Project1: An R Notebook Data Story on Horror Stories

Jiongjiong Li

01/30/2018

Section 0 Preparations, install and load the needed packages

```
packages.used <- c("ggplot2", "dplyr", "tibble", "tidyr",  "stringr", "tidytext", "topicmodels", "wordcloud", "
ggridges","SnowballC")

# check packages that need to be installed.
packages.needed <- setdiff(packages.used, intersect(installed.packages()[,1], packages.used))

# install additional packages
if(length(packages.needed) > 0) {
  install.packages(packages.needed, dependencies = TRUE, repos = 'http://cran.us.r-project.org')
}

library(ggplot2)
library(dplyr)
```

```
##
## 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(tibble)
library(tidyr)
library(stringr)
library(tidytext)
library(topicmodels)
library(wordcloud)
```

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

```
library(ggridges)
library(SnowballC)


source("../lib/multiplot.R")
```

Section 1: Data Preparation

Data load and overview

```
spooky <- read.csv('../data/spooky.csv', as.is = TRUE)
```

Data overview

```
head(spooky)
```

```
##        id
## 1 id26305
## 2 id17569
## 3 id11008
## 4 id27763
## 5 id12958
## 6 id22965
##



                         text
## 1

                     This process, however, afforded me no means of ascertaining the dimensions of my dungeon; a
s I might make its circuit, and return to the point whence I set out, without being aware of the fact; so perfe
ctly uniform seemed the wall.
## 2


                                                                     It never once occurred to me that the fumb
ling might be a mere mistake.
## 3

                                                   In his left hand was a gold snuff box, from which, as he cap
ered down the hill, cutting all manner of fantastic steps, he took snuff incessantly with an air of the greates
t possible self satisfaction.
## 4

                                             How lovely is spring As we looked from Windsor Terrace on the sixt
een fertile counties spread beneath, speckled by happy cottages and wealthier towns, all looked as in former ye
ars, heart cheering and fair.
## 5

                                                                             Finding nothing else, not even gol
d, the Superintendent abandoned his attempts; but a perplexed look occasionally steals over his countenance as
he sits thinking at his desk.
## 6 A youth passed in solitude, my best years spent under your gentle and feminine fosterage, has so refined t
he groundwork of my character that I cannot overcome an intense distaste to the usual brutality exercised on bo
ard ship: I have never believed it to be necessary, and when I heard of a mariner equally noted for his kindlin
ess of heart and the respect and obedience paid to him by his crew, I felt myself peculiarly fortunate in being
 able to secure his services.
##   author
## 1    EAP
## 2    HPL
## 3    EAP
## 4    MWS
## 5    HPL
## 6    MWS
```

```
summary(spooky)
```

```
##       id                text              author
##  Length:19579       Length:19579       Length:19579
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```
glimpse(spooky)
```

```
## Observations: 19,579
## Variables: 3
## $ id     <chr> "id26305", "id17569", "id11008", "id27763", "id12958", ...
## $ text   <chr> "This process, however, afforded me no means of ascerta...
## $ author <chr> "EAP", "HPL", "EAP", "MWS", "HPL", "MWS", "EAP", "EAP",...
```

change the data type of author

```
sum(is.na(spooky))
```

```
## [1] 0
```

```
spooky$author <- as.factor(spooky$author)
```

This notebook was prepared with the following environmental settings.

```
print(R.version)
```

```
##                _
## platform       x86_64-w64-mingw32
## arch           x86_64
## os             mingw32
## system         x86_64, mingw32
## status
## major          3
## minor          4.3
## year           2017
## month          11
## day            30
## svn rev        73796
## language       R
## version.string R version 3.4.3 (2017-11-30)
## nickname       Kite-Eating Tree
```

Section 2: Data Cleaning

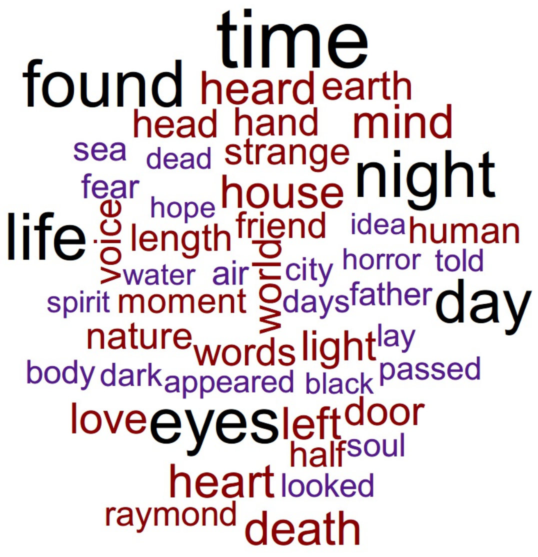
We use the unnest_token to drop the punctuation and transform the words to lower cases, remove the stop words from the data to focus on the really important words. Besides, we will have the data related to certain author.

```
spooky_wrd<- unnest_tokens(spooky,word,text)
spooky_wrd_withstop<-spooky_wrd # data with stop words
spooky_wrd<-anti_join(spooky_wrd,stop_words,by="word") # without stop words
EAP<-filter(spooky_wrd_withstop,author=="EAP")
MWS<-filter(spooky_wrd_withstop,author=="MWS")
HPL<-filter(spooky_wrd_withstop,author=="HPL")
EAP_nstop<-filter(spooky_wrd,author=="EAP")
MWS_nstop<-filter(spooky_wrd,author=="MWS")
HPL_nstop<-filter(spooky_wrd,author=="HPL")
```

Word Colud Generation

We generate one word cloud graph of the whole spooky file.

