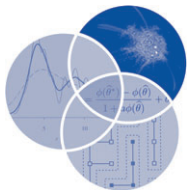
Advanced Review



A practical guide to text mining with topic extraction

Andrew Karl, James Wisnowski* and W. Heath Rushing

Text analytics continue to proliferate as mass volumes of unstructured but highly useful data are generated at unbounded rates. Vector space models for text data—in which documents are represented by rows and words by columns—provide a translation of this unstructured data into a format that may be analyzed with statistical and machine learning techniques. This approach gives excellent results in revealing common themes, clustering documents, clustering words, and in translating unstructured text fields (such as an open-ended survey response) to usable input variables for predictive modeling. After discussing the collection and processing of text, we explore properties and transformations of the document-term matrix (DTM). We show how the singular value decomposition may be used to drastically reduce the size of the document space while also setting the stage for automatic topic extraction, courtesy of the varimax rotation. This latent semantic analysis (LSA) approach produces factors that are compatible with graphical exploration and advanced analytics. We also explore Latent Dirichlet Allocation for topic analysis. We reference published R packages to implement the methods and conclude with a summary of other popular open-source and commercial software packages.

How to cite this article:
WIREs Comput Stat 2015, 7:326–340. doi: 10.1002/wics.1361

Keywords: text mining; text analytics; singular value decomposition; latent semantic; analysis; unstructured data; Latent Dirichlet Allocation

INTRODUCTION

Organizations and individuals are increasingly finding themselves with large collections of digital text documents. They expend significant resources to collect and store this data yet most fail to take advantage of the rich information contained through analytics. Typical questions from such a large collection of text (a corpus) include:

- Can the corpus be split into clusters of similar documents?
- What are the major themes in the corpus?
- Do specific words group together?
- What words are closely related to a specific word?
- Can the information present in the documents be used to predict a numeric response?
- Can the documents be classified into existing categories?
- Do the documents take a positive or a negative tone?

Through the process of text mining, answers to these questions may be derived using existing statistical and machine learning methods. The general flow of text mining in this article is:

1. Define the problem statement to be solved succinctly.
2. Collect the appropriate text and structured data.
3. Process and filter the text by removing words/characters such as misspellings, common

*Correspondence to: james.wisnowski@adsurgo.com
Adsurgo LLC, Denver, CO, USA

Conflict of interest: The authors have declared no conflicts of interest for this article.

words (stopwords), extraneous words inherited from text collection processing, rare words occurring in a single document or two, words that are too short or too long, numbers, and nonstandard characters. Additionally, it is sometimes useful to change synonyms to a single representative word.

4. Transform the text into an appropriately weighted matrix to allow for statistical analyses.
5. Explore and discover topics and common themes.
6. Group similar documents and words.
7. Create new structured variables from the text to use in predictive analytics.

The key to translating text to numerical values is to transform the corpus into a document-term matrix (DTM), with one row for each document (page, paragraph, record, file, etc.) and one column for each word that appears throughout the entire corpus. This DTM vector space model allows the corpus to be analyzed using existing data-mining methods (with some modifications), treating documents as observations and words as variables. Many text mining methods use the transpose of the DTM, called a term-document matrix. However, the DTM is better suited for our analytical approach as it matches the expected format of variables-as-columns and observations-as-rows. A major challenge for text mining problems is that the DTM is often extremely large. Fortunately the DTM is also sparse due to the relatively infrequent occurrence of most words across a single row. We can use widely available sparse matrix algorithms to efficiently store and manipulate the DTM. One such useful DTM manipulation is the singular value decomposition (SVD), which provides us with both eigenvectors representing the document space and eigenvectors representing the word-space. Latent semantic analysis (LSA) uses these factors from a reduced-rank singular value decomposition of the DTM to discover useful relationships.

The commonly used 'bag-of-words' approach assumes the order that the words in a document appear in, as well as their parts of speech and other syntax, may be ignored. This approach is extremely effective at extracting patterns from the corpus (see e.g., Refs 1–4). This ignoring of context, while seeming extreme, has actually produced surprisingly good results for the questions raised above. However, depending on the research objective, it may be useful to include the complexity to account for the context. Many software packages (e.g., SAS Text Miner, OpenNLP[5]) provide functionality to account for word order and tag parts of speech.

We will focus our discussion on LSA using the 'bag-of-words' approach. We use a running example of 9466 abstracts comprising the 1990 National Science Foundation (NSF) grant awards, obtained from the University of California Irvine Machine Learning Repository.[6] The abstracts have a median length of 154 words and cover a wide range of topics from life sciences, engineering, mathematics, geology, oceanography, and numerous other scientific fields.

COLLECTING AND PROCESSING TEXT

After clearly defining the problem that the study is meant to solve, the next step in a text mining progression is to determine what the appropriate sources of unstructured text are and how they need to be extracted. Ideally, the data may be available within the enterprise and require only minor formatting and queries to extract the appropriate fields, such as free-form survey comments. More often, it may be necessary to assemble text from disparate sources, such as word or pdf documents and text from websites. Speech-to-text is also a common source of text data though it is often accompanied by high rates of data translation errors. For new users who want practice with text analysis without needing to build their own corpus, the book by Miner et al.[1] provides several processed example corpa along with introductions to the text mining functionality of several software packages.

Extracting Text from Files

Text stored locally on a computer tends to be in Word documents or PDF files. While many text mining packages include functionality for converting text from these files, it is sometimes necessary to do so manually in order to process the text before sending it to the text mining software. Owing to the nature in which text is embedded in PDF files, text may run together when converting these files. Programs such as Adobe Acrobat Professional that have the ability to perform optical character recognition (OCR) tend to do a better job of converting these files. While OCR is most commonly used to extract text from scanned documents, it is also useful when converting a digital PDF that was created with nonstandard encoding. Alternatively, PDFBox is an open-source library that has been shown to have good performance.[3] Apache Tika (https://tika.apache.org) can extract many common file types and can autodetect the content followed by parsing with the appropriate library.

Web and Twitter Crawling

The Internet provides a constant source of new text. Some companies have found that web crawling for material relating to their products can provide faster feedback than would be available through more traditional channels, such as customer surveys or focus groups. *Scrapy*[7] is a popular open-source web crawling framework. It is also possible to build a webcrawler in R around the *RCurl* package.[8] A challenge with the analysis of web text is extracting the relevant text from a larger amount of embedded html code. This process can be challenging to automate due to variations in how html sites may be constructed. However, this extra structure also provides some benefits. Metadata such as the page, date, class, and title may be clearly demarcated. In an analysis of a large news website, it may be sufficient to use the page titles rather than the full text of each article.

Twitter data is a special subset of web data that requires minimal processing, as it is not encapsulated in HTML. There are various application programming interfaces (APIs) for connecting to Twitter to extract data. The R package *streamR*[9] makes use of the Twitter Public Stream API, allowing real-time monitoring of Twitter for a search phrase for a user-specified length of time. Tweets are typically informal with many 'special' ways to say the same thing which likely will require some additional cleaning prior to analytics.

Preprocessing Text

Text from different sources and formats must be processed to remove extraneous details (code, comments, etc) and converted into a plain text format. This step also involves transcoding all of the text into a common encoding: many of the R text mining functions expect characters that can be encoded as UTF-8. Additionally, it may be necessary to remove a common header (such as from an email server) from a collection of messages before they are processed. Though there are many software packages capable of doing this preprocessing step, we have found the use of regular expressions through the *grep* function in R or Perl[10] provides reliable results.

Once the relevant text has been extracted and represented in a plain text format, the words themselves need to be processed. Punctuation is usually separated from adjacent words, and the case must be normalized, typically to lowercase. You will need to carefully consider whether removing numbers or not as they could provide context to problems or may be an essential part of the corpus. For example, social media analytics would want to include numbers due to the proliferation of 'slang' terms (e.g., be4 as one representation of before).

Words longer than 30 characters are usually removed: nearly all English words consist of fewer than 30 letters. Anything longer is likely to be a web URL or a string of garbled text (common when converting PDF to text). As a word may take different forms depending on its grammatical use, it is typically useful to stem the text.[11] Stemming effectively chops the endings off of words. Plural and singular nouns are reduced to the same token, and conjugated verbs are also mapped into a single representative word.

