Free text analysis for epidemiologic surveillance and early anomaly detection

Joe Brew (UF-ID: 0402-8902)

Contents

Background

This document serves as the "final project" for PHC 7065 ("Data Management"), taught by Dr. Arbi Ben Abdallah.

Data source

I use data from the Florida Department of Health's Electronic Surveillance System for the Early Notification of Community-based Epidemics (ESSENCE). I am authorized to use these data as surveillance epidemiologist for the Florida Department of Health in Alachua County.[1]

Problem

Florida's Electronic Surveillance System for the Early Notification of Community-based Epidemics (ESSENCE) classifies chief complaint data on a daily basis for all participating EC facilities within Florida (approximately 79% of all facilities). County and State epidemiologists rely on these classifications to detect abnormalities (outbreaks, accidents, exposures, etc.). However, the ESSENCE's classification algorithms fail to classify approximately half of all cases into any category at all. Here's are some examples (using hypothetical data):

Chief complaint / discharge diagnosis	ESSENCE classification
Cough congestion fever	Influenza like illness
Vomiting abdominal pain	Gastrointestinal
Lethargic swollen feet	None
Mouth pain	None
Ate at Domino's pizza	None

By not classifying many observations, the ESSENCE system allows health threats to go undetected. This is a problem from a surveillance perspective because it means that epidemiologists lose the opportunity to intervene early and effectively. The current algorithms use fixed disease-related terms, meaning that even in the event of 10 patients presenting at the same time with "pizza" as the chief complaint, no alarm would be set off. Misclassification (or in this case, non-classification) means that Florida epidemiologists may detect a new or unusual health event late (or not at all), leaving Floridians vulnerable to emerging health threats.

Solution

I have assembled the chief complaints and discharge diagnoses of all ED visits among Alachua County residents since January 2012 (data ~ 75 mb). My code "product" is a program which takes raw chief complaints and discharge diagnoses from any day, and spits back a list of terms that appeared in reverse order of "likelihood" to appear (ie, need to investigate). In other words, it gives the surveillance epidemiologist needed information about what happened yesterday, and how unusual it was.

Code

Data preparation

First, I prepare the data, extracting only the date and chief complaint / discharge diagnosis for the 349,828 visits of interest.

[1] As you can imagine, I'm not authorized to share the raw or identifiable data.

```
#####
# READ IN AND PREPARE RAW EMERGENCY ROOM SURVEILLANCE DATA
#####
if(file.exists('/media/joebrew/JB/fdoh/private/surv/historical/ccdd_only.csv')){
  df <- read.csv('/media/joebrew/JB/fdoh/private/surv/historical/ccdd_only.csv',
                 stringsAsFactors = FALSE)
} else {
  df <- read.csv('/media/joebrew/JB/fdoh/private/surv/historical/alless1213updated.csv',
                 stringsAsFactors = FALSE)
  # SUBSET TO INCLUDE ONLY DATE AND CHIEF COMPLAINT / DISCHARGE DIAGNOSIS (CCDD)
df <- df[,c('Date', 'CCDD')]

# SAVE A SMALLER SUBSET TO SAVE TIME
write.csv(df, '/media/joebrew/JB/fdoh/private/surv/historical/ccdd_only.csv',
          row.names = FALSE)
}

# FORMAT DATE
df$Date <- as.Date(df$Date, format = '%Y-%m-%d')

# ORDER BY DATE
df <- df[rev(order(df$Date)),]
```

Here is what the raw data look like.

	Date	CCDD
320135	2015-01-22	ABDOMINAL PAIN |
320133	2015-01-22	FV | flu
320132	2015-01-22	TOOTH PAIN | DENTAL DISORDER NOS; NO PROC/PATIENT DECISION
320131	2015-01-22	CHEST PAIN | Chest pain with low risk for cardiac etiology
320130	2015-01-22	ABDOMINAL PAIN | Abdominal pain acute epigastric
320129	2015-01-22	ALLERGIC RASH | Allergic reaction initial encounter

Next, I strip the CCDD of characters of all non-alphanumeric characters, remove trailing or leading white space, and remove pipes.

```
# REMOVE ALL NON ALPHANUMERICS
df$CCDD <- gsub("[^a-zA-Z0-9]"," ",df$CCDD)

# REMOVE PIPES
df$CCDD <- gsub('[|]', '', df$CCDD)

# DEFINE AND USE FUNCTION TO STRIP TRAILING OR LEADING WHITESPACE
strip_white <- function(character_string){
  gsub('^\\s+|\\s+$', '', character_string)
}
df$CCDD <- strip_white(df$CCDD)
```

Here is what the raw data look like after cleaning.

