Twitter and Sentiment Analysis (text analytics)

Tool: Bag of words model, Logistic Regression, Classification And Regression Trees (CART), Random Forests.

The Analytics Edge: Sentiment analysis from massive amount of unstructured data can be automated using tools of analytics, which replace the more common method of polling users on their opinions. New media of communication, such as Twitter and Facebook, give companies and organizations a chance to understand sentiments using tools of analytics.

1 Overview (Twitter)

Twitter is a social networking and communication website established in March 2006. The service enables users to send and read short messages called "tweets", which were originally restricted to 140 characters (this limit was recently doubled for most languages). Twitter is one of the biggest social networks worldwide: as of April 2018, the company has more than 300 million users, a total revenue of about $ 2.5 billion, and an evaluation of over $ 20 billion. Twitter is not only used by celebrities to reach out to their followers, but also by companies to communicate with their customers, hear their thoughts, and understand trends. "Twitter mood" has been shown to have a predictive power on stock market prices. Twitter has been used to predict box office performance after movies have been released based upon how many times films are mentioned in tweets.

2 Sentiment Analysis

Sentiment Analysis refers to the use of text analytics, natural language processing, and computational linguistics to identify and extract subjective information in source materials. The basic task in sentiment analysis is to classify the *polarity* of a given text, that is, whether the opinion expressed in a document is positive, neutral, or negative (as illustrated in Figure 1). More advanced tasks look at emotional states, such as "angry" or "sad".

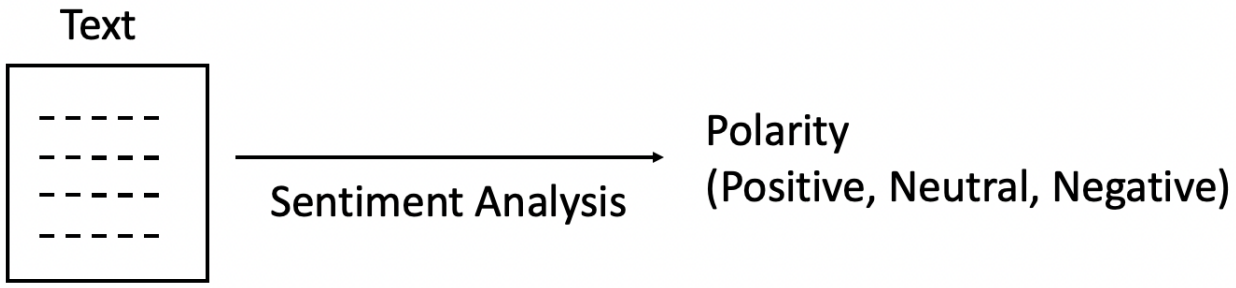

Figure 1: Sentiment analysis.

2.1 Working with data from Twitter

Twitter data can be collected using an interface called API or retrieved from specialized websites, such as `www.sentiment140.com`. Also R has a dedicated package (`TwitteR`) that can be used to import tweets directly from Twitter. To predict the sentiment of tweets, it is important to have information on the sentiments in the training dataset. There are a few ways this information can be obtained. For example, one can:

1. Carefully go through every tweet in the training set and provide a sentiment polarity for the tweets;
2. Use centralized work places such as Amazon Mechanical Turk, where small tasks are assigned to individuals who work remotely and do the sentiment categorization for a few tweets at a small price;
3. Leverage the information contained in emoticons.

Key Question: Is it possible to correctly predict (classify) the sentiment of a tweet based on the information contained in previous tweets?

2.2 Challenges in text analytics (tweets)

Tweets are textual data, typically with poor spelling (short forms) and use of non-traditional grammar—for example: *"U say that iphone 5S didnt bring anything new 2?"*. There is an additional source of complexity, namely the ambiguity in the English language that sometimes even humans cannot decipher. Consider the following examples:

- *"John saw the man on the mountain with a telescope"*. Who has the telescope? John, the man on the mountain, the man?
- *"John and Mary took two trips around France. They were both wonderful"*. They refers to John and Mary or to the two trips?
- *"Medicine helps dog bite victims"*. Does the medicine help the dog to bite victims or does it help the victims who are bitten by the dog?

2.3 Relation with Turing test

The problem of classifying tweets with a computer program is part of the broader problem of understanding and analyzing human language as it is spoken.

In this regard, Alan Turing introduced the *Turing test*, which is described in his 1950 paper "Computing machinery and Intelligence". This is a test of a machine's ability to exhibit intelligent behavior that is undistinguishable from a human. Turing proposed that the human evaluator would judge between natural language conversations with a human and a machine. If the evaluator cannot reliably tell the machine from the human using a text only channel, then the machine passes the test (see Figure 2).