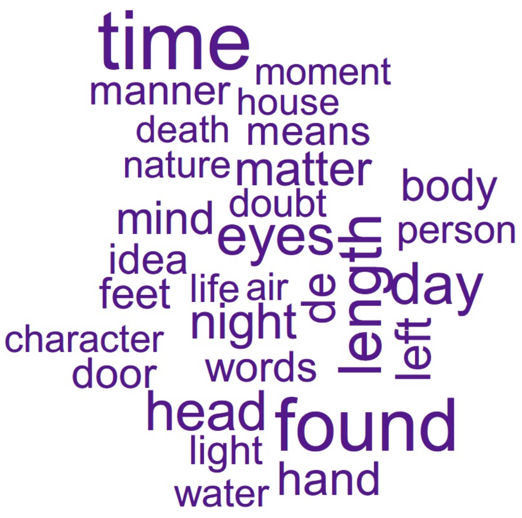
```
#png('../figs/whole_cloud.png')
 spooky_wrd %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 50, color = c("purple4", "red4", "black")))
```



```
#dev.off()
```

We draw the worldcloud for EAP

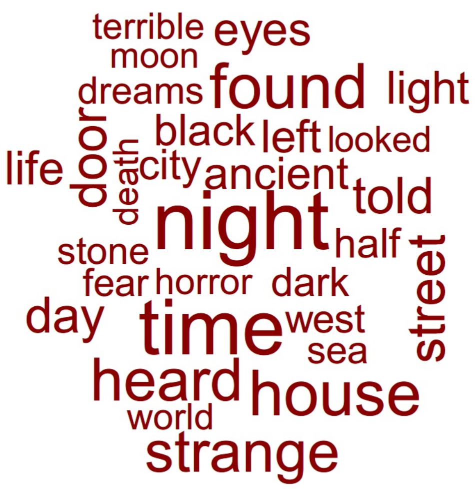
```
words_EAP <- count(group_by(EAP_nstop, word))$word
freqs_EAP <- count(group_by(EAP_nstop, word))$n
#png('../figs/EAP_cloud.png')
wordcloud(words_EAP, freqs_EAP, max.words = 30, color = c("purple4"))
```



```
#dev.off()
```

We draw the worldcloud for HPL

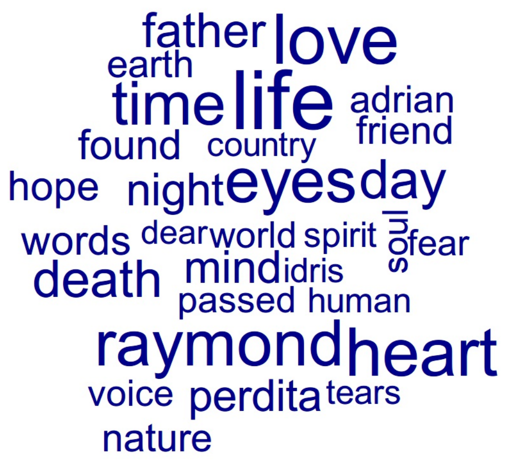
```
words_HPL <- count(group_by(HPL_nstop, word))$word
freqs_HPL <- count(group_by(HPL_nstop, word))$n
#png('../figs/HPL_cloud.png')
wordcloud(words_HPL, freqs_HPL, max.words = 30, color = c("red4"))
```



```
#dev.off()
```

We draw the worldcloud for MWS

```
words_MWS <- count(group_by(MWS_nstop, word))$word
freqs_MWS <- count(group_by(MWS_nstop, word))$n
#png('../figs/MWS_cloud.png')
wordcloud(words_MWS, freqs_MWS, max.words = 30, color = c("blue4"))
```



```
#dev.off()
```

We can see that different authors have different preferences for using words. For Edgar Allan Poe, he most frequently use words like “mind” and “manner”. HP Lovecraft likes to use words “strange” and “horror”. While for Mary Shelley, her most frequent words are “love” and “death”.These frequent words are pretty normal for the horror authors.

