This is a quick post looking at using biterm topic modelling, a new technique for me, on the free text responses to a survey question. I'm interested in whether this type of topic modelling might be a shortcut to analysing free text, quicker than having a human read the answers and code them. The question I'm using to try this out is from the New Zealand Election Study which collected data just after the 2017 New Zealand election. The actual question is "What was the single most important issue for you in the election?".

Human-coded version

This question had been classified/coded by hand by the original researchers. Here are the percentages of voters categorised by the issue they mentioned in that free text:

Single most important issue in the New Zealand election (2017)	Percentage naming most important:
None Named	20.6
Economy	14.0
Housing	11.1
Health	7.8
Inequality	6.7
Tax	6.5
Environment	5.3
Immigration	5.2
Government	4.8
Poverty	4.2
Education	3.6
Welfare	3.2
Party Bias	2.7
Change/continuation	1.4
Infrastructure	0.8
Law and Order	0.8
Moral	0.6
Kindness Common Sense	0.4
Maori	0.4
Rural Regional	0.1
Defence	0.0
Pike River	0.0

When we compare the issues they identified by the party vote of respondents, we see this complex but understandable relationship:

All R code is at the bottom of the post today, as the sequence of creating various tables and charts didn't naturally follow the sequence for including them in this narrative.

The relationship between vote and issue of importance is pretty much as we'd expect. The environment is the top issue for Greens voters (followed by inequality); immigration for New Zealand First; housing for Labour and the Māori Party; and the economy for National voters.

To examine the actual words in the free text that was classified this way against issue, we can knock out the common stopwords and reduce words to their meaningful stems. This gives us this impression:

Again, all seems straightforward. Words with the stem "hous" are strongly associated with the human-coded "Housing" issue (of course), "ineq" with Inequality, "poverti" with Poverty, and so on. So this makes sense in retrospect, but could I have created those topics from scratch with an unsupervised topic model?

Biterm topic modelling

Biterm topic modelling is an approach to gleaning the latent topics discussed in a number of documents. It is explicitly designed to cope efficiently and effectively with short documents, with a sparse document term matrix. It's available via a C++ library by Xiaohui Yan, wrapped in the `BTM` R package maintained by Jan Wijffels. While I use Latent Dirichlet Allocation estimation for topic modelling quite regularly, biterm topic modelling with its emphasis on pairs of words is a newer technique for me. It seems to be very promising.

Is topic modelling appropriate with its many-to-many relationship of documents to topics, or should I be looking for a simpler classification? While the original survey question asked for a single issue of most importance, this is the world of surveys and respondents don't always do what they're asked, and they don't always agree with the researcher on what a single issue is. For example, many people gave responses like "housing and poverty". This can meaningfully be parsed as a single issue, but then how does it relate to others who think just "housing" is the issue, independently of poverty? Then there are responses such as "No party had policy that would remedy major problems, police, education, health, councils". From the respondent's view point this is a single 'issue' (the parties' policies are no good!) but from the point of view of the researchers this is multiple subject-matter issues grouped under a single meta-issue. The original coders coped with this by classifying each response to up to 3 issues (`rimpissue`, `rimpissue2`, `rimpissue3`) and retaining the full original text in the `rmpissuex` column. In the charts earlier in this blog, I used the `rimpissue` column, the first issue identified by the coders.

Like other topic modelling approaches, biterm topic models allow each document to refer to multiple topics and it will generally allow many more than 3 topics per document. We can control this to a degree by tweaking the prior distribution of topics for each document. The prior distribution of the probability of each document discussing each of the k models is a Dirichlet distribution. I built myself a Shiny app to check that the alpha parameter to a Dirichlet distribution works the way I think it does. The `BTM()` function defaults to a value of alpha of 50/k. By changing this to a value of less than one we greatly decrease the parameter space where multiple topics are allowed, without reducing it to zero.

Fitting a biterm topic model with 21 topics (the same number as that chosen by the human coders) and visualising the various links between words that make up biterms and clusters of biterms that make up topics gets me this picture (you may need to zoom in):

Generally there are some sensible results here. For instance, topic 1 hones in on the words "clean", "green", river" and "environment" – very sensible. Topic 15 features "climate change". The human-coded topics did not distinguish between general concern for the environment and climate change as two separate issues, and I would say splitting them out is probably an improvement. Topic 7 clearly relates neatly to health, 2 to foreign land ownership and immigration, and so on. Basically, there are some sensible clusterings of words into topics here that make sense with what we know of political attitudes in New Zealand.

