HappyDB_report

Shilin Li (sl4261)

9/11/2018

```
library(tidyverse) # general utility & workflow functions
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──
```

```
## ✔ ggplot2 2.2.1     ✔ purrr   0.2.4
## ✔ tibble  1.4.2     ✔ dplyr   0.7.4
## ✔ tidyr   0.8.0     ✔ stringr 1.2.0
## ✔ readr   1.1.1     ✔ forcats 0.3.0
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(tidytext) # tidy implimentation of NLP methods
library(tidyr) # tidy implimentation of NLP methods
library(topicmodels) # for LDA topic modelling
library(tm) # general text mining functions, making document term matrixes
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(wordcloud) # text mining graph
```

```
## Loading required package: RColorBrewer
```

```
library(scales) # jitter plot in ggplot
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(data.table) # Manipulate lists and data frames to data table
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

Read The Data

This project mainly focuses on the exploration data analysis on the HappyDB dataset. The goal of the report is finding interesting pattern among people's happy moments. In the report, I will focus on the topic difference in people regarding gender, marital status, parenthood, and age groups. Similarities and differences in topics will be discussed.

```
hm_data <- read_csv("~/Documents/GitHub/Fall2018-Proj1-lishilin63/data/cleaned_hm.csv")
```

```
## Parsed with column specification:
## cols(
##   hmid = col_integer(),
##   wid = col_integer(),
##   reflection_period = col_character(),
##   original_hm = col_character(),
##   cleaned_hm = col_character(),
##   modified = col_character(),
##   num_sentence = col_integer(),
##   ground_truth_category = col_character(),
##   predicted_category = col_character()
## )
```

```
demographic_data <- read_csv("~/Documents/GitHub/Fall2018-Proj1-lishilin63/data/demographic.csv")
```

```
## Parsed with column specification:
## cols(
##   wid = col_integer(),
##   age = col_character(),
##   country = col_character(),
##   gender = col_character(),
##   marital = col_character(),
##   parenthood = col_character()
## )
```

Preliminary Cleanning hm_data Text

I used a common stopword list in Kaggle (https://www.kaggle.com/rtatman/stopword-lists-for-19-languages#englishST.txt (https://www.kaggle.com/rtatman/stopword-lists-for-19-languages#englishST.txt)) to exclude these words appearing in the hm_data. Also, since our data are mostly related to happy moments, I also add words “happy”, “happier”, “happiest” etc. in the stopwords list. The cleaning process also references the pre-processing document shared in class.

The new hm_data has two new columns in which the “text” column is my analysis focus on. It has seperated word or phrase cleaned up with stopwords, and thus more relevant to the happy moments topics.

I merged hm_data with the demographic_data. In this way, the whole file contains all the information about happy moments regarding age, gender, marital status and parenthood. The inner join process is by wid that each moment is paried by the recorders. Then, we could focus on our interest groups (gender, marital, parenthhood, age)

```
# We clean the text by converting all the letters to the lower case, and removing punctuation, numbers, empty words and extra white space.
corpus <- VCorpus(VectorSource(hm_data$cleaned_hm))%>%
  tm_map(content_transformer(tolower))%>%
  tm_map(removePunctuation)%>%
  tm_map(removeNumbers)%>%
  tm_map(removeWords, character(0))%>%
  tm_map(stripWhitespace)

# Stemming words and converting tm object to tidy object.
# Stemming reduces a word to its word *stem*. We stem the words here and then convert the "tm" object to a "tidy" object for much faster processing.
stemmed <- tm_map(corpus, stemDocument) %>%
  tidy() %>%
  select(text)

# Creating tidy format of the dictionary to be used for completing stems
# We also need a dictionary to look up the words corresponding to the stems.
dict <- tidy(corpus) %>%
  select(text) %>%
  unnest_tokens(dictionary, text)

# Removing stopwords that don't hold any significant information for our data set
# We remove stopwords provided by the "tidytext" package and also add custom stopwords in context of our data.
data("stop_words")

word <- c("happy","ago","yesterday","lot","today","months","month",
                "happier","happiest","last","week","past","day","time","positive","experience","temple"
,"favorite","extremely","tonight","function","movement","promotion")

stop_words <- stop_words %>%
  bind_rows(mutate(tibble(word), lexicon = "updated"))

# Combining stems and dictionary into the same tibble
# Here we combine the stems and the dictionary into the same "tidy" object.
completed <- stemmed %>%
  mutate(id = row_number()) %>%
  unnest_tokens(stems, text) %>%
  bind_cols(dict) %>%
  anti_join(stop_words, by = c("dictionary" = "word"))

