Concepts in Text Mining

TEXT
PRE-PROCESSING

Yoonsang Kim
Hy Tran

Nov 25, 2013
Purpose

- Clustering and classifying models require structured data: a data frame or matrix containing
  - Features or words as columns: n-dimensional feature vector
  - Document as rows: e.g., a tweet is a document
- To analyze unstructured text data, one has to extract info from text and turn that into numerically structured data matrix, which is a representation of documents
- Some Natural Language Processing (NLP) techniques are used
- Provide features for feature selection
  - Informative features can improve classifiers’ performance
Feature & Corpus

• Feature:
  Given a set of terms, feature is a function of the term(s) in document
  - Presence/absence of terms
  - Term frequency
  - Term frequency-inverse document frequency
  - n-gram word, etc.

• Corpus:
  A set of text documents + meta data
Overview

- Useful procedures for pre-processing
  - Tokenization and N-gram
  - Part of Speech tagging
  - Text transformation
    - Convert to lowercase, remove punctuation, etc.
    - Stemming, lemmatizing, etc.
  - Singular value decomposition
  - There are more....
Tokenization & N-grams

- Breaks a stream of text up into words, phrases, or symbols called *tokens*
- Turn text into vector of tokens
  - “Have a good day!” → “Have”, “a”, “good”, “day!”
  - Tokens serve as units for further processing
- N-gram is a sequence of $n$ adjacent tokens
  - Word n-gram: e.g. bigram “Have a”, “a good”, “good day!”
  - Character n-gram: e.g. trigram “Hav”, “ave”, “ve”, “e a”, ...
  - More context than single token
Tokenization & N-grams

- Tokenization is a necessary step for text mining
- N-gram is often not good enough by itself
  - Many modern applications don’t rely just on n-gram
  - Fast and simple, serve as basic features
  - Combined with other methods to add more features
- N-gram of words is useful for explorative analysis
  - Easier for human to interpret than character grams
- N-gram of characters can outperform words for certain classification tasks
Part of Speech Tagging

- Use machine learning to tag each word in a sentence according to a particular part of speech
  - “This is a sample sentence” → This/D T is/V BZ a/D T sample/N N sentence/N N
  - Many English POS taggers follow *Penn Treebank convention*

- Requires original context to recognize POS
  - The same word can have different POS depend on context
  - Common approach is to look at nearby words to help tag

- Different POS has different implication
  - Nouns can be important indicators of *topic*
  - Adjectives can be important indicators of *sentiment*
Text Transformation

- To remove contexts that don’t convey much information and to speed up modeling (via dimension reduction)
- Performed after other NLP methods (as those depend on context)
- Convert to lowercase
  - Change uppercase to lowercase
  - The amount of context lost is small
  - The speed gain is often large
Text Transformation

- **Remove punctuation**
  - [ ], . ; ? @ & “ * ( ) ... etc.
  - Have high frequencies and don’t indicate topics

- **Remove numbers**
  - Unlikely that the exact number appears several times

- **Remove stop words**
  - a, about, an, the, and, we, this, on, it, in, with, etc.
  - Have too high frequencies and no information about topic

- **Remove these can improve topic modeling and classification**
**Text Transformation**

- **Stemming**
  - Convert words to a base form. Different morphological variants of a word, all of which have similar or identical semantics, are considered as equivalent.
  - Suffix removal: -ly, -ed, -ing, -er, etc.
  - “fishing”, “fish”, “fisher” → “fish”
  - Fast: simple search and replace algorithm for English
  - Increases the recall and reduces the size of feature vector. Lost of context is small
  - Can hurt precision: “cop” and “cope” have the stem “cop”

- **Stemming is helpful in**
  - Discovering big picture and theme
  - Keywords search (Google uses this in their search engine)
• **Lemmatization**
  
  - Process of determining the *lemma* for a given word
  - Closely related to stemming, but involve understanding context and determining the POS of a word
  - “better” and “good”: stemming will miss this link
  - “meeting” has different meaning as a verb and a noun
  - Capable of grouping by synonym: “mom” and “mother”
  - Slow: better for last stage or fine grain analysis
Singular Value Decomposition

- $A = \text{term-document matrix of } m \times n \text{ dimension}$
  - $m = \# \text{ of terms/features in documents}$
  - $n = \# \text{ of documents in corpus}$

- **SVD**: factor $A$ into the product of 3 matrices
  
  $$A = U \Sigma V^T$$

  - $U = m \times r$ ; its columns called left singular vector
  - $\Sigma = r \times r$ ; diag matrix with elements $d_1 >, ..., > d_r$ called *singular values*
  - $V^T = r \times n$ ; its rows called right singular vector
  - $r = \text{rank}(A) = \# \text{ of linearly independent columns/rows in } A$
• Delete some insignificant dimensions in the transformed “concept” space to approximate $A$
• Use $k$ largest singular values in $\Sigma$, so that $A_k = U_k \Sigma_k V_k^T$

The truncated $A$ captures most of the important underlying structures in the association of terms and documents
Singular Value Decomposition

\[ A = \begin{pmatrix}
C_1 & C_2 & C_3 & C_4 & C_5 & m_1 & m_2 & m_3 & m_4 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix} \]

\begin{align*}
& \text{human} \\
& \text{interface} \\
& \text{computer} \\
& \text{user} \\
& \text{system} \\
& \text{response} \\
& \text{time} \\
& \text{EPS} \\
& \text{survey} \\
& \text{trees} \\
& \text{graph} \\
& \text{minors} \\
\end{align*}

\[ U = \begin{pmatrix}
0.22 & -0.11 & 0.29 & -0.41 & -0.11 & -0.34 & 0.52 & -0.06 & -0.41 \\
0.20 & -0.07 & 0.14 & -0.55 & 0.28 & 0.50 & -0.07 & -0.01 & -0.11 \\
0.24 & 0.04 & -0.16 & -0.59 & -0.11 & -0.25 & -0.30 & 0.06 & 0.49 \\
0.40 & 0.06 & -0.34 & 0.10 & 0.33 & 0.38 & 0.00 & 0.00 & 0.01 \\
0.64 & -0.17 & 0.36 & 0.33 & -0.16 & -0.21 & -0.17 & 0.03 & 0.27 \\
0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & 0.28 & -0.02 & -0.05 \\
0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & 0.28 & -0.02 & -0.05 \\
0.30 & -0.14 & 0.33 & 0.19 & 0.11 & 0.27 & 0.03 & -0.02 & -0.17 \\
0.21 & 0.27 & -0.18 & -0.03 & -0.54 & 0.08 & -0.47 & -0.04 & -0.58 \\
0.01 & 0.49 & 0.23 & 0.03 & 0.59 & -0.39 & -0.29 & 0.25 & -0.23 \\
0.04 & 0.62 & 0.22 & 0.00 & -0.07 & 0.11 & 0.16 & -0.68 & 0.23 \\
0.03 & 0.45 & 0.14 & -0.01 & -0.30 & 0.28 & 0.34 & 0.68 & 0.18 \\
\end{pmatrix} \]

\[ \Sigma = \begin{pmatrix}
3.34 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2.54 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.35 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1.64 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.50 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1.31 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.56 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.36 \\
\end{pmatrix} \]

Web data Mining, Ch 6.7
Liu, 2011
Training & Validation

- Training set: to fit the model
- Validation set: to estimate prediction error for model selection
- Test set: assessment of the generalization error of the final chosen model

- Difficult to give a general rule on how much training data is enough
**K-Fold Cross-Validation**

- Use part of the available data to fit the model, and a different part to test it
- Useful when data are scarce
- Split data into $K$ roughly equal-sized parts: $K = 5$ or $10$
- Obtain average prediction error


Penn Treebank POS
http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html
Reference

