Data Appendix for Measuring political media slant using textual data: evidence from Singapore*

Shen Yan Shun (Lucas) †

March 9, 2019

1 Downloading and parsing of textual data

The sample period is from 1 January 2005 to 31 December 2016. Newspaper articles come from the *Factiva* electronic archive. I query Factiva by names of politicians who served in parliament during the sample period, for the source of *The Straits Times*, in the periods defined above. The query results are downloaded in plain text format, with individual articles automatically separated using a custom written script that obtains the content of the newspaper article itself, and article metadata (article title, date of publication, authors, newspaper section, and the length of the article by word count). The query results from Factiva yield 112,273 newspaper articles, of which 62,132 articles are unique as identified by the title-date tuple.

Transcripts of politicians' speeches in parliament are downloaded from the publicly available and official repository. I write a script to automatically extract speaker-speech chunks by parsing the HTML tags.

*Draft of the main paper is available at `https://lsys.github.io/media-paper/paper.pdf`

†Division of Economics, School of Social Sciences, Nanyang Technological University. Email: lucas@lucasshen

2 Classifying newspaper articles

2.1 Selecting the best classifier

This section details, quite considerably, the use of supervised machine learning classifiers (mainly the *random Forest*) to classify each of the 62,132 unique newspaper articles as *positive* or *negative*; positive if they contain at least one quote from parliament; negative if they do not contain any quote from parliament. A supervised learning algorithm learns from a development set where each observation has a pre-existing label (positive or negative) and a vector of features to learn from.[1]

The initial development set is 1,419 arbitrarily chosen newspaper articles over the period October 2011 to December 2012. The articles are manually labelled, with 275 articles (19%) labelled as positives while the remaining 1144 articles (81%) are labelled negatives. Newspaper articles are mainly represented by a vector of n-grams, with $n = (1, 5)$. These n-grams features are complemented with additional feature engineering using words in the article title such as "parliament" and "debate", as well as the number of quotes in the text using automated text processing.

We choose the period October 2011 to December 2012 since this period is the first few months of the 12th parliament where the opposition gained a significant number of seats as either elected members of parliament, or as non-constituency members of parliament.

Using this initial development set, I put to test the classification performance of six well-known and commonly used supervised classifiers: (1) naive bayes, (2) decision tree, (3) support vector machine, (4) logit, (5) bagged decision trees, and (6) random forest.[2] With the exception of the naive bayes, there are four variants of each classifier depending on how class imbalance—the fact that only 275 of 1,419 newspaper articles in the development set are positives—is addressed. The four variants are: (1) no action taken to address class imbalance, (2) using sample class weights, (3) using SMOTE, and (4) using both sample class weights and SMOTE. Naive bayes has only variants (1) and (3) since it uses the development set's prior class probabilities innately.

SMOTE is the *synthetic minority oversampling technique* to address class imbalances, it augments the existing number of minority class by creating synthetic observations (rather than oversampling with replacement). This technique works

[1] *Features* in the machine learning nomenclature is akin to right-hand-side variables in econometrics.

[2] Varian (2014) provides descriptions of some of these common classifiers.

by selecting a minority class observation a with feature representation x_a, randomly choosing 1 out of the k-nearest-neighbours of a in the feature space, say, observation b. The feature vector of the newly created observation c is then $\mathrm{x}_c = \mathrm{x}_a + r \cdot (\mathrm{x}_a - \mathrm{x}_b)$, where r is a random number between 0 and 1. Oversampling in this way is repeated until the development sample is made up of equal shares of minority and majority class observations (Chawla et al., 2002).[3]

Almost all the classifiers require some model parameters which are not (directly) learned from estimation. Decision trees, for example, can have the maximum tree depth and minimum tree node sample split set before the estimation. We take an agnostic approach to what the best model parameters should be by conducting grid-searches over pre-defined values of model parameters for the best cross-validation scores.

Cross-validation scores for a classifier are from stratified k-folds cross-validation of the development set, which works as follows. The development set is first randomly partitioned into k equally sized and equally represented subsets. In the first of k rounds, the classifier is trained on the first $(k-1)$ subsets and evaluated for classification performance on the left-out subset. This is repeated k times, with the left-out set rotating to the next subset and the training set composed of the other $(k-1)$ subsets, until all k subsets have been used as the left-out set. The final out-of-sample performance scores are then averaged over these k-folds. In all implementations, $k = 5$ unless otherwise mentioned; the choice is arbitrary.[4] [5]

The out-of-sample classification performance of the optimally-tuned classifiers are shown in Table 1, along with the optimally tuned model parameters in the last column (where applicable). Column (1) indicates whether class-weights are used to address class imbalance. Column (2) indicates whether SMOTE is used to address class imbalance. For SMOTE variants, the optimal tuning for the non-SMOTE counterpart is carried over. For instance, the class-weighted decision tree in row (3) has its optimally-tuned maximum depth of 4 and a minimum sample split of 300 carried over to the class-weighted decision tree with SMOTE in row (5).[6]

[3] All implementations of grid-searches and estimations use the *scikit-learn* library (Pedregosa et al., 2012); all implementations of SMOTE use the *imblearn* package (Lemaitre et al., 2017). Both are open-source and actively maintained.

[4] For SMOTE, creation of synthetic observations is done *within* the k-folds split, within each $(k-1)$ subsets. This avoids the accidental contamination of training set information from the validation set, which leads to overstated out-of-sample prediction performance.

[5] In Figure 1, the learning curves are very similar for cross-validation scores based on $k = 5$ and for $k = 10$.

[6] Ideally, the SMOTE variants should have their own grid-searches for the SMOTE variant-specific optimal model parameters, where time permits.

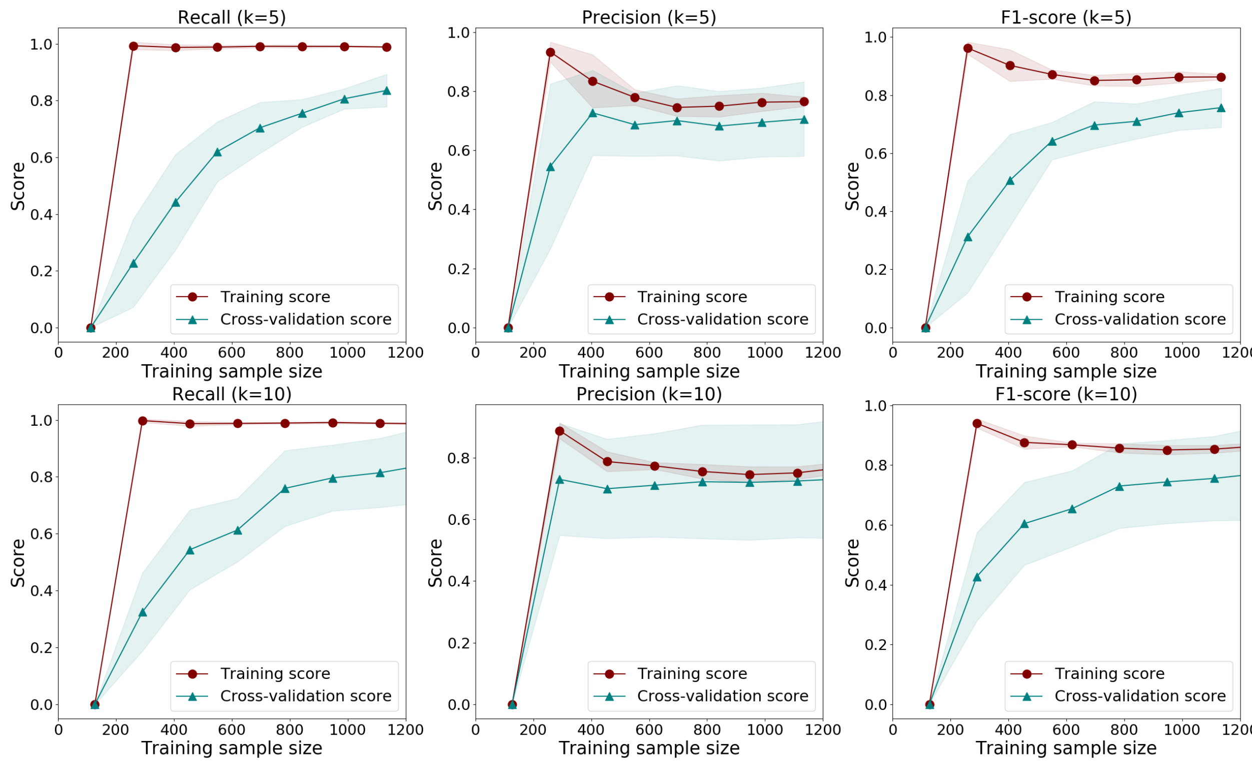


Figure 1. Random forest learning curves

For the five score columns (columns (3)-(7)), the two best scores for each column is highlighted in bold. In column (3), the random forest with class weights and the bagged decision trees without addressing class imbalance have the best *accuracy* scores with 89% accuracy, higher than the baseline score of 81% from a dummy classifier which classifies all observations as 'no' (the majority class). In column (4), the decision tree with class-weights and support vector machine (SVM) with class-weights do best in terms of *recall*—how many of the newspaper articles that contains quote(s) from parliament are correctly classified as such (a dummy classier which classifies as 'yes' for all observations achieves a 100% recall score). In column (5), bagged decision trees without addressing class imbalance and the random forest without addressing class imbalance do best in terms of *precision*—how many of the articles predicted as containing quote(s) from parliament really do contain quote(s) from parliament. In column (7), the two random forest variants without SMOTE do best in terms of the area under the Receiver operating characteristic (ROC) curve. The practical emphases are on recall and precision. Recall ultimately determines how big the sample data will be. Precision determines how much time is saved in the dataset construction pipeline by avoiding false positive classifications. The final metric I consider is the *F-score* in column (6), a handy metric which takes the (harmonic) mean of the recall rate and precision rate, which is topped by the