# Stem completion
# Lastly, we complete the stems by picking the corresponding word with the highest frequency.
completed <- completed %>%
  group_by(stems) %>%
  count(dictionary) %>%
  mutate(word = dictionary[which.max(n)]) %>%
  ungroup() %>%
  select(stems, word) %>%
  distinct() %>%
  right_join(completed) %>%
  select(-stems)
```

```
## Joining, by = "stems"
```

```
# Pasting stem completed individual words into their respective happy moments
# We want our processed words to resemble the structure of the original happy moments. So we paste the w
ords together to form happy moments.
completed <- completed %>%
  group_by(id) %>%
  summarise(text = str_c(word, collapse = " ")) %>%
  ungroup()

# Keeping a track of the happy moments with their own ID
hm_data <- hm_data %>%
  mutate(id = row_number()) %>%
  inner_join(completed)
```

```
## Joining, by = "id"
```

```
head(hm_data)
```

```
## # A tibble: 6 x 11
##    hmid   wid reflection_period original_hm      cleaned_hm       modified
##   <int> <int> <chr>             <chr>            <chr>            <chr>
## 1 27673  2053 24h               I went on a suc… I went on a suc… True
## 2 27674     2 24h               I was happy whe… I was happy whe… True
## 3 27675  1936 24h               I went to the g… I went to the g… True
## 4 27676   206 24h               We had a seriou… We had a seriou… True
## 5 27677  6227 24h               "I went with gr… "I went with gr… True
## 6 27678    45 24h               I meditated las… I meditated las… True
## # ... with 5 more variables: num_sentence <int>,
## #   ground_truth_category <chr>, predicted_category <chr>, id <int>,
## #   text <chr>
```

```
hm_data <- hm_data %>%
  inner_join(demographic_data)
```

```
## Joining, by = "wid"
```

```
head(hm_data)
```

```
