INTRODUCTION TO NATURAL LANGUAGE PROCESSING AND APPLICATIONS

DATA 602

Dr. Tony Diana
tonydian@umbc.edu

What is Natural Language Processing or NLP?

- It is one of the fastest growing areas in Machine Learning
- The basic objective of NLP is to transform **unstructured text** data into **knowledge** (structured data) and use that knowledge to improve
  - Interactions among humans (*sentiment analysis*), and, also
  - Exchanges between humans and machines (*chatbots*)
- NLP spans multiple disciplines. Here are some of them:
  - **Machine Learning**: Almost any NLP process is built upon a Machine Learning Systems
  - **Language and Grammar**: Understanding how a language and general grammar rules work
  - **General Artificial Intelligence**: Knowledge about rule-based systems and knowledge-based systems
  - **Statistics**: Building language models and measuring the accuracy of predictive models
  - **Algebra**: Matrix operations (CountVectorizer, Bag of Words)
- NLP can be defined as follows

> *"Natural language processing* is an area of research in computer science and artificial intelligence (AI) concerned with processing natural languages such as English or Mandarin. This processing generally involves translating natural language into data (numbers) that a computer can use to learn about the world. And this understanding of the world is sometimes used to generate natural language text that reflects that understanding," Cole Howard (2019: 11)

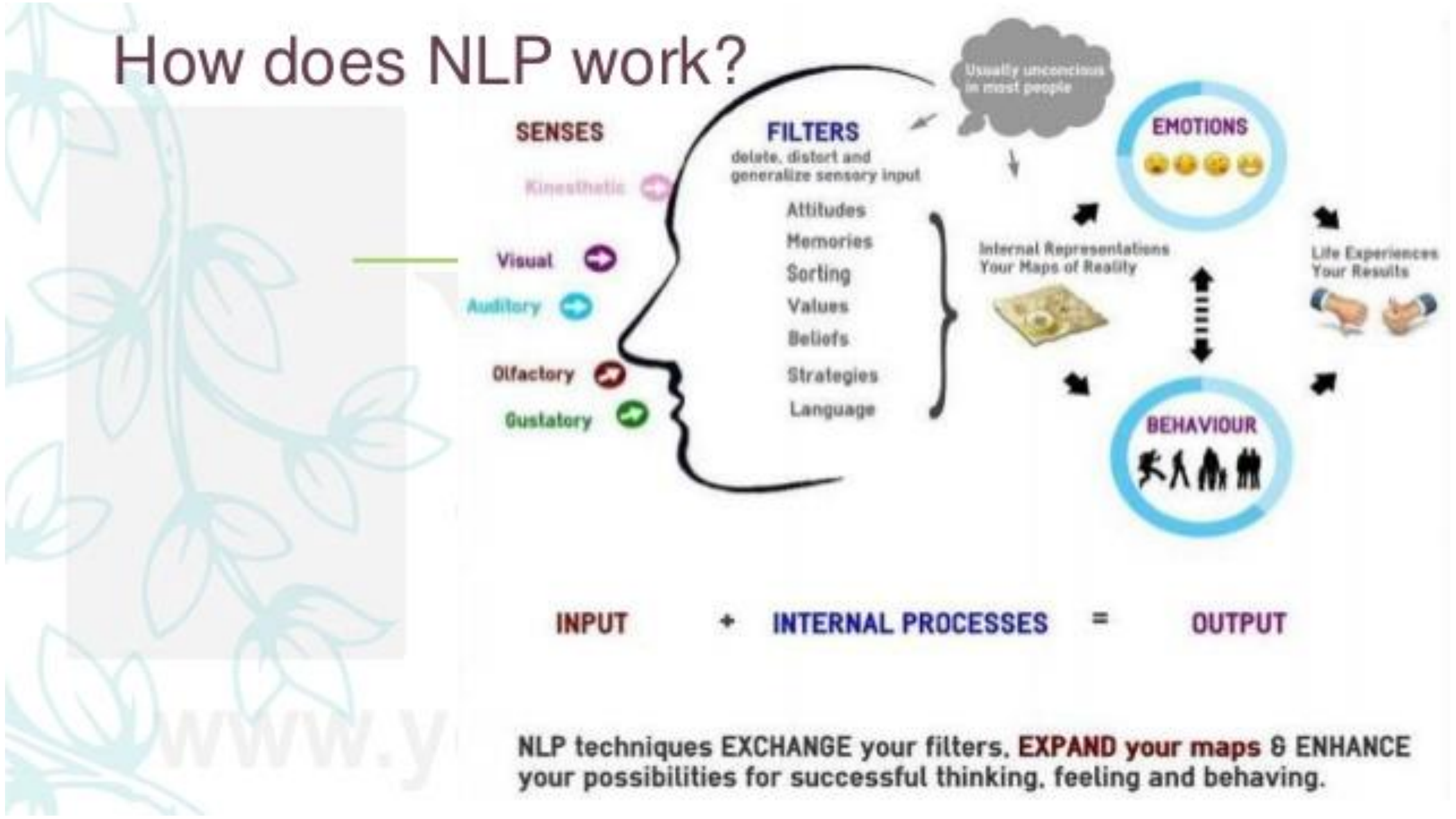
Introduction to Natural Language Processing (Cont.)

What are the Challenges of NLP?

- **NLP** deals with extremely diverse and dynamic languages that continually evolve
- Here are some of the challenges:
  - **Splitting text into sentences.** Not all sentences start with a capital letter and end with a period
  - **Ambiguity**. Language can be figurative, ambiguous, and/or sarcastic:
    - How is the computer supposed to know you mean 'smart' when you use the word 'sharp'?
    - How can the computer interpret the following sentence accurately: *I saw Mary on top of the hill with a telescope*? Did you see her with a telescope, or did you notice she had a telescope in her possession?
  - **Emotions**. They are hard to define, identify, and measure.
    - How do you interpret the following sentence: 'The best I can say about this product is that it was definitely interesting.' Is it sarcastic or a true statement?
  - **Co-Reference**. It is a relationship between two words or phrases in which **both refer to the same person** or **thing** and one stands as a **linguistic antecedent** of the other, as the two pronouns in 'She taught herself but not in 'She taught her'

