Measuring Sustainability Reporting using Web Scraping and Natural Language Processing

Alessandra Sozzi
Office for National Statistics
alessandra.sozzi@ons.gov.uk

Abstract

Nowadays the Web represents a medium through which corporations can effectively disseminate and demonstrate their efforts to incorporate sustainability practices into their business processes. This led to the idea of using the Web as a source of data to measure how UK companies are progressing towards meeting the new sustainability requirements recently stipulated by the United Nations. The main challenges associated with this project come from not knowing a priori the structure of companies' websites and thus where to find the relevant information. This report describes the steps taken to develop a web scraping program able to collect sustainability information from websites of a sample of 100 companies and the use of Natural Language Processing techniques to process and validate the data collected. The results show that it is possible to discern the number of companies publishing sustainability information via scraping of their websites. However, the mere action of searching for keywords might not be sufficient in the context of Official Statistics, thus this project goes a step further, applying text analysis to the content of the page to extract additional insights.

1. INTRODUCTION

In September 2015 the United Nations stipulated its requirements for Sustainable Development Goals (SDG's). The Goals are being followed-up and reviewed using a set of global indicators, which result from the aggregation of the national level indicators produced by each member state. However, there are several indicators, such as the sustainability one described below, which currently National Statistical Institutes cannot provide estimates for or cannot disaggregate to the required level. The aim of this research was to carry out an initial proof-of-concept for the indicator "Number of companies publishing sustainability reports, by turnover band, geography, national or global company, sector and number of employees" using an alternative Web-based data source. This indicator relates to SDG Target 12.6 which is "To encourage companies, especially large and transnational companies, to adopt sustainable practices and to integrate sustainability information into their reporting cycle" [1]. This document outlines the steps taken to develop a web scraping program able to collect sustainability information from websites of a sample of the 100 largest UK private companies (ranked by their latest sales) and the use of Natural Language Processing (NLP) techniques to process and extract additional insights from the data collected. Using this method would fill a current gap in reporting the UK's progress against the SDG's and could, with minimal development, be used by other National Statistics Institutes to monitor progress by their countries against the stated indicator.

2. METHODS

Web scraping is a technique employed to extract data from websites programmatically, without the need of a user interaction. The web scraping program, developed in Python, accessed all 100 companies' websites provided and looked for pages that might contain sustainability information (solely in the form of HTML content). However, the mere action of searching for keywords might not be sufficient in the context of Official Statistics, and thus this project went a step further, applying text analysis to the content of the page to extract additional insights.

2.1. The scraper

The scraper needs to navigate through each company website, accessing every internal link. While recursively traversing websites, the scraper flags only the web pages that suggest sustainability content, i.e. at least one of the predefined keywords is found in the URL address of the page or the text of the hyperlink leading to the page. Keywords were chosen by inspecting a sample of websites and looking at what were the most common words used to introduce sustainability content. The list of keywords is as follows: *csr*, *environment*, *sustainab*, *responsib*[1], *footprint*. Once a page is found, it needs to be cleaned before it is saved in a MongoDB[2] database. A high-level overview of the architecture of the scraper is shown in Figure 1.

[1] *sustainab* and *responsib* are both stemmed to match word variations such as *sustainable*, *sustainability* and *responsible*, *responsibility*.
[2] A document-oriented database program. https://www.mongodb.com/

