



Text Mining

Predictive Modeling II
Auburn University

Part I. What is Text Mining?

What is Text Mining?

- ***Text mining*** is an exercise to gain knowledge from text.
- Text:
 - Web pages
 - Medical records
 - Customer surveys
 - Email filtering (spam)
 - DNA sequences
 - Incident reports
 - Drug interaction reports
 - News stories (e.g. predict stock movement)

What is Text Mining?

- There are many examples of text-based documents (all in ‘electronic’ format...)
 - e-mails, corporate Web pages, customer surveys, customer reviews on products and services, résumés, medical records, DNA sequences, technical papers, incident reports, news stories and more...
- Not enough time or patience to read
 - Can we extract the most vital kernels of information...
- So, we wish to find a way to gain knowledge (in summarised form) from all that text, without reading or examining them fully first...!

What is Text Mining ?

- **Structured Data:** generally refers to data that has a defined length and format.
E.g. numbers, dates, and groups of words and numbers called strings...
- **Unstructured Data:** refers to data that does not follow a specified format.
E.g. text messages, books, pictures, videos, the body of an email message, data generated from social media platforms...
- **Semi-structured Data:** does not necessarily conform to a fixed schema, but may be self-describing and may have simple label/value pairs.
E.g. XML, emails...

Text Mining

- Typically falls into one of two categories
 - **Analysis of text:** I have a bunch of text I am interested in, tell me something about it
 - E.g. sentiment analysis, text cluster analysis
 - **Retrieval:** There is a large corpus of text documents, and I want the one closest to a specified query
 - E.g. web search, library catalogs, legal and medical precedent studies

Text Mining Applications

Yahoo Buzz

[Yahoo!](#)
[My Yahoo!](#)
[Mail](#)
[Make Y! your home page](#)
 Search:
[Web Search](#)



[Sign In](#)
 New User? [Sign Up](#)

[Buzz Index Home](#) - [Help](#)

What the world is searching for...

Tuesday, August 26, 2008

[Buzz Log](#) | [Overall](#) | [Actors](#) | [Movies](#) | [Music](#) | [Sports](#) | [TV](#) | [Video Games](#) | [Decliners](#)

top overall searches

Leaders

[RSS](#)
[MY Y!](#)

| Rank | Prev. | Subject (Days on Chart) | Move | Score |
|------|-------|--|------|-------|
| 1 | ↑ | Amanda Peet (1) | +542 | 549 |
| 2 | ↑ | Dancing With The Stars (1) | +304 | 327 |
| 3 | ↓ | 2008 Olympics (17) | -377 | 323 |
| 4 | ↑ | Ellen Barkin (1) | +273 | 274 |
| 5 | ↑ | Luciana Barroso (1) | +245 | 246 |
| 6 | ↑ | Mia Hamm (1) | +229 | 229 |
| 7 | ↓ | Hi-5 (301) | -17 | 219 |
| 8 | ↑ | Jessica Biel (2) | +160 | 198 |
| 9 | ↑ | Kara DioGuardi (1) | +184 | 184 |
| 10 | ↑ | Brooke Mueller (1) | +155 | 155 |
| 11 | ↑ | Giant Squid (1) | +151 | 152 |
| 12 | ↑ | Kim Kardashian (225) | +84 | 140 |
| 13 | ↓ | NFL (26) | -31 | 117 |
| 14 | — | WWE (623) | +15 | 104 |
| 15 | ↓ | myYearbook (31) | -19 | 99 |
| 16 | ↓ | Club Penguin (7) | -49 | 94 |
| 17 | ↓ | Limewire (34) | -26 | 88 |
| 18 | ↓ | RuneScape (623) | -36 | 88 |
| 19 | ↑ | Drudge Report (26) | +25 | 85 |
| 20 | ↑ | Barack Obama (4) | +19 | 82 |

Movers

[RSS](#)
[MY Y!](#)

| Rank | Subject | 1-Day Move |
|------|---------------------------------------|------------|
| 1 | Kara DioGuardi | Breakout! |
| 2 | Brooke Mueller | Breakout! |
| 3 | Genie Francis | Breakout! |
| 4 | Goliath Grouper Fish | Breakout! |
| 5 | Hari Puttar | Breakout! |
| 6 | Tropical Storm Gustav | Breakout! |
| 7 | Mia Hamm | 33250% |
| 8 | Ellen Barkin | 26799% |
| 9 | Luciana Barroso | 20830% |
| 10 | Giant Squid | 16720% |
| 11 | Amanda Peet | 7688% |
| 12 | 2010 Chevrolet Camaro | 5363% |
| 13 | Ellen Adama | 5143% |
| 14 | Ellen Muth | 4893% |
| 15 | Cody Linley | 4855% |
| 16 | Jeffrey Ross | 4688% |
| 17 | Ellen Pompeo | 4372% |
| 18 | Ted McGinley | 4248% |
| 19 | Toni Braxton | 3394% |
| 20 | Susan Lucci | 2655% |