The frequency of terms in a corpus (the *dictionary* or *lexicon*) will be roughly proportional to its rank, a behavior described by Zipf's law.[12] This has a few important practical implications. A small collection of words will occur so frequently (and likely in so many documents) that they do not give any discriminating power. These are referred to as stopwords. Software packages contain a list of common stopwords that may be automatically removed and an option to supply them. A large proportion of infrequent words will appear in so few documents that they are not useful when searching for the dominating patterns in a corpus. When clustering documents or searching for major themes, a best practice is to remove terms that appear in fewer than 1% of the documents in the corpus. Even when the goal of the text mining analysis is prediction or classification, it is generally safe to remove words that appear in only a single document. Typically, there will be a handful of medium frequency words that provide the most flexibility in differentiating between documents.

Figure 1 shows the frequency of the top 100 words in the NSF corpus produced without stemming or removing stopwords. The top three words are 'the', 'of', and 'and'. From Figure 2, over half of the terms in the corpus appear in only one document and are therefore not useful to detect patterns in the corpus. Removing these noise terms speeds up the subsequent analysis of the DTM and improves the quality of the analysis.

REPRESENTATION OF TEXT IN A DTM

The DTM will typically be sparse (most entries are 0). Even for modestly sized applications, the full DTM will be too large to hold in memory if represented as a dense matrix. Special software and algorithms are available for storing and manipulating sparse matrices.[13] In the NSF abstract example, 99.8% of the 44.5 million components of the DTM are 0. The sparse representation requires 18 MB to store, whereas the dense representation would require 3.6 GB. Matrix

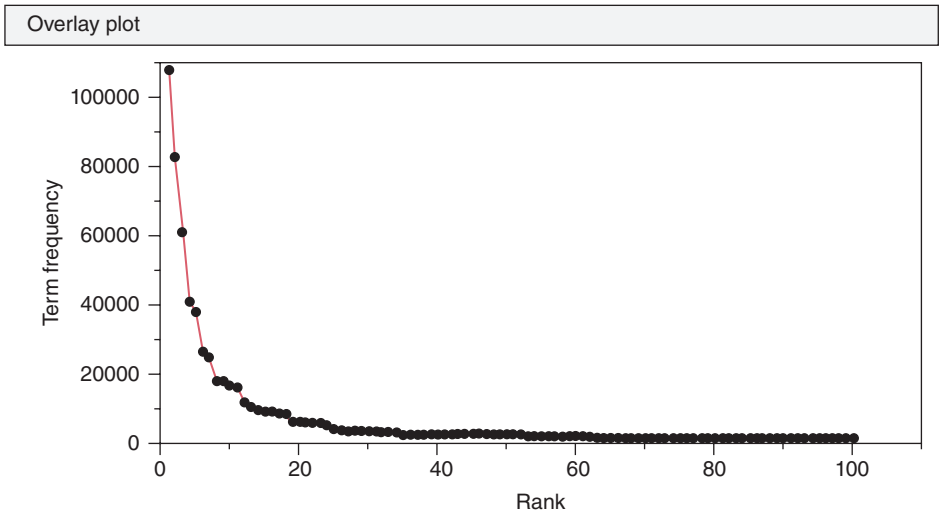


FIGURE 1 | Word frequency for National Science Foundation corpus with first four terms 'the', 'of', 'and'.

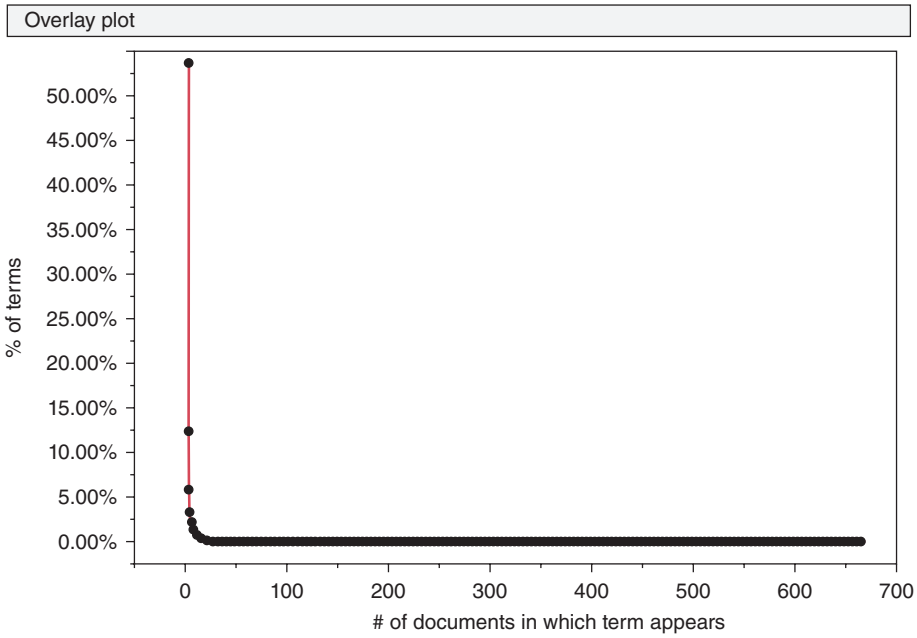


FIGURE 2 | Distribution of word frequency by documents.

multiplication involving the sparse representation is rapid thanks to algorithms that avoid explicitly performing multiplications by 0.

The raw term counts that appear in the DTM are often transformed. If a corpus contains a mix of short and long documents, the long documents (with higher term counts) may dominate the subsequent analysis of the DTM. Furthermore, words that occur in many documents (stopwords) will contribute columns with large entries to the DTM. If this tendency is not accounted for, these large column entries will lead to suboptimal document clustering results. A transformation may be local (a transformed component in the DTM depends only on the corresponding component in the original DTM) or global (a transformed component also depends on other components in the original matrix).

Weighting of the DTM

Several local transformations of the raw term frequency entries in the DTM are frequently used to dampen the right skew associated with count data. Depending on the analysis objective, it may be useful to use the actual term frequencies (TF) or transform using a zero–one (nonoccurrence or occurrence of the

particular term) binary representation, a log of the TF, or a scaled version of one of these by inverse document frequency.

A binary transformation replaces nonzero entries in the DTM with 1. This is useful when performing topic extraction to ensure that each topic discussed within a document receives equal weight, regardless of how many times it appears within that document. This also enhances the interpretability of analytics such as classification and regression trees (CART) built from the DTM.[14] That is, it is more informative to know that there is an association between the presence of a term and a response of interest than it is to know that the response is associated with a term appearing more than an arbitrary number of times in a document. A ternary weighting scheme is the same as binary except if a term appears twice or more, the value is 2 rather than 1 for the binary representation.[14]

Motivated by the fact that counting processes often follow a Poisson distribution, a log transformation is sometimes applied to the term frequency counts. This transformation dampens the presence of high counts in longer documents without sacrificing as much information as the binary weighting scheme.

The global term frequency-inverse document frequency (*tf-idf*)[14] transformation is perhaps the most frequently used. Unlike local transformations, which depend only on a single component of the DTM, the *tf-idf* transformation takes into account the relative frequency of each word across the corpus as well as the length of each document. It shrinks the weight of terms that appear in many documents while also inflating the weight of terms that appear in only a few documents. This often improves predictive and clustering performance.

The inverse document frequency (*idf*) for term t is:

$$idf_t = \log_2\left(\frac{N}{df_t}\right),$$

where N is the number of documents in the corpus and df_t is the number of documents containing term t. If a term appears in every document, its *idf* is 0. Terms that appear in most documents give little discriminating power (hence the removal of stopwords). To address this, we multiply the TF in the original DTM by an inverse document frequency, which downweights words that appear in many documents. That is, the *tf-idf* weighting[15] for document d and term t is:

$$tf\text{–}idf_{t,d} = tf_{t,d} \times idf_t.$$

Likewise, the inverse document frequency may be used to transform binary or log-transformed DTM's.