Data Visualization

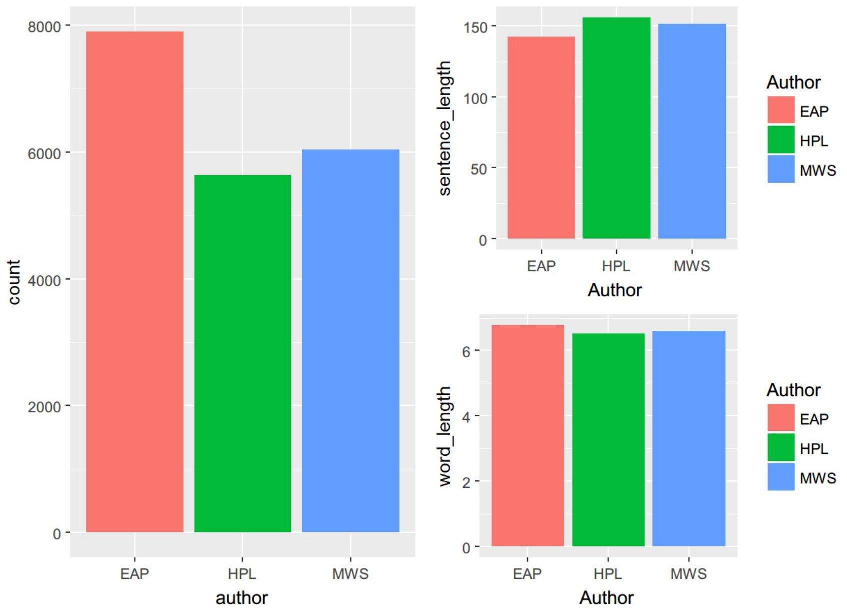
Data Visualization and comparation, We have done some numerical summaries of the data to provide some nice visualizations.

```
p1 <- ggplot(spooky) +
      geom_bar(aes(author, fill = author)) +
      theme(legend.position = "none") # the whole spooky data analysis
spooky$sen_length <- str_length(spooky$text)
spooky$author2<-as.character(spooky$author)
head(spooky$sen_length)
```

```
## [1] 231  71 200 206 174 468
```

```
EAPsen_length<-mean(((filter(spooky,author2=="EAP"))$sen_length))
MWSsen_length<-mean(((filter(spooky,author2=="MWS"))$sen_length))
HPLsen_length<-mean(((filter(spooky,author2=="HPL"))$sen_length))
dt1=as.data.frame(matrix(c("EAP","MWS","HPL"),nrow=3,ncol=1))
colnames(dt1)<-"Author"
dt1$sentence_length<-c(EAPsen_length,MWSsen_length,HPLsen_length)
p2 <- ggplot(dt1,aes(x=Author,y=sentence_length,fill=Author))+
      geom_bar(stat="identity", position=position_dodge())
spooky_wrd$word_length <- str_length(spooky_wrd$word)
EAPwrd_length<-mean(((filter(spooky_wrd,author=="EAP"))$word_length))
MWSwrd_length<-mean(((filter(spooky_wrd,author=="MWS"))$word_length))
HPLwrd_length<-mean(((filter(spooky_wrd,author=="HPL"))$word_length))
dt1$word_length<-c(EAPwrd_length,MWSwrd_length,HPLwrd_length)
p3 <- ggplot(dt1,aes(x=Author,y=word_length,fill=Author))+
      geom_bar(stat="identity", position=position_dodge())
layout <- matrix(c(1, 2, 1, 3), 2, 2, byrow = TRUE)
#png('../figs/author_compare.png')
multiplot(p1, p2, p3, layout = layout)
```

```
## Loading required package: grid
```



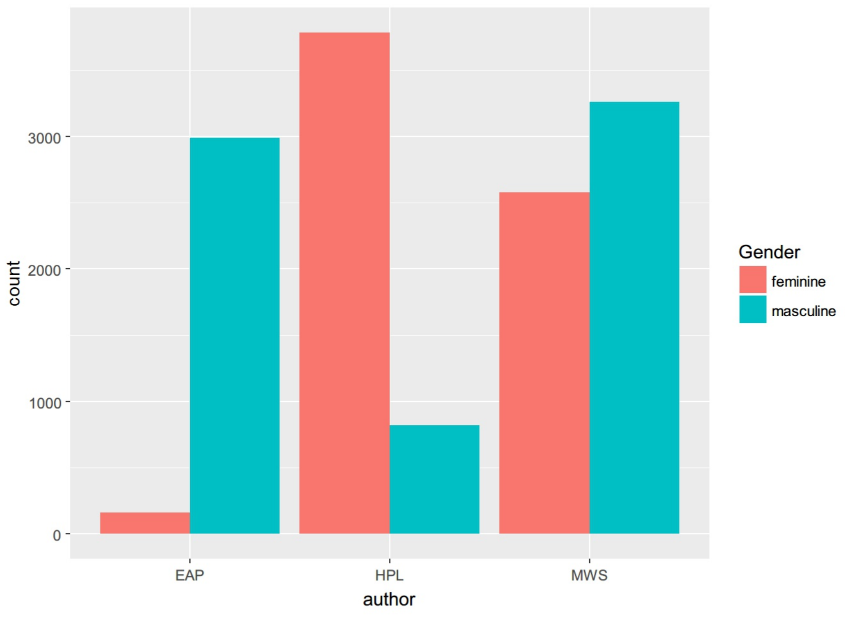
```
#dev.off()
```

So in the spooky data, Edgar Allan Poe has the most sentences and HP Lovecraft has the least. As for the average sentence length, HP Lovecraft writes the longgest sentence while Edgar Allan Poe writes the shortest sentence. When it comes to the average word length, we can see that Edgar Allan Poe uses the longgest word and there is no clear difference in word length between HP Lovecraft and Mary Shelley.

