Assignment 5

Mirrien Liang

26/02/2022

## Import necessary libraries and data:

library(stringr)  
library(tidyverse)  
  
load("translink.RData")

## We want to *tidy* data so that the function can run faster.

### First, we extract post content and date/time:

time=c()   
tweet=c()  
  
for (i in seq\_along(data)) {  
 time[i]=as.character(data[[i]]$created)  
 tweet[i]=data[[i]]$text  
}  
  
f=data.frame(Time = time, Post = tweet)  
# head(f)

### Second, we remove post rows that are irrelevant to bus routes.

After closely examining the data, I selected the following keywords:

Skytrain|SkyTrain|StationAlert|Elevator|elevator|Expo|WCE|West Coast Express

|time|Time|SeaBus|Compass|Handy|morning|Morning|evening|desk|Desk|Transit|

transit|Good|Rehab| night |tonight|will be|changes|Congrats|Pattullo Bridge|

Valentines|year|Year|multi

Due to possible mismatches (e.g., “night” with “knight”), space and case of letter do matter.

pattern\_non\_bus = "Skytrain|SkyTrain|StationAlert|Elevator|elevator|Expo|WCE|West Coast Express|time|Time|SeaBus|Compass|Handy|morning|Morning|evening|desk|Desk|Transit|transit|Good|Rehab| night |tonight|will be|changes|Congrats|Pattullo Bridge|Valentines|year|Year|multi"  
  
df <- f[!grepl(pattern\_non\_bus,f$Post),]

### Third, remove url substrings that can cause confusion:

pattern\_url = "(.\*) (.\*)" #Greedy match to the last space  
for (i in seq\_along(df$Post)) {  
 if (str\_detect(df$Post[i],"http")) {  
 df$Post[i] <- str\_replace(df$Post[i],pattern\_url,"\\1")  
 }  
}

### Now, extract all types of numbers (route#, road/street#, time, date, duration):

Consider the following cases (in order):

1. (Optional) Start with the letter “Hwy” or month, e.g., “Hwy 9” or “Feb 19” (only a few cases)
2. (Optional) Routes start with an upper letter, e.g., “R5”, “N9”
3. (Optional) All numbers after “Regular route” are not bus routes, e.g., “Regular route to 96 and 116, then 116, 75A, Scott Rd”
4. 1 to 3 digits, e.g., “9”, “99”, “240”
5. (Optional) to determine street names and time, find numbers that both satisfy the above conditions and follow by a space or a colon with some digits and patterns as shown below. Some examples could be “5:10”, “10:45am”, “3pm”, “3 PM”, “49 Ave”, “49th”, “30 minutes”, and truncated road names that end with dots “Regular route to Granville & Davie, Pacific, Cambie, Nelson, Cambie Bridge, 6 Ave, 4…”. Some other corner cases such as (and majorly are) typos are also included.

Note that by including strings and symbols like “Regular route”, “Feb”, “am”, “:”, “Ave”, we are able to later distinguish bus routes from other types of numbers.

The pattern consists the following elements:

([A-Z]|Hwy\\s|Regular.\*|Jan\\s|Feb\\s|February\\s|Mar\\s)? # Possible prefix

(\\d){1,3} # 1 to 3 digits

(\\s|:\\d{1,})? # Possible space before suffix or “hour:minute” format

(Ave|St |St,|St/|St…|Rd|th|st|nd|rd|pm|PM|am|AM|min|minutes|due|Station|…)? # Possible suffix

pattern\_number = "([A-Z]|Hwy\\s|Regular.\*|Jan\\s|Feb\\s|\\February\\s|Mar\\s)?(\\d){1,3}(\\s|:\\d{1,})?(Ave|St |St,|St/|St…|Rd|th|st|nd|rd|pm|PM|am|AM|min|minutes|due|Station|…)?"  
df1 <- mutate(df,Numbers = str\_extract\_all(Post,pattern\_number))

### Next, extract bus routes.

Now that we have all numbers extracted, we need to further extract only the bus routes by pattern that “optionally starts with one upper case letter, followed by 1 to 3 digits, and optionally ends with a space”.

