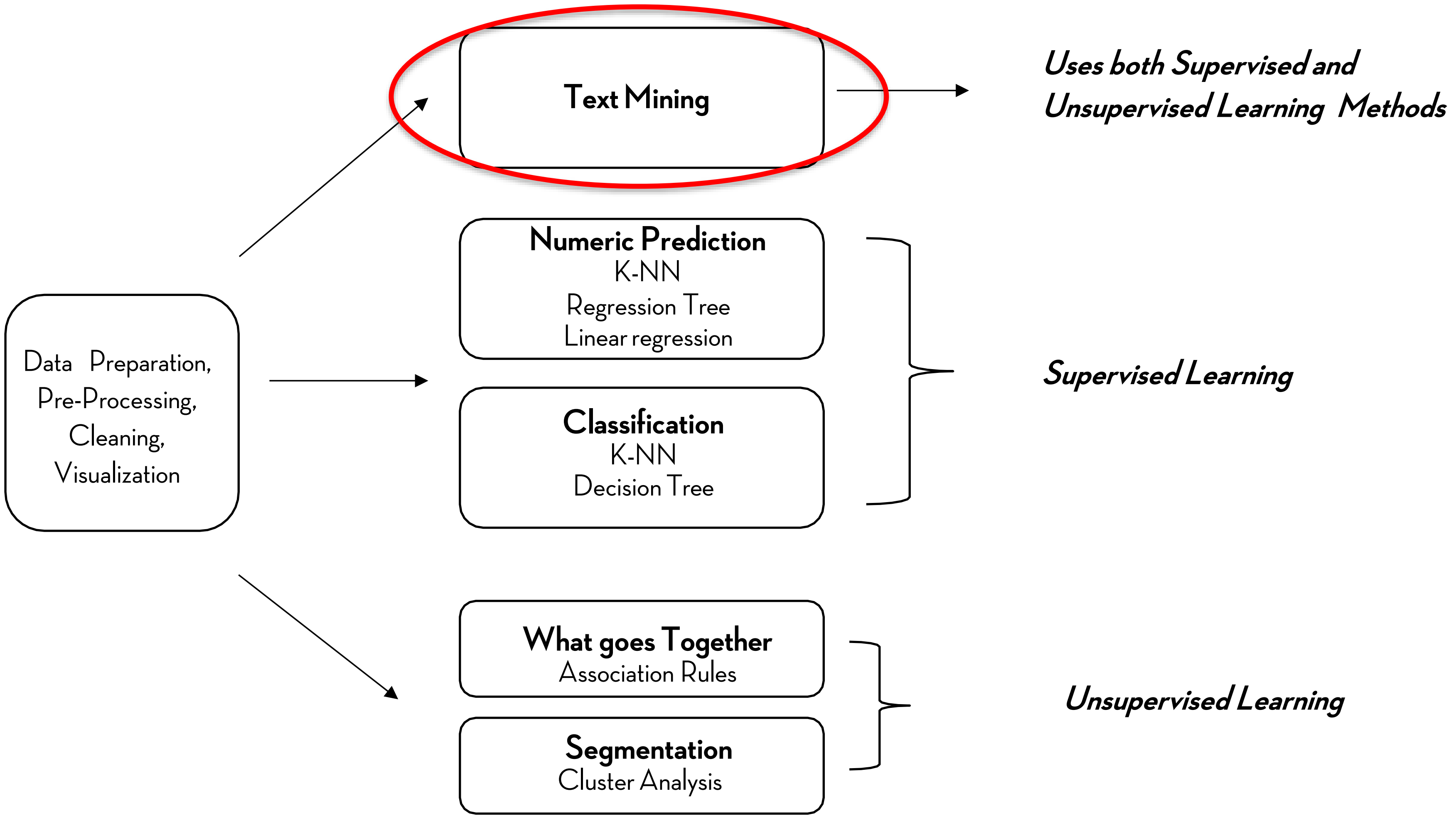
CARLSON SCHOOL OF MANAGEMENT
UNIVERSITY OF MINNESOTA

IDSC 4444 (004)
Text Mining

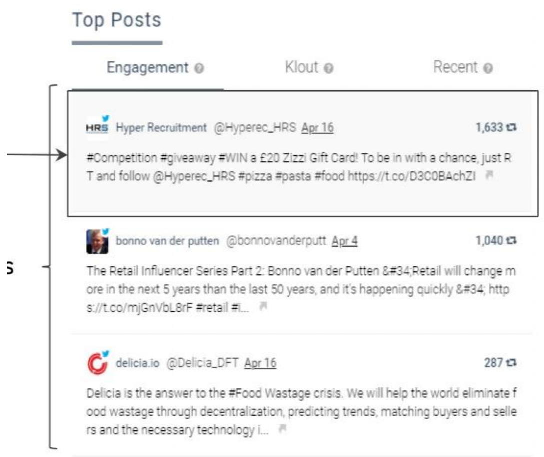
Zihong Huang
Information & Decision Sciences
Carlson School of Management
huan0707@umn.edu