He/She difference

we decide to take a look at the He/She, to explore over the gender differenc. (This requires the stop words not be removed as “he” or “she” are usually taken as the stop words)

```
male<-c("he","him","his")
female<-c("she","her")
heEAP<-dim(filter(EAP,word %in% male))[1]
sheEAP<-dim(filter(EAP,word %in% female))[1]
heHPL<-dim(filter(HPL,word %in% male))[1]
sheHPL<-dim(filter(HPL,word %in% female))[1]
heMWS<-dim(filter(MWS,word %in% male))[1]
sheMWS<-dim(filter(MWS,word %in% female))[1]
dt2=as.data.frame(matrix(c(rep("masculine",3),rep("feminine",3)),nrow=6,ncol=1))
colnames(dt2)<-"Gender"
dt2$author<-c(rep(c("EAP","HPL","MWS"),2))
dt2$count<-c(heEAP,sheEAP,heHPL,sheHPL,heMWS,sheMWS)
#png('../figs/gender_difference.png')
ggplot(dt2, aes(x = author, y = count, fill = Gender))+
   geom_bar(stat="identity", position=position_dodge())
```



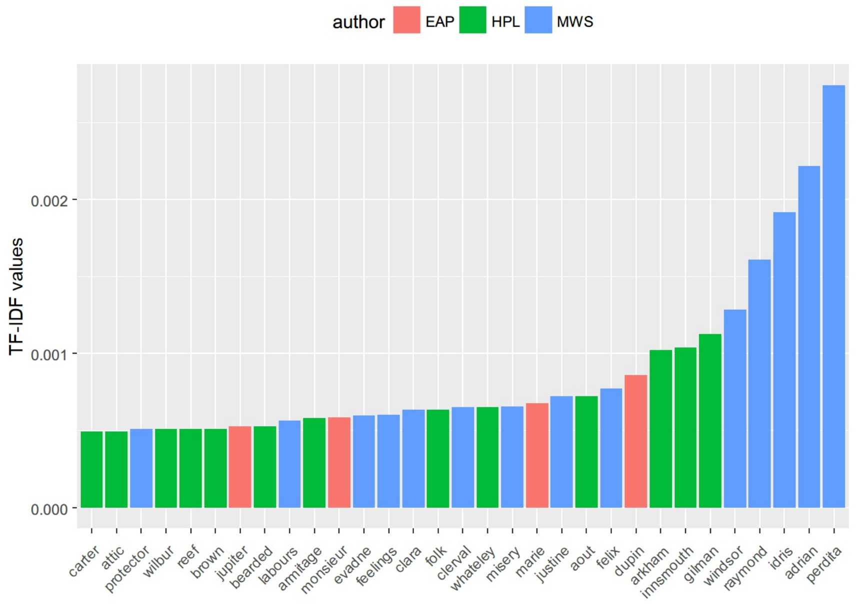
```
#dev.off()
```

Both Edgar Allan Poe and HP Lovecraft have very clear preference in using masculine third-person word or feminine third-person word. Edgar Allan Poe uses the masculine third-person word more often (he,him,his) while HP Lovecraft uses the feminine third-person word (she,her) more often. Mary Shelley uses masculine third-person word and feminine third-person word almost equally (A bit more in masculine).

TF-IDF

We try to find words that are characteristic for a specific author by using tf-idf as a heuristic index to indicate how frequently a certain author uses a word.

```
frequency <- count(spooky_wrd, author, word)
tf_idf    <- bind_tf_idf(frequency, word, author, n)

tf_idf    <- arrange(tf_idf, desc(tf_idf))
tf_idf    <- mutate(tf_idf, word = factor(word, levels = rev(unique(word))))

# Grab the top thirty tf_idf scores in all the words
tf_idf_30 <- top_n(tf_idf, 30, tf_idf)
#png('../figs/tf_idf.png')
p_tf<-ggplot(tf_idf_30) +
  geom_col(aes(word, tf_idf, fill = author)) +
  labs(x = NULL, y = "TF-IDF values") +
  theme(legend.position = "top", axis.text.x  = element_text(angle=45, hjust=1, vjust=0.9))
p_tf
```



