelock     whittaker 
##             1             1             1             1             1 
##         wight       william          with        writer           www 
##             1             1             3             2             2 
##           you 
##             1
```

```
prop.table(table(test_sample))
```

```
## test_sample
##                           a       account      accounts      accuracy 
##   0.002754821   0.022038567   0.005509642   0.002754821   0.002754821 
##          adam     adventure advertisement        almost    alteration 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##       america      american           and     anecdotes     anonymous 
##   0.002754821   0.002754821   0.038567493   0.002754821   0.005509642 
##        anyone      anywhere       archive          army       arrival 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.011019284 
##     ascertain            at        attack      attempts        author 
##   0.002754821   0.016528926   0.002754821   0.002754821   0.002754821 
##     available          away         ayres             b        battle 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.005509642 
##          been     behaviour        belief        better         black 
##   0.005509642   0.002754821   0.002754821   0.002754821   0.002754821 
##         brash         brian        bridge        buenos           but 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##            by          cape     character       charles         chuck 
##   0.008264463   0.005509642   0.002754821   0.002754821   0.002754821 
##            co           coe     confirmed      contents  conversation 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##          copy    correction          cost         could          date 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##     departure   distributed         ebook     edinburgh     education 
##   0.002754821   0.002754821   0.013774105   0.005509642   0.005509642 
##      encoding       english      expected    expressing         fails 
##   0.002754821   0.005509642   0.002754821   0.002754821   0.002754821 
##       farther           few          file         first     following 
##   0.002754821   0.002754821   0.002754821   0.008264463   0.002754821 
##           for          from             g       general      generous 
##   0.016528926   0.011019284   0.002754821   0.002754821   0.002754821 
##    generously          give       glasgow          good         greif 
##   0.002754821   0.002754821   0.011019284   0.002754821   0.002754821 
##     gutenberg           has          have      highland       himself 
##   0.011019284   0.008264463   0.005509642   0.008264463   0.002754821 
##           his          hope          http            ib        images 
##   0.011019284   0.002754821   0.002754821   0.002754821   0.002754821 
##            in  inaccuracies      included        infant      infantry 
##   0.005509642   0.002754821   0.002754821   0.002754821   0.005509642 
##   inhabitants      internet            is          isle            it 
##   0.002754821   0.002754821   0.002754821   0.002754821   0.011019284 
##         james         joins       journal          june            la 
##   0.002754821   0.002754821   0.013774105   0.002754821   0.002754821 
##      language     libraries       license         light        london 
##   0.005509642   0.002754821   0.005509642   0.008264463   0.002754821
##         made       madeira         march           may         monte 
##  0.005509642   0.002754821   0.002754821   0.002754821   0.002754821 
##         more           net            no       nothing   observation 
##  0.002754821   0.002754821   0.008264463   0.002754821   0.002754821 
##      obvious            of        online            or           org 
##  0.002754821   0.068870523   0.005509642   0.013774105   0.002754821 
##          own          page         pains     parentage         party 
##  0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##       passed          pgdp         plata        priest        prince 
##  0.002754821   0.002754821   0.002754821   0.005509642   0.002754821 
##      printed      prisoner     prisoners       private      produced 
##  0.002754821   0.002754821   0.002754821   0.002754821   0.005509642 
##      project proofreading    publishers            re    recruiting 
##  0.008264463   0.002754821   0.002754821   0.002754821   0.002754821 
##     regiment       related       release  restrictions        result 
##  0.008264463   0.002754821   0.002754821   0.002754821   0.002754821 
##        river             s         sails           set       seventy 
##  0.002754821   0.005509642   0.005509642   0.002754821   0.008264463 
##    situation       soldier         south       spanish         stage 
##  0.002754821   0.013774105   0.005509642   0.005509642   0.002754821 
##        start    statements        street  sufficiently          tait 
##  0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##        taken          team         terms            th          than 
##  0.005509642   0.002754821   0.002754821   0.002754821   0.005509642 
##         that           the          them         there          this 
##  0.002754821   0.077134986   0.002754821   0.002754821   0.011019284 
##        title            to          town         under          upon 
##  0.002754821   0.005509642   0.005509642   0.005509642   0.002754821 
##          use           utf        verbal         video             w 
##  0.005509642   0.002754821   0.002754821   0.002754821   0.002754821 
##          was          what    whatsoever     whitelock     whittaker 
##  0.002754821   0.002754821   0.002754821   0.002754821   0.002754821 
##        wight       william          with        writer           www 
##  0.002754821   0.002754821   0.008264463   0.005509642   0.005509642 
##          you 
##  0.002754821
```

I will save the data, sort it, and finally save the result as a character vector

```
probs_of_word<-sort(prop.table(table(test_sample)),decreasing=TRUE)
freq_word<-names(sort(prop.table(table(test_sample)), decreasing = TRUE))
head(freq_word)
```