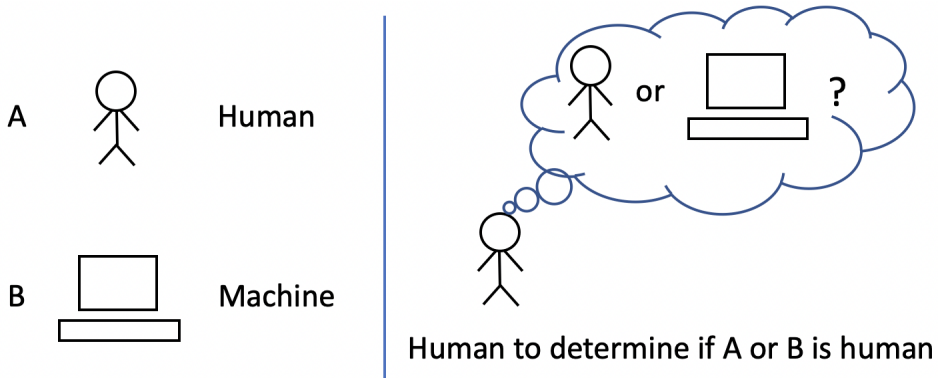

Figure 2: Turing test.

3 Summary

Data: large unstructured datasets containing tweets and a corresponding value determining the class/polarity of each tweet.

Model: A classification model (e.g., logistic regression, CART, Random Forest) that predicts the sentiment of a tweet based on key words contained in the tweet itself.

Value and Decision: the model replaces the option of polling users on their opinions and allows exploiting the information contained in tweets.

4 Bag of words model

Bag of words is a simple approach to represent text in a computer program. In particular, the text is represented as a set (bag) of words, disregarding the grammar and word order—but keeping multiplicity. The so-called *document-term matrix*, for a problem with N documents and B terms, is illustrated in Figure 3. Each element in the document-term matrix represents a measure of the frequency of occurrence of the terms (words) in the document.

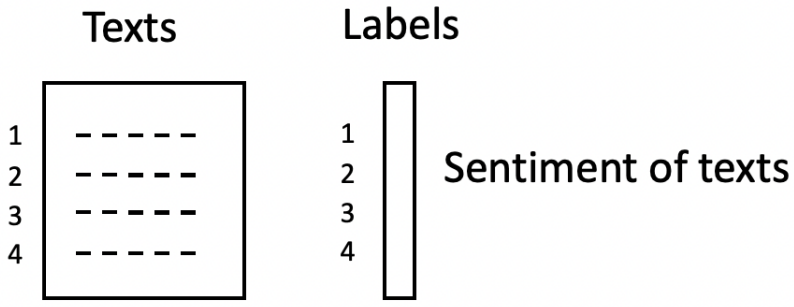

Figure 3: Bag of words approach.

Let's consider the following example:

1. *John likes to watch movies. Mary likes movies too.*
2. *John also likes to watch football.*

In this example, we have two documents and nine words, which we can organize in the following matrix:

$$\begin{pmatrix} & \text{John} & \text{likes} & \text{to} & \text{watch} & \text{movies} & \text{also} & \text{Mary} & \text{football} & \text{too} \\ \text{Doc1} & 1 & 2 & 1 & 1 & 2 & 0 & 1 & 0 & 1 \\ \text{Doc2} & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

4.1 Pre-processing

Data are often unstructured, and hence pre-processing needs to be done. For example:

1. Stopwords are words that are filtered out in the processed text. These typically refer to the most common words in the language, such as *"the"*. Example: *"Dharma and Greg rock"*. In a bag of words model, the stopword *"and"* would be removed. However, *"Dharma and Greg rock"* might refer to the show *"Dharma and Greg"*, where it is part of the same name. In these cases, it is possible to use an n-gram, namely a contiguous sequence of n items (see Google Ngram Viewer). We will not use these features in this work, though.
2. Removing punctuations, converting upper case to lower case. These are other types of preprocessing commonly used.
3. Stemming: the Porter stemming algorithm is used to remove inflected words to their word stem, base or root form. Example: *"cats"* should be identified with the root *"cat"*. Martin Porter in 1980 invented the Porter stemmer, one of the most common algorithms for stemming in English. Example: *"revive"* and *"revival"* would be stemmed to *"reviv"*.

4.2 Back to R!

- Pre-processing
  In R, the pre-processing steps can be carried out with the `tm` (text mining) package.
  `twitter <- read.csv("twitter.csv",stringsAsFactors=FALSE)` – Load data.
  `corpus <- Corpus(VectorSource(twitter$tweet))` – Create a corpus, which represents a collection of documents
  - Convert text to lower case
    ```
    corpus <- tm_map(corpus, function(x) iconv(enc2utf8(x), sub = "byte"))
    corpus <- tm_map(corpus, content_transformer(function(x) iconv(enc2utf8(x), sub = "bytes")))
    corpus <- tm_map(corpus, content_transformer(tolower))
    ```
  - Remove stopwords
    ```
    corpus <- tm_map(corpus,removeWords,stopwords("english"))
    ```
  - Remove punctuation
    ```
    corpus <- tm_map(corpus,removePunctuation)
    ```
  - Stemming
    `corpus <- tm_map(corpus,stemDocument)` – using package `SnowballC`

  - Create DTM
    `dtm <- DocumentTermMatrix(corpus)`
  - Removing sparse terms
    `dtm <- removeSparseTerms(dtm,0.995)`

- Preparing the DTM for model learning
  `twittersparse <- as.data.frame(as.matrix(dtm))`
  `colnames(twittersparse)<-make.names(colnames(twittersparse))`
  Basic visualization with package `wordcloud`

- Train and test a classifier