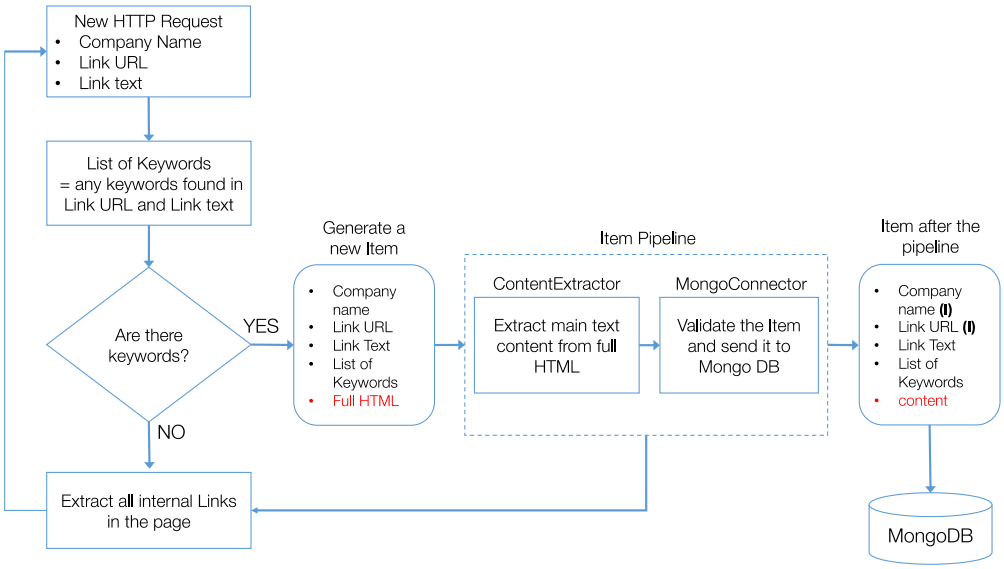


Figure 1. High level overview of the web scraping program

The (I) after Company name and Link URL indicates that these two elements are indexes to support the efficient execution of queries in the database. "Full HTML" and "content" are highlighted in red to emphasise the fact that the "Full HTML" field is replaced by the "content" extracted.

2.2. The Content Extractor

Web pages are often cluttered with additional features, such as navigation panels, pop-up ads and advertisements, around the main textual content. These noisy parts tend to affect the performances of NLP tasks negatively. To avoid this, content extraction methods exist to analyse the HTML behind a webpage and extract what is considered to be the important text. In our scenario, this task is integrated in the scraper architecture and assigned to the *ContentExtractor* component of the Item Pipeline. The *ContentExtractor* receives the raw web page and extracts just the main textual content. The Item then proceeds to the *MongoConnector*, which checks its validity before sending it to a MongoDB database. A comparison of four different content extraction methods was performed on a sample of 30 web pages to find the most suitable for the task to be incorporated within the *ContentExtractor*. The methods initially considered were:

- **Dragnet:** uses machine learning models [2] to extract the main article content and optionally user-generated comments from a web page. It is the result of an ensemble of diverse feature sets and algorithms.
- **Readability:** was originally thought as a browser add-on to turn any web page into a clean view, to give the user a better reading experience. The Readability algorithm gives a score to each part of the HTML page based on a series of deterministic rules such as the number of commas, class names, link density, and various others.
- **BeautifulSoup get_text():** BeautifulSoup is a Python library for parsing HTML files. The get_text() method of this library returns all the visible text of an HTML document. It is the most inclusive method compared to the others because it returns the clutter together with the informative content without trying to make any distinction.
- **<p> Tags:** this method simply extracts all the text enclosed within <p> tags on a web page. In the HTML language, the <p> tag is commonly used to surround textual paragraphs. This method is the most restrictive compared to the other methods, as it does not take in consideration other commonly used HTML tags

often used to display text (such as title tags <h1> <h2> ... <h6>, or tags like <li>, , and so on).

The four extraction algorithms mentioned above were benchmarked against the true content, the gold truth, manually extracted for each of the 30 web pages by a simple action of copying and pasting. For each web page and each of the four methods:

- **Precision**: is the percentage of true tokens[3] among all the retrieved tokens by the method. Precision can be seen as a measure of exactness or quality.
- **Recall**: is the percentage of true tokens retrieved by the method among all the true tokens. Recall can be seen as a measure of completeness or quantity.
- **F1 score**: is the harmonic mean of precision and recall.

Figure 2 visualises the precisions against recalls calculated for each method on the 30 web pages.

