Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 13593–13598

Topic analysis in news via sparse learning: a case study on the 2016 US presidential elections

Giuseppe C. Calafiore * **Laurent El Ghaoui** **
Alessandro Preziosi *** **Luigi Russo** ****

* *giuseppe.calafiore@polito.it*
** *elghaoui@berkeley.edu*
*** *alessandro.preziosi@polito.it*
**** *luigi_russo@me.com*

Abstract: Textual data such as tweets and news is abundant on the web. However, extracting useful information from such a deluge of data is hardly possible for a human. In this paper, we discuss automated text analysis methods based on sparse optimization. In particular, we use sparse PCA and Elastic Net regression for extracting intelligible topics from a big textual corpus and for obtaining time-based signals quantifying the strength of each topic in time. These signals can then be used as regressors for modeling or predicting other related numerical indices. We applied this setup to the analysis of the topics that arose during the 2016 US presidential elections, and we used the topic strength signals in order to model their influence on the election polls.

Keywords: Text analytics, news analysis, big data, sparse optimization.

1. INTRODUCTION

A huge amount of textual data is being produced every minute; for instance, bloggers on Wordpress publish 350 blog posts per minute, while Twitter users send over 100,000 tweets per minute. This goldmine of textual data can provide a lot of information on global events and opinions, but the vast majority of it is *unstructured*, hence it is not easy to feed this information to mathematical models. Textual data is high-dimensional (a dataset can contain hundreds of thousands of different words), and not readily measurable or interpretable by a machine. In this paper, we describe a method to treat this high dimensional unstructured data, and produce time-based signals allowing us to quantify and possibly predict phenomena over time. As case study, we apply this method to news articles about the 2016 US presidential elections.

The initial phase in our approach is constituted by suitable technologies for retrieving large amounts of textual data and representing them in numerical format, as discussed in Section 3. Then, we applied a sparse automated topic extraction method to the data. A first step is thus a dimensionality reduction obtained via sparse PCA, as described in Section 4. This allows us to transform the data from a high-dimensional space (where each dimension corresponds to a word) to a more interpretable space, where each dimension represents the concept of a "topic." Such a sparse optimization approach has been used in the context of text analytics in, e.g., El Ghaoui et al. (2013).

After computing the principal components (or topics), we project the topics onto a sliding windowed portion of the news in a given time interval, and thus synthesize a time-signal representing the strength of each topic over time, as shown in Section 5.2. Combining these signals with the public polls, we describe a method for understanding which topics had the most positive and negative impact for each candidate. To this purpose, we compute a sparse Elastic Net regression of the poll results, using the topic strengths as inputs, as discussed in Section 6.

In Section 3, we discuss scraping the source data, we apply a data cleaning step, and finally we code data into a suitable numerical representation. In Section 4 we describe Sparse Principal Component Analysis. In Section 5 we extract a list of principal components from the news articles, and we present some examples of the extracted topics, then we include the time component in order to compute the strength of each topic over time. In Section 6 we cross topic signals with polls data, presenting both a descriptive and a predictive model. Section 7 sums up the results of our study.

2. STATE OF THE ART

Latent Dirichlet Allocation (LDA, see Blei et al. (2003)) is the most common algorithm used for topic extraction, but it does not incorporate sparsity directly into the model, so applying it for text processing tasks requires a final 'thresholding' step to make the result interpretable. Instead of plain LDA, we used sparse Principal Component Analysis (SPCA), since sparse solutions are not only faster to compute, but also more interpretable and provide better generalization to new data.

An example of application of LDA to political data can be found in Jahanbakhsh and Moon (2014), who analyzed the tweets preceding the 2012 election.

In our contribution, we use sparse PCA instead of LDA, and news articles instead of tweets; From a qualitative point of view, we observe that the resulting topics are more

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2017.08.2380

easily interpretable, due to the sparsity constraint. Additionally, instead of using a classical supervised sentiment analysis, we show that the raw topic data can be directly used to compute a sparse regression of the poll results, in a completely unsupervised fashion.

