Creating Event Data

Zachary C. Steinert-Threlkeld

09.08.2020

SETUP

```
knitr::opts_chunk$set(echo = TRUE, eval=FALSE)
knitr::opts_knit$set(root.dir ='/Users/Zack/Documents/UCLA/Courses/EventDataFromSocia
lMedia/')
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=80),tidy=TRUE)

library(dplyr)  # for sample_n
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(rtweet)  # parsing tweets
library(jsonlite)  # parsing tweets
```

```
##
## Attaching package: 'jsonlite'
```

```
## The following object is masked from 'package:rtweet':
##
##     flatten
```

```
library(PRROC)  # for evaluating classifier fit
```

We will use functionality in `rtweet` and other packages. In addition, let us use two datasets that I have provided. One is 50,000 tweets from the United States on January 21, 2017, the first Women's March. The second is the ten minute sample of tweets using a mixture of keywords from the streaming API.

```
wm <- read.csv("Data/teaching_US_2017-01-21.csv.gz")
```

Below I have attempted to use `parse_stream` . You will get an error message because there are tweets with broken JSON formatting. This happens to be everytime I work with tweets.

```
fails <- parse_stream("Data/teaching_tweets_mixture.json")
```

```
## Error: parse error: unallowed token at this point in JSON text
##           mUpqFiqezqXLS35.mp4?tag=10"},,{"content_type":"application\/
##                       (right here) ------^
```

The solution is to read the file line by line and skip bad tweets. Frankly, I *strongly prefer* Python for my data processing because it is very easy to read line by line and skip bad lines. In R, as you will see shortly, this process is more cumbersome and `tryCatch` is horrible. Nonetheless, I wanted to keep everything in R for this class, so I tortured myself for 90-120 minutes figuring things out.

The below code shows my hacked solution. It: 1. Uses `readLines` to create a character vector that is a giant string for each line of the file it reads. 2. The end goal is to have a data frame. However, tweets will have different fields depending on what they contain, so a data frame would have different columns. R does not like this, and I could not find an obvious equivalent of `pandas.io.json.json_normalize` , the Python function that coerces lists of tweets with different fields into one data frame. I there made a function to keep the columns of interest. - *Modify df in this function for the fields that you want.* This step will require a lot of working with tweets, adding fields as you see new ones you want. - *The if statement is for tweets without a field wanted.* See my comments in the function for ideas of ways to improve this process. 3. The loop reads each tweet, handles any errors, and builds a data frame. This data frame is what we will analyze. - It would be nice to make this loop its own function.

Note that this code is slow. It takes more time to convert the tweets to a data frame than it did to download the tweets, much slower than my Python equivalent. Hopefully it is my fault and not R's. `readJSON` is notoriously slow, but `stream_in` from `jsonlite` appears to only work on a file, not a loaded string.

```
thetweets <- readLines("Data/teaching_tweets_mixture.json")
orig_n <- length(thetweets)  # To know later how many tweets I lose

# tweet is JSON formatted string
parseTweet <- function(tweet) {
    temp <- fromJSON(tweet)

    # There must be a better way of doing this.
    if (is.null(temp$user$location)) {
        temp$user$location <- NA
    }

    # How to have data.frame handle fields with NULL values?  The loop above is my
    # answer, but I am sure there is a more elegant solution.
    df <- data.frame(lang = temp$lang, text = temp$text, created_at = temp$created_at
,
        id = temp$id_str, source = temp$source, user.id = temp$user$id_str, user.sn =
temp$user$screen_name,
        user.location = temp$user$location, user.created_at = temp$user$created_at)
    return(df)
}
```

Below implements the above function. I have run it before compiling this document because the code runs very slowly.

```
tweet_mixture <- NA
i <- 0
for (line in 1:length(thetweets)) {
    i <- i + 1
    # This tryCatch structure is here:
    # https://stackoverflow.com/questions/8093914/use-trycatch-skip-to-next-value-of-loop-upon-error
    skip_to_next <- FALSE
    tryCatch({
        tweet_df <- parseTweet(tweet = thetweets[line])
    }, error = function(e) {
        # message(sprintf('Error: %s', e)) # No need to show the errors when compling,
        # though they are useful for you to see when you are working on your own.
        skip_to_next <<- TRUE
    })
    if (skip_to_next) {
        # Go to next line if there was an error
        next
    }
    if (!skip_to_next) {
        tweet_mixture <- rbind(tweet_mixture, tweet_df)
    }
    if (i%%1000 == 0) {
        # I like to know how far along I am.  It calms my worry.
        print(i)
    }
}

print(paste0(round(nrow(tweet_mixture)/orig_n * 100, 2), "% of tweets are kept."))

write.csv(tweet_mixture, "../Data/teaching_tweets_mixture.csv")
```

TYPES OF EVENTS

In the first class, I alluded to a coding scheme called CAMEO, which stands for Conflict and Mediation Event Observations. Created by Philip Schrodt and Deborah Garner in the early 2000s, wtih support from the National Science Foundation, CAMEO includes 20 parent event types arranged in appproximate order of increasing intensity. The following screenshot shows the events that CAMEO records. Note that each has several child and grandchild events.

These categories in turn are derived from extensive dictionaries.

of these, especially `MAKE PUBLIC STATEMENT`, `APPEAL`, and `EXPRESS INTENT TO COOPERATE` are verbal in nature, they can be coded directly from tweets. Others, such as `PROTEST`, `ASSAULT`, or `FIGHT` are physical, so tweets themselves do not count as events. They may be evidence of events elsewhere, but knowing that requires more work than for the verbal events. As we discuss different methods for coding event data, it is important to keep this distinction in mind.

IDENTIFY EVENTS

This section now presents, at a very high level, versions of the three approaches to creating event data that I discussed in the first class. The manual approach entails reading all documents (tweets). The automatic approach involves rules, often derived from classification algorithms of varying copmlexity. The hybrid approach uses rules to filter raw data for analysis by a team of humans.