## # A tibble: 6 x 16
##    hmid   wid reflection_period original_hm      cleaned_hm       modified
##   <int> <int> <chr>             <chr>            <chr>            <chr>
## 1 27673  2053 24h               I went on a suc… I went on a suc… True
## 2 27674     2 24h               I was happy whe… I was happy whe… True
## 3 27675  1936 24h               I went to the g… I went to the g… True
## 4 27676   206 24h               We had a seriou… We had a seriou… True
## 5 27677  6227 24h               "I went with gr… "I went with gr… True
## 6 27678    45 24h               I meditated las… I meditated las… True
## # ... with 10 more variables: num_sentence <int>,
## #   ground_truth_category <chr>, predicted_category <chr>, id <int>,
## #   text <chr>, age <chr>, country <chr>, gender <chr>, marital <chr>,
## #   parenthood <chr>
```

Processed Text Look Up

The first analysis I did focused on the processed text which shows more relevant words and phrase towards topics. I used VCorpus function to vectorize the processed text and ranked words with frequency.

From both the word rank and the word palette, “friend”,“family”,“life”,“love”,“birthday” etc. appears the most frequently in HappyDB and words like “shopping”, “dog”, “wife”, “school” and so on also show up a lot of times. Overall, we could see that a majority of people’s happy moments are comming from their friends and family. This indeed implies the general trend of happiness.

Without subgrouping the hm_data, we could see from the pie chart that a great number of people’s happy moments come from achievements and affections. For example,

Achievements:

- I made a new recipe for peasant bread, and it came out spectacular!
- I was shorting Gold and made $200 from the trade.
- Managed to get the final trophy in a game I was playing.

Affections:

- I went on a successful date with someone I felt sympathy and connection with.
- I was happy when my son got 90% marks in his examination
- I went with grandchildren to butterfly display at Crohn Conservatory

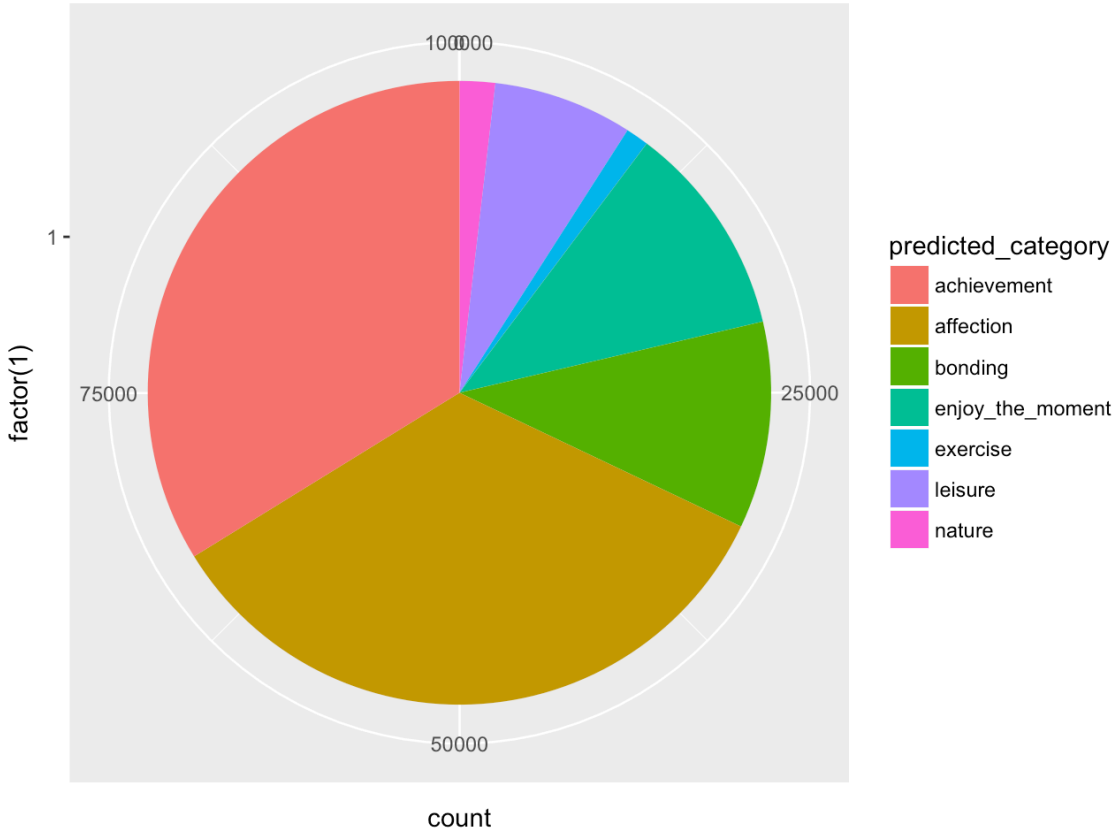
Other happy moments also come from bonding, enjoy_the_moment and leisure which have relative smaller proportions. Nevertheless, the exercise and nature appears the lowest frequency among all the happy moments in the HappyDB.