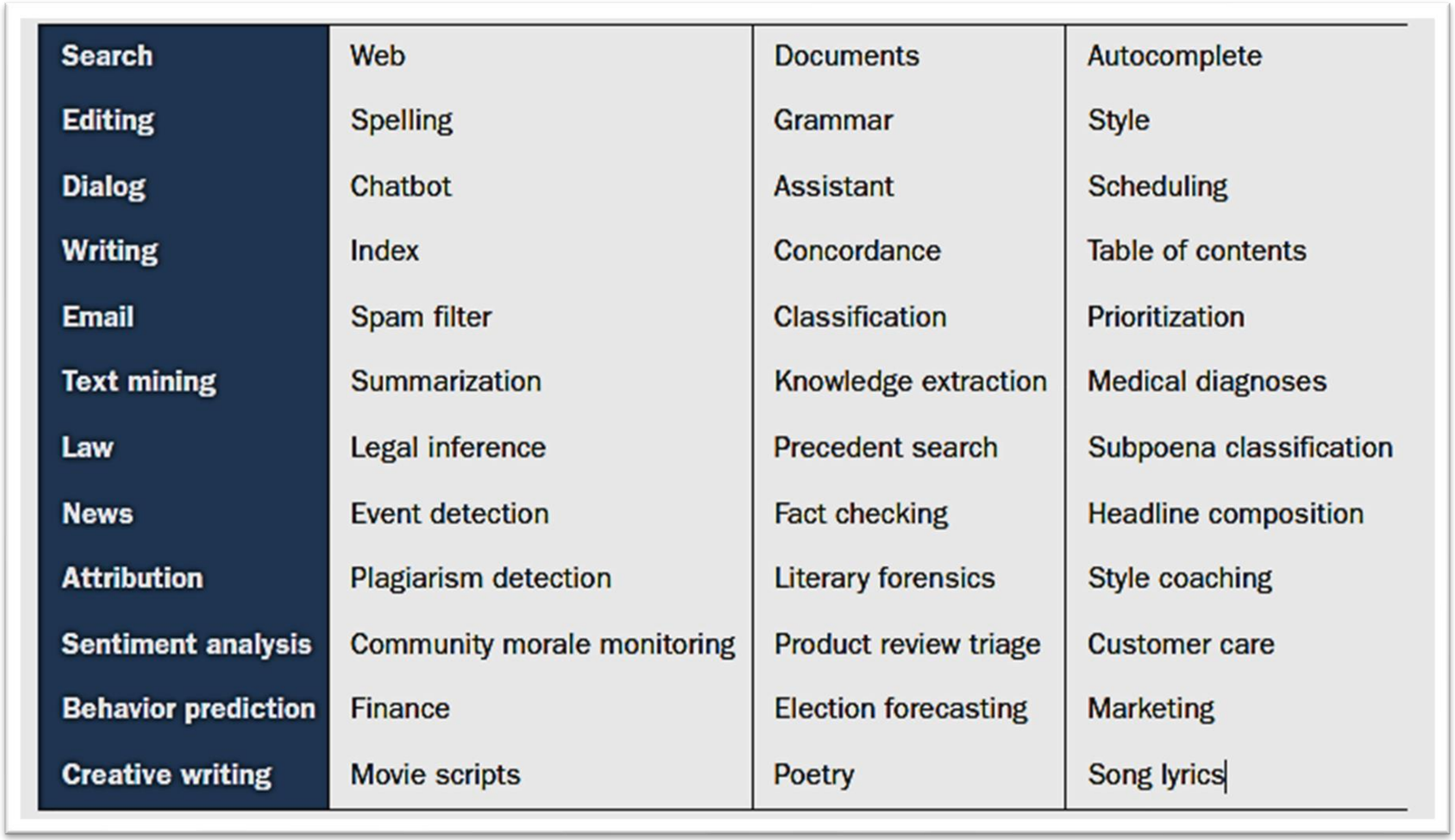
- For NLP, it is possible to use supervised (**Logistic Regression, SVM**) and unsupervised models (**Non-Negative Matrix Factorization, PCA**), as well as **Neural Networks** (recurrent neural networks, LSTM) to measure the accuracy of model, generate and summarize text, as well as to predict outcomes

How Does Natural Language Processing Work?


How does NLP work?
SENSES
Kinesthetic
Visual
Auditory
Olfactory
Gustatory
FILTERS
delete, distort and generalize sensory input
Attitudes
Memories
Sorting
Values
Beliefs
Strategies
Language
Usually unconcious in most people
Internal Representations Your Maps of Reality
EMOTIONS
BEHAVIOUR
Life Experiences Your Results
INPUT
+
INTERNAL PROCESSES
=
OUTPUT
NLP techniques EXCHANGE your filters, EXPAND your maps & ENHANCE your possibilities for successful thinking, feeling and behaving.
www.y

Source: YUP Training Solutions

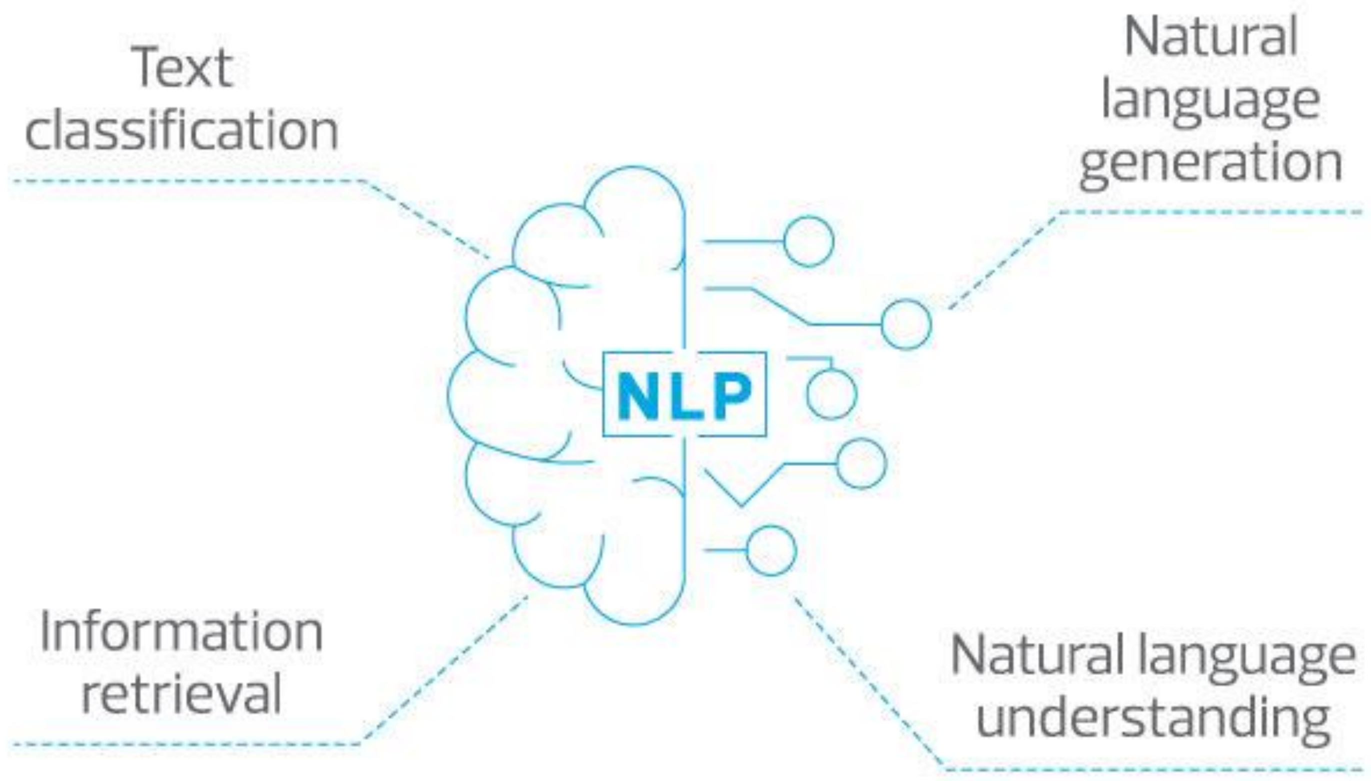
Some Practical Applications of Natural Language Processing



