R Notebook

Introduction

Motivation: We want methods that depend a little as possible on specific words. Also, would be nice to know if we could learn from mental diseases from social media data. To do this, we study speech transcriptions from youtube videos. Here we are going to study three illness: bipolar, depression and schizophrenia.

The bipolar disease is a mood disorder that makes an individual go trough major depressive episodes and manic episodes. Here, we are only interested in manic episodes, since the bipolar depression is difficult to differentiate from unipolar depression. Maniac episodes are characterized by very fast speech (pressured speech), racing thoughts and difficulty to focus. In the data, we expect increased number of words in a period of time and many word repetitions.

People with schizophrenia many times present a very laconic behavior (poverty-of-speech). Also, some individuals suffer with a "confusion", that decreases their cognitive abilities. So we would expect decreased number of words compared to bipolar.

People with depression can be letargic, confused and have a very negative world view. We would expect less confusion than people with schizophrenia and bipolarity. Also, less words spoken in comparison with bipolar.

Schizophrenia and depression are difficult to compare, since we don't have much information about the subjects, so we don't know how depressed are the individuals and if they are in a schizophrenic crysis period. Also, even with highly controled populations, not every person present the same symptoms.

Some speech mistakes can be indication of less cognitive resources. For example, when concentraded in some parallel task while speaking, people usually produce longer pauses between words and fillers, like "um", "ah", "so", and others. We assume that serious mental illness (depression, bipolar and schizophrenia), limit the available resources and make people produce more disfluencies.

Hypothesis: Disfluencies (pauses and fillers, like "um", "ah", ...) and vocabulary diversity can identify speech transcriptions of people that report having depression, manic episodes and schizophrenia.

The dataset is composed of transcriptions of videos from YouTube. In these videos, individuals report being depressive, schizophrenic or being in a manic episode of bipolarity. The transcriptions have also the time markings of what was said, here called utterances. The silent pauses are marked with sp and filled pauses with fp.

Data dictionary:

sp means "sillent pause", is a time period in which the person doesn't make any sound _fp means "filled pause", is a filler sound that people make while they are organizing what they are going to say [word] is a action or happening that occured during the video is an utterance that wasn't able to be transcribed word + "-" (word followed by dash) means that the person did not complete the word.

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ----------------------------------------------------------- tidyverse 1.2.1
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.7
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts -------------------------------------------------------------- tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()    masks stats::lag()
```

```
## Had to change colnames that conflicted with R functions. (dplyr::id and stats::end)
df <- read_delim('data.csv',delim = ',')
```

```
## Parsed with column specification:
## cols(
##   group = col_character(),
##   id = col_character(),
##   max_length = col_double(),
##   begin = col_double(),
##   end = col_double(),
##   utt = col_character()
## )
```

```
colnames(df)[2] <- 'ID'
colnames(df)[5] <- 'END'

head(df)
```

```
## # A tibble: 6 x 6
##   group  ID    max_length begin   END utt
##   <chr>  <chr>      <dbl> <dbl> <dbl> <chr>
## 1 schizo sf1         418.  0     0.6  SIL_begin
## 2 schizo sf1         418.  0.6   1.08 so
## 3 schizo sf1         418.  1.08  1.3  i'm
## 4 schizo sf1         418.  1.3   1.78 back
## 5 schizo sf1         418.  1.78  2.25 sp
## 6 schizo sf1         418.  2.25  2.84 um
```

How many instances are in each group?

```
df %>%
  group_by(ID,group) %>%
  summarise(n = n()) %>%
  group_by(group) %>%
  count()
```

```
## # A tibble: 3 x 2
## # Groups:   group [3]
##   group         nn
##   <chr>      <int>
## 1 bipolar        4
## 2 depression     7
## 3 schizo         8
```

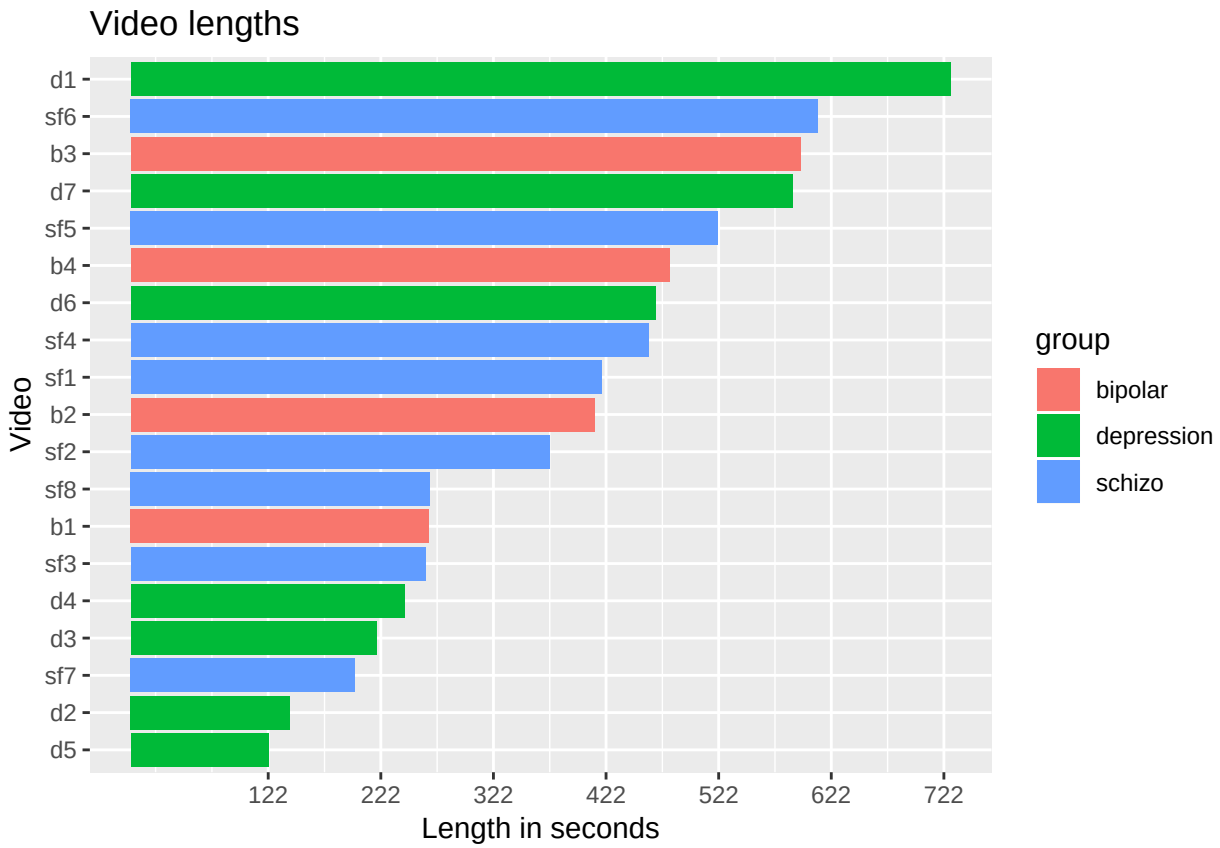
What are the video lengths?

```
df %>%
  group_by(ID,group,max_length) %>%
  summarise(n = n()) %>%
  arrange(max_length) -> video.lengths

print(video.lengths)
```

```
## # A tibble: 19 x 4
## # Groups:   ID, group [19]
##    ID    group      max_length     n
```

```
##    <chr> <chr>           <dbl> <int>
##  1 d5    depression       122.   215
##  2 d2    depression       141.   319
##  3 sf7   schizo           199.   554
##  4 d3    depression       218.   544
##  5 d4    depression       243    585
##  6 sf3   schizo           262.   469
##  7 b1    bipolar          265.   840
##  8 sf8   schizo           266.   607
##  9 sf2   schizo           372.  1039
## 10 b2    bipolar          412.  1078
## 11 sf1   schizo           418.   791
## 12 sf4   schizo           460.  1183
## 13 d6    depression       466.  1009
## 14 b4    bipolar          478.  1590
## 15 sf5   schizo           522.  1666
## 16 d7    depression       588.  1690
## 17 b3    bipolar          595.  2056
## 18 sf6   schizo           610.  1498
## 19 d1    depression       728.  1418
```

```
video.lengths %>%
  ggplot(aes(y=max_length,x=reorder(ID,max_length),fill = group)) +
  geom_bar(stat='identity') +
  scale_y_continuous(breaks = seq(122, 727,by=100)) +
  ggtitle('Video lengths') +
  ylab('Length in seconds') +
  xlab('Video') +
  coord_flip()
```



Is there strange words? That is, some kind of dirt from data annotation?

```
df %>%
  filter(grepl("[^[:alnum:] | \' ]", utt)) %>% ## _is not alphanumeric_ or is apostrophe
  select(utt) %>%
  unique()
```

```
## # A tibble: 119 x 1
##    utt
##    <chr>
##  1 SIL_begin
##  2 [sigh]
##  3 um_fp
##  4 [nose]
##  5 ah_fp
##  6 [singing]
##  7 <unk>
##  8 [noise]
##  9 pa-
## 10 [mama_i_love_you]
## # ... with 109 more rows
```

The hyphen must be in the end to get the stutterings. Going to define a helper functions to remove non word symbols.

```
is.sound <- function(utt){
  grepl('\\[', utt)
}
```

```
is.filled.pause <- function(utt){
  (grepl('\\_fp', utt) | grepl('\\_FP', utt) | (utt == 'um'))
}

is.sil <- function(utt){
  grepl('SIL', utt)
}

is.stuttering <- function(utt){
  grepl('[:alnum:]-', utt)
}

is.silent.pause <- function(utt){
  utt == 'sp'
}

is.um <- function(utt){
  grepl('\\<um\\>', utt)
}

is.unk <- function(utt){
  grepl('<', utt)
}

## Mega slow
is.word <- function(utt){
  ! (is.sound(utt) | is.filled.pause(utt) | is.sil(utt) | is.stuttering(utt) | is.silent.pause(utt) | is
}

# is.word <- function(utt){
#  ! grepl("[^[:alnum:] | \' ]",utt)
# }

##Testing
# df %>%
#   filter(is.word(utt))
```

We need to now the *duration* of the utterances, not when it begins and when it ends.