Table 1. Classification performance

	W	S	Accuracy	Recall	Precision	F-score	ROC AUC	Optimal model parameters
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
NB			0.72 ±0.08	0.85 ±0.10	0.39 ±0.08	0.54 ±0.09	0.81 ±0.10	—
NB		✓	0.68 ±0.07	0.84 ±0.11	0.36 ±0.07	0.50 ±0.08	0.74 ±0.08	—
DT			0.86 ±0.04	0.63 ±0.48	0.70 ±0.26	0.62 ±0.24	0.88 ±0.11	max depth=4, min split=150
DT	✓		0.79 ±0.09	**0.94** ±0.07	0.49 ±0.12	0.64 ±0.10	0.88 ±0.07	max depth=4, min split=300
DT		✓	0.81 ±0.09	0.64 ±0.60	0.51 ±0.17	0.52 ±0.41	0.75 ±0.23	max depth=4, min split=150
DT	✓	✓	0.79 ±0.08	0.89 ±0.14	0.49 ±0.11	0.63 ±0.08	0.83 ±0.05	max depth=4, min split=300
SVM			0.77 ±0.03	0.10 ±0.05	0.28 ±0.19	0.15 ±0.08	0.59 ±0.05	C=0.05
SVM	✓		0.19 ±0.00	**1.00 ±0.00**	0.19 ±0.00	0.32 ±0.00	0.77 ±0.10	C=0.05
SVM		✓	0.57 ±0.32	0.32 ±0.35	0.18 ±0.10	0.22 ±0.10	0.47 ±0.10	C=0.05
SVM	✓	✓	0.44 ±0.60	0.60 ±0.98	0.12 ±0.19	0.19 ±0.32	0.50 ±0.00	C=0.05
Logit			0.86 ±0.05	0.50 ±0.42	0.67 ±0.11	0.55 ±0.34	0.90 ±0.05	C=0.1, intercept=T, penalty=l1
Logit	✓		0.83 ±0.03	0.73 ±0.46	0.56 ±0.07	0.61 ±0.21	0.85 ±0.16	C=0.8, intercept=F, panalty=l1
Logit		✓	0.84 ±0.10	0.56 ±0.68	0.58 ±0.26	0.51 ±0.49	0.73 ±0.29	C=0.1, intercept=T, penalty=l1
Logit	✓	✓	0.82 ±0.04	0.68 ±0.59	0.53 ±0.12	0.56 ±0.34	0.77 ±0.23	C=0.8, intercept=F, panalty=l1
Bag-DT			**0.89 ±0.06**	0.64 ±0.45	**0.79 ±0.17**	0.68 ±0.29	0.93 ±0.05	—
Bag-DT	✓		0.88 ±0.04	0.63 ±0.34	0.73 ±0.17	0.66 ±0.19	0.93 ±0.04	—
Bag-DT		✓	0.87 ±0.06	0.81 ±0.31	0.65 ±0.21	0.70 ±0.12	0.84 ±0.12	—
Bag-DT	✓	✓	0.87 ±0.07	0.80 ±0.33	0.66 ±0.21	0.70 ±0.14	0.84 ±0.13	—
RF			0.86 ±0.01	0.43 ±0.12	**0.77 ±0.18**	0.54 ±0.08	**0.94 ±0.02**	min split=5, max features=sqrt
RF	✓		**0.89 ±0.08**	0.83 ±0.12	0.70 ±0.25	**0.75 ±0.14**	**0.93 ±0.03**	min split=95, max features=sqrt
RF		✓	0.88 ±0.10	0.83 ±0.12	0.69 ±0.27	**0.74 ±0.16**	0.86 ±0.08	min split=5, max features=sqrt
RF	✓	✓	0.87 ±0.10	0.85 ±0.10	0.65 ±0.24	0.73 ±0.15	0.86 ±0.07	min split=95, max features=sqrt

a) All scores are computed from stratified 5-fold cross validations from the best performing model from the gridsearches, which optimises for recall.
b ±values are the 95% confidence intervals computed as 2 times of the standard deviation from the cross-validation scores.
c) Column 2 indicates whether (class) weights are used to address class imbalances. Column 3 indicates whether SMOTE (synthetic minority oversampling technique) is used to address class imbalances. These are computed with the *imblearn* package (Lemaitre et al., 2017). Where applicable, model parameters are carried over from the non-SMOTE counterparts.
d) F-score in column 7 is the harmonic mean of the recall and precision scores. ROC AUC in column 8 is the area under the Receiver operating curve score.
e) The last column indicates the tuned model parameters and their optimal values, where applicable.

random forest with class weights and the random forest with SMOTE.

The random forest with class weights is the best performing classifier by accuracy, the mean of recall and precision, and the area under the ROC curve, and I use it to classify the rest of the newspaper articles in the sample. A brief description of the random forest is as follows. The random forest classifier constructs a bag of decision trees, where each tree is trained on a bootstrap sample from the development set. In addition, each tree node is split based on the best split from a bootstrapped set of features (as opposed to a split based on the best split among *all* features). The best split is usually determined by the information gain or the Gini impurity, which are measures of misclassification from a split. This process of bootstrapping over both observations and features is repeated until we arrive at a collection of trees—the random forest. The individual trees in the random forest then vote for the most popular class (Ho, 1995; Breiman, 2001; Loh, 2014).

Injecting randomness into the construction of decision trees prevents overfitting and improves out-of-sample prediction performance by minimising the correlation between trees while maintaining their individual predictive performance (Breiman, 2001). In an empirical comparison of several supervised learning algorithms in out-of-sample predictions, Caruana and Niculescu-Mizil (2006) find that random forests almost always place first (39% of the times) or second (53% of the times), with only a 0.1% chance of ranking below third place. This is consistent with the findings in Table 1. In addition to being a well-performing classifier, it turns out that the random forest classifier is computationally cheap, with little out of the box tuning of model parameters needed to achieve reasonably good performance.

Figure 1 shows the random forest learning curves for recall, precision, and F-score, where the horizontal axes are the size of the training set, and the vertical axes are the scores from stratified k-fold cross-validation. The random forest classifier is generally well-behaved in the sense that the gap in prediction performance between training and cross-validation scores reduces as the training sample increases. The curves also indicate some scope for better classification performance with additional training observations, which the random forest can get from relearning new classifications. The process of classifying the sample work sets by year and adding the newly labelled newspaper articles into the development set incrementally is described below.

2.2 Classification workflow

Panel A and panel B of Table 2 summarises the classification by year. Column (2) of Panel A describes the initial development set, with a total of 1419 newspaper articles where 275 are positives and 1144 are negatives. The classification takes places in stages, by year, with new classifications augmenting the existing classifier as follows.

First, the initial development set trains the initial random forest, which classifies the remaining (non-initial development set) 4,912 newspaper articles in 2012, where 471 are predicted positives (column (2) of panel A). The 471 predicted positives are then examined, semi-automatically, to extract quotes from parliament.[7] This process of extracting quotes determines for sure, that of the 471 predicted positives,

[7] Extraction of quotes is described in further detail below.

157 are true positives and 314 are false positives. The existing development set is then augmented with the 471 predicted positives, now with *true positive*/*true negative* (yes/no) labels, to yield a development set of 1890 articles of 432 positives and 1458 negatives (column (3) of panel A). This newly augmented development set in 2012 is used to retrain the random forest, which then classifies the work set in 2013 (column (3) of panel B). The predicted positives in 2013 are used to augment the development set for the year 2014, and so on. This repeats until the year 2016 (the end of the sample period), jumps back to 2011, then moves backwards until the year 2005 (the start of the sample period). This order of workflow is reflected in Table 2.