3. SOURCE DATA

The textual corpus we used as a case study in this paper pertains to news articles dealing with the 2016 United States presidential elections. The data covers the year 2016, thus focusing on the last part of the presidential campaign, and on the race between Hillary Clinton and Donald Trump.

3.1 Scraping and data cleaning

We used news articles obtained from the EventRegistry python API (see Leban et al. (2014)). For each day of 2016 we downloaded news articles in English mentioning the main Democratic and Republican candidates. We obtained a total of 144608 articles, about 500 news articles per day, from about 5185 different RSS feeds of major newspapers and blogs (Washington Post, Outside the Beltway, USA Today, Daily Mail, The Hill, Huffington Post, LA Times, CNN, NY Times, Reuters, etc.). In the experiments described in this paper, we only selected articles mentioning either the keyword "Clinton" or "Trump".

Each news article is saved in an SQLite database, containing information about the title, content, source and date of publication. The documents span from January 1st 2016 to November 3rd 2016. The full dataset can be downloaded at: `https://goo.gl/FJ8lzm`

Before analysing our dataset, we applied some simple data cleaning. We removed common unfocused expressions using a list of stop words, and we used regular expressions to detect and remove urls and other short words (1 or 2 characters). We did not apply any stemming, so singular and plural words, for example, are treated as different entities.

3.2 Textual data coding

Once textual data is pre-processed as discussed above, a key step is to code it into a suitable numerical representation. In this paper, we used a standard "vector space model," in which each document is represented as a high-dimensional vector where each entry corresponds to a "term" in the document collection, see, e.g. Chapter 6 in Manning et al. (2008). By term we mean either a single word, or a short sentence (an m-gram). In the machine learning terminology terms are usually referred to as *features*, and the collection of all features appearing in the document corpus constitutes the *dictionary*. For each term i and document j, we define the *term frequency* tf_{ij} as the number of occurrences of term i in document j; we define the *document frequency* df_i as the number of documents in which term i occurs, and the *collection frequency* cf_i as the total number of occurrences of term i in the collection. A crude encoding of the document collection is given by the term-frequency (TF) matrix $[M]_{ij} = \mathrm{tf}_{ij}$, $i = 1, \dots, n$, $j = 1, \dots, N$, where n is the total number of terms and N is the total number of documents in the considered collection. Document j is thus represented by an n-dimensional vector, given by the j-th column of M; the i-th row of M is instead an N-dimensional row vector that contains the counts of term i in each of the N documents. The information that is captured by the term frequency is how important a given word is within a document (the higher the count, the more likely is that the word is a good description of that document). Term frequency is often dampened by a slowly monotonic increasing function (such as $1 + \log \mathrm{tf}_{ij}$), in order to avoid giving too much importance to highly occurring words. Document frequency can instead be interpreted as an indicator of informativeness of a term: a term occurring in all documents (such as stop words, or an unfocused term such as "do", "be", "like", etc.) is not likely to have a semantic relevance in a specific document. On the contrary, focused terms appearing only in few documents (such as, say, "nuclear") are likely to identify documents referring to a narrowly defined topic. Term i appearing in document j is thus more relevant if the overall document frequency of that term is low: this idea leads to a commonly employed encoding of the document corpus, in which a term's frequency in a document is scaled proportionally to the (log) inverse of the document frequency of that term. Namely, the so-called TF-IDF (term frequency, inverse document frequency) matrix of the document collection is defined as

$$[M]_{ij} = \begin{cases} (1 + \log \mathrm{tf}_{ij}) \log \dfrac{N}{\mathrm{df}_i}, & \text{if } \mathrm{tf}_{ij} \geq 1 \\ 0, & \text{if } \mathrm{tf}_{ij} = 0. \end{cases}$$

Further, the columns of M are usually normalized by dividing them by their Euclidean norm. This so-called *cosine normalization* allows for measuring the similarity between two documents (or between a document and a query) by simply computing the inner product of their corresponding vector encodings.

