Registration No. 2201538

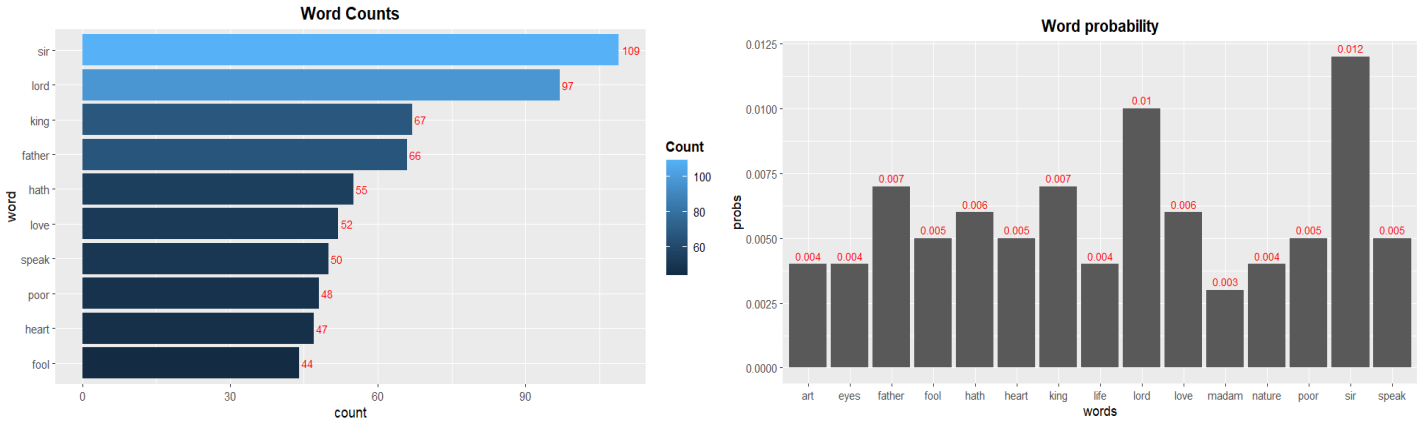
MA331 Assignment Autumn 2022(Course Work)

Text analytics of the "King Lear" by the William Shakespeare.

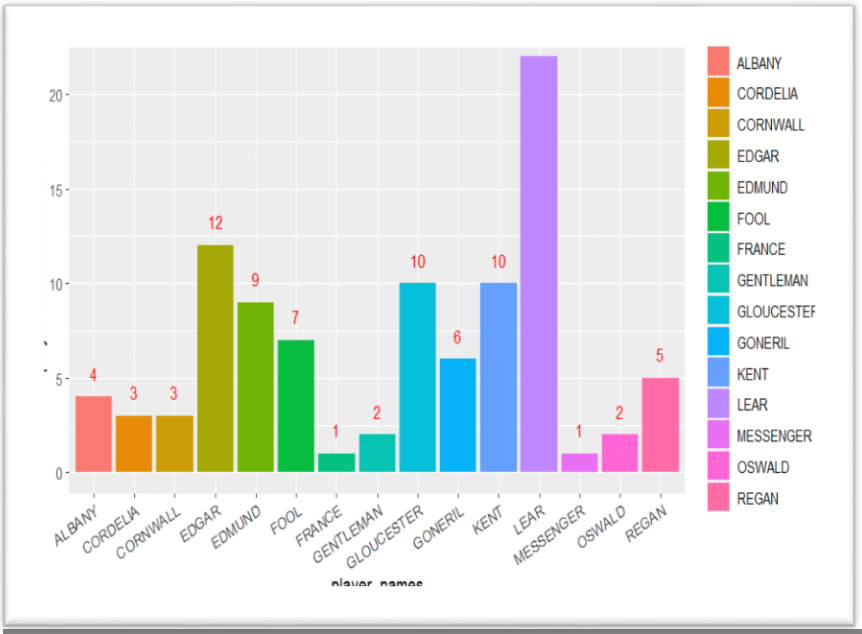
In this project I am demonstrating about the text and sentimental analysis of "King Lear" which is one of the popular play of the William Shakespeare. In this play, there are several characters and their dialogues are present. Based on those dialogues I am doing this text and sentiments analysis.