```
## [1] "the" "of"  "and" "a"   "at"  "for"
```

```
head(probs_of_word)
```

```
## test_sample
##        the         of        and          a         at        for 
## 0.07713499 0.06887052 0.03856749 0.02203857 0.01652893 0.01652893
```

Based on this work, `the` is the most frequently used word and has a probability of occurring of .077. Thus we now have $P(c)$.

Your assignments is to now read in the entire document, Journal of a Soldier, and report the 10th most common word and its probability of occurrence.

4. (10pts) (October 30) Finding $P(w|c)$ is difficult. Professor Norvig made, what he called, the `trivial` model in the he looked at the distance from the given word to the closest words in the corpus and assumed that words with a distance of 1 were infinitely more likely than words with a distance of 2. Also, he wrote his own code to calculate the distance between two words but luckily for us, as we saw in part 2, R has a function called `adist` that does this for us. Thus Professor Norvig looked for words with a distance of zero and if it existed returned this as the correct spelling. If this was not the case, he found the words with distance one and returned the one that is most probable. If there was not a word or set of words with a distance of 1, he went to a distance of 2 and repeated. This stopped after 2 because he claimed that 98% of spelling errors were within a distance of 2. Let's implement this in R.

First we need to find the distance between our word and the words in the sorted list, this is what I called freq_word above. As an example, suppose my word is "tha". I would type:

```r
adist("tha",freq_word)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]    1    3    3    2    3    3    5    6    3     7     6     4     6
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
## [1,]     8     3     3     2     3     5     2     6     4     3     7
##      [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35]
## [1,]     8     7     3     7     9     5     4     4     9     8     7
##      [,36] [,37] [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46]
## [1,]     3     3     7     7     7     4     6     6     8     3     5
##      [,47] [,48] [,49] [,50] [,51] [,52] [,53] [,54] [,55] [,56] [,57]
## [1,]     4     6     4     1     2     3     5     3     5     3     3
##      [,58] [,59] [,60] [,61] [,62] [,63] [,64] [,65] [,66] [,67] [,68]
## [1,]     8     7     3     8    12     6     8     6     7     8     6
##      [,69] [,70] [,71] [,72] [,73] [,74] [,75] [,76] [,77] [,78] [,79]
## [1,]     7     6     4     8     4     7     4     8     3     5     3
##      [,80] [,81] [,82] [,83] [,84] [,85] [,86] [,87] [,88] [,89] [,90]
## [1,]     7     6     5     4     4     4     6     6     3     7     5
##      [,91] [,92] [,93] [,94] [,95] [,96] [,97] [,98] [,99] [,100] [,101]
## [1,]     4     3     3     9     7    11     4     9     4      5      4
##      [,102] [,103] [,104] [,105] [,106] [,107] [,108] [,109] [,110] [,111]
## [1,]      8     10      8      7     10      5      5      3      4      9
##      [,112] [,113] [,114] [,115] [,116] [,117] [,118] [,119] [,120] [,121]
## [1,]      3      6      8     10      4      4      5      7      4      3
##      [,122] [,123] [,124] [,125] [,126] [,127] [,128] [,129] [,130] [,131]
## [1,]      3      5     11      8      5      9      7      3      4      5
##      [,132] [,133] [,134] [,135] [,136] [,137] [,138] [,139] [,140] [,141]
## [1,]      5      4      2      8      6      6      5      3      5      4
##      [,142] [,143] [,144] [,145] [,146] [,147] [,148] [,149] [,150] [,151]
## [1,]      3      5     10      7      3      3      4      5      8      5
##      [,152] [,153] [,154] [,155] [,156] [,157] [,158] [,159] [,160] [,161]
## [1,]      6      4      4      6      6      8      9      6     11      9
##      [,162] [,163] [,164] [,165] [,166] [,167] [,168] [,169] [,170] [,171]
## [1,]      3      9      6      6     11      6      5      3      7      4
##      [,172] [,173] [,174] [,175] [,176] [,177] [,178] [,179] [,180] [,181]
## [1,]      4      9      5     11      3      2      4      1      1      2
##      [,182] [,183] [,184] [,185] [,186] [,187] [,188] [,189] [,190] [,191]
## [1,]      3      4      4      3      5      5      3      3      2      8
##      [,192] [,193] [,194] [,195] [,196]
## [1,]      8      7      4      6      3
```

Notice this gives us all the distances. Next we want to extract those words that meet a specified distance, Professor Norvig used 2.

```
freq_word[adist("tha",freq_word)<=2]
```

```
##  [1] "the"  "a"    "this" "has"  "than" "to"   "la"   "team" "th"   "that"
## [11] "them" "what"
```

Now the problem is that some of these words could have distance of 0, 1, or 2. Based on Professor Norvig's suggestion we should ignore higher distances. For example, if we have a distance of 1, we should ignore all distances of 2. We will now implement this idea:

```
freq_word[adist("tha",freq_word)<=min(adist("tha",freq_word),2)]
```

```
## [1] "the"  "than" "th"   "that"
```

Since the list is ordered by frequency, we would suggest the first element.