4. SPARSE PCA

Principal Component Analysis (PCA) is a classical method for dimensionality reduction that has been used in text analytics for topic modeling and latent semantic indexing (LSI), see, e.g., Papadimitriou et al. (2000); Hofmann (2001). A key step of standard PCA applied to the documents encoding matrix M amounts to finding vectors $p \in \mathbb{R}^n$ and $q \in \mathbb{R}^N$ such that $\|M - pq^\top\|_F^2$ is minimized (here the suffix F stands for the Frobenius matrix norm). The interpretation is that $pq^\top$ is the best rank-one approximation of M, where vector p contains the terms that best summarize the entire document set, hence gives a description of the main *topic* discussed in the corpus. Vector q contains values q_i, $i = 1, \dots, N$, representing the relevance of the main topic p in the i-th document. Once the first topic p and corresponding topic distribution q are extracted from M, the coding matrix is suitably *deflated* to a matrix M_1, and PCA can be applied again to M_1 in order to extract the second topic, and so on; see Mackey (2009). Observe that M is typically a very sparse matrix, whereas p and q, as obtained from standard PCA, are generally non-sparse. It would indeed be important for the purpose of semantic interpretation of the topic to have a sparse topic model p, containing only few really important terms defining the topic. Similarly, a sparse q vector would point us only to the few most relevant documents in which the topic is discussed. Sparse PCA (SPCA) addresses these requirements, by posing the problem as

$$\min_{p,q} \|M - pq^\top\|_F^2 \quad \text{subject to:} \quad \|p\|_0 \leq k, \ \|q\|_0 \leq h,$$

where $\|\cdot\|_0$ denotes the cardinality (i.e., the number of nonzero entries) of its argument, and k, h are given upper bounds on the desired cardinalities, see, e.g., Zhang et al. (2012); Zou et al. (2006). Empirically, we find that low values of p are desirable, we obtain the best results with $p = 4$ (rarely a more than 4 keywords are necessary to define a topic). The value of q doesn't change the resulting topics much. Depending on the kind of dataset, it should be roughly proportional to how many topics per document we expect to find. Unfortunately, SPCA is not solvable exactly and efficiently for realistic high-dimensional instances. There exist, however, heuristics that proved to be very effective and extremely scalable in practice. One such algorithm, which we employed in the present study, is a thresholded version of the power iteration method, see Shen and Huang (2008): we alternate over the two variables p, q a thresholding and projection operation:

$$[T_k(Mq)]_{\mathcal{B}} \to p, \quad [T_h(M^\top p)]_{\mathcal{B}} \to q, \tag{1}$$

where $T_t(v)$ is the hard thresholding operator on vector v, which returns a vector of the same size as v obtained by zeroing out all but the t largest components of v, and $[v]_{\mathcal{B}}$ denotes projection of v onto the unit Euclidean ball $\mathcal{B}$, that is $[v]_{\mathcal{B}} = v/\|v\|_2$. Clearly, for $k = n$, $h = N$, the recursion in (1) reduces to the standard power iteration method for plain PCA. In practice, the thresholded power iteration method for SPCA is much faster that its plain counterpart, since p and q are maintained sparse at each iteration and thus, being M also sparse, only sparse matrix/vector products are performed at each iteration.

5. TOPIC EXTRACTION

The matrix M can be quite big, in our case it has dimension 144608 by 191482 (the first dimension being the documents, the second being the features), but using sparse PCA, without any knowledge about the articles in our dataset, we can quickly extract a list of principal components that describe all the main topics that were discussed in this election cycle. Then, for each topic, we can extract the most relevant articles, or analyze the evolution of the topic over time. In the next section we list some interesting topics obtained by running SPCA on our textual corpus.

