Narrative

Hani Warith

11/24/2019

Brief substantive background / goal

As the British Parlimentary election approaches on December 12, 2019 the Labor and Conservative party candidates Boris Johnson and Jeremy Corbyn have pitched the vote as an election of a lifetine in which Britain's future domestically.

As our class has unfolded, three debates have occurred. I wanted to apply the skills we learned in class to analyze the content of the debates. I used sentiment analysis and topic modelling to analyze the debates.

Collecting data

Collecting data was the most challenging part of my project. I used the getcaption package to scrape closed caption data from Youtube as follows:

```
#Scraping captions from Youtube API

library(youtubecaption)

get_caption("https://www.youtube.com/watch?v=9kEB5pqWpJw", savexl= T, openxl= T)
```

This process yielded 3 CSV files (one for each debate) containing all the captions (available in the Scraped Captions Folder).

Cleaning / pre-processing data

I then converted these scripts to plain text. (Plain text transcripts) I then listened to each debate and marked out the portions corresponding to Jeremy Corbyn and to Boris Johnson. Next I created a CSV file which combined metadata about the debates and the individual documents: each response to a question by either question. In total, across the three debates the candidates had 170 responses. (Debate_Data) After creating the CSV I had to clean the data and preprocessed the data for sentiment analysis. I later preprocessed the data for a topic model as well.

```
#Preprocessing for Sentiment analysis
debate_responses_df
library(tm)
debate_responses<-Corpus(VectorSource(debate_responses_df$Response))
debate_responses <- DocumentTermMatrix(debate_responses,
           control = list(stopwords = TRUE,
                          tolower = TRUE,
                          removeNumbers = TRUE,
                          removePunctuation = TRUE))

words <- data.frame(word = colnames(debate_responses_ordered))
head(words)

words <- merge(words,sentiments, all.x=T)
words$core[is.na(words$core)] <- 0

scores <- as.matrix(debate_responses) %*% words$core
```

```
scores

# put it in the original documents data frame
debate_responses_df$sentiments <- scores
```

```
# Pre-processing for Topic Model
temp<-textProcessor(documents = debate_responses_df$Response, metadata = debate_responses_df)

meta<-temp$meta
vocab<-temp$vocab
docs<-temp$documents

out <- prepDocuments(docs, vocab, meta)

docs<-out$documents
vocab<-out$vocab
meta <-out$meta
```

Analysis and visualization

With the data cleaned I was ready to carry out my analysis. The first thing I did was look at which words were used most often in my corpus:

```
#Looking at top words used in debate

sorted_responses <- sort(frequency, decreasing = T)

head(sorted_responses)
```

Given how Salient Brexit and the National Health Service are in British Politics I also wanted to check what words were associated with each:

```
#Looking at words correlated with Brexit and NHS

findAssocs(debate_responses, "brexit", 0.3)
findAssocs(debate_responses, "nhs", 0.3)
```

I find it particularly interesting that “dither”, “delay” and “deadlock” are highly correlated with Brexit while “employed”, “principles”, and “privatization” are correlated with the NHS.

I then made a word cloud to visualize some of the more commonly used words:

```
#Making a WordCloud - Figure 1
set.seed(29)

wordcloud(names(sorted_responses), sorted_responses, max.words=100, colors=brewer.pal(6,"Dark2"))
```

Next I carried out the sentiment analysis and found the Mean Valence score for each candidate:

```
debate_responses_df<-debate_responses_df %>%
  group_by(Speaker) %>%
  filter(Speaker=="Jeremy Corbyn"|Speaker=="Boris Johnson")

debate_responses_df %>%
  summarize(Mean_Sentiment=mean(sentiments))
```

I found that the valence of each candidate’s responses was pretty neutral and both candidates had pretty similar mean scores. I visualized these sentiments for both candidates jointly:

```r
ggplot(debate_responses_df, aes(x = sentiments))+
    geom_histogram(stat = "count", fill="red")+xlab("Sentiments")+ylab("Count")+ggtitle("Sentiments of
```

and for each candidate separately:

```r
#Sentiments for each speaker
ggplot(data = debate_responses_df, aes(x = sentiments)) +
  geom_histogram(fill="blue") +
  facet_wrap( ~ Speaker) + xlab("Sentiments")+ ylab("Count")+ggtitle("Sentiments by Candidate")
```

and then also visualized my results as a boxplot for each candidate:

```r
ggplot(debate_responses_df, aes(x=Event, y=sentiments)) +  geom_boxplot() + facet_wrap( ~ Speaker) + th
```

This visualization suggests that Boris Johnson was most negative in the second debate whil Jeremy Corbyn was most negative in the first.

Having finished my sentiment analysis I ran my topic model:

```r
model <- stm(docs, vocab, 5, data = meta, seed = 15, max.em.its = 15)

labelTopics(model)

labels <- c("Progress", "Poverty", "Health Care", " Referenda and Union", "Economy")
```

The results of the model fit neatly into five labels - unsurprisingly “Health Care” and “Referenda and Union” were two topics given the salience of the NHS and of Brexit. Interestingly, the topic model placed Scotland and Brexit into the same category. During the debates, Boris Johnson repeatedly sought to connect Scotland’s demands for a referendum and Brexit so its interesting that the topic model may have identified this trend.

I then visualized the prevalence of each topic:

```r
plot.STM(model, type="summary", custom.labels = labels, main="")
```

And finally I plotted each topic for both candidates:

```r
# Estimate Covariate Effects
prep <- estimateEffect(1:5 ~ Speaker, model, meta = meta, uncertainty = "Global", documents=docs)
```

```r
Speakers <- c("Corbyn", "Johnson")
plot.estimateEffect(prep, "Speaker", method = "pointestimate", topics = 1, printlegend = TRUE, labeltyp
```

```r
Speakers <- c("Corbyn", "Johnson")
plot.estimateEffect(prep, "Speaker", method = "pointestimate", topics = 2, printlegend = TRUE, labeltyp
```

```r
Speakers <- c("Corbyn", "Johnson")
plot.estimateEffect(prep, "Speaker", method = "pointestimate", topics = 3, printlegend = TRUE, labeltyp
```

```r
Speakers <- c("Corbyn", "Johnson")
plot.estimateEffect(prep, "Speaker", method = "pointestimate", topics = 4, printlegend = TRUE, labeltyp
```

```r
Speakers <- c("Corbyn", "Johnson")
plot.estimateEffect(prep, "Speaker", method = "pointestimate", topics = 5, printlegend = TRUE, labeltyp
```

Future work

Going forward, it will be interesting to see how the Candidates speech evolves in the final days before the election. In a future project, I would be curious to compare sentiment scores in a debate setting, where candidates are more likely to be seeking to persuade undecided voters with sentiment scores in campaign stump speeches, where they are more likely to be addressing their core base. My results indicate that, at least in a debate setting, the candidates kept things fairly civil and struck a fairly neutral tone.

I would also be curious to make use of the QDap package which provides a number of tools for quantitative discourse analysis and has a number of additional functions allowing for other visualizations and analysis tools.