```
df %>%
  mutate(uttlen = END - begin) -> df

head(df)
```

```
## # A tibble: 6 x 7
##   group  ID    max_length begin   END utt       uttlen
##   <chr>  <chr>      <dbl> <dbl> <dbl> <chr>      <dbl>
## 1 schizo sf1         418.  0     0.6  SIL_begin  0.6
## 2 schizo sf1         418.  0.6   1.08 so         0.48
## 3 schizo sf1         418.  1.08  1.3  i'm        0.220
## 4 schizo sf1         418.  1.3   1.78 back       0.48
## 5 schizo sf1         418.  1.78  2.25 sp         0.47
## 6 schizo sf1         418.  2.25  2.84 um         0.590
```

How to represent the evolution of speech? That is, the number of words spoken through time? Possible

solution: create a new table, with groups, ids and timesteps. For each timestep, count the number of words that an individual said until that time.

```
## Max timestep. After 122s, we begin to loose instances, since not many people have long videos.
## However, we put a little more time.
t.sample <- seq(0,500,by=1)

unique(df$ID) %>%
  ##For each InDividual, create a subtable with a column with the id and a column with timesteps
  lapply(
         function(idd){
           data.frame(
             ID = rep(idd, length(t.sample)), times = t.sample, stringsAsFactors = FALSE)
           }
         ) %>%
  ##Concatenate subtables.
  bind_rows() %>%

  #This join is a little slow if you increase the number of timesteps
  inner_join(df, by='ID') %>%
  group_by(ID,times) %>%
  filter(times >= END, is.word(utt) ) %>%
  count() %>%
  ungroup() -> ids.time.evolution

head(ids.time.evolution)
```

```
## # A tibble: 6 x 3
##   ID    times     n
##   <chr> <dbl> <int>
## 1 b1        2     3
## 2 b1        3     5
## 3 b1        4    11
## 4 b1        5    13
## 5 b1        6    14
## 6 b1        7    18
```

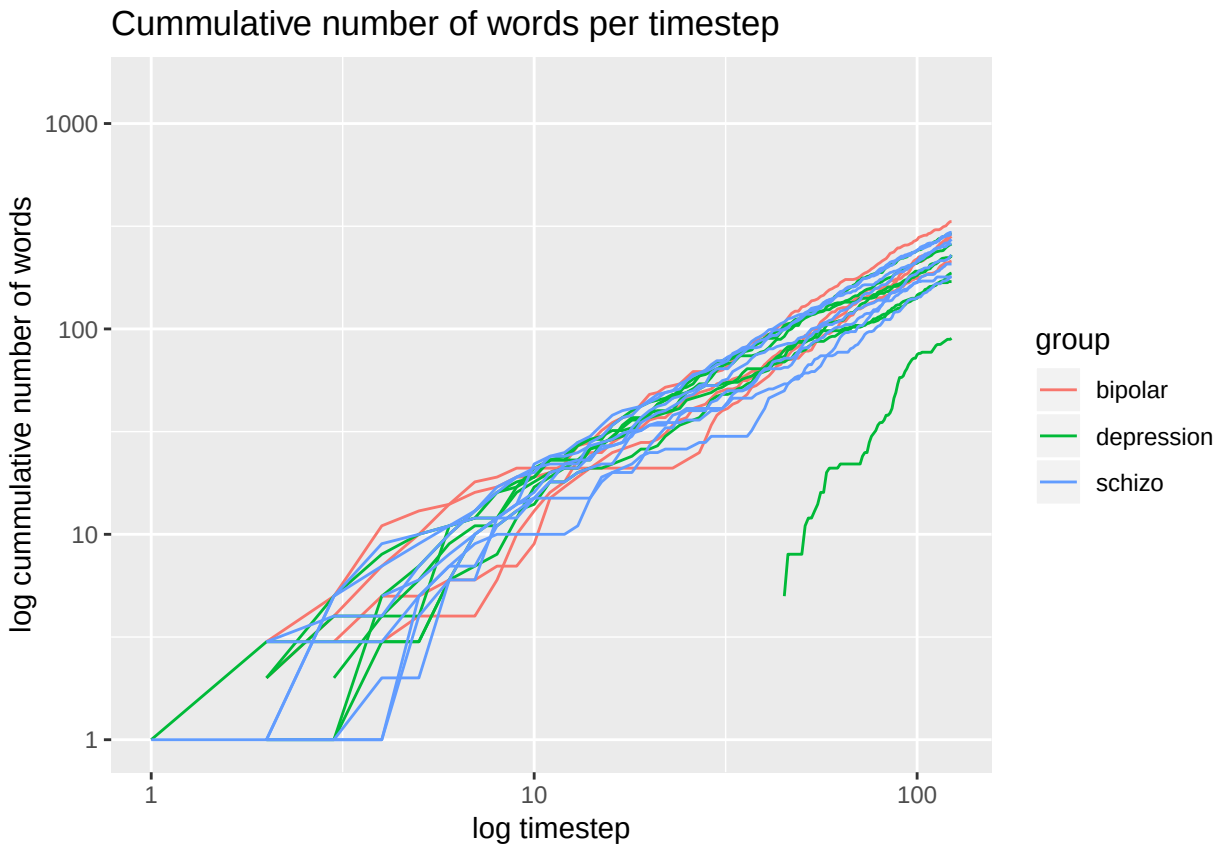
Now, we need the evolution by groups. We want a cummulative and a "point-based" number of words in a interval measures.

```
ids.time.evolution %>%
  left_join(df %>% group_by(group,ID) %>% summarise(k = 0), by='ID') %>%
  select(group,ID,times,n) -> evolution.cumm
```

Cummulative number of words per interval. Going to plot all the individuals to get a grip of the data and to inspect patterns.

```
evolution.cumm %>%
  ggplot(aes(x=times,y=n,group=ID)) +
  scale_y_continuous(trans = 'log10') +
  scale_x_continuous(trans = 'log10',limits = c(1,123)) +
  ylab('log cummulative number of words') +
  xlab('log timestep') +
  ggtitle('Cummulative number of words per timestep') +
  geom_line(aes(color=group))
```

```
## Warning: Removed 7163 rows containing missing values (geom_path).
```



Who is the late talker? Answer: The D1 begins at 43.88!!!

```
df %>%
  filter(is.word(utt)) %>%
  group_by(group,ID) %>%
  summarise(l = min(begin))
```

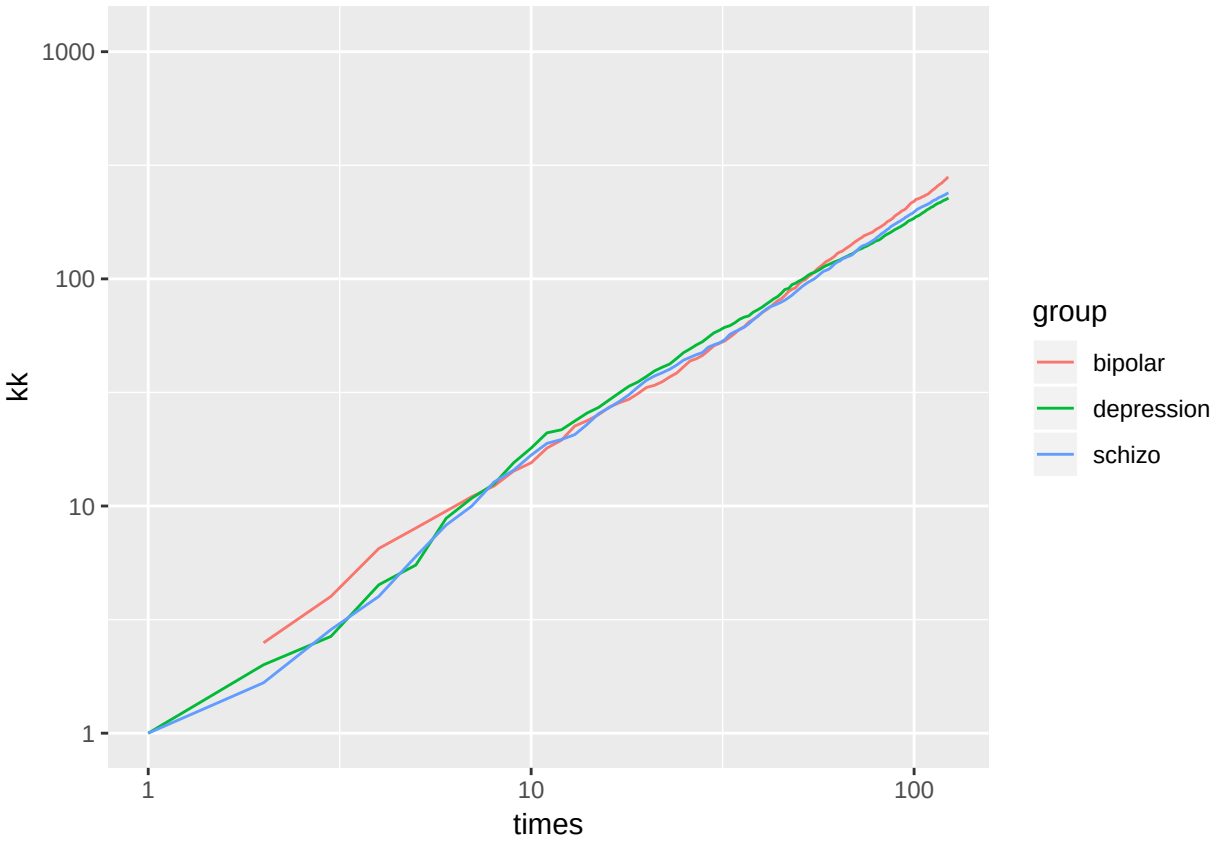
```
## # A tibble: 19 x 3
## # Groups:   group [?]
##    group      ID        l
##    <chr>      <chr> <dbl>
##  1 bipolar    b1     1.36
##  2 bipolar    b2     2.80
##  3 bipolar    b3     1.73
##  4 bipolar    b4     1.11
##  5 depression d1    43.9
##  6 depression d2     1.43
##  7 depression d3     1.09
##  8 depression d4     2.52
##  9 depression d5     0.67
## 10 depression d6     0.88
## 11 depression d7     2.27
## 12 schizo     sf1    0.6
## 13 schizo     sf2    0.2
## 14 schizo     sf3    1.02
## 15 schizo     sf4    1.73
## 16 schizo     sf5    3.11
## 17 schizo     sf6    2.39
```

```
## 18 schizo     sf7    0.91
## 19 schizo     sf8    1.62
```

The last plot was not helpful. Going to plot the mean between the individuals for each group. That is, the mean series.