5.1 Example topics

We ran SPCA on a subset of our dataset, removing from M the rows that do not mention "Clinton" or "Trump". We set the sparsity requirement to a total of 4 keywords per topic, and we compute the first 100 principal components. For each topic, we provide the list of keywords, ordered by decreasing weight. Many of the automatically generated topics clearly define a human-understandable subject, and can offer a good overview of the main themes discussed in this election cycle:

[1] [black, african, american, americans]
[2] [women, men, woman, female]
[3] [court, supreme, justice, scalia]
[4] [rally, protesters, supporters, event]
[5] [emails, departement, fbi, email]
[6] [nuclear, policy, korea, foreign]
[7] [wall, mexico, border, street]
[8] [big, banks, money, polls]
[9] [tax, returns, taxes, release]
[10] [foundation, donors, charity, work]
[11] [superdelegates, won, pledged, primaries]
[12] [gun, guns, control, background]
[13] [abortion, life, pro, abortions]
[14] [immigration, immigrants, illegal, issue]
[15] [climate, change, energy, global]
[16] [trade, china, deal, tpp]
[17] [health, care, plan, insurance]
[18] [war, iraq, military, isis]
[19] [north, korean, kim, test]
[20] [star, tweet, anti, image]
[21] [income, class, middle, pay]

We can see that news have talked a lot about african americans [1] and women [2]. Another big topic of the elections has been the death supreme court judge Scalia, and the need for his replacement [3]. Trump's rallies have brought a great amount of supporters and protestors and received a lot of news coverage [4]. Another big subject was the investigation regarding state department emails that were erased from Clinton's server [5]. Many other campaign subjects also appear among the main topics. Nuclear policy [6], the wall with Mexico [7], banks [8], gun regulation [12], abortion [13], immigration [14], climate change [15], international trade [16], health care [17], middle class income [21]. Other topics involve controversies during the campaign, about Donald Trump's tax returns [9], the Clinton Foundation [10] or Trump tweeting the Star of David [20].

In the next section, we discuss how to analyze the dynamics of topic evolution over time.

5.2 Topic dynamics

Once we computed the principal components using SPCA, we can analyze the data in a much simpler space, where each dimension corresponds to a human-interpretable topic. If we include the time component into our analysis, we can extract useful insights from the dataset, by computing the evolution of the relative *strength* of each topic over time. The strength S_t of topic t in a certain time window w, is obtained by computing the *cosine similarity* of the topic vector of interest with respect to all the documents in that time window:

$$S_t = \sum_i (d_i v_t), \tag{2}$$

where d_i is the normalized i-th row of M_w (M_w is a portion of the coding matrix M, containing the columns corresponding to the documents in the considered time window) representing the i-th document published in the time window w, and v_t is a normalized column vector representing a topic.

Figure 1 shows an example of topic strength over time, computed over a daily window. Marco Rubio was competing in the Republican primaries against Trump and lost, so, as expected, news mentioning him taper off after he loses to Trump.

In Figure 2 we can see instead that Mike Pence appears mostly after July 15, date in which Trump chose him as

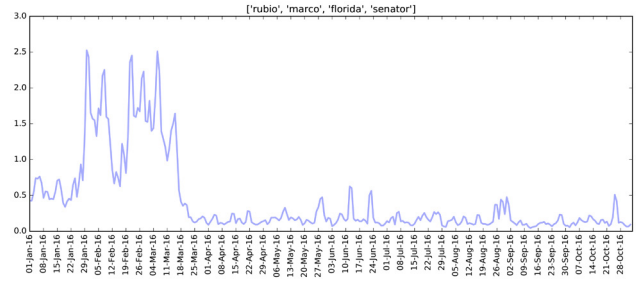

Fig. 1. Strength over time, for topic "Marco Rubio".