	Date	CCDD
320135	2015-01-22	ABDOMINAL PAIN
320133	2015-01-22	FV flu
320132	2015-01-22	TOOTH PAIN DENTAL DISORDER NOS NO PROC PATIENT DECISION
320131	2015-01-22	CHEST PAIN Chest pain with low risk for cardiac etiology
320130	2015-01-22	ABDOMINAL PAIN Abdominal pain acute epigastric
320129	2015-01-22	ALLERGIC RASH Allergic reaction initial encounter

Data aggregation

The next phase of the process consists of getting the words that appeared in yesterday's chief complaints and discharge diagnoses, as well as the words that appeared in ALL days prior to then.

```
# DEFINE A DAY THAT WE'RE INVESTIGATING (CALLED YESTERDAY)
yesterday <- max(df$Date)

# GET A VECTOR OF ALL THE WORDS THAT APPEARED YESTERDAY
words <- unlist(strsplit(as.character(
                                              toupper(df$CCDD[which(df$Date == yesterday)])), " "))

# How many words appeared yesterday in free text?
length(words)
```

```
## [1] 2042
```

```
# GET A CORRESPONDING VECTOR OF ALL WORDS THAT HAVE EVER APPEARED (EXCLUDING YESTERDAY)
wordsBL <- unlist(strsplit(as.character(
                                                toupper(df$CCDD[which(df$Date != yesterday)])), " "))

# How many words appeared prior to yesterday in free text?
length(wordsBL)
```

```
## [1] 1980300
```

```
# CONSTRUCT A DATAFRAME OF OUR WORD COUNTS (YESTERDAY ONLY)
wordsDF <- as.data.frame(table(words))
colnames(wordsDF) <- c("word","count")

# CONSTRUCT A DATAFRAME OF BASELINE WORD COUNTS (PRIOR TO YESTERDAY)
wordsBLDF <- as.data.frame(table(wordsBL))
colnames(wordsBLDF) <- c("word","count")

# CHANGE TO CHARACTERS
wordsBLDF$word <- as.character(wordsBLDF$word)
wordsDF$word <- as.character(wordsDF$word)

# IN YESTERDAY'S DATA, GET THE EXPECTED ACCOUNT
dividend <- length(unique(df$Date)) - 1

wordsDF$expected <- NA
```

```
for (i in 1:nrow(wordsDF)){

  the_word <- wordsDF$word[i]

  sub_data <- wordsBLDF[which(wordsBLDF$word == the_word),]

  if(nrow(sub_data) > 0){
      replacement <- sub_data$count / dividend
  } else{
    replacement <- NA
  }
  wordsDF$expected[i] <- replacement
}
```

Our results look like this:

	word	count	expected
449	WITH	7	4.0149254
450	WITHDRAWAL	1	0.3441128
451	WITHOUT	1	0.8532338
452	WOUND	4	4.8333333
453	WRIST	5	2.9585406
454	YEAST	1	0.2330017

Prioritizing action

Now that we have measures of observed vs. expected, we can calculate an "unlikelihood ratio" which is simply the observed number of times a word appears over the "expected"[2]

```
# DEFINE AN "UNLIKELIHOOD RATIO" COLUMN
wordsDF$likelihood <- wordsDF$count / wordsDF$expected

# ORDER BY UNLIKELIHOOD
wordsDF <- wordsDF[rev(order(wordsDF$likelihood)),]

# Remove NA expected rows (this will need to be tweaked when this goes into production)
wordsDF <- wordsDF[which(!is.na(wordsDF$expected)),]
```

We can make this a bit more elegant by running chi-squared tests.

```
# DEFINE A FUNCTION TO GET THE P VALUE USING CHI-SQUARED OF EACH
# WORD'S PROBABILITY
chi_squared <- function(row = 1){
  sub_data <- wordsDF[row,]
  # success
  x <- c(sub_data$count, sub_data$expected * dividend)
  # trials
  n <- c(sum(wordsDF$count), sum(wordsBLDF$count))
```

[2] "Expected" is understood here as the number of times the word previously appeared divided by the number of days.

```
  # test
  suppressWarnings(
    temp <- prop.test(x = x, n = n)
    )

  # Extract p-value
  temp$p.value
}

# LOOP OVER EACH ROW TO GET P VALUE
wordsDF$p <- NA
for (i in 1:nrow(wordsDF)){
  wordsDF$p[i] <- chi_squared(i)
}
```