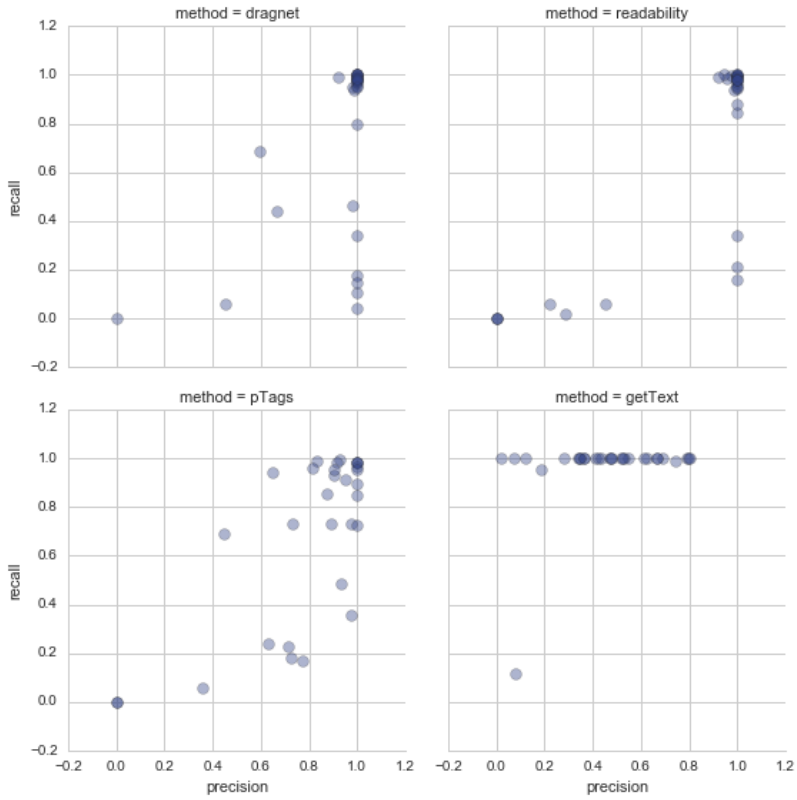


Figure 2. Precision vs. Recall

Contents extracted with the Dragnet and Readability methods (upper two charts) achieve most of the times full precisions and recalls. In the case of the <p> Tags method (bottom left), precisions and recalls are more dispersed, although most of the times above 0.5. The get_text() method (bottom right) shows a different story: precision varies widely between 0 and 0.8, whereas the recall is mostly constant to 1. It is no surprising to see Dragnet and Reliability performing better, in terms of precision, compared to the latter two methods, as they both apply smart rules when deciding whether to keep or discard the pieces of text extracted. Finally, in Table 1 the scores of each method are averaged among all web pages:

[3] Tokens are words separated by whitespace characters, such as a space or line break, or by punctuation characters.

Table 1. Average precision, recall and F1 score for the sample of 30 web pages

	Dragnet	Readability	<p> Tags	get_text()
Precision	0.92	0.82	0.80	0.46
Recall	0.73	0.71	0.68	0.97
F1	0.76	0.73	0.71	0.59

For our NLP task, it is more important to have a high precision, i.e. a high quality of the retrieved content, compared to recall and the F1 score. Therefore, Dragnet is the preferred method. The lower recall implies that the Dragnet method will probably retrieve less content. On the other side, there is the guarantee that the extracted content is relevant for the NLP task.

2.3. Topic Modelling and Latent Dirichlet Allocation

A topic model represents a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. A topic is intended here as a probability distribution over a collection of words.
Latent Dirichlet Allocation (LDA) is a special case of topic modelling [3]. Given a collection of documents, it assigns to each topic a distribution over the words of the entire corpus (topic-words distributions) and to each document a distribution over topics (document-topic distributions) in an entirely unsupervised way. Given its unsupervised nature, any prior annotations or labelling of the documents is not required. Instead, the topics emerge from the analysis of the original texts.
The fact that documents can be described as a distribution over the topics reflects the assumption that documents exhibit multiple topics (but typically not many). This is a distinguishing characteristic of LDA, in contrast with the assumption made by typical mixture models where documents exhibit only a single topic.
LDA uses a generative probabilistic model to explain how the observed words in a collection of documents are generated from an underlying latent structure.
The generative process can be described as follows:

- For each topic, sample a distribution over words from a Dirichlet prior
- For each document, sample a distribution over topics from a Dirichlet prior
- For each word in the document
    - Sample a topic from the document's topic distribution
    - Sample a word from the topic's word distribution
    - Observe the word

The documents themselves are observed, while the topic structure is hidden. The central computational problem for topic modelling is to use the observed documents to infer the hidden topic structure.
Given the observed words in a set of documents and a pre-specified number of topics, LDA works by "reversing" the generative process and inferring back the probability distribution over words associated with each topic and the distribution over topics for each document. This "reversing" can be done in several ways, one of which is Gibbs sampling, an example of a Markov Chain Monte Carlo (MCMC) technique, originally proposed for LDA by Steyvers and Griffiths in [4].
Inferring the word-topic assignments allows for easily deriving the two unknown, hidden parameters, which represent the ultimate goal of LDA:

- the distribution over vocabulary for topic k
- the topic proportion for topic k in document d

Here we used LDA to identify topics on the text extracted from scraped web pages. Topics allow us to understand on what areas of sustainability companies are focusing their action: environment, communities, supporting charities, employee well-being, etc. The statistical model we have described is conditioned on three parameters: the Dirichlet hyperparameters α and β and the number of topics T. Selecting the number of topics is one of the most problematic modelling choices in finite topic modelling [5]. There is no clear method for choosing this parameter and the degree to which LDA is robust to a poor setting of it is not well-understood. In practice, varying the number of topics tends to vary how "finely grained" the resulting topics are. To decide the appropriate number of topics for the scraped pages a series of numbers were experimented: 5, 7, 10, 15 and 20. For assessment and evaluation, the LDAvis [6] was used (Section 3.2). In the model with 5 topics, the resulting topics were broad and overlapped in the 2-dimensional representation. As the number of topics increased from 5 to 10 results improved, and meaningful topics (words combinations) began emerging. In the solution with 20 topics, the impression was that the topics were picking out inconsistent word combinations. The final choice was 15 topics.
In accordance with [5], the fact that the number of topics and the composition of the inferred topics can vary in this manner should reinforce the idea that an individual topic has no interpretation outside the particular model in use. This characteristic/limitation of LDA was already made clear by Blei and his co-authors in the original [3].
α and β, also called *concentration parameters*, are set respectively to 0.001 and 0.1. The low value of β affects the granularity of the model, leading to topics which may contain a mixture of just a few of the words. The low α value defines the smoothness of the topic proportions, meaning that it is more likely for a document to contain a mixture of just a few of the topics instead of many.
On the corpus and the related vocabulary two additional tunings improved the results:

- Removing trailing words: trailing words are words that appear few times in the whole corpus. Words that appear less than 2 times were removed.
- The removal of pages with < 5 words.

Because of this, 2 companies were dropped from the analysis.

3. RESULTS

A total of 563 sustainability-related web pages were collected from 59 companies. 35 companies did not have any sustainability pages published on their websites and thus no pages were found by the scraper. For the remaining 6 companies the following problems were identified: 2 websites had an entrance form which didn't allow the scraper to move further, 1 website had the sustainability content published on a separate website (since the scraper does not follow external links the content was missed), 1 website was down for several days, 1 website had some content related to sustainability in the "Mission & Values" page, 1 website had only content related to a charity foundation (no keyword was matched and so the content was missed). Two additional findings were found:

1. If we consider the companies ranked according to sales, in the first half of the rank, 80% of the companies have sustainability-related content on their websites. In the second half of the rank, only 45% of the companies have sustainability-related content on their websites.
2. Companies on top of the rank are mainly construction or manufacturing companies and big retailers whereas companies in the bottom are mostly companies working in the service industry (recruitment consultancies, travel agencies and so on).