Regardless of the weighting scheme of the DTM, longer documents will be represented by row vectors with larger magnitudes than shorter documents. This property will give greater weight to longer documents in subsequent analysis and may produce suboptimal clustering results. To prevent this, the rows in the transformed DTM may be normalized so that the sum of each document vector is 1: the normalized DTM represents each document as a mixture of the terms that appear in it. For example, if a document D′ is created by pasting two copies of a document D together, D and D′ will be identical after normalization.[15]

The SVD

The DTM will tend to have several columns with few nonzero entries (Zipf's law). Many of these columns represent noise that is not useful for clustering the documents or building predictive models as seen in the NSF abstract example in Figure 3. For smaller corpora, there will often be more terms (columns) than documents (rows) in the DTM, leading to issues with subsequent analyses of the DTM. It is thus common to apply a dimensionality reduction procedure to the weighted DTM.

The reduced-rank SVD is a standard dimensionality reduction technique used in text mining.[16] The SVD reduces the DTM to a dense matrix with many fewer columns. The new (orthogonal) columns are linear combinations of the columns in the original DTM, selected to preserve as much of the variance structure

file.name	text of 1990 NSF Award Abstracts	aa	aaa	ab	abandon	abelian	aberr	abil	abiot	abl	ablat	abnorm	aboard	abort	abound
a9000006	Commercial exploitation over the past two hundred ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000031	Studies of chickens have provided serological and ...	0	0	0	0	0	0	0	0	1	0	0	0	0	0
a9000038	This research is part of an ongoing program by the ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000040	This SBIR proposal is aimed at 1 the synthesis of new ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000043	Dr Chisholm will investigate fundamental aspects of ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000045	This research will study the complexity of computatior	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000046	Duke University will operate the RV CAPE HATTERAS ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000048	The Scripps Institute of Oceanography will operate ...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a9000049	Bermuda Biological Station will operate the RV WEAT....	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FIGURE 3 | Portion of the very sparse National Science Foundation document-term matrix.

file.name	text	SVD1	SVD2	SVD3
a9000006	Commercial exploitation over the past two hundred ...	0.0221022908	0.0011192494	0.0000365208
a9000031	Studies of chickens have provided serological and	0.0265216975	0.0127392977	0.0046128733
a9000038	This research is part of an ongoing program by the prin ...	0.0048169574	0.0295638292	0.0103079577
a9000040	This SBIR proposal is aimed at 1 the synthesis of new ...	0.0178877164	0.0117685537	0.0073317583
a9000043	Dr Chisholm will investigate fundamental aspects of	0.0312545193	0.009322851	0.0066186216
a9000045	This research will study the complexity of computation	0.0185357152	0.0231188763	0.015040584
a9000046	Duke University will operate the RV CAPE HATTERAS ...	0.0073200182	0.0100655772	0.0113419483
a9000048	The Scripps Institute of Oceanography will operate four	0.0063642026	0.0094867337	0.0114474443
a9000049	Bermuda Biological Station will operate the RV WEATHI...	0.0084351651	0.0088749983	0.0124502687

FIGURE 4 | Sparse National Science Foundation document-term matrix reduced to a dense three-dimensional representation.

of the original DTM as possible. For a DTM X, the reduced-rank SVD factorization[16] is:

$$X \approx UDV^t,$$

where U is a dense d by s (where d is the number of documents and s the reduced number of dimensions) matrix with orthogonal columns (which produces a map to a rank-reduced description of documents), D is a diagonal matrix with nonnegative entries (the s largest singular values), V is a dense w by s (w is the number of terms) matrix with orthogonal columns with s as the rank of the SVD factorization (for $s \in \{1, \ldots, \min(d, w)\}$. V gives us a map to a rank-reduced description of terms. The s columns of U and V are organized in decreasing order of the magnitude of the corresponding singular values. Setting $s = 2$, for example, will give the best possible two-dimensional representation of the document and term spaces, respectively. The appropriate value of s is a matter of debate, and is application-dependent. Smaller values of s represent a greater extent of dimensionality reduction at the cost of a loss of structure of the original DTM. Values anywhere from 30 to 300 are commonly used. In practice, using a cumulative scree plot to select s so that roughly 75–80% of the original variance is recovered tends to strike a good balance.

An important computational tool for the calculation of the reduced-rank SVD is the Augmented Implicitly Restarted Lanczos Bidiagonalization Method.[17] This routine allows for an approximation of the s largest singular values and their associated singular vectors without requiring the calculation of all of the singular values. Available via the *irlba* package in R,[18] this routine requires 3.8 min to approximate the first 300 singular values and left and right singular vectors of the NSF corpus (again, about 9500 rows of 150 words each), while the standard R *svd* function takes 7.1 min to calculate the same vectors (and all of the singular values). As dimensionality increases, this gap in time is magnified.

FINDING TOPICS AND THEMES

In natural language processing, the use of a rank-reduced SVD is referred to as LSA.[15] LSA is effective for topic extraction and modeling. We will also discuss an alternative to using the SVD, called Latent Dirichlet Allocation (LDA). A popular LSA technique is to plot the corpus dictionary using the first two vectors resulting from the DV matrix. Similar words (words that either appear frequently in the same documents, or appear frequently with common sets of words throughout the corpus) are plotted together, and a rough interpretation can often be assigned to dimensions appearing in the plot, depending on the weighting scheme of the DTM. SVD reduces the DTM for the NSF example to $s = 3$ dimensions in Figure 4.

The SVD is related to the method of principal components analysis (PCA). The difference between the two approaches is that we do not subtract the column means of X prior to calculating the SVD, because doing so would produce a dense matrix for which calculation of the SVD would be much more computationally expensive. While the standard presentation of the calculation of the principal components of an $m \times n$ matrix X involves an eigendecomposition of the covariance matrix X^TX, this process can be numerically unstable. Instead, the right singular vectors of X yield the eigenvectors of X^TX, while the singular values of X yield the square root of the eigenvalues of X^TX. Much of the intuition from PCA applies to the reduced-rank SVD representation.

A factor analysis typically follows a PCA to study the relationship between the original variables

and the components (now called factors) resulting from a PCA. 'Factor loadings' describe the pairwise correlation between the original variables (terms) and the factors, and are produced by multiplying the eigenvectors by the square root of their associated eigenvalues. Thus UD produces the factor loadings describing the document space, and VD produces those for the term space. As $V^TV = I$, the approximation $XV \approx UD$ allows us to map new documents (using the original dictionary) into the same space without having to rerun the SVD. That is, the reduced dimensionality representation of the DTM is formed by postmultiplying the DTM by its right singular vectors (V) corresponding to the s largest singular values. When new documents are added to the corpus, they may be mapped into the same space by removing words that did not appear in the original corpus and postmultiplying the DTM of the new documents by the right singular vectors V of the original DTM. While this prevents having to recalculate the SVD, it also allows the components to be used for predictive modeling, because otherwise the composition of the components would change with each new SVD.

Varimax Rotation of the SVD of the DTM

It may be difficult to interpret the SVD factors when several terms are highly correlated with each factor. It turns out that a change of basis vectors representing the linear subspace spanned by V may result in a representation that is more understandable. The rotation is chosen to produce a loading matrix with simple structure,[19] where each factor will be strongly correlated with only a few of the original variables, and the other correlations should be near zero.[20] A 'rigid rotation' is simply a change of basis vectors representing a linear subspace via multiplication by an orthogonal matrix.

We can preserve the variance structure of the data represented by the loading matrix $L = XV \approx UD$ by postmultiplying by an arbitrary orthogonal matrix T. The covariance matrix of LT is $LT(LT)^T = LTT^TL^T = LL^T$, identical to the covariance matrix of L. Thus the representation XVT of the document space preserves the original correlation structure of the document space represented by XV while improving the interpretability of the resulting factors. Alternatively, a rotation may be applied to the term space VD.

Different factor rotations impose different criteria on LT when selecting T.[21] The varimax rotation[22] is one of the most widely used factor rotation due to its record of effectiveness. The varimax criterion selects T such that most of the entries in LT are near 0 or 1 by maximizing the sum of the intracolumn variance in LT: this makes large loadings larger and small loadings smaller. See the Refs 23–26 for further discussion of the varimax rotation, including Ref 27 for a concise statement of the varimax objective function. The Refs 21 and 28 discuss the varimax rotation in the context of text mining. The varimax criterion may be used to select a rotation matrix that is tuned to either of the loading matrices produced by the SVD: UD or VD. The former will attempt to create factors in which documents are cleanly separated while the later will seek factors that cleanly separate terms. These representations are more useful for topic extraction than the raw SVD output.