Our results look like this:

	word	count	expected	likelihood	p
420	TOWN	1	0.0008292	1206	0
407	TESTI	1	0.0008292	1206	0
403	TAPE	1	0.0008292	1206	0
305	PARTIALLY	1	0.0008292	1206	0
239	LAPAROSCOPY	1	0.0008292	1206	0
237	LACERTAION	1	0.0008292	1206	0

Now, I can subset my data to only look at events with three or more occurrences yesterday, and prioritize action based on the p-values.

```
# Subset
action <- wordsDF[which(wordsDF$count >=3),]

# Order
action <- action[order(action$p),]
```

My final dataframe looks like this - it is a table of terms of concern, ranked by lowest p-value (ie, likelihood to occur).

	word	count	expected	likelihood	p
113	DIALYSIS	3	0.1981758	15.138075	0.0000060
220	INITIAL	23	7.7611940	2.963462	0.0000304
7	ABNORMAL	8	1.6650083	4.804781	0.0001526
141	ENCOUNTER	23	8.5978441	2.675089	0.0002659
47	BEAT	4	0.5149254	7.768116	0.0003472
310	PELVIC	8	1.8300166	4.371545	0.0005062
330	PSYCHIATRIC	3	0.3051410	9.831522	0.0005782
234	LABS	3	0.3308458	9.067669	0.0011267
253	LUMBAR	6	1.3092869	4.582647	0.0023205
225	IRREGULAR	4	0.6542289	6.114068	0.0028366
136	EKG	3	0.3747927	8.004425	0.0028661
87	CLEARANCE	3	0.3764511	7.969163	0.0029566
343	RENAL	5	1.0381426	4.816294	0.0045957

	word	count	expected	likelihood	p
258	MEDICAL	3	0.5140962	5.835484	0.0196606
14	ACCIDENT	12	4.7636816	2.519060	0.0207339
363	SHORTNESS	19	9.0961857	2.088788	0.0302066
335	RASH	13	5.5472637	2.343498	0.0310949
60	BREATH	19	9.2164179	2.061539	0.0349299
137	ELBOW	5	1.4004975	3.570160	0.0355977
122	DIZZY	4	0.9809287	4.077768	0.0381054

Data visualization

Since the data are still somewhat difficult to manage, we can use visualization to get a better idea of what should be of concern. The below visualizatoin shows words in our dataset, with size a reflection of the inverse likelihood of occurrence (ie, the bigger the word, the more we should be concerned).

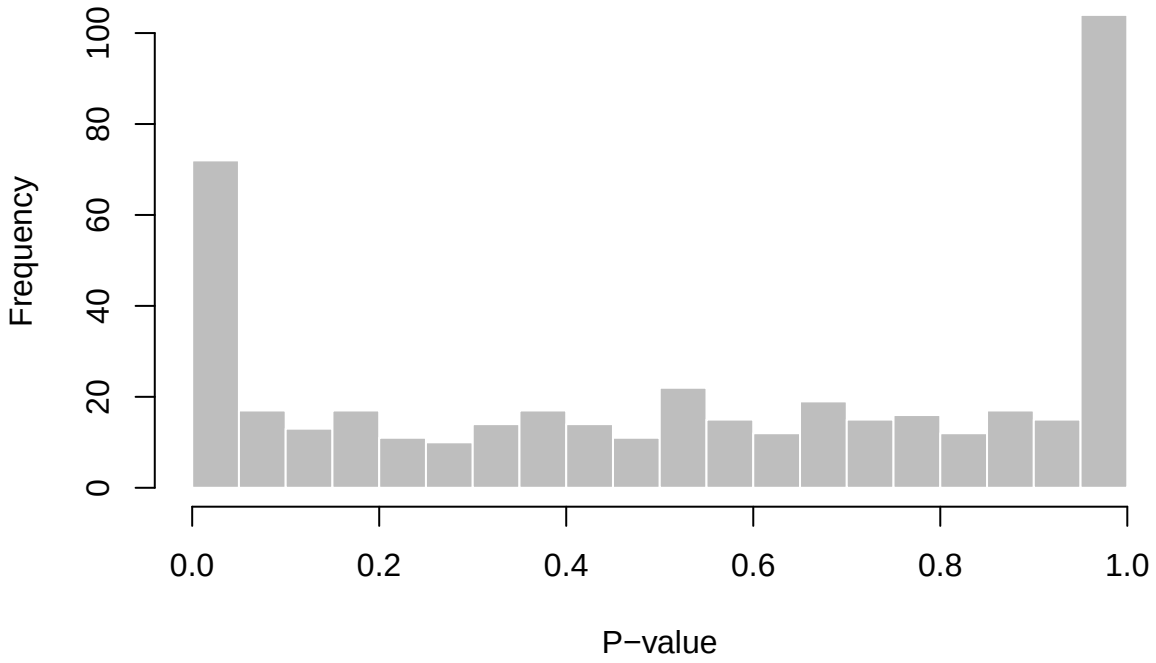
```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
suppressWarnings(
wordcloud(words=wordsDF$word,
          freq=1 - wordsDF$p,
colors = adjustcolor('darkblue', alpha.f = 0.6)))
```

Also, we can plot the distribution of p-values (expecting, of course, that this should be uniform).