```
hm_text <- VCorpus(VectorSource(hm_data$text))
tdm <- TermDocumentMatrix(hm_text)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)
```

```
##            word  freq
## friend   friend 10892
## family   family  4692
## watched watched  4385
## home       home  4211
## played   played  4058
## feel       feel  3946
## finally finally  3922
## found     found  3720
## son         son  3633
## enjoyed enjoyed  3564
```

```
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors= brewer.pal(8, "Dark2"))
```


```
ggplot(hm_data, aes(x=factor(1), fill=predicted_category)) +
  geom_bar(width = 1) +
  coord_polar("y")
```



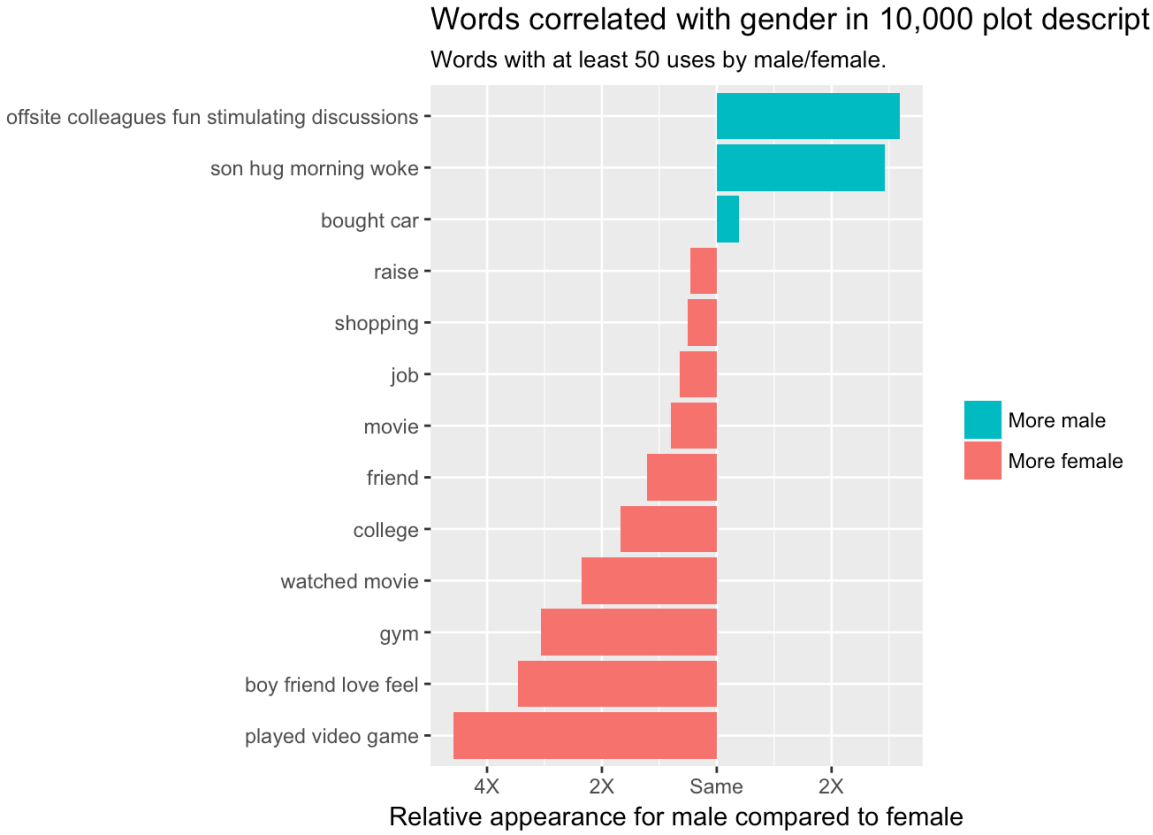
Informative Descriptive Words By Gender

Reference: http://varianceexplained.org/r/tidytext-gender-plots/ (http://varianceexplained.org/r/tidytext-gender-plots/)

Since male and female shares a lot of commmon happy moments, I graphed a most skewed words plot to see the difference between males and females. The happy words I chose came from at least 50 times of using by both genders. We could observe that there is a stong skewness in words like “movie”,“gym”, “boyfriend” and “video games” are more from female whereas “son”, “colleagues” and “discussions” are more likely from male.

```
hm_gender <- hm_data[hm_data$gender != "o" ,]

m_f_counts <- hm_gender %>%
  count(gender, text) %>%
  spread(gender, n, fill = 0) %>%
  mutate(total = m + f,
         m = (m + 1) / sum(m + 1),
         f = (f + 1) / sum(f + 1),
         log_ratio = log2(f / m),
         abs_ratio = abs(log_ratio)) %>%
  arrange(desc(log_ratio))
```