1] Horizontal bar plot to show the top 10 most spoken words

Once the data from the csv file has been read into the data frame. The text field has been subdivided into unigram tokens. After further examination, it is discovered that it contains numerous stop words such as verbs, pronouns, and prepositions that will be useless for text analytics. The top 10 most frequently occurring words after removing stop words are represented in the bar plot below.



2] Vertical bar plot to show the top 15 player's involvement in the play

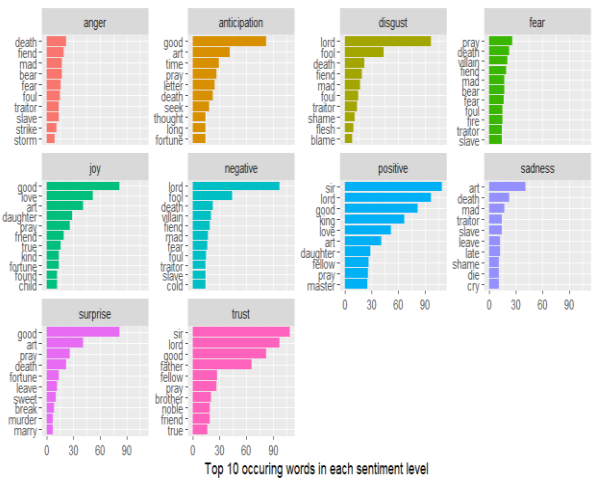


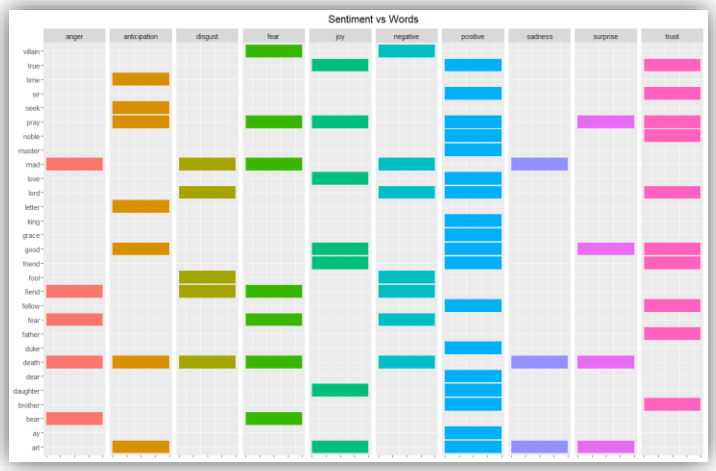
- Vertical bar plot shows that the percentage involvement of each player in the play so that we can easily analysis the main characters of play and their percentage involvement in play
- From the graph we can say that most of the involved player are king Lear, Edgar, Kent and Gloucester.

3] Top 10 occurring words in each NRC sentiment level and Using facet grid, represented the words belong to more than one sentiment by using NRC sentiments:

A group of the graphs shows the most frequent used words in a play with their sentimental classification with NRC sentiments But there are some repeated words are present in two groups like ‘true’ which is present in both positive and trust sentiment so for detailed analysis of that type of word I used the facet grid representation.

