Categorization and Classification

Boolean Queries, Model-Based Classifiers, Auto-Categories and Concept Topics
Introduction

Categorization of text documents means sorting them into groups. Lexalytics uses a combination of natural language processing (NLP) and machine learning to automatically classify customer reviews, support tickets, or any other type of text document based on their contents. Much like a librarian places a book about horses on a shelf labeled “Animals,” our NLP classifiers can categorize a product description as “Sauces” because the text mentions ketchup or mustard. And because our classifiers work at the speed of computers, we categorize thousands of documents in less than the time it takes to drink your morning coffee. When you get back to your desk, you can find and focus on the documents you care about.

Where other systems limit your ability to customize categories or configure models, we take a hybrid approach. With Lexalytics, you can reliably sort through huge numbers of diverse documents at multiple levels of analysis.

This white paper will discuss the four methods Lexalytics uses for text document categorization: Query Topics, model-based classifiers, Auto-Categories and Concept Topics.

Note: In the context of this white paper, we use the terms document categorization and document classification synonymously.
TEXT CATEGORIZATION USING BOOLEAN QUERY TOPICS

Query topics are search-based classifiers defined by Boolean operators. To categorize cellphone reviews by product, for example you might create a query for *iPhone*, another for *Pixel* and another for *Galaxy*. Similarly, to classify product information brochures you could create a query for each product or service. Medical affairs liaisons at pharmaceutical firms, meanwhile, use queries to monitor media sources for mentions of their company’s drugs and therapies.

In simplest terms, query topics are rules that decide whether a document belongs in a given category by looking for the presence or absence of key words and phrases. Therefore, queries work best when you know that a particular set of words are useful as categorization markers. A political analyst might create a general politics category using a query topic like this:

**Politics** — elect OR congress OR (president NOT ceo) OR senate OR senator OR representative

Here, the query topic is “Politics” and the words that define its contents are “elect,” “congress,” “president,” “senate,” and “representative.”

Lexalytics supports the query operators AND, OR, NOT, WITH, NEAR, and CONTEXT.

The **AND** operator looks for documents that have **all** of the words you specify. The query *representative AND clark*, for example, will only categorize documents that contain both of those words. The **OR** operator, on the other hand, looks for documents that contain any of the specified words.

Queries with **NOT** ignore all words after the operator. This narrows down the meanings of words and phrases or otherwise limits the scope of the query. For a business analyst, a query of *president* may match documents related to the political figure. But a query of *president NOT trump* will match articles related to business leaders while excluding articles about politics.

The **WITH** operator looks for where the specified words on both sides of the operator to appear in the same sentence. This is useful for improving the precision of a query topic for words that can mean many things, including ambiguous company names. A tweet that contains the word “alphabet” could be talking about a set of letters or the holding
company of Google. But the word “alphabet” with the letters “inc.” certainly refers to the latter. The query *alphabet WITH inc.* will only categorize documents that contains the word “alphabet” if the letters “inc.” also appear in the same sentence.

The **NEAR** operator looks for when specific words are found within a defined distance of each other, regardless of how many sentences that distance covers in the document. NEAR operators are useful for improving the precision of query topics.

For instance, news articles might discuss a made-up company called “Antarctic Camera” and sometimes shorten that to “Antarctic.” But to a computer, the word “Antarctic” by itself could be referring to the continent, not a company. Using the NEAR operator here helps you improve the relevancy of documents sorted into this category. The query *antarctic NEAR/25 camera*, for instance, will return documents wherein “antarctic” is within 25 words of “camera.”

Text to the left of the **CONTEXT** operator supplies the general idea you’re searching for, while the text to the right of the operator supplies the conceptual context. This is useful when the terms of your query can be more or less relevant depending on the ideas or topics associated with them. This operator is used for our **Concept Topics** categorization tool, discussed later.
THE LIMITS OF BOOLEAN QUERIES FOR CATEGORIZATION

The main advantage of query topics, their simplicity, is also their primary drawback. Query-based categorizers search for exactly what you tell them to, not what you mean by those words. For example, a simple query of *president* will return both political and business documents because it won't know if by “president” you mean a politician or a corporate executive.

To illustrate this limitation in greater detail, let's say you've downloaded a huge number of tweets with the goal of understanding how people are responding to Apple CEO Tim Cook's latest keynote. The first step is to categorize the tweets to focus on only those that are relevant to your analysis. To do this, you configure a basic query topic: *Apple – apple*

This takes a few seconds to set up and a bit longer to run on your documents. Unfortunately, you then realize that category now includes tweets that encompass:

- A type of food and the tree it comes from
- Cooking recipes that involve apples
- A technology company and its executives
- A record label founded by The Beatles

And much, much more...
The trouble is that “apple” can mean many different things. To find only relevant mentions, you need to add depth to your query. For example, changing it to *Apple – apple WITH tim WITH cook* might do a better job of only categorizing tweets that mention Apple’s CEO by name.

But what if a tweet refers to Cook the person without his first name? *Apple – apple WITH cook* would catch those – but it would also pick up tweets about cooking apples.

