Technology Assisted Review with Iterative Classification

Corbin Faidley, Robert Robinson, and Stephen Hughes
Computer Science
Coe College
Cedar Rapids, IA 52402
{cafaidley, rrrobinson, shughes}@coe.edu

Abstract

Manually extracting information from large text-oriented datasets can be impractical, and language variations in the text can defy simple keyword-driven searches. This suggests the need for an intermediate approach that would allow for a technology assisted review of the documents. This project explores the impact of keeping the human in the loop in the early stages of data mining efforts to help direct the outcome. The research team developed a tool to allow users to interactively classify individual pages of a large text document and feed the results into a learning algorithm. As the algorithm's predictive accuracy improves, it continually assumes a more informed role in selecting text segments for the human to evaluate until it is reliable enough to finish the classification task unsupervised. We expect that the incremental training approach can efficiently identify a core set of data, and believe it shows promise in this kind of application.

1. Motivation & Background

Advances in computing power have given rise to numerous data mining techniques which allow analysts to filter and reorganize data to reveal latent (and often unintuitive) relationships that exist within very large datasets. In other words, using these techniques, we are able to find the answer to questions that we did not think (or know how) to explicitly ask. Many of these techniques build an initial model and then rely on the computer's power to iteratively refine it millions of times until patterns emerge. As the model evolves, outlying data is identified and minimized, reliable differences are magnified, and similarities are reinforced. Finally, after numerous revisions of the original model, a better understanding of the data is shared with the user.

With most traditional data mining techniques, the user first provides an initial set of parameters, then leaves the algorithm to work uninterrupted until a convergent model emerges. However, if the initial set of parameters is not clearly defined (or well understood), the results may not align with the user's information needs. Moreover, the human's target knowledge may shift as they become more acquainted with the corpus of knowledge. Perhaps they wish for their investigation to become broader – expanding their search criteria. Conversely, they may need to refine their criteria to be more selective. In a large collection of data leaving a model to converge may not be the best option. Keeping the the human in the loop can allow for a more flexible model to adapt to changes in the process.

This project proposed to explore the possibility of keeping the human in the loop to mitigate these problems. By providing intermediate snapshots into the decision making process, users would be able to monitor the progress of the data mining, as well as intercede when necessary to help steer the calculations in new directions or act on new hypotheses. More broadly, we were hoping to understand the limits of automation with respect to these data mining techniques by asking the question, "Under what conditions does human judgment enhance the automated knowledge extraction process?"

This kind of research always looks to develop tools and methods that we can generalize, but the proof-of-concept must always be grounded in a practical application, which is informed by the dataset(s) selected. For this project, we elected to compile a new dataset that was aimed at understanding the contents of catalogs for colleges and universities in the United States. More specifically, we wished to collect information on the classes that each institution requires all of its students to take - its general education requirements.

In the area of college catalogs we searched in, we knew that all catalogs must contain certain materials, such as their general education requirements. In a few cases we have seen classifiers applied where the data is structured, but the answer is not known. With this research we wanted to see how a classifier would work when we have an unstructured data set, but the answer is known. A problem with this is that not all colleges

are easy to collect this information from, as the structures of these documents can vary from the simplest things to being radically different. So we wanted a machine learning algorithm (Naïve Bayes in this case) to collect similar features based off of certain criteria.

2. Data Acquisition

The first step of any data mining task is to first establish a working data set. In our case, the catalogs needed to be collected. The initial parameters for our data set included four year programs that offered B.S. and B.A. degrees. This excluded institutions that offered nursing or seminary preparation, as they were considered a different kind of institution; their requirements were expected to have great variation compared to the standard undergraduate programs within the data set collected. A list of the names of each of these institutions was acquired from the Integrated Postsecondary Education Data System, provided by the National Center for Education Statistics (n.d.), to use as the foundation for our data set.