Manual

In many ways, this approach is the simplest. Load data and read it. If you read tweets that are the result of parameters passed to the Search or Streaming API, that still counts as a manual approach.

This approach requires the least amount of coding. Its main purpose is to add metadata that may help coders or later aggregation. It also can be used to do light filtering, such as by keeping only certain sources,

```
orig <- data.frame(parse_stream("Data/random_60s.json"))
```

Below are some ways I like to inspect the data.

```
dim(orig)
```

```
## [1] 3053   90
```

```
head(orig)
```

```
##               user_id           status_id          created_at
## 1          2345075802 1302690550269632517 2020-09-06 19:30:04
## 2           159917373 1302690550278098945 2020-09-06 19:30:04
## 3 1237330744504020997 1302690550278045696 2020-09-06 19:30:04
## 4 1297649834795634690 1302690550265520130 2020-09-06 19:30:04
## 5           411131798 1302690550265425920 2020-09-06 19:30:04
## 6 1264835279443111937 1302690550257250304 2020-09-06 19:30:04
##       screen_name
## 1   firebrand_bot
## 2  Hrozvitnir_bot
## 3 nation1st_INDIA
## 4           qq664
## 5      wonder8428
## 6        MammukaL
##
text
## 1
これを見ろ
## 2
出させたね、これを……
## 3 जय जौहड़ी, जय बागपत, जय यूपी, जय हरियाणा-दिल्ली-मुंबई-चेन्नई, जय अंडमान इत्यादि इत्यादि । जय भारत \U0
001f1ee\U0001f1f3
## 4                                                    يـاحل\nلـعلّك بـخير مـنين مـارحت و أحسن حـال
مٍ مـايـشبـه قـيـمـتـه .. غـيـر تـحـقـيـقـه
## 5                          เธอเห็นสุพัดไหม??\nกิฟ:ไม่นะ\nถ้าเจอแล้วตะโกนพร้อมกันนะ "คิดถึง"\nกิฟ:
หลวงพี่!!!!!!! \n\n#สุพัดอยู่ไหน
## 6
#HappyBirthdayMammukka ❣️
##                source display_text_width  reply_to_status_id
## 1  IRC:AMAKUDARI_AXIS                  NA                <NA>
## 2    グラズヘイムの中                 NA                <NA>
## 3 Twitter for Android                  NA                <NA>
## 4  Twitter for iPhone                  NA                <NA>
## 5 Twitter for Android                  NA                <NA>
```

```
## 6 Twitter for Android                  NA 1302690546172002305
##      reply_to_user_id reply_to_screen_name is_quote is_retweet
## 1                <NA>                 <NA>    FALSE      FALSE
## 2                <NA>                 <NA>    FALSE      FALSE
## 3                <NA>                 <NA>    FALSE       TRUE
## 4                <NA>                 <NA>    FALSE       TRUE
## 5                <NA>                 <NA>    FALSE       TRUE
## 6 1264835279443111937              MammukaL    FALSE      FALSE
##   favorite_count retweet_count quote_count reply_count
## 1              0             0           0           0
## 2              0             0           0           0
## 3              0             0           0           0
## 4              0             0           0           0
## 5              0             0           0           0
## 6              0             0           0           0
##                hashtags symbols urls_url urls_t.co urls_expanded_url
## 1                    NA      NA       NA        NA                NA
## 2                    NA      NA       NA        NA                NA
## 3                    NA      NA       NA        NA                NA
## 4                    NA      NA       NA        NA                NA
## 5              สุพัดอยู่ไหน      NA       NA        NA                NA
## 6 HappyBirthdayMammukka      NA       NA        NA                NA
##   media_url media_t.co media_expanded_url media_type ext_media_url
## 1        NA         NA                NA         NA            NA
## 2        NA         NA                NA         NA            NA
## 3        NA         NA                NA         NA            NA
## 4        NA         NA                NA         NA            NA
## 5        NA         NA                NA         NA            NA
## 6        NA         NA                NA         NA            NA
##   ext_media_t.co ext_media_expanded_url ext_media_type    mentions_user_id
## 1             NA                    NA           <NA>                  NA
## 2             NA                    NA           <NA>                  NA
## 3             NA                    NA           <NA>  715401445785210880
## 4             NA                    NA           <NA>  924560983678488576
## 5             NA                    NA           <NA> 1180007593458012161
## 6             NA                    NA           <NA>                  NA
##   mentions_screen_name lang quoted_status_id quoted_text quoted_created_at
## 1                   NA   ja             <NA>        <NA>              <NA>
## 2                   NA   ja             <NA>        <NA>              <NA>
## 3      realshooterdadi   hi             <NA>        <NA>              <NA>
## 4                F30ii   ar             <NA>        <NA>              <NA>
## 5            jalanwaris   th             <NA>        <NA>              <NA>
## 6                   NA  und             <NA>        <NA>              <NA>
##   quoted_source quoted_favorite_count quoted_retweet_count quoted_user_id
## 1          <NA>                   NA                  NA           <NA>
## 2          <NA>                   NA                  NA           <NA>
## 3          <NA>                   NA                  NA           <NA>
```

```
## 4          <NA>                    NA                    NA           <NA>
## 5          <NA>                    NA                    NA           <NA>
## 6          <NA>                    NA                    NA           <NA>
##   quoted_screen_name quoted_name quoted_followers_count
## 1               <NA>        <NA>                     NA
## 2               <NA>        <NA>                     NA
## 3               <NA>        <NA>                     NA
## 4               <NA>        <NA>                     NA
## 5               <NA>        <NA>                     NA
## 6               <NA>        <NA>                     NA
##   quoted_friends_count quoted_statuses_count quoted_location
## 1                   NA                    NA            <NA>
## 2                   NA                    NA            <NA>
## 3                   NA                    NA            <NA>
## 4                   NA                    NA            <NA>
## 5                   NA                    NA            <NA>
## 6                   NA                    NA            <NA>
##   quoted_description quoted_verified   retweet_status_id
## 1               <NA>              NA                <NA>
## 2               <NA>              NA                <NA>
## 3               <NA>              NA 1302603424312164352
## 4               <NA>              NA 1297280600446046210
## 5               <NA>              NA 1302650460973670400
## 6               <NA>              NA                <NA>
##
retweet_text
## 1
<NA>
## 2
<NA>
## 3 जय जौहड़ी, जय बागपत, जय यूपी, जय हरियाणा-दिल्ली-मुंबई-चेन्नई, जय अंडमान इत्यादि इत्यादि । जय भारत \U0
001f1ee\U0001f1f3
## 4                                                                 یاحلnلعلّك بخیر منین مارحت وأحسن حال
مٍ مایشبه قیمته .. غیر تحقیقه
## 5                          เธอเห็นสุพัดไหม??\nกิฟ:ไม่นะ\nถ้าเจอแล้วตะโกนพร้อมกันนะ "คิดถึง"\nกิฟ:
หลวงพี่!!!!!!! \n\n#สุพัดอยู่ไหน
## 6
<NA>
##    retweet_created_at      retweet_source retweet_favorite_count
## 1                <NA>                <NA>                     NA
## 2                <NA>                <NA>                     NA
## 3 2020-09-06 13:43:52  Twitter for iPhone                   4811
## 4 2020-08-22 21:12:52  Twitter for iPhone                     13
## 5 2020-09-06 16:50:46 Twitter for Android                     15
## 6                <NA>                <NA>                     NA
##   retweet_retweet_count     retweet_user_id retweet_screen_name
## 1                    NA                <NA>                <NA>
```

```
