Math 377 Project Solution

Professor Bradley Warner Section M2A

Monday, June 08, 2015

Documentation: None

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 psuedo 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. You start by opening a new R Markdown file. You should have the following elements:

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.

Components

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

Solution

Grading: 5 - Reads well and discusses; 4 - Minor issues, 3- OK, 2 - Weak.

Sample solution:

Research on spell checkers started in the late 1950s. They orginally started as stand alone products but then moved into word processing applications and eventually most computer application that require writing, such as web browsers and blogs. The spell checkers usually have a dictionary, in English the optimum appears to be around 90,000 words, and words are compared with this dictionary one at a time. This means that context is not accounted for in most spell checkers. The next stage is to account for the context in a spell checker.

2. (10pts) 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.

Solution

3 pts

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

```
##   row word.no not.found        suggestion     more.suggestions
## 1 1   3       evl              evil           ev, evils, elva, e, eel, ell, erl, esl, eva, eve, evy
## 2 1   4       creatres         creatures      creates, create, creature, cremates, creators, cerates
## 3 1   6       derv             dev            derive, de, dear, deer, dere, derk, derm, deva, devi, 
## 4 1   7       exterimanitation exterminations experimentation, experimentations, electrification, exe
## 5 2   1       tes              teds           tees, tegs, test, ties, tyes, teas, tens, tess, tews, 
```

2 pts

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

```
##      [,1]
## [1,]    3
```

5 pts

```
check_spelling(c("Robots are evl creatres and derv exterimanitation.","tes"),range=3)
```

```
##   row word.no not.found        suggestion    more.suggestions
## 1 1   3       evl              evil          ev, evils, evilly, ervils, evelyn, evenly, eviler, evol
## 2 1   4       creatres         creatures     creates, creators, cremates, create, creature, cerates,
## 3 1   6       derv             dev           derive, derived, dervish, decurve, deprave, deprive, de
## 4 1   7       exterimanitation extermination exterminations, experimentation, exterminating, experime
## 5 2   1       tes              teas          tegs, tess, test, ties, toes, tyes, teds, tees, tens, t
```

3. (10pts) 10 pts - Correct, 8 pts - Wrong but code looks reasonable, 5 pts - Some code but can't figure out what is wrong, 2 pts - Some effort

Read in the entire document, Journal of a Soldier, and report the 10th most common word and its probability of occurrence.

Solution

```
test_sample2<-readLines("~/Classes/Math 377/Fall 2015/Project/Journal of a Soldier.txt")
test_sample2<-paste(test_sample2,collapse=" ")
test_sample2<-tolower(test_sample2)
test_sample2<-strsplit(test_sample2, "[^a-z]+")
test_sample2<-unlist(test_sample2)
probs_of_word2<-sort(prop.table(table(test_sample2)),decreasing=TRUE)
freq_word2<-names(sort(prop.table(table(test_sample2)), decreasing = TRUE))
head(probs_of_word2,n=10)
```

```
## test_sample2
##        the         of        and         to         we          a
## 0.06641952 0.03131724 0.02976175 0.02888030 0.02226946 0.02193244
##          i         in        was       were
## 0.01799186 0.01765484 0.01436238 0.01280689
```

4. (10pts)

Write a function called, my_spell_checker that takes as input the character vector, the vector of sorted words, 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:

Solution

```
my_spell_checker("off",freq_word) 2 pts
my_spell_checker("tha",freq_word) 1 pt
my_spell_checker("drvvve",freq_word) 2 pts
my_spell_checker("you're",freq_word) 2 pts
my_spell_checker("hgkdjurhc",freq_word) 1 pt
my_spell_checker("hgkdjurhc",freq_word,range=6) 2 pts
```

```
my_spell_checker<-function(word,sorted_words,range=2){
    ans<-sorted_words[adist(word,sorted_words)<=min(adist(word,sorted_words),range)][1]
    if(is.na(ans))ans<-word
    return(ans)
}
my_spell_checker("off",freq_word2)
```

```
## [1] "off"
```

```
my_spell_checker("tha",freq_word2)
```