To illustrate, we obtain the first two SVD factor loadings of the NSF Abstract normalized binary DTM (after stemming, removing stopwords and words that appear in fewer than 15 abstracts, and reducing the rank to $s = 300$).

The first component is dominated by the term 'will'. 'Research', 'use', and 'study' are the next largest terms. The second component has large positive weightings on 'award' and 'student'. 'Research' has relatively large (in magnitude) weightings on both factors. This bivariate plot of these factors in Figure 5 does not give much insight into the major research topics in 1990.

Figure 6 clearly illustrates after applying the varimax rotation (tuned to UD), the largest component involves topics from genetics research. The second component may be interpreted to involve the development of computer algorithms, including parallel processing and simulations. Because the varimax rotation reduces the extent to which terms have large loadings on multiple components, we may consider the s components as individual topics (Figure 7), without having to examine biplots as is typically done in a LSA. This facilitates both the processes of topic extraction and of predictive modeling, as significant factors may be assigned an interpretation.

Probabilistic Topic Models

Probabilistic Topic Models such as Latent Dirichlet Allocation (LDA) offer an alternative to LSA using SVD. LDA still makes use of the 'bag-of-words' approach with the DTM, but uses a different method for the analytics. The general idea is that there are collections of terms that define a topic and each document contains one or more topics with certain probabilities. Blei[29] defines the goal of LDA is to automatically discover topics that are hidden (latent) using a statistical model to assign probabilities the topics are contained in a specific document. The text data is treated as arising from a generative process with hidden variables where there is a joint distribution over

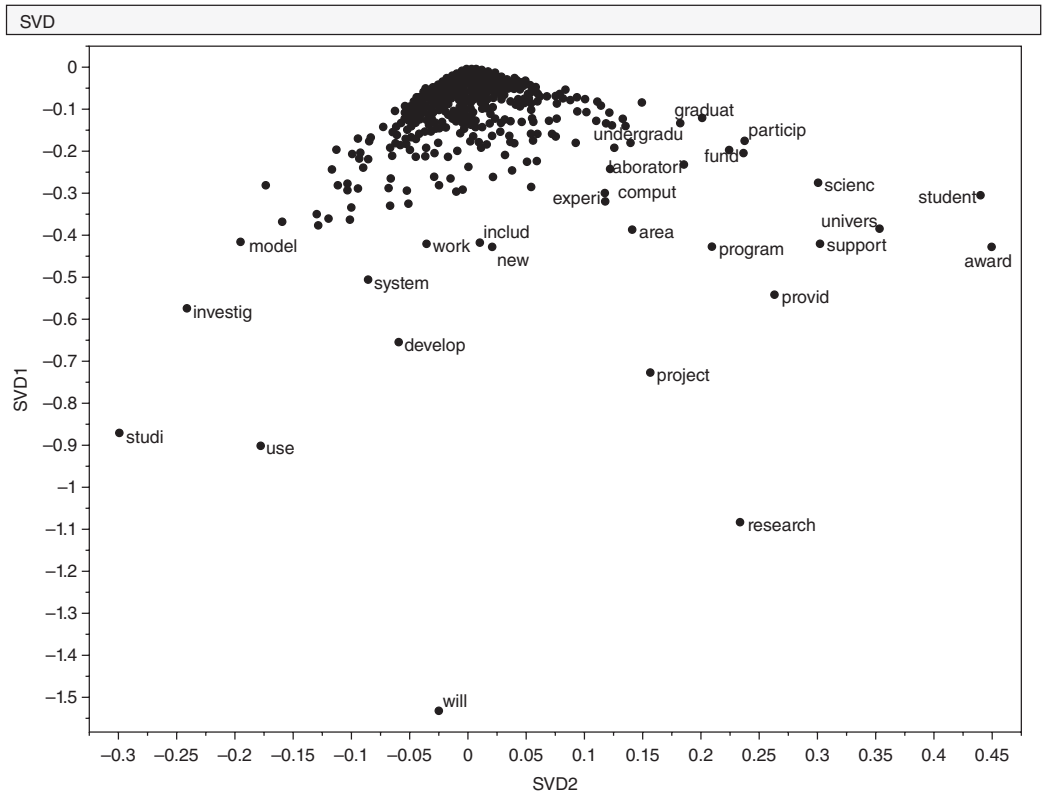


FIGURE 5 | Plot of first two singular value decomposition factors for National Science Foundation abstracts showing no real insights.

the observed (words in the documents) and hidden (random variables). The goal is to find the posterior distribution (given the observed text) in a Bayesian context, which will allow theme extraction, information retrieval, similar documents clustering, and word grouping exploration.

The hidden random variables of interest are what words are assigned to what topic, the proportions of topics on each document, and the topic distribution over the corpus. The Dirichlet distribution is the conjugate of the multinomial and is characterized by the positive vectors summing to 1. For example, the topic proportions for a particular document would be required to sum to 1 as would word membership to a topic. Note that a distinguishing characteristic of LDA models is they are mixed-membership models, which allow modeling of more than one topic per document and assume that topics are independent. The posterior distribution to estimate these hidden random variables based only on the DTM is intractable and requires advanced algorithms for approximation. Blei and Lafferty[30] suggest variational expectation maximation and Markov Chain Monte Carlo (e.g., Gibbs Sampling) methods, among others.

Blei[29] indicates the LDA is a developing methodology, and that many useful practical applications are now available based on relaxing some of the assumptions of the strict LDA formulation. Correlated Topic Models (CTM) relax the independence of topics requirement in the posterior approximation. The CTM would indicate in our NSF example the word *parallel* might be more associated with a *computer* topic than one on *oceanography*. A dynamic topic model can show the differences over time as in longitudinal studies by incorporating the order in which the documents input into the DTM. Griffiths et al.[31] extend LDA methods by modeling syntax rather than the general 'bag-of-words'.

Grun and Hornick[4] have developed the *topicmodels* package that enhances the R functions *tm* and *lda* to allow several approximation methods of the posterior distribution. The package also can provide estimates using the CTM. The number of topics must be specified prior, though you can iteratively solve for the optimal number by using training and validation sets. Additionally, the estimates for the prior distribution of terms on topics need to be specified for the Bayesian estimate in the Gibbs Sampling application. Output includes the most likely topic for each document, the distribution of weights for the terms on each document, the posterior distribution, and the ability to classify new documents.