As done with the initial random forests from Table 1, all subsequent implementations use $n = 1000$ decision trees, with Gini impurity criterion used to measure split quality at tree nodes. While a sufficiently large number of trees is sensible to obtain convergence in the prediction errors (Breiman, 2001), and too large a number may incur unnecessary computing costs (partly also because of convergence), the choice of a 1000 trees is mostly arbitrary.[8]

We briefly discuss here an alternative to using machine learning methods to classify newspaper articles—the 'hard-coding' of rules. Hard-coding requires a pre-existing set of rules out of which a newspaper article can be classified as one that contains quote(s) from parliament, or not. Examples of rules are to classify as *yes* if the article contains the word 'parliament', if the article contains the words 'parliament' *and* 'debate', or if the article contains the word 'parliament' n times (and classify as no otherwise).

Machine learning methods is superior to hard-coding for two related reasons. First, since no clear set of hard rules exist, one would have to introspect about what characterises an article that contains quote(s) from parliament. This may induce bias in the eventual sample if the manually curated rules are more sensitive towards articles where the coverage favours the ruling party more than average, or vice versa, where the coverage favours the opposition more than average. Outsourcing of this introspection to machine learning eliminates any potential researcher-induced bias in classification, and in the eventual sample. If anything, it makes more sense if the hard-coded rules are derived from a transparent classifier such as the decision tree (transparent because rules are not obvious in a black-box type of classifier such as a neural network).

[8] The random forest parameters of maximum features and minimum sample splits are retuned with each augmented development set, with little change in the optimal hyperparameters.

Table 2. Classification of sample news articles by year

	2012	2013	2014	2015	2016	2011	2010	2009	2008	2007	2006	2005
Panel A. Development Sets												
Total	1419	1890	2311	2736	3138	3675	3881	4201	4385	4590	5075	5422
Positives	275	432	715	1060	1379	1843	2004	2257	2433	2617	2989	3224
Negatives	1144	1458	1596	1676	1759	1832	1877	1944	1952	1973	2086	2198
Panel B. Work sets												
Total	4912	4942	4499	7221	7087	4667	4367	4582	4394	4722	5198	4111
Pred pos	471	421	425	402	537	206	320	284	205	485	347	384
Pred neg	4441	4521	4074	6819	6550	4461	4047	4398	4189	4237	4851	3727
True pos	157	283	345	319	464	161	253	176	184	372	235	284
False pos	314	138	80	83	73	45	67	8	21	113	112	100
Precision	0.33	0.67	0.81	0.79	0.86	0.78	0.79	0.95	0.9	0.77	0.68	0.74
Panel C. Top 15 features												
1	parliament#	grc	parliament	parliament yesterday	parliament#	parliament yesterday	grc	parliament yesterday	grc	parliament yesterday	parliament yesterday	parliament yesterday
2	grc	parliament yesterday	parliament yesterday	parliament#	grc	grc	parliament#	grc	parliament yesterday	said	said	said
3	parliament yesterday	said	grc	yesterday	parliament	parliament	parliament	parliament	parliament	parliament	parliament	parliament
4	mps	mps	said	grc	said	said	said	parliament#	said	parliament#	asked	yesterday
5	debate	parliament#	mps	said	yesterday	debate	debate	said	parliament#	yesterday	yesterday	grc asked
6	government	government	yesterday	mps	grc asked	grc asked	grc asked	yesterday	yesterday	asked	parliament#	mps
7	mp	yesterday	government	government	debate	yesterday	yesterday	grc asked	grc asked	mps	mps	parliament#
8	asked	debate	debate#	grc asked	asked	asked	asked	debate	debate	grc asked	responding	asked
9	mr	responding	asked	asked	mps	added	minister	mps	mps	debate	grc asked	responding
10	aljunied grc	asked	debate	debate	added	minister	added	asked	asked	minister	debate	debate
11	minister	grc asked	grc asked	mr	minister	responding	responding	responding	responding	responding	minister	mr
12	constituency mp	mp	replying	debate#	mr	mps	mps	minister	mr	added	added	minister
13	approach	mr	added	replying	replying	debate#	replying	added	added	mr	mr	added
14	chua chu kang grc	debate#	minister	minister	debate#	told parliament	mr	told parliament	minister	replying	told parliament	nominated mp
15	house	event	responding	added	government	director	executive	mr	told parliament	told parliament	nominated mp	told parliament

a) Classification starts in the year 2012, going forwards to 2016, jumps back to 2011, then backwards til 2005.
b) The # symbol indicates those words that occur in the title of the newspaper article.

Second, manual introspection is unlikely to be perfect nor complete. Rather than introspect, we turn the classification problem into a purely empirical one by letting the data tell us what works, as in Mullainathan and Spiess (2017). Pang et al. (2002) provide a concrete case using movie reviews, demonstrating that machine learning definitively outperforms human introspection. Some evidence of this is also present in the top features shown in panel C of Table 2. Some features agree with human intuition on what the useful phrases would be, such as '*parliament*' and '*debate*'. Phrases such as '*parliament yesterday*', '*yesterday*', '*responding*', and '*replying*' are obvious (if at all) only on hindsight.[9] [10]

3 Extracting quotes from classified articles

Once the random forest classifies articles in a work set, the predicted positives are examined to ascertain whether they do contain quote(s) from parliament, and if they do, extract the quote(s).

I write a custom script for a simple graphic user interface (GUI) to facilitate the extraction of quotes. The GUI does a handful of tasks automatically: (i) cycles through the predicted positive newspaper articles; (ii) searches for quotes in the current displayed article and cycles through the quotes; (iii) cycles through the downloaded parliament transcripts; (iv) searches for the top n matches in the current transcript for the current quote and cycles through these n matches; and finally (v) stores the quote and other relevant metadata such as the dates and full speech. Quotes are extracted as strings i) inside single inverted commas, ii) inside double inverted commas, and iii) after speech colons.[11]

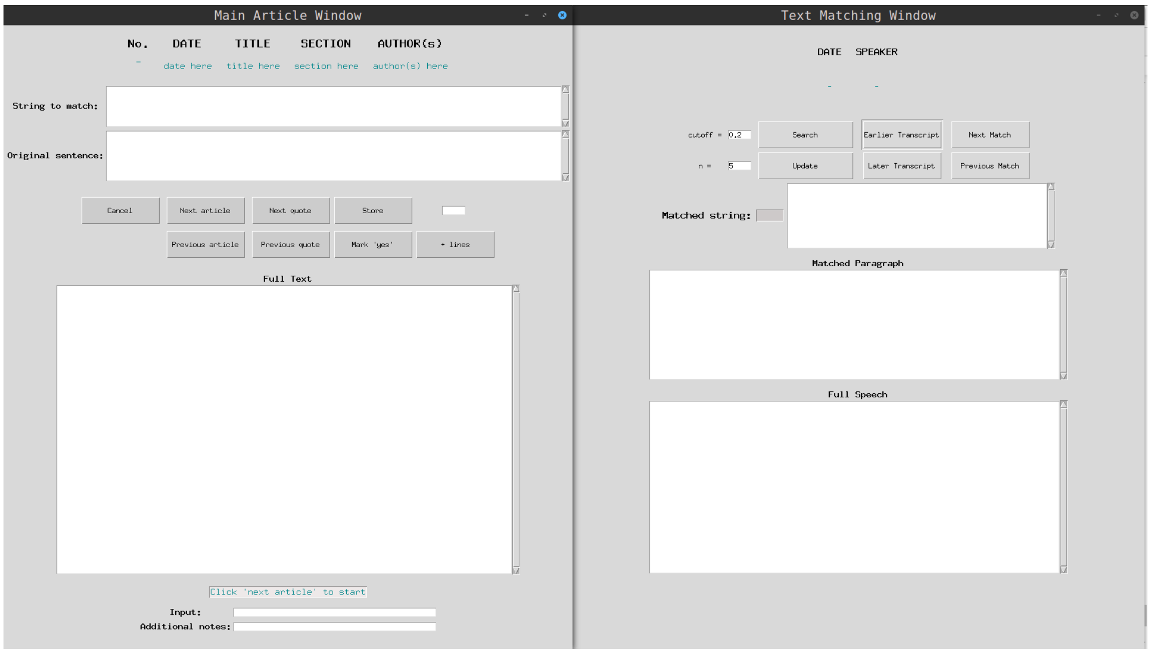
This extraction procedure is partially automated with manual supervision coming in mainly between task (iv) and task (v). I look through the top n matches and stores the match that looks most likely to be the source of the quote in the newspaper article.

The partially automated quote extraction relies on the use of standard edit distance measures from the NLP literature (e.g. Levenshtein distance or Ratcliff-

[9] The two reasons are related because incomplete or imperfect introspection itself may give rise to bias.

[10] I also tried a *naive* classification which classifies as *yes* if a news article contains "*parliament**", and *no* otherwise. The naive classifier has an accuracy score of 0.78 (compared to 0.89 from the random forest), a better recall score of 0.93 (compared to 0.83), a precision score of 0.47 (compared to 0.70), and a F-score of 0.66 (compared to 0.93). Overall, it clearly underperforms relatively to the random forest classifier.

[11] Screenshots of the GUI can be seen in Figure 2.