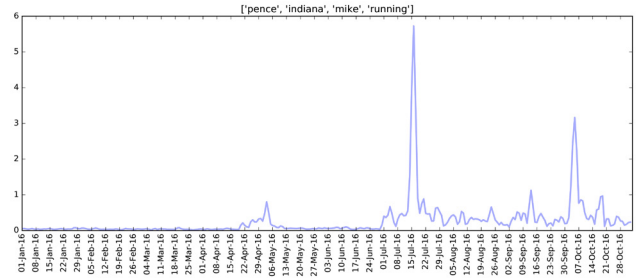

Fig. 2. Strength over time, for topic "Mike Pence".

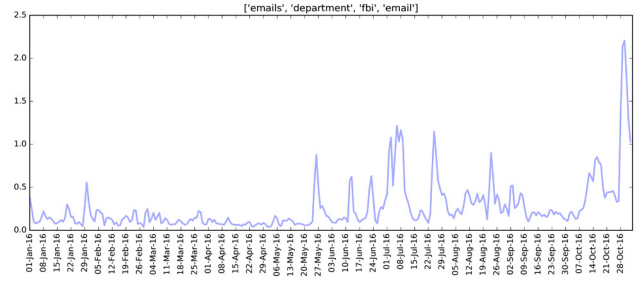

Fig. 3. Strength over time, for topic "emails".

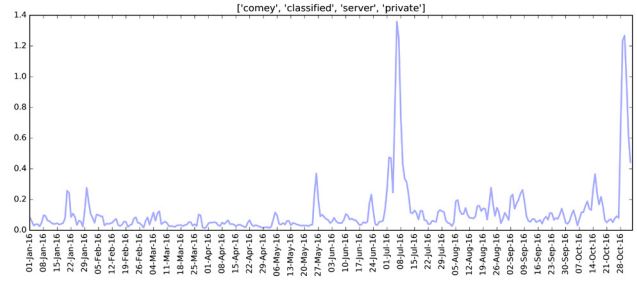

Fig. 4. Strength over time, for topic "Comey" (director of the FBI).

vice president, corresponding to a peak in news mentioning him.

From Figure 3 we can see how the controversy about Clinton's State Department emails has been a hot topic in the last part of the campaign. We have a big peak on May 25th, corresponding to the release of an 83-page report about the State Department's email practices. A second noticeable peak is around July 7th, when the State Department reopened its probe into the email controversy.

Still related to the email scandal, figure 4 shows a topic relative to FBI director Comey, with two spikes, one on July 5th, when the director recommended no charges against Hillary, and one at the end of october, when Comey reopened the case.

We have shown how to produce, from an initial dataset of unstructured textual data, a series of time signals representing how much different subjects have been discussed over time. This kind of temporal signals can then be used to instantiate a predictive or descriptive model, as illustrated in the next section.

6. CROSSING TOPIC SIGNALS WITH POLLS DATA

6.1 Descriptive modelling

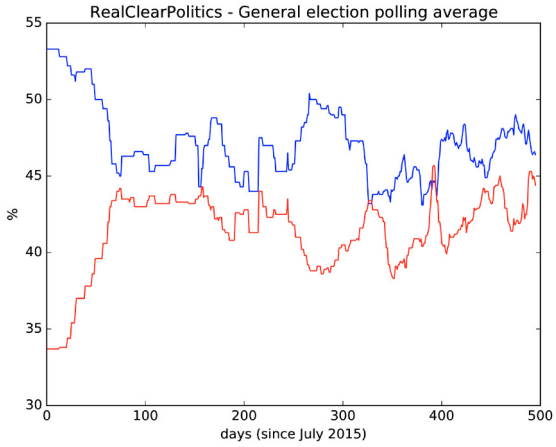

Fig. 5. National polling average. Blue for Clinton, red for Trump.

In this section we discuss how to relate the variation in the polling average to the main topics discussed in the news a few days earlier. The objective of such a model is not necessarily predicting the election's result, but understanding which topics had the most positive and negative impact for each candidate. To this purpose, we model the variation in the polls as a linear combination of the strengths of each topic. The topics with a strong positive coefficient will be the ones who were associated with a rise in the polls for the selected candidate.