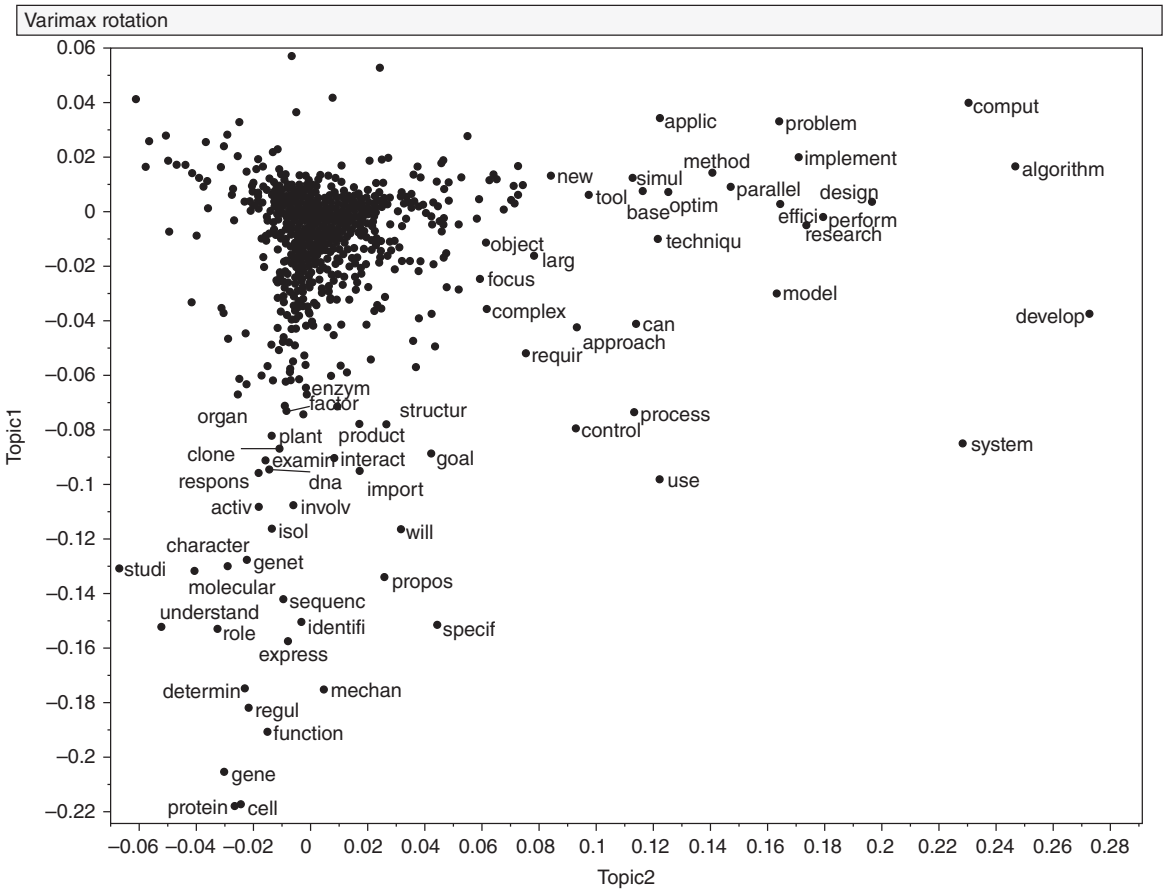


FIGURE 6 | Plot of first two varimax rotated singular value decomposition factors clearly showing the genetic and also the computer algorithm research.

The first 10 terms for the first five topics from *topicmodels* on the NSF data set using the Vector Expectation Maximization option are:

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"polit"	"workshop"	"parallel"	"algebra"	"women"
[2,]	"ligand"	"minor"	"machin"	"polynomi"	"matrix"
[3,]	"blood"	"wave"	"graph"	"robot"	"schedul"
[4,]	"chiral"	"propag"	"logic"	"ring"	"household"
[5,]	"soviet"	"strengthen"	"workstat"	"singular"	"gcms"
[6,]	"calcium"	"disabl"	"languag"	"combinator"	"job"
[7,]	"legal"	"committe"	"vision"	"arithmet"	"matric"
[8,]	"attitud"	"allianc"	"compil"	"theorem"	"gender"
[9,]	"microtubul"	"phd"	"invers"	"ellipt"	"logic"
[10,]	"parti"	"honor"	"concurr"	"conjectur"	"ethic"

Much like the varimax rotated eigenvectors from LSA using SVDs, these topics should be viewed as more exploratory than definitive. They often provide insight but may not be as cleanly interpretable as hoped. For example, Topic 1 in this case appears to have a few distinct focus areas. Looking at the first three NSF abstracts that are identified as having a predominant 'Topic 1' loading, they are all focused on chemistry areas more so than political attitudes and Soviet studies. Topic 3 appears to be more homogenous in the definitions of the top 10 words. These topics were generated using a default value of 30 total topics. If that value is increased, then there is less ambiguity on each topic, though care should be taken to not have too many topics.

Similarly, the first 10 terms for the first five topics using Gibbs Sampling are:

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"ocean"	"earthquak"	"deform"	"isotop"	"manifold"
[2,]	"circul"	"soil"	"strain"	"rock"	"topolog"
[3,]	"indian"	"seismic"	"ceram"	"miner"	"singular"
[4,]	"phytoplankton"	"damag"	"microstructur"	"tecton"	"supercomput"
[5,]	"color"	"fault"	"melt"	"mantl"	"symmetri"

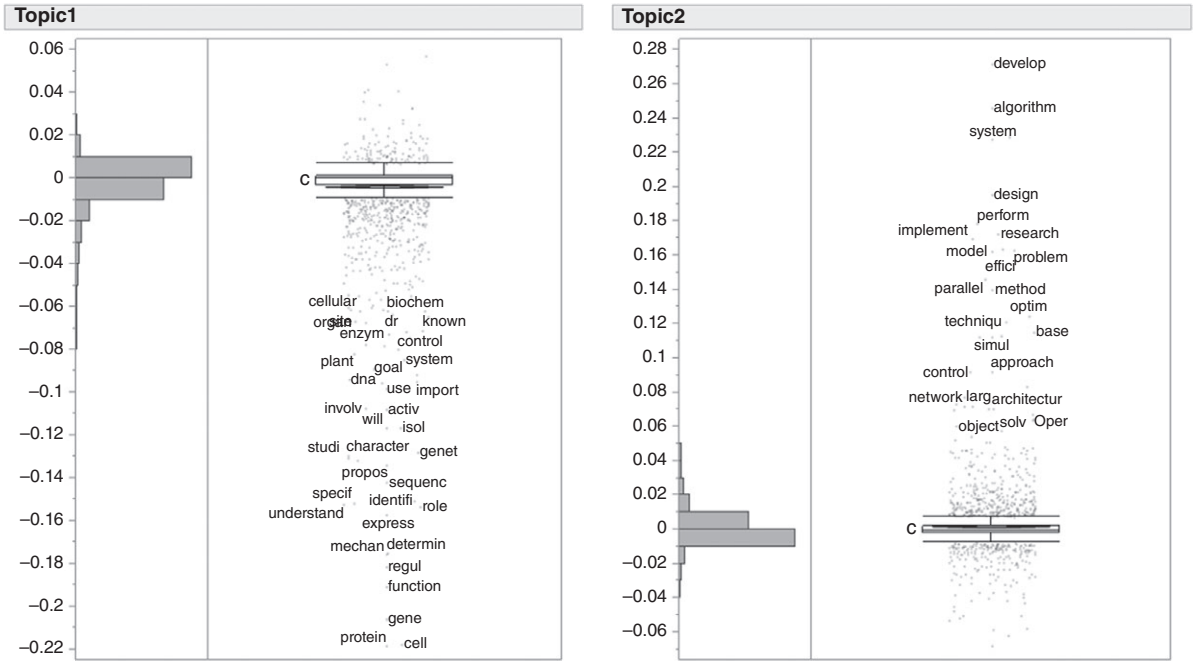


FIGURE 7 | Stand-alone topics from singular value decomposition factors using varimax rotation.

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[6,]	"equatori"	"prieta"	"failur"	"ridg"	"nsfnet"
[7,]	"india"	"loma"	"alloy"	"plate"	"harmon"
[8,]	"trap"	"geotechn"	"grain"	"volcan"	"ellipt"
[9,]	"ozon"	"francisco"	"plastic"	"crustal"	"orbit"
[10,]	"tracer"	"wall"	"fold"	"crust"	"negat"

From this sample of the first five topics and our experience with other examples, the Gibbs sampler has more interpretable and separated topics.

Finally, the CTM option that does not require linearly independent topics shows a fairly homogenous collection of top words for each of the topics.

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"ice"	"membran"	"renov"	"minor"	"particl"
[2,]	"sediment"	"brain"	"specimen"	"soil"	"solar"
[3,]	"basin"	"receptor"	"museum"	"forest"	"acceler"
[4,]	"tecton"	"hormon"	"arfmp"	"video"	"plasma"
[5,]	"antarct"	"neuron"	"firm"	"filter"	"rna"
[6,]	"ocean"	"neural"	"databas"	"strengthen"	"bed"
[7,]	"arctic"	"peptid"	"speech"	"reu"	"ionospher"
[8,]	"sedimentari"	"nerv"	"plant"	"centrifug"	"aerosol"
[9,]	"fauna"	"nervous"	"nest"	"allianc"	"detector"
[10,]	"stream"	"game"	"curat"	"phd"	"wind"

APPLICATION OF CLUSTERING AND TREE MODELS

Returning to LSA with rank-reduced SVDs, we can use the DTM and eigenvectors to discover additional structure. Now that the text of the corpus has been converted to a data matrix with documents (or paragraphs, survey responses, incident report, etc.) as rows and terms as columns, we have opened the possibility of using standard statistical and machine learning tools to answer questions of the type posed in the introduction. The full DTM X does have some peculiarities (sparseness, all entries are nonnegative) that require some modification of existing methods,[14] which will be discussed below. By contrast, the rank-reduced SVD provides a dense representation XV of the document space that is more amenable to existing routines.