We were confident that the necessary information would be available, but needed to determine what forms of catalogs we should collect. After some preemptive searches, we became aware of two different formats: PDF's and web-based catalogs. We found that the methods used to collect and parse PDF's were more algorithmic than those used for HTML-based catalogs. The websites quite often varied in many ways, making it difficult to both pinpoint and collect the desired information. Furthermore, based on a small sample set, it was concluded that most schools offered a PDF version of their catalog online. Therefore, PDF's were chosen.

A web-crawler was developed to collect the catalogs from 2,153 four-year, degree-granting schools to form the corpus of our data set through a simple Google search. We generated the following query for each college: "[college name] catalog filetype:pdf". By grabbing the first result of each search, the crawler was able to collect 2,097 pdf documents. However, not all of the returned PDF's contained the desired information. Non-catalogs were expunged from the data set through a combination of programmatic and manual filters. For example, some pdfs were less than 5 pages. These were typically “bulletins” instead of true catalogs. These were easily identified through a simple program that set a minimum page requirement for a catalog.

By sampling the results, it was determined that the crawler was about 70% effective, meaning that it was able to retrieve approximately 1,500 catalogs. Our goal was to create a dataset with enough data to perform interesting comparisons. While only 70% of the expected data was collected, we deemed this result sufficient to allow us to proceed to the next phase.

3. Data Cleaning

After enough catalogs were collected, the next step was to parse them in order to find the general education requirements. Although the catalogs we decided to use were all in a PDF format, they still provided various complications. First, the data in a PDF file contains many variations in formatting as different softwares use different methods. While PDF files may appear well-organized when viewed through PDF-viewing software, internally they are quite disorganized and nonuniform, as PDF's prioritize external aesthetic over internal structure. We relied on an external library (PyPDF2) that aided us in extracting the raw text from the PDF. This was much easier to parse programmatically, but was still not a perfect solution. The raw text often contained line breaks and spaces in improper locations. This made parsing more difficult, as it caused various words to be separated or incorrectly joined. Also, we were unable to collect images through the library we used, even though some may have contained valuable information. Overall, this part of the process allowed us to get the primary data (i.e. the raw text) into a format that we could programmatically work with. However, this came at the expense of some structure and image data.

To find the goal within the raw text would be a tremendous undertaking. A data mining program would likely take an especially long time to parse the entirety of each catalog in detail. A more efficient method would be to divide the text by pages in the PDF, then remove the majority, if not all, of the pages which are unrelated to our goal. The next goal, then, was to classify each page in each PDF document as being either related or unrelated to the desired information.

4. Data Reduction

At this stage, we were ready to direct the user to the data that they seek. Each catalog contains an average of 400 pages, of which probably 5-10 would contain relevant information about general education requirements. Clearly, we needed a mechanism to separate the desired pages from irrelevant pages. We identified four options for parsing the text: manually finding the desired data in the raw text, searching by keyword(s) to find the general whereabouts of the data, using a standard Naïve Bayes algorithm to guide our search, or using a continuous learning algorithm. While the former two are not directly related to machine learning, our primary reason for undertaking this research, they were useful for providing benchmark results against which our success rates with the other methods could be compared. Our hope was that incorporating the machine learning approaches we could significantly reduce the human's workload and improve the overall

performance. The hypothesized benefits of using these machine learning techniques are outlined in Figure 1.