## 2                    NA                <NA>                <NA>
## 3                   368  715401445785210880     realshooterdadi
## 4                    94  924560983678488576               F30ii
## 5                    27 1180007593458012161          jalanwaris
## 6                    NA                <NA>                <NA>
##          retweet_name retweet_followers_count retweet_friends_count
## 1                <NA>                     NA                    NA
## 2                <NA>                     NA                    NA
## 3  Dadi Chandro Tomar                 180676                   242
## 4                41                    366                    شعاف
## 5 Jalan™.สอบไม่ตอบเดม.                  427                   260
## 6                <NA>                     NA                    NA
##   retweet_statuses_count retweet_location
## 1                     NA             <NA>
## 2                     NA             <NA>
## 3                   6610    जौहड़ी, बागपत
## 4                  12082             <NA>
## 5                  21737       \U0001f49b
## 6                     NA             <NA>
##
retweet_description
## 1
<NA>
## 2
<NA>
## 3 Veteran pistol shooter, coach, sports enthusiast, farmer \U0001f449 contact : ch
andro.tomar@gmail.com
## 4                                                                    منتظر من بيض ا
لايام مايرضي الشفوف .
## 5                                                  friend\U0001f49b\U0001f469
❤️\U0001f469 @Ismepatx
## 6
<NA>
##   retweet_verified place_url place_name place_full_name place_type country
## 1               NA      <NA>       <NA>            <NA>       <NA>    <NA>
## 2               NA      <NA>       <NA>            <NA>       <NA>    <NA>
## 3             TRUE      <NA>       <NA>            <NA>       <NA>    <NA>
## 4            FALSE      <NA>       <NA>            <NA>       <NA>    <NA>
## 5            FALSE      <NA>       <NA>            <NA>       <NA>    <NA>
## 6               NA      <NA>       <NA>            <NA>       <NA>    <NA>
##   country_code geo_coords coords_coords                    bbox_coords
## 1         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
## 2         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
## 3         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
## 4         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
## 5         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
## 6         <NA>     NA, NA        NA, NA NA, NA, NA, NA, NA, NA, NA, NA
```

```
##                                                    status_url
## 1   https://twitter.com/firebrand_bot/status/1302690550269632517
## 2  https://twitter.com/Hrozvitnir_bot/status/1302690550278098945
## 3 https://twitter.com/nation1st_INDIA/status/1302690550278045696
## 4           https://twitter.com/qq664/status/1302690550265520130
## 5      https://twitter.com/wonder8428/status/1302690550265425920
## 6        https://twitter.com/MammukaL/status/1302690550257250304
##               name     location
## 1 ファイアブランド ネオサイタマ
## 2  シュライバーbot グラズヘイム
## 3       ☠ VibhuT ☠          <NA>
## 4       \u1dc2عونك          <NA>
## 5        justlove.          <NA>
## 6    Mammuka Lover          <NA>
##
description
## 1 【非公式自動bot】ニンジャスレイヤー@NJSLYRに登場するアマクダリニンジャ・ファイアブランドっぽい自動botです。定期的に原作の台詞をツイートします。都道府県名リプライで天気予報・フォローしてでフォロー・スシを頼むと投げてよこします。スシ配り兄ちゃん重点。他反応語句等はURLご確認ください。
## 2
ゲーム「Dies irae」より、ウォルフガング・シュライバーの非公式bot。詳細はURLの説明書をご参照ください
ご用がございましたらDMで
## 3
Nation first, \n\n#isupportCAA #isupportNPR #isupportNRC\n#makeinindia #vocalforlocal
#boycottchinese \n  \U0001f1ee\U0001f1f3
## 4
              شبه متفاعل ..
## 5
Love is in the air.
## 6
<NA>
##                            url protected followers_count friends_count
## 1 http://twpf.jp/firebrand_bot     FALSE              24            21
## 2    http://tokiya312.mond.jp/     FALSE             560           572
## 3           http://www.bjp.org     FALSE            1149          2742
## 4     http://tellonym.me/qo_i6     FALSE              99            22
## 5                        <NA>     FALSE               7            58
## 6                        <NA>     FALSE             367           571
##   listed_count statuses_count favourites_count  account_created_at
## 1            2          48972                0 2014-02-15 12:46:37
## 2           37         467976                0 2010-06-26 17:27:26
## 3            0          26633            74705 2020-03-10 10:53:43
## 4            0           7796               53 2020-08-23 21:40:14
## 5            0          12343            18828 2011-11-13 01:24:29
## 6            0           4169              600 2020-05-25 08:26:59
##   verified profile_url profile_expanded_url account_lang
## 1    FALSE        <NA>                 <NA>           NA
```

```
## 2    FALSE        <NA>                 <NA>           NA
## 3    FALSE        <NA>                 <NA>           NA
## 4    FALSE        <NA>                 <NA>           NA
## 5    FALSE        <NA>                 <NA>           NA
## 6    FALSE        <NA>                 <NA>           NA
##                                                 profile_banner_url
## 1                                                             <NA>
## 2                                                             <NA>
## 3 https://pbs.twimg.com/profile_banners/1237330744504020997/1595792179
## 4 https://pbs.twimg.com/profile_banners/1297649834795634690/1599100045
## 5           https://pbs.twimg.com/profile_banners/411131798/1597421615
## 6                                                             <NA>
##                             profile_background_url
## 1 http://abs.twimg.com/images/themes/theme14/bg.gif
## 2 http://abs.twimg.com/images/themes/theme15/bg.png
## 3
## 4
## 5  http://abs.twimg.com/images/themes/theme1/bg.png
## 6
##                                                          profile_image_url
## 1  http://pbs.twimg.com/profile_images/437941903900684288/EwX1-7nQ_normal.png
## 2  http://pbs.twimg.com/profile_images/945313447356416001/66g18ky-_normal.jpg
## 3 http://pbs.twimg.com/profile_images/1301948802236968960/QkvaFsS7_normal.jpg
## 4 http://pbs.twimg.com/profile_images/1301846111007760384/HV9XRCOb_normal.jpg
## 5 http://pbs.twimg.com/profile_images/1264219187993436166/03lG6srk_normal.jpg
## 6 http://pbs.twimg.com/profile_images/1300796599975604229/jn1yJV2g_normal.jpg
```

```
sort(names(orig))
```

```
##  [1] "account_created_at"      "account_lang"
##  [3] "bbox_coords"             "coords_coords"
##  [5] "country"                 "country_code"
##  [7] "created_at"              "description"
##  [9] "display_text_width"      "ext_media_expanded_url"
## [11] "ext_media_t.co"          "ext_media_type"
## [13] "ext_media_url"           "favorite_count"
## [15] "favourites_count"        "followers_count"
## [17] "friends_count"           "geo_coords"
## [19] "hashtags"                "is_quote"
## [21] "is_retweet"              "lang"
## [23] "listed_count"            "location"
## [25] "media_expanded_url"      "media_t.co"
## [27] "media_type"              "media_url"
## [29] "mentions_screen_name"    "mentions_user_id"
## [31] "name"                    "place_full_name"
## [33] "place_name"              "place_type"
## [35] "place_url"               "profile_background_url"
## [37] "profile_banner_url"      "profile_expanded_url"
## [39] "profile_image_url"       "profile_url"
## [41] "protected"               "quote_count"
## [43] "quoted_created_at"       "quoted_description"
## [45] "quoted_favorite_count"   "quoted_followers_count"
## [47] "quoted_friends_count"    "quoted_location"
## [49] "quoted_name"             "quoted_retweet_count"
## [51] "quoted_screen_name"      "quoted_source"
## [53] "quoted_status_id"        "quoted_statuses_count"
## [55] "quoted_text"             "quoted_user_id"
## [57] "quoted_verified"         "reply_count"
## [59] "reply_to_screen_name"    "reply_to_status_id"
## [61] "reply_to_user_id"        "retweet_count"
## [63] "retweet_created_at"      "retweet_description"
## [65] "retweet_favorite_count"  "retweet_followers_count"
## [67] "retweet_friends_count"   "retweet_location"
## [69] "retweet_name"            "retweet_retweet_count"
## [71] "retweet_screen_name"     "retweet_source"
## [73] "retweet_status_id"       "retweet_statuses_count"
## [75] "retweet_text"            "retweet_user_id"
## [77] "retweet_verified"        "screen_name"
## [79] "source"                  "status_id"
## [81] "status_url"              "statuses_count"
## [83] "symbols"                 "text"
## [85] "url"                     "urls_expanded_url"
## [87] "urls_t.co"               "urls_url"
## [89] "user_id"                 "verified"
```

