SABLE: Tools for Web Crawling, Web Scraping, and Text Classification

Brian Dumbacher[1], Lisa Kaili Diamond[1]
Brian.Dumbacher@census.gov, Lisa.Kaili.Diamond@census.gov
[1]U.S. Census Bureau, 4600 Silver Hill Road, Washington, DC 20233

Proceedings of the 2018 Federal Committee on Statistical Methodology (FCSM) Research Conference

Abstract
For many economic surveys conducted by the U.S. Census Bureau, respondent data or equivalent-quality data can sometimes be found online such as on respondent websites and government agency websites. An automated process for finding useful data sources and then scraping and organizing the data is ideal but challenging to develop. Websites and the documents on them have various formats, structures, and content, so a long-term solution needs to be able to deal with different situations. To this end, Census Bureau researchers are developing a collection of tools for web crawling and web scraping known as SABLE, which stands for Scraping Assisted by Learning (as in machine learning). Elements of SABLE involve machine learning to perform text classification and autocoding. SABLE is based on two key pieces of open-source software: Apache Nutch, which is a Java-based web crawler, and Python. This paper gives an overview of SABLE and describes research to date, potential applications to economic surveys, efforts in moving to a production environment, and future work.

Key Words: U.S. Census Bureau, economic statistics, web crawling, web scraping, text classification

1. Introduction

1.1 Background

For many economic surveys conducted by the U.S. Census Bureau, respondent data, equivalent-quality data, and relevant administrative records can sometimes be found online. For example, the Census Bureau conducts public sector surveys of state and local governments to collect data on public employment and finance (U.S. Census Bureau, 2017a). Much of this data is publicly available on respondent websites in Comprehensive Annual Financial Reports (CAFRs) and other publications. Another example of an online data source is the Securities and Exchange Commission (SEC) EDGAR database. The EDGAR (Electronic Data Gathering Analysis and Retrieval) database contains financial filing information for publicly traded companies and is used often by Census Bureau analysts to impute missing values and validate responses for many economic surveys. Going directly to online sources such as these and collecting data passively has a lot of potential to reduce respondent and analyst burden (Dumbacher and Hanna, 2017). For the most part, the Census Bureau's processes for collecting economic data from online sources are manually intensive. Efficiency can be improved greatly by using automated methods such as web scraping (Mitchell, 2015).

1.2 Challenge

An automated process for finding useful data sources and then scraping and organizing the data is ideal but challenging to develop. Websites and the documents on them have various formats, structures, and content, so a long-term solution needs to be able to deal with different situations. To this end, Census Bureau researchers are developing tools for web crawling and web scraping that are assisted by machine learning. This collection of tools is known as SABLE, which stands for Scraping Assisted by Learning. Elements of SABLE involve machine learning to perform text classification [for a discussion of text analytics topics, see Hurwitz *et al.* (2013, chap. 13)] and autocoding (Snijkers *et al.*, 2013, p. 478). Text classification models are used for different reasons, such as predicting whether a document contains useful data or mapping scraped data to Census Bureau terminology and classification codes.

Disclaimer: Any views expressed are those of the authors and not necessarily those of the U.S. Census Bureau.

1.3 Outline

The rest of the paper is organized as follows. Section 2 gives an overview of SABLE, its machine learning methodology, underlying software, and architecture design. Section 3 covers potential applications and ongoing areas of research such as public sector surveys, SEC metadata, and text classification problems for assigning codes to survey write-in responses. SABLE is currently being moved from a research environment to a production environment, and Section 4 describes this effort. Lastly, Section 5 describes future work, particularly ideas for quality assurance.

2. SABLE Overview

2.1 Main Tasks

SABLE performs three main tasks: web crawling, web scraping, and text classification. Web crawling is the automated process of systematically visiting and reading web pages. Web crawlers, also known as spiders or bots, are typically used to build search engines and keep website indices up to date. For SABLE, web crawling is used to discover potential new data sources on external public websites and to compile training sets of documents for building classification models.