3.1. The topics

The topics are here presented in a tabular layout[4] to promote comparison of words both within and across the latent topics.
For each topic, the 5 most relevant words are listed on the left. The ranking of the terms within topics is computed based on their relevance [6], that is, based on how much discriminatory the terms are for the specific topic.

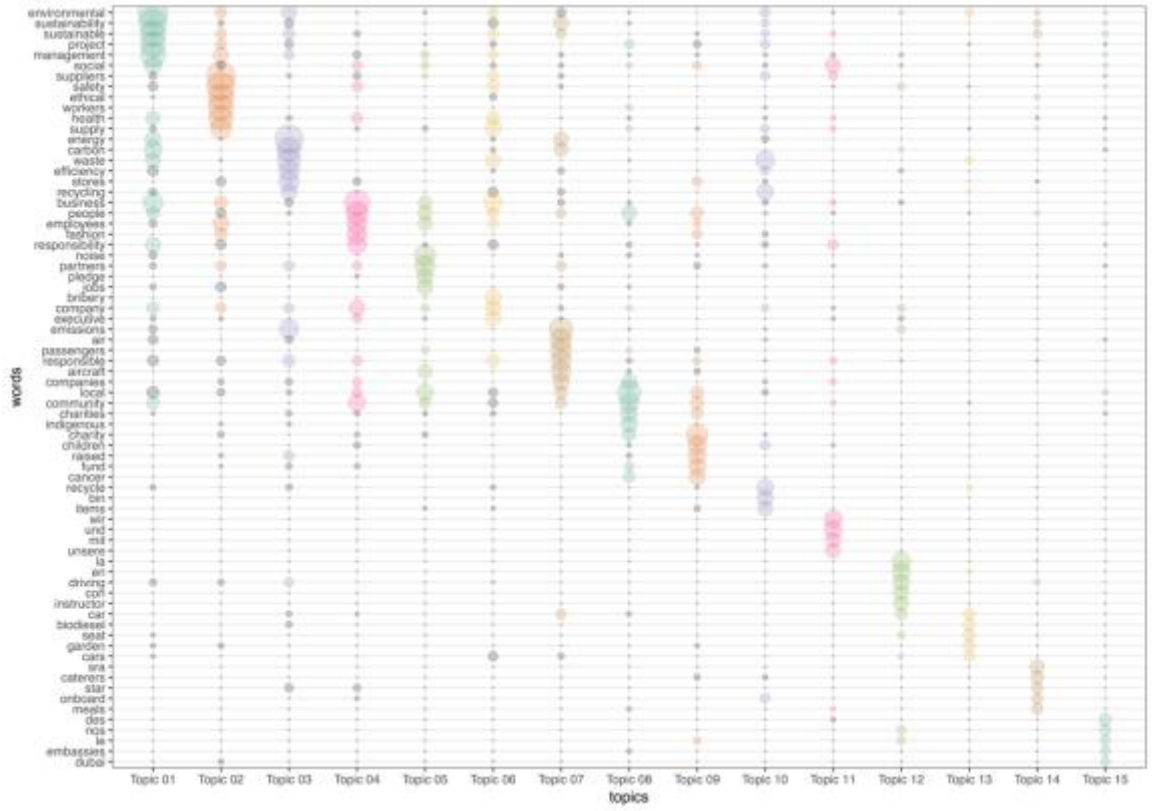


Figure 3. Tabular representation of the topics

[4] This tabular layout is originally inspired by [7] but for this paper the computation of the relevant words and the numbering of the topics follows [6] to make the visualisation consistent with Figure 4.

3.2. Visualising topics as distributions over words

The LDAvis package [6] allows exploring in more detail the topic-words distributions produced by the LDA algorithm. The left panel visualises the topics as circles in the two-dimensional plane whose centres are determined by computing the Jensen–Shannon[5] divergence between topics, and then by using multidimensional scaling to project the inter-topic distances onto two dimensions. Each topic's overall prevalence is encoded using the areas of the circles. The right panel depicts a horizontal bar chart whose bars represent the individual terms that are the most useful for interpreting the currently selected topic on the left. A pair of overlaid bars represents both the corpus-wide frequency of a given term as well as the topic-specific frequency of the term. The λ slider allows ranking the terms according to term relevance. An interactive version of the visualisation is available here.