Clustering

While topic extraction procedures can be useful for identifying prevalent themes in a corpus, it is likely that documents are 'active' on several topics at once. A cluster analysis provides the capability to examine document proximity across all factors simultaneously.[28] The results of a cluster analysis may also be used to provide a representative subsample of the corpus. A stratified sample of one document from each cluster will provide a sample that

is more representative of the corpus than a simple random sample (of the same size) from the entire corpus.

A cluster analysis of the full DTM requires some special accommodations. The quality of the results from a clustering algorithm suffers in the presence of irrelevant variables. A related issue is the sparsity of the DTM: not all statistical software packages provide sparse matrix functionality. Furthermore, there is more information in the overlap of positive entries (shared words between two documents) than in the overlap of zero entries (words that are absent from both of two documents). The Euclidean metric gives equal weight to these and often performs poorly when compared to other metrics such as cosine and Jaccard.[32]

By contrast, the rank-reduced SVD representation of the DTM may be used with standard clustering software. As with other clustering applications, Ward's method tends to perform well with up to around 30,000 documents (depending on computer memory availability), after which *K*-means may be used. It is not necessary to standardize the columns, as the SVD has already scaled them appropriately. Another advantage is that the dimensionality reduction provided by the SVD eliminates redundant/irrelevant variables that can often cause problems with clustering algorithms.

For applications with short documents, a sample of the documents in each cluster can be read in order to assign a theme to the cluster. For longer entries, such as the NSF abstracts, separate analyses may be run on the largest clusters to identify the topics that are present (either using a topic extraction procedure, or, in many cases, by simply examining the TF). In the NSF example, a hierarchical clustering suggests around 400 clusters. The largest three clusters each consist of just under 90 abstracts and deal with conference funding, hosting teacher workshops and summer programs, and undergraduate summer research programs.

Classification and Prediction

In some cases, it may be important to be able to predict responses for future observations. Some examples of text analytics classification problems are identification of fraudulent insurance claims, loan defaults, likely customers for cross or upsell, and email spam. Likewise, the documents may be used to forecast a claim amount, a repair cost, or a product rating. Special care must be taken when handling new observations: they need to be transformed to the space spanned by the SVD on the training data. If X_0 is the DTM formed by a new collection of documents with the same columns as the original DTM $X \approx UDV^t$ (words not appearing in the original corpus are excluded, and columns of 0's are introduced where necessary), then X_0V (or X_0VT, if a rotation T was applied to the original corpus) furnishes the reduced-rank representation of the new documents.

Models ranging in complexity from regression and naïve Bayes[14] to support vector machines[33] have proven useful in text mining. Because a text analysis begins as a screening procedure, several different modeling techniques should be used in combination with different options for the DTM (weighting, reduced-rank, etc). This process may be described as something of an art: experience helps develop an intuition that is not easy to teach. While fitting multiple models with multiple settings will increase the number of false positives when fitting a response (funding dollars, fraud indicator, etc.), subject-matter knowledge will often quickly dismiss spurious results. Remaining significant results may be further investigated through cross-validation or, ideally, by performing a hypothesis test on new data.

A random forest provides a useful approach when the (rotated) SVD vectors are used as predictors: especially when a rotation is applied to the document space, documents will often cluster into two clusters within the factors. This grouping works naturally with the cut-points created by CART methods, and less well with regression. The clarity that the varimax rotation can often bring to the interpretation of the factors produced by the SVD can be extremely useful when building a predictive model. If a significant association is found between an SVD factor and a response, then there is a possibility of assigning an interpretation to the result. We processed a corpus of 3235 aircraft accident reports provided by Miner et al.[34] into a *tf-idf* DTM and obtained a (term-space tuned) varimax rotation of 150 SVD factors. We fit the binary fatality indicator using the SVD factors. The Column Contributions plot indicates that Topic 36 (SVD36) is most strongly associated with the fatality indicator. The associated topic plot suggests that Topic 36 involves pilots continuing to fly in visual flight rules (VFR) when in fact the conditions have worsened requiring instrument flight rules (IFR). This is consistent with subject-matter expertise that general aviation fatalities are most prominent in the transition between VFR and IFR where pilots may not be qualified or proficient enough in instrument meteorological conditions characterized by low visibility. In subsequent predictive analysis, we use these significant SVD factors that originated from a text field along with structured variables (e.g., time of day, month, location, wind speed).

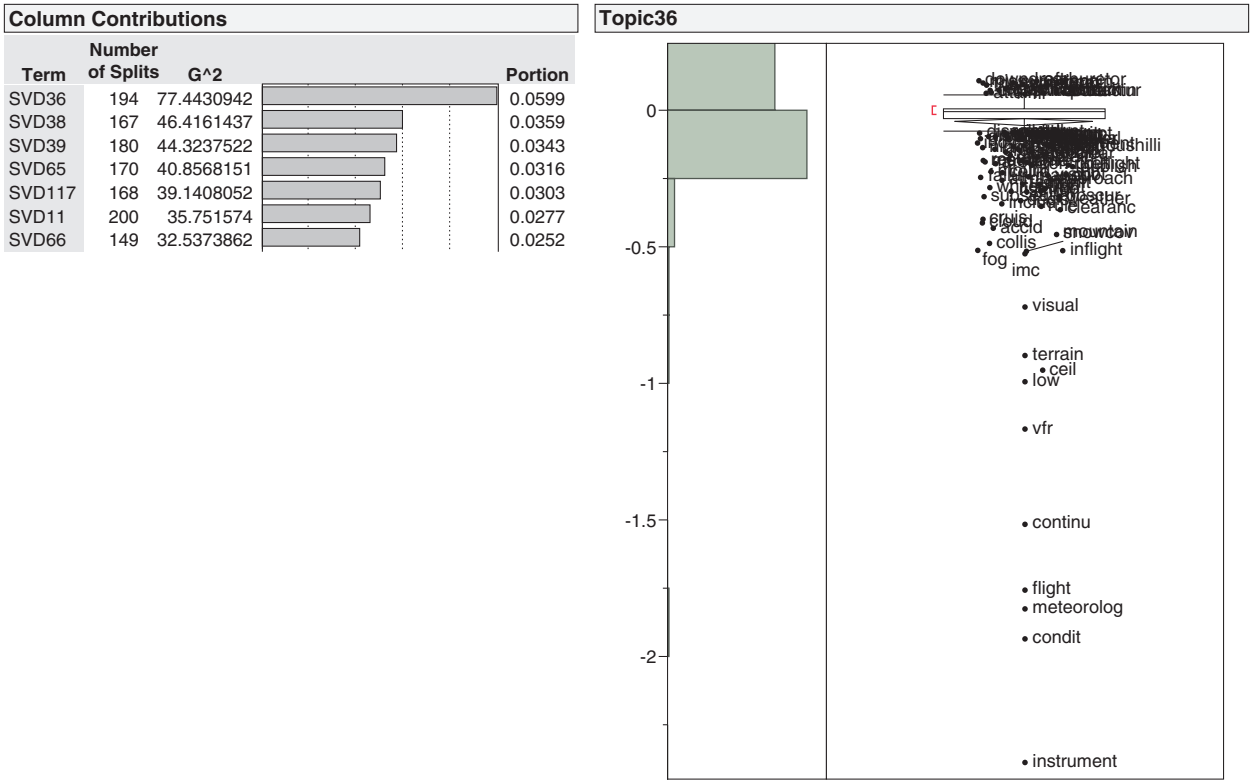


FIGURE 8 | Singular value decomposition factors most associated with fatalities.

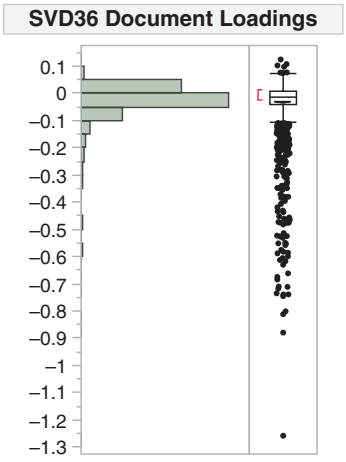
The document loadings on Topic 36 in Figure 8 indicate which documents are active on the topic. Sorting on Topic 36 document loadings and reading the first few entries confirms our interpretation of the topic plot (Figure 9).