```
m_f_counts %>%
  filter(!text %in% c("himself", "herself"),
         total>= 50) %>%
  group_by(direction = ifelse(log_ratio > 0, 'More male', "More female")) %>%
  top_n(15, abs_ratio) %>%
  ungroup() %>%
  mutate(text = reorder(text, log_ratio)) %>%
  ggplot(aes(text, log_ratio, fill = direction)) +
  geom_col() +
  coord_flip() +
  labs(x = "",
       y = 'Relative appearance for male compared to female',
       fill = "",
       title = "Words correlated with gender in 10,000 plot descriptions",
       subtitle = "Words with at least 50 uses by male/female.") +
  scale_y_continuous(labels = c("4X", "2X", "Same", "2X"),
                     breaks = seq(-2, 1)) +
  guides(fill = guide_legend(reverse = TRUE))
```



Stories About Parenthood

Reference: https://www.tidytextmining.com/twitter.html (https://www.tidytextmining.com/twitter.html)

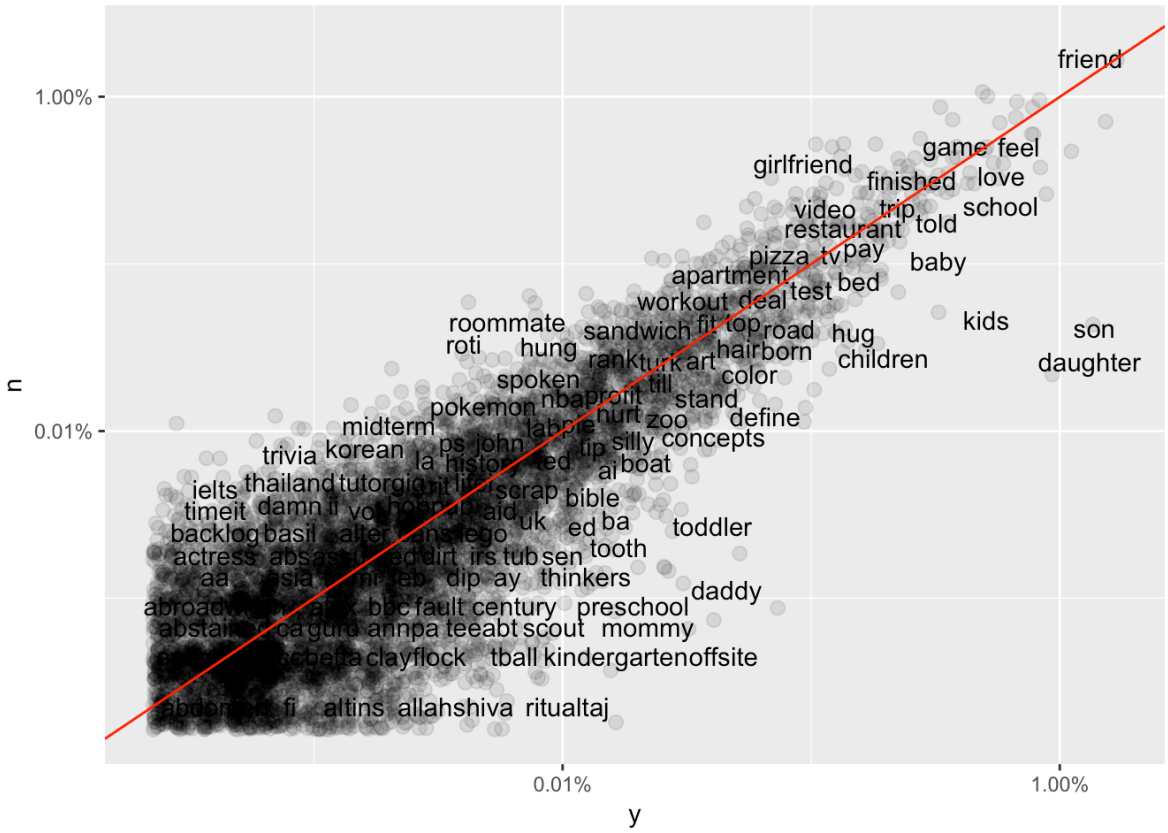
The two distictive groups have great difference in their happy moments. People who do not have child talk in diverse words such as girlfriend, roommate, game and deadlift, while people with children talk a lot about grandchildren, son, daughter, daddy etc. It quite makes sense that people with children have a lot of happy moments with their children and people without have other things that make them happy.

```
# Create a bag of words using the text data
bag_of_words <-  hm_data %>%
  unnest_tokens(word, text)

# Count frequency of each word
word_count <- bag_of_words %>%
  count(word, sort = TRUE)
```

```
# Calculate word frequency for each parenthood category
frequency <- bag_of_words %>%
  group_by(parenthood) %>%
  count(word, sort = TRUE) %>%
  left_join(bag_of_words %>%
              group_by(parenthood) %>%
              summarise(total = n())) %>%
  mutate(freq = n/total)
```

```
## Joining, by = "parenthood"
```

```
frequency <- frequency %>%
  select(parenthood, word, freq) %>%
  spread(parenthood, freq) %>%
  arrange(y, n)