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

```
my_spell_checker("drvvve",freq_word2)
```

```
## [1] "drove"
```

```r
my_spell_checker("you're",freq_word2)
```

```
## [1] "your"
```

```r
my_spell_checker("hgkdjurhc",freq_word2)
```

```
## [1] "hgkdjurhc"
```

```r
my_spell_checker("hgkdjurhc",freq_word2,range=6)
```

```
## [1] "hour"
```

5. (15pts)

My function below as an example

```r
p_of_w_given_c<-(1/1.5)*((1/3)^(seq(1:20)-1))
my_suggestions<-function(word,prob_words,cond_probs,my_range=2,n=3){
    dist1<-adist(word,names(prob_words))
    temp1<-min(dist1,my_range)
    if(temp1==0)return(word)
    ans<-names(prob_words)[dist1<=temp1][1]
    if(is.na(ans)){
        ans<-word
        return(ans)}
    ans_temp<-numeric(0)
    for(i in 1:my_range){
        ans_temp<-c(ans_temp,prob_words[dist1==i]*cond_probs[i])
    }
    ans<-sort(ans_temp,decreasing=TRUE)
    min_n<-min(length(ans),n)
    return(ans[1:min_n])
}
```

```r
my_suggestions("akk",probs_of_word2,p_of_w_given_c,2,3)
```

```
##         and           a          at
## 0.006613722 0.004873876 0.001832024
```

```r
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
```

```r
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
```

```r
my_suggestions("the",probs_of_word2,p_of_w_given_c,2,3)
```

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

```r
my_suggestions("thethethethethethe",probs_of_word2,p_of_w_given_c,2,3)
```

```
## [1] "thethethethethethe"
```

```r
my_suggestions("bradley",probs_of_word2,p_of_w_given_c,2,3)
```

```
##        badly
## 1.152216e-05
```

Solution

Points are 2, 2, 3, 3, 2, 3

```r
my_suggestions("off",probs_of_word2,p_of_w_given_c,2,3)
```

```
## [1] "off"
```

```r
my_suggestions("tha",probs_of_word2,p_of_w_given_c,2,3)
```

```
##         the          to           a
## 0.044279677 0.006417846 0.004873876
```

```r
my_suggestions("drvvve",probs_of_word2,p_of_w_given_c,2,3)
```

```
##        drove        drive
## 6.913299e-05 5.761082e-06
```

```r
my_suggestions("you're",probs_of_word2,p_of_w_given_c,2,3)
```

```
##         your
## 0.0002650098
```

```r
my_suggestions("hgkdjurhc",probs_of_word2,p_of_w_given_c,2,3)
```

```
## [1] "hgkdjurhc"
```

```r
my_suggestions("hgkdjurhc",probs_of_word2,p_of_w_given_c,6,3)
```

```
##         hour        hours      hundred
## 1.493614e-06 7.112447e-07 3.556224e-07
```

6. 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)
```

```r
test_sample3<-readLines("~/Classes/Math 377/Fall 2015/Project/big.txt")
test_sample3<-paste(test_sample3,collapse=" ")
test_sample3<-tolower(test_sample3)
test_sample3<-strsplit(test_sample3, "[^a-z]+")
test_sample3<-unlist(test_sample3)
probs_of_word3<-sort(prop.table(table(test_sample3)),decreasing=TRUE)
freq_word3<-names(sort(prop.table(table(test_sample3)), decreasing = TRUE))
head(probs_of_word3,n=10)
```

```
## test_sample3
##         the          of         and          to          in           a
## 0.072406664 0.036212380 0.034663458 0.026025867 0.019949606 0.019139860
##        that          he         was          it
## 0.011320157 0.011219731 0.010323129 0.009663571
```

Solution

1 pt each and 5 pts for last one.