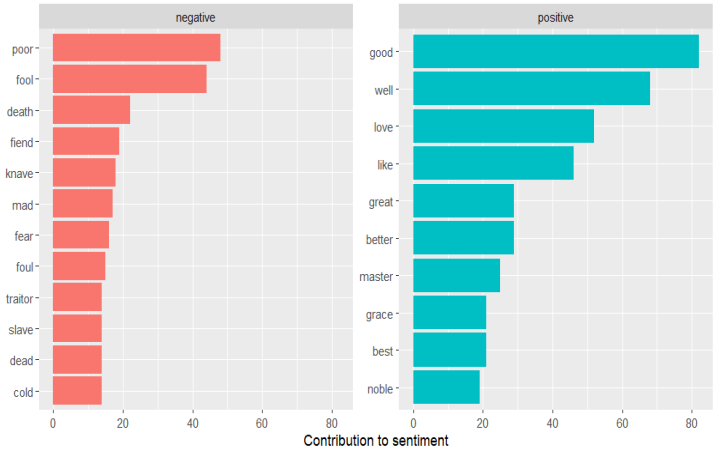
The 2nd graph illustration below depicts the word and its sentiment. When I was analyzing the tokens and its sentiment. I've noticed that each word can represent multiple sentiments. For example, let’s take word ‘true’ it is under sentiments joy, positive and trust.

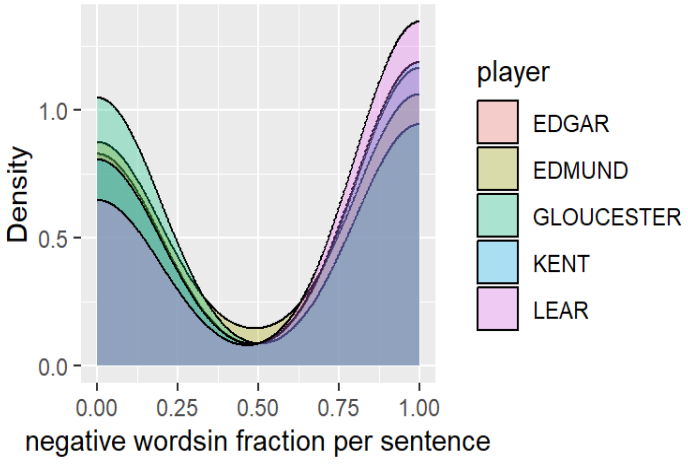




5]Words contribution to top two sentiments by Bing sentiment analysis.

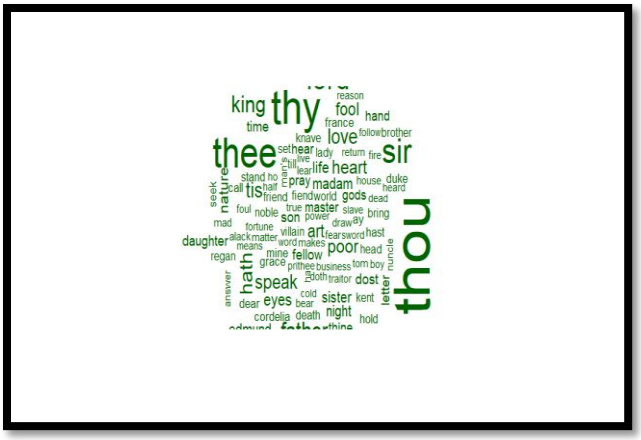
We can determine how much each word contributed to each sentiment by using ‘Bing’ here with arguments for both the word and the sentiment.2nd graph shows that the density index of negative word per sentence throughout the play

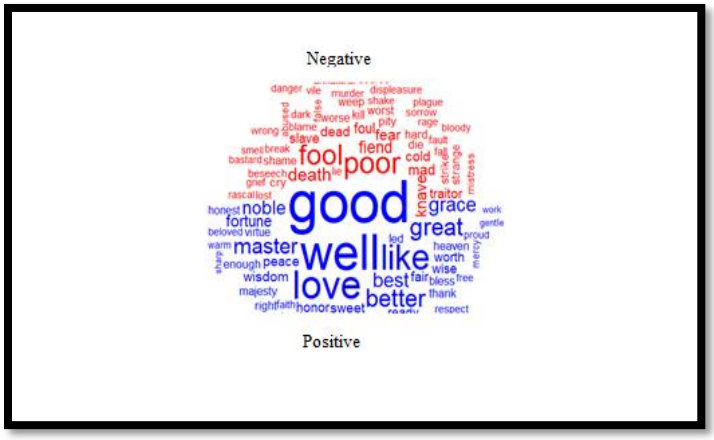




6] Word cloud to show the most appearance to least appearance in the text.