No. DATE TITLE SECTION AUTHOR(s)

- date here title here section here author(s) here

String to match:

Original sentence:

Full Text

Click 'next article' to start

Input:

Additional notes:

DATE SPEAKER

- -

cutoff = 0.2

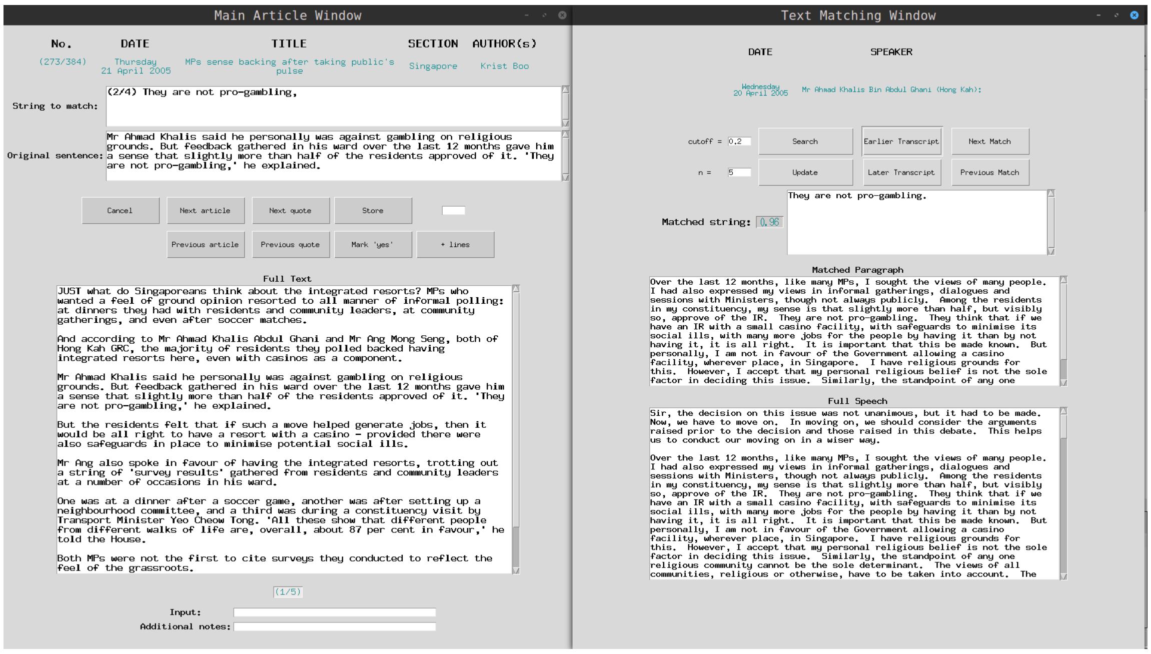
n = 5

Matched string:

Matched Paragraph

Full Speech

(a) GUI example without text



No. DATE TITLE SECTION AUTHOR(s)

(273/384) Thursday 21 April 2005 MPs sense backing after taking public's pulse Singapore Krist Boo

String to match: (2/4) They are not pro-gambling,

Original sentence: Mr Ahmad Khalis said he personally was against gambling on religious grounds. But feedback gathered in his ward over the last 12 months gave him a sense that slightly more than half of the residents approved of it. 'They are not pro-gambling,' he explained.

Full Text

JUST what do Singaporeans think about the integrated resorts? MPs who wanted a feel of ground opinion resorted to all manner of informal polling: at dinners they had with residents and community leaders, at community gatherings, and even after soccer matches.

And according to Mr Ahmad Khalis Abdul Ghani and Mr Ang Mong Seng, both of Hong Kah GRC, the majority of residents they polled backed having integrated resorts here, even with casinos as a component.

Mr Ahmad Khalis said he personally was against gambling on religious grounds. But feedback gathered in his ward over the last 12 months gave him a sense that slightly more than half of the residents approved of it. 'They are not pro-gambling,' he explained.

But the residents felt that if such a move helped generate jobs, then it would be all right to have a resort with a casino - provided there were also safeguards in place to minimise potential social ills.

Mr Ang also spoke in favour of having the integrated resorts, trotting out a string of 'survey results' gathered from residents and community leaders at a number of occasions in his ward.

One was at a dinner after a soccer game, another was after setting up a neighbourhood committee, and a third was during a constituency visit by Transport Minister Yeo Cheow Tong. 'All these show that different people from different walks of life are, overall, about 87 per cent in favour,' he told the House.

Both MPs were not the first to cite surveys they conducted to reflect the feel of the grassroots.

(1/5)

Input:

Additional notes:

DATE SPEAKER

Wednesday 20 April 2005 Mr Ahmad Khalis Bin Abdul Ghani (Hong Kah):

cutoff = 0.2

n = 5

Matched string: 0.96 They are not pro-gambling.

Matched Paragraph

Over the last 12 months, like many MPs, I sought the views of many people. I had also expressed my views in informal gatherings, dialogues and sessions with Ministers, though not always publicly. Among the residents in my constituency, my sense is that slightly more than half, but visibly so, approve of the IR. They are not pro-gambling. They think that if we have an IR with a small casino facility, with safeguards to minimise its social ills, with many more jobs for the people by having it than by not having it, it is all right. It is important that this be made known. But personally, I am not in favour of the Government allowing a casino facility, wherever place, in Singapore. I have religious grounds for this. However, I accept that my personal religious belief is not the sole factor in deciding this issue. Similarly, the standpoint of any one

Full Speech

Sir, the decision on this issue was not unanimous, but it had to be made. Now, we have to move on. In moving on, we should consider the arguments raised prior to the decision and those raised in this debate. This helps us to conduct our moving on in a wiser way.

Over the last 12 months, like many MPs, I sought the views of many people. I had also expressed my views in informal gatherings, dialogues and sessions with Ministers, though not always publicly. Among the residents in my constituency, my sense is that slightly more than half, but visibly so, approve of the IR. They are not pro-gambling. They think that if we have an IR with a small casino facility, with safeguards to minimise its social ills, with many more jobs for the people by having it than by not having it, it is all right. It is important that this be made known. But personally, I am not in favour of the Government allowing a casino facility, wherever place, in Singapore. I have religious grounds for this. However, I accept that my personal religious belief is not the sole factor in deciding this issue. Similarly, the standpoint of any one religious community cannot be the sole determinant. The views of all communities, religious or otherwise, have to be taken into account. The

(b) GUI example with text

Figure 2. Graphics user interface with quote matching and extraction

Obershelp pattern recognition algorithm). Edit distance is a solution to the problem of string similarity. For example, the Levenshtein distance between *intention* and *execution* is five, because five operations are needed to convert *intention* to *extention*: (1) delete i, (2) substitute n by e, (3) substitute t by x, (4) insert u (between *n* and *t*), and finally (5) substitute n by c. This minimum edit distance provides a measure of string similarity or probability alignment between two sequences (Jurafsky and Martin, 2000). The edit distance is implemented using built-in methods in the Python Standard Library.

Let the base string similarity be f. Generally, for a given quote q and a set of speeches $S = \{s_1, \ldots, s_L\}$, the matched speech is speech $s^* \in S$ which satisfies:

$$s^* = \underset{s_\ell}{arg\,max}\Big\{f(q, s_\ell) \,\Big|\, s_\ell \in \{s_1, \ldots, s_L\}\Big\}.$$

4 Modelling topic distributions

This section details the modelling of the topic distribution for the textual data. The goal here is to attribute to speeches and newspaper articles a topical structure. More concretely, we want to be able to say something about whether a speech is about, say, *crime* or *climate change*. The reason this is potentially fruitful is because of the possibility that the media covers politicians differently depending on the area of discussion. For instance, it may well be the case that the ruling party politicians are covered more than opposition politicians in areas such as *healthcare* (as in Petrocik (1996); Puglisi (2011)), and a comparison between ruling party and opposition politicians without controlling for topics may result in an overstatement of coverage of ruling party politicians simply because the ruling party tend to talk more about certain political topics of interest to journalists (and more generally to the public).

4.1 Latent Dirichlet Allocation

We use the *Latent Dirichlet Allocation* (LDA) to model topic distributions (Blei et al., 2003). As a quick establishment of nomenclature, a *document* is some textual data made up of a collection of words. In this study, the document unit is either a newspaper article, or a parliamentary speech.[12] A *corpus* is the collection of

[12] We could have also take an entire parliamentary sitting transcript, which is made up of several speeches, as the document. This however, makes less ontological sense, since the unit of observation is a speech and not an entire sitting.

documents. We have two corpora here: the sample of 3,425 news article corpus, and the sample of 5,227 parliamentary speech corpus. An LDA model is trained on each corpus, from which the topic distributions of individual documents can be retrieved.[13] A *topic* is understood here as a collection of words that occur frequently together. Here are three examples of topics automatically learned from the parliamentary speeches:[14]

1. ⟨*cpf, retirement, minimum_sum, saving, cpf_saving*⟩
2. ⟨*police, home_team, officer, crime, inquiry*⟩
3. ⟨*premium, medishield_life, medishield, insurance, insurer*⟩

LDA models documents as outputs of a simple generative process: topics are characterised by their probabilistic association to words, and documents in turn are characterised by their probabilistic association to topics. To generate a new document given a $K \times 1$ topic distribution $\boldsymbol{\theta}_d$, each word placement in the document is first assigned a topic, according to $\boldsymbol{\theta}_d$, say topic j, and then assigned a word randomly given the topic-specific word distribution for topic j. Documents are multinomial distributions over topics, and topics are multinomial distributions over words in the vocabulary. In reality, documents are not really generated this way, but the generative process as described allows us to reverse the process using bayesian reasoning: to infer the topic distribution of documents given the observed words in the document.