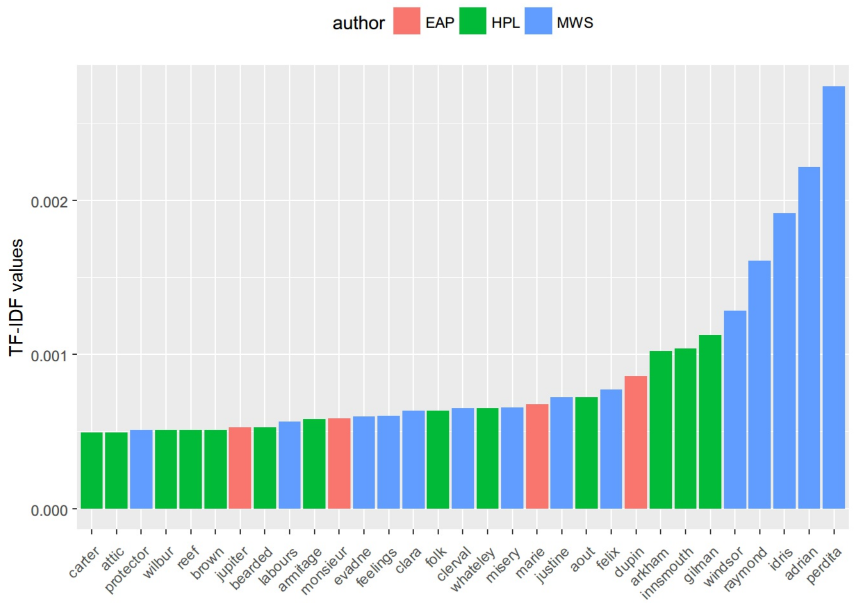
```
#dev.off()
```

The above graph shows the thirty tf_idf scores in all the words and we see that words from Mary Shelley have the highest TF-IDF values. Besides, most of the words in the top 30 are actually names. Names work quite well in identifying authors as there is little possibilty that different authors will use the same name for their characters.

TF-IDF after stemming

We use the “wordstem” function to extract the stems of each of the given words and then compute the TF-IDF values of words again to find if the results changes or not.

```
test<-apply(as.array(spooky_wrd$word),1,wordStem)
spook_wrd_2<-spooky_wrd
spooky_wrd$word<-test
spooky_wrd_stem<-spooky_wrd
spooky_wrd<-spook_wrd_2
frequency <- count(spooky_wrd, author, word)
tf_idf    <- bind_tf_idf(frequency, word, author, n)

tf_idf    <- arrange(tf_idf, desc(tf_idf))
tf_idf    <- mutate(tf_idf, word = factor(word, levels = rev(unique(word))))

# Grab the top thirty tf_idf scores in all the words
tf_idf_30 <- top_n(tf_idf, 30, tf_idf)
#png('../figs/tf_idf_stem.png')
p_ts<-ggplot(tf_idf_30) +
  geom_col(aes(word, tf_idf, fill = author)) +
  labs(x = NULL, y = "TF-IDF values") +
  theme(legend.position = "top", axis.text.x  = element_text(angle=45, hjust=1, vjust=0.9))
p_ts
```



```
#dev.off()
```

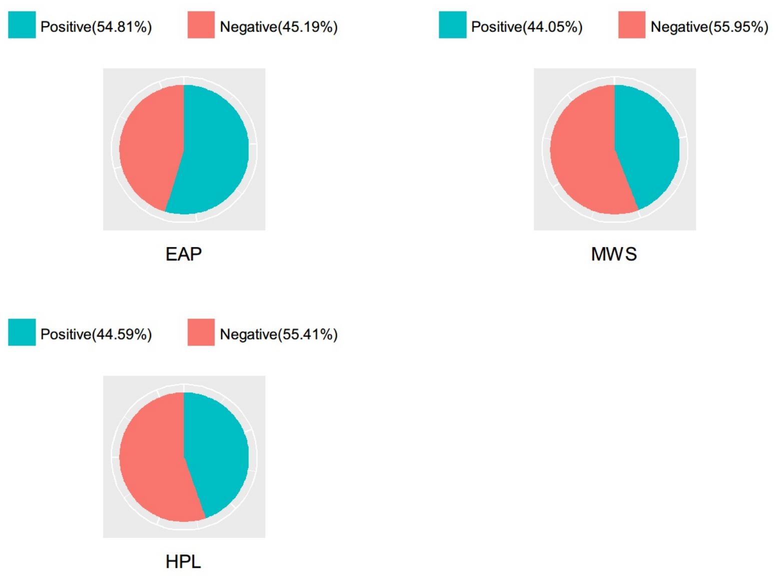
Note that the graph result actually don't change. Maybe it's because the high TF-IDF value words are names. The stems of names are the same as names.