```
evolution.cumm %>%
  filter(ID != 'd1') %>% ## removed D1 since it made the value decrease for depression and added noise
  group_by(group,times) %>%
  summarise(kk = mean(n)) %>%
  ggplot(aes(x=times,y=kk,group=group)) +
  scale_y_continuous(trans = 'log10') +
  scale_x_continuous(trans = 'log10',limits = c(1,123)) + ##After 123 we start to lose individuals
  geom_line(aes(color=group))
```

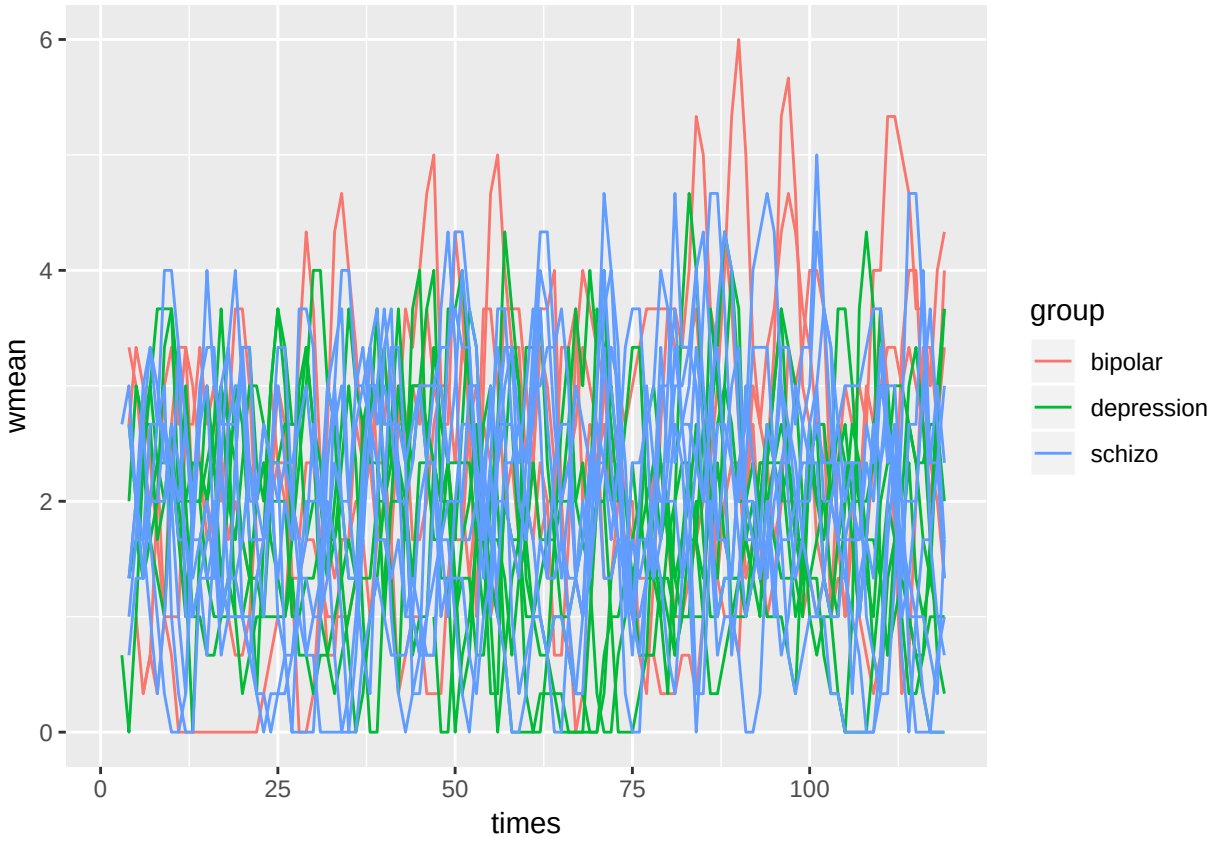
```
## Warning: Removed 1131 rows containing missing values (geom_path).
```



Not helpful either. Mania seem to talk a bit more. However, we begin having less and less data after 123s. Going to focus in the instaneous measure of number of words. We are going to regain how many words are said in a second by the $\delta = x_{i+1} - x_i$, where x_i is the cummulative number of words.

```
#Roll mean
library(RcppRoll)

# How many words were said in the timestep delta (x_i+1 - x_i). We use a rolling window to smoothe
# the time series. We have little data and the variance number of words that a person can say in a smal
ids.time.evolution %>%
  group_by(ID) %>%
  # filter(times < 120) %>%
```

```
  mutate(word.inc = n - lag(n)) %>% ## Interval
  mutate(wmean = roll_mean(word.inc,n=3,fill = NA) ) %>% ## Rolling window mean
  left_join(df %>% group_by(group,ID) %>% summarise(k = 0), by='ID') %>% ##Retrieving group names
  select(group,ID,times,wmean,word.inc) -> evolution.wmean

evolution.wmean %>%
  filter(times < 120) %>% # Comment for even trashier plot
  ggplot(aes(x=times,y=wmean,group=ID)) +
  geom_line(aes(color=group),na.rm = TRUE )
```



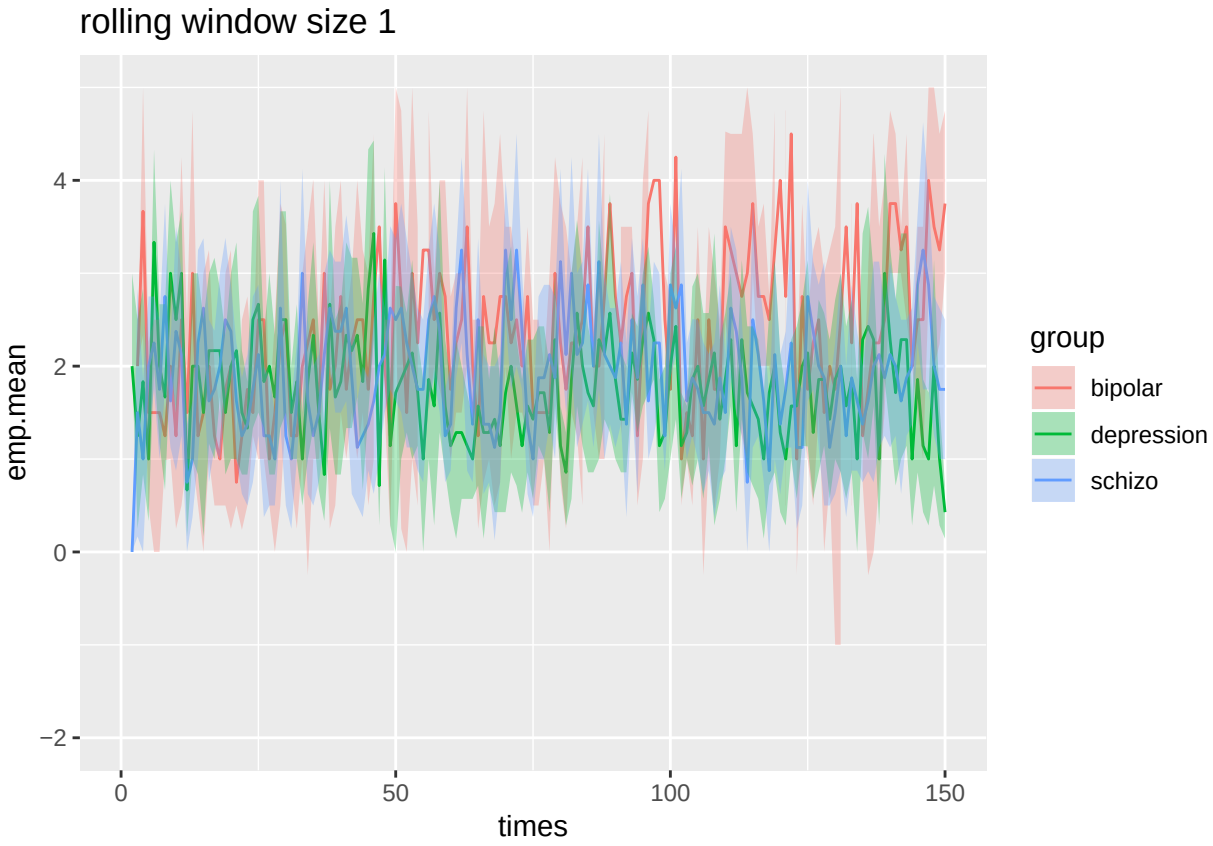
(trash). Now the mean number of words in a instant of time.

```
##Empirical bootstraping of confidence intervals
bootstrap.ci <- function(emp.data,conf,n.samples){

  emp.mean <- mean(emp.data)

  ## Sampling all points togheter without a loop
  tmp <- sample(emp.data, length(emp.data) * n.samples, replace = TRUE)
  boot.samples <- matrix(tmp, nrow = length(emp.data), ncol = n.samples )