The classification tree on the fatality response using the word columns from the DTM (binary weighting) as input variables in Figure 10 provides an interesting relationship among the terms. The proportion of fatalities is indicated by the blue for each node along with the number of records (count). If 'land' is in the narrative cause text, then very few fatalities are expected. Those that contain 'mountain' are associated with fatalities. If 'land' and 'low' are not in the text, then the likely cause of a fatality is a stall or spin. With 'land' not in the text then dark and autorotate are the active words to explain fatality when 'low' is present in the narrative.

Sentiment Analysis

While the bag-of-words approach discards information about the grammatical context in which words appear, the words still carry information via their definitions. Psychologists developed the Harvard IV-4 dictionary[35] of approximately 1600 positive words and 2000 negative words. For each DTM row (corresponding to a specific document, paragraph, record, etc.) a count is tallied for instances of positive words and a count for negative. These sentiment lists perform reasonably well with corpora from a wide range of disciplines. They are useful, for example, for sorting news articles or Twitter posts according to the tone of the topic discussion. However, better performance may be achieved when analyzing text from a specific source by developing custom sentiment lists, from domain knowledge or given a set of training data that has either been classified (e.g., positive, negative) or rated (e.g., overall satisfaction).[36–38]

To illustrate: using a corpus of car-owner reviews along with numerical overall satisfaction ratings,[39] it is possible to build a custom sentiment analysis list to be used when predicting the overall rating of new reviews. By building a lasso regression and additionally fitting a CART to model the overall user rating (scaled from 1 to 5) using the columns of the binary DTM (not the SVD representation), it is possible to find which words are associated with the overall rating. Words in the list {transmission, lemon, squeak, cheap, plastic, fix, problem, horrible, dealership, sent, claim, window, etc.} are most strongly associated with lower ratings, while words in the list {good, handle, smooth, accelerate, great, reliable,



narr_cause	SVD36 from U
pilot's continued visual flight into instrument meteorological conditions.	1.2586525754
The pilot's continued flight into instrument meteorological conditions. Contributing factors were fog, low ceilings, d...	0.879509973
The pilot's continued VFR flight into instrument meteorological conditions while maneuvering. Factors in the accide...	0.8118630715
The pilot's continued visual flight rules flight into instrument meteorological conditions resulting in the airplane colli...	0.8011472086
The pilot's continued VFR flight into instrument meteorological conditions, and subsequent collision with terrain dur...	0.745395256
The pilot's poor judgment in that he intentionally initiated VFR flight into instrument meteorological conditions. Co...	0.7393123011
The pilot's continued VFR flight into instrument metrological conditions during cruise flight. Factors associated in th...	0.7390912097
The pilot's continued VFR flight into instrument meteorological conditions, and his failure to maintain terrain cleara...	0.7351538413
The in-flight loss of control of the airplane in instrument meteorological conditions for undetermined reasons.	0.7176443542
The pilot encountering instrument meteorological conditions and his subsequent loss of control of the airplane.	0.7114139375
The pilot's continued VFR flight into instrument meteorological conditions, and subsequent collision with trees while...	0.7101342894
The pilot's continued VFR flight into instrument meteorological conditions, and subsequent in-flight collision with te...	0.6842974747
The pilot's inflight decision to continued visual flight into instrument meteorological conditions resulted in the inflig...	0.6748188452
The pilot's intentional continuation of a visual flight rules (VFR) flight into instrument meteorological conditions (IM...	0.6617060754
The pilot continued visual flight flight into instrument flight conditions resulting in an in-flight collision with terrain. ...	0.6300856776
The pilot's intentional flight into instrument meteorological conditions and failure to maintain clearance from terrain...	0.6168120814
the pilot's inadvertent visual flight rules (VFR) flight into instrument meteorological conditions (IMC) during cruise f...	0.6126443124
The pilot's failure to maintain adequate clearance from an obstruction (antenna tower), and his continued flight into...	0.6056780679
the pilot's inadverent VFR flight into instrument meteorological weather conditions which resulted in an in-flight coli...	0.5989421647
The pilot's continued VFR flight into IMC conditions resulting in the airplane colliding with mountainous terrain.	0.588182416

FIGURE 9 | Records associated with low document loadings for Topic 36.

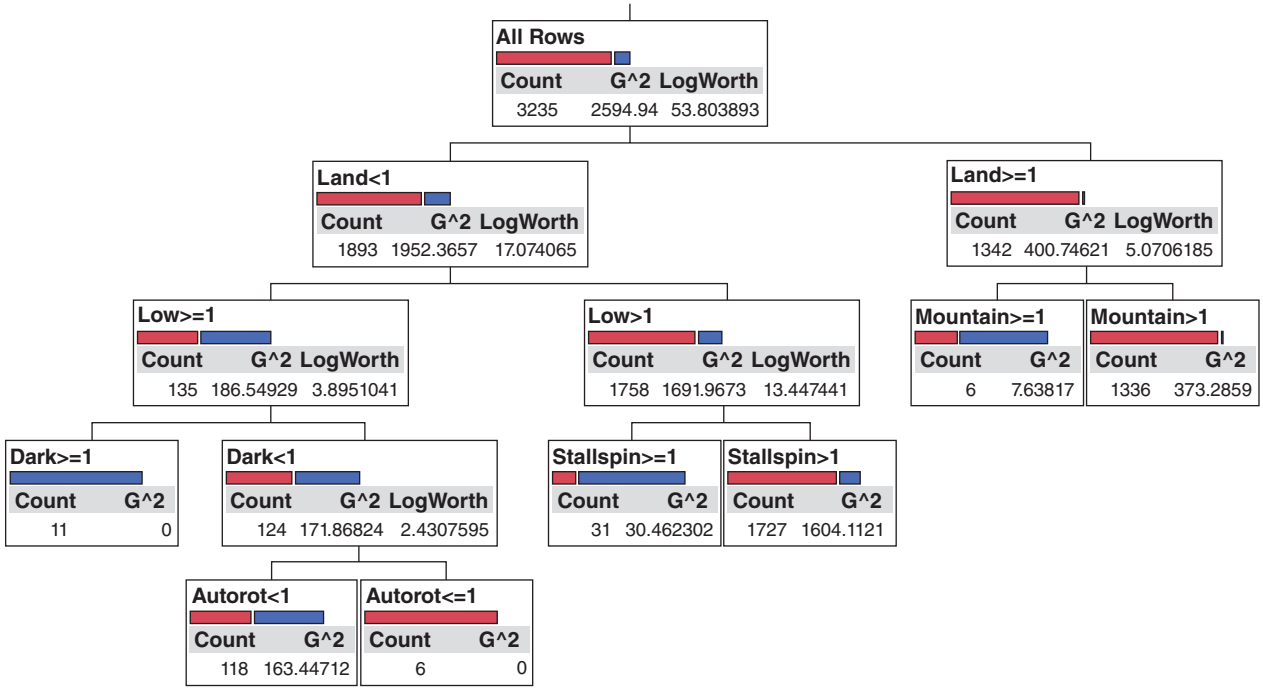


FIGURE 10 | Classification tree for fatality response variable with words as factors.

navigation, bass, fast, comfort, etc.} are associated with higher ratings. Negation can be problematic with this approach (e.g., 'not reliable'). A quick solution is to concatenate all of the occurrences of 'not' in the corpus to the following word before forming the DTM, though more sophisticated methods are also available.[40,41]

SOFTWARE

There are numerous software choices for text mining—each must be balanced against the cost, capabilities, and user learning curve. Many organizations have modified or developed specific tools from open-source utility languages such as Perl, Java, and Python. Ingersoll et al. give examples and Java code to implement many of the advanced text analytics tools available through the popular open-source Apache Software Foundation.[3]

The R programming language offers several open-source packages for text mining.[9,42] The language's sparse matrix libraries are well maintained; newly published approaches are often first made available via the Comprehensive R Archive Network; programmers have the ability to compile C code into R packages to speed up processing, as

done by the wordcloud package.[43] However, while R offers tremendous flexibility for customizing program behavior, analysis of the results (especially graphically) can be challenging and limited compared to commercial packages. We have found it useful to import the DTM and its SVD decomposition into a standard statistical package such as IBM SPSS or SAS JMP after building the matrices in R. Automating this procedure with an add-in to customize input options allows for rapid exploration and discovery while taking advantage of the user's existing proficiency in a statistics application. The figures in this article are generated from SAS JMP.