Given the high number of topics, computing an ordinary least squares regression would result in overfitting and would not produce a good model because there are more topics than samples (days). To solve this problem, we looked for sparse solutions once again, penalizing solutions that use too many variables. The use of an ℓ_1-norm regularization term in the least squares regression leads to the well known Lasso criterion, see, e.g., Tibshirani (1996). The Lasso, however, has some downsides: if there is a group of variables among which the pairwise correlations are very high, then the Lasso tends to select only one variable from the group, and does not care which one is selected. For problems where the number of variables n is larger than the number of observations N, the Lasso is also not the ideal method, because it can only select at most N variables out of n candidates. Better results can be obtained by using an Elastic Net (EN) criterion, which includes both ℓ_1 and ℓ_2 regularization terms. Simulation studies and real data examples show that the EN in most cases outperforms the Lasso in terms of prediction

accuracy, see, e.g., Zou and Hastie (2005), and, if there are highly correlated variables, EN will tend to keep all in the model (grouping effect), while Lasso only keeps one. The elastic net problem can be expressed, in its basic formulation, as

$$\hat{\beta} = \text{argmin}(\|y - X\beta\|^2 + \lambda_2 \|\beta\|_2^2 + \lambda_1 \|\beta\|_1) \quad (3)$$

Where β is the coefficients vector while λ_1 and λ_2 are, respectively, the ℓ_1 and ℓ_2 penalties. In our case, each column of the matrix X represents a topic, each row represents a day. Element X_{ij} represents the strength of topic j at day i, as computed in (2). The observation vector y is the variation in the poll results after w days:

$$y(t) = \text{polls}(t + w) - \text{polls}(t).$$

The function polls(t) returns the average polling data, obtained from the web at time t. A script to download this poll data is provided at `https://goo.gl/FJ8lzm`. As shown in Figure 5, the poll signals for the two candidates are not specular due to third parties and variable voter turnout, therefore when modelling the news influence on a specific candidate we will use the polling signal relative to that candidate, obtaining a different β for each candidate. Thanks to the high interpretability of the automatically generated topics it is easy to understand and evaluate the results. As an example, in the next paragraphs we show the results of a descriptive model built using 4-word topics, and computing their impact on Trump's polls on a 3-day window ($w = 3$).

The results show that the topics most positively correlated to a raise in the polls of Trump are:

- [emails, department, fbi, email] (+0.73)
- [gun, guns, control, backgorund] (+0.65)

When the news have been talking a lot about guns and immigration or about the FBI investigation on the departement emails erased from Clinton's servers, Trump has seen a rise in polls. Conversely, the most negative topics for Trump, identified by the method, were:

- [women, men, woman, female] (-0.21)
- [sanders, bernie, democratic, hillary] (-0.30)
- [palin, sarah, endorsement, alaska] (-0.66)

Trump's comments on women have had a negative impact on polls, but also two major events seem correlated with his fall in the polls. The endorsement by Sarah Palin was also correlated with a fall in the polls.

Other minor topics, relative to a very punctual event, were also associated with a fall in polls, for example, the talk given by Khizr Khan, the father of a Muslim U.S. soldier killed in combat, at the DNC convention. In Figure 6 we can see Trump's fall in the polls after the Democratic convention (July 25-28) and, during the same period, a lot of mentions in the news of Khan's talk.

Running the same analysis on Clinton, the two most negative topics were

- [sanders, bernie, democratic, hillary] (-0.24)
- [court, supreme, scalia, justice] (-0.59)

As seen in Figure 7, the news of the death of Supreme Court judge Scalia was associated with a fall in Clinton's polls. It also appears that when Bernie Sanders was mentioned a lot in the news, this had a negative correlation with Clinton's results, as many people preferred Sanders to Hillary as the democratic party nominee.