# Scatter plot of word frequency from "yes" and "no" group
ggplot(frequency,aes(y, n)) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.25, height = 0.25) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  geom_abline(color = "red")
```



Interesting Words By Marital

Supervised topic modeling with TF-IDF

Reference: https://www.kaggle.com/rtatman/nlp-in-r-topic-modelling (https://www.kaggle.com/rtatman/nlp-in-r-topic-modelling)

In HappyDB with labeled data, I used supervised topic modeling with TF-IDF (term frequency-inverse document frequency). The general idea behind how TF-IDF works is this:

- Words that are very common in a specific document are probably important to the topic of that document
- Words that are very common in all documents probably aren't important to the topics of any of them

So a term will recieve a high weight if it's common in a specific document and also uncommon across all documents. The reason I used TF-IDF is because this method works quite well when a variable has many categories. In the analysis of marrital status (5 groups) and age (I subgrouped it with 6 levels), TF-IDF yields a good result.

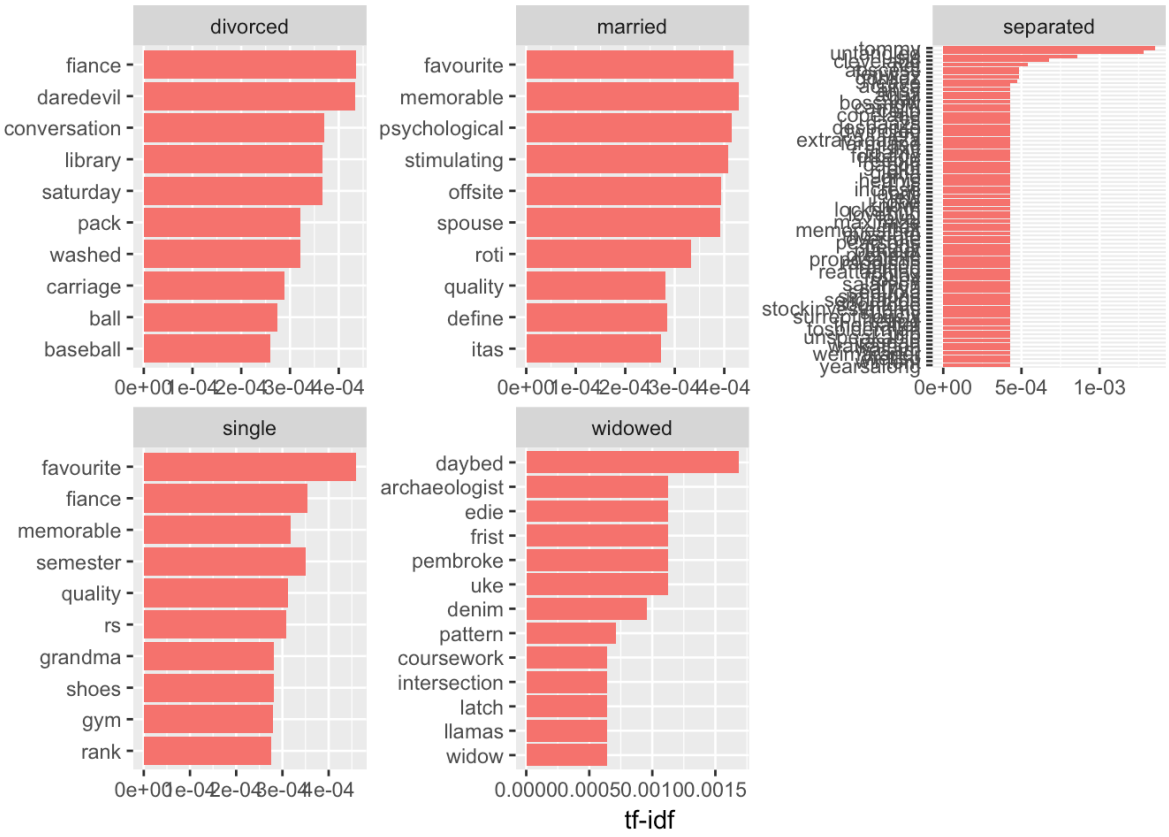
The marital status contains five categories (divorced, married, separated, single and windowed). We could observe distinctive top words people are using. Divorced group has a lot of happy moments coming from finance, conversations and weekend. Married couples have many physchological happy moments with their spouse and memorable events. Separated group tends to have happy moments in shopping (Tommy), resturant and stock trading. Single group also focus on finance. Also, they mentioned a lot about semester. grandma, which may indicate a great proportion would be students. Widowed group does not have a focused topic in their top words and it might becasue of an relevant small sample size.

```
# function that takes in a dataframe and the name of the columns
# with the document texts and the topic labels. If plot is set to
# false it will return the tf-idf output rather than a plot.
top_terms_by_topic_tfidf <- function(text_df, text_column, group_column, plot = T){
    # name for the column we're going to unnest_tokens_ to
    # (you only need to worry about enquo stuff if you're
    # writing a function using using tidyverse packages)
    group_column <- enquo(group_column)
    text_column <- enquo(text_column)

    # get the count of each word in each review
    words <- text_df %>%
      unnest_tokens(word, !!text_column) %>%
      count(!!group_column, word) %>%
      ungroup()

    # get the number of words per text
    total_words <- words %>%
      group_by(!!group_column) %>%
      summarize(total = sum(n))

    # combine the two dataframes we just made
    words <- left_join(words, total_words)

    # get the tf_idf & order the words by degree of relevence
    tf_idf <- words %>%
      bind_tf_idf(word, !!group_column, n) %>%
      select(-total) %>%
      arrange(desc(tf_idf)) %>%
      mutate(word = factor(word, levels = rev(unique(word))))

    if(plot == T){
        # convert "group" into a quote of a name
        # (this is due to funkiness with calling ggplot2
        # in functions)
        group_name <- quo_name(group_column)

        # plot the 10 most informative terms per topic
        tf_idf %>%
          group_by(!!group_column) %>%
          top_n(10) %>%
          ungroup %>%
          ggplot(aes(word, tf_idf, fill = as.factor(group_name))) +
          geom_col(show.legend = FALSE) +
          labs(x = NULL, y = "tf-idf") +
          facet_wrap(reformulate(group_name), scales = "free") +
          coord_flip()
    }else{
        # return the entire tf_idf dataframe
        return(tf_idf)
    }
}
```