Let's add some metadata to make some later work easier.

```
orig$created_at_date <- substr(orig$created_at, 1, 10)
orig$created_at_hour <- substr(orig$created_at, 12, 13)

orig$account_created_at_date <- substr(orig$account_created_at, 1, 10)
```

Let's look at some columns that seem promising.

```
table(orig$place_type)
```

```
##
##   admin    city country
##       2      26       1
```

```
table(orig$country_code)
```

```
##
## AZ BR CA CO DE ES GB JP LU MX NG NL PL SE TR US UY VE ZA
##  1  4  1  1  1  1  4  2  1  1  2  1  1  1  1  2  1  2  1
```

```
table(orig$place_full_name)
```

```
##
##       Araçatuba, Brasil              Azerbaijan       Barcelona, España
##                       1                       1                       1
##         Bello, Colombia          Enugu, Nigeria   Florianópolis, Brasil
##                       1                       1                       1
##       Glasgow, Scotland       Grudziądz, Polska      Haarlem, Nederland
##                       2                       1                       1
##          Hatay, Türkiye          Hausa, Nigeria           Kenfig, Wales
##                       1                       1                       1
##        Lincoln, Ontario     Manchester, England    Maracaibo, Venezuela
##                       1                       1                       1
##      Maturín, Venezuela Milnerton, South Africa     Montevideo, Uruguay
##                       1                       1                       1
## Niederanven, Luxembourg              Ohio, USA       São Paulo, Brasil
##                       1                       1                       1
##  Sonneberg, Deutschland       Stockholm, Sweden          Viamão, Brasil
##                       1                       1                       1
##             Visalia, CA        Zapopan, Jalisco            北海道 森町
##                       1                       1                       1
##             滋賀 米原市
##                       1
```

```
sources <- data.frame(table(orig$source))
sources <- sources[order(sources$Freq, decreasing = TRUE), ]

head(orig$hashtags)
```

```
## [[1]]
## [1] NA
##
## [[2]]
## [1] NA
##
## [[3]]
## [1] NA
##
## [[4]]
## [1] NA
##
## [[5]]
## [1] "สุพัดอยู่ไหน"
##
## [[6]]
## [1] "HappyBirthdayMammukka"
```

```
htags <- data.frame(table(unlist(orig$hashtags)))
htags <- htags[order(htags$Freq, decreasing = TRUE), ]
```