While doing so, we also omit rows that contain no bus route info, e.g., a few posts missing bus routes.

The intermediate “Numbers” column is also removed for concision.

Store the result in a new data frame for error detection.

pattern\_bus\_route = "^[A-Z]?\\d{1,3}\\s?$"  
  
for (i in seq\_along(df1$Numbers)){  
 df1$Routes[[i]] <- na.omit(str\_extract(df1$Numbers[[i]],pattern\_bus\_route))  
}  
  
df2 <- df1 %>% filter(!(Routes=="character(0)")) %>%  
 select(-Numbers)

At this point, we should have a total of 697 valid observations (posts).

### Lastly, include Status of posts.

Now that we have a tidy data frame with Date/Time, Post, and each corresponding bus route number(s), we want to determine how a post indicates START and STOP. To do so, we want to mutate a new column “Status” to mark the indication of each post.

Note that the Status is binary: either START (TRUE) or STOP (FALSE).

We want to find statements that (1) contain “regular route”, “onward”, “detour”, “experienc(ing)”, or “suspend”; **and** (2) do not contain “clear”, “ended”, “back”, “over”,“cancel”, or “return”.

Note that some words such as “Glover” may contain these keywords but should not be considered as a determinant, we want to add space before or after.

The pattern used in the following code is:

^(?!.\*( clear|Clear|CLEAR| ended| back| over|cancel|return)) # Exclude them

.\*(regular route|Regular route|onward|detour|experienc|suspend) # Include them

pattern\_status = "^(?!.\*( clear|Clear|CLEAR| ended| back| over|cancel|return)).\*(regular route|Regular route|onward|detour|experienc|suspend)"  
  
df2 <- mutate(df2, Status = str\_detect(Post,pattern\_status))

### Overview of the tidy data frame

Let’s take a look into the tidy data frame:

head(df2,n=3)

## Time  
## 1 2020-02-23 16:08:12  
## 2 2020-02-23 15:42:19  
## 3 2020-02-23 15:36:48  
## Post  
## 1 #RiderAlert 173 Cedar/Coquitlam Station and 174 Rocklin/Coquitlam Station detours over as of 8:05am. Servics back to regular route ^sp  
## 2 #RiderAlert 173 Cedar/174 Rocklin detour 7:35am onwards. Regular route to Dewdney Trunk Rd at Westwood, then Westwo…  
## 3 #RiderAlert 173/174 Coquitlam Central Station detour 7:30am onwards. From Port Coquitlam Station, regular route to…  
## Routes Status  
## 1 173 , 174 FALSE  
## 2 173 , 174 TRUE  
## 3 173, 174 TRUE

Note that in some cases, the post was truncated and failed to show whether a disruption occurs or not. In these cases, the default status will be FALSE, which is an approximation of the status.

For example, in row 32, “#RiderAlert 403 Bridgeport Station/Three Rd/ 407 Bridgeport Station/ Gilbert/ 430 Brighouse Station/ Metrotown Stat…” will be considered as FALSE.

Another corner case is that, for example, in row 112 “2020-02-19 19:23:30 240 Vancouver detour from Fri Feb 28 at 7:00 PM to Sun Mar 1 at 7:00 PM. Regular route to Keith and Forbes, then Fo…”, the post announces a future delay, which should have been considered as neither START or STOP on the posted date. In such cases, the default status will be TRUE, which is an approximation of the status. Since there is only 1 out of 697 instances where a post declares a future disruption, the impact of the case can be ignored.

