Pain Points Analysis

Carlos Mercado

August 24, 2017

Synopsis

Given an excel file with over 2,000 comments across several categories of healthcare.gov's help pages (real users submitting advice with the "Give Feedback" widget) the goal is to synthesize the data in an understandable way and identify the pain points of healthcare.gov - whether it be User Interface or Software.

Data was provided by the Center for Medicaid and Medicare Services (CMS) from the Open Enrollment Period (9 million Americans enrolled).

The general outline for this presentation is:

- moving the data to a statistical system
- general data features
- specific Natural Language Processing
- Pain Points and notes for further analysis

The reasons I chose Natural Language Processing in R:

1. The data is small, but rich. So, doing an extensive process such as n-gram modeling is computationally possible on a single laptop.
2. The data comes from a specific population - people who *wanted to enroll* and who also *saught help* in doing so and also *decided to give feedback*.
3. This population may have unique (possibly demographic) similarities that can be identified by both the *language use* and the *most common feedback*.

Demographics and other valuable information would be helpful in the future for making actual product recommendations for the system.

Download File

```
csv.to.read <- "Business Analyst Data Analysis Presentation - Open Enrollment Help Page Comments
 - Comments.csv"
#This is my local download of the 2,500+ comments (second sheet of the excel doc)

medicare <- read.csv(csv.to.read, stringsAsFactors = FALSE)
#reads as 2,179 observations of 2 columns (URL and comment)
```

General Data inspection

```
countuniques <- function(txt,column){length(unique(txt[,column]))}

countuniques(medicare,1)
```

```
## [1] 51
```

```
countuniques(medicare,2)
```

```
## [1] 2175
```

There are 51 unique URLs for where the comments are sourced and there are 2175 unique comments (some comments may be blank or coincidentally an exact match).

I want to do some data cleaning, such as shortening the URLs and eventually removing punctuation and stopwords and other typical NLP tasks.

```
library(dplyr) #for grouping
library(tidytext) #for NLP



#for visual purposes I'll remove the healthcare.gov characters that are
#inside all of the URLs and sort them by the number in each category
medicare$URL <- gsub("https://www.healthcare.gov/","",
                     medicare$URL, fixed = TRUE)
#removing "http...gov/" from each URL

medicareGROUPED <- group_by(medicare, URL)

groupcounts <- summarise_all(medicareGROUPED,length)
#number of comments in each category

groupcounts <- groupcounts[order(groupcounts$Comment, decreasing = TRUE),]
```

The most common categories of feedback are: * Not understanding the coverage you have * Parents, Caretakers, and Relative Questions * Adding other income * logging in to your marketplace account * Deduction questions

```
groupcounts #only the top 10 are shown
```

```
## # A tibble: 51 x 2
##                                                          URL Comment
##                                                        <chr>   <int>
##  1                          help/what-health-coverage-do-i-have/     213
##  2                 help/parent-and-caretaker-relative-questions/     162
##  3                                      help/add-other-income/     128
##  4 help/i-am-having-trouble-logging-in-to-my-marketplace-account/     114
##  5                                    help/deduction-questions/     108
##  6                                   help/automatic-enrollment/     107
##  7                                   help/disability-questions/     103
##  8                         help/found-not-eligible-for-medicaid/     101
##  9                                  help/losing-health-coverage/     101
## 10                                 help/information-on-medicare/      95
## # ... with 41 more rows
```

```
## [1] "mean: 43"
```

```
## [1] "median: 20"
```

Some questions so far

For a full analysis, that eventually leads to recommendations, how far ahead are constraints known?

The average number of feedbacks per category is 43, but seeing the median is 20, it is heavily skewed toward the top 10 categories. For feasibility, it may be prudent to only seek to solve the most common pain points (for example, those with 80 or more comments).

Do clients generally know what needs to be fixed first? Or is there a hands off approach for NAVA to select the features that best serve the population?