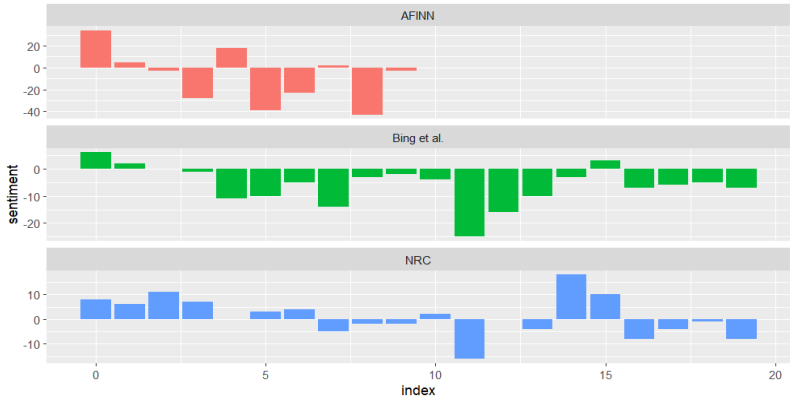
Word cloud is one of my favorites to represent the words based on the counts. Larger the count larger the size of text in figure.





7] Comparing `the three – lexicons sentiment together:`

Although the outcomes of the three distinct lexicons for measuring sentiment of player 'Lear' are varied in absolute terms, their relative trajectories across the play are comparable. The narrative has similar sentimental peaks and valleys at roughly the same locations, although the absolute values fluctuate greatly.



Conclusion:

- King Lear, who speaks for most of the play, is followed by his family members Edger, Kent, and Gloucester, who collectively talk for 70% of the lines.
- Using NRC, analysis of the top player revealed the top sentiment, fear, trust, and positive and negative sentiment.
- The first few words of the conversation were mostly negative, with substantial contributions from King Lear, Kent, and Edgar.
- However, from around 1000 to the end of the conversation, there are both positive and negative sides, so it maintains a close to neutral conversation with a -0.046 score.
- Lear, Kent, and Edgar are at the highest positions, allowing them to use most of the words available to them to express their unpleasant feelings. Conversely, characters in lower positions are not permitted to use as many words to express their negative feelings