the buzz log

Recent Entries


[What's the Buzz: Creatures, Mysteries, and a Rumor](#)
[The Return of Anwar Ibrahim](#)
[Viva La Revolution, Part Deux](#)
[The Kennedy Effect](#)
[Jill Biden, Buzz Superstar?](#)
[What's the Buzz: A Cold Front Is Coming](#)
[Olympic Fever Continues](#)
[Post-Olympian Ambitions](#)
[Five Things We Learned This Weekend](#)
[The Buzz Week in Review](#)
[more >](#)

what's the buzz?

A subject's buzz score is the percentage of Yahoo! users searching for that subject on a given day, multiplied by a constant to make the number easier to read. Weekly leaders are the subjects with the greatest average buzz score for a given week.

For more detailed information, visit our [FAQ](#).

Twitter trend



Worldwide Trends · [Back to Discover](#)

[#NashVsChad](#)

[#LloreEscuchando](#)

[Male Vocalist of the Year](#)

[#cmaawards2014](#)

[#Whatsappbuchon](#)

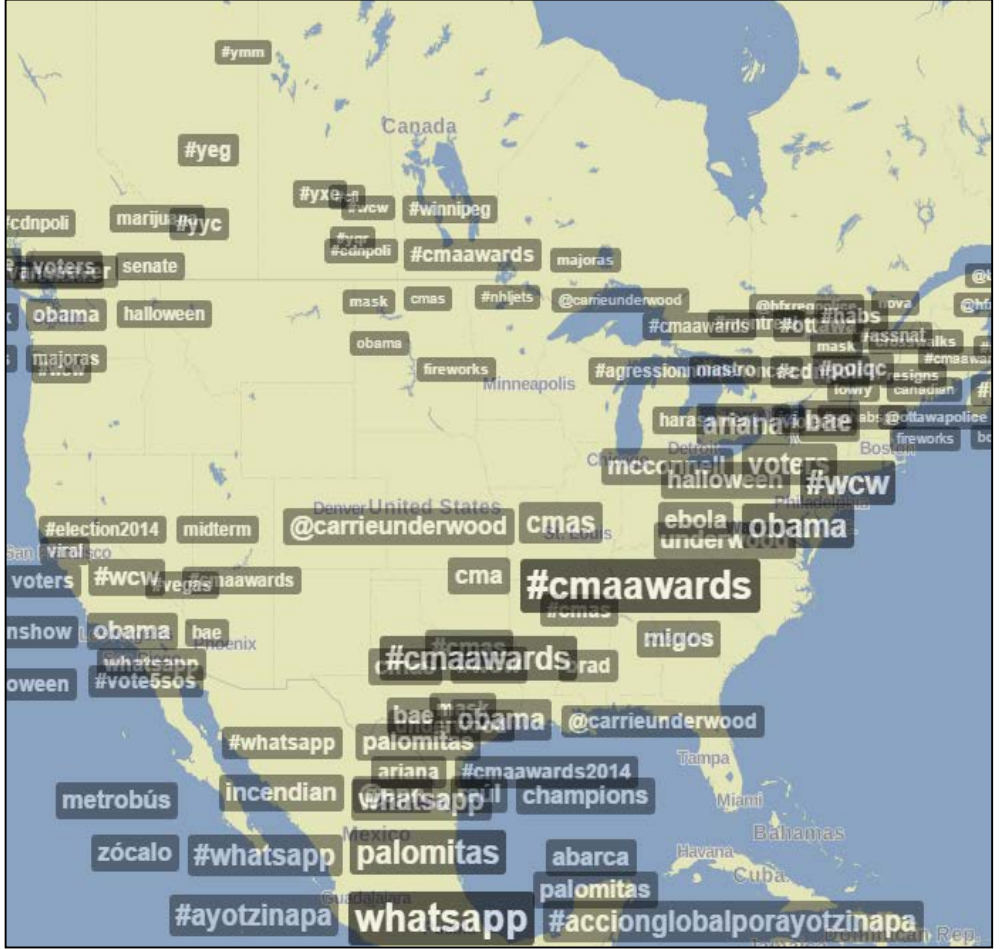
[Thomas Rhett](#)