Section3: Sentiment Analysis

Nrc lexicon

In the sentiment analysis part, we want to measure what is the proportion of the sentiment was positive or negative. We first use the nrc lexicons.

```
nrc_filter <- filter(get_sentiments('nrc'), sentiment %in% c("positive","negative"))
sentiments_nrc <- inner_join(spooky_wrd, nrc_filter, by = "word")
EAP_nrc<-filter(sentiments_nrc,author=="EAP")
HPL_nrc<-filter(sentiments_nrc,author=="HPL")
MWS_nrc<-filter(sentiments_nrc,author=="MWS")
EAP_pos<-dim(filter(EAP_nrc,sentiment=="positive"))[1]
EAP_neg<-dim(EAP_nrc)[1]-EAP_pos
HPL_pos<-dim(filter(HPL_nrc,sentiment=="positive"))[1]
HPL_neg<-dim(HPL_nrc)[1]-HPL_pos
MWS_pos<-dim(filter(MWS_nrc,sentiment=="positive"))[1]
MWS_neg<-dim(MWS_nrc)[1]-HPL_pos
#plot pie graph for EAP
dt3 = data.frame(A = c(EAP_pos, EAP_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt3$B)
myLabel = paste(myLabel, "(", round(dt3$A / sum(dt3$A) * 100, 2), "%)        ", sep = "")

p4 = ggplot(dt3, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "EAP", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt3$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#plot pie graph for HPL
dt4 = data.frame(A = c(HPL_pos, HPL_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt4$B)
myLabel = paste(myLabel, "(", round(dt4$A / sum(dt4$A) * 100, 2), "%)        ", sep = "")

p5 = ggplot(dt4, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "HPL", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt4$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#plot pie graph for MWS
dt5 = data.frame(A = c(MWS_pos, MWS_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt5$B)
myLabel = paste(myLabel, "(", round(dt5$A / sum(dt5$A) * 100, 2), "%)        ", sep = "")

p6 = ggplot(dt5, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "MWS", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt5$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#png('../figs/nrc_pos.png')
multiplot(p4,p5,p6,cols=2)
```



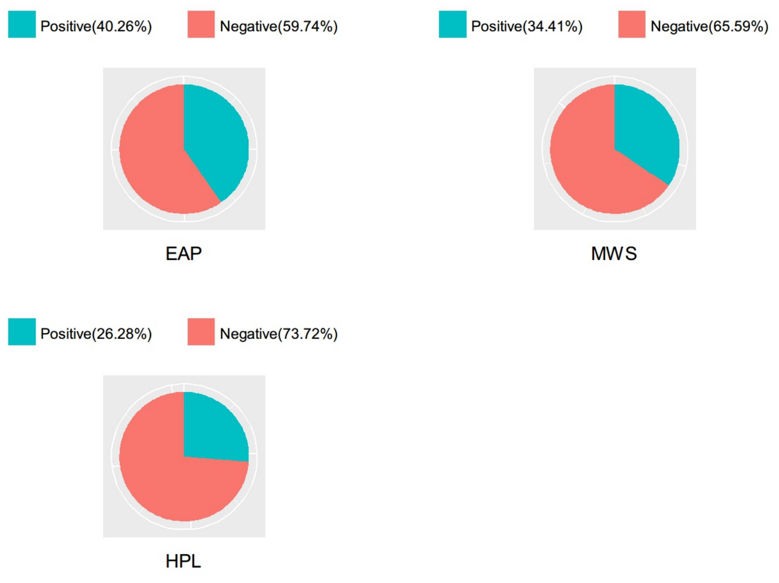
```
#dev.off()
```

Overall we can see that in different authors' text, the negative sentiment accounts more than the positive sentiment. In Edgar Allan Poe's article, he has more positive parts.

Bing lexicon

Now we repeat what we did in last part again, but this time we use the "bing" lexicons, we use the setiment analysis package of bing

```r
sentiments_bing<-inner_join(spooky_wrd, get_sentiments('bing'), by = "word")
EAP_bing<-filter(sentiments_bing,author=="EAP")
HPL_bing<-filter(sentiments_bing,author=="HPL")
MWS_bing<-filter(sentiments_bing,author=="MWS")
EAP_pos<-dim(filter(EAP_bing,sentiment=="positive"))[1]
EAP_neg<-dim(EAP_bing)[1]-EAP_pos
HPL_pos<-dim(filter(HPL_bing,sentiment=="positive"))[1]
HPL_neg<-dim(HPL_bing)[1]-HPL_pos
MWS_pos<-dim(filter(MWS_bing,sentiment=="positive"))[1]
MWS_neg<-dim(MWS_bing)[1]-HPL_pos
#plot pie graph for EAP
dt3 = data.frame(A = c(EAP_pos, EAP_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt3$B)
myLabel = paste(myLabel, "(", round(dt3$A / sum(dt3$A) * 100, 2), "%)        ", sep = "")

p4 = ggplot(dt3, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "EAP", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt3$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#plot pie graph for HPL
dt4 = data.frame(A = c(HPL_pos, HPL_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt4$B)
myLabel = paste(myLabel, "(", round(dt4$A / sum(dt4$A) * 100, 2), "%)        ", sep = "")

p5 = ggplot(dt4, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "HPL", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt4$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#plot pie graph for MWS
dt5 = data.frame(A = c(MWS_pos, MWS_neg), B = c('Positive','Negative'))

myLabel = as.vector(dt5$B)
myLabel = paste(myLabel, "(", round(dt5$A / sum(dt5$A) * 100, 2), "%)        ", sep = "")

p6 = ggplot(dt5, aes(x = "", y = A, fill = B)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(x = "", y = "MWS", title = "") +
  theme(axis.ticks = element_blank()) +
  theme(legend.title = element_blank(), legend.position = "top") +
  scale_fill_discrete(breaks = dt5$B, labels = myLabel)+
  theme(axis.text.x = element_blank())
#png('../figs/bing_pos.png')
multiplot(p4,p5,p6,cols=2)
```