Some popular full-scale commercial text mining programs include Clarabridge, SAS Text Miner integrated with Enterprise Miner, ANGOSS, IBM SPSS Modeler Premium, and StatSoft's STATISTICA. Chakraborty et al.[2] provide an introduction to SAS Text Miner that uses many of the methods discussed in this review along with advanced components of concept linking, parts of speech tagging, and ontology management.

http://www.kdnuggets.com/software/text.html provides a comprehensive list of text mining software packages. They are summarized and broken down into commercial versus open-source application. Additionally, a link is provided to each application's website for further review.

CONCLUSION

Unstructured text has valuable information that can be easily harnessed using widely available statistical methods. Just getting the text from potentially disparate sources into a usable format with rows as documents and columns as words can be a challenge. We show the bag-of-words text mining procedure supplemented by LSA that translates a DTM into meaningful factors through SVD can rapidly lead to discovery. A varimax rotation results in general topics from the corpus more clearly loading on the SVD factors. Hierarchical clustering of the rows of the SVD-U matrix consolidates documents with similar themes together while clustering the SVD-V columns collects similar words. Probabilistic Topic Models such as LDA are also highly effective in finding common themes and grouping documents. The SVDs themselves can be used as factors combined with structured data for predictive analytics. Decision trees and regression methods can help identify the most significant SVDs explaining a response variable. Sentiment analysis determines the proportion of positive versus negative words for each document and is especially useful for emerging fields such as social network analysis. Our experience has shown these methods that are inexpensively implemented with open-source software solutions are often all that is needed, as the complexities of grammatical and semantic metadata may offer only marginal additional insight.

REFERENCES

1. Miner G, Elder J, Hill T, Nisbet R, Dursun D, Fast A. *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*. Oxford: Academic Press; 2012.
2. Chakraborty G, Pagolu M, Garla S. *Text Mining and Analysis: Practical Methods, Examples, and Case Studies Using SAS*. Cary, NC: SAS Institute Inc; 2013.
3. Ingersoll G, Morton T, Farris A. *Taming Text*. Shelter Island, NY: Manning; 2013.
4. Grun B, Hornik K. Topic models: an R package for fitting topic models. *J Stat Softw* 2011, 40:1–30.
5. The Apache Software Foundation. OpenNLP. 2015. Available at: https://opennlp.apache.org/ (Accessed March 24, 2015).
6. Lichman M. UCI machine learning repository. University of California Irving, School of Information and Computer Science. 2013. Available at: http://archive.ics.uci.edu/ml (Accessed February 13, 2015).
7. Scrapinghub S. A fast and powerful web crawling framework. 2015. Available at: http://scrapy.org/ (Accessed March 10, 2015).
8. Temple Lang D. RCurl: general network (HTTP/FTP/ …) client interface for R. CRAN. 2015. Available at: http://cran.r-project.org/web/packages/RCurl/index.html (Accessed March 10, 2015).
9. Barbera P. streamR: access to Twitter streaming API via R. 2014. Available at: http://CRAN.R-project.org/package=streamR (Accessed March 10, 2015).
10. Bilisoly R. *Practical Text Mining with Perl*. Hoboken, NJ: John Wiley & Sons; 2008.
11. Porter MF. An algorithm for suffix stripping. *Program* 1980, 14:130–137.
12. Yang C. Who's afraid of George Kingsley Zipf? Or: do children and chimps have language? *Significance* 2013, 10:29–34.
13. Bates D, Maechler, M. Matrix: sparse and dense matrix classes and methods. CRAN. 2015. Available at:

http://CRAN.R-project.org/package=Matrix (Accessed February 13, 2015).

14. Weiss S, Indurkhya N, Zhang T, Damerau F. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. New York: Springer; 2005.
15. Manning CD, Raghavan P, Schutze H. *Introduction to Information Retrieval*. New York: Cambridge University Press; 2008.
16. Albright R. *Taming Text with the SVD*. Cary, NC: SAS Institute Inc; 2004.
17. Baglama J, Reichel L. Augmented implicitly restarted Lanczos bidiagonalization methods. *SIAM J Sci Comput* 2005, 27:19–42.
18. Baglama J, Reichel L. irlba: fast partial SVD by implicitly-restarted Lanczos bidiagonalization. CRAN. 2015. Available at: http://cran.r-project.org/web/packages/irlba/index.html (Accessed February 13, 2015).
19. Thurstone LL. *Multiple Factor Analysis*. Chicago, IL: University of Chicago Press; 1945.
20. Kline P. *An Easy Guide to Factor Analysis*. New York: Routledge; 1993.
21. Visinescu LL, Evangelopoulos N. Orthogonal rotations in latent semantic analysis: an empirical study. *Decis Support Syst* 2014, 62:131–143.
22. Kaiser HF. The varimax criterion for analytic rotation in factor analysis. *Psychometrika* 1958, 23:187–200.
23. Richman MB. Rotation of principal components. *J Climatol* 1985, 6:293–335.
24. Dunteman GH. *Principal Components Analysis*. Newbury Park, CA: Sage Publications, Inc; 1989.
25. Ramsay J, Silverman BW. *Functional Data Analysis*. New York: Springer; 2005.
26. Pett MA, Lackey NR, Sullivan JJ. *Making Sense of Factor Analysis: The Use of Factor Analysis for Instrument Development in Health Care Research*. Thousand Oaks, CA: Sage Publications, Inc; 2003.
27. Khattree R, Naik DN. *Multivariate Data Reduction and Discrimination with SAS Software*. Cary, NC: SAS Institute, Inc; 2000.
28. Evangelopoulos N, Zhang X, Prybutok VR. Latent semantic analysis: five methodological recommendations. *Eur J Inf Syst* 2012, 21:70–86.
29. Blei D. Probabilistic topic models. *Commun ACM* 2012, 55:77–84.
30. Blei D, Lafferty J. Topic models. In: Srivastava A, Sahami M, eds. *Text Mining: Classification, Clustering, and Applications*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. Boca Raton, FL: Chapman and Hall; 2009.
31. Griffiths T, Steyvers M, Blei D, Tenenbaum J. Integrating topics and syntax. *Adv Neural Inf Process Syst*, 2005 17, 537–544.
32. Strehl A, Ghoash J, Mooney R. Impact of similarity measures on web-page clustering. In: *Workshop on Artificial Intelligence for Web Search*, *AAAI*, 2000, 58–64.
33. Jiang EP. Content-based spam email classification using machine-learning algorithms. In: Berry MW, Kogan J, eds. *Text Mining: Applications and Theory*. West Sussex: John Wiley & Sons; 2010.
34. Nisbet R, Elder J, Miner G. *Handbook of Statistical Analysis and Data Mining Applications*. Burlington, MA: Academic Press; 2009.
35. Kelly E, Stone P. Descriptions of inquirer categories and use of inquirer dictionaries. Available at: http://www.wjh.harvard.edu/~inquirer/homecat.htm (Accessed March 10, 2015).
36. Tang H, Tan S, Cheng X. A survey on sentiment detection of reviews. *Expert Syst Appl* 2009, 36: 10760–10773.
37. da Silva NFF, Hruschka ER, Hruschka Jr. ER. Tweet sentiment analysis with classifier ensembles. *Decis Support Syst* 2014, 66:170–179.
38. Prabowo R, Thelwall M. Sentiment analysis: a combined approach. *J Informetrics* 2009, 3:143–157.
39. Lahoti S, Mathew K. Text mining: automobile brand review using STATISTICA data miner and text miner. In: Nisbet R, Elder J, Miner G, eds. *Handbook of Statistical Analysis and Data Mining Applications*. Burlington, MA: Academic Press; 2009.
40. Cruz Diaz N, Vazquez J, Alvarez V. A machine-learning approach to negation and speculation detection in clinical texts. *J Am Soc Inf Sci Technol* 2012, 63:1398–1410.
41. Carrillo-de-Albornoz J, Plaza L. An emotion-based model of negation, intensifiers, and modality for polarity and intensity classification. *J Am Soc Inf Sci Technol* 2013, 64:1618–1633.
42. Feinerer I, Hornik K, Meyer D. Text mining infrastructure in R. *J Stat Softw* 2008, 25:1–54.
43. Fellows I. wordcloud: Word Clouds. CRAN. 2014. Available at: http://cran.r-project.org/web/packages/wordcloud/index.html (Accessed February 13, 2015).