  ##Bootstrap means for the n.samples
  boot.sample.means <- colMeans(boot.samples, na.rm = TRUE)

  boot.cutoffs <- boot.sample.means - emp.mean

  it <- quantile(boot.cutoffs, c(conf, 1-conf ))
```

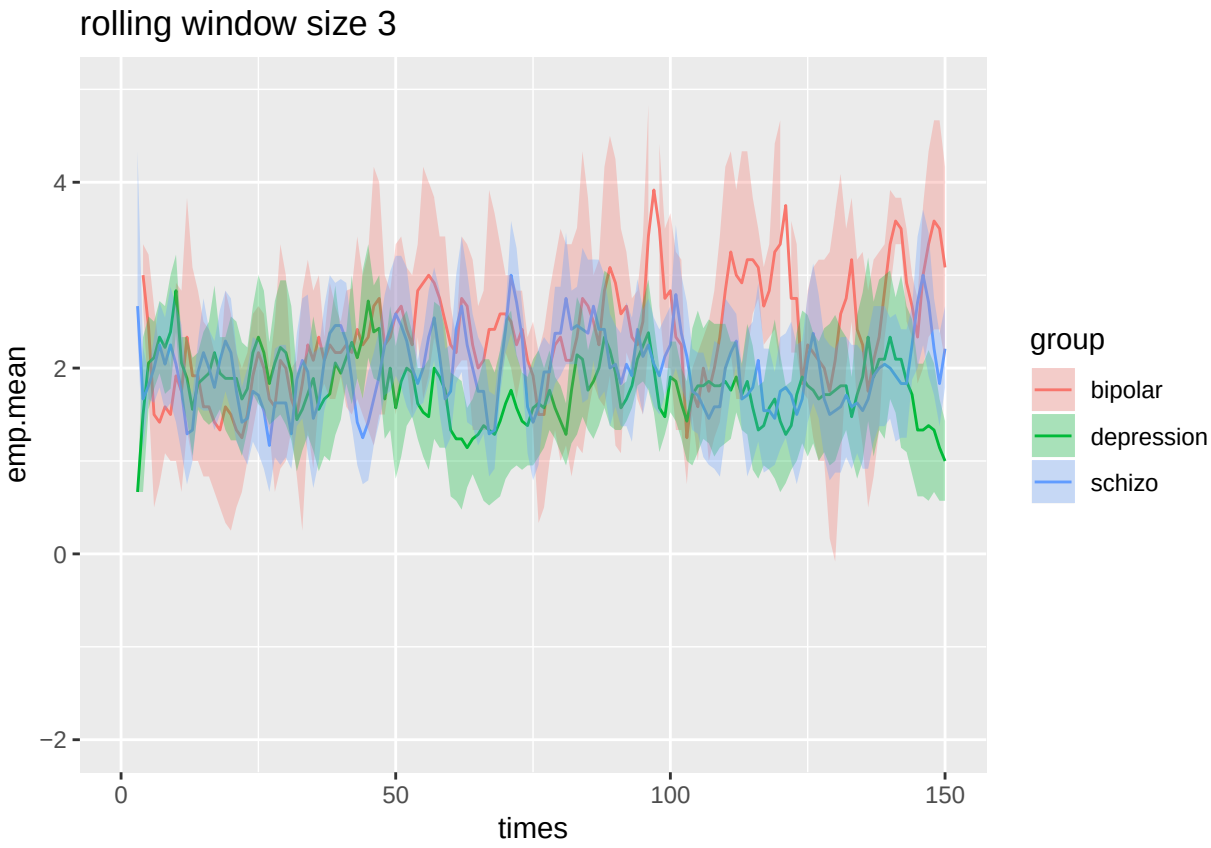
```
  ans <- as.data.frame(list(emp.mean = emp.mean, c1 = emp.mean - it[2], c2= emp.mean - it[1]  ))
  return( ans )
}

# In each point of the time series, my sample is too small to be able to assume the std known.
# So, to have more reliable confidence intervals, I assume the sigma unkown and thus use the
# t distribution to estimate the intervals.
ci.student <- function(emp.data,conf){
  emp.mean <- mean(emp.data)
  emp.sd <- sd(emp.data)

  n <- length(emp.data)
  it <- qt(1 - (conf/2),df=n-1) * (emp.sd/sqrt(n))
  ans <- as.data.frame(
    list(
      emp.mean = emp.mean,
      c1 = emp.mean - it,
      c2 = emp.mean + it)
    )
  return( ans )

}

for(i in c(1,3,5,10,15,30)){
ids.time.evolution %>%
  group_by(ID) %>%
  # filter(times < 120) %>%
  mutate(word.inc = n - lag(n)) %>% ## Interval
  mutate(wmean = roll_mean(word.inc,n=i,fill = NA) ) %>% ## Rolling window mean
  left_join(df %>% group_by(group,ID) %>% summarise(k = 0), by='ID') %>% ##Retrieving group names
  select(group,ID,times,wmean,word.inc) -> evolution.wmean

evolution.wmean %>%
  filter(!is.na(wmean)) %>%
  group_by(group,times) %>%
  # do(ci.student(.$wmean,0.05)) %>%  ##UNCOMMENT FOR NEAT T STUDENT - mean CIs!
  do(bootstrap.ci(.$wmean,0.05,1000)) %>% ##UNCOMMENT FOR NEAT BOOTSTRAP CIs!
  select(times,emp.mean,group,c1,c2) -> ans

  p <- ans %>% ggplot(aes(x=times,y=emp.mean,group=group)) +
  ylim(c(-2,5)) +
  xlim(c(0,150)) +
  geom_line(aes(color=group)) +
  ggtitle(paste('rolling window size',as.character(i))) +
  geom_ribbon(aes(ymin=c1,ymax=c2,fill=group),alpha=0.3)

  print(p)
#   evolution.wmean %>%
#     group_by(ID,group) %>%
#     summarise(mean=mean(wmean,na.rm = TRUE),mean=sd(wmean,na.rm = TRUE)) -> ans
#
#   p <- ans %>% ggplot(aes(x=group,y=mean)) +
#     ggtitle(as.character(i)) +
```

```
#     geom_boxplot() +
#     geom_point()
# print(p)

}
```

```
## Warning: Removed 1050 rows containing missing values (geom_path).
```

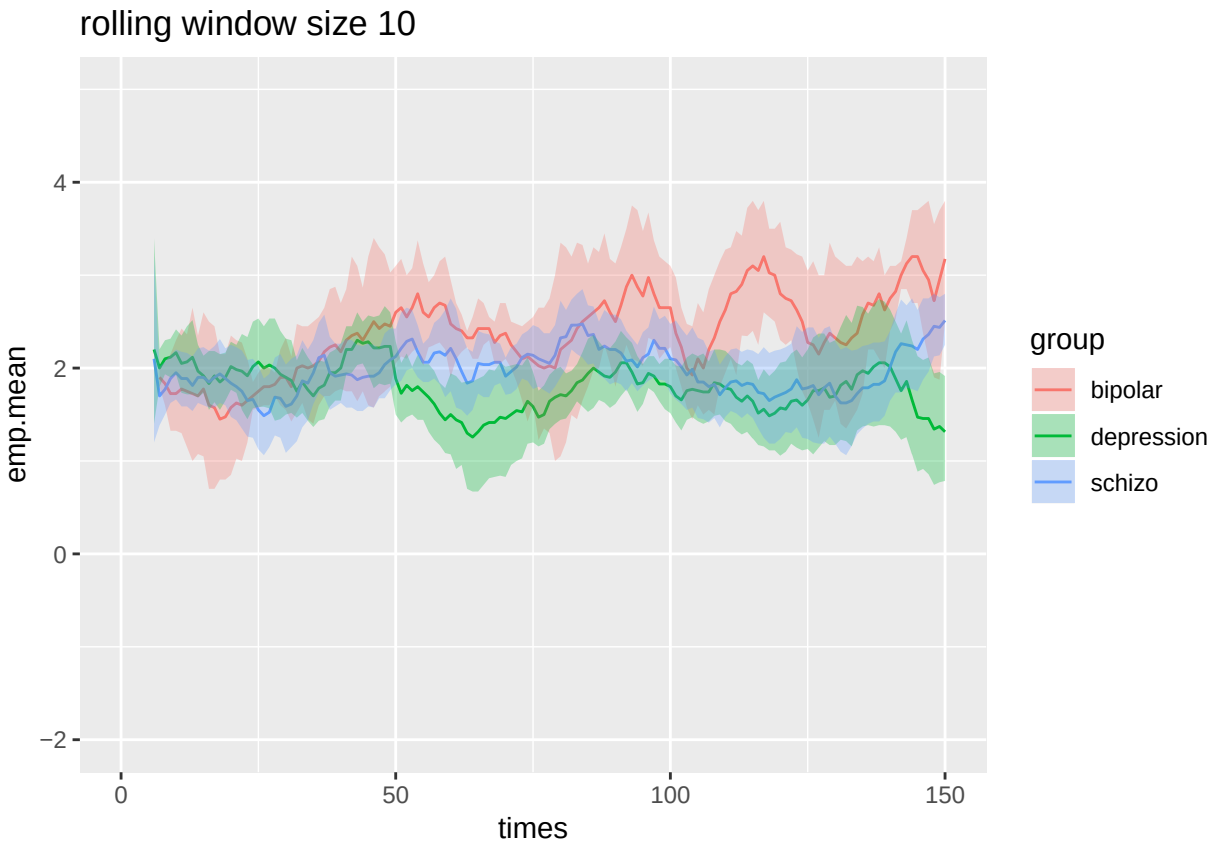



```
## Warning: Removed 1047 rows containing missing values (geom_path).
```

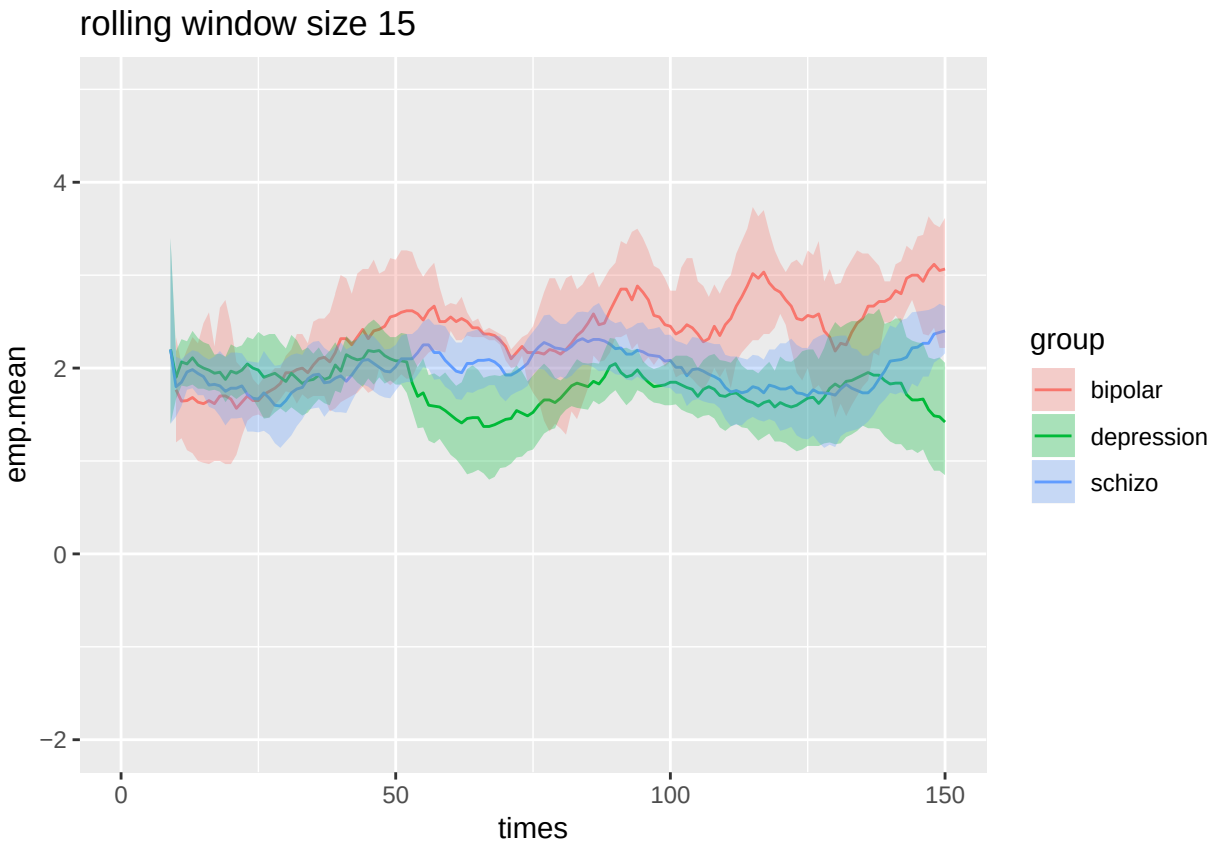