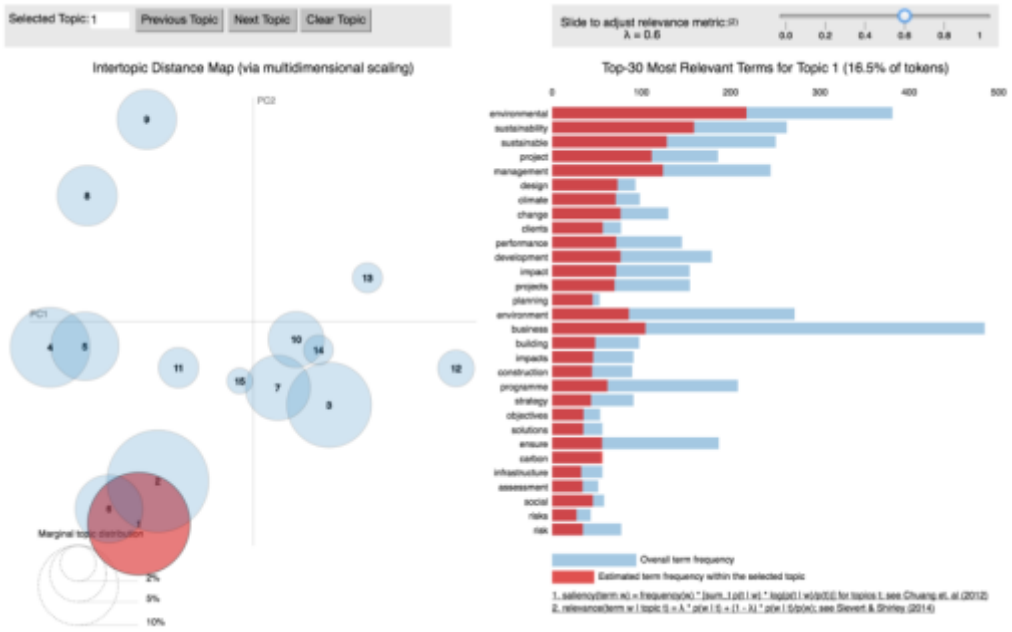


Figure 4. Topics as distribution over words

3.3. Visualising companies as distributions over topics

To understand how the industry affects sustainability reporting, the document-topic distributions are grouped by company and averaged. The average is computed by combining the topic-distributions (one for each web page) for a certain company and weighting the topic proportions by the number of words in each web page.
These can be referred to as company-topic distributions. This allows comparing the topic distribution of companies that belong to the same industry. As an example, Fashion Retailers (on the left) and Transport companies (on the right) are shown in Figure 5. Each plot is constructed as follows:

- Topic numbers are on the x-axis. This number is taken from the topics visualisation in Figure 4
- The y-axis measures the weighted topic proportions
- Each line represents a company belonging to that particular industry
- The boxes are used to show some of the most frequent terms for some of the topics

[5] A popular method of measuring the similarity between two probability distributions. Two distributions p and q are similar if they are similar to their average (p + q)/2.