Web scraping involves finding and extracting data and contextual information from web pages and documents. This is an automated process and an example of passive data collection, whereby the respondent has little awareness of the data collection effort or does not need to take any explicit actions. In order to scrape data from some documents, they might have to be converted to a format more amenable to analysis. This is especially true for documents in Portable Document Format (PDF). Models based on the frequencies and locations of important word sequences can be employed to find useful data in documents.

Text classification is the task of assigning text to a category, or class, based on its content and important word sequences. SABLE uses machine learning to classify text. Text classification models can be used to predict whether a document contains useful data or to map scraped data to the Census Bureau's terminology and classification codes. The models developed for this task have also found applications beyond web scraping to the automation of classifying survey write-in responses.

Table 1, which is adapted from Dumbacher and Hanna (2017), summarizes the tasks performed by SABLE. Not all three tasks may be relevant to a given application. For example, data sources may already be determined, so it may not be necessary to perform web crawling. In this case, the problem would consist of just scraping and classifying data from known websites and documents.

Table 1. Three Main Tasks Performed by SABLE

Web Crawling
• Scan websites • Discover documents • Compile a training set of documents for building classification models
Web Scraping
• Find the useful data in a document using the frequencies and locations of important word sequences • Extract numerical values and contextual information such as data labels
Text Classification
• Predict whether a document contains useful data • Map scraped data to the Census Bureau's terminology and classification codes using data labels associated with the scraped data • Classify survey write-in responses

2.2 Machine Learning Methodology

Some SABLE applications use machine learning to fit text classification models and perform autocoding. Specifically, supervised learning is used to assign a class to a piece of text using a set of predictors, or features, and a training set of data (Hastie *et al.*, 2009, chap. 1). This training set contains classes that are assigned by hand and regarded as truth. Creating a large, representative, and good-quality training set is an important but manually intensive and time-consuming task. Text classification models for SABLE are based on features that are 0/1 variables indicating the presence of word sequences in the text. These word sequences are known as *n*-grams. Common so-called "stop" words such as articles and prepositions are removed from the text before creating features because they are not expected to be predictive of the class. Generally speaking, machine learning algorithms pick up on complicated patterns and associations between the presence of *n*-grams and classes. Some algorithms that we have tried include Naïve Bayes and support vector machines, which are mentioned in Section 3.1. To evaluate model performance, the fitted models can be applied to a separate test or validation dataset with classes that can be regarded as truth. For each observation in the test set, the predicted class can be compared to the true class.

Figure 1 illustrates fitting and evaluating text classification models in the context of predicting whether documents scraped from government websites contain useful data on tax revenue collections. For more details about this application and the machine learning methodology, see Section 3.1 and Dumbacher and Capps (2016). Also, for an excellent overview of classification concepts and model evaluation, see Tan, Steinbach, and Kumar (2006, chap. 4).