Comparing the BTM-chosen topics numbers 1 to 21 with the human-coded topics made from the same free text, and overlaying the party vote choices most related to each combination of topic, we see this relationship:

For example, the human-coded topic of "environment" is strongly related with BTM topics 1, 15 and 17. Human-coded topic of "housing" comprises two distinct topics 10 (which examination of the earlier chart shows is focused on homelessness) and 13 (focused on affordability).

All this is pretty good and makes me feel that an unsupervised biterm topic model with free text survey data is going to get results than are much better than nothing, and not gibberish. However, looking a bit closer at some edge cases and we see limitations with the method. For example, while most of topic 15 is about "climate change", some of the respondents mentioned "change" in the context of "we need a change of government". These references are both coded as topic 15. A human reader could tell these are unrelated (or coincidentally related) topics, but not the topic model.

Human coding of a few thousand short survey responses is not a big investment compared to the overall cost of running a survey. I think I would trust the human coding over the biterm topic model in this case if forced to choose just one – but it is nice to have both and get some extra nuance (such as distinguishing between affordability and homelessness as two separate housing issues; or climate change and clean and green as two separate environment issues). If I had 100,000 survey responses to code I think I would get a human to classify a few thousand of them and use that to train a machine learning model, hoping to get good enough results to avoid manual coding of the rest. But I would run an unsupervised topic model as well – maybe even *before* I got the human to do hand-coding – for a second opinion.