This is the **disambiguation trap**: the more meanings a word has, the more layers you need to add to your query topic. A query to categorize mentions of “apple” will require pages of Boolean operators. This quickly becomes cumbersome, unreliable and unsustainable.
MACHINE LEARNING FOR TEXT CATEGORIZATION

Machine learning models categorize documents by learning statistical connections between words and phrases and human-provided examples of where documents should be sorted. In effect, the models learn the likelihood that a given set of words are related to a particular category.

Model-based classification can be an efficient solution to the disambiguation trap. Once you've trained a categorization model for “apple,” for instance, it will reliably categorize your documents without the same upkeep requirements as pages-long queries.

Of course, machine learning models need training data to learn from. And the models you train are only as good as the data you feed them. Training data sets for categorization models are large document collections manually tagged with appropriate categories.

The training data you produce should be similar in structure and content to the data you plan to process through the model. For example, if you’re building a model to classify incoming customer support tickets, your training data should be pre-tagged tickets from your own archives. Machine learning models stumble and fall when the content they’re asked to process looks very different from the content they were trained on.

Expect to have at least a hundred pieces of training content prepared per category. The more training data you have, the more effective the model will be. The exact size of the smallest feasible training set will vary based on the quality of your documents and the complexity of your category taxonomy.
THREE APPROACHES TO MODEL-TRAINING

Let’s take a brief interlude to talk about the model-training process. Once you have training data, you have three options for actually building the model:

1. **Train it yourself using an open source algorithm**
2. **Use a self-service machine learning platform (paid or open source)**
3. **Have a text analytics/NLP vendor build and deploy it for you**

Training your own models based on an open source toolkit gives you total control over the process. But unless you’re already an experienced data scientist, there’s a lot for you to learn and build before you have anything operational. Before starting from scratch, think carefully about whether it’s truly cost-effective for you to invest your time and energy like this.

**Do-it-yourself machine learning platforms** dramatically accelerate the model-training process. But without an established data science program to use it, a trained model is all dressed up with nowhere to go. Before you dive into training your own text processing model, make sure you can actually deploy it into your existing data analytics infrastructure. That might be your enterprise’s business intelligence architecture, a visualization tool, or a platform-based natural language processing solution.
Lastly, consider that without a methodology for data gathering, annotating, and re-training, your model will quickly grow obsolete. Training a model on a pre-curated dataset is easy; scaling across your company and dealing with changing requirements is much harder. If you don’t have a good pipeline for sampling and annotating this data and managing the complexity as it grows over time, you’ll quickly run into problems.

The third option for model-training is to have your existing text analytics provider do it for you. Lexalytics, for example, has been using machine learning in the context of text analytics for well over a decade. We have hundreds machine learning models currently in use, including many for text categorization and classification. Along the way, we’ve learned a lot about how to reduce training data requirements while maximizing the model’s flexibility and minimizing its fragility.

We offer several types of machine learning engagement. If you’re a new or existing customer, you can pay us to train a text classification model and we’ll deploy it directly into your Lexalytics solution. Even if you’re not a direct customer of ours, we can build the model and then hand it over to you to deploy it yourself into your existing data analytics infrastructure. For more information, refer to our data sheet, Tuning and Machine Learning Services.
Queries are good for categorizing specific words and phrases but are inherently limited and require constant maintenance. Model-based classifiers require data to train on and are only really applicable when the data you’re analyzing is relatively consistent in content and form. To fill this gap, Lexalytics built the Concept Matrix.

The Concept Matrix draws on the words, phrases and the links between hundreds of thousands of Wikipedia™ articles to understand their relationships: large concepts, the topics and entities that branch off of them, and the semantic ideas that link them together.

Picture it this way: The Politics page on Wikipedia contains a link to an article on Governance. The Governance page has a link to the Corporation entry, which in turn includes links that discuss the legal influence of large corporations. The Concept Matrix represents this chain of association; the closer an article is to the original topic, the stronger their relationship.

Of course, two pages can be linked together without being more than superficially related. To account for this, we performed careful semantic analysis on the contents of each page to understand the language used and how it relates to the underlying ideas being discussed. From the results, we are able to gauge the relative strength of each link and dramatically improve the accuracy and reliability of the Concept Matrix.

The Lexalytics Concept Matrix is the foundation of our Concept Topics/User Categories and Auto-Categories features. When queries require too much work and machine learning models aren’t in the cards, these tools offer the perfect alternative.
**Auto-Categories**

Auto-Categories use the Concept Matrix to compare your documents to Wikipedia’s category taxonomy. We whittled down their taxonomy into 400 or so first-level categories and 4,000 second-level categories. **Auto-Categorization works best at sorting documents into broad groups**, such as Agriculture, Food, Health, or Electronics.

For example, *Food* is a first-level category clarified by the second-level categories *Food_preparation* and *Dumplings*, among others. When you run a document collection through our auto-categorizer, we evaluate the semantic association between those documents and the articles in each category. The system calculates a **relevancy score** between 0 and 1 that represents its confident that a particular document falls into a particular category. A low score represents uncertainty, while a high score indicates confidence.