Appendix

```
# requied packages and library installation
install.packages("tidyr")
install.packages("dplyr")
install.packages("stringr")
install.packages("tidytext")
install.packages("ggplot2")
install.packages("textdata")
install.packages("RColorBrewer")
install.packages("reshape2")
install.packages("wordcloud")
install.packages("wordcloud2")
install.packages("theme")
install.packages("ggthemes")

library(dplyr)
library(tidyr)
library(stringr)
library(tidytext)
library(ggplot2)
library(tidytext)
library(textdata)
library(reshape2)
library(wordcloud)
library(wordcloud2)
library("ggthemes")
```

```
################################################################################

# a main play  word list read in r
getwd()
setwd('C:/Users/ADMIN/Desktop/MA331')
king_lear <- read.csv("King_Lear_words_and_players_only.csv")
head(king_lear)

# a modern word list list read in r
modern_words <- read.delim("modern_word_count.txt")
str(modern_words)
names(modern_words)
head(modern_words,30)
nrow(modern_words)

# conversion into  tidy formate
token_learking <-king_lear %>%
  unnest_tokens(word, text)
head(token_learking)

# removal of stop (common) words
data("stop_words")
stop_words
real_learking<- token_learking%>% anti_join(stop_words)
head(real_learking)
count_real_learking<-  real_learking %>% count(word, sort = TRUE)
head(count_real_learking)
```

```
# removal of customise stop words
final_stop_words <- bind_rows(tibble(word = c("e'er","lt","de","thou","thy","thee","tis"),
                                     lexicon = c("custom")), stop_words)
head(final_stop_words)

real_learking<- token_learking %>% anti_join(final_stop_words)
head(real_learking)

#############################################################################

# graph of count of words after removal of common words
plot_real_words1 <- real_learking %>%
  count(word, sort = TRUE) %>%
  head(10)%>%
  mutate(word = reorder(word, n)) %>%ggplot(aes(n, word, fill=n)) +
  geom_bar(stat="identity") + xlab("count") + ggtitle("Word Counts")+
  theme(plot.title = element_text(hjust=0.5, face="bold"), legend.title = element_text(face="bold")) +
  labs(fill = "Count")+geom_text(aes(label=n), hjust=-0.2, size=3,color="red")
plot_real_words1

unigram_probs <- real_learking %>%
  count(word, sort = TRUE) %>%
  mutate(p = n / sum(n))%>%
  mutate(probs=round(p,3))%>%
  na.omit(word)%>%
  head(15)

unigram_probs%>%
  ggplot(aes(word,probs))+
```

```
  geom_bar(stat="identity", aes(fill=word))+
  labs(y="player Involvements in %", x="player_names",size=2.8)+
  geom_text(aes(label=probs), hjust=-0, size=3.5, color="blue")+
  scale_x_discrete(guide = guide_axis(angle = 30)) + coord_flip()



king_unigrams <-  king_lear %>%
  unnest_tokens(wordss, text, token = "ngrams", n = 1)
head(king_unigrams)

Player_involvement<- king_unigrams%>%
  count(player, sort = TRUE)%>%
  mutate(Total_word= sum(n))%>%
  mutate(player_involvement_percentage=round((n/Total_word)*100),)%>%
  arrange(desc(player_involvement_percentage))%>%
  head(15)

Player_involvement%>%
  ggplot(aes(player,player_involvement_percentage))+
  geom_bar(stat="identity", aes(fill=player))+
  labs(y="player Involvements in %", x="player_names",size=2.8)+
  geom_text(aes(label=player_involvement_percentage), vjust=-1.2, size=3.5, color="red")+
  scale_x_discrete(guide = guide_axis(angle = 30))

mean_word_length<- king_unigrams%>%
  mutate(word_length= nchar(king_unigrams$wordss))
head(mean_word_length)
means<-aggregate(x = mean_word_length$word_length,                # Specify data column
          by = list(mean_word_length$player),             # Specify group indicator
```

```
          FUN = mean)%>%na.omit(x)%>%
  mutate(mean_length= round(x,2))
means%>%ggplot(aes(Group.1,mean_length))+
  geom_bar(stat="identity")+
  scale_fill_grey()+
  labs(y="mean word length", x="player_names",size=2.8)+
  geom_text(aes(label=mean_length), vjust=-0.6, size=3.5, color="Red")+
  scale_x_discrete(guide = guide_axis(angle = 30))

#################################################################################
get_sentiments("nrc")

nrc_trust <- get_sentiments("nrc") %>%
  filter(sentiment == "trust")
trust_nrc_learking <-token_learking %>%
  filter(player == "KENT") %>%
  inner_join(nrc_trust) %>%
  count(word, sort = TRUE)
hist(trust_nrc_learking$n, col = "green")

learking_wide <- token_learking %>% # Assigning the resultant of the functionality
  anti_join(get_stopwords()) %>% # Removing stop words by doing anti join
  count(player, word) %>% # count by speaker and word
  group_by(word) %>% # grouping by word
  filter(sum(n) > 15) %>% # filtering only sum of word count is greater than 10
  ungroup() %>% # ungrouping it