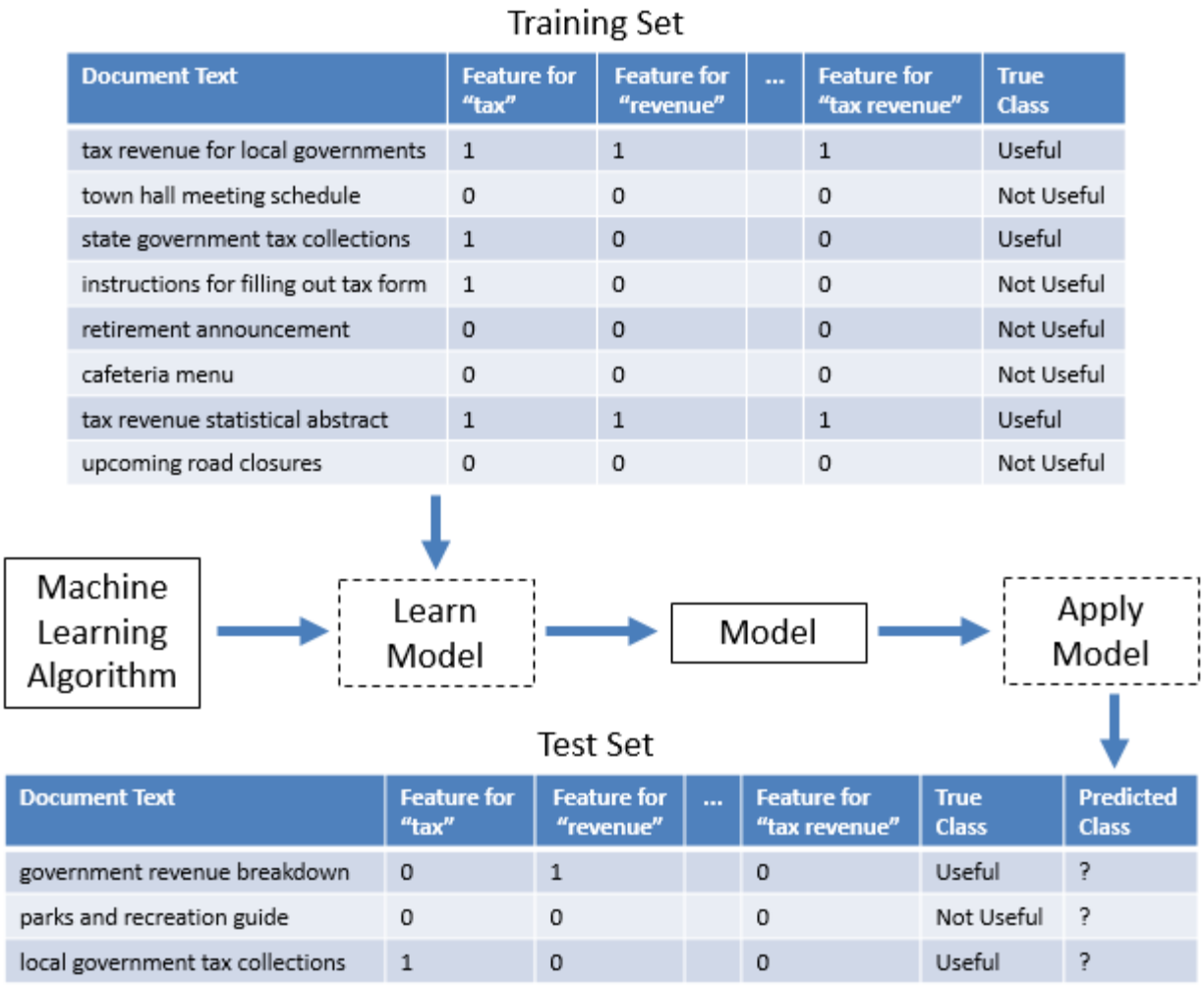

Figure 1. Illustration of machine learning process for fitting and evaluating text classification models. Based on Figure 4.3 from Tan, Steinbach, and Kumar (2006, p. 148).

2.3 Software

SABLE is based on two key pieces of open-source software: Apache Nutch, which is a Java-based web crawler (Apache, 2017), and Python. To run Nutch, one supplies a list of seed URLs, or starting points of the crawl, and sets parameters related to politeness and depth. Politeness refers to how frequently the web crawler jumps from one web page to another. Visiting pages too frequently can burden websites' servers. To avoid this, Nutch is able to incorporate a delay as it crawls. Websites provide politeness parameters such as this to web crawlers through a file called "robots.txt." Depth refers to how many levels of links to follow. A deeper crawl will map a website more extensively but will take longer to run. Nutch also has filters that one can apply to limit crawling to certain website domains and file types. Nutch first visits the seed URLs and then iteratively follows links down to the specified depth, effectively indexing the website. As Nutch crawls, it stores information about the pages and documents it comes across. This information includes date and time stamps and whether links are duplicates, are broken, or redirect to other URLs.

Python is a popular programming language for Big Data and data science applications. SABLE uses Python to scrape text and data from documents, process the scraped data, perform text analysis, and fit and evaluate classification models. There are three main Python modules: scikit-learn, the Natural Language Toolkit (NLTK), and PDFMiner. Scikit-learn is a commonly used machine learning module with many options for classification (Pedregosa *et al.*, 2011). NLTK is used to process and analyze text and also has some machine learning capability (Bird, 2006). The NLTK and scikit-learn modules have complementary features that make it easy to fit classification models for text. Lastly, PDFMiner converts PDFs to TXT format and is used in many SABLE applications (Shinyama, 2013).