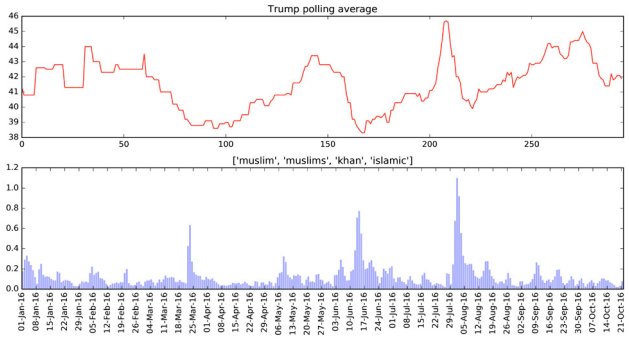

Fig. 6. Above: Donald Trump national polling average. Below: Strength of topic "Khizr Khan".

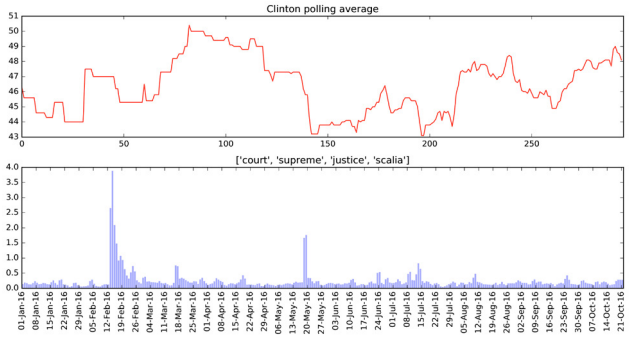

Fig. 7. Above: Hillary Clinton national polling average. Below: Strength of topic "Scalia".

6.2 Predictive modelling

In addition to analyzing past data, we can use EN regression for building a predictive model. In order to do this, and find suitable parameters for the Elastic Net criterion, we use cross-validation. 80% of the dataset is used for training and 20% for testing, the testing days are always contiguous, starting at a random day of the year. For each combination of parameters, we computed the mean absolute error over 100 different splits of training and test data.

For finding suitable ℓ_1 and ℓ_2 penalties, we rewrite the penalties in eq. (3) as

$$\alpha(\rho \|\beta\|_1 + \frac{(1-\rho)}{2} \|\beta\|_2^2)$$

From Figure 8 it is possible to observe that values of α close to zero lead to very poor out-of-sample results. This corresponds to an ordinary least-squares regression without regularization, which results in overfitting and in bad predictive performance on validation data. As shown in the chart, increasing the regularization leads to better results in the testing. The optimal ratio between the ℓ_1 and ℓ_2 penalty is dependent on the overall amount of regularization. For high levels of α, using ℓ_2 penalty is preferred, as ℓ_1 can be too aggressive. In our case, we found a suitable ρ value of 0.7, with an α value of 0.06. We tested models on different windows, from 1 to 30 days. Smaller windows are good for a descriptive model but it is harder to build a predictive model on those smaller scales, due to the noisiness of the polls data. The models with the best predictive power are those computed on a window of ten or more days.

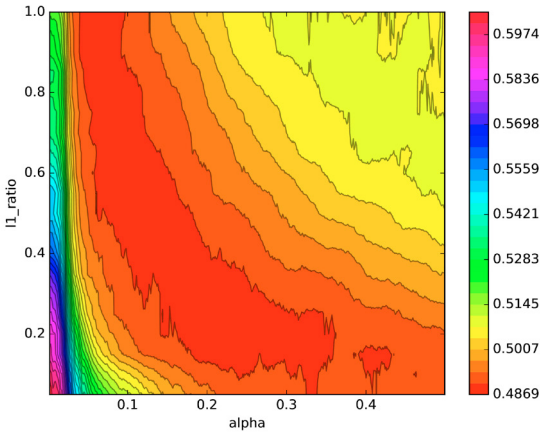

Fig. 8. Mean absolute error in the prediction of the poll percentage change. Results of the cross-validations over different α and ℓ_1 ratio (ρ.)