```
## Warning: Removed 1044 rows containing missing values (geom_path).
```

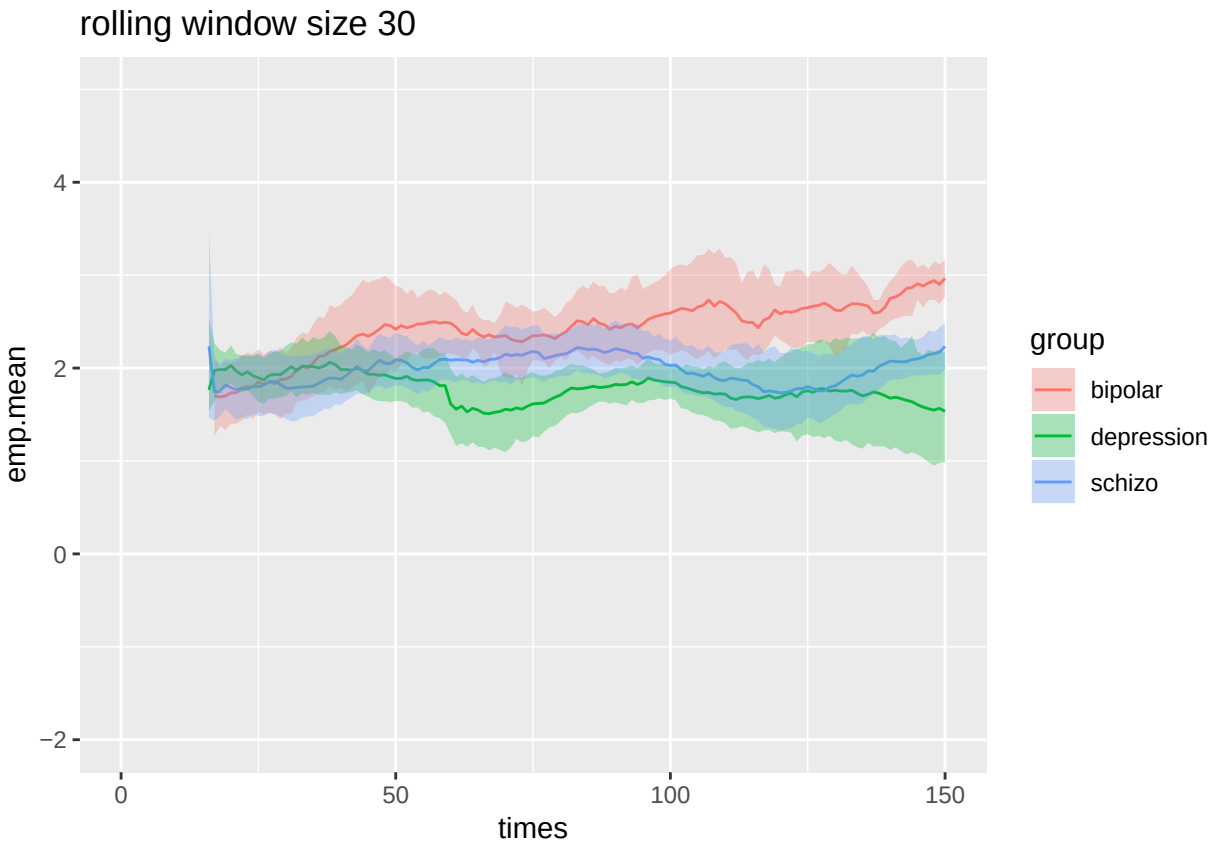

```
## Warning: Removed 1035 rows containing missing values (geom_path).
```




```
## Warning: Removed 1029 rows containing missing values (geom_path).
```




```
## Warning: Removed 1005 rows containing missing values (geom_path).
```

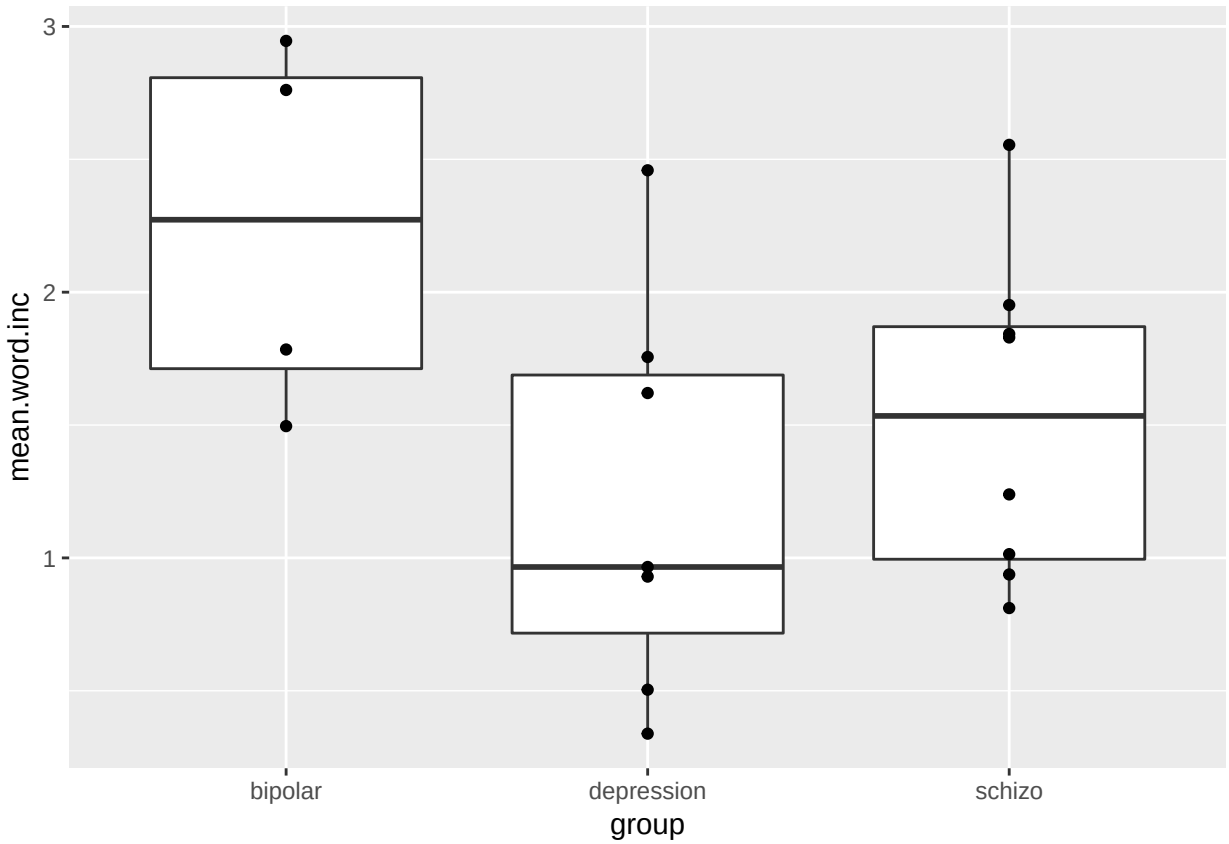


There's a tradeoff between the timestep size and data sparsity. If we make the timestep of size 0.5, for example, lots of time points would have number of words zero. So we picked a sensible value: 1. The rolling window mean size add some smoothness to the series. The colored bands are the confidence intervals. We can't assume normality of the data, so we didn't use the z distribution, anyway, we also implemented the confidence intervals using the t distribution. These confidence intervals are very conservative, so we opted to a simulation approach. We used the empirical bootstrap confidence interval link.

To have strong conclusions, I would need more data. However, bipolar seem to have more words per second. I'd also like to investigate if the series are periodic.

Distribution of increases.

```
ids.time.evolution %>%
  group_by(ID) %>%
  mutate(word.inc = n - lag(n)) %>% ## Interval
  filter(!is.na(word.inc)) %>%
  left_join(df %>% group_by(group,ID) %>% summarise(k = 0), by='ID') %>% ##Retrieving group names
  ungroup() %>%
  group_by(ID,group) %>%
  summarise(mean.word.inc = mean(word.inc)) -> ids.word.increase

ids.word.increase %>%
  ggplot(aes(y=mean.word.inc,x=group)) +
  geom_boxplot() +
  geom_point()
```



Rate of pauses and mean time of pauses.

Distribution of pauses.