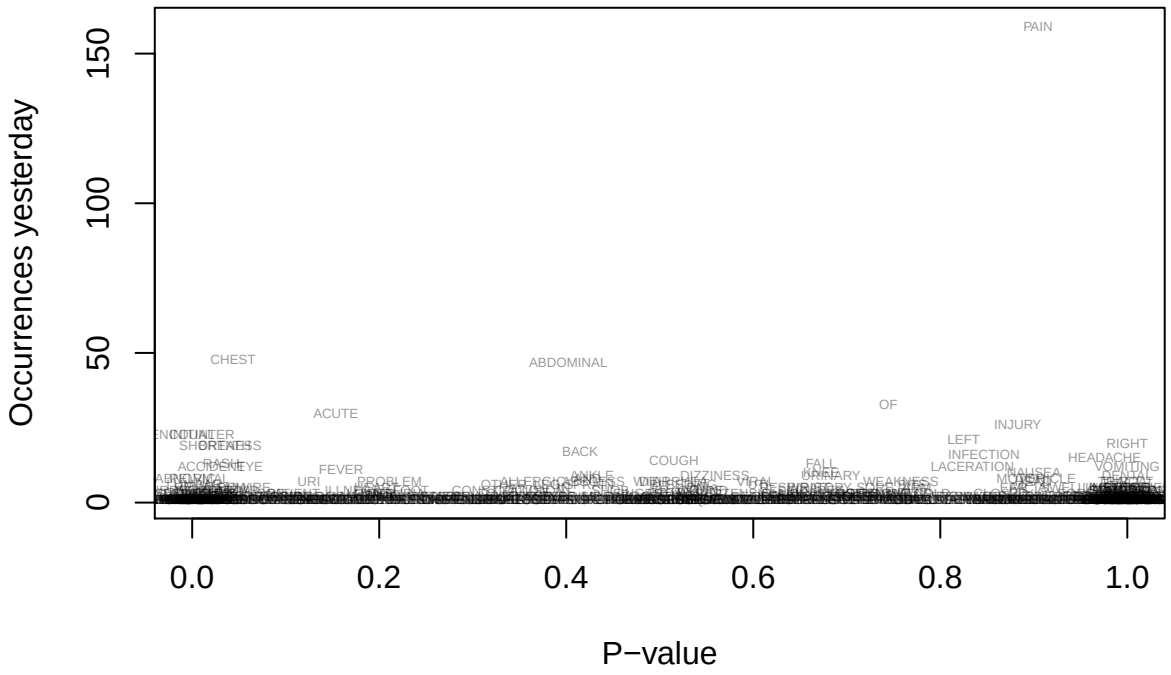
```
hist(wordsDF$p, breaks = 30,
     xlab = 'P-value', main = NA,
     border = 'white', col = 'grey')
```



The non-uniform distribution is likely due to the prepondernce of 1-occurrence words which had never previously appeared. This will be addressed in clustering (see "Next steps" section).

Finally, we can examine any potential correlation between the p-value and actual count of occurrences yesterday. We should be most concerned about words with a low p-value and high occurrence (upper left) and least concerned about words with a high p-value and low occurrence (bottom right).

```
wordsDF <- wordsDF[which(wordsDF$word != ''),]
plot(x = wordsDF$p,
     y = wordsDF$count,
     xlab = 'P-value',
     ylab = 'Occurrences yesterday',
     type = 'n')
text(x = wordsDF$p,
     y = wordsDF$count,
     labels = wordsDF$word,
     col = adjustcolor('black', alpha.f = 0.4),
     cex = 0.4)
```



Next steps

This concludes my assignment. Though outside the scope of this project, my hope is to improve this program by using "clustering" algorithms which will account for mispelling and variatoins on similar words. I've devised a weighted "fuzzy" (approximate string) matching algorithm to cluster words based on similarity. Once I've implemented it fully, I'll use random forest regression to quantify an "expected" value for any emergency department word (adjusted for seasonality, time of day and location). This will improve on the current implementation in that random forest's bootstrap aggregation features will give a better estimate of "likelihood" for low incidence words than the somewhat rudimentary framework I've laid out here.