2.4 Architecture Design

The architecture design for SABLE is fairly simple. Figure 2 illustrates this design. SABLE resides on a Linux server behind the Census Bureau's firewall and crawls and scrapes data from external public websites. Apache Nutch is self-contained and consists of the application itself, parameter files for customizing crawls, and directories for storing crawl results. The Python programs are located in a separate folder. Supplementary files consist of lists of common "stop" words that are useful for text analysis. For some problems involving PDF-to-TXT conversion and the classification of entire documents, additional folders are used to organize documents according to file format and class.

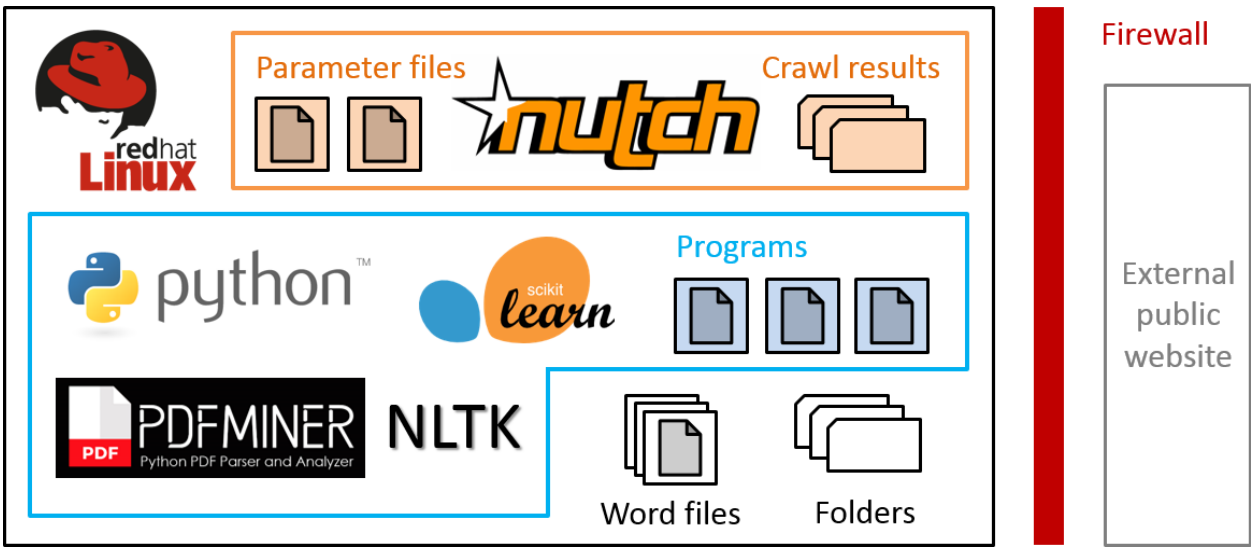

Figure 2. SABLE architecture design. SABLE resides on a Linux server behind the Census Bureau's firewall and crawls and scrapes data from external public websites. Apache Nutch and Python are the two key pieces of software.

3. Applications

3.1 Quarterly Summary of State and Local Government Tax Revenue

The first application of SABLE was to the Quarterly Summary of State and Local Government Tax Revenue (QTax). QTax is a survey of state and local governments that collects data on tax revenue collections such as general sales and gross receipts tax, individual income tax, and corporate net income tax. As with other public sector surveys, much of this data is publicly available on government websites. In fact, instead of responding via questionnaire, some respondents direct QTax analysts to their websites to collect data. State and local governments publish CAFRs and statistical reports, most of which are in PDF format.