[Blake and Miranda](#)

[#MasterChefPastayMisterio](#)

[Earl Thomas Conley](#)

[Donald Sloan](#)



<https://mobile.twitter.com/trends>

<http://trendsmap.com/>

<Retrieved on Nov 5, 2014>



- [illegible]

Sentiment Analysis

- Sentiment Analysis
 - Automatically determine tone in text: positive, negative or neutral
 - Typically uses collections of good and bad words
- There are sentiment word lists out there:
 - See http://neuro.imm.dtu.dk/wiki/Text_sentiment_analysis
- *Example*: Assume you are marketing analysts for Apple and what the feelings of the public are toward the iPhone 6.

Two Customer Reviews of iPhone6 from Amazon.com

2 of 2 people found the following review helpful

★★★★★ Go apple!!

By [Diamond](#) on October 20, 2014

LOVE my new phone. Going from the 5 it took me a while to get used to the size but it's **pretty amazing**. 🍌🍌🍌🍌

[Comment](#) | Was this review helpful to you?

11 of 16 people found the following review helpful

★★★★★ **Really disappointed in the design**

By [Pat Patterson](#) on October 15, 2014

As far as the function goes it's an Apple and it's great. As far as **design** it's **bad** (6 only). The screens glass is raised above the frame of the phone so it is probably the **weakest screen** ever!!! It has no built in protection and if you think a 25 dollar cover is going to protect your investment you will be sadly let down!

I have had my new phone for 17 days and I purchased a Pelican Voyager, comparable to the Otter Box Defender (not really!!!). Anyway I was on my patio last night and turn to go in the house and dropped my phone from about 30" it landed on in/outdoor carpet and the screen shattered!!! I will say that my patio is concrete and that the carpet was an area rug and it fell flat on its face, it should have had no issues with it but a bad design and a manufacturer in a hurry to get it on the market leave me to believe that Apple is just burning their loyal customers. Before this phone I had a 4S and it was built very well!!! My 4S fell off my truck at 30 miles per hour and lived another day. Apple added 200.00 to the cost and built a phone that is worth a heck of a lot less.

[Comment](#) | Was this review helpful to you?

Text Mining: Sentiment Analysis



Source: Singapore Institute of Manufacturing Technology

Text Clustering

| Cluster ID | Descriptive Terms | Frequency | Percentage |
|------------|--|-----------|------------|
| 1 | +return +power +find +double +account +function +work +text +head +parent +support +image +music +book +up ... | 295 | 18% |
| 2 | +brande +pay +producte +device +store +play +branda +version free +tablet +display +credit +good +price +fact ... | 401 | 24% |
| 3 | 'credit card' +card +credit us american free +download +buy +double +address +brande +memory bought +account +store ... | 26 | 2% |
| 4 | +flashcontent +javascript +onclick +alert +html +preset +scale +code +document +parent +length +catch +style +versionafox +log ... | 5 | 0% |
| 5 | +flash +support disappointed +productf1 +buy +producta +download +work +versiona bought +new +up +none +old +branda ... | 420 | 25% |
| 6 | 'battery life' +battery +charge +life battery hours charging +charger +power +read +time reading +old +the +great ... | 116 | 7% |
| 7 | +easy +great +love +producta +versiona books hd movies games reading +recommend +read +sound +music +happy ... | 411 | 25% |

Figure: Text clustering results of customer reviews for Kindle Fire HD 7" from Amazon.com (Nagarajan et al. 2012)

Part II. Terms and Approaches

Corpus & Lexicon

- **Corpus** - Body of text, singular. Corpora is the plural of this.
Example: A collection of medical journals.
- **Lexicon** - Words and their meanings.
 - Example: English dictionary.
 - Various fields will have different lexicons. For example: To a financial investor, the first meaning for the word "Bull" is someone who is confident about the market, as compared to the common English lexicon, where the first meaning for the word "Bull" is an animal.
 - As such, there is a special lexicon for financial investors, doctors, children, mechanics, and so on.