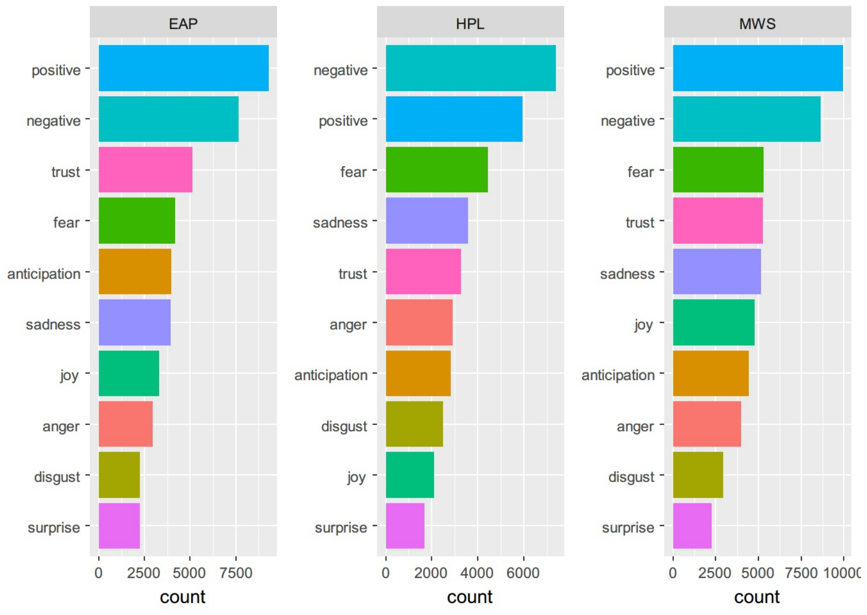
```r
#dev.off()
```

Using the “bing” lexicon, the result actually doesn’t change. In text,the negative sentiment accounts more than the positive sentiment. Besides, we can see that now the neagtive sentiment is accounting more part than before. Now in Edgar Allan Poe’s article, he has more negative parts.

Emotion rank

In this part we will try to find what are the top few emotions represented by each author and the difference

```
sentiments <- inner_join(spooky_wrd, get_sentiments('nrc'), by = "word")
dt6<-as.data.frame(count(sentiments, author, sentiment))
EAP_emotion<-filter(dt6,author=="EAP")
HPL_emotion<-filter(dt6,author=="HPL")
MWS_emotion<-filter(dt6,author=="MWS")
p7<-ggplot(EAP_emotion,aes(x=reorder(sentiment,n),y=n,fill=sentiment))+
   geom_bar(stat="identity", position=position_dodge())+
   coord_flip()+
   labs(x = NULL, y = "count")+
   facet_wrap(~ author)+
   theme(legend.position = "none")
p8<-ggplot(HPL_emotion,aes(x=reorder(sentiment,n),y=n,fill=sentiment))+
   geom_bar(stat="identity", position=position_dodge())+
   coord_flip()+
   labs(x = NULL, y = "count")+
   facet_wrap(~ author)+
   theme(legend.position = "none")
p9<-ggplot(MWS_emotion,aes(x=reorder(sentiment,n),y=n,fill=sentiment))+
   geom_bar(stat="identity", position=position_dodge())+
   coord_flip()+
   labs(x = NULL, y = "count")+
   facet_wrap(~ author)+
   theme(legend.position = "none")
#png('../figs/emotion_difference.png')
multiplot(p7,p8,p9,cols=3)
```



```
#dev.off()
```

We can see that the top emotions are pretty different among these authors. For Edgar Allan Poe, his top emotions are “trust, fear and anticipation”. HP Lovecraft’s top emotions are “fear, sadness and trust”. Mary Shelley’s top emotions are “fear, trust and sadness”.

Section 4: Topic Modelling

In the topic modelling part, we try to visualize author topics and we choose 6 topics.