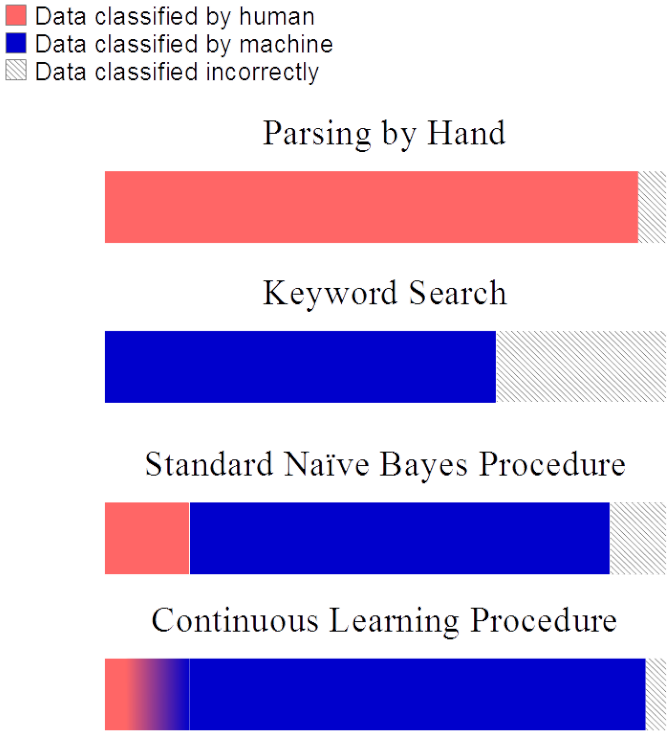

Figure 1: Projected Benefits of Technology Assisted Review

In classifying the data, it can be classified with either a hit, false alarm, correct reject, or miss. A hit or correctly rejected means the data was correctly identified as either a good or bad page. A false alarm is when the algorithm think it is a hit page when it is not. Then a miss is when the algorithm negatively identifies positive pages.

4.1 - Hand Parsing

The first method was parsing the catalogs by hand. Although hand parsing is the most time intensive of these methods in gathering data, the accuracy is reasonably high. Presumably, a human filter could quickly adapt to variations in structure, presentation or phrasing with the most flexibility. Moreover, they can utilize embedded navigational tools, such as tables of contents or indexes, that are designed to aid human readers. It is important to note that some hand parsing is required in order to prime the machine learning algorithms.

We manually opened each catalog in a PDF viewer, and set guidelines for what constituted general education requirements in order to maintain continuity while

determining pages. We parsed through 180 pdfs to get an accurate base set, which we planned to use as our benchmark for the project.

4.2 - Keyword search

The second method involved using a keyword search. Removing all pages of data which did not contain a particular keyword, such as "education", was expected to narrow the data down slightly. Optimally, the keyword would appear on all of the pages which contained our desired information. However, this was not guaranteed. While some institutions would use terms such as "general education requirements", other institutions may refer to the same information using terms such as "core curriculum" or "liberal arts core". A single keyword would not be able to find all of these pages, as a single keyword may not contain enough context.

To solve this problem requires a search with multiple keywords. By finding the pages containing any one of multiple keywords, such as "core" and "general", we could find many more of the pages described above. Such a search, as expected, would also return many more of the pages that are unrelated to our goal; Figure 2 demonstrate this issue. As more keywords are added, we find more occurrences of words in undesired pages. This is a central issue with any simple keyword search: the same word can have different uses depending on the context. While a specified keyword may appear on the desired pages, it is likely to appear on other pages as well. For example, the word "education" may refer to "general education" on one page, but may also be used in the context of a teaching degree on another page. Moreover, the human navigational cues, like indexes and tables of contents are likely to contain the keyword(s) but contain no actual content associated with those concepts. The search must take extra measures to ensure these pages are not included in the recommended output. Tables of contents and indexes could potentially be eliminated by virtue of being near the beginning and end of the document. However, other references can occur arbitrarily. For example, it may also be used on a page outlining the requirements of a specific degree; a page about the courses in the mathematics field may specify, that "this course satisfies the mathematical sciences general education requirement". Searching for multiple keywords exacerbates this problem: we find a few more of the desired pages, but more of the undesired pages, as well.

Another challenge with the multiple keyword search, is knowing exactly which terms to use. Many people can come up with some reasonable synonyms if pressed, but the familiarity of one's own vernacular can be a powerful barrier. For example, if you are part of a community that uses “general education”, the phrases “core curriculum”, “liberal arts core” or “liberal studies” may not be understood as useful search terms.