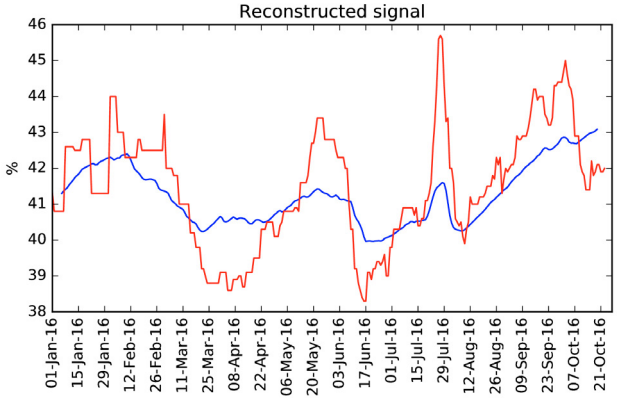

Fig. 9. Donald Trump's poll signal reconstructed using only the news as input. Original signal in red, reconstructed signal in blue.

Models computed with these parameters can predict the movement in polls with a mean absolute error of 0.48 percentage points, on a ten-day window. Figure 9 shows the reconstructed poll signal, using only the topics as an input. The initial state of the system corresponds to the real, initial, value of the polls. Then, each subsequent state is based only on the previous state, and 18 input topics selected by the sparse regression. We can see that the drift is moderate, even if each prediction is based on previous outputs.

7. CONCLUSIONS

We described a methodology for acquiring large and unstructured textual data and transforming it into understandable topics and related time-signals of topic strengths. These signals can then be used as inputs in a sparse regression setting in order to model or predict other related quantitative signals. Using sparse PCA allow us to identify the main topics covered by the corpus, even with no prior knowledge. Forcing sparsity has several advantages, such as speeding up computations, while making the results intelligible by a human operator.

Our key contribution was to introduce time information, by splitting our dataset into time slices and projecting the data in each slice onto our principal components (topics), in order to analyze the relative importance of each topic over time. Once textual data has been converted to real-valued time signals, the information can be used to evaluate a topic's effect on a certain numerical outcome (the election polls in our case). A similar approach, relating a topic strength signal to a numerical outcome, could be used to predict other variables, such as macroeconomic indicators, or market movements.

REFERENCES

Blei, D.M., Ng, A.Y., and Jordan, M.I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan), 993–1022.

El Ghaoui, L., Pham, V., Li, G.C., Duong, V.A., Srivastava, A., and Bhaduri, K. (2013). Understanding large text corpora via sparse machine learning. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 6(3), 221–242.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42, 177–196.

Jahanbakhsh, K. and Moon, Y. (2014). The predictive power of social media: On the predictability of us presidential elections using twitter. *arXiv preprint arXiv:1407.0622*.

Leban, G., Fortuna, B., Brank, J., and Grobelnik, M. (2014). Event registry: learning about world events from news. In *Proceedings of the 23rd International Conference on World Wide Web*, 107–110. ACM.

Mackey, L. (2009). Deflation methods for sparse pca. *Advances in neural information processing systems*, 21, 1017–1024.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Papadimitriou, C., Raghavan, P., Tamaki, H., and Vempala, S. (2000). Latent semantic indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2), 217–235.

Shen, H. and Huang, J. (2008). Sparse principal component analysis via regularized low rank matrix approximation. *J. Multivar. Analysis*, 99, 1015–1034.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Zhang, Y., d'Aspremont, A., and Ghaoui, L.E. (2012). Sparse PCA: convex relaxations, algorithms and applications. In M. Anjos and J. Lasserre (eds.), *Handbook on Semidefinite, Cone and Polynomial Optimization*, International Series in Operational Research and Management Science. Springer.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *J. Computational and Graphical Statistics*, 15(2), 265–286.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67, 301–320.