**CAL Project - Knowledge Transfer Documentation - Data Engineering 2**

**4/28/2020**

For questions about this file, email musch.sam@gmail.com.

# Introduction

This whole file is to get our data in the right format to run clusters. At the end of this file, we are going to have:

- 1 row per person
- 1 column for each broad category

This section is just getting our libraries & datasets ready.

```
library(tidyverse)   # general purpose
library(flexclust)   # for clustering
library(data.table)  # for large files
library(dtplyr)      # for large files
library(lubridate)   # for dates


indy <- read_csv('../individual_info.csv') %>%
  select(ID_DEMO, MEMBERSHIP_TYPE_CODE,
         MEMBERSHIP_STATUS_CODE)


events <- read_csv('../events_cleaned.csv') %>%
  select(ID_DEMO, DATE_EVENT, YEAR_FISCAL,
         broad_cat, NBR_GROUP) %>%
  inner_join(indy, by = 'ID_DEMO')


emails <- fread('../emails_cleaned.csv') %>%
    merge.data.table(indy, by = 'ID_DEMO')
```

# Emails

We want to look at someone's open rate for our clusters. However, clicks are categorized separately. Here, we are just creating a column that indicates if a person clicked or opened an email.

Additionally, we are making sure that we are only looking at the relevant categories.

```
emails <- emails %>%
  select(ID_DEMO, DATE_CONTACT, YEAR_FISCAL,
         CODE_PATH, CODE_OUTCOME, broad_cat) %>%

  filter(broad_cat %in%
         c('Learning', 'Legislature',
           'Social', 'Sports')) %>%

  # New column: actual opens
  mutate(click_open =
         ifelse(CODE_OUTCOME == 'CL' |
                CODE_OUTCOME == 'OE', 1, 0))
```

This chunk is answering: "What is the person's open rate?"

*Note that dplyr::summarize is just to make sure that R doesn't accidently think we are using a different package.*

```
email_avg <-
  emails %>% group_by(ID_DEMO) %>%

  # Total possible, total clicked
  dplyr::summarize(total_possible = n(),
                   total_clicked = sum(click_open)) %>%

  # Open rate
  mutate(avg_click_or_open =
         total_clicked / total_possible)
```

This is answering: "What is the person's open rate **for each specific category?**" This is the exact same as the chunk above, except we are also grouping by category.

```
clickthru_rates <-
  emails %>% group_by(ID_DEMO, broad_cat) %>%

  # Total possible, total clicked
  dplyr::summarize(total_possible = n(),
                   total_clicked = sum(click_open)) %>%

  # Open rate
  mutate(avg_click_or_open =
         total_clicked / total_possible)
```

## Adjusting Scores

At this point, we have two email dataframes:

- Average open rate **per person**
  - ID_DEMO
  - avg_click_or_open
- Average open rate **per person per category**
  - ID_DEMO
  - avg_click_or_open
  - broad_cat

In order for us to compare the two, we need for both to only have 1 row per person. We do this by taking the "broad_cat" column and converting each category its own column.

This operation gives us a dataframe where we have only 1 row per-person.

`spread(broad_cat, click_or_open)` means that the categories from `broad_cat` will become the new column, `click_or_open` will become the values.

```
clickthru_rates_spread <-
  clickthru_rates %>%
  spread(broad_cat, click_or_open) %>%
  rename(email_Learning = Learning,
         email_Legislature = Legislature,
         email_Social = Social,
         email_Sports = Sports) %>% replace(is.na(.), 0)
```

Now that we have 1 row per-person and columns for each category, we can join in the person's average open rate. As you can see in the `mutate` statement, we adjust the person's per-category scores relative to that person's average open rate.

```
clickthru_rates_spread <-
  clickthru_rates_spread %>%
  inner_join(email_avg, by = 'ID_DEMO') %>%

  # Adjusting
  mutate(email_Learning =
             email_Learning - avg_click_or_open,
         email_Legislature =
             email_Legislature - avg_click_or_open,
         email_Social = email_Social - avg_click_or_open,
         email_Sports = email_Sports - avg_click_or_open)
```

At this point we now have a dataframe called `clickthru_rates_spread`. It contains:

- ID_DEMO
- email_Learning
- email_Legislature
- email_Social
- email_Sports

## Events

The procedure for events is very similar to the one we used for emails. For emails, we used the `summarize` function to find a person's open rate. For events, we just use the `count` function which provides us with how many events the person went to.

This part counts the number of events the person went to, and then uses this number to calculate how many of each type we expect them to go to.

```
# How many event types are there?
poss_cats <- n_distinct(events$broad_cat)


# How many has the person gone to (in total)?
event_person_avg <-
  events %>%
  group_by(ID_DEMO) %>%

  # Per person: how many total? How many expected per cat?
  summarise(counts = n(),
            expected_per_cat = counts / poss_cats)
```

This section calculates how many events a person **actually** went to from **each category**.

```r
events_adj <-
  events %>%
  count(ID_DEMO, broad_cat) %>%
  rename(total_type_person = n) %>%
  select(ID_DEMO, broad_cat, total_type_person)
```

## Adjusting Scores

We have to perform the same type of operation that we did for events. Currently, all of the event types are in 1 single column. For our purposes, we need each category to be it's own column.

```r
# Getting each cat as a column
events_spread <-
  events_adj %>%
  spread(broad_cat, total_type_person) %>%
  rename(event_Learning = Learning,
         event_Legislature = Legislature,
         event_other_networking = Networking,
         event_Social = Social,
         event_Sports = Sports) %>%
  select(-Other) %>% replace(is.na(.), 0)
```

Now we know how many events of each type that we expect someone to attend, and we also have the result of how many they actually did attend.

We can use these to calculate a person's adjusted scores.

```r
# Getting the "adjusted" counts
events_spread <-
  events_spread %>%
  inner_join(event_person_avg, by ='ID_DEMO') %>%

  mutate(event_Learning = event_Learning - expected_per_cat,
         event_Legis = event_Legislature - expected_per_cat,
         event_other_networking =
            event_other_networking  - expected_per_cat,
         event_Social = event_Social - expected_per_cat,
         event_Sports = event_Sports - expected_per_cat)
```

At this point we now have a dataframe called `events_spread`. It contains:

- ID_DEMO
- event_Learning
- event_Legis

- event_other_networking
- event_Social
- event_Sports

## Connect

`clickthru_rates_spread` has 1 row per-person with email categories as columns

`events_spread` has 1 row per-person with event categories as columns

We can now join these two together.

```
per_person_with_id <-
  clickthru_rates_spread %>%
  full_join(events_spread, by = 'ID_DEMO') %>%
  inner_join(indy, by = 'ID_DEMO')
```

We want to minimize the number of columns that will go into our clustering algorithm, so we create a "super" column for each category. The 5 columns that will go into our clusters are:

- Learning
- Legislature
- Social
- Sports
- Networking

```
# Creating improved columns for our clustering
cluster_df <-
  per_person_with_id %>%
  replace(is.na(.), 0) %>% as_tibble() %>%


  mutate(learning = event_Learning + email_Learning,
         legis = event_Legis + email_Legislature,
         social = email_Social + event_Social,
         sports = email_Sports + event_Sports,
         networking = event_other_networking +
            event_other_networking) %>%



write.csv(cluster_df, '../cluster_prep.csv', row.names = F)
```