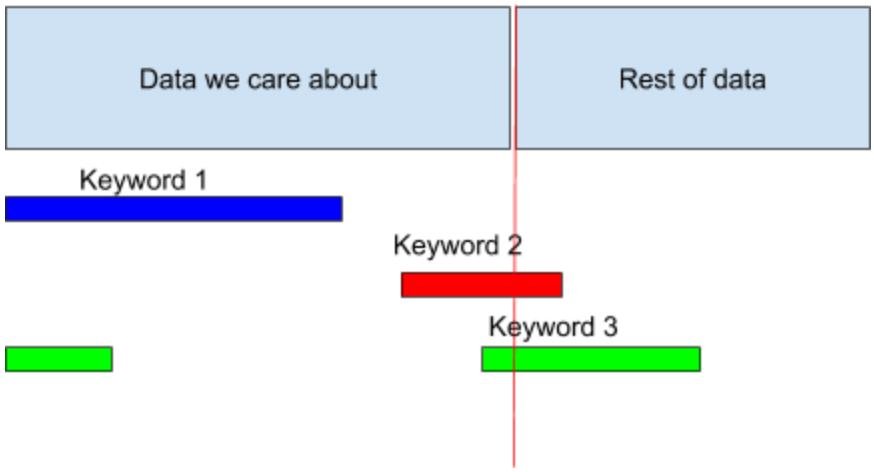

Figure 2: The Effect of Multiple Keywords.

One approach to the problem of context is the use of n-grams rather than keywords. N-grams are simply groups of n consecutive words. A 1-gram, or unigram, is a single keyword. A two-gram, or bigram, refers to two adjacent words. The phrase "general education requirements" contains two bigrams: "general education" and "education requirements." From the example above, we may be interested in the bigram, “general education”, but less interested in “education requirements” which would also include information about getting an education degree. Bigrams are potentially useful because they provide much more context than a word alone, and can thus be used for a more discerning search.

In deciding what sort of n-gram searches to use, we primarily considered bigrams and trigrams. We soon found that bigrams were most effective in finding relevant data. N-grams greater than two were perhaps too specific - an individual trigram would show up in only a few catalogs. The effectiveness of bigrams in particular is widely reported. Research done by Chade-Meng Tan, Yuan-Fang Wang, and Chan-Do Lee (2002) supports the claim that the use of bigrams results in much better performance, especially when compared to four-grams and greater.

The use of bigrams rather than unigrams showed a clear improvement in search accuracy, resulting in many more desired pages and fewer undesired pages. The system was still far from perfect, unfortunately. In order to improve on these results, we would need to use a data classification technique capable of making different types of distinctions between pages.

4.3 - Naïve Bayes

Naïve Bayes was considered as a machine learning approach that could help automate the process of classifying pages. Instead of relying on human judgement, like with hand

parsing, a model could be constructed to recognize key features. This approach can also remove the need for the user to explicitly provide keywords. Instead, attributes of desirable pages are inferred through their repeated presence, even if the user is not actively aware of that trend.

Like other classification algorithms, Naïve Bayes relies on being trained with a set of pre-classified data. A model is derived based on the recognition of certain relationships between features in the data . This model is tested on a separate set of pre-classified data to understand how accurate the predictions will be. If the algorithm is found to agree with the manual classifications within a reasonable threshold, it can be concluded that it is accurate enough to begin classifying the rest of the data.

Specifically the Naïve Bayes algorithm is given a set of user-defined features to focus on. During the training process the algorithm pulls items from these features and sets a weight to them based on how prevalent they are in the pre-classified data. During testing and classification, each item within a unit of data (in our case, pages) is compared to the list of weights. The more features the unit of data contains of higher weight, the higher score it gets. A higher score means that unit of data is considered to be similar to what is being sought after and therefore is marked more positively.