## Construct the translink() function.

Now that we have everything we need set up, we can formulate the function that takes year, month, day, and hour as arguments and return a list with two vectors *start* and *stop*.

translink <- function(y,m,d,h){  
 # take four arguments of time, e.g., 2020,1,26,3  
   
 # Create an empty list to store results  
 ret = list(start=c(),stop=c())  
   
 # format input ymd\_h  
 # i.e., (2020,1,26,3) -> "2020-01-26 03"  
 if(nchar(m)==1){m=str\_c("0",m)}  
 if(nchar(d)==1){d=str\_c("0",d)}  
 if(nchar(h)==1){h=str\_c("0",h)}  
 date = str\_c(y,m,d,sep="-")  
 datetime = str\_c(date,h,sep = " ")  
   
 # Match time, then match Status, then combine/append vectors  
 for (i in seq\_along(df2$Time)) {  
 if (str\_detect(df2$Time[[i]], datetime)){  
 if (df2$Status[[i]]) {  
 ret$start <- c(ret$start, df2$Routes[[i]])  
 } else {  
 ret$stop <- c(ret$stop, df2$Routes[[i]])  
 }  
 }  
 }  
   
   
 # Remove all the Space in the Routes column  
 ret$start <- gsub('\\s+', '', ret$start)  
 ret$stop <- gsub('\\s+', '', ret$stop)  
   
 # Delete duplicates  
 ret$start = unique(ret$start)  
 ret$stop = unique(ret$stop)  
   
 # If no match, append warning message  
 if (length(ret$start)==0) {ret$start <- c(ret$start,"No detour had started")}  
 if (length(ret$stop)==0) {ret$stop <- c(ret$stop,"No detour had ended")}  
   
 return(ret)  
}

## Test cases.

Now we want to test the function with some general or corner cases.

### (1) On 2020-01-26, at 3 a.m., there are 3 posts and they are:

## [1] "#RiderAlert 23 Beach/Main Street Station detour UPDATE 7:25pm: clear, resuming regular route.^jd"   
## [2] "#RiderAlert 406 Steveston/Brighouse Station detour 6:30 PM. Regular route to No 3 Rd &amp; Saba, then Saba, Buswell, We…"  
## [3] "#RiderAlert 401 One Road/Brighouse Station detour 6:30 PM. Regular route to Westminster &amp; Minoru Blvd, Minoru, Gran…"

From the text we can see that route 23 resumed regular route (STOP), route 406 and 401 started to detour (START). The function produces the correct results.

translink(2020, 1, 26, 3)$start

## [1] "406" "401"

translink(2020, 1, 26, 3)$stop

## [1] "23"

### (2) On 2020-02-13, at 17 o’clock, there are 6 posts:

## [1] "#RiderAlert 99 UBC/Commercial-Broadway detours have cleared, service is back to regular route with residual delays. ^CK"   
## [2] "#RiderAlert 4,9,14,R4,25,33,49,480 detours in the UBC areas have cleared. Some delays as service returns to normal.…"   
## [3] "$RiderAlert 9:15 AM R4 Joyce/ 49 Metrotown/ 480 Bridgeport Station detour. Regular route from UBC Loop, Wesbrook Ma…"   
## [4] "#RiderAlert 9:10 AM R4/49/480 UBC detour. Regular route to Marine Dr &amp; Wesbrook Mall, then Marine Dr-Chancellor Blv…"  
## [5] "#RiderAlert 9:00 AM 25 Brentwood/33 29th Ave Station detour. From UBC Loop, Wesbrook Mall northbound, Chancellor Bl…"   
## [6] "#RiderAlert 9:00 AM 25 UBC/33 UBC detour. Regular route to 16 Ave &amp; Wesbrooke, then 16 Ave, marine Dr-Chancellor Bl…"

The posts can be easily read by human-mind but are a disaster for machine: first, they are in STOP, STOP, START, START, START, and START, respectively; second, there are multiple routes involved.

We are expecting to see that routes 99, 4, 9 ,14, R4, 25, 33, 49, and 480 detours had cleared.

We are expecting to see that routes R4, 49, 480, 25, and 33 started detour (duplicates are removed).