As detailed in Dumbacher and Capps (2016), we used SABLE to crawl state government websites, discover potential new sources of tax revenue information, and build a classification model for predicting whether a PDF contains useful data. To do so, we first created a list of seed URLs of home pages of state government departments of revenue, taxation, and finance. We used Nutch to crawl these websites to a depth of three and discovered approximately 60,000 PDFs. To create a training set for use with machine learning, we first selected a random sample of 6,000 PDFs, where the sample size was chosen based on an estimate of how long it would take to classify the PDFs manually. Then we applied a PDF-to-TXT conversion algorithm based on the PDFMiner module to extract text and put it in the simple format of a single string of words separated by spaces. The text in this format could then be used as input to classification models. About 1,000 PDFs could not be converted to TXT format for various reasons. For the approximately 5,000 PDFs that could be converted, we manually classified them as positive (contains useful data on tax revenue collections) or negative. Lastly, these 5,000 PDFs were randomly divided into training and test sets.

Naïve Bayes and support vector machine models using various sets of features were fit on the training set and evaluated on the test set. The support vector machine using features based on 1-grams and 2-grams performed very well with an accuracy of 98 percent and an F_1 score of 0.89, which is a measure that balances recall and precision (Tan, Steinbach, and Kumar, 2006, p. 297). Such a model could be used to classify future PDFs discovered through more extensive web crawling.

3.2 Annual Survey of Public Pensions

Another public sector survey is the Annual Survey of Public Pensions (ASPP), which collects data on revenues, expenditures, financial assets, and membership information for defined benefit public pension funds administered by state and local governments. As with QTax, much of this information can be found online and in CAFRs. There is interest in examining the feasibility of scraping specialized content not currently collected in ASPP from the CAFRs of the largest state- and local-administered pension plans. The main pension statistics are service cost and interest. Figure 3 is a screenshot from the CAFR of the Santa Barbara County Employees' Retirement System showing pension statistics for fiscal years ended June 30, 2014-2016. In general, there is no standardization in CAFRs across governments, but the pension terminology is fairly consistent across government entities and throughout time.

We are currently considering a two-stage approach to scraping service cost and interest. After converting the CAFRs from PDF to TXT format, we use models based on the location of important word sequences to identify tables containing the pension statistics. For example, the phrases "required supplementary information" and "changes in net pension liability" tend to indicate the beginnings of tables, whereas the phrases "service cost" and "differences between expected and actual experience" indicate table content. In the second stage, we parse the identified tables and use regular expressions to scrape service cost and interest data. At the same time, we try to scrape information on what units the figures are in (for example, dollars or thousands of dollars), the names of the pension funds, and the corresponding time period. It is challenging dealing with tables that have complicated structures. It may make sense to group the tables according to structure and build a separate scraping model for each structure type.

REQUIRED SUPPLEMENTARY INFORMATION – PENSION

CHANGES IN NET PENSION LIABILITY

	Fiscal Year Ended		
	2016	2015	2014
Total pension liability			
Service Cost (MOY)	$ 71,218,683	$ 70,056,133	$ 66,696,324
Interest (includes interest on service cost)	241,733,937	231,804,221	220,238,560
Differences between expected & actual experience	(31,199,454)	(27,900,755)	-
Benefit payments, including refunds of member contributions	(146,657,716)	(137,771,219)	(131,100,585)
Net change in total pension liability	135,095,450	136,188,380	155,834,299
Total pension liability - beginning	3,260,156,781	3,123,968,401	2,968,134,102
Total pension liability - ending	3,395,252,231	3,260,156,781	3,123,968,401

Figure 3. Screenshot from the CAFR of the Santa Barbara County Employees' Retirement System showing pension statistics for fiscal years ended June 30, 2014-2016. The two main items are service cost and interest. Source: http://cosb.countyofsb.org/uploadedFiles/sbcers/benefits/SBCERS 6-30-2016 CAFR With Letters.pdf

3.3 Securities Exchange Commission Filing Metadata

The EDGAR database on the SEC website contains financial filing information for publicly traded companies. EDGAR is used often by Census Bureau analysts to impute missing values and validate responses for many economic surveys. In particular, the 10-K and 10-Q reports provide valuable annual and quarterly information, respectively. For the most part, going into EDGAR or visiting company websites to find out when new 10-K and 10-Q reports are available is a manual process. Ideally, analysts would be notified when new filings and data become available.