```
top_terms_by_topic_tfidf(text_df = hm_data,
                         text_column = text,
                         group_column = marital,
                         plot = T)
```

```
## Joining, by = "marital"
```

```
## Selecting by tf_idf
```



DIY Observations By Age Group

I used a common division criteria, 10 years old, in separating ages into 6 groups. The 0~19 group seems a majority of students in which their happy moments contain busboy and midterm. People greater than 20 years old share the common happy moments with the most outstanding “wife” word. Age groups in 50~59 and 60+ tend to mention granddaughter, grandchildren and grandkits a lot. This indeed makes sense that senior people’s happy moments come a lot from their grandchildren.

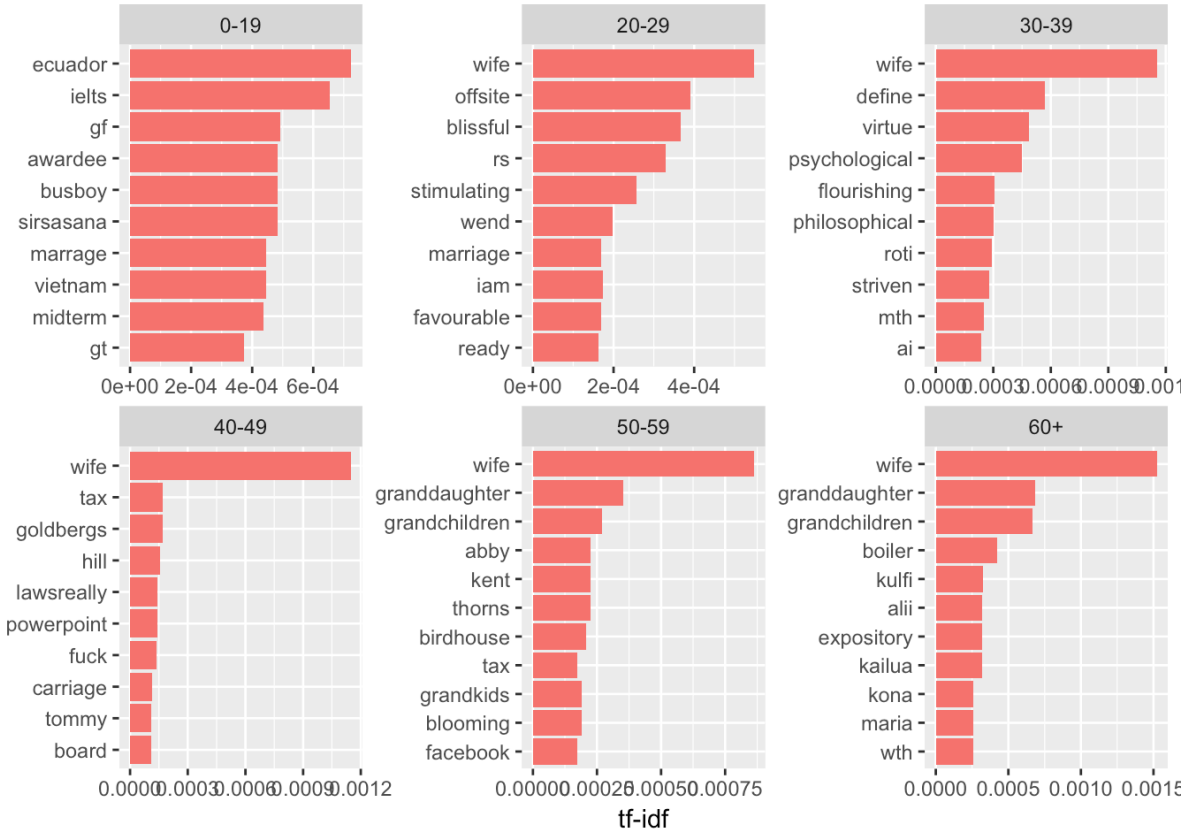
In the bar chart, we could see that the happyDB survey collects a majority of people whose ages are between 20 to 40 years old. The happy moments are not evenly distributed among our age groups. Therefore, some of the top-words extracted (eg. 60+ age group) might not represent accurately the happy topics people in that age groups are talking.