An Overview


Data Preparation,
Pre-Processing,
Cleaning,
Visualization
Text Mining
Uses both Supervised and
Unsupervised Learning Methods
Numeric Prediction
K-NN
Regression Tree
Linear regression
Classification
K-NN
Decision Tree
Supervised Learning
What goes Together
Association Rules
Segmentation
Cluster Analysis
Unsupervised Learning

Text Mining

- "Set of methods/techniques that model and structure the information content of textual sources for business intelligence, exploratory data analysis, research and so on"
- We generate a lot of text data: emails, social media posts, news feeds, tweets, online reviews…
- Text Mining or Text Analytics help us deriving high-quality information from text



How Is Textual Data Different?

- The data is NOT numerical anymore
- Text data is **unstructured**: cannot be directly interpreted in a tabular format
  - Structure in text is linguistic: words, sentences, paragraphs...
- Text is easy for humans to interpret with context and implied meaning
  - Difficult for machines and algorithms to process
- Text information is very noisy, can contain typos, grammatical mistakes, etc..
- Text can be subjective: even when read by humans, may be interpreted in slightly different ways
- We need methods to understand, label, organize text to derive information and insights from it, in an automatic way

Text Mining: Definitions

- **Document:** A piece of text you are interested in analyzing
  - Example: an email, a tweet, a review
- **Corpus**: a collection of documents (usually of the same type) that comprise our dataset
  - Examples:
    - ✓ All the tweets in a given week
    - ✓ All the reviews about a given restaurant
- **Word Token**: A single word
- **Vocabulary:** the set of unique word tokens that make up your dataset

Text Mining: Example

Corpus:
the collection of reviews for a product

Most Recent Customer Reviews

Easy way to build a website!
Nice entry-level website builder manual!
Published 5 days ago by Brian

Five Stars
Good book.
Published 1 month ago by Marc Lawrence

and easy to read
Clear, concise, and easy to read. Perfect for anyone who wants to learn about web design, but doesn't know where to begin.
Published 1 month ago by Wanda Skoczylas

It was is very easy to understand
This book was extremely helpful for me. I had no experience whatsoever when I began to create my website. Read more
Published 7 months ago by Cindy Beaton

Incredibly easy to understand the fundamentals of web design
Incredibly easy to understand the fundamentals of web design. It is at the top of my favorite list. The book itself is designed in a way that makes it enjoyable to comprehend the... Read more
Published 7 months ago by Susan Lesko

A very good starter book for web design
A very good starter book for web design. Good coverage of many details that can speed a new web designer on their way to a good site design
Published 8 months ago by Randy

Document: an individual review

Visualizing Text: Word Cloud

- One of the easiest text representation tool you may use: Word Cloud
- Visualize the frequent words in a corpus/document: the size of the word is proportional to the relative word token frequency
- Used to get a very quick representation of the most prominent terms, from which one may infer general topic/sentiment

Structuring Text – Bag of Words

- How to "structure" text in your Corpus:
  - **Term Document Matrix**
- Each row is a document (example, one review)
  - Each document is comprised of different words
  - Treat each word as an "attribute"
    - ✓ Unique words are the attributes
- Each element in the matrix reflects the importance of a word in a document as numeric weight
- Weights can be defined in different ways:
  - Binary
  - Frequency
  - TF-IDF

	Word1	....	Word N
Doc 1	$W_{1,1}$		
....			
Doc M	$W_{1,M}$		$W_{N,M}$

Term Document Matrix
Cell $W_{N,M}$ tells us how does word N matter to Doc M

Bag of Words

Document	Text
D1	Welcome to Data Analytics
D2	Data analysts study data.
D3	Data Mining finds patterns from data.

Vocabulary = { analysts, analytics, data, finds, from, mining, patterns, study, to, welcome}

- ❑ Assume you have a Corpus composed by 3 documents. We want to transform it into a Term Document Matrix.

BINARY WEIGHTING:

- ❑ W = 1 if the word is present in the document, 0 otherwise
- ❑ Even if the same words appears more than one time in the same document, the weight will still be 1
- ❑ Can enable document similarity comparison
  - What distance metric would you use?