To this end, we recently started using Python to scrape filing metadata from the EDGAR database. Every company in EDGAR has a Really Simple Syndication (RSS) feed, which can be queried to obtain recent filing information. In order to query this RSS feed, one needs to supply the desired filing type (for example, 10-K or 10-Q) and the Central Index Key (CIK) of the company, which is a unique filer identifier used in EDGAR. We wrote a Python script that uses the Beautiful Soup module (Crummy, 2017) to submit a query to the feed and fetch results in XML format. Figure 4 shows part of an XML file created using this method. It contains recent 10-Q filings for Apple Computer, Inc. (CIK = 0000320193). Because the XML file is structured and based on standardized tags, it can be parsed easily using regular expressions or methods within Beautiful Soup to scrape filing dates and, in turn, determine whether a filing was made recently. Other pieces of useful information contained in the XML file include the URL to the corresponding report and an indicator for whether the filing is an amended version. The next step is to work with various survey teams to see how they can best use this information and incorporate web scraping and a filing notification process into their production cycles.

```
- <entry>
    <category term="10-Q" scheme="http://www.sec.gov/" label="form type"/>
  - <content type="text/xml">
      <accession-nunber> 0000320193-17-000009 </accession-nunber>
      <act> 34 </act>
      <file-number> 001-36743 </file-number>
      <file-number-href> http://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&filenum=001-
        36743&owner=exclude&count=100 </file-number-href>
      <filing-date> 2017-08-02 </filing-date>
      <filing-href> http://www.sec.gov/Archives/edgar/data/320193/000032019317000009/0000320193-17-000009-index.htm
        </filing-href>
      <filing-type> 10-Q </filing-type>
      <film-number> 171000359 </film-number>
      <form-name> Quarterly report [Sections 13 or 15(d)] </form-name>
      <size> 10 MB </size>
      <xbrl_href> http://www.sec.gov/cgi-bin/viewer?action=view&cik=320193&accession_number=0000320193-17-
        000009&xbrl_type=v </xbrl_href>
    </content>
    <id> urn:tag:sec.gov,2008:accession-number=0000320193-17-000009 </id>
    <link type="text/html" rel="alternate"
      href="http://www.sec.gov/Archives/edgar/data/320193/000032019317000009/0000320193-17-000009-index.htm"/>
    <summary type="html"> <b>Filed:</b> 2017-08-02 <b>AccNo:</b> 0000320193-17-000009 <b>Size:</b> 10 MB </summary>
    <title> 10-Q - Quarterly report [Sections 13 or 15(d)] </title>
    <updated> 2017-08-02T16:31:28-04:00 </updated>
  </entry>
```


Figure 4. Screenshot of an XML file containing information on recent 10-Q filings for Apple Computer, Inc. (CIK = 0000320193). This XML file was created in Python using information scraped from the company's RSS feed on EDGAR.

3.4 Economic Census Write-ins

The Census Bureau classifies business establishments according to the North American Industry Classification System (NAICS). NAICS groups establishments into industries based on the activities in which they are primarily engaged and where revenue is generated. For more information about NAICS, see U.S. Census Bureau (2017b). To assign NAICS codes to business establishments, the Census Bureau uses information from different sources such as the Economic Census, the Internal Revenue Service (IRS), and the Social Security Administration. Aspects of NAICS coding can be manually intensive, and efficiency can be improved through autocoding. Kornbau (2016, sec. 2) and Kearney and Kornbau (2005) describe a NAICS autocoder that was developed to assign NAICS codes to new businesses.

Another application of SABLE involves developing a NAICS autocoder for write-in responses to the Economic Census. The self-designated kind of business (SDKB) question on the Economic Census form asks respondents to describe their business and gives the respondent the option of writing in a description if it does not appear on a checklist. The machine learning methodology and text classification models used in SABLE are being applied to this setting to examine the feasibility of assigning a NAICS code to an establishment based on the business description and other information such as the business name. The plan is to use the hundreds of thousands of SDKB write-ins from the 2002, 2007, and 2012 Economic Census and business descriptions from the IRS's SS-4 form as a training set to build and evaluate classification models [the SS-4 form is used by businesses to apply for an Employer Identification Number].