Search	Web	Documents	Autocomplete
Editing	Spelling	Grammar	Style
Dialog	Chatbot	Assistant	Scheduling
Writing	Index	Concordance	Table of contents
Email	Spam filter	Classification	Prioritization
Text mining	Summarization	Knowledge extraction	Medical diagnoses
Law	Legal inference	Precedent search	Subpoena classification
News	Event detection	Fact checking	Headline composition
Attribution	Plagiarism detection	Literary forensics	Style coaching
Sentiment analysis	Community morale monitoring	Product review triage	Customer care
Behavior prediction	Finance	Election forecasting	Marketing
Creative writing	Movie scripts	Poetry	Song lyrics

Cole Howard (2019)

Key Areas in Natural Language Processing



Differences between Text Analytics, Natural Language Processing, and Text Mining

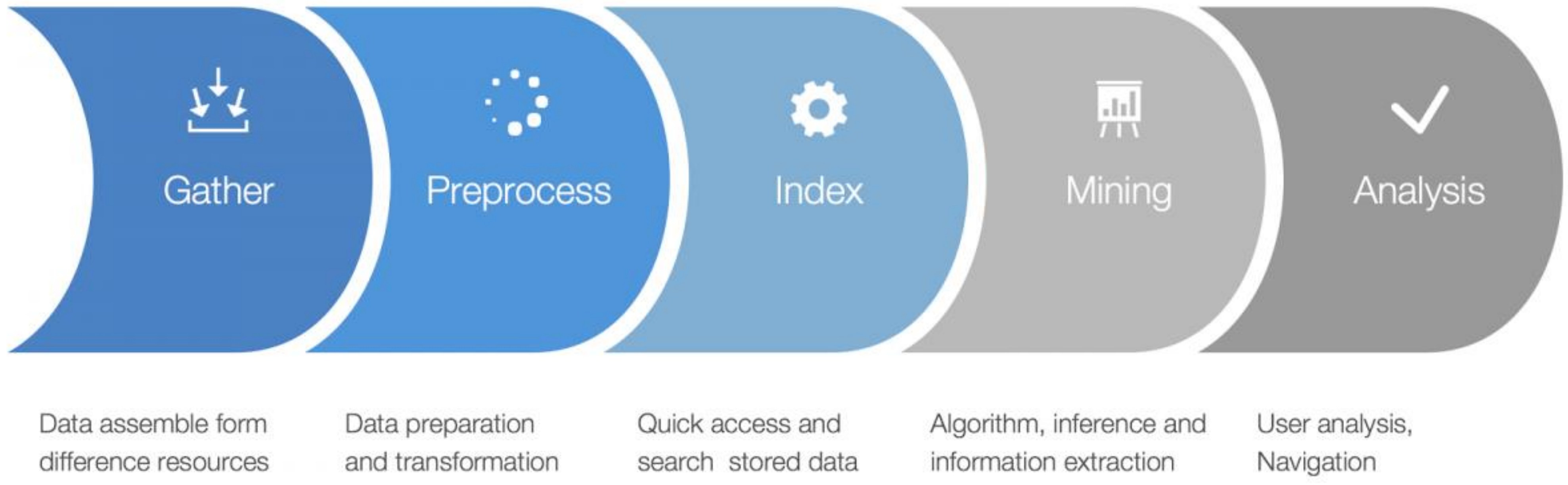
- **Text mining** is a process of **exploring** sizeable textual data and find patterns
  - Finding **frequency counts of words**, **length of sentences**, **presence/absence of specific words** is known as **text mining**
  - Text mining is **preprocessed data** for **text analytics**

- **Natural Language Processing** is one of the components of text mining
  - NLP helps identify **sentiments**, find **entities** in the sentence and **categories** of blogs/articles/posts

- **Text analytics** is the automated process of translating large volumes of **unstructured text** into **quantitative data** to **uncover insights, trends, and patterns**
  - In **text analytics**, statistical and machine learning algorithm are primarily used to **classify information**

The Text Mining Process

1. **Collect text(s)** into a document called **'corpus'**
2. **Analyze** the **corpus**
3. **Feature Generation**: Create a bag of words
4. **Feature Selection**: Count words and derive some statistics
5. **Text/Data Mining**: Use **classification** (**supervised learning**) or **clustering** (**unsupervised learning**)
6. **Analyze** and **communicate** the results


Gather
Preprocess
Index
Mining
Analysis
Data assemble form difference resources
Data preparation and transformation
Quick access and search stored data
Algorithm, inference and information extraction
User analysis, Navigation