More formally, for K topics, D documents, and N terms, $\boldsymbol{\Phi}$ is the $N \times K$ topic-specific term distribution, where $\boldsymbol{\Phi}_{n,k}$ is the probability of word n appearing in topic k. $\boldsymbol{\Theta}$ is the $K \times D$ document-specific topic distribution, where $\boldsymbol{\Theta}_{k,d}$ is the probabilistic association of document d to topic k. $\boldsymbol{Z}$ is the $N \times D$ topic-to-word assignment matrix, where.$\boldsymbol{Z}_{d,n}$ is the is the probabilistic assignment of topic k to word placement n. $\boldsymbol{W}$ is the $N \times D$ document-term matrix, where $\boldsymbol{W}_{d,n}$ is the observed probability of word n appearing in document d. The generative process described above corresponds to the following joint probability distribution of both the observed and unobserved

[13] We could also use the trained model to get the topic distributions of *unseen* documents, but this is beyond the ambit of this study.

[14] These are taken from the LDA model trained on the parliamentary speech corpus on $K = 92$ topics. The words/phrases in the angular brackets are the top 5 words for topic 4, 18, and 31, with the relevance metric at 0.5. The phrases are words joint by '_', these are word that co-occur frequently and identified as such before the model is trained. We describe this in more detail below. The visualisation of the LDA results can be viewed on `https://lsys.github.io/media-lda/ldavis.html`.

variables:

$$(1)\qquad P\left(\boldsymbol{\Phi}, \boldsymbol{Z}, \boldsymbol{\Theta}, \boldsymbol{W} \mid \alpha\right) = \prod_{k=1}^{K} P(\boldsymbol{\phi}_k) \prod_{d=1}^{D} P(\boldsymbol{\theta}_d \mid \alpha) \prod_{n=1}^{N} P(z_{d,n} \mid \boldsymbol{\theta}_d) P(w_{d,n} \mid \boldsymbol{\phi}_k,\, z_{d,n}),$$

where α is the Dirichlet parameter for the document topic distributions. Going from left to right (from topics to document to terms) we first take the joint probability of $\boldsymbol{\phi}_k$, the distribution over terms for topic k, then take the joint probability over topic distributions for documents, and finally take the joint probability over words. $z_{d,n}$ is the topic-assignment for the n^{th} word in the d^{th} document, which is symmetrical to the d^{th} document's topic distribution. [15]

What we really want here is the document-specific topic distribution given the observed documents (and words), which can be recovered from the posterior:[16]

$$(2)\qquad P\left(\boldsymbol{\Phi}, \boldsymbol{Z}, \boldsymbol{\Theta} \mid \boldsymbol{W}, \alpha\right) = \frac{P\left(\boldsymbol{\Phi}, \boldsymbol{Z}, \boldsymbol{\Theta}, \boldsymbol{W} \mid \alpha\right)}{P(\boldsymbol{W})}.$$

Latent in the LDA refers to the problem of inferring the latent or unobservable topic distribution $\boldsymbol{\Theta}$ of a corpus given the observation of the documents and words. Dirichlet in the LDA refers to the Dirichlet distribution over the K-topics multinomial distribution.

The LDA does use the *bag-of-words* assumption in the sense that documents are treated as a collection of words without regard for order or syntax. This means that a document generated as described by the LDA above is unlikely to be grammatical. This is not an issue since we are only looking at inferring the topic structure rather than generating text.[17]

One major advantage of using LDA to model the topic distribution of the corpus of newspaper articles and parliamentary speeches is that the LDA is unsupervised; no human input is needed to impose a topic structure to the corpus before or after

[15] For a topic j, the probability that the n^{th} word in document d is assigned to topic j is $P(z_{d,n} = j \mid \boldsymbol{\theta}_d) = \boldsymbol{\theta}_{d,j}$. The probability of the n^{th} word in document d given this topic assignment is then $P(w_{d,n} \mid z_{d,n} = j, \Phi) = \Phi_{w_{d,n},j}$

[16] The denominator however is computationally intractable given the vast possibilities of the hidden topic structure. Instead, the posterior is estimated using techniques such as Gibbs sampling, a Markov Chain Monte Carlo technique, which works on the lower dimension of $\boldsymbol{Z}$, the topic-to-word assignments. $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$ are then estimated indirectly from the posterior estimate of $\boldsymbol{Z}$. Blei (2012) furnishes more details on this.

[17] Griffiths et al. (2004) provides details on the extension of topic models to account for syntax.

training. In addition to saving resources on the manual classification of each document's topic(s), the unsupervised automation conveniently avoids research-induced bias from the imposition of a corpus topic structure.[18] [19]

Another advantage of this generative model is how there is no notion of mutual exclusivity. Association of a document to topic j does not preclude association to some other topic i. There is similarly no restriction of a word to any one topic. Instead, words have different probabilistic associations to different topics. This allows for polysemy, where words can have different meanings in different contexts; a common example being (financial) bank in a *Finance* topic vs. (river) bank in a *Fishing* topic.[20]

More specifically for the econometric analyses in mind, a topic distribution for a newspaper article may capture variation in news reporting that is not already captured by the section of a newspaper article.

4.2 Choosing optimal number of topics

The main disadvantage using LDA lies in the choice of K, the number of topics to model. This is chosen by the user before the model is trained, and assumed fixed throughout. A model trained using an arbitrarily chosen K might have too many topics so that many of the K topics contain words with little meaning or coherence, or have too few topics to properly capture the topic granularity of a corpus.[21]

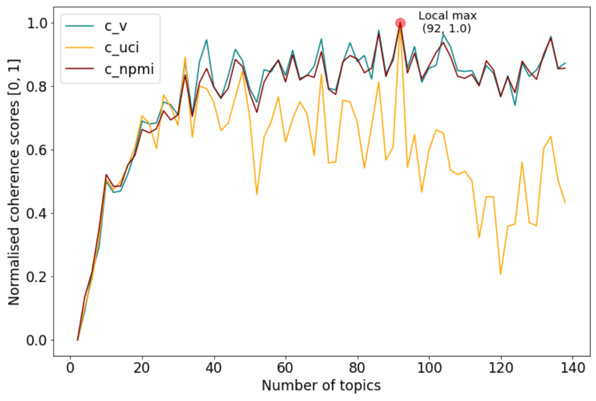
The traditional way to evaluate topic models is to use predictive modelling metrics: such as perplexity and held-out likelihood. However, Chang et al. (2009) show that the traditional measures do not correlate well with how humans evaluate the topics from a model. To measure coherence of topics, they develop a *word intrusion* task, where human participants are presented with six words in a topic. Five of these words are the top five words in the topic (by probability of occurrence). The sixth word, the intruder word, is a randomly chosen low probability word in the same topic but with high probability in some other topic. These six words are then shuffled and presented to the subject. The more coherent a topic is, the easier it

[18] Human annotation is needed after training to label the topic clusters identified by LDA, since LDA merely looks for clusters of topics and do not name them. In fact for this study, no human annotation is needed at all since all we require are (probabilistic) measures of document's topics.

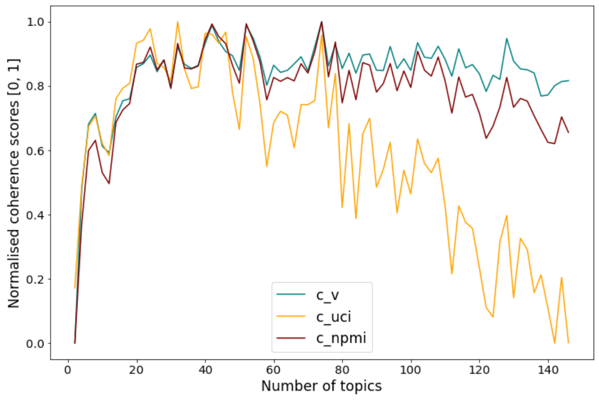
[19] Avoiding researcher-induced biases by using the unsupervised LDA is similar to the case of using a machine learning algorithm vs. hand-curating rules to classify newspaper articles.

[20] Pritchard et al. (2000) develop what is essential the same model for the study of population genetics. The no mutual exclusivity notion meaningfully models individuals as coming from multiple ethnic origins.

[21] For example, should primary/secondary education be in the same topic as tertiary education? A K that is too low may have these two humanly distinguishable topics clustered as one.