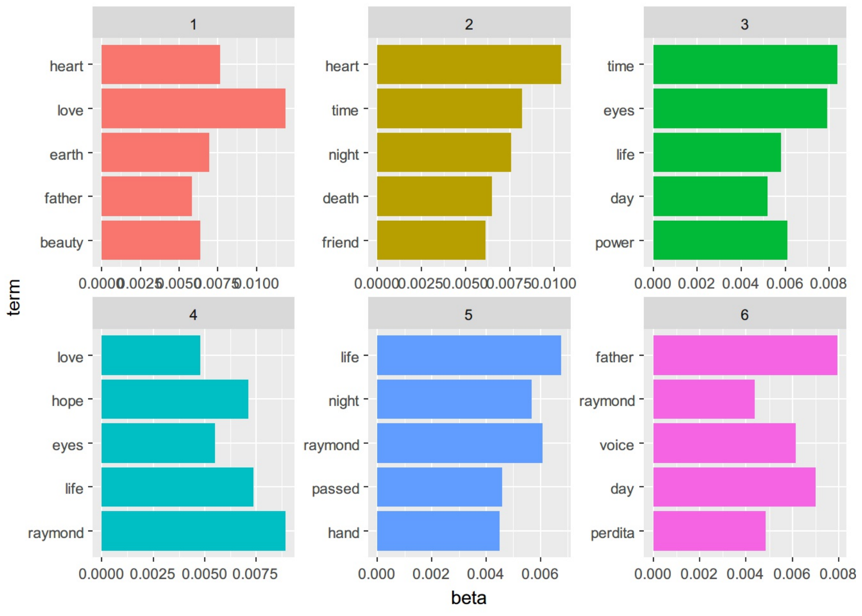
```
#EAP
sent_wrd_freqs_EAP <- count(filter(spooky_wrd,author=="EAP"), id, word)
spooky_wrd_tm_EAP <- cast_dtm(sent_wrd_freqs_EAP, id, word, n)
spooky_wrd_lda_EAP  <- LDA(spooky_wrd_tm_EAP, k = 6, control = list(seed = 1234))
spooky_wrd_topics_EAP <- tidy(spooky_wrd_lda_EAP, matrix = "beta")
spooky_wrd_topics_5_EAP <- ungroup(top_n(group_by(spooky_wrd_topics_EAP, topic), 5, beta))
spooky_wrd_topics_5_EAP <- arrange(spooky_wrd_topics_5_EAP, topic, -beta)
spooky_wrd_topics_5_EAP <- mutate(spooky_wrd_topics_5_EAP, term = reorder(term, beta))
png('../figs/EAP_topic.png')
ggplot(spooky_wrd_topics_5_EAP) +
  geom_col(aes(term, beta, fill = factor(topic)), show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 3) +
  coord_flip()
dev.off()
```

```
## png
##   2
```

```
#HPL
sent_wrd_freqs_HPL <- count(filter(spooky_wrd,author=="HPL"), id, word)
spooky_wrd_tm_HPL <- cast_dtm(sent_wrd_freqs_HPL, id, word, n)
spooky_wrd_lda_HPL  <- LDA(spooky_wrd_tm_HPL, k = 6, control = list(seed = 1234))
spooky_wrd_topics_HPL <- tidy(spooky_wrd_lda_HPL, matrix = "beta")
spooky_wrd_topics_5_HPL <- ungroup(top_n(group_by(spooky_wrd_topics_HPL, topic), 5, beta))
spooky_wrd_topics_5_HPL <- arrange(spooky_wrd_topics_5_HPL, topic, -beta)
spooky_wrd_topics_5_HPL <- mutate(spooky_wrd_topics_5_HPL, term = reorder(term, beta))
png('../figs/HPL_topic.png')
ggplot(spooky_wrd_topics_5_HPL) +
  geom_col(aes(term, beta, fill = factor(topic)), show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 3) +
  coord_flip()
dev.off()
```

```
## png
##   2
```

```
#MWS
sent_wrd_freqs_MWS <- count(filter(spooky_wrd,author=="MWS"), id, word)
spooky_wrd_tm_MWS <- cast_dtm(sent_wrd_freqs_MWS, id, word, n)
spooky_wrd_lda_MWS  <- LDA(spooky_wrd_tm_MWS, k = 6, control = list(seed = 1234))
spooky_wrd_topics_MWS <- tidy(spooky_wrd_lda_MWS, matrix = "beta")
spooky_wrd_topics_5_MWS <- ungroup(top_n(group_by(spooky_wrd_topics_MWS, topic), 5, beta))
spooky_wrd_topics_5_MWS <- arrange(spooky_wrd_topics_5_MWS, topic, -beta)
spooky_wrd_topics_5_MWS <- mutate(spooky_wrd_topics_5_MWS, term = reorder(term, beta))
#png('../figs/MWS_topic.png')
ggplot(spooky_wrd_topics_5_MWS) +
  geom_col(aes(term, beta, fill = factor(topic)), show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 3) +
  coord_flip()
```



```
#dev.off()
```

In the above, we see that for Edgar Allan Poe, the first topic is characterized by words like “doubt”, “time”, and the third topic includes the word “death”, and the fifth topic the word “individual”. For HP Lovecraft, the first topic is characterized by words like “strange”, “house”, and the third topic includes the word “death”, and the sixth topic the word “time”,“life. For Mary Shelley, she first topic is characterized by words like”heart“,”love“, and the third topic includes the word”time" and “life”, and the fifth topic the word “hand”. Note that the words “eyes”, “time” and “life” appear in many topics.