```r
my_spell_checker("off",freq_word3)
```

```
## [1] "off"
```

```r
my_spell_checker("tha",freq_word3)
```

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

```r
my_spell_checker("drvvve",freq_word3)
```

```
## [1] "drove"
```

```r
my_spell_checker("you're",freq_word3)
```

```
## [1] "your"
```

```r
my_spell_checker("hgkdjurhc",freq_word3)
```

```
## [1] "hgkdjurhc"
```

```r
my_spell_checker("hgkdjurhc",freq_word3,range=6)
```

```
## [1] "duroc"
```

The last one is interesting, I want to explore

```r
my_suggestions("hgkdjurhc",probs_of_word3,p_of_w_given_c,5,3)
```

```
##        duroc     honduras
## 1.489291e-08 7.446453e-09
```

```r
my_suggestions("hgkdjurhc",probs_of_word3,p_of_w_given_c,6,3)
```

```
##      hundred        hours         hour
## 5.684126e-07 4.120371e-07 3.896977e-07
```

So 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 common misspelled words. This is not an easy matter. 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 to vectors, the first has the answers and the second has the common misspellings. For the three lines above your answer vector would be

```
access
accessing
accommodation
accommodation
accommodation
```

and your example vector would be

```
acess
accesing
accomodation
acommodation
acomodation
```

The two vectors should have length 48. Print out the 53rd through the 70th value of each vector. Make sure you include your code to clean the data.

```r
test_data<-readLines("~/Classes/Math 377/Fall 2015/Project/test_data.txt")
#test_data<-paste(test_data,collapse=" ")
test_data<-gsub("'","",test_data)
test_data<-strsplit(test_data,":")
test_data<-unlist(test_data)
answers<-test_data[seq(1,95,by=2)]
examples<-test_data[seq(2,96,by=2)]
#Remove leading blank space
examples<-gsub("^ ","",examples)
examples<-gsub("\\s+$","",examples)
num_of_words<-numeric(0)
for(i in 1:length(examples)){
    res<-gregexpr(" ",examples[i])[[1]]
    if(res[1]==-1)temp=0
    else temp=length(res)
    temp<-temp+1
    num_of_words<-c(num_of_words,temp)
}
final_ex<-unlist(strsplit(examples," "))
final_ans<-rep(answers,num_of_words)
final_ex[53:70]
```

```
##  [1] "concider"     "concidérable" "contenpted"   "contende"
##  [5] "contended"    "contentid"    "cartains"     "certans"
##  [9] "courtens"     "cuaritains"   "curtans"      "curtians"
## [13] "curtions"     "descide"      "descided"     "definately"
## [17] "difinately"   "defenition"
```

```r
final_ans[53:70]
```

```
##  [1] "consider"     "considerable" "contented"    "contented"
##  [5] "contented"    "contented"    "curtains"     "curtains"
##  [9] "curtains"     "curtains"     "curtains"     "curtains"
## [13] "curtains"     "decide"       "decided"      "definitely"
## [17] "definitely"   "definition"
```

Solution

```r
cbind(final_ans,final_ex)[53:70,]
```

```
##       final_ans      final_ex
##  [1,] "consider"     "concider"
##  [2,] "considerable" "conciderable"
##  [3,] "contented"    "contenpted"
##  [4,] "contented"    "contende"
##  [5,] "contented"    "contended"
##  [6,] "contented"    "contentid"
##  [7,] "curtains"     "cartains"
##  [8,] "curtains"     "certans"
##  [9,] "curtains"     "courtens"
## [10,] "curtains"     "cuaritains"
## [11,] "curtains"     "curtans"
## [12,] "curtains"     "curtians"
## [13,] "curtains"     "curtions"
## [14,] "decide"       "descide"
## [15,] "decided"      "descided"
## [16,] "definitely"   "definately"
## [17,] "definitely"   "difinately"
## [18,] "definition"   "defenition"
```

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. For the example above you would want to check each of the accommodation misspellings against the correct spelling and report.

Solution