Add place names

In this class, I am not going to add places to tweets if they do not already have it. It is common to use the `location` field (the user self-reported location), a place name mentioned in a tweet, or both to assign location to a tweet. You could also assign it when you are coding tweets and see a place name.

Automatic

The automatic approach requires having a rule or series of rules that will identify events. I will load a third dataset to make the coding a little more productive.

I am going to use the processed .csv from the last class. I want to do this to show you how different parts of a data processing pipeline fit together and why it is important to think carefuly about the data to keep.

```
mixture <- read.csv("Data/teaching_tweets_mixture.csv")

dim(mixture)
```

```
## [1] 25917    10
```

```
head(mixture)
```

```
##   X lang
## 1 1 <NA>
## 2 2   en
## 3 3   en
## 4 4   en
## 5 5   en
## 6 6   en
##
text
## 1
<NA>
## 2                                                                    Retweets a
re highly appreciated, people need to see the amazing stuff this man makes &lt;3
## 3                                                                 Dropping onl
y one F-bomb while staying with the parents is a cause for celebration, I think.
## 4 RT @decentralgames: \U0001f525 Blackjack is now live in @decentraland! \U0001f52
5\n\nThe first weekly Decentral Games Casino Night is Friday, September 25th at…
## 5                   RT @GoodShepherd316: OK let’s relax a little...\nThe Lion fro
```

```
m the Tribe of Judah looking for that last saved Gentile at the moment of the R…
## 6                    RT @SulloJewaliya: Let us come together on 5 September on be
half of teachers day de to gift our respected teachers who came forward in this…
##                       created_at          id
## 1                            <NA>          NA
## 2 Thu Sep 03 17:57:31 +0000 2020 1.30158e+18
## 3 Thu Sep 03 17:57:31 +0000 2020 1.30158e+18
## 4 Thu Sep 03 17:57:31 +0000 2020 1.30158e+18
## 5 Thu Sep 03 17:57:31 +0000 2020 1.30158e+18
## 6 Thu Sep 03 17:57:31 +0000 2020 1.30158e+18
##                                                                                so
urce
## 1
<NA>
## 2             <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App
</a>
## 3   <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone
</a>
## 4             <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App
</a>
## 5             <a href="https://mobile.twitter.com" rel="nofollow">Twitter Web App
</a>
## 6 <a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android
</a>
##        user.id        user.sn                        user.location
## 1           NA           <NA>                                 <NA>
## 2 1.116815e+18       ShiroAF2              Your local sewer drain
## 3 1.166954e+18 zachary_allard                          Los Angeles
## 4 4.062820e+08       sugumatu オーストラリア、シンガポール、カナダ
## 5 3.013312e+09      fernics17                                 <NA>
## 6 1.300488e+18        jinirox                    New Delhi, India
##                  user.created_at
## 1                           <NA>
## 2 Fri Apr 12 21:27:30 +0000 2019
## 3 Thu Aug 29 06:02:04 +0000 2019
## 4 Sun Nov 06 14:04:56 +0000 2011
## 5 Sun Feb 08 11:42:24 +0000 2015
## 6 Mon Aug 31 17:39:17 +0000 2020
```

Question: what are some fields that perhaps we would like to use?

To recap, the three datasets are now `mixture`, `wm`, `orig`.

There are two broad automatic approaches: using a dictionary or dictionaries and building a classifier to identify events.

Dictionary

We will now use a **noun dictionary** to identify events.

```
wm_protest <- c("protest", "march", "whywemarch", "womensmarch", "whyimarch")

wm$protest <- grepl(paste(wm_protest, collapse = "|"), wm$text, ignore.case = TRUE)
```

```
protest <- c("protest", "demonstration", "gather")

mixture$protest <- grepl(paste(protest, collapse = "|"), mixture$text, ignore.case =
TRUE)
orig$protest <- grepl(paste(protest, collapse = "|"), orig$text, ignore.case = TRUE)
```

We will now use a **verb dictionary** for a similar task. Let us see if we identify actions based on words.

```
attack <- c("attack", "assault", "run at", "charge", "fight")

wm$attack <- grepl(paste(attack, collapse = "|"), wm$text, ignore.case = TRUE)
orig$attack <- grepl(paste(attack, collapse = "|"), orig$text, ignore.case = TRUE)
mixture$attack <- grepl(paste(attack, collapse = "|"), mixture$text, ignore.case = TR
UE)
```

Classifier

I will demonstrate a very simplified, high-level event classifier. Note that it does not actually identify protest because I label the training data randomly. I have tried to explain the purpose of each step and indicate any steps I skip, like not having a testing set of data. Please ask questions.

Event data require supervised learning. You could use unsupervised learning to discover themes in text, but I would not create event data without a model trained on labeled data.

Building an event classifier is the same as building any other classifier. You need labeled data and a held out test set to evaluate the trained model, then you apply that model to the full dataset.