Doc	analyst	analytics	data	finds	from	mining	patterns	study	to	welcome
D1	0	1	1	0	0	0	0	0	1	1
D2	1	0	1	0	0	0	0	1	0	0
D3	0	0	1	1	1	1	1	0	0	0

Bag of Words

Document	Text
D1	Welcome to Data Analytics
D2	Data analysts study data.
D3	Data Mining finds patterns from data.

Vocabulary = { analysts, analytics, data, finds, from, mining, patterns, study, to, welcome}

FREQUENCY WEIGHTING:

- ❑ Count how many times a given word appears in the document
- ❑ The Term Document Matrix is now called **Term Frequency Matrix**
- ❑ Any numeric distance (Euclidian, Manhattan, Max- coordinate) can be applied to find document similarity

Doc	analyst	analytics	data	finds	from	mining	patterns	study	to	welcome
D1	0	1	1	0	0	0	0	0	1	1
D2	1	0	**2**	0	0	0	0	1	0	0
D3	0	0	**2**	1	1	1	1	0	0	0

TF-IDF Weighting

- **Term Frequency – Inverse Document Frequency**
- Some words appear many times in many different documents. These are less important or informative than words that appear only in a few.
- Words that only appear in a few documents effectively distinguish those documents from the rest, and therefore should bear more weight in representing those documents
- Example: based on the matrix before, the word "data" appears in all the 3 documents while "mining" only appears in one (D3). The word "mining" should therefore be given more weight in representing D3.

TF-IDF Weighting

- Term Frequency (TF): the number of times the word w appears in document D
- Inverse Document Frequency **(IDF) = log(N / n_w)** where N is the total number of documents in the corpus, n_w is the total number of documents that contain w
  - Usually the natural log or log in base 10 are used in the calculations
- Final weight TD-IDF(w, D) = TF x IDF
- Higher TF, means a word appears more **frequently** in a document; Higher IDF means a word appears more **uniquely** in a document
- A word is generally more important to a document if it has high TF and high IDF

Bag of Words

Document	Text
D1	Welcome to Data Analytics
D2	Data analysts study data.
D3	Data Mining finds patterns from data.

Vocabulary = { analysts, analytics, data, finds, from, mining, patterns, study, to, welcome}

TF-IDF WEIGHTING:

- ❑ Each cell: TF * IDF = TF * $\log(N/n_w)$
- E.g., word "data" in D2: 2 * log(3/3) = 0
  - If the word appears in all the documents, log(1) = 0
- E.g., "analysts" in D2: 1 * log(3/1) = 1.1 (using natural log)
  - If the word appears in few document, log() is going to be a positive number
- This representation is called the TF-IDF matrix

Doc	analyst	analytics	data	finds	from	mining	patterns	study	to	welcome
D1	0	1.1	0	0	0	0	0	0	1.1	1.1
D2	1.1	0	**0**	0	0	0	0	1.1	0	0
D3	0	0	**0**	1.1	1.1	1.1	1.1	0	0	0

Bag of Words

- Transform your Corpus into a Term Document Matrix
  - The order of the words does not matter
  - The translation of text to matrix form removes sentence structure and associated information
  - All that matters is whether a given word is present or absent in the document
- The TDM (Text Document Matrix) is often <u>very sparse</u>
  - Sparse means that a lot of the elements of a matrix may be zeros
  - This is due to the fact that not all the words will always appear in all the documents

Pre-Processing

- In the examples used before, we were simply taking the text of each document as it is and dividing it into attributes/grams
- Nevertheless:
  - Should words such as “the”, “a”, “in”, “of” be counted as separate words in the TDM?
  - Should words such as analyst, analytics, analysis and analyzing be in different columns?
  - Should DATA and data be considered as two different words?
- Text-Mining requires a good amount of pre-processing

Pre-Processing

- ❑ Starting from the row text, severally pre-processing steps are typically performed, before constructing the TDM
- ❑ **Tokenization**: break down each document into single word tokens (separated by whitespaces)
- ❑ **Lower-casing**: transform each word in lower-case
- ❑ **Stop-words-removal**: remove "filler" words
  - Words such as articles, prepositions, do not carry as much value as nouns, adjectives, etc..