```
sapply(final_ex,my_spell_checker,sorted_words=freq_word3)
```

```
##               acess            accesing        accomodation
##            "access"           "acceding"     "accommodation"
##        acommodation          acomodation              acount
##     "accommodation"      "accommodation"             "count"
##              adress                adres         addresable
##             "dress"              "acres"        "addresable"
##             aranged            arrainged          arragment
##          "arranged"           "arranged"      "arrangement"
##            articals                 annt               anut
##          "articles"               "anna"              "nut"
##                arnt            auxillary             avaible
##              "aunt"           "axillary"        "available"
##              awfall                afful           basicaly
##              "wall"              "awful"        "basically"
##             begining              benifit           benifits
##         "beginning"            "benefit"         "benefits"
##             beetween              bicycal            bycicle
##           "between"            "bicycle"          "bicycle"
##              bycycle              biscits           biscutes
##           "bicycle"           "biscuits"         "disputes"
##              biscuts             bisquits           buiscits
##          "biscuits"           "biscuits"         "biscuits"
##             buiscuts                biult                cak
##          "biscuits"                "but"              "can"
##               carrer             cemetary           semetary
##            "career"           "cemetery"        "secretary"
##             centraly              cirtain          chalenges
##           "central"            "certain"       "challenges"
##            chalenges               chaper          chaphter
##        "challenges"            "chapter"          "chapter"
##              chaptur               choise            chosing
##           "chapter"             "choose"          "closing"
##          clearical           comittee           compair
##         "clerical"        "committee"         "company"
##          completly           concider      conciderable
##       "completely"         "consider"    "considerable"
##         contenpted           contende         contended
##        "contented"        "contended"       "contended"
##          contentid           cartains           certans
##        "contented"         "captains"        "certains"
##           courtens         cuaritains           curtans
##         "countess"         "curtains"        "curtains"
##           curtians           curtions           descide
##           "curtis"         "portions"          "decide"
##           descided         definately        difinately
##          "decided"       "definitely"      "definitely"
##         defenition        defenitions       discription
##       "definition"      "definitions"     "description"
##           desicate          dessicate        dessiccate
##         "delicate"         "delicate"      "dessiccate"
##   diagrammaticaally           diffrent            dirven
## "diagrammaticaally"        "different"           "given"
```

```
my_guess<-sapply(final_ex,my_spell_checker,sorted_words=freq_word3)
final_ans==my_guess
```

```
##             acess          accesing      accomodation      acommodation
##              TRUE             FALSE              TRUE              TRUE
##        acomodation            acount            adress             adres
##              TRUE             FALSE             FALSE             FALSE
##        addresable           aranged         arrainged         arragment
##             FALSE              TRUE              TRUE              TRUE
##           articals              annt              anut              arnt
##              TRUE             FALSE             FALSE              TRUE
##          auxillary           avaible            awfall             afful
##             FALSE              TRUE             FALSE              TRUE
##           basicaly          begining           benifit          benifits
##              TRUE              TRUE              TRUE              TRUE
##            beetween           bicycal           bycicle           bycycle
##              TRUE              TRUE              TRUE              TRUE
##            biscits          biscutes           biscuts          bisquits
##              TRUE             FALSE              TRUE              TRUE
##           buiscits          buiscuts             biult               cak
##              TRUE              TRUE             FALSE             FALSE
##             carrer          cemetary          semetary          centraly
##              TRUE              TRUE             FALSE             FALSE
##            cirtain        chalenges         chalenges            chaper
##              TRUE              TRUE              TRUE              TRUE
##           chaphter           chaptur            choise           chosing
##              TRUE              TRUE             FALSE             FALSE
##          clearical          comittee           compair         completly
##              TRUE              TRUE             FALSE              TRUE
##           concider      conciderable        contenpted          contende
##              TRUE              TRUE              TRUE             FALSE
##          contended         contentid          cartains           certans
##             FALSE              TRUE             FALSE             FALSE
##           courtens        cuaritains           curtans          curtians
##              FALSE              TRUE              TRUE             FALSE
##           curtions           descide          descided        definately
##              FALSE              TRUE              TRUE              TRUE
##          difinately        defenition       defenitions       discription
##               TRUE              TRUE              TRUE              TRUE
##           desicate         dessicate        dessiccate diagrammaticaally
##              FALSE             FALSE             FALSE             FALSE
##           diffrent            dirven
##               TRUE             FALSE
```

```r
sum(final_ans!=my_guess)/length(final_ans)
```

```
## [1] 0.3717949
```

```r
sum(final_ans==my_guess)/length(final_ans)
```

```
## [1] 0.6282051
```