```
# Divide ages into subgroups

hm_data$age <- as.numeric(hm_data$age)

agebreaks <- c(0,20,30,40,50,60,500)
agelabels <- c("0-19","20-29","30-39","40-49","50-59","60+")

setDT(hm_data)[ , agegroups := cut(age,
                                   breaks = agebreaks,
                                   right = FALSE,
                                   labels = agelabels)]
```

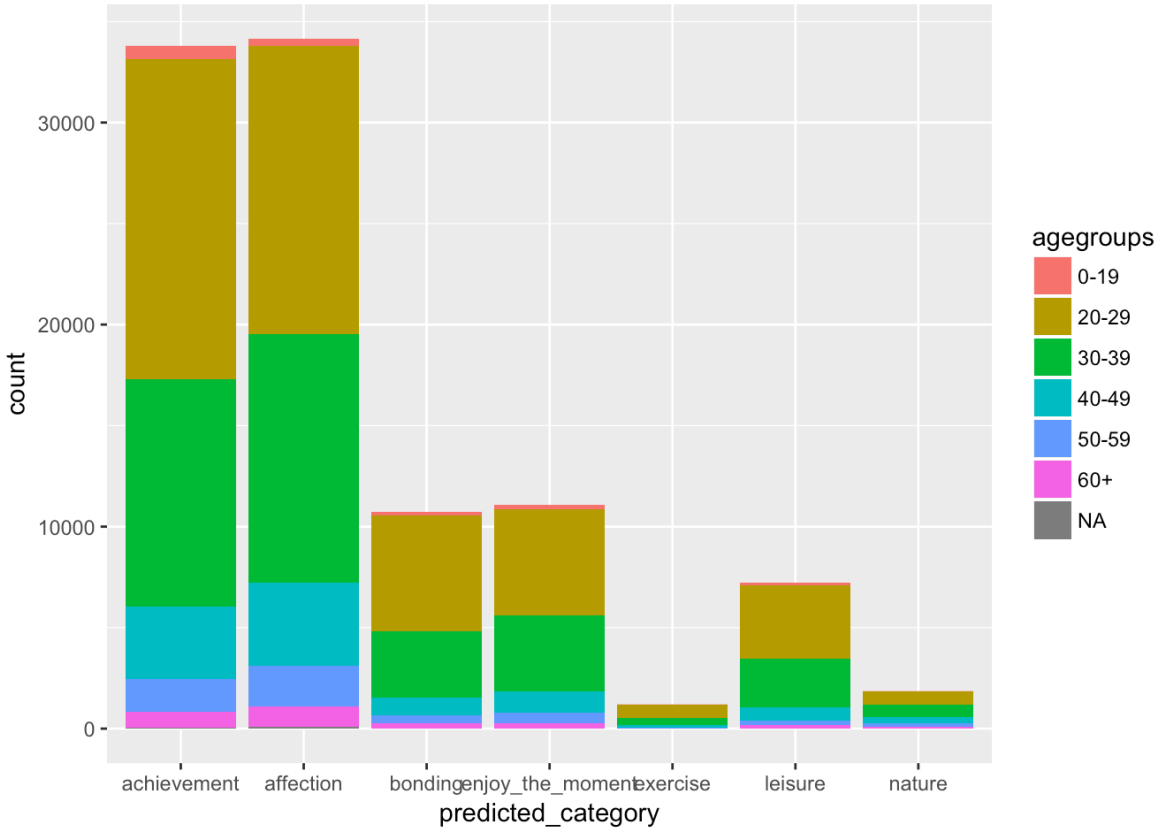
```
top_terms_by_topic_tfidf(text_df = hm_data,
                         text_column = text,
                         group_column = agegroups,
                         plot = T)
```

```
## Joining, by = "agegroups"
```

```
## Selecting by tf_idf
```




```
ggplot(data=hm_data, aes(x=predicted_category,fill = agegroups)) +
    geom_bar(stat="count")
```



Conclusion

The EDA of the happyDB just scratches the surface of people's happy moments. A lot of interesting patterns are found in this dataset. Since the survey sample size is around 10000 in which some of the groups such as 0~20 age group, 60+ age group, divorced group, separated group and widowed group do not contain a large enough size, the inference in these small sample-size groups need to be further varified and researched. More detailed information and interesting facets are also encouraged to explore.

Lastly, I also find some moments are worth reading especially about "Boss" and "someone". I listed several of them below.

Boss:

- My current boss told me how much she is going to miss me and my work etiquette after I graduate and can no longer work for her at the end of this quarter.
- My boss give me bonus as Motorcycle its a very happiest moment for me.
- My boss randomly complimented me on a good job I've been doing lately.

Someone:

- On the train this morning, someone told me I had a nice smile.
- Someone who I admire in my school program told me that they thought I was very intelligent, warm, and funny.
- Someone at the store gave me fifty cents to cover the difference in money I didn't have from forgetting my wallet at home.