Bag of Words

- The most naïve approach
- Treats a document as a list of words
- For short sequences of text, this approach might not be sufficient
 - Man eats fish vs. Fish eats man
- Could be useful for longer document with much redundant information



Tokenization

- A sentence or data can be split into words using the method *word_tokenize()*

```
from nltk.tokenize import sent_tokenize, word_tokenize

data = "All work and no play makes jack a dull boy, all work and no play"
print(word_tokenize(data))
```

This will output:

```
['All', 'work', 'and', 'no', 'play', 'makes', 'jack', 'dull', 'boy', ',', 'all', 'work', 'and',
```

- A paragraph can be split into sentences using the method *sent_tokenize()*

```
from nltk.tokenize import sent_tokenize, word_tokenize

data = "All work and no play makes jack dull boy. All work and no play makes jack a dull boy."
print(sent_tokenize(data))
```

Outputs:

```
['All work and no play makes jack dull boy.', 'All work and no play makes jack a dull boy.']
```

Image Source: <https://pythonspot.com/tokenizing-words-and-sentences-with-nltk/>

Tokenization

```
from nltk.tokenize import sent_tokenize, word_tokenize

EXAMPLE_TEXT = "Hello Mr. Smith, how are you doing today? \
The weather is great, and Python is awesome. \
The sky is pinkish-blue. You shouldn't eat cardboard."

print(sent_tokenize(EXAMPLE_TEXT))
print(word_tokenize(EXAMPLE_TEXT))
```

Tokenization

```
from nltk.tokenize import sent_tokenize, word_tokenize

EXAMPLE_TEXT = "Hello Mr. Smith, how are you doing today? \
The weather is great, and Python is awesome. \
The sky is pinkish-blue. You shouldn't eat cardboard."

print(sent_tokenize(EXAMPLE_TEXT))
print(word_tokenize(EXAMPLE_TEXT))
```

```
['Hello Mr. Smith, how are you doing today?', 'The weather is great, and Python is awesome.', 'The sky is pinkish-blue.', 'You shouldn't eat cardboard.']
['Hello', 'Mr.', 'Smith', ',', 'how', 'are', 'you', 'doing', 'today', '?', 'The', 'weather', 'is', 'great', ',', 'and', 'Python', 'is', 'awesome', '.', 'The', 'sky', 'is', 'pinkish-blue', '.', 'You', 'should', "n't", 'eat', 'cardboard', '.']
```

Term-document matrix

- Most common form of representation in text mining is the *term - document* matrix
 - Term: typically a single word, but could be a word phrase like “data mining”
 - Document: a generic term meaning a collection of text to be retrieved
 - Can be large - terms are often 50k or larger, documents can be in the billions (www).
 - Can be binary, or use counts

Term-document matrix

Example: 10 documents: 6 terms

| | Database | SQL | Index | Regression | Likelihood | linear |
|-----|----------|-----|-------|------------|------------|--------|
| D1 | 24 | 21 | 9 | 0 | 0 | 3 |
| D2 | 32 | 10 | 5 | 0 | 3 | 0 |
| D3 | 12 | 16 | 5 | 0 | 0 | 0 |
| D4 | 6 | 7 | 2 | 0 | 0 | 0 |
| D5 | 43 | 31 | 20 | 0 | 3 | 0 |
| D6 | 2 | 0 | 0 | 18 | 7 | 6 |
| D7 | 0 | 0 | 1 | 32 | 12 | 0 |
| D8 | 3 | 0 | 0 | 22 | 4 | 4 |
| D9 | 1 | 0 | 0 | 34 | 27 | 25 |
| D10 | 6 | 0 | 0 | 17 | 4 | 23 |

$$D_1 = (d_{i1}, d_{i2}, \dots, d_{it})$$

- Each document now is just a vector of terms.

Term-document matrix

- We have lost all semantic content
- Be careful constructing your term list!
 - Not all words are created equal!
 - Words that are the same should be treated the same!
- Stop Words
- Stemming

Stop words

- Many of the most frequently used words in English are worthless in retrieval and text mining – these words are called *stop words*.
 - the, of, and, to,
 - Typically about 400 to 500 such words
 - For an application, an additional domain specific stop words list may be constructed
- Why do we need to remove stop words?
 - Reduce indexing (or data) file size
 - Stop words accounts 20-30% of total word counts.
 - Improve efficiency
 - stop words are not useful for searching or text mining
 - stop words always have a large number of hits



Stemming

- Techniques used to find out the root/stem of a word:
 - E.g.,

| | |
|---------|-------------|
| • user | engineering |
| • users | engineered |
| • used | engineer |
| • using | |
 - stem: use engineer

Usefulness

- improving effectiveness of retrieval and text mining
 - matching similar words
- reducing indexing size
 - combining words with same roots may reduce indexing size as much as 40-50%.