```
df %>%
  filter(group == 'depression') %>%
  filter(utt == 'sp') %>%
  .$uttlen %>%
  log() %>%
  shapiro.test() %>%
  print()
```

```
##
##  Shapiro-Wilk normality test
##
## data:  .
## W = 0.99326, p-value = 0.00124
```

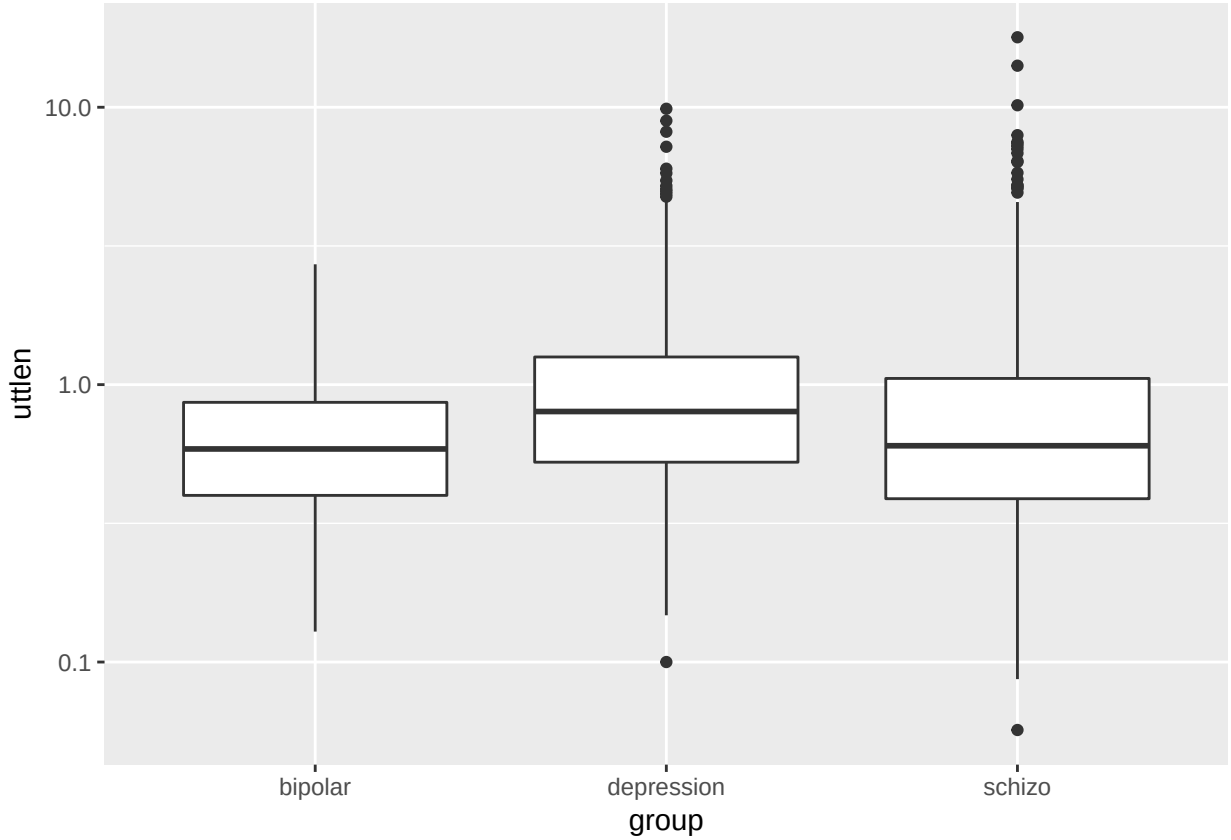
```
df %>%
  filter(group == 'bipolar') %>%
  filter(utt == 'sp') %>%
  .$uttlen %>%
  log() %>%
  shapiro.test() %>%
  print()
```

```
##
##  Shapiro-Wilk normality test
##
## data:  .
## W = 0.99653, p-value = 0.3172
```

```
df %>%
  filter(group == 'schizo') %>%
  filter(utt == 'sp') %>%
  .$uttlen %>%
  log() %>%
  shapiro.test() %>%
  print()
```

```
##
##  Shapiro-Wilk normality test
##
## data:  .
## W = 0.98902, p-value = 3.751e-07
```

```
df %>%
  filter(utt == 'sp') %>%
  ggplot(aes(y=uttlen,x=group)) +
  scale_y_continuous(trans = 'log10') + ##Without log the plot is very uninformative

  geom_boxplot()
```



The pauses distribution are very long tailed. Who are the individuals with the long pauses?

```
df %>%
  filter(utt == 'sp', uttlen > 5) %>%
  arrange(desc(uttlen))
```

```
## # A tibble: 26 x 7
##    group      ID    max_length begin   END utt   uttlen
##    <chr>      <chr>      <dbl> <dbl> <dbl> <chr>  <dbl>
##  1 schizo     sf3          262. 191.  208.  sp     17.9
##  2 schizo     sf3          262. 113.  127.  sp     14.1
##  3 schizo     sf6          610. 292.  302.  sp     10.2
##  4 depression d1           728. 607.  616.  sp      9.88
##  5 depression d1           728. 639.  648.  sp      8.94
##  6 depression d5           122.  55.4  63.6 sp      8.16
##  7 schizo     sf3          262. 103.  110.  sp      7.93
##  8 schizo     sf3          262. 166.  174.  sp      7.50
##  9 schizo     sf1          418. 392.  399.  sp      7.40
## 10 schizo     sf8          266. 210.  217.  sp      7.25
## # ... with 16 more rows
```

Even a pause of 5 seconds is very long. So, we are cutting pauses greater than 8 seconds, since this length of pause is too idiosyncratic.

We try a log-normal fit. If we add the idiosyncratic pauses, the log-normal will understimate the tail values. The fit by group can be inspected commenting the filter line. The bipolar present a very good fit, but for schizophrenia and depression, the log-normal understimate the tail values, this can be observed in the Q-Q plot.

```
## Ugly fix to unmask the _select_ function name conflict
#
# library('fitdistrplus') ## CAN'T FIX THE SELECT NAME CONFLICT
#
# unloadNamespace("tidyverse")
# library('tidyverse')
#
# df %>%
#   filter(utt == 'sp',uttlen < 8) %>%
#   # filter(group=='schizo') %>% ##Change groups here! <------------
#   mutate(luttlen = log(uttlen)) %>%
#   # select(uttlen) %>%
#   select(luttlen) %>%
#   unlist() %>%
#   as.vector() %>%
#   # fitdist(distr = 'lnorm') %>%
#   fitdist(distr = 'norm') %>%  ## It's taking the mean and sd, however we get plots!
#   plot()
```

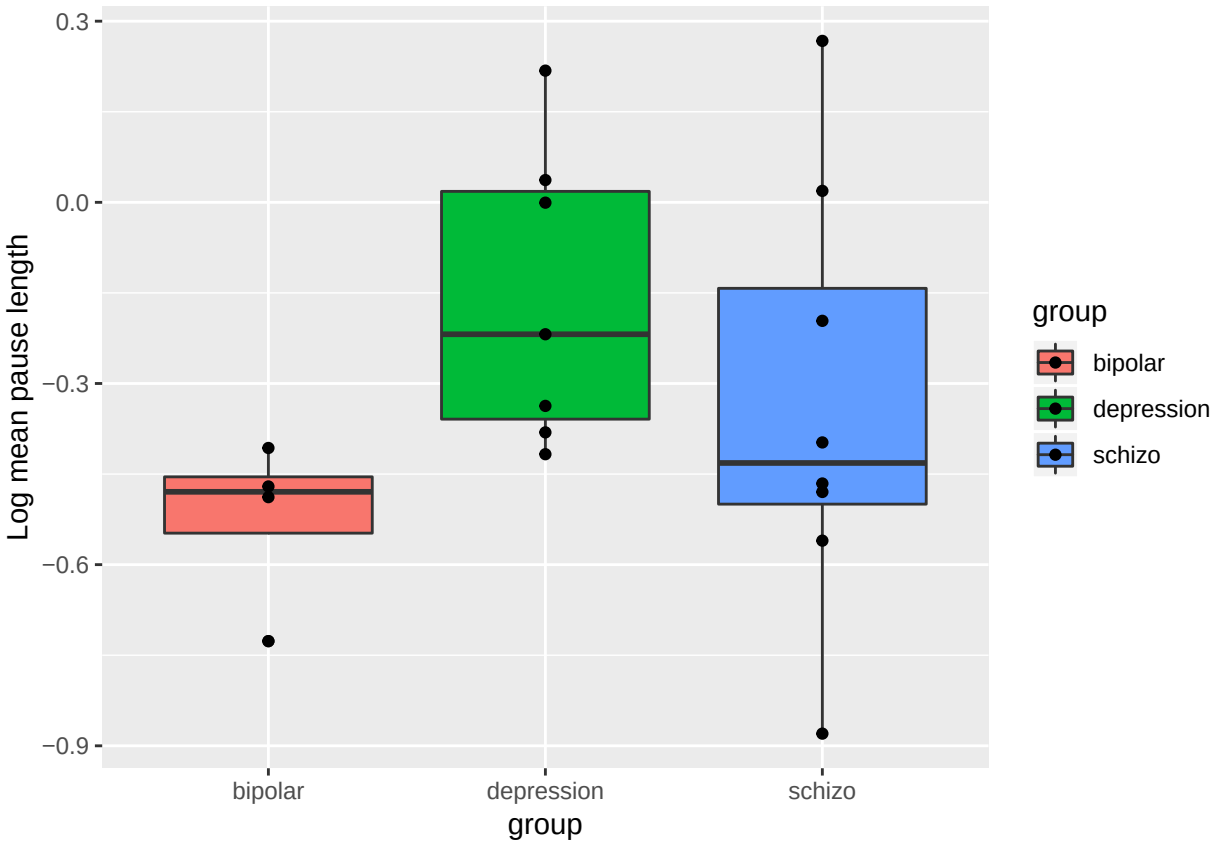
```
df %>%
  filter(utt == 'sp', uttlen < 8) %>%
  mutate(luttlen = log(uttlen)) %>%
  select(luttlen) %>%
  summarise(m = mean(luttlen),sd=sd(luttlen))
```

```
## # A tibble: 1 x 2
##        m    sd
##    <dbl> <dbl>
## 1 -0.386 0.728
```

```
# Cummulative distr... not important
# df %>%
#   filter(utt == 'sp', uttlen < 8) %>%
#   mutate(luttlen = log(uttlen)) %>%
#   ggplot(aes(x=luttlen)) +
#   geom_line(aes(y =..y..), stat='ecdf') +
#   stat_function(aes(x=uttlen),fun=pnorm,color="blue",args=list(mean=-0.386,sd=0.72)) +
#   scale_x_continuous(trans = 'log10',limits=c(NA,15)) +
#   scale_y_continuous(trans = 'log10',limits=c(NA,1))
```

Now, we want to know how the (log) mean pause length for each individual speech describe the groups.