Because it is too much work to generate labels for a class, I am going to randomly create them. The below code creates a training dataset and uses `rbinom` to assign labels.

```
desiredSize <- 1000

train_test <- sample_n(wm, desiredSize)
# train_test <- wm %>% sample_frac(round(desiredSize/nrow(wm), 2)) # Get 1000
# rows.  Could also use sample_frac()

train <- sample_frac(train_test, 0.8)
test <- anti_join(train_test, train, by = "id")
```

Did I split the train and test correctly?

```
ifelse(sum(test$id %in% train$id) == 0, "Yes!", "Sad :(")
```

```
## [1] "Yes!"
```

Now, use labels in the training data to make a model. Remember we have them from earlier.

```
train$protest_DV <- ifelse(train$protest == TRUE, 1, 0)

model <- glm(protest_DV ~ user.statuses_count + user.followers_count + user.friends_c
ount,
    data = train, family = "binomial")
```

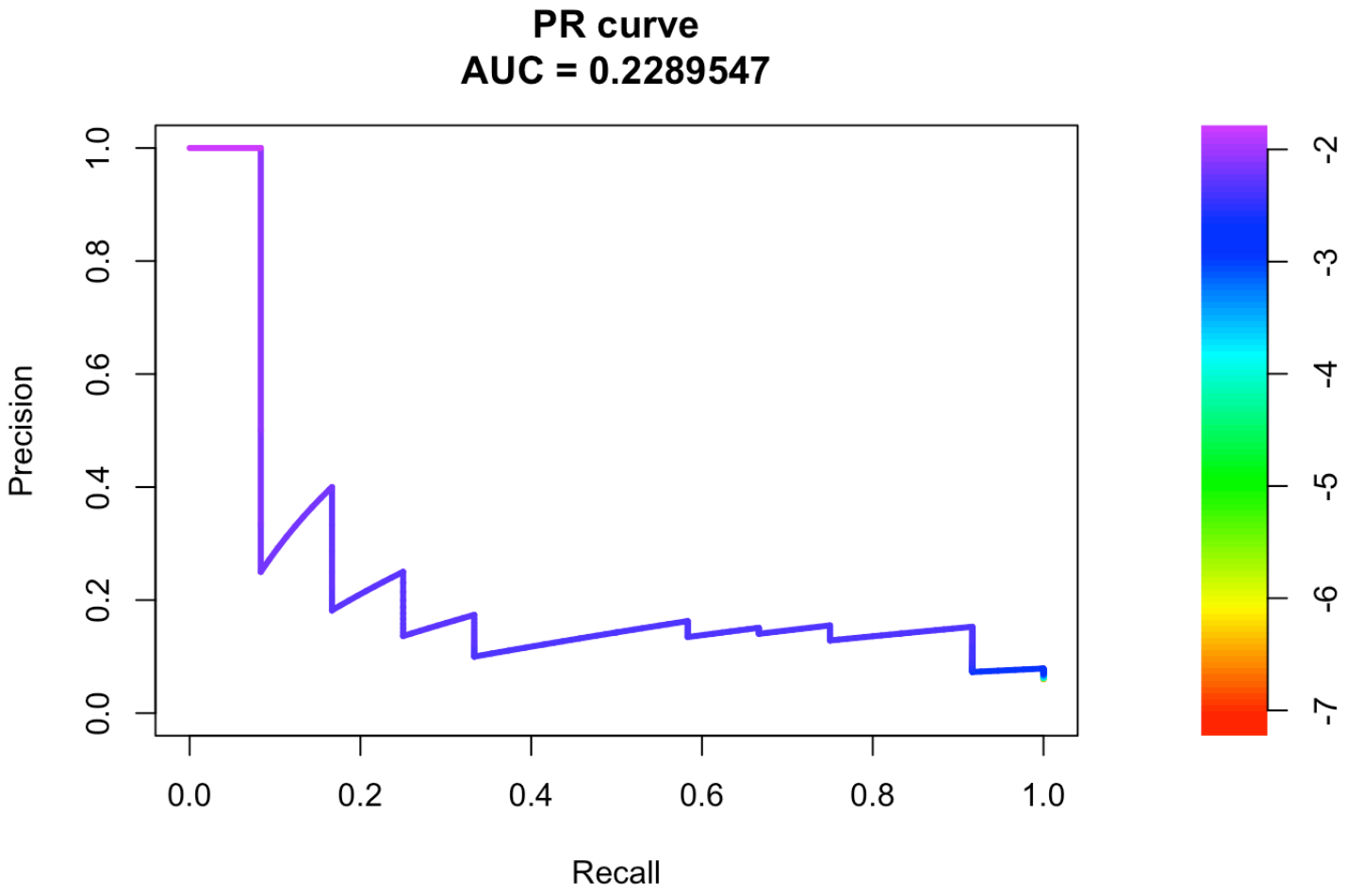
Use this model to guess on the test data.

```
test$protest_predicted <- predict.glm(object = model, newdata = test)
```

Evaluate the model using a PR or AUC curve.

```
pr <- pr.curve(scores.class0 = test$protest_predicted, weights.class0 = test$protest,
    curve = TRUE)

plot(pr)
```



Let's say we like this model. Now, apply it to the full dataset.

```
wm$protest_predicted <- predict.glm(object = model, newdata = wm)
```

Note that I have skipped several steps. In no particular order, building the classifier should: 1. Iterate over `train_test` several times to generate several train and test sets. 2. Use tweet text that has been cleaned (remove stopwords). 3. Be more complicated than a logistic regression? This point is phrased as a question because tweets are short text in often very colloquial styles. A logistic regression is probably too simple, but in my expeirence it certainly is good enough at the start of a project because of the nature of tweets. My knowledge of NLP stopped just before deep learning text models became widely used.

Aggregation

Newspaper articles are convenient because they already aggregate events by day. It is highly likely you will get multiple tweets per day about an event, in which case you need to aggregate.

The current best resolution in event data is city day, though the Crowd Counting Consortium and ACLED, which are hybrid approaches, can record intracity variation. We will therefore need to determine how finely to aggregate the tweets. This task is saved for the coding section.