When a document in your collection matches a category with a relevancy score of 0.45 or higher, that document is assigned to that category. If a category is not returned in the output, its relevancy score did not reach the minimum threshold. This threshold, called the **category weight**, is customizable. Custom weights do not affect the categorization algorithm, only the relevancy score.

Auto-categories work out of the box, requiring no customization at all. They’re the easiest tool to use in our categorization toolbox. But auto-categories can’t be changed or tuned. This means that if the categories themselves aren’t how you want to sort your content, or if you’re having accuracy or precision issues that require tuning, you’ll need to use a different categorization method.

### Engine Score | Custom Weight | Adjusted Score | Result
--- | --- | --- | ---
0.30 | 1.70 | 0.30 \* 1.70 = 0.51 | Category will be returned
0.35 | 0.95 | 0.35 \* 0.95 = 0.33 | Category will **NOT** be returned
0.35 | 1.50 | 0.35 \* 1.50 = 0.53 | Category will be returned

View the Full List
CONCEPT TOPICS & USER CATEGORIES

Concept Topics use the Concept Matrix to broaden the scope of a user-defined category beyond a few example keywords. For example, a Concept Topic written as *Bread*: *rye, sourdough, baguette* will also categorize documents that mention pumpernickel and brioche, because the Concept Matrix understands that those words are related to the larger idea of bread.

Similarly, a Concept Topic can correctly categorize the sentence, “I had some great chicken wings,” as Food, because it understands that chicken wings are related to the concept of food. “American Airlines had to announce a gate change,” will categorize to Aviation, even though none of those words are in the list of terms defined by the Aviation concept topic.

Concept Topics are known as User Categories in Semantria. They are functionally the same thing, and we’re actively working on standardizing names to Concept Topics.

Semantria ships with a default list of 40 Concept Topics, and users can create their own (hence their alternate name, User Categories).

To create a new Concept Topic, give it a name and define 4-6 examples of words that fall into that category. For a Vegan category, for instance, you might use *vegan, diet, vegetarian, animal, tofu, veganism* as example words. The Concept Topic evaluates these against the Concept Matrix to identify other words and phrases that are related to the larger idea, and then categorizes documents containing those items as well.

Because the Concept Matrix works by comparing the contents of a document against the web of associations formed by Wikipedia articles, **Concept Topics work best on longer documents**. In other words, the more words there are in your documents, the better Concept Topics are at categorizing those documents. The one or two keywords found in an average tweet simply aren’t enough for the Concept Matrix to work off of. News articles, press releases and similar pieces, on the other hand, are perfect candidates for Concept Topic-based categorization.

<table>
<thead>
<tr>
<th>Sentence for Analysis</th>
<th>Food</th>
<th>Agriculture</th>
</tr>
</thead>
<tbody>
<tr>
<td>I like chicken.</td>
<td>0.58</td>
<td>No match</td>
</tr>
<tr>
<td>I like chickens.</td>
<td>No match</td>
<td>0.71</td>
</tr>
<tr>
<td>I like to eat chickens.</td>
<td>0.59</td>
<td>0.51</td>
</tr>
</tbody>
</table>

Default Concept Topics

- Aviation: aviation, airplane, flying
- Biotechnology: biotech, biotechnology, applied biology, gene therapy, genetic engineering
- Food: food, meals, vegetable, meat, fruit

View the Full List
SUMMARY

Categorization of text documents means sorting them into groups. Lexalytics uses a combination of natural language processing (NLP) and machine learning to automatically classify customer reviews, support tickets, or any other type of text document based on their contents. Applications of NLP-based document classification center around sorting and organizing, including email routing and customer review analytics.

Lexalytics provides four methods for text document categorization: Boolean Query Topics, model-based classifiers, Auto-Categories and Concept Topics. Boolean queries are simple to understand, easy to set up, and work well when you’re focusing on a very specific set of words. Model-based classifiers are efficient and requires less human input than queries but require training data and expertise to build. Auto-Categories and Concept Topics are good at sorting content into broader, idea-based categories but don't work so well on shorter documents.

Most text categorization tools rely on only one or two of these methods. Because Lexalytics supports all four, we’re able to solve both simple and complex document classification problems faster and more effectively than other providers.

Lexalytics processes BILLIONS of text documents every day, GLOBALLY.

We help our clients unlock the full value of their text data through innovative text analytics solutions delivered through the modular Lexalytics Intelligence Platform® and “semi-custom” enterprise applications.

Our solutions combine natural language processing, semi-structured data parsing, and machine learning to reveal context-rich patterns and insights within comments, reviews, contracts, medical files and other complex text documents.

Enterprises and data analysts rely on Lexalytics to improve customer experiences, reduce employee turnover, manage regulatory compliance, improve process automation, and more by gaining deeper insights and greater value from their data.

For more information, visit www.lexalytics.com or call 1-800-377-8036