```
# lnorm.fit <- function(obs){
#   ans <- fitdist(obs,'lnorm') ## can produce nans during the optimization, once they search in negati
#   ss <- ans$estimate
#   mlog <- ss[1]
#   sdlog <- ss[2]
#   return(
#     as.data.frame(
#       list(mlog = mlog, sdlog = sdlog)
#       )
#   )
# }

lnorm.fit <- function(obs){
  ans <- log(obs)
  mlog <- mean(ans)
  sdlog <- sd(ans)
  return(
    as.data.frame(
      list(mlog = mlog, sdlog = sdlog)
      )
  )
}

df %>%
  filter(utt == 'sp', uttlen < 8) %>%
  group_by(group,ID) %>%
  do(lnorm.fit(.$uttlen)) -> lognorm.fit

lognorm.fit %>%
  ggplot(aes(x=group,y=mlog,fill=group)) +
  ylab('Log mean pause length') +
  geom_boxplot() +
  geom_point()
```



The bipolar group have shorter pauses than depression (t=-3.161,p=0.005), however the difference from schizophrenia is not significative.

```
print( t.test(lognorm.fit$mlog[lognorm.fit$group == 'bipolar'],
              lognorm.fit$mlog[lognorm.fit$group == 'depression'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  lognorm.fit$mlog[lognorm.fit$group == "bipolar"] and lognorm.fit$mlog[lognorm.fit$group == "de
## t = -3.1611, df = 8.9559, p-value = 0.0058
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.1536417
## sample estimates:
##  mean of x  mean of y
## -0.5230465 -0.1570340
```

```
print( t.test(lognorm.fit$mlog[lognorm.fit$group == 'bipolar'],
              lognorm.fit$mlog[lognorm.fit$group == 'schizo'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  lognorm.fit$mlog[lognorm.fit$group == "bipolar"] and lognorm.fit$mlog[lognorm.fit$group == "sc
## t = -1.2888, df = 9.8013, p-value = 0.1135
## alternative hypothesis: true difference in means is less than 0
```

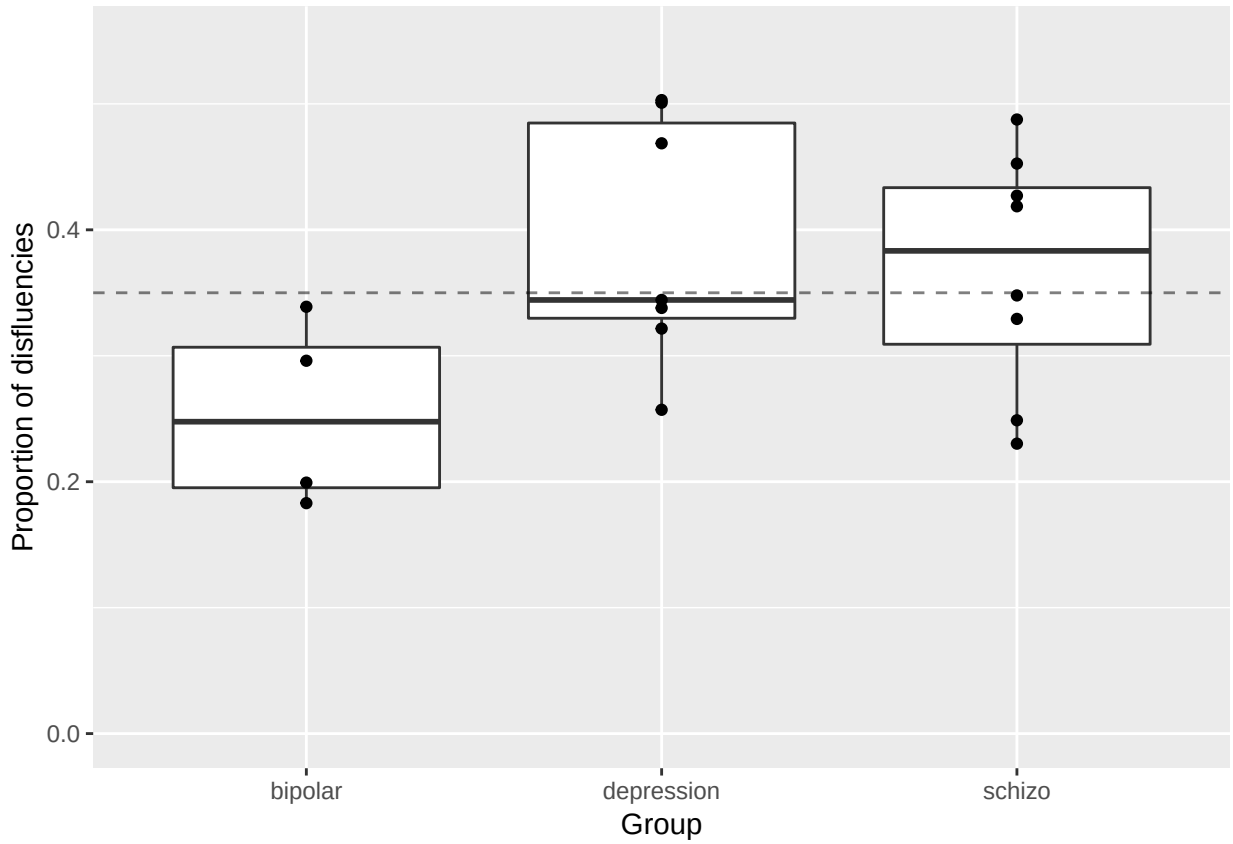
```
## 95 percent confidence interval:
##        -Inf 0.07629536
## sample estimates:
##  mean of x  mean of y
## -0.5230465 -0.3366153
```

```
print( t.test(lognorm.fit$mlog[lognorm.fit$group == 'schizo'],
              lognorm.fit$mlog[lognorm.fit$group == 'depression'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  lognorm.fit$mlog[lognorm.fit$group == "schizo"] and lognorm.fit$mlog[lognorm.fit$group == "dep
## t = -1.1471, df = 12.344, p-value = 0.1365
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf 0.09878923
## sample estimates:
##  mean of x  mean of y
## -0.3366153 -0.1570340
```

Time spent on pauses+disfluencies vs time spent speaking

```
df %>%
  filter(!is.na(utt)) %>%
  group_by(group, ID) %>%

  mutate(ssp = if_else(utt == 'sp' & uttlen < 8, uttlen, 0),
         sword = if_else(is.word(utt), uttlen, 0),
         sfp = if_else(is.filled.pause(utt), uttlen, 0)) %>%

  summarise(t.pause = sum(ssp),
            t.word = sum(sword),
            t.fp = sum(sfp)) %>%
  mutate(prop.disf = (t.pause + t.fp)/(t.word + t.pause + t.fp) ) -> times.disfluencies


times.disfluencies %>%
  ggplot(aes(y=prop.disf,x=group)) +
  geom_boxplot() +
  xlab('Group') +
  ylab('Proportion of disfluencies') +
  ylim(0,0.55) +
  geom_abline(slope=0, intercept=0.35, linetype = 'dashed', alpha=0.5) +
  geom_point()
```




```
times.disfluencies %>%
  ungroup() %>%
  summarise(mean = mean(prop.disf), sd=sd(prop.disf)) %>%
  print()
```

```
## # A tibble: 1 x 2
##    mean    sd
##   <dbl> <dbl>
## 1 0.352 0.102
```

The bipolar group produces less disfleuncies than the depression and schizophrenia groups. However, the difference between schizo e depression it's not significative. T test for unequal variances.

```
print( t.test(times.disfluencies$prop.disf[times.disfluencies$group == 'bipolar'],
              times.disfluencies$prop.disf[times.disfluencies$group == 'depression'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  times.disfluencies$prop.disf[times.disfluencies$group == "bipolar"] and times.disfluencies$pr
## t = -2.5704, df = 7.9475, p-value = 0.01664
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.03758772
## sample estimates:
## mean of x mean of y
## 0.2543008 0.3905162
```

```
print( t.test(times.disfluencies$prop.disf[times.disfluencies$group == 'bipolar'],
              times.disfluencies$prop.disf[times.disfluencies$group == 'schizo'],
              alternative = 'less'))
```

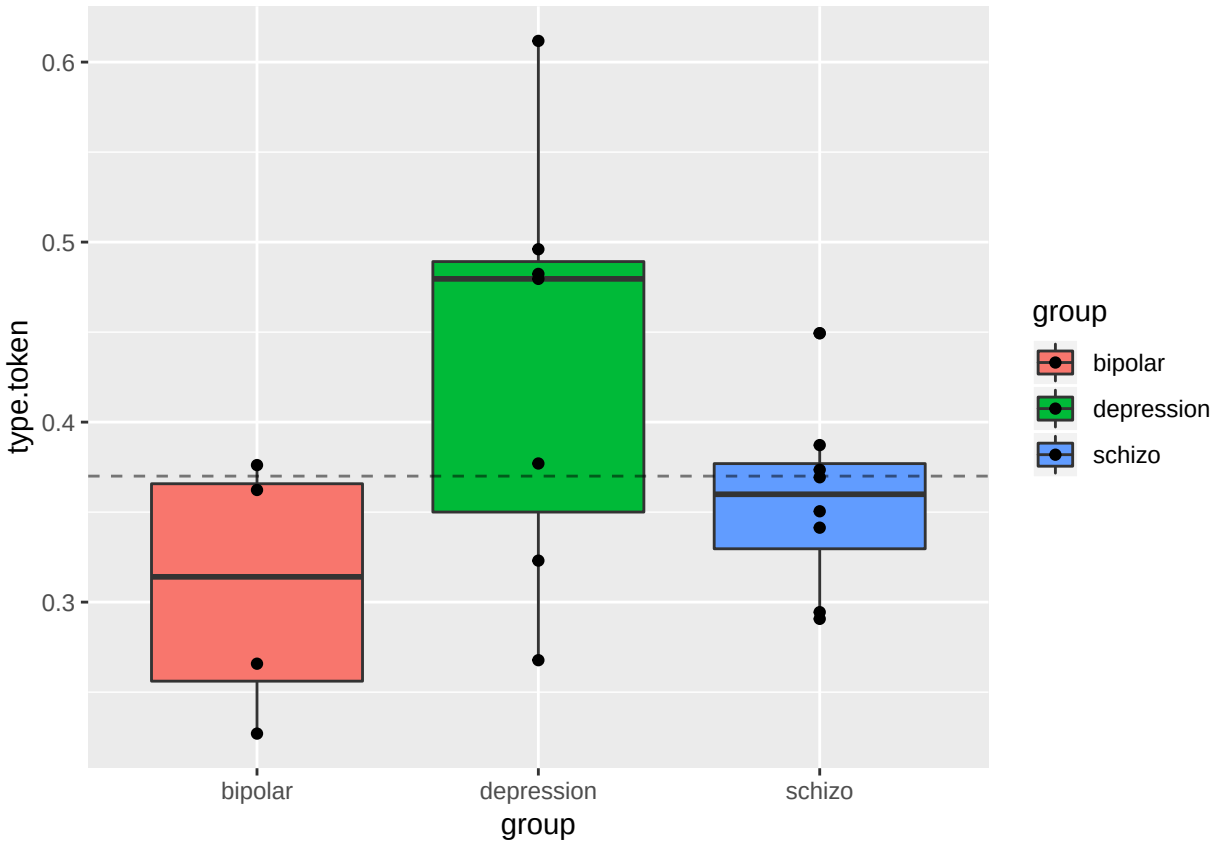
```
##
##  Welch Two Sample t-test
##
## data:  times.disfluencies$prop.disf[times.disfluencies$group == "bipolar"] and times.disfluencies$pro
## t = -2.2531, df = 7.5783, p-value = 0.02805
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.01913905
## sample estimates:
## mean of x mean of y
## 0.2543008 0.3677676
```