The effectiveness of applying this algorithm to college catalogs in search of general education requirements, depends on which features are included in the model. In choosing our features, several options were explored, including page number, bigrams, headings, and the table of contents. Our intuition for these attributes are outlined in Table 1. Observationally, using bi-gram based keywords seemed to consistently provide the best results in this dataset - although that could easily be different with another dataset or another domain. In effect, we are asking the Naïve Bayes algorithm to implicitly find keywords (or more accurately key bi-grams) for us, based on the pages that we have approved.

Feature	Rationalization
Page Number	Many “gen ed” requirements we found we closer to the beginning of the catalog. So we considered that if a page were to be within a certain place in relation to the catalog it could have a higher chance of being a page of interest.
Headings	By using large headings as we saw many colleges do in their catalogs, we thought it might be a good idea to take advantage of this. A problem with this was the formatting of PDF’s, as we found difficulties in extracting that formating.
*Key Words	Unigrams - words like “general” and “education" were too vague. Bigrams and Trigrams - showed the most promise as they provide enough context, but not too specific.
Linking to the Table of Contents	We thought by looking at the table of contents this might help us find where we needed to go, but without context and various words this method seemed to cause more problems than we deemed necessary.

Table 1: Some of the features considered using with Naïve Bayes.

It is important to note that when feeding the algorithm the training set, both wanted and unwanted pages must be supplied. This allows Naïve Bayes to not only understand what to look for, but also what to not look for. Searching only for positives can lead the algorithm to falsely identify more pages, as it might be unaware of certain intricacies that might cause a piece of data to differ ever so slightly. Therefore, it is necessary for some features to be chosen which appear in data with one classification, but not in data with the other - and vice versa.

As stated earlier, our target pages comprise approximately 1% of the total number of pages. There are more than enough negative examples to draw from. A heuristic that we adopted was to include some negative pages that were adjacent to relevant pages as well as some additional random negative pages. The rationale for this approach was to attempt to include a mix of some pages that would be conceptually close to the target with some pages that were representative of the general vocabulary used in this type of document.

4.4 - Continuous Learning

A different approach for classifiers is to add continuous learning. A frequently given example is a very simple Spam filter; it can determine the probability of an email being

classified as spam based on the presence or absence of several keywords. Likewise, we attempted to apply this technique to determine the probability that a page from the catalog was related to our search term. A slight advantage a spam filter has over our catalog minner, is it usually primed with as a starting point, where the catalog minner is starting from scratch, without predefined examples.

The Naïve Bayes algorithm requires a rather large collection of examples that have been pre-classified as either positive or negative to "train" the classifier. Unlike the spam example, this classifier is unique to our current search, so building the training set is doing the work that we hoped to automate. Additionally, this is an example where the initial training set can have a profound effect on the outcome; depending on the composition of the sample, the user may not be able to reliably interpret the outcome.

To overcome these problems we proposed and implemented a modification to the Naïve Bayes algorithm. Instead of providing the training data as an initial burst, we would allow the user to interactively classify individual pages and feed the results – in real time – to the algorithm. The algorithm would use the information (as it receives it) to attempt to classify remaining pages. As interaction with the algorithm progressed, it would share different information with the viewer. Initially, it would only share pages that contain the search terms. Next, it would attempt to confirm what it deemed as "high-probability" examples. Finally, it would try to consult the viewer with boundary cases. This variable threshold approach offers an efficient way for the automation and the human participant to share the burden of classifying the pages of the document. The user is relieved of repetitive assessment, only being called in on the cases for which the classifier does not have a strong judgement.