(a) Parliamentary speeches



(b) Newspaper articles

Figure 3. (Normalised) Coherence scores

should be for the subject to identify the intruder word. Chang et al. (2009) find that the traditional metrics of perplexity and held-out likelihood are in fact, negatively correlated with human intuitions of topic coherence measured by word intrusion.

Instead of using the traditional perplexity score or using human evaluations, I automate the evaluation of topic coherence and topic model quality using coherence scores recently found to correlate well with human intuition as follows. We train LDA models for the two corpora (one for speeches and one for articles) for topic numbers in steps of 2, starting from 2 topics, up to a 100 over topics. Each iteration computes coherence scores using the top $n = 10$ words. The topics can then be ranked according to the corresponding coherence scores.

Figure 3 plots the (normalised) coherence scores against the number of topics K. The coherence score *c_uci* is based on the pointwise mutual information (PMI) between topic words (Newman et al., 2010). Coherence score *c_npmi* is based on the normalised PMI (NPMI) (Lau et al., 2014). NPMI was introduced as a means of providing better interpretability to the PMI metric, where the NPMI is always between -1 and 1. For comparison, the PMI goes from 0 to $-\infty$. The NPMI also reduces PMI's bias towards words of lower frequency (Bouma, 2009). Coherence score *c_v* combines NPMI and the indirect cosine measure (Röder et al., 2015). These three measures correlate well with human evaluations of topic models (Newman et al., 2010; Lau et al., 2014; Röder et al., 2015).

Figure 3a shows the normalised coherence scores for the topic models trained on the parliamentary speeches. For this corpora, there is the convenient result where all three measures agree on $K^* = 92$. The coherence scores bob up and down, but exhibit a gradual trend upwards as the number of topics increases. We note however that lower number of topics can achieve coherence scores very similar to the local optimum at $K^* = 92$. With an eye on possible robustness checks to varying K, I train two additional speech LDA models for $K = 50$ and $K = 100$ topics.

Figure 3b shows the normalised coherence scores for the model trained on the newspaper articles. The behaviour of the coherence scores is similar here. The best topic number K^*, however, is less obvious. To simplify things, I set $K^* = 40$ as this is the smallest number of topics that appear in the five best Ks from the *c_npmi* and *c_uci*. For comparison, $K = 42$ and $K = 44$ also appears in the best five Ks for the three measures. Similarly, with an eye on robustness checks, and keeping in mind the somewhat arbitrary choice of K, I train two additional article LDA models for $K = 30$ and for $K = 50$ topics and the baseline results survive.[22]

4.3 Implementation

The implementation of the LDA models uses the *gensim* library (Řehůřek and Sojka, 2010). Pre-processing of text is done using both gensim and the *SpaCy* library. Pre-processing of text before training the LDA model is as follows. The documents are first cleaned by removing irrelevant information such as corresponding email addresses in newspaper articles, and time stamps in the speech transcripts. Then a simple algorithm using NPMI is passed over each corpus to look for phrases that co-occur together frequently. Each pass connects two individual tokens if the NPMI exceeds some threshold. These words together usually mean something else entirely as opposed to when they appear individually. In the example of topics above, *minimum sum, cpf saving, home team, and medishield life* are examples of these identified phrases. Two passes are given over the document to potentially identify longer phrases such as *early childhood education* scheme.[23] [24]

Pre-processing of text also involves removing certain words or phrases that are of certain *type* using named-entity recognition. Types that are removed are: *Person, Date, Time, Percent, Money, Quantity, Ordinal* and *Cardinal*.[25] Stopwords are also removed, and the remaining words lemmatised, as is customary. Finally, the Dirichlet parameter is assumed to be symmetric, with $\alpha = \frac{1}{K}$.[26]

[22] Scores are normalised to be between 0 and 1 for comparison. Since I am merely ranking topic numbers by scores, the monotonic transformation does not destroy any relevant information.

[23] The threshold used lies between 0.55 to 0.7.

[24] Examples of words that occur together frequently are names, geopolitical entities (e.g. united nations), locations (e.g. botanical gardens), and phrases such as mean test, early childhood education, life expectancy, maternity leave, environmental sustainability, and so on.

[25] More details on named-entity types can be found at https://spacy.io/usage/linguistic-features

[26] Mechanically, small α values lead to sparse distributions. For topic modelling, a smaller than unit α value captures the assumption that documents are unlikely to have high associations to

5 Additional textual and language measures

This section describes four additional textual and language measures used in the regression analyses as both media coverage outcomes and as controls of speech tonality and language proficiency: (i) objectivity of article and speech components, (ii) polarity of article and speech components, (iii) English grade/readability measure of speech, and (iv) lexical richness of speech.

Objectivity and polarity of article and speech components. I generate objectivity measures (how subjective or objective sounding a text is) and polarity measures (how negative or positive sounding a text is) using the *TextBlob* API for the *pattern* sentiment analyser. The pattern analyser computes objectivity and polarity measures by using parts of speech tagging (to identify nouns, adjectives, etc.), giving different scores of objectivity and polarity to different tokens in different parts of speech. The final scores for a text is the weighted combination of the individual tokens' scores.[27]

The anatomy of a speech can be broken down into three levels: (i) sentence (or sentences) in a speech, (ii) paragraph (or paragraphs) of a speech, and (iii) the entire speech. Objectivity and polarity scores are generated for these three speech components. A distinction can also be made between quotes and the entire sentence containing the quote. Objectivity and polarity scores are also generated for these two quote components, giving a total of five measures of objective and polarity.

These 5 measures can be consolidated into a single principal component using principal component analysis. The results from the principal component analyses for objectivity and polarity are presented in Table 3 and Table 4. The Kaiser-Meyer-Olkin (KMO) measures of sampling adequacy for objectivity and polarity are mostly above 0.7, indicating the five variables have more than sufficient commonality to warrant the analysis. Both the first principal component of objectivity and polarity explain more than 50 per cent of the standardised variation in the linear combination of the five variables.

English language grade/readability of speeches. One concern with media coverage of politician speeches has to do with how coherent the speeches are and how comprehensible they are. We use readability and lexical richness to measure how sophisticated politicians' speeches are. Readability measures of speeches are gener-

numerous topics.

[27] To my knowledge, no large-scale sentiment analyser which is trained on politician speeches or newspaper corpora is currently available. Documentation on the TextBlob API can be found at `https://textblob.readthedocs.io/en/dev/`, while information on the *pattern* sentiment analyser can be found at `https://www.clips.uantwerpen.be/pages/pattern-en#sentiment`.

Table 3. Principal component analysis for textual objectivity

	PC 1	PC 2	PC 3	PC 4	PC 5	
Eigenvalue	2.782	1.086	0.54	0.355	0.237	
Variance (%)	0.556	0.217	0.108	0.071	0.048	
Cumulative variance (%)	0.556	0.773	0.881	0.952	1.000	
Variable	Loading 1	Loading 2	Loading 3	Loading 4	Loading 5	KMO
Full speech objectivity	0.250	0.768	0.581	0.103	0.000	0.659
Paragraph objectivity	0.395	0.471	−0.756	−0.209	0.085	0.754
Sentence objectivity	0.509	−0.191	−0.102	0.765	−0.331	0.798
Quote objectivity	0.515	−0.300	0.185	−0.056	0.779	0.716
Quote sentence objectivity	0.507	−0.248	0.216	−0.598	−0.526	0.771
Overall						0.751

Table 4. Principal component analysis for textual polarity

	PC 1	PC 2	PC 3	PC 4	PC 5	
Eigenvalue	3.063	1.002	0.474	0.293	0.167	
Variance (%)	0.613	0.2	0.095	0.059	0.034	
Cumulative variance (%)	0.613	0.813	0.908	0.967	1.000	
Variable	Loading 1	Loading 2	Loading 3	Loading 4	Loading 5	KMO
Full speech polarity	0.266	0.807	0.522	−0.068	0.000	0.716
Paragraph polarity	0.419	0.391	−0.791	0.203	−0.074	0.802
Sentence polarity	0.501	−0.191	−0.059	−0.772	0.336	0.82
Quote polarity	0.507	−0.297	0.207	0.047	−0.781	0.726
Quote sentence polarity	0.496	−0.265	0.236	0.596	0.522	0.787
Overall						0.775

ated using *Textatistic*. In total, there are five measures of readability: (1) Flesch, (2) Flesch-Kincaid, (3) Gunning-Fog, (4) SMOG (Simple Measure of Gobbledygook), and (5) Dale-Chall.

The first four measures use different weightages of three different pieces of textual information: (i) average word per sentence, (ii) average syllable per word, and (iii) the fraction of text made up of polysyllabic (three or more syllables) words. The Dale-Chall measure uses a list of 3000 easy words, with the measure being a weighted average of the fraction of text made up of difficult words, and the average word per sentence.

In addition to the five measures of readability enumerated above, the principal component analysis for readability also includes the number of difficult words (words not on Dale-Chall's list of easy words), the number of sentences, the number of syllables, and the number of polysyllabic words (words with 3 or more syllables).