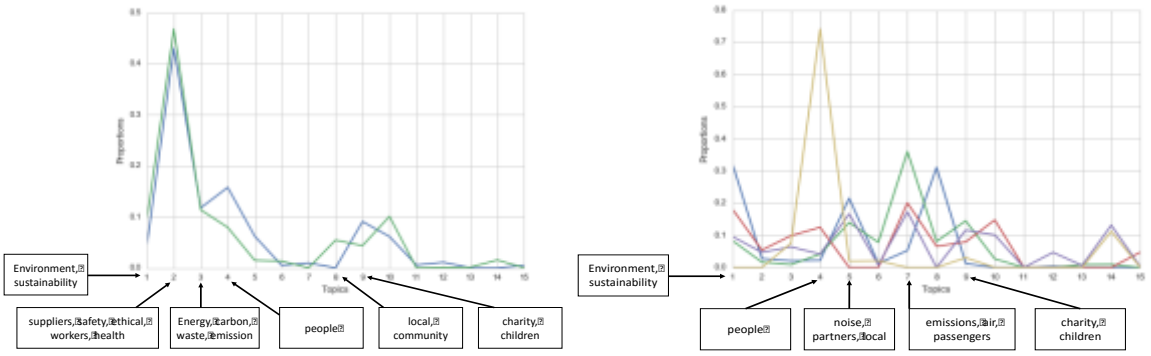


Figure 5. Fashion Retailers (on the left) and Transport companies (on the right) company-topic distributions

The two fashion retailers split their distributions between topics that relate to the environment, and topics related to people, charities and the community. Companies in Transport industry have more heterogeneous topics distributions, although they still show some commonalities.

4. CONCLUSIONS

A gap was identified in producing data for indicator 12.6 for the SDG's in the UK and this paper discusses the development of a prototype to fill this gap. Given the generality of this approach, only small developments would be needed to allow other countries to use this prototype in their reporting of progress towards the SDG's.

The results show that it is possible to discern the number of companies publishing sustainability information via scraping of their websites.

Overall, 563 sustainability-related web pages were scraped from 59 companies. The scraper could not access or did not find the sustainability content for 6 companies. For the remaining 35 companies, the scraper rightfully did not find any sustainability content as there was any published. This result is a good starting point.

Future developments could be directed towards making the scraper more robust in terms of the keywords used to identify sustainability-related pages and the handling of specific obstacles, such as forms, pages translated into multiple languages, and so on.

LDA was then used to determine topics in the web pages. By tuning the model parameters, the topics became much clearer and consistent.

The extent to which the prevalence of sustainability reporting is related to industry and size of the company is worth exploring more; especially when a much larger number of companies will be considered.

The analysis of the text extracted from the web pages shows that the subject of sustainability is much more nuanced than what might result from a mere keyword analysis.

APPENDIX - WEB SCRAPING ETIQUETTE

Web scraping is defined as the mechanism through which a computer reads a web page without the need of user intervention. Instead of presenting the data served by the website on a computer screen, the web scraping software directly grabs the content, or a subset of it, from the returned page. Web scraping practices can fall into unethical territory in two main ways:

1. **The amount of burden imposed on the website.** This can take the form of the scraper reading the pages of a website much faster than a person could. If unexpected, the situation can cause difficulty for the servers to handle it and, ultimately, cause degraded performance for the rest of users of the website. Although, presumably, the web scraper is likely to be blocked before arriving at this. How much burden the web scraper can impose on a website vastly depends on several factors, such as how much traffic the website normally handles.
2. **How the data extracted are used.** This aspect is often debated, as there is no clear distinction between legal and illegal web scraping.

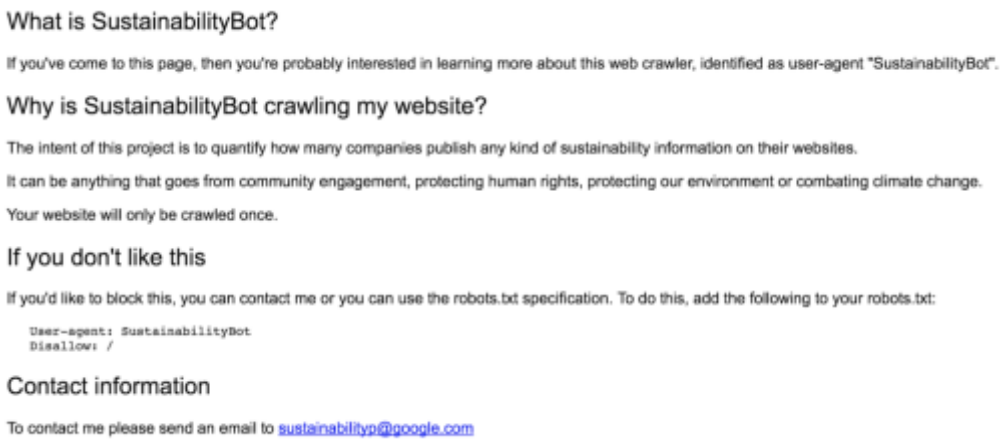
Among web scraping developers, exists a series of widely recognised rules of thumb, or 'etiquette', to follow for polite behaviour. The following steps were taken to comply with the web scraping etiquette:

- **Limiting the crawl depth to 3:** Websites contain multiple pages, which in turn can contain subpages. A website's homepage has a crawl depth of 0. Pages linked directly (with one click) by the homepage have a crawl depth of 1; pages linked directly by depth-1 pages have a crawl depth of 2, and so on. In this project the depth limit is 3, considered an exhaustive depth for the purpose of this project.
- **Download delay set to 5:** the number of seconds that the downloader waits between every consecutive page download from the same website. The download delay parameter can be used to avoid hitting servers too hard. For this project, the download delay is set to a conservative 5 seconds.
- **Setting a meaningful User Agent:** the User Agent string is the way the web browser tells a website information about the browser and operating system accessing the web page. This information allows the website to customise content for the capabilities of a particular device. Given the fact that the scraper is not a browser but a script, the user agent field can be used instead to include a URL where the website administrator may find out more information about the project.

  The user agent string set in this project:

  SustainabilityBot http://goo.gl/4VlxkT

  points to this page to know more about the project:



What is SustainabilityBot?

If you've come to this page, then you're probably interested in learning more about this web crawler, identified as user-agent "SustainabilityBot".

Why is SustainabilityBot crawling my website?

The intent of this project is to quantify how many companies publish any kind of sustainability information on their websites.

It can be anything that goes from community engagement, protecting human rights, protecting our environment or combating climate change.

Your website will only be crawled once.

If you don't like this

If you'd like to block this, you can contact me or you can use the robots.txt specification. To do this, add the following to your robots.txt:

```
User-agent: SustainabilityBot
Disallow: /
```

Contact information

To contact me please send an email to sustainabilityp@google.com

Figure 6. Page created to give information about the project

- **Respecting robot.txt:** website owners use the robots.txt file to give instructions about their site to web robots (such as web scrapers). A robots.txt is a file that can be accessed, when present, at the root of a website by simply specifying the root URL followed by /robots.txt.
  robots.txt files use the Robots Exclusion Standard[6], a protocol with a small set of commands that can be used to indicate what can be accessed on the website and by who. The instructions in robots.txt files cannot force a robot to follow the proposed behaviour; instead, these instructions act as directives. This web scraping project respects robots.txt policies.

REFERENCES

[1] United Nations Statistics Division, Final list of proposed Sustainable Development

[2] M. E. Peters and D. Lecocq, Content Extraction Using Diverse Feature Sets

[3] D. M. Blei, A. Y. Ng and M. I. Jordan, 2003, Latent Dirichlet Allocation, Journal of machine Learning research, 3, 993-1022

[4] M. Steyvers and T. Griffiths, 2006, Probabilistic Topic Models, Latent Semantic Analysis: A Road to Meaning

[5] H. M. Wallach, D. Mimno and A. McCallum, 2009, Rethinking LDA: Why Priors Matter, Advances in neural information processing systems, 22, 1973–1981

[6] C. Sievert and K. E. Shirley, 2014, LDAvis: A method for visualizing and interpreting topics, Proceedings of the workshop on interactive language learning, visualization, and interfaces, 63-70

[7] J. Chuang, C. D. Manning and J. Heer, 2012, Termite: Visualization Techniques for Assessing Textual Topic Models, Advanced Visual Interfaces, 74–77

[6] http://www.robotstxt.org/