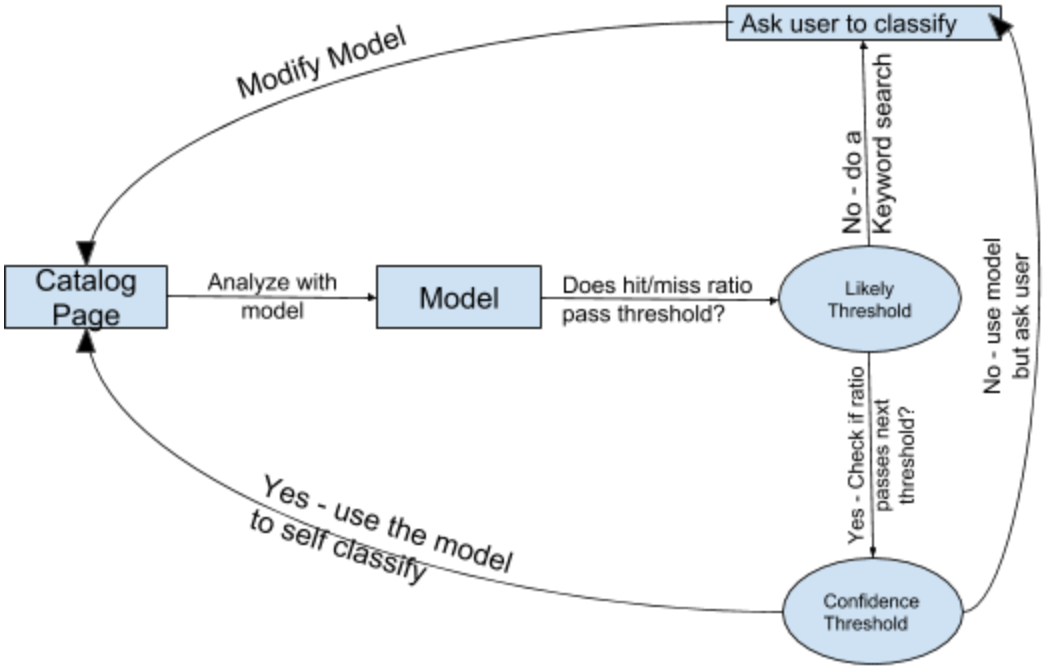

Figure 3: A chart describing the continuous learning process.

When initially identifying catalogs with a keyword, false alarms and misses will naturally occur at a high rate, as keywords can be on the table of contents, index, or other locations. Then the keyword may not always be on pages that the requirements are on. While the program is using a keyword search to find the requirements, in the background, a Naïve Bayes program is being used to analyze and create a model. Once the false alarms and misses are reduced to a certain level the program would start giving suggestions based on this model. Now, with the program giving suggestions from the model. In the background the program is still trying to analyze and improve the model, as the human is still involved in the process. As this hit to miss ratio continues to decrease there will be a point where the program will begin to solely rely on the model. Both this threshold and where the model shifts from keyword search remain a question to us.

5 - Conclusion

In developing a prototype we were able to see a few of these strategies in action and remain optimistic about a) ML approach over keyword/manual b) CAL over NB. In fine tuning these prototypes we have not yet evaluated these relationships. We have observed that the algorithms/approaches have variability and can be tuned.

1. Here are some of the areas that we think we can tune:

- What techniques would allow us to parse web-based catalogs, rather than just those in a PDF format?
- Could other approaches to filtering out irrelevant documents, such as machine learning approaches, prove more successful?
- At which thresholds do you allow a continuous learning program to send information to the user based on the model, and when is the model good enough to classify the data without the human?

2. Here are some questions that we think someone implementing this approach will need to consider.

- Which ways would be most effective in mining data from unstructured formats?
- Would other Machine Learning algorithms perform better on this data set than Naïve Bayes?
- Would this model be transferable to other similar applications?

References

U.S. Department of Education. Institute of Education Sciences, National Center for Education Statistics. (n.d.). *The Integrated Postsecondary Education Data System.* Retrieved March 16, 2018, from https://nces.ed.gov/ipeds/

Jiang, Y. G., Yang, J., Ngo, C. W., & Hauptmann, A. G. (2010). Representations of keypoint-based semantic concept detection: A comprehensive study. *IEEE Transactions on Multimedia*, *12*(1), 42-53.

Home page for the PyPDF2 project. (n.d.). Retrieved March 16, 2018, from http://mstamy2.github.io/PyPDF2/