Table 5. Principal component analysis of speech grade/readability

	PC 1	PC 2	PC 3	PC 4	PC 5	PC 6	PC 7	PC 8	PC 9	
Eigenvalue	4.65	3.757	0.378	0.092	0.06	0.046	0.009	0.006	0.002	
Variance (%)	0.517	0.418	0.042	0.01	0.007	0.005	0.001	0.000	0.000	
Cumulative variance (%)	0.517	0.935	0.977	0.987	0.994	0.999	1.000	1.000	1.000	
Variable	Loading 1	Loading 2	Loading 3	Loading 4	Loading 5	Loading 6	Loading 7	Loading 8	Loading 9	KMO
Flesch	−0.424	−0.184	0.088	0.203	−0.080	0.796	0.056	0.110	0.288	0.591
Flesch-Kincaid	0.427	0.139	0.360	0.575	−0.032	−0.146	0.004	0.121	0.549	0.542
Gunning-Fog	0.429	0.141	0.395	0.133	−0.003	0.423	0.158	−0.226	−0.606	0.563
SMOG	0.420	0.183	0.091	−0.694	0.106	0.311	−0.215	0.149	0.352	0.588
Dale-Chall	0.345	0.220	−0.821	0.261	0.087	0.250	0.124	0.071	−0.003	0.784
#Difficult words	−0.163	0.479	−0.054	0.123	−0.356	0.063	−0.679	−0.366	−0.007	0.754
#sentences	−0.262	0.414	0.080	0.023	0.751	0.004	0.161	−0.376	0.149	0.71
#syllables	−0.209	0.457	0.119	0.079	0.134	−0.023	−0.114	0.784	−0.285	0.662
#polysyllable	−0.141	0.485	0.026	−0.198	−0.516	−0.055	0.640	−0.072	0.144	0.718
Overall										0.638

The PCA for speech readability is presented in Table 5. The KMO measures are over 0.5, indicating sampling adequacy, and the first principal component explains more than 50 per cent of the standardised variation.

Lexical diversity of speeches. A second way to gauge language sophistication is lexical diversity—the use of unique terms in a text. The earliest known and simplest measure of lexical diversity is the type-token ratio—the ratio of the number of unique terms/vocabulary to the number of total words in a text (Chotlos, 1944). Since then, various measures have been recommended to more accurately capture lexical diversity. Of these measures, four are known to be relatively robust to varying text lengths: (1) Maas, (2) MSTTR, (3) MTLD, and (4) HD-D (McCarthy and Jarvis, 2010; Torruella and Capsada, 2013). Robustness to text lengths is especially desirable since we are comparing speeches of varying lengths (the standard deviation of speech length is 1,825 words). Otherwise, shorter speeches would have inflated estimates of language sophistication simply because word repetition is harder to avoid in longer speeches.

I generate 11 measures of lexical richness in total, including the four length-insensitive measures, using *LexicalRichness*, a small custom module that I wrote.[28] These 11 measures are well-known measures of lexical diversity in the quantitative linguistic literature. Descriptions, formulations, and algorithms can be found in Section 34 of the appendix.

The final PCA analysis of 6 lexical richness measures (including the length-insensitive measures) is presented in Table 6. The KMO measures are mostly

[28] The code is open-source and hosted at `https://github.com/LSYS/LexicalRichness`.

Table 6. Principal component analysis for speech lexical richness

	PC 1	PC 2	PC 3	PC 4	PC 5	PC 6	
Eigenvalue	4.401	0.91	0.288	0.176	0.17	0.055	
Variance (%)	0.733	0.152	0.048	0.029	0.028	0.009	
Cumulative variance (%)	0.733	0.885	0.933	0.962	0.99	1.00	
Variable	Loading 1	Loading 2	Loading 3	Loading 4	Loading 5	Loading 6	KMO
Corrected TTR	0.298	0.777	−0.235	0.101	0.492	−0.002	0.725
Mean segmental TTR	0.442	−0.242	0.122	0.576	0.052	−0.629	0.841
Moving average TTR	0.450	−0.227	0.057	0.384	0.037	0.770	0.81
MTLD	0.428	−0.142	0.558	−0.597	0.354	−0.053	0.947
HD-D	0.408	0.407	0.149	−0.132	−0.792	−0.027	0.817
Maas	−0.403	0.317	0.770	0.369	0.036	0.086	0.949
Overall							0.851

above 0.8, and the first principal component captures more than 73 per cent of the standardised variation.[29]

6 Formulations and algorithms for readability and lexical richness measures

6.1 Readability measures

The five readability measures used in this paper are computed using the open-source *Textatistic* library written in python by Erin Hengel.[30] This section provides details on the readability measures plus their formulations.

[R1] ***Flesch ease of reading measure.*** The Flesh score, developed by Rudolph Flesch, is used as an indication of how easy it is to read a text. The score uses a weighted average of (i) average word per sentence, and (ii) average syllables per word. The higher the score, the greater the ease of reading.

$$\text{Flesch} = 206.835 - 1.015\frac{\text{\#words}}{\text{\#sentences}} - 84.6\frac{\text{\#syllables}}{\text{\#words}}$$

[29] The corrected TTR and root TTR are perfectly correlated, so root TTR was dropped. PCA is then run in a few rounds, dropping variables with low KMO until all variables have KMO ≥ 0.5. This results in the 6 lexical richness measures seen in Table 6.

[30] The source code is hosted at `https://github.com/erinhengel/Textatistic`, with additional documentation at `http://www.erinhengel.com/software/textatistic/`.

[R2] ***Flesch-Kincaid grade level measure.*** The Flesch-Kincaid measure indicates English language competency at US grade levels. Like the Flesch measure, the Flesch-Kincaid measure also uses a weighted average of (i) average word per sentence, and (ii) average syllables per word, but with different weights. The higher the scores, the higher the grade level needed to understand the text (the lower the readability).

$$\text{Flesch-Kincaid} = -15.59 + 0.39\frac{\text{\#words}}{\text{\#sentences}} + 11.8\frac{\text{\#syllables}}{\text{\#words}}$$

[R3] ***Gunning-Fog index.*** The Gunning-fog measure, developed by Robert Gunning, uses a weighted combination of (i) average words per sentence, and (ii) fraction of text made up of polysyllabic words. Higher scores indicate higher reading levels needed to understand the text (or lower readability).[31]

$$\text{Gunning-fog} = 0.4\frac{\text{\#words}}{\text{\#sentences}} + 40\frac{\text{\#polysyllabic}}{\text{\#words}}$$

[R4] ***SMOG.*** The Simple Measure of Gobbledygook (SMOG) is a measure of readability that uses the fraction of text made up of polysyllabic words to estimate the years of education needed to understand a text.[32]

$$\text{SMOG} = 3.1291 + 1.0430\sqrt{30\frac{\text{\#polysyllabic}}{\text{\#sentences}}}$$

[R5] ***Dale-Chall readability measure.*** The Dale-Chall measure, developed by Edgar Dale and Jeanne Chall, uses its own procured list of 3,000 easy words — words that should be familiar to most 4th-grade students — to compute readability.

[31] Though polysyllabic literally means more than one syllable, the term is usually used to refer to words with three or more syllables — the meaning used in this paper.
[32] The SMOG has a history of being used to estimate how difficult it is for patients to comprehend health-related forms and document.

Higher scores indicate higher grade levels.[33]

$$\text{Dale-Chall} = \begin{cases} \left(3.6365 + 15.79\frac{\text{\#}{\sim}\text{Dale-Chall words}}{\text{\#words}} + 0.0496\frac{\text{\#words}}{\text{\#sentence}}\right) & \text{if } \frac{\text{\#}{\sim}\text{Dale-Chall words}}{\text{\#words}} > 0.05 \\ 0 & \text{otherwise} \end{cases}$$

6.2 Measures of Lexical Richness

Loosely speaking, the concept of textual lexical richness has also been called *lexical diversity* and *vocabulary diversity*. The measures are computed using *LexicalRichness*, a small python module I wrote for this paper.[34] This section provides short write-ups on the measures of lexical richness, plus their formulations and implementation psuedocode. Of these measures, (i) Maas, (ii) MSTTR, (iii) MTLD, and (iv) HD-D have been found to be robust to varying text lengths (McCarthy and Jarvis, 2010; Torruella and Capsada, 2013). I write APIs for 11 measures of lexical richness, but the implementation in this paper focuses on the above 4 measures.

[L1] ***Type-token ratio (Chotlos, 1944).*** The type-token ratio (TTR) is the earliest measure of lexical richness, and is used as the base measure of more sophisticated lexical richnes measures:

$$\text{TTR} = \frac{t}{w},$$

where t is the number of unique terms, and w is the total number of words in the text. An obvious drawback of this measure is its sensitivity to text length, making comparisons across texts of varying lengths problematic. As a text grows longer, it becomes harder to avoid repetition, especially with the common uses of functional words and stopwords. As a result the term count (t) increases at a slower rate than the word count (w).