The function produces the expected results.

translink(2020,2,13,17)

## $start  
## [1] "R4" "49" "480" "25" "33"   
##   
## $stop  
## [1] "99" "4" "9" "14" "R4" "25" "33" "49" "480"

### (3) On 2020-02-12, at 19 o’clock, there are 9 posts:

## [1] "10 Granville detour 11:45am. Regular route to Hastings and Granville, then Hastings, Howe, Granville Bridge, resume…"   
## [2] "#RiderAlert14 Hastings/16 29 Ave Stn detour. Regular route to Granville and Davie, then Davie, Seymour, Hastings re…"   
## [3] "#RiderAlert 20 Victoria detour. Regular route to Robson and Seymour, then Seymour, Hastings resume regular route due to march.^RR"  
## [4] "#RiderAlert4 Powell/7Dunbar/50 Waterfront Stn detour. Regular route to Granville and Davie, then Davie, Seymour, Co…"   
## [5] "#RiderAlert 240 15 St/246 Highland/257 Horseshoe Bay detour has ended. Resuming regular route.^RR"   
## [6] "#RiderAlert 50 False Creek detour. Regular route to Cordova and Granville, then Cordova, Howe, Granville Bridge, re…"   
## [7] "#RiderAlert 14 UBC/16 Arbutus detour. Regular route to Hastings and Granville, then Hastings, Howe, Granville Bridg…"   
## [8] "#RiderAlert 4 UBC/7 Dunbar detour. Regular route to Pender and Granville, then Pender, Howe, Granville Bridge, resu…"   
## [9] "#RiderAlert 240 15th Street/246 Highland/257 Horseshoe Bay Express detour. Regular route to Georgia and Seymour, th…"

We are expecting start to have route 10, 14, 16, 20, 4, 7, 50, 240, 246, and 257 with duplicates removed.

We are expecting stop to have route 240, 246, and 257.

The function produces the following results:

translink(2020,2,12,19)

## $start  
## [1] "10" "14" "16" "20" "4" "7" "50" "240" "246" "257"  
##   
## $stop  
## [1] "240" "246" "257"

### (4) On 2020-02-14, at 13 o’clock, there is just one post:

## [1] "#RiderAlert 342 Langley Centre detour. Regular route to 60 Ave &amp; 152 St, then 152 St, 64 Ave, 168 St, Hwy 10, resum…"

Despite occurrences of many numbers, we only expect route 342 in start.

translink(2020,2,14,13)

## $start  
## [1] "342"  
##   
## $stop  
## [1] "No detour had ended"

### (5) On 2020-02-12, at 6 a.m., there are 6 posts:

## [1] "#RiderAlert As of 10:00pm: 99 UBC B-Line/Commercial-Broadway Station B-Line is experiencing delays up to 30 minutes due to march. ^nv"   
## [2] "#RiderAlert As of 10:00PM: 84 UBC/VCC-Clark Station is experiencing delays up to 30 due to march. ^nv"   
## [3] "#RiderAlert As of 10:00pm: 17 Oak/Downtown is experiencing delays up to 55 minutes due to march. ^nv"   
## [4] "#RiderAlert As of 10:00pm: 50 False Creek/Granville Island/Waterfront Station is experiencing delays up to 35 minutes due to march. ^nv"  
## [5] "#RiderAlert As of 10:00pm: 15 Cambie/Olympic Village Station is experiencing delays up to 35 minutes due to march. ^nv"   
## [6] "#RiderAlert As of 10pm: 9 Alma/UBC/Commercial-Broadway Station/Boundary is experiencing delays up to 85 minutes due to march. ^nv"

The statements mix together time, route numbers, and delay minutes, but we are expecting to find route 99, 84, 17, 50, 15, and 9 in start and none in stop.

translink(2020,2,12,6)

## $start  
## [1] "99" "84" "17" "50" "15" "9"   
##   
## $stop  
## [1] "No detour had ended"

## Summary

Overall, I have examined closely into each of 968 observations to find keywords/patterns that can distinguish:

1. bus-related posts from other irrelevant ones such as skytrain, seabus, and etc.,
2. numbers of bus routes from numbers of other types such as time, station number, street number and etc., and
3. posts that indicates a start of a disruption from posts announcing an end of a disruption.

Furthermore, most corner cases are tested and are found to be covered by such patterns. It is confident to say that the function is competent to perform the required task.