4. Moving to a Production Environment

4.1 Research Environment

Much of the initial research for SABLE was done in the Census Bureau's Center for Applied Technology (CAT), which is a sandbox-like environment for collaboration and innovation. Workstations in the CAT are not connected to the Census Bureau network, so users are able to experiment with software not currently approved for use Census Bureau-wide. The CAT is a great place to develop proofs-of-concept and showcase successful efforts, which can be used to support a business case for moving projects into production. Confidential data are not allowed in the CAT. This did not pose a problem for SABLE because the web scraping and text classification problems at the time dealt with publicly available data from state and local government websites.

We had access to a single Linux server in the CAT and were able to install and experiment with various Python modules for web scraping and machine learning. During the summer of 2017, we obtained access to the CAT's new cloud environment, which is an Amazon Web Services (AWS) instance. We installed Apache Nutch in this environment and in July 2017 successfully crawled the Alabama Department of Revenue website as a test. Apache Nutch is designed to take advantage of the parallel processing that AWS offers, so we might use the cloud environment to explore this in the future.

4.2 Authority to Operate

After doing further testing in the CAT and demonstrating SABLE's usefulness, it was time to move from a research environment to a production environment. One of the first steps in the process was obtaining approval from the Census Bureau's Standards Working Group (SWG) to use Apache Nutch outside of the CAT. In July 2017, we defended our request before the SWG and answered questions related to software requirements and the current availability of software that performs similar functions. Apache Nutch 1.13, the most recent version at the time, was approved by the SWG. Later in the year, we were also able to obtain two new Linux servers for SABLE, one for development and another for production.

In consultation with the Census Bureau's Economic Applications Division and Office of Information Security, it was determined that SABLE needs an Authority to Operate (ATO). The ATO process involves SABLE undergoing a risk profile to determine what security controls are needed and a later security assessment to determine whether those controls are being met. In preparation for the assessment, we need to establish evidence for satisfying the controls, write documentation, and develop procedures for tasks such as making change requests, managing code, and auditing users. The assessment is expected to take between six and ten weeks after submitting evidence.

4.3 GitHub Repository

To share our work with the public, other government agencies, and interested private companies, we had some SABLE files uploaded to a new repository on the Census Bureau's GitHub account in October 2017. This repository is located at the following URL: https://www.github.com/uscensusbureau/SABLE. It currently contains two Python programs, one for converting PDFs to TXT format and another for fitting and evaluating basic text classification models such as Naïve Bayes, logistic regression, and decision trees. There are also supplementary files of stop words in various languages and some documentation files.

GitHub users can comment on our work, propose changes to the code, and even copy, or "fork," the repository to their own account so they can edit the code themselves and take the project in their own direction. To get these files onto GitHub, we had to obtain approval from Census Bureau information technology officials and submit the code to an internal Python review process that checks for things such as references to servers, unused imported modules, and proper exception handling. We plan to update the files on GitHub and release new features periodically.

5. Next Steps

5.1 Quality Assurance

With SABLE going through the ATO process and into production, we are thinking about how to integrate quality into the system early on and establish procedures for assessing quality on a regular basis. The following are some ideas for quality assurance. Regarding web crawling, the URLs and content of websites change frequently, and it would be important to re-crawl websites to make sure the most up-to-date pages and documents are being discovered. Perhaps subject matter experts could conduct manual crawls of certain sections of websites and check whether Nutch is discovering important documents. These manual crawls could also inform how deep to crawl.

In assessing data scraped to supplement or replace current survey collections, we could check relationships such as current year data to prior year data, how the scraped data compare to data for respondents within the same sampling stratum, and outliers. Static bounds could be set for current year versus prior year comparisons as well as within-stratum comparisons. This should be done every survey statistical period with subject matter experts investigating any data points that do not meet the criteria. In general, there should be some sort of basic quality and reliability check run for each statistical period for a given survey, with a more thorough analysis done annually. The analysis would include checking the bounds mentioned previously and, as with web crawling, manual scraping to check whether SABLE is scraping the desired data properly.