```
print( t.test(times.disfluencies$prop.disf[times.disfluencies$group == 'depression'],
              times.disfluencies$prop.disf[times.disfluencies$group == 'schizo'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  times.disfluencies$prop.disf[times.disfluencies$group == "depression"] and times.disfluencies$
## t = 0.45418, df = 12.566, p-value = 0.6713
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf 0.1116856
## sample estimates:
## mean of x mean of y
## 0.3905162 0.3677676
```

Type token ratio

```
df %>%
  group_by(ID,group) %>%
  filter( is.word(utt) )  %>%
  summarise(tokens = length(utt),
            types = length( unique(utt))) %>%
  mutate(type.token = types/tokens) -> type.token.ids



type.token.ids %>%
  ggplot(aes(x=group,y=type.token, fill=group)) +
  geom_boxplot() +
  geom_point() +
  geom_abline(slope=0, intercept=0.37, linetype = 'dashed', alpha=0.5)
```




```
type.token.ids %>%
  ungroup() %>%
  summarise(mean = mean(type.token), sd=sd(type.token)) %>%
  print()
```

```
## # A tibble: 1 x 2
##    mean     sd
##   <dbl>  <dbl>
## 1 0.375 0.0950
```

Bipolar have less diversity in vocabulary than depression, however we didn't find significative differences to schizo group.

```
print( t.test(type.token.ids$type.token[type.token.ids$group == 'bipolar'],
              type.token.ids$type.token[type.token.ids$group == 'depression'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  type.token.ids$type.token[type.token.ids$group == "bipolar"] and type.token.ids$type.token[ty
## t = -2.1938, df = 8.8143, p-value = 0.02826
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.02048558
## sample estimates:
## mean of x mean of y
## 0.3077977 0.4339299
```

```
print( t.test(type.token.ids$type.token[type.token.ids$group == 'bipolar'],
              type.token.ids$type.token[type.token.ids$group == 'schizo'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  type.token.ids$type.token[type.token.ids$group == "bipolar"] and type.token.ids$type.token[ty
## t = -1.2098, df = 4.5548, p-value = 0.1427
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf 0.03459371
## sample estimates:
## mean of x mean of y
## 0.3077977 0.3570560
```

```
print( t.test(type.token.ids$type.token[type.token.ids$group == 'schizo'],
              type.token.ids$type.token[type.token.ids$group == 'depression'],
              alternative = 'less'))
```

```
##
##  Welch Two Sample t-test
##
## data:  type.token.ids$type.token[type.token.ids$group == "schizo"] and type.token.ids$type.token[typ
## t = -1.6004, df = 7.9782, p-value = 0.07414
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf 0.01248128
## sample estimates:
## mean of x mean of y
## 0.3570560 0.4339299
```

most frequent words

I could remove stopwords (me, that, the, an, and, ...), however we would still have little frequency of meaningful words.

```
df %>%
  filter(is.word(utt)) %>%
  group_by(group) %>%
  count(utt) %>%
  mutate(prop = prop.table(n)) %>%
  arrange(group,desc(prop)) %>%
  top_n(70) -> groups.freqs
```

```
## Selecting by prop
```

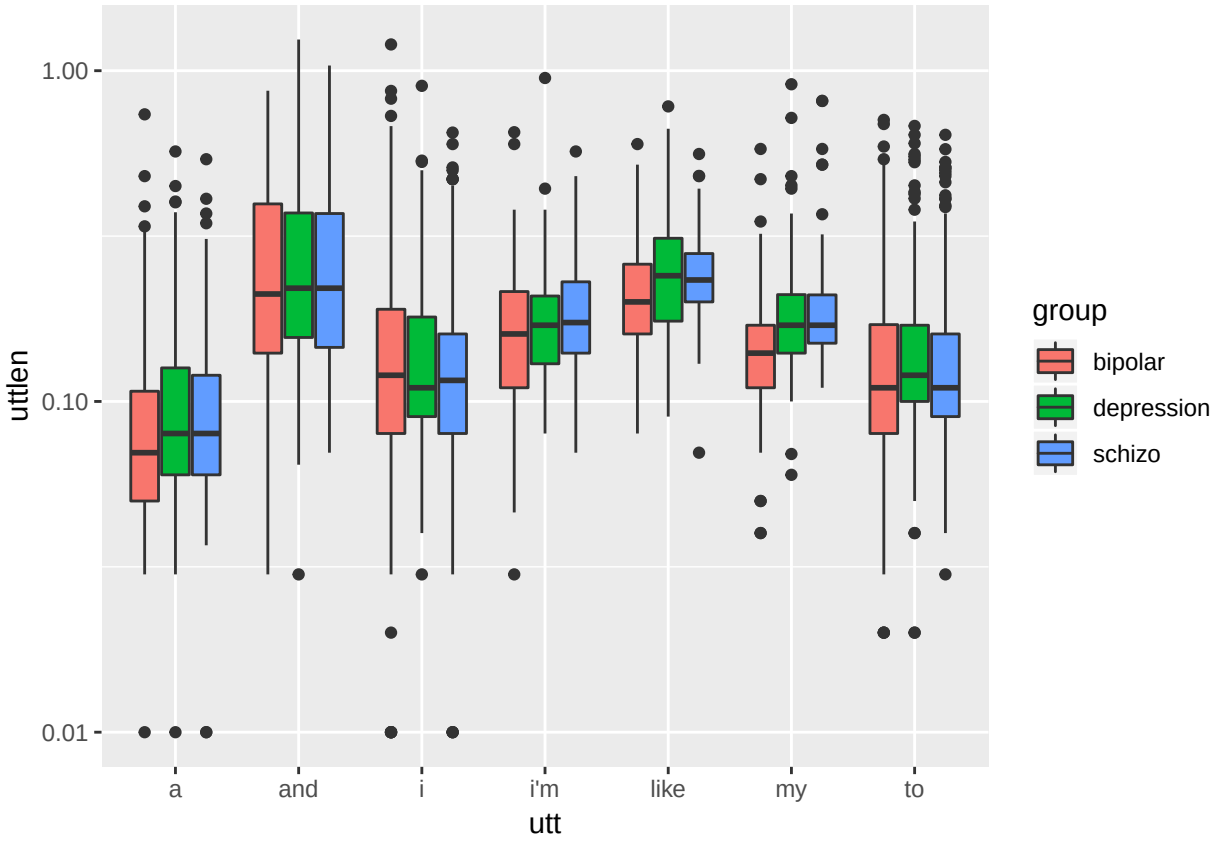
```
groups.freqs
```

```
## # A tibble: 224 x 4
## # Groups:   group [3]
##    group   utt       n   prop
##    <chr>   <chr> <int>  <dbl>
##  1 bipolar i       370 0.0768
##  2 bipolar like    193 0.0401
##  3 bipolar and     171 0.0355
##  4 bipolar to      110 0.0228
```

```
##  5 bipolar i'm      87 0.0181
##  6 bipolar you      82 0.0170
##  7 bipolar it       80 0.0166
##  8 bipolar a        79 0.0164
##  9 bipolar know     77 0.0160
## 10 bipolar that     76 0.0158
## # ... with 214 more rows
```

Testing the time dist for some words.

```
df %>%
  filter(utt == 'i' | utt == 'like' | utt == 'and' | utt == 'to' | utt == "i'm" | utt == "a" | utt == "
  ggplot(aes(x=utt,y=uttlen,fill=group)) +
  scale_y_continuous(trans = 'log10') +
  geom_boxplot()
```



As last resort we are going to use all variables studied in last sessions to try to inspect if the groups are separable. To do this, we use t-sne, a technique of reduction of dimensionality. We project the four variables in 2 dimensions and inspect if the plotted points appear clustered in their groups. However, after several tries, the points don't form clusters that reflect the groups.

With the information presented in the last sessions, we can't confirm the hyphotesis.

```
type.token.ids %>%
  left_join(times.disfluencies, by=c('ID','group')) %>%
  left_join(lognorm.fit, by=c('ID','group')) %>%
  left_join(ids.word.increase, by=c('ID','group')) %>%
  ungroup() %>%
  select(group,mlog,prop.disf,type.token,mean.word.inc)-> discriminative.df
```

```r
library(Rtsne)

discriminative.df %>%
  select(-group) %>%
  # scale() %>%
  as.matrix() %>%
  Rtsne(.,dims = 2,perplexity = 6,max_iter=50000, theta=0) -> tsne.out

ot.data <- bind_rows(list(group = discriminative.df$group,
                          x = tsne.out$Y[,1],
                          y = tsne.out$Y[,2]))

ggplot(ot.data) +
  geom_point(aes(x=x,y=y,color=group),size=7)
```