Actual Analysis

To stay true to the goal of identifying pain points, I want to trim the data to those categories that have twice the average number of complaints. These seem to signify areas where small improvements can have a major impact (the 20/80 principle I mentioned in my Outline for Product Requirements).

```
#Top 10 Pain Points (with 1 bonus)

groupcounts[groupcounts$Comment >= 86,] #URLs with more than 85 comments
```

```
## # A tibble: 11 x 2
##                                                          URL Comment
##                                                        <chr>   <int>
##  1                           help/what-health-coverage-do-i-have/     213
##  2                  help/parent-and-caretaker-relative-questions/     162
##  3                                       help/add-other-income/     128
##  4 help/i-am-having-trouble-logging-in-to-my-marketplace-account/     114
##  5                                    help/deduction-questions/     108
##  6                                   help/automatic-enrollment/     107
##  7                                   help/disability-questions/     103
##  8                          help/found-not-eligible-for-medicaid/     101
##  9                                    help/losing-health-coverage/     101
## 10                                  help/information-on-medicare/      95
## 11                               help/reconciling-your-tax-credit/      89
```

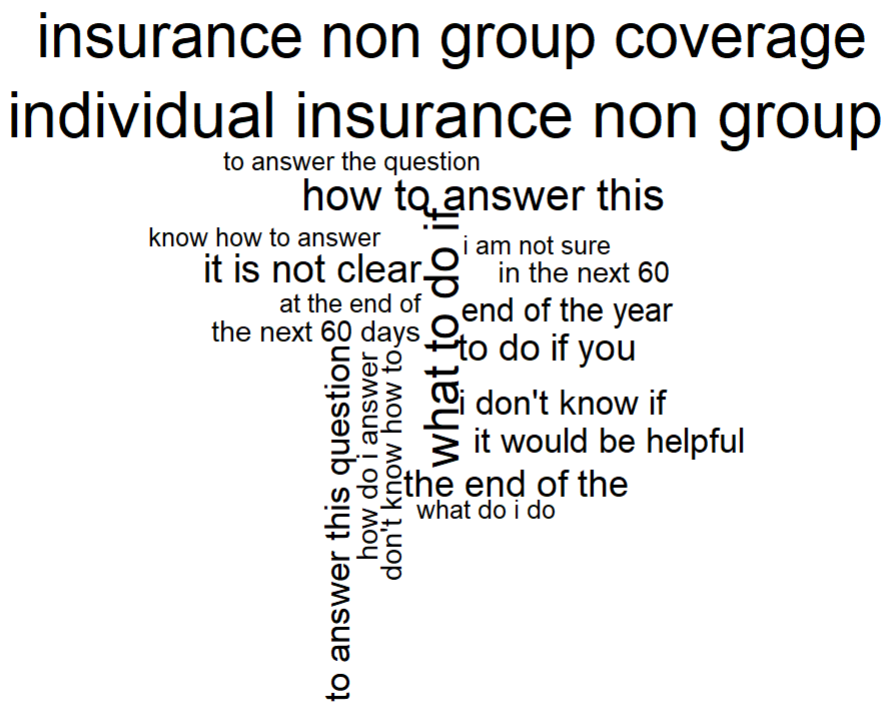
These 11 categories alone contain over 1,300 of the comments.

```
topPainURLs <- groupcounts[groupcounts$Comment >= 86,][,1]
#the top URL categories

#subset
topMedicare <- medicareGROUPED[which(medicareGROUPED$URL %in% topPainURLs$URL),]

medicare.bigrams <- unnest_tokens(topMedicare,bigram,Comment,
                                  token="ngrams",n=2)
medicare.bigrams <- group_by(medicare.bigrams,URL)
medicare.trigrams <- unnest_tokens(topMedicare,trigram,Comment,
                                   token="ngrams",n=3)
medicare.trigrams <- group_by(medicare.trigrams,URL)

medicare.quadgrams <-unnest_tokens(topMedicare,quadgram,Comment,
                                   token="ngrams",n=4)
medicare.quadgrams <- group_by(medicare.quadgrams,URL)

medicare.pentagrams <- unnest_tokens(topMedicare,pentagram, Comment, token = "ngrams", n = 5)
medicare.pentagrams <- group_by(medicare.pentagrams, URL)

count2 <- count(medicare.bigrams, bigram, sort = TRUE)
count3 <- count(medicare.trigrams, trigram, sort = TRUE)
count4 <- count(medicare.quadgrams,quadgram,sort = TRUE)
count5 <- count(medicare.pentagrams,pentagram, sort=TRUE)
```

Viewing Results and Ideas for Further Analysis

Common Word Groupings

- Most common word quadruplets among all categories:



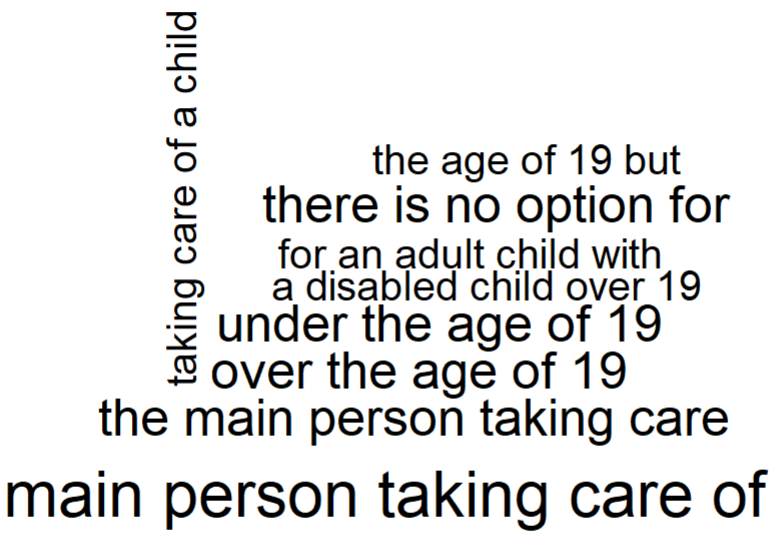
Common Word Groupings

- Most common word quadruplets among all categories:

```
## # A tibble: 35,181 x 2
##                          quadgram     n
##                             <chr> <int>
##  1 individual insurance non group    26
##  2   insurance non group coverage    25
##  3                  what to do if    19
##  4             how to answer this    17
##  5                it is not clear    15
##  6                 the end of the    14
##  7                   to do if you    14
##  8                i don't know if    13
##  9            it would be helpful    13
## 10        to answer this question    13
## # ... with 35,171 more rows
```

Specific Analysis, Within Groups

Most common 5-word groups in the parent, caretaker, relative category:



Looking at the most common problems based on counting the different word pairs (or triplets, or quadruplets) we see a few things:

1. The comments show a clear *lack of understanding*, “how to”,“if you”,“how do i”,what to do if“… People are commenting in the help section because the help mechanisms (whether they be FAQs or live chat, etc) aren’t working. Users feel that there specific situations are unique enough to warrant specific instruction -”if…"

This is different than feeling like a feature *should exist, but doesn’t* or that an interface is *too difficult to use*.

2. Within each URL category there are *fundamental* features that are not being explained. For example, in Automatic Enrollment, the most common trigram is “how to cancel”. An easily identifiable user-story that is still difficult for some users.

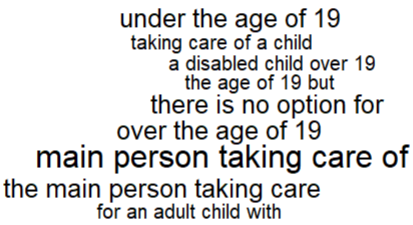
```
count3[6,]
```

```
## # A tibble: 1 x 3
## # Groups:   URL [1]
##                          URL       trigram     n
##                        <chr>         <chr> <int>
## 1 help/automatic-enrollment/ how to cancel    15
```

3. Looking at the broadest tested case- 5-grams it becomes clear that there are numeric benchmarks (19, 60) that cause people to seek advice, i.e coverage ending within 60 days or disabled children over the age of

19. For example, in the parent and caretaker questions several comments including that a feature is missing “there is no option for” or seek extra advice, “19 but…”

```
library(wordcloud)

caretaker5 <- count5[count5$URL == "help/parent-and-caretaker-relative-questions/",]
with(caretaker5, wordcloud(pentagram, n, scale = c(1,.1)))
```



To stay within the spirit of making this presentation within the allotted time, here are a few key steps to what a further analysis would look like:

- More data - All of these comments are from a specific window during the Open Enrollment Period, but healthcare management and using healthcare.gov after you’ve successfully enrolled are also common user-stories.
- More data on the commenters - are there demographics that have more issues with certain topics than others? For example - using HSA Contributions and lowering income were the most common topics in “add-other-income”, but they were rarely commented compared to the parent and caretaker category.
- More NLP - I decided against removing words or engaging in sentiment analysis, but those may also provide valuable insights - do users give more “negative” feedback in certain categories compared to others?