In terms of assessing the quality of machine learning methods, it is also imperative to get analysts and other subject matter experts involved. We are considering best practices for having subject matter experts help create good-quality training sets and assess the quality of classification model predictions in ways other than accuracy, precision, recall, and other common evaluation criteria. For example, in classification problems, not all classes are equally important. There are different misclassification costs that analysts could help identify and quantify.

5.2 Future Work

The ATO process is ongoing, and we will continue working to ensure SABLE functions properly and meets the security controls identified in the risk profile. As mentioned previously, we would like to update the SABLE files on the Census Bureau's GitHub account periodically. Ultimately, the goal is to use SABLE to create a data product based on scraped data. The public sector applications seem like good candidates for such a data product.

In terms of future applications, there is a project very similar to the NAICS autocoding project. This one involves the North American Product Classification System (NAPCS), which is a hierarchical classification system of goods and services and complements NAICS. For more information about NAPCS, see U.S. Census Bureau (2017c). Respondents to the 2017 Economic Census are able to write in descriptions of their products. An autocoder that assigns NAPCS codes to product descriptions could be based partly on these training data and could improve efficiency greatly in future Economic Censuses.

Acknowledgments

The authors would like to thank Carma Hogue, Andrew Baer, and Stephen Kaputa of the U.S. Census Bureau for reviewing this paper and providing helpful comments. Thanks also to Carol Caldwell, Anne McGaughey, Charles Stockton, Douglas Peed, Andrea Roberson, Selvaratnam Sridharma and the staff of the Census Bureau's Center for Applied Technology for their contributions and help regarding SABLE.

References

The Apache Software Foundation. (2017). Apache Nutch. <http://nutch.apache.org>. Accessed December 1, 2017.

Bird, S. (2006). NLTK: The Natural Language Toolkit. *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*. Sydney, Australia: Association for Computational Linguistics, 69–72.

Crummy. (2017). Beautiful Soup. <https://www.crummy.com/software/BeautifulSoup/>. Accessed December 14, 2017.

Dumbacher, B. and Capps, C. (2016). Big Data Methods for Scraping Government Tax Revenue from the Web. *2016 Proceedings of the American Statistical Association, Section on Statistical Learning and Data Science*. Alexandria, VA: American Statistical Association, 2940–2954.

Dumbacher, B. and Hanna, D. (2017). Using Passive Data Collection, System-to-System Data Collection, and Machine Learning to Improve Economic Surveys. *2017 Proceedings of the American Statistical Association, Business and Economic Statistics Section*. Alexandria, VA: American Statistical Association, 772–785.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Second Edition). Berlin, Germany: Springer.

Hurwitz, J., Nugent, A., Halper, F., and Kaufman, M. (2013). *Big Data for Dummies*. Hoboken, NJ: John Wiley & Sons, Inc.

Kearney, A.T. and Kornbau, M.E. (2005). An Automated Industry Coding Application for New U.S. Business Establishments. *2005 Proceedings of the American Statistical Association, Business and Economic Statistics Section*. Alexandria, VA: American Statistical Association, 867–874.

Kornbau, M.E. (2016). Automating Processes for the U.S. Census Business Register. *25th Meeting of the Wiesbaden Group on Business Registers*.

Mitchell, R. (2015). *Web Scraping with Python: Collecting Data from the Modern Web*. Sebastopol, CA: O'Reilly Media, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Shinyama, Y. (2013). PDFMiner. <http://www.unixuser.org/~euske/python/pdfminer/index.html>. Accessed December 12, 2017.

Snijkers, G., Haraldsen, G., Jones, J., and Willimack, D.K. (2013). *Designing and Conducting Business Surveys*. Hoboken, NJ: John Wiley & Sons, Inc.
Tan, P.N., Steinbach, M., and Kumar, V. (2006). *Introduction to Data Mining*. New York, NY: Pearson.
U.S. Census Bureau. (2017a). Federal, State, & Local Governments. <https://www.census.gov/govs/classification/>. Accessed December 1, 2017.
U.S. Census Bureau. (2017b). North American Industry Classification System. <https://www.census.gov/eos/www/naics/>. Accessed December 1, 2017.
U.S. Census Bureau. (2017c). North American Product Classification System. <https://www.census.gov/eos/www/napcs/>. Accessed December 1, 2017.