Basic stemming methods

- remove ending

- if a word ends with a consonant other than s, followed by an s, then delete s. (E.g. toys – toy)
- if a word ends in es, drop the s. (E.g. databases – database)
- if a word ends in ing, delete the ing unless the remaining word consists only of one letter or of th. (E.g. laughing – laugh)
- If a word ends with ed, preceded by a consonant, delete the ed unless this leaves only a single letter. (E.g. attached – attached)
-

- transform words

- if a word ends with “ies” but not “eies” or “aies” then “ies --> y.” (E.g. babies – baby)

Part of Speech tagging

- labeling words in a sentence as nouns, adjectives, verbs...etc.
- it also labels by tense, and more
- POS tag list:
 - CC coordinating conjunction
 - EX existential there (like: "there is" ... think of it like "there exists")
 - IN preposition/subordinating conjunction
 - JJ adjective 'big'
 - JJR adjective, comparative 'bigger'
 - JJS adjective, superlative 'biggest'
 - NNP proper noun, singular 'Harrison'
 - ...

Example:

- Time flies like an arrow.
- Fruit flies like a banana.



Term Frequency-Inverse Document Frequency (TF-IDF)

- TF-IDF counts in both the frequency of a term in a document and the frequency of documents with that term.
- For a given document d and term t

$$\text{TF}(d,t) = \frac{\text{\# times term } t \text{ appears in document } d}{\text{total number of terms in document } d}$$

$$\text{IDF}(t) = \log\left(\frac{\text{total number of documents}}{\text{\# documents containing term } t}\right)$$

$$\text{TF-IDF}(t,d) = \text{TF}(d,t) * \text{IDF}(t)$$

- Document with a relative high frequency for terms that are relatively rare overall -> high TF-IDF
- Document with a relative low frequency for terms that are shown up frequently overall -> low TF-IDF

Review Questions

Machine Learning - Specialty (MLS-C01)

Sample Exam Questions

- 1) **A Machine Learning team has several large CSV datasets in Amazon S3. Historically, models built with the Amazon SageMaker Linear Learner algorithm have taken hours to train on similar-sized datasets. The team's leaders need to accelerate the training process.**

What can a Machine Learning Specialist do to address this concern?

- A. Use Amazon SageMaker Pipe mode.
 - B. Use Amazon Machine Learning to train the models.
 - C. Use Amazon Kinesis to stream the data to Amazon SageMaker.
 - D. Use AWS Glue to transform the CSV dataset to the JSON format.
- 2) **A term frequency-inverse document frequency (tf-idf) matrix using both unigrams and bigrams is built from a text corpus consisting of the following two sentences:**

- 1. Please call the number below.
- 2. Please do not call us.

What are the dimensions of the tf-idf matrix?

- A. (2, 16)
- B. (2, 8)
- C. (2, 10)
- D. (8, 10)

Answers

- 1) A – Amazon SageMaker Pipe mode streams the data directly to the container, which improves the performance of training jobs. (Refer to this [link](#) for supporting information.) In Pipe mode, your training job streams data directly from Amazon S3. Streaming can provide faster start times for training jobs and better throughput. With Pipe mode, you also reduce the size of the Amazon EBS volumes for your training instances. B would not apply in this scenario. C is a streaming ingestion solution, but is not applicable in this scenario. D transforms the data structure.
- 2) A – There are 2 sentences, 8 unique unigrams, and 8 unique bigrams, so the result would be (2,16). The phrases are “Please call the number below” and “Please do not call us.” Each word individually (unigram) is “Please,” “call,” “the,” “number,” “below,” “do,” “not,” and “us.” The unique bigrams are “Please call,” “call the,” “the number,” “number below,” “Please do,” “do not,” “not call,” and “call us.” The tf-idf vectorizer is described at this [link](#).

Term Frequency-Inverse Document Frequency (TF-IDF)

[Example]

- Consider a document containing 100 words wherein the word cat appears 3 times.

The term frequency (i.e., tf) for cat =

- Now, assume we have 10 million documents and the word cat appears in one thousand of these.

idf =

Tf-idf weight =

Term Frequency-Inverse Document Frequency (TF-IDF)

[Example]

- Consider a document containing 100 words wherein the word cat appears 3 times.

The term frequency (i.e., tf) for cat is then $(3 / 100) = 0.03$.

- Now, assume we have 10 million documents and the word cat appears in one thousand of these.

$idf = \log(10,000,000 / 1,000) = 4$.

Thus, the Tf-idf weight is the product of these quantities: $0.03 * 4 = 0.12$.

Reference

- Book. Data Mining Techniques (3rd edition). Linoff & Berry.
- <http://www.tfidf.com/>
- <https://pythonspot.com/tokenizing-words-and-sentences-with-nltk/>