  pivot_wider(names_from = "player", values_from = "n", values_fill = 0)
learking_wide
```

```
learking_wide_sentiment <- learking_wide %>% # Assigning the resultant of the functionality
  inner_join(get_sentiments("nrc"), by = "word")
head(learking_wide_sentiment)

learking_wide_sentiment %>%  ggplot(aes(x=word, fill=sentiment)) +   # Plotting word and its sentiment
  facet_grid(~sentiment) + # Facet_grid is using for plot
  geom_bar() + #Create a bar for each word per sentiment
  theme(panel.grid.major.x = element_blank(), # Making x axis without any grid
        axis.text.x = element_blank(),  # Making x axis without any label
        legend.position='none', # removing lengend of graph
        plot.title = element_text(hjust=0.5)) + #Place the words on the y-axis
  xlab(NULL) + ylab(NULL) + # removing x and y labels
  ggtitle("Sentiment vs Words") + # adding title of the graph
  coord_flip()

learking_sentiments <-   token_learking %>% # Assigning the resultant of the functionality
  inner_join(get_sentiments("nrc"), by = "word")


learking_sentiments  %>% count(word, sentiment, sort = TRUE) %>% # count by word and sentiment
  group_by(sentiment) %>% # grouping by sentiment
  slice_max(n, n = 10) %>% # slicing the top 10 words
  ungroup() %>% # ungrouping it
  mutate(word = reorder(word, n)) %>%  # reordering word
  ggplot(aes(n, word, fill = sentiment)) +  # Plotting word and its sentiment
  geom_col(show.legend = FALSE) + # removing legend
  facet_wrap(~sentiment, scales = "free_y") + # facetwrap is using to plot
  labs(x = "Top 10 occuring words in each sentiment level", y = NULL)
```

```
get_sentiments("afinn")

afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

AFINN<- get_sentiments("afinn")

learking_play %>%
  # by word and value count number of occurences
  inner_join(AFINN, "word") %>%
  count(player, value, sort=T) %>%
  mutate(contribution = n * value,
         sentiment = ifelse(contribution<=0, "Negative", "Positive")) %>% #another variable
  arrange(desc(abs(contribution))) %>%
  head(20)  %>%
  ggplot(aes(x=reorder(player, contribution), y=contribution, fill=sentiment)) +
  geom_col(aes(fill=sentiment), show.legend = F) +
  labs(x="player", y="Contribution", title="player with biggest contributions in positive/negative sentiments") +
  coord_flip()

p3 <- learking_play %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  filter(!word=="fool")%>%
  filter(player==c("LEAR","EDGAR","KENT","EDGAR","GLOUCESTER","EDMUND"))%>%
```

```
  group_by(player, `linenumber`, sentiment) %>%
  count() %>%
  spread(sentiment, n, fill = 0) %>%
  group_by(player, `linenumber`) %>%
  summarise(neg = sum(negative),
            pos = sum(positive)) %>%
  arrange(`linenumber`) %>%
  mutate(frac_neg = neg/(neg + pos)) %>%
  ggplot(aes(frac_neg, fill = player)) +
  geom_density(bw = .2, alpha = 0.3) +
  theme(legend.position = "right") +
  labs(x = "negative wordsin fraction per sentence",y='Density')
##########################################################################################
get_sentiments("bing") %>%
  count(sentiment)
bing_word_counts <- learking_play %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

bing_word_counts

bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
```

```
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
learking_play %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100,color='darkgreen',rotateRatio=0.5))

learking_play %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE)%>%
  acast(word ~ sentiment, value.var = "n", fill = 0)%>%
  comparison.cloud(colors = c("red", "blue"),
                   max.words = 100, title.size = 2,rotateRatio=0.5)

#################################################################################################
learking_play <- king_lear %>%
  group_by(player) %>%
  mutate(
    linenumber = row_number(),
  ) %>%
  ungroup() %>%
  unnest_tokens(word,text )
learking_play

prejudice <- learking_play %>%
  filter(player == "LEAR")

bing_and_nrc <- bind_rows(
```

```
  prejudice %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
        prejudice %>%
    inner_join(get_sentiments("nrc") %>%
                 filter(sentiment %in% c("positive",
                                         "negative"))
    ) %>%
    mutate(method = "NRC")) %>%
  count(method, index = linenumber %/% 40, sentiment) %>%
  pivot_wider(names_from = sentiment,
              values_from = n,
              values_fill = 0) %>%
  mutate(sentiment = positive - negative)


bind_rows(afinn,
          bing_and_nrc) %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y")


################################################################################
```