Hybrid

A hybrid approach applies filters to a dataset to generate a smaller dataset of content to code. This smaller dataset will consist of items very likely to be a protest.

Here is a filter based on a dictionary.

```
wm_hybrid <- subset(wm, protest == TRUE)

nrow(wm_hybrid) - nrow(wm)
```

```
## [1] -46405
```

Maybe you only want tweets from old accounts, thinking they are less likely to be bots.

```
orig_hybrid <- subset(orig, as.Date(account_created_at_date) <= as.Date("2016-01-1"))

nrow(orig_hybrid) - nrow(orig)
```

```
## [1] -1970
```

A better approach, especially as you get very large datasets, is to filter based on the output of the classifier (or classifiers) you generate for an automatic pipeline. That is, instead of taking the “automatic” part as the ending step for identifying events, treat it like a middle step.

```
# What classifier output value to use to filter?  I like to use the following two
# pieces of output to decide.
summary(wm$protest_predicted)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## -75.644  -2.728  -2.437  -2.746  -2.344  90.359       9
```

```
quantile(wm$protest_predicted, probs = seq(0, 1, by = 0.01), na.rm = TRUE)
```

```
##         0%         1%         2%         3%         4%         5%
## -75.643607  -6.891433  -5.392941  -4.714185  -4.322468  -4.049877
##         6%         7%         8%         9%        10%        11%
##  -3.830192  -3.681580  -3.563024  -3.463885  -3.362399  -3.279166
##        12%        13%        14%        15%        16%        17%
##  -3.200145  -3.136942  -3.081640  -3.031842  -2.984958  -2.948644
##        18%        19%        20%        21%        22%        23%
##  -2.911362  -2.878987  -2.847992  -2.820326  -2.795733  -2.771258
##        24%        25%        26%        27%        28%        29%
##  -2.748054  -2.727943  -2.710134  -2.691997  -2.673663  -2.656140
##        30%        31%        32%        33%        34%        35%
##  -2.639556  -2.624043  -2.610069  -2.595660  -2.582346  -2.570952
##        36%        37%        38%        39%        40%        41%
##  -2.559098  -2.548201  -2.537764  -2.527081  -2.516769  -2.507128
##        42%        43%        44%        45%        46%        47%
##  -2.498213  -2.489813  -2.480848  -2.473367  -2.465467  -2.457816
##        48%        49%        50%        51%        52%        53%
##  -2.450720  -2.443781  -2.437348  -2.430915  -2.425216  -2.419512
##        54%        55%        56%        57%        58%        59%
##  -2.413895  -2.408601  -2.403471  -2.399168  -2.394605  -2.390517
##        60%        61%        62%        63%        64%        65%
##  -2.386237  -2.382303  -2.378910  -2.375497  -2.372455  -2.369124
##        66%        67%        68%        69%        70%        71%
##  -2.365905  -2.363020  -2.360191  -2.357433  -2.354651  -2.352105
##        72%        73%        74%        75%        76%        77%
##  -2.349887  -2.347647  -2.345803  -2.344245  -2.342872  -2.341461
##        78%        79%        80%        81%        82%        83%
##  -2.340113  -2.338693  -2.337175  -2.335375  -2.333479  -2.331691
##        84%        85%        86%        87%        88%        89%
##  -2.329516  -2.327406  -2.324866  -2.322146  -2.319478  -2.315586
##        90%        91%        92%        93%        94%        95%
##  -2.310895  -2.304997  -2.297653  -2.288121  -2.276787  -2.260102
##        96%        97%        98%        99%       100%
##  -2.238366  -2.204012  -2.151610  -2.031222  90.358518
```