Here's the R code for today.

```
library(tidyverse)
library(haven)
library(tidytext)
library(SnowballC)
library(BTM)
library(textplot)
library(ggraph)
library(concaveman)
library(scales)
library(glue)
library(nzelect)

nzes <- read_spss("NZES2017SPSS/NZES2017Release14-07-19.sav")
the_caption <- "Data from the New Zealand Election Study, analysed at
http://freerangestats.info"

# Table of most important topics:
nzes %>%
  mutate(issue = as.character(as_factor(rimpissue)),
         issue = gsub("^[0-9]*\\. ", "", issue)) %>%
  group_by(issue) %>%
  summarise(est = sum(rwt)) %>%
  ungroup() %>%
  mutate(prop = round(est / sum(est) * 100, 1)) %>%
  select(issue, prop) %>%
  arrange(desc(prop)) %>%
  rename(`Percentage naming most important:` = prop,
         `Single most important issue in the New Zealand election
(2017)` = issue)

# Full responses and human-chosen topics (issues), dropping uneeded
variables
full_resp <- nzes %>%
  mutate(pv = fct_lump(as_factor(rpartyvote), 6)) %>%
  mutate(pv = gsub("^[0-9]\\. ", "", pv)) %>%
  mutate(issue = as.character(as_factor(rimpissue)),
         issue = gsub("^[0-9]*\\. ", "", issue)) %>%
  mutate(rmpissuex = str_to_sentence(rmpissuex),
         rmpissuex = gsub("nz", "NZ", rmpissuex, ignore.case = TRUE))
%>%
  select(rjacknum, issue, rmpissuex, rwt, pv) %>%
  filter(issue != "None Named")

# Word stems
stems <- full_resp %>%
  unnest_tokens("word", "rmpissuex") %>%
  anti_join(stop_words, by = "word") %>%
  mutate(word_stem = wordStem(word))

grouped_stems <- stems %>%
  group_by(issue, word_stem) %>%
  summarise(sample_mentions = length(unique(rjacknum)),
            weighted_mentions = sum(rwt)) %>%
  mutate(prop_nz = weighted_mentions / sum(nzes$rwt))  %>%
  arrange(desc(weighted_mentions)) %>%
  ungroup()

#--------------------Biterm Topic Modelling---------------
k <- length(unique(stems$issue))
stems_for_btm <- select(stems, rjacknum, word_stem)

# alpha deliberately made small to make it less likely to have
# multiple topics per document:
set.seed(123)
btm_mod <- BTM(data = stems_for_btm,
               k    = k,
               alpha = 0.8)

topics_btm <- predict(btm_mod, newdata = stems_for_btm) %>%
  as.data.frame() %>%
  rownames_to_column(var = "rjacknum") %>%
  mutate(rjacknum = as.numeric(rjacknum)) %>%
  as_tibble()

names(topics_btm)[-1] <- glue("topic_{1:k}")

topics_btm <- topics_btm %>%
  mutate(top_topic_btm = apply(topics_btm[, -1], 1,
                               function(x){which(x == max(x))}),
         top_topic_btm = fct_reorder(as.character(top_topic_btm),
                                     top_topic_btm)) %>%
  full_join(full_resp, by = "rjacknum")

#------------Various visualisations---------------
# Words used by human-chosen topic
grouped_stems %>%
  mutate(issue = fct_lump(issue, 11, w = prop_nz)) %>%
  group_by(issue, word_stem) %>%
  summarise(prop_nz = sum(prop_nz)) %>%
  ungroup() %>%
  mutate(issue = fct_reorder(issue, -prop_nz, .fun = mean)) %>%
  group_by(issue) %>%
  arrange(desc(prop_nz)) %>%
  slice(1:10) %>%
  mutate(word_stem = tidytext::reorder_within(word_stem, prop_nz,
issue)) %>%
  ggplot(aes(x = prop_nz, y = word_stem)) +
  geom_segment(aes(yend = word_stem, xend = 0)) +
  geom_point() +
  facet_wrap(~issue, scales = "free") +
  scale_y_reordered() +
  scale_x_continuous(label = percent_format(accuracy = 0.1)) +
  labs(caption = the_caption,
       x = "Proportion of voters using this word stem",
       y= "",
       subtitle = "Responses to question on the 'single most important
issue in the election', grouped into topics by human coders. Most
mentioned topics at top left of chart.",
       title = "Words used to describe important issues in New
Zealand, 2017")
```

```
#  Chart of human-coded topic by party
topics_btm %>%
  group_by(issue, pv) %>%
  summarise(rwt = sum(rwt)) %>%
  group_by(pv) %>%
  mutate(prop = rwt / sum(rwt)) %>%
  drop_na() %>%
  ungroup() %>%
  mutate(pv = fct_reorder(pv, -rwt),
         issue = fct_reorder(issue, rwt)) %>%
  ggplot(aes(y = issue, x = pv, alpha = prop, fill = pv)) +
  scale_fill_manual(values = c(parties_v, Maori  = "#EF4A42", Other =
"grey")) +
  geom_tile() +
  scale_alpha_continuous(guide = "none", range = c(0, 1)) +
  labs(caption = the_caption,
       x = "",
       fill = "Party Vote",
       y = "",
       subtitle = "Responses to question on the 'single most important
issue in the election', grouped into topics by human coders. Most
mentioned topics at top left of chart.",
       title = "Most important issues for voters in New Zealand,
2017")

# For each combination of 'issue' (human-chosen) and topic, what is
the party?
# for which this is most distinctive to that party? Used for
colouring.
associated_parties <- topics_btm %>%
  filter(!is.na(pv) & pv != "Other") %>%
  count(top_topic_btm, pv, issue) %>%
  group_by(pv) %>%
  mutate(n = n / sum(n)) %>%
  group_by(top_topic_btm, issue) %>%
  arrange(desc(n)) %>%
  slice(1) %>%
  select(top_topic_btm, issue, pv)

d <- topics_btm %>%
  count(top_topic_btm, issue) %>%
  group_by(top_topic_btm) %>%
  mutate(prop_this_btm_topic = n / sum(n, na.rm = TRUE)) %>%
  group_by(issue) %>%
  mutate(prop_this_human_topic = n / sum(n, na.rm = TRUE)) %>%
  ungroup() %>%
  drop_na() %>%
  left_join(associated_parties, by = c("top_topic_btm", "issue")) %>%
  mutate(pv = replace_na(pv, "Other"))

# Heatmap comparing human and BTM topics
d %>%
  filter(prop_this_human_topic > 0.05) %>%
  ggplot(aes(x = top_topic_btm, y = issue,
             fill = pv, alpha = prop_this_human_topic)) +
  scale_fill_manual(values = c(parties_v, Maori  = "#EF4A42", Other =
"grey")) +
  geom_tile() +
  scale_alpha_continuous(guide = "none", range = c(0, 1)) +
  labs(x = "Topic identified by biterm topic modelling",
       y = "Human-coded topic",
       title = "Comparison of human- and machine-coded topics in free
text questions",
       subtitle = "Responses to question on the 'single most important
issue in the election'",
       fill = "Party vote most associated with this human-chosen
topic:")

# Word cloudy chart of topic modelling results
set.seed(123)
plot(btm_mod, top_n = 7, labels = 1:k,
           title = "21 computer-chosen topics in response to free text
questions",
           subtitle = "Responses to question on the 'single most
important issue in the election'") +
  theme(text = element_text(family = "Roboto")) +
  scale_fill_manual(values = colours()[101:121])
```

That's all folks. Take care out there. If you're in New Zealand (or anywhere else with elections coming…), don't forget to vote!