```
freq_word[adist("tha",freq_word)<=min(adist("tha",freq_word),2)][1]
```

```
## [1] "the"
```

Write a function called, my_spell_checker that takes as input the character vector, the vector of sorted words, this is your dictionary, and an option for distance with a default of 2. In your code, you need to account for the issue that you might not find a word that is within the range. In that case, your code should return the original word. Read in the entire file Journal of a Soldier, I call it freq_word in my example below, and run your function on the following:

```
my_spell_checker("off",freq_word)
my_spell_checker("tha",freq_word)
my_spell_checker("drvvve",freq_word)
my_spell_checker("you're",freq_word)
my_spell_checker("hgkdjurhc",freq_word)
my_spell_checker("hgkdjurhc",freq_word,range=6)
```

`5.` (15pts) (November 17) I like that we have a list of suggestions but without knowing $P(w|c)$ we cannot calculate the probabilities. Let's modify Professor Norvig's code by instead of assuming an infinite probability let's assume that $P(w|c)$ for a distance of 1 has a probability of 3 times that of a distance of 2, and likewise a distance of 3 has 3 times the probability of 2. This could continue indefinitely but at some point we need to stop. Let's stop at 20 and call everything with a distance of 20 or higher the same probability. If the distance is 0, then we just return the word. For the rest, we have $P(w|c) = p$ for a distance of 1, $P(w|c) = p/3$ for a distance of 2, $P(w|c) = p/3^2$ for a distance of 3, and on. Find p and then use this to write a function that returns the top three words, based on $P(c|w) = P(w|c)P(c)$, as a default with the option to change this value. Call the function, `my_suggestions`. For reference, the probability $P(w|c)$ for a distance of 2 is 0.22222.

Now I will run my function below as an example

```
my_suggestions("akk",probs_of_word2,p_of_w_given_c,2,3)
        and           a          at
0.006613722 0.004873876 0.001832024
my_suggestions("akk",probs_of_word2,p_of_w_given_c,2,5)
        and           a          at          as         all
0.006613722 0.004873876 0.001832024 0.001520926 0.001111889
my_suggestions("akk",probs_of_word2,p_of_w_given_c,3,5)
        and         the           a          of          to
0.006613722 0.004919964 0.004873876 0.002319796 0.002139282
my_suggestions("the",probs_of_word2,p_of_w_given_c,2,3)
 "the"
my_suggestions("thethethethethethe",probs_of_word2,p_of_w_given_c,2,3)
 "thethethethethethe"
my_suggestions("bradley",probs_of_word2,p_of_w_given_c,2,3)
       badly
1.152216e-05
```

The first option is the word, the second is the table of probabilities, the third is the conditional probabilities, the fourth the maximum distance, and the last the number of words to report.

Using your own code, perform the equivalent to the following statements:

```
my_suggestions("off",probs_of_word2,p_of_w_given_c,2,3)
my_suggestions("tha",probs_of_word2,p_of_w_given_c,2,3)
my_suggestions("drvvve",probs_of_word2,p_of_w_given_c,2,3)
my_suggestions("you're",probs_of_word2,p_of_w_given_c,2,3)
my_suggestions("hgkdjurhc",probs_of_word2,p_of_w_given_c,2,3)
my_suggestions("hgkdjurhc",probs_of_word2,p_of_w_given_c,6,3)
```

`6.` (December 4) The last thing we need to do is validate the spell checker. This is what Professor Norvig did in the final phase. We will only do an abbreviated evaluation.

`a.` (10 pts) First read into R Professor Norvig's big.txt document, on the course website, and process it as we did above for the Journal of a Soldier. We want to use this bigger document to improve the accuracy. Use the new word frequency table in your spell checker from part 4 on the following words:

```
off
tha
drvvve
you're
hgkdjurhc  (with default settings)
hgkdjurhc  (with range=6)
```

There are still some problems with the spell checker. But we will proceed any way.

`b.` (10pts) There is a file on the course website called test_data.txt that contains only up through the letter d of Professor Norvig's test data. The first few lines are below.

```
'access': 'acess'
'accessing': 'accesing'
'accommodation':'accomodation acommodation acomodation'
```

The correct spelling is before the colon and the incorrect is after. Read the data in and create a vector of the strings with the correct and all the incorrect spellings. This vector should be of length 48.

We next need to create a vector with the correct spelling and another with the incorrect spellings. This is not an easy matter since some words have multiple misspelled words. This is good practice because in analysis getting data into your computer in a clean and efficient manner is difficult. You may want to use functions such as gsub, strsplit, and unlist to split the data apart. You want to also remove leading and trailing blank spaces. You want two vectors, the first has the answers and the second has the common misspellings. Each of these vectors will be of length 78 because that is the total number of misspelled words in the text file. As an example, for the three lines above with the words access, accessing, and accommodation, your answer vector would be

```
access
accessing
accommodation
accommodation
accommodation
```

and your example vector would be

```
acess
accesing
accomodation
acommodation
acomodation
```

Print out the 53rd through the 70th value of each vector. Make sure you include your code to clean the data.

`c.` (5 pts) After cleaning your data, run the data through your function my_spell_checker and compare with the correct answer, this is easier if you use the sapply function. Report your error rate.