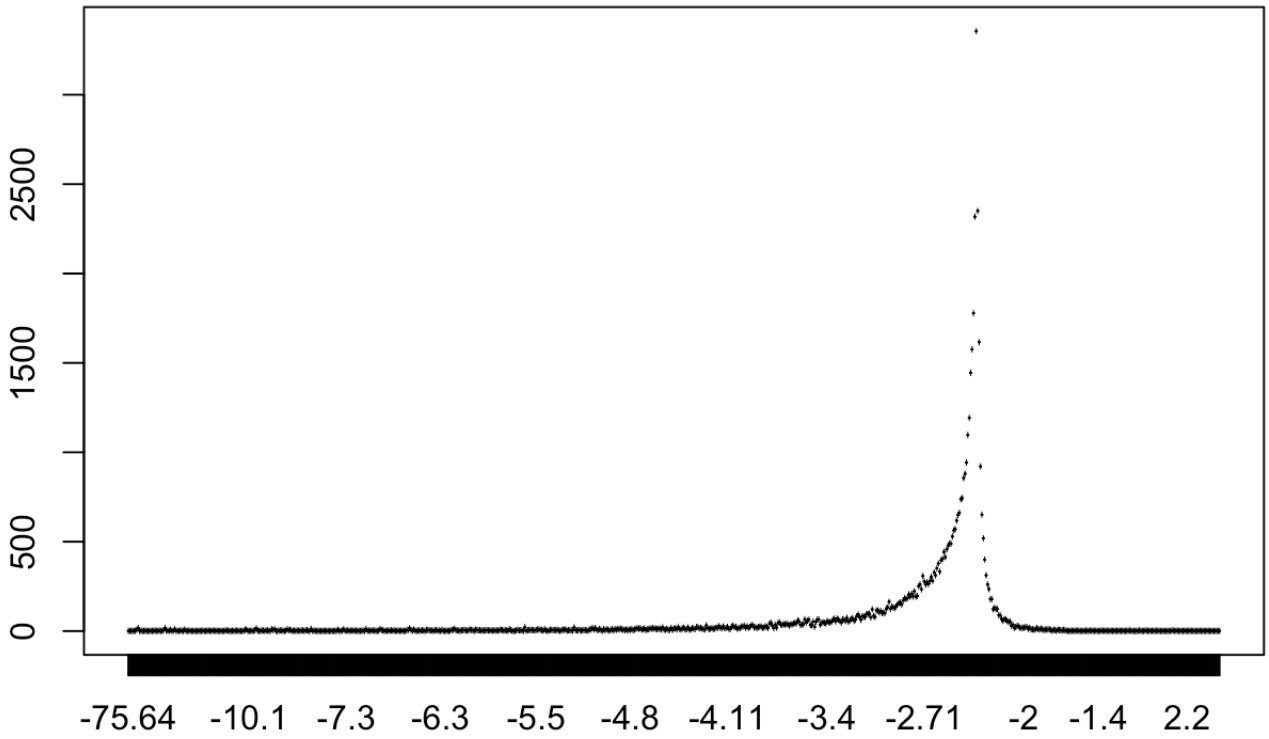
```
predicted_freq <- data.frame(table(round(wm$protest_predicted, 2)))
tail(predicted_freq, 100)  # I like -2.11 because decay rate seems to slow there
```

```
##      Var1 Freq
## 678 -1.59    1
## 679 -1.56    1
## 680 -1.54    1
## 681 -1.53    1
## 682 -1.51    3
## 683  -1.5    1
```

```
## 684 -1.49    1
## 685 -1.48    3
## 686 -1.46    2
## 687 -1.45    2
## 688 -1.43    1
## 689 -1.42    2
## 690 -1.41    1
## 691  -1.4    1
## 692 -1.39    1
## 693 -1.38    2
## 694 -1.37    2
## 695 -1.33    2
## 696 -1.32    1
## 697 -1.31    3
## 698 -1.29    1
## 699 -1.28    1
## 700 -1.27    2
## 701 -1.25    1
## 702 -1.24    2
## 703  -1.2    1
## 704 -1.17    2
## 705 -1.12    1
## 706 -1.11    2
## 707  -1.1    1
## 708 -1.08    2
## 709 -1.07    1
## 710 -1.01    1
## 711    -1    2
## 712 -0.98    1
## 713 -0.89    4
## 714 -0.86    1
## 715 -0.85    1
## 716 -0.82    1
## 717 -0.77    1
## 718 -0.74    1
## 719 -0.62    1
## 720 -0.59    4
## 721 -0.55    1
## 722 -0.53    1
## 723 -0.49    1
## 724 -0.48    3
## 725 -0.42    3
## 726  -0.4    1
## 727 -0.39    2
## 728 -0.38    1
## 729 -0.33    2
## 730 -0.28    1
```

```
## 731 -0.24    1
## 732 -0.22    2
## 733 -0.21    2
## 734  0.04    1
## 735  0.07    1
## 736  0.13    1
## 737  0.19    1
## 738  0.24    1
## 739  0.26    1
## 740   0.3    1
## 741  0.37    1
## 742  0.41    1
## 743  0.45    2
## 744  0.69    4
## 745  0.75    1
## 746  1.01    2
## 747  1.37    2
## 748  1.39    1
## 749  1.65    1
## 750  1.69    1
## 751  1.77    1
## 752  1.87    2
## 753  2.07    1
## 754   2.2    1
## 755  2.39    2
## 756  2.41    1
## 757   2.7    2
## 758  2.89    1
## 759  3.29    1
## 760  3.71    1
## 761  3.98    1
## 762  3.99    1
## 763  5.35    2
## 764   5.7    1
## 765  7.15    1
## 766  9.29    1
## 767 10.45    1
## 768 11.04    2
## 769 11.33    1
## 770 11.58    1
## 771 11.72    1
## 772 14.95    1
## 773 16.52    1
## 774 17.17    1
## 775 18.48    1
## 776 32.35    1
## 777 90.36    1
```

```
plot(x = predicted_freq$Var1, predicted_freq$Freq)
```




```
wm_hybrid2 <- subset(wm, protest_predicted >= -2.11)
```

YOUR TURN

For this section, we will work with the four other datasets available at this class' respository (https://github.com/ZacharyST/APSA2020_EventDataFromSocialMedia/tree/master/Data). You can start with whichever dataset you want.

Location Information

Look at the strings in the user location field. Can you use them to assign place to the tweets?

```
# Location from user profile here
```

Do any tweets mention place names? You could download a place name dictionary or make your own vector of places.

Actors

Choose a dataset and identify actors documented in tweets. You could use the user profile description or the tweet text. This process could be manual, automatic, or hybrid; your choice.

```
Here
```

Sample 10 Minutes

Load the `teaching_random_sample10min.json` dataset and do the following. 1. Manual. Look at a random collection of the tweets. Do they appear to contain any political events? 2. Automatic. Build a dictionary to identify possible events. 3. Hybrid. Look at the events from Step 2. Would you classify them as actual events?

Before converting the .json to a data frame, I would select a small sample after `readLines`, like 1000 tweets. **Hint: look at the documention for `readLines`.** It takes my MacBook Pro about 15 minutes to convert about 30000 tweets.

```
Here
```

Hong Kong

Load `teaching_HK_2019-03-01_2019-12-31.csv`. Try the following exercises: 1. Keep only English tweets. 2. Keep only tweets from users in Hong Kong. 3. Identify events that are APPEALs or DEMANDs.

```
Here
```

Lebanon

Load `teaching_LB_2019-10-01_2019-12-08.csv` Try the following exercises: 1. Keep only Arabic tweets. 2. Build a dictionary to identify tweets about police. Hint: Use Google Translate for Arabic words. 3. How many events involving police do you find? Do your results change when you aggregate by city-day?

```
Here
```

Women's March

Load `teaching_US_2017-01-21.csv`. Try the following exercises: 1. Read tweets to get a feel for what they are talking about.
2. Separate the data frame into tweets from news organizations and tweets from not news organizations. Hint: you will need to create a vector of account names. 3. Build a dictionary to identify events. 4. Compare the events recorded via newspapers and via the other tweets.

```
Here
```