As orientation values, the TTRs of Shakespeare's *Much Ado about Nothing* and *Macbeth* score 0.142 and 0.201 (Malvern and Richards, 2012). A number of suggestions have been made to account for the weakness of the TTR, some of which are described below.

[L2] ***Root type-token ratio (Guiraud 1954, 1960).*** The root type-token ratio

[33] The list of 3,000 easy words can be found at `http://www.readabilityformulas.com/articles/dale-chall-readability-word-list.php` or in plain text format at `https://github.com/erinhengel/Textatistic/blob/master/textatistic/dale_chall.txt`.

[34] The source code is open-source and hosted at `https://github.com/LSYS/LexicalRichness`.

(RTTR) uses the ratio of term count to the *square* of total word count.

$$\text{RTTR} = \frac{t}{\sqrt{w}}$$

[L3] ***Corrected type-token ratio (Carrol 1964).*** The corrected type-token ratio (CTTR) uses the ratio of term count to the square of 2 times the total word count.

$$\text{CTTR} = \frac{t}{\sqrt{2w}}$$

[L4] ***Herdan (1960, 1964).*** Herdan's measure takes the ratio of log of term count to the log of word count.

$$\text{Herdan} = \frac{\log t}{\log w}$$

[L5] ***Summer (1966).*** Summer's measure is similar to Herdan's, but takes double logs instead.

$$\text{Summer} = \frac{\log(\log t)}{\log(\log w)}$$

[L6] ***Maas (1972).*** Maas's measure uses the ratio of the difference between log of word count and log of term count, to square of the log of word count. Unlike the previous 5 measures, the lower the Maas measure, the higher lexical richness is.

$$\text{Maas} = \frac{\log w - \log t}{(\log w)^2}$$

[L7] ***Dugast (1978).*** Dugast's measure is the inverse of Maas's measure: the ratio of the square of the log of word count, to the difference between log of word count and log of term count.

$$\text{Dugast} = \frac{(\log w)^2}{\log w - \log t}$$

[L8] ***Mean segmental type-token ratio (Johnson 1944).*** The mean segmental type-token ratio (MSTTR) estimates a text's lexical richness by taking the average TTR of fixed segments of a text. For example, if a text has 210 words, and a segment size s of 100 is chosen, the first 100 words form the first segment, the next 100 words form the second segment. The MSTTR is the TTR of the first segment plus the TTR of the second segment, divided by two. The remaining 10 words are discarded.

```
choose s # size of segment
split text into segments of size s
for segment in segments
    compute ttr
compute mean ttr
```

[L9] ***Moving average type-token ratio (Covington 2007, Covington and McFall 2010).*** The moving average type-token ratio (MATTR) uses a sliding window across a text, computing the lexical richness as the average TTR score of the windows. For a selected window size of x for example, we first compute TTR for the tokens in positions 1 to x, then compute the TTR for tokens in position 2 to $x+1$, then compute the TTR for tokens in position 3 to $x+2$, and so on till the end of the text for tokens in position y to $x+y-1$ where $x+y-1$ is the last position. The average of the TTR score for all windows is then taken as the final measure.

```
choose x # size of windows
split text into sliding windows of size x
for each window in text
    compute ttr
compute mean ttr
```

[L10] ***MTLD (McCarthy 2005, McCarthy and Jarvis 2010).*** The Measure of Textual Lexical Diversity (MTLD) measures lexical richness using the mean length of sequential words in a text that is able to maintain a minimum TTR threshold score. The recommended threshold t is in the range of [0.66, 0.75], with $t = 0.72$ a common choice. This measure iterates over words in a text with a running TTR score until the score falls below the threshold, at which point the factor counter is increased by 1 and the running TTR score is reset. This process is repeated until the end of the text. The score is the total number of words divided by the count of factors in the text (the average length of sequential words able to maintain the minimum threshold). The same process is repeated with the reversed text, that is, going backwards from the last word to the first. The final MTLD is the average of the forward and reverse scores.

```
choose t
# initialise counters and unique term set
word_counter = 0
factor_counter = 0
set = {}

for word in text
    word_counter += 1
    if word not in set
        add word to set
    ttr = len(set) / word_counter
    if ttr < threshold
        factor_count += 1
        # reset counter and set
        word_counter = 0
        set = {}
score = len(text) / factor_count

repeat with reverse-ordered text
mtld = (forward_score + reverse_score) / 2
```

[L11] *HD-D (McCarthy and Jarvis 2007).* The HD-D score uses the hypergeometric distribution to compute the probability of each word in the text appearing at least once in a random draw d of words without replacement. For example, consider a text containing the word *medium*. Using the hypergeometric distribution, we can find the probability of getting *medium* at least once if we drew a random sample of d words without replacement from the text, say ρ. That is, the probability of getting *medium* at least once in the sample of d words is ρ, or that *medium* will occur in ρ ($\times 100$) per cent of all possible combinations of d words drawn from the text. Its *contribution* to the final score is $\rho \times \frac{1}{d}$. The final score sums over every individual word's *contribution*.

```
choose d
hdd = 0
```

```
for word in text
    compute $\rho$ # prob word of occuring at least once in d draws from text
    contribution = $\rho$ / d
    hdd += contribution
```

References

Blei, D. (2012). Probabilistic topic models. *Communications of the ACM 55*(4).

Blei, D., A. Y. Ng, and M. I. Jordan (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research 3*, 993–1022.

Bouma, G. (2009). Normalized (Pointwise) Mutual Information in Collocation Extraction. *Proceedings of German Society for Computational Linguistics (GSCL 2009)*, 31–40.

Breiman, L. (2001). Random Forests. *Machine Learning 45*(1), 5–32.

Caruana, R. and A. Niculescu-Mizil (2006). An empirical comparison of supervised learning algorithms. *Proceedings of the 23th International Conference on Machine Learning*, 161–168.

Chang, J., S. Gerrish, C. Wang, and D. M. Blei (2009). Reading Tea Leaves: How Humans Interpret Topic Models. *Advances in Neural Information Processing Systems 22*, 288—-296.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16*, 321–357.

Chotlos, J. W. (1944). IV. A Statistical and Comparative Analysis of Individual Written Language Samples. *Psychological Monographs 56*(2), 75–111.

Řehůřek, R. and P. Sojka (2010). Software framework for topic modelling with large corpora. In *Language Resources and Evaluation Conference*.

Griffiths, T. L., M. Steyvers, D. M. Blei, and J. B. Tenenbaum (2004). Integrating Topics and Syntax. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, Cambridge, MA, USA, pp. 537–544. MIT Press.

Ho, T. K. (1995). Random Decision Forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition, Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on VO - 1*, pp. 278.

Jurafsky, D. and J. H. Martin (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall series in artificial intelligence. Upper Saddle River, N.J. : Prentice Hall, 2000.

Lau, J. H., D. Newman, and T. Baldwin (2014). Machine Reading Tea Leaves : Automatically Evaluating Topic Coherence and Topic Model Quality. *Proceedings* *of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)* (Eacl), 530–539.

Lemaitre, G., F. Nogueira, and C. K. Aridas (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research 18*, 1–5.

Loh, W.-Y. (2014, dec). Fifty Years of Classification and Regression Trees. *International Statistical Review 82*(3), 329–348.

Malvern, D. and B. Richards (2012). Measures of Lexical Richness. *The Encyclopedia of Applied Linguistics* (2000).

McCarthy, P. M. and S. Jarvis (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior Research Methods 42*(2), 381–392.

Mullainathan, S. and J. Spiess (2017). Machine Learning: An Applied Econometric Approach. *Journal of Economic Perspectives 31*(2), 87–106.

Newman, D., J. Lau, K. Grieser, and T. Baldwin (2010). Automatic Evaluation of Topic Coherence. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL* (June), 100–108.

Pang, B., L. Lee, and S. Vaithyanathan (2002). Thumbs up?: sentiment classification using machine learning techniques. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 79–86.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay (2012). Scikit-learn: Machine Learning in Python. *12*, 2825–2830.

Petrocik, J. R. (1996). Issue Ownership in Presidential Elections, with a 1980 Case Study. *American Journal of Political Science 40*(3), 825–850.

Pritchard, J. K., M. Stephens, and P. Donnelly (2000). Inference of Population Structure Using Multilocus Genotype Data. *Genetics 155*(2), 945–959.

Puglisi, R. (2011). Being The New York Times: The Political Behaviour of a Newspaper. *B.E. Journal of Economic Analysis and Policy 11*(1).

Röder, M., A. Both, and A. Hinneburg (2015). Exploring the Space of Topic Coherence Measures. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining - WSDM '15*, 399–408.

Torruella, J. and R. Capsada (2013). Lexical Statistics and Tipological Structures: A Measure of Lexical Richness. *Procedia - Social and Behavioral Sciences 95*, 447–454.

Varian, H. R. (2014). Big Data: New Tricks for Econometrics. *Journal of Economic Perspectives 28*(2), 3–28.