Pre-Processing

- **Stemming**: reducing words to the etymological roots
  - E.g., {"engineer", "engineering", "engineered"} → "engineer"

- **Punctuation tagging**: identify punctuation marks and special characters as attributes
  - Example: #hashtags

Exploratory Text Analytics

- Once we have pre-processed the text, and transformed the data into TDM, what else can we do?
- **Association Rules:**
  - We can find interesting relationships among terms and phrases
    - Each document can be thought of as a transaction
    - Each word can be thought of as an item
    - Example:
      - {welcome, data, analytics}
      - {data, analytics, helpful, deriving, insights}
  - Use all the metrics we have seen for association rules

Exploratory Text Analytics

❑Cluster analysis:

- Explore whether the documents can be naturally clustered into groups
- Useful to see whether documents naturally cluster into "themes" or "topics"
- Hierarchical clustering and k-Means can be applied to this context – once we measure "distance" between documents, can apply these methods as usual
- But how do we measure distance between documents?
  - ✓ Once we get the TDM all the usual distance metrics will apply: Euclidian, Manhattan, Max-Coordinate
  - ✓ But, there is one more specifically suited for text

Cosine Similarity

- **Cosine Similarity** is well suited to measure similarity between documents
  - Suppose there are two documents **d1(w11, … ,w1n)** and **d2(w21, … ,w2n).**
  - Each document can be represented by a vector of words. The words are the dimensions of the document
  - The similarity between two documents, d1 and d2, can be expressed as:

$$Sim(d_1, d_2) = \frac{\sum_{i=1}^{N} w_{1i} * w_{2i}}{\sqrt{\sum_{i=1}^{N} w_{1i}^2} * \sqrt{\sum_{i=1}^{N} w_{2i}^2}}$$

- Where the w are the weights from the TDM, and N are the number of words in a given document.

Cosine Similarity: Example

$$Sim(d_1, d_2) = \frac{\sum_{i=1}^{N} w_{1i} * w_{2i}}{\sqrt{\sum_{i=1}^{N} w_{1i}^2} * \sqrt{\sum_{i=1}^{N} w_{2i}^2}}$$

DOC	data	analytics	prediction
D1	2	1	1
D2	1	0	2
D3	3	0	0

- Sim(D1, D2) = ((2*1) + (1*0) + (1*2)) / [sqrt(2^2 + 1^2 + 1^2) * sqrt(1^2 + 0^2 + 2^2)]

  = 4 / [sqrt(6)*sqrt(5)] = 4/ (2.45*2.23) = 0.73

- Sim(D2, D3) = 3/ (2.23 * 3) = 0.447

- Sim(D1, D3) = 6 / (2.45 * 3) = 0.816

- NOTE: since in text mining the values of the matrix are never negative, Cosine is between 0 and 1, where 1 means the two documents are the same. So the higher the cosine similarity, the more similar the two documents

Text Classification

- **Classifying text into existing categories**
  - Categories can be topics, language, sentiment, etc..
- How to classify text?
  - Transform the text into the TDM format
  - Then, use one of the classification algorithm we have seen (example, k-NN)
- Example: Sentiment Analysis
  - Use of text analysis to systematically identify and extract affective states and subjective information from text
    - ✓ Example: Classify the text into "positive" or "negative" sentiment (identify the *polarity* of a given text)

Text Classification

- Once "classified", you can use text as attribute in all (numeric) prediction algorithms
- Example:
  - You want to predict company stock prices; among the usual financial variables, you can use Tweets from company's employees as an attribute to capture the "mood" around a company

ELSEVIER

Journal of Computational Science
Volume 2, Issue 1, March 2011, Pages 1-8

Twitter mood predicts the stock market

Johan Bollen [a, 1], Huina Mao [a, 1], Xiaojun Zeng [b]

Show more

https://doi.org/10.1016/j.jocs.2010.12.007

Get rights and content

Abstract

Behavioral economics tells us that emotions can profoundly affect individual behavior and decision-making. Does this also apply to societies at large, i.e. can societies experience mood states that affect their collective decision making? By extension is the public mood correlated or even predictive of economic indicators? Here we investigate whether measurements of collective mood states derived from large-scale Twitter feeds are correlated to the value of the Dow Jones Industrial Average (DJIA) over time. We analyze the text content of daily Twitter feeds by two mood tracking tools, namely OpinionFinder that measures positive vs. negative mood and Google-Profile of Mood States (GPOMS) that measures mood in terms of 6 dimensions (Calm, Alert, Sure, Vital, Kind, and Happy). We cross-validate the resulting mood time series by comparing their ability to detect the public's response to the presidential election and Thanksgiving day in 2008. A Granger causality analysis and a Self-Organizing Fuzzy Neural Network are then used to investigate the hypothesis that public mood states, as measured by the OpinionFinder and GPOMS mood time series, are predictive of changes in DJIA closing values. Our results indicate that the accuracy of DJIA predictions can be significantly improved by the inclusion of specific public mood dimensions but not others. We find an accuracy of 86.7% in predicting the daily up and down changes in the closing values of the DJIA and a reduction of the Mean Average Percentage Error (MAPE) by more than 6%.

Questions?