UMBC

Preparing Text Before Analysis

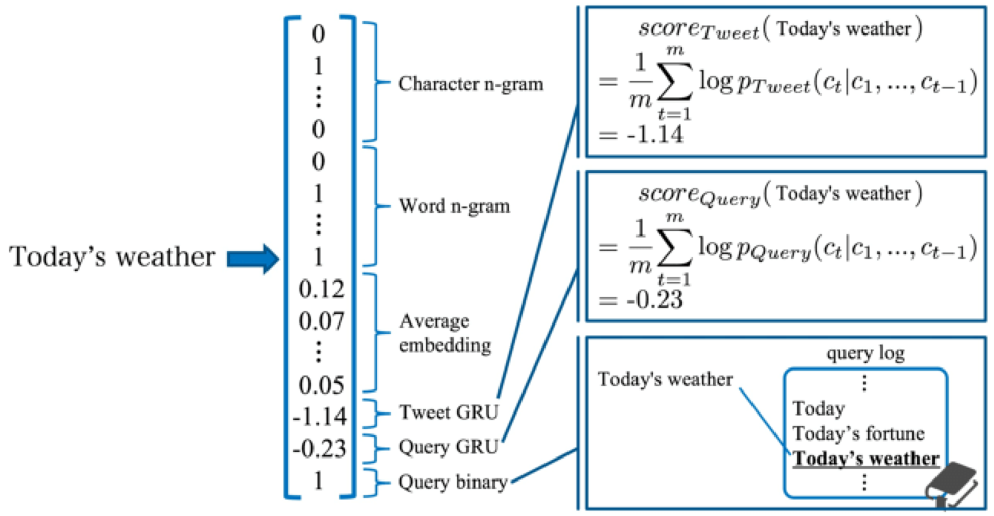
Text Preparation and Key Concepts

- **Tokenization** is a process of breaking down a text paragraph into smaller chunks such as a word or sentence
- **Sentence Tokenization** breaks text paragraph into sentences
- **Word Tokenization** breaks text paragraph into words
- **Determine the frequency distribution of words** (**Term Frequency**, **Inverse Document Frequency**)
- Eliminate **stopwords** considered as noise in the text. Text may contain stop words such as ‘is,’ ‘am,’ ‘are,’ ‘this,’ ‘a,’ ‘an,’ or ‘the’ that do not add any information on sentiment
- **Stemming** is a process of linguistic normalization, which **reduces words to their root** or chops off the derivational affixes
  - For example, ‘connection,’ ‘connected,’ and ‘connecting’ reduce to a common word ‘connect’
- **Lemmatization** reduces words to their **base word**, which is linguistically correct **lemmas**
  - It transforms root words with the use of vocabulary and morphological analysis
  - **Stemmer** works on an individual word without knowledge of the context
    - For example, "better" has "good" as its lemma
- **Part-of-Speech** (POS) **Tagging** serves to identify the grammatical group of a given word
  - **POS tagging** looks for relationships within the sentence and assigns a corresponding tag to the word such noun, pronoun, adjective…

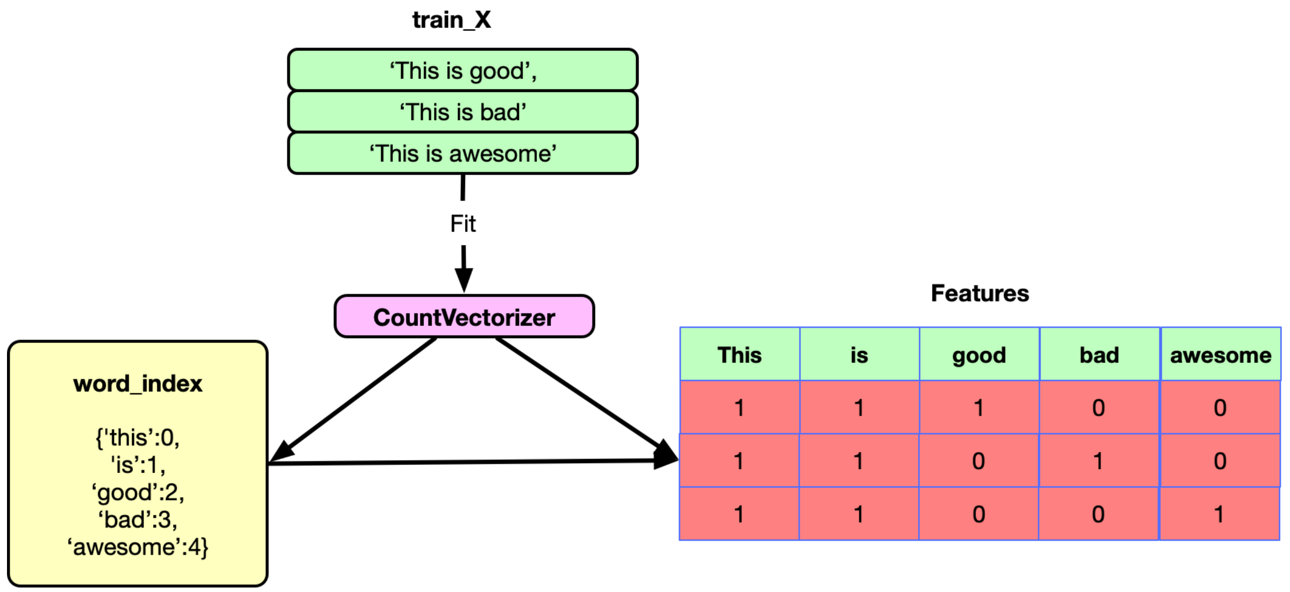
- These text treatments make it possible to evaluate the **sentiment** in a text

Preparing Text Before Analysis (Cont.)

How Does Vectorization Work?



Today's weather
0
1
⋮
0
0
1
⋮
1
0.12
0.07
⋮
0.05
-1.14
-0.23
1
Character n-gram
Word n-gram
Average embedding
Tweet GRU
Query GRU
Query binary
$score_{Tweet}(\text{Today's weather}) = \frac{1}{m}\sum_{t=1}^{m} \log p_{Tweet}(c_t|c_1, ..., c_{t-1}) = -1.14$
$score_{Query}(\text{Today's weather}) = \frac{1}{m}\sum_{t=1}^{m} \log p_{Query}(c_t|c_1, ..., c_{t-1}) = -0.23$
query log
Today's weather
Today
Today's fortune
Today's weather



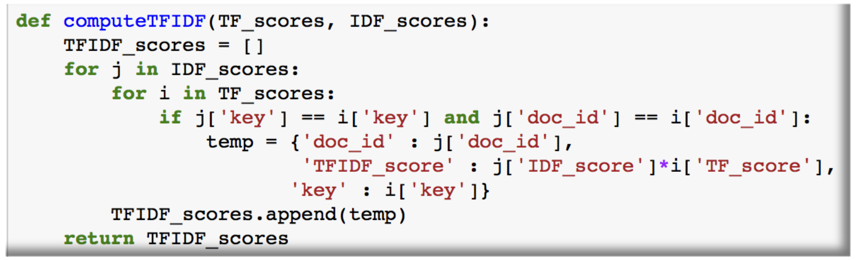
This	is	good	bad	awesome
1	1	1	0	0
1	1	0	1	0
1	1	0	0	1

UMBC

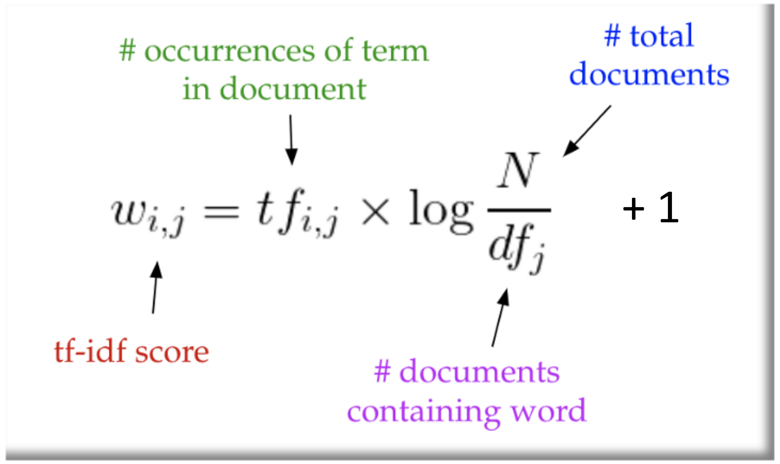
Feature Engineering: TF and IDF

Key Metrics

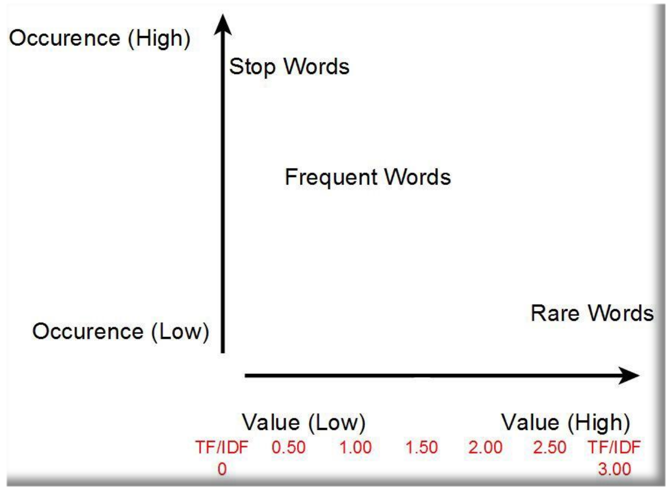
- With **Term Frequency** (**TF**), you **count the number of words in each document**
  - **Term Frequency** provides **more weight to longer documents**
  - **Term frequency** is basically the output of the BOW model
- **Inverse Document Frequency** (**IDF**) measures the **importance of a given word in a document**
  - **IDF** is the **logarithmically-scaled inverse ratio of the number of documents** that contain the word to the total number of documents such that IDF = log (1 + n/1 + df(d,t)) + 1 with n as number of documents and df(d,t) as the document frequency of the term t. **IDF = 1/DF**
- According to **Zipf's law**, "for a given corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table"
- **BM 25 (Best Match)** improves on TF-IDF by considering the length of the document. It is based on BOW retrieval and provides ranking based on query terms appearing in a document



```
def computeTFIDF(TF_scores, IDF_scores):
    TFIDF_scores = []
    for j in IDF_scores:
        for i in TF_scores:
            if j['key'] == i['key'] and j['doc_id'] == i['doc_id']:
                temp = {'doc_id' : j['doc_id'],
                        'TFIDF_score' : j['IDF_score']*i['TF_score'],
                       'key' : i['key']}
        TFIDF_scores.append(temp)
    return TFIDF_scores
```


# occurrences of term in document
# total documents
tf-idf score
# documents containing word

$$w_{i,j} = tf_{i,j} \times \log \frac{N}{df_j} + 1$$



UMBC

Feature Engineering: Computing Similarity

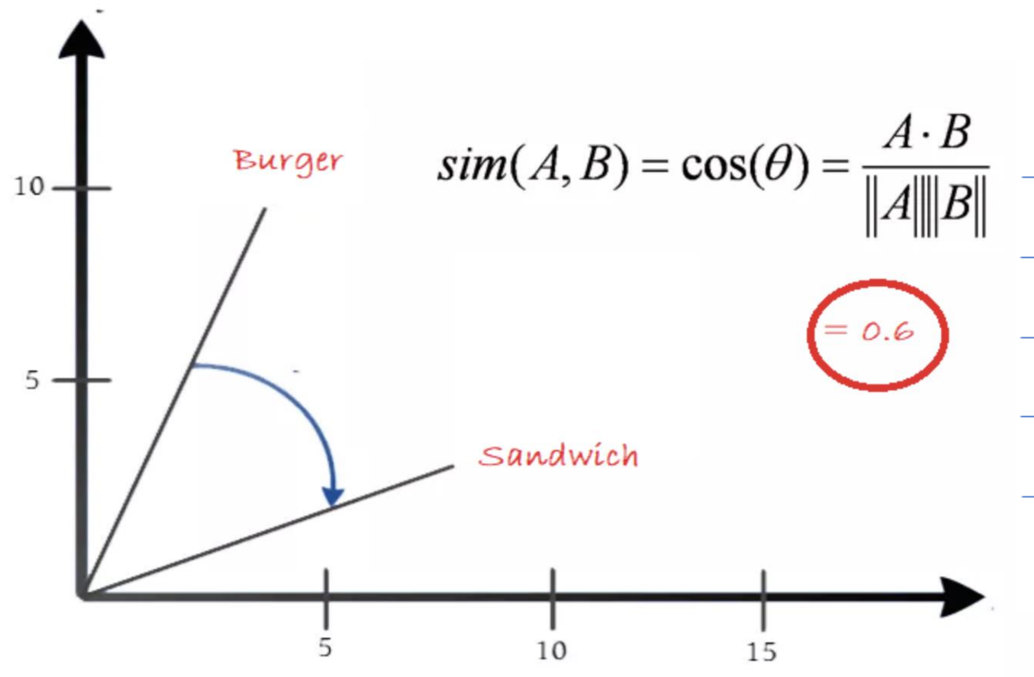
- **Feature engineering** is a method for **extracting new features from existing features**
- These new features are extracted as they tend to effectively explain variability in data
- One application of feature engineering could be to calculate how similar different ‘pieces’ of text are
- There are various ways of calculating the similarity between two ‘pieces’ of texts
- The most popular methods are **cosine similarity** and **Jaccard similarity**
- **Cosine similarity**: The cosine similarity between two texts is the **cosine of the angle between their vector representations**
  - **BOW** and **TF-IDF** matrices can be regarded as **vector** representations of texts
  - The **cosine similarity** measures the **cosine of the angle** between **two vectors** projected in a multi-dimensional space
  - The two vectors are **arrays** containing the **word counts** of two documents. Below is the formula:

$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \, \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a} \cdot \vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

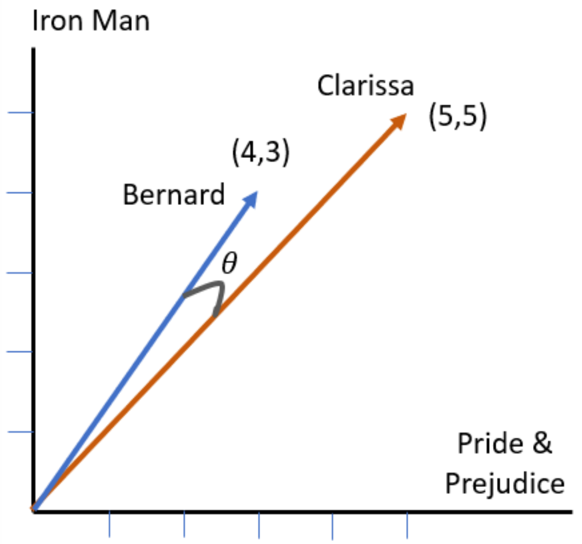
  - **cosΘ = -1** indicates **strongly opposite vectors**, **cosΘ = 0** means **independent (orthogonal) vectors** and **cosΘ = 1** means **positive co-linear vectors**. **Intermediate values** are used to assess the **degree of similarity**
- **Jaccard similarity**: This is the **ratio** of the **number of terms common between two text documents** to the **total number of unique terms present in those texts**

Feature Engineering: Computing Similarity



$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

$= 0.6$



Calculating:

$$b.c = \sum_{i=1}^{n} b_i c_i = (4 \times 5) + (3 \times 5) = 35$$

$$\|b\| = \sqrt{4^2 + 3^2} = 5$$

$$\|c\| = \sqrt{5^2 + 5^2} = 5\sqrt{2}$$

$$similarity = \frac{35}{5 \times 5\sqrt{2}} \sim 0.989$$

Text Summarization

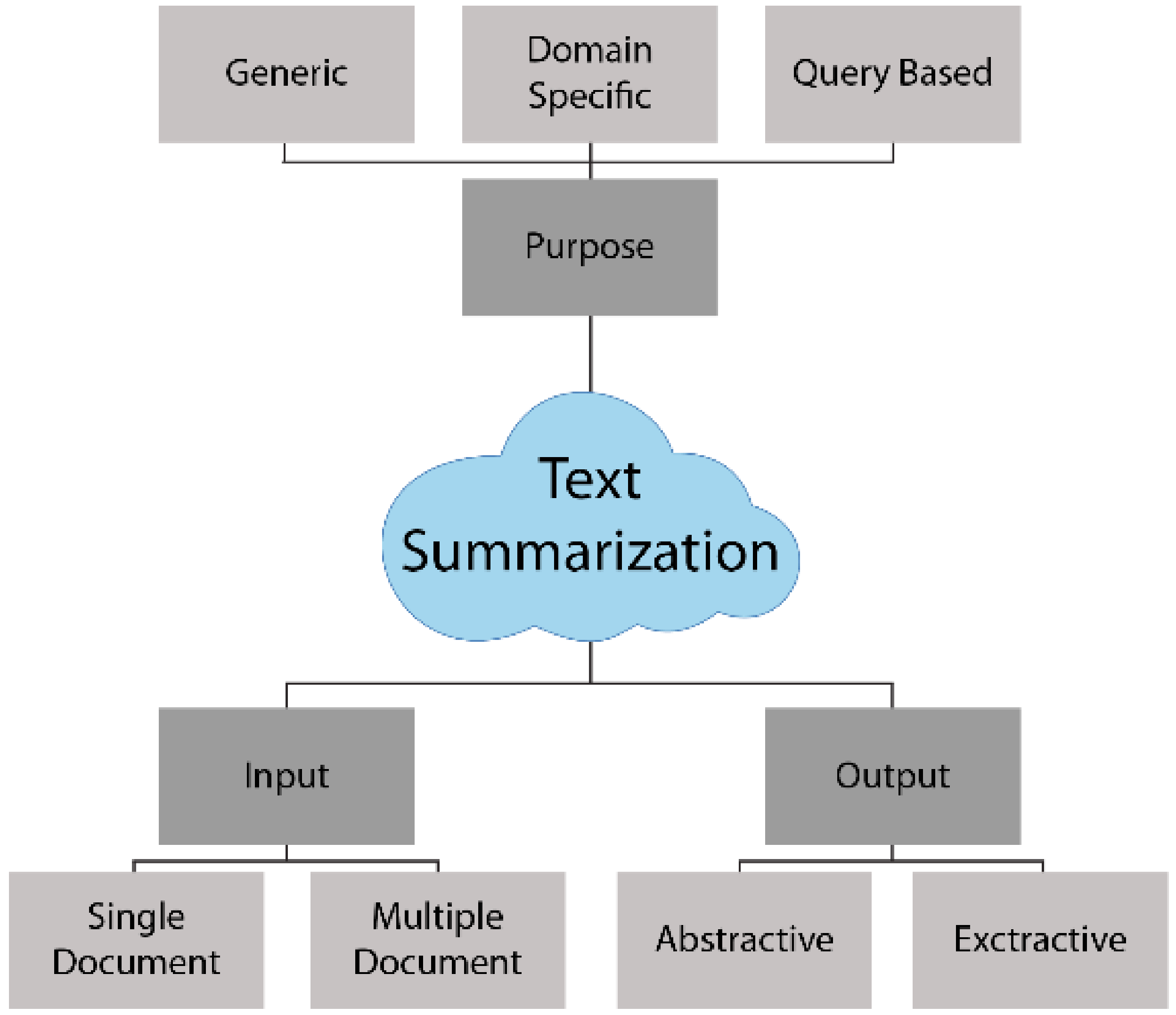
- "**Text summarization** is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)," Mani and Maybury (1999)
- **Automated text summarization** is the process of using natural language processing (NLP) tools to produce concise versions of a text that preserve all the key information present in the original content
- Some of these tools such as **Gensim** and **NLTK** are frameworks that contain algorithms for text summarization They also have easy-to-use interfaces
- **Automated text summarization** provides benefits related to:
  - **Sampling**: An automated summary can be a sample of a document or article ahead of the article's actual release. This allows the reader to decide whether to read the full article or not
  - **Searching**: Summaries can assist in searches. They can be shown in search results for users to select
  - **Indexing:** Summaries can be used in search indexes, acting as a compressed representation of the original document. This also helps reduce the amount of storage required
  - **Reading time**: If a summary is well constructed, it can be used instead of the actual document. This can result in a much shorter reading time and is acceptable if the reader gets the gist of the original document
  - **Answering questions:** Chatbots can provide automated summaries as responses to user questions
- Google's *Smart Reply*, which suggests short replies to emails in smartphone, is an application of text summarization

Text Summarization (Cont.)

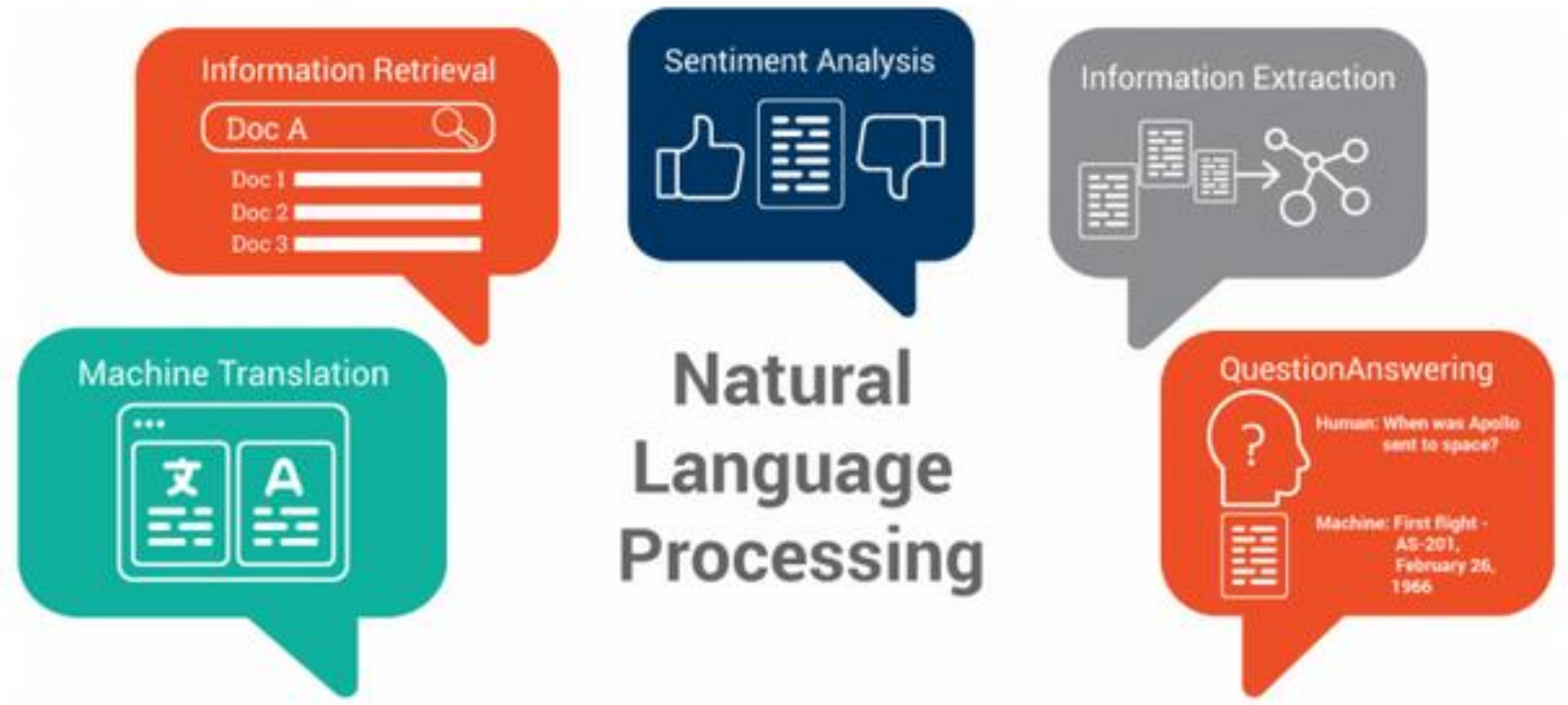
- **Generic Summarization**: Text summarizers can operate on any text input of sufficient length and they can perform adequately on sources from many different domains

- **Domain-Specific Summarization**: Text summarization can be specific to a domain, such as finance, shopping, medicine, and travel

- **Query-Based Summarization**: They are centered around providing a response to user input, either in a search bar or from a chatbot



APPLICATIONS OF NATURAL LANGUAGE PROCESSING



Latent Semantic Analysis (LSA)

- It is based on the **distributional hypothesis** which states that the semantics of words depends on contexts where we find the words. Under this hypothesis, the semantics of two words will be similar if they tend to occur in similar contexts
- **LSA** computes how frequently words occur in the documents – and the whole corpus – and assumes that similar documents will contain approximately the same **distribution** of word frequencies for certain words
- The standard method for computing word frequencies is known as **TF-IDF**
    - **TF-IDF** computes frequencies by taking into consideration, not only the frequency of words in a document, but also the frequency of words in all the corpus of documents
    - Words with a higher frequency in the full corpus will be better candidates for document representations than less frequent words, regardless of how many times they appear in individual documents
    - Once **TF-IDF** frequencies have been computed, we create a **Document-Term Matrix** which shows the **TF-IDF** value for each term in a document. This matrix will have **rows** for every **document** in the corpus and **columns** for every **term** considered
- The **Document-Term Matrix** can be decomposed into the product of 3 matrices (**USV**) by using **Singular Value Decomposition (SVD)**
    - The **U matrix** is known as the **Document-Topic Matrix** and the **V matrix** is known as the **Term-Topic Matrix**
    - The **S matrix** is diagonal and **LSA** will consider each **singular value**, i.e., each of the numbers in the main diagonal of matrix S, as a potential topic found in the documents
    - If we keep the largest t **singular values** together with the first t columns of U and the first t rows of V, we obtain the t more frequent topics found in our original **Document-Term Matrix**
    - Since we do not keep all the singular values of the original matrix, we call the methodology **Truncated SVD**
    - We can set the value of **t** as a hyperparameter

UMBC

Latent Dirichlet Allocation (LDA)

- **Latent Dirichlet Allocation (LDA)** and **LSA** are based on the same underlying assumptions:
  - The distributional hypothesis, (i.e., similar topics make use of similar words) and
  - The statistical mixture hypothesis (i.e., documents talk about several topics) for which a statistical distribution can be determined
- The purpose of **LDA** is mapping each document in a corpus to a **set of topics** which covers a most of the words in a document
- The goal of **LDA** is to determine the **mixture of topics** that a document contains
- **LDA** assigns **topics** to **arrangements of words**, e.g., n-grams such as ’increased inflation’ for a topic related to ‘economics’
- **LDA** assumes that documents are written with **arrangements of words** and that those **arrangements determine topics**
- **LDA** also ignores syntactic information and treats documents as **bags of words**
- It also assumes that all words in the document can be assigned a **probability of belonging to a topic**

Latent Dirichlet Allocation (Cont.)

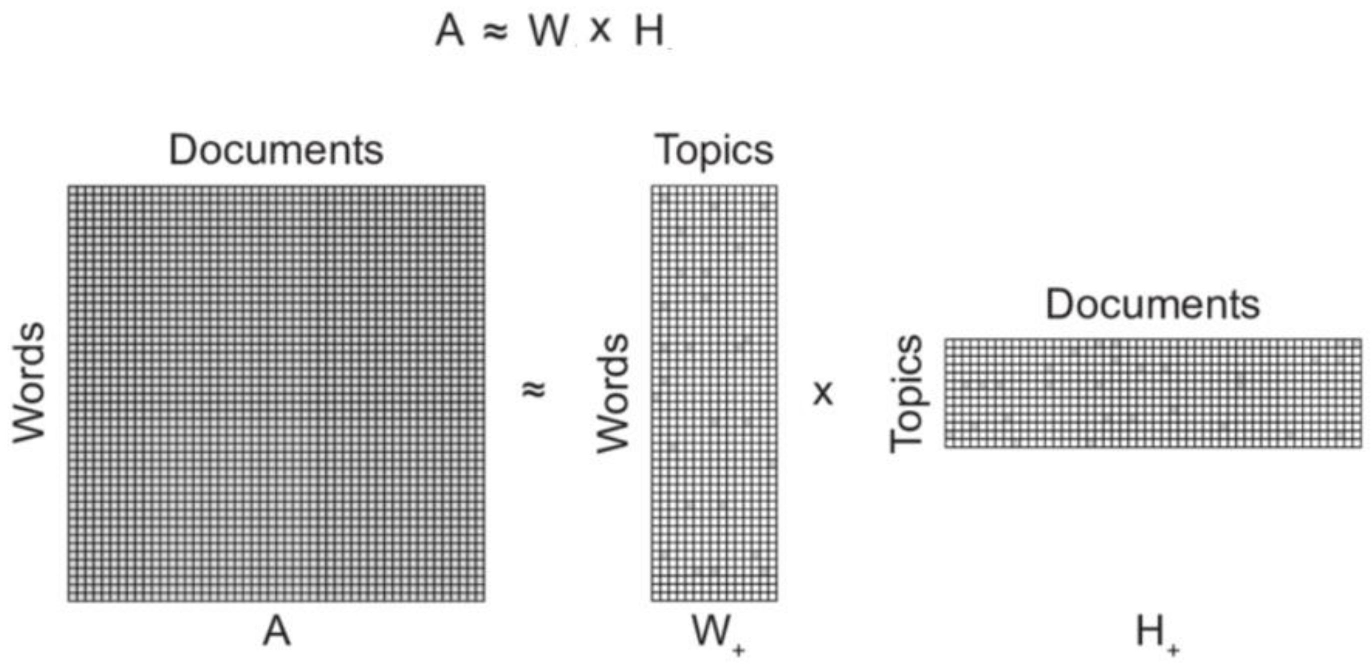
- LDA assumes that the distribution of topics in a document and the distribution of words in topics are Dirichlet distributions
- LSA does not assume any distribution and therefore, leads to more difficult to explain vector representations of topics and documents
- There are three hyperparameters
  - Two of them control document and topic similarity, known as **alpha** and **beta**, respectively
    - A *low value* of **alpha** will assign *fewer topics* to each *document*, whereas a *high value* of **alpha** will have the opposite effect
    - A *low value* of **beta** will use *fewer words* to *model a topic,* whereas a *high value* will use *more words*, thus making topics more similar between them
  - The third parameter involves the **number of topics** the algorithm will detect since LDA cannot decide on the number of topics by itself
- The output of the algorithm is a vector that contains the coverage of every *topic* for the *document* being modeled
  - The vector looks like [0.2, 0.5, etc.] where the first value shows the coverage of the first topic and so on

Non-Negative Matrix Factorization (NMF)

- **NMF** was first introduced by Paatero and Tapper in 1994, and popularized in an article by Lee and Seung in 1999
- It belongs to the family of linear algebra algorithms that are used to identify the **latent** or **hidden structure** present in the data
- In the **bag-of-words matrix representation,** each row corresponds to a word, and each column to a document
- Given a set of documents, **NMF** identifies **topics** and simultaneously classifies the **documents** among these different **topics**
- The input is the **Term-Document Matrix**
- The output is two non-negative matrices of the original n-words by k topics and those same k topics by the m original documents
- **NMF** is an unsupervised technique:
  - **Topics** are not labeled
  - **NMF** decomposes (or factorizes) high-dimensional vectors into a lower-dimensional representation
  - These lower-dimensional vectors are **non-negative**, which means that their coefficients are non-negative, too
- Using the original matrix (A), **NMF** generates two matrices:
  - **W** includes the topics and
  - **H** contains the coefficients (weights) for those topics

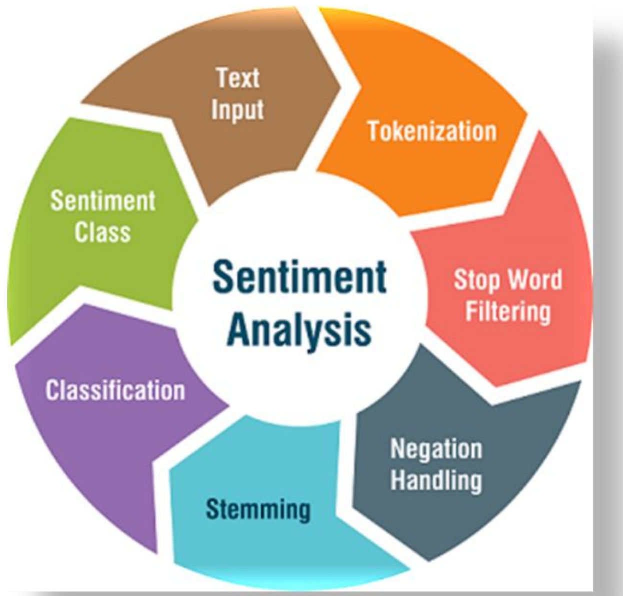
Non-Negative Matrix Factorization (Cont.)

- **NMF** makes it possible to automatically extract sparse and easily interpretable factors
- The first matrix **W** has every topic and what terms in it
- The second matrix **H** has every document and what topics in it



Sentiment Analysis Process

- **Sentiment Analysis** is the **interpretation** and **classification of emotions** (positive, negative and neutral) within text data using **text analytics**
- Sentiment analysis requires complex text treatment to
    - Identify **patterns in opinions** and **behaviors** through comments in Facebook posts or tweets, for instance
    - Match **identified patterns** with **sentiment classes (i.e. happiness, anger…)**
    - Derive the **degree of subjectivity** (degree of emotion) and **polarity** or **orientation** (positive, neutral, or negative)



- There are two main approaches:
    - **Lexicon-based**: You count the number of positive and negative words in given text. The larger count will determine the sentiment of the text
    - **Machine-learning-based approach**: You develop a classification model, which is trained using the pre-labeled dataset of positive, negative, and neutral

Sentiment Analysis (Cont.)

- Sentiment analysis, also called **opinion mining**, is defined as

  "The **field of study** that **analyzes** people's opinions, sentiments, evaluations, attitudes, and emotions from **written language**."[1]

- Sentiment analysis is applied in almost every type of **business** and **social domain** because
  - **Sentiments** are **central** to almost all **human activities**
  - **Sentiments** influence **people's behaviors**
  - **Sentiments** can be **indicators** of **positive (opportunities) and negative (threats) risks**
- People's **beliefs** and **perceptions of reality**, and the **choices** they make, are largely conditioned on how others see and evaluate the world
- For this reason, when people need to decide, they often seek out the opinions of others
- One of the key functions of sentiment analysis is to identify **influencers: those who drive the discussion, influence beliefs** and **perceptions**
- Today, social media generate a large amount of **information** at a great velocity, which is readily available on the Internet for mining and analysis
- Sentiment analysis is a tool to **categorize** sentiments and **opinions** and to provide some **insights** on what is **influencing** behaviors
- Sentiment analysis has mostly replaced '**traditional' research methodologies** such as **surveys, interviews,** and **panels,** which are **more costly, slower to administer,** and subject to **respondents' biases**

[1] Liu, Bing. *Sentiment Analysis and Opinion Mining.* Synthesis Lectures on Human Language Technologies, May 2012, Volume 5, No. 1, pp. 1-167.

UMBC

Recommendation or Recommender Systems

- Machine learning algorithms in recommender systems are typically classified into two categories — **content based and collaborative filtering methods** although modern recommenders combine both approaches
- **Content-based methods** depend on **similarity** of item attributes, while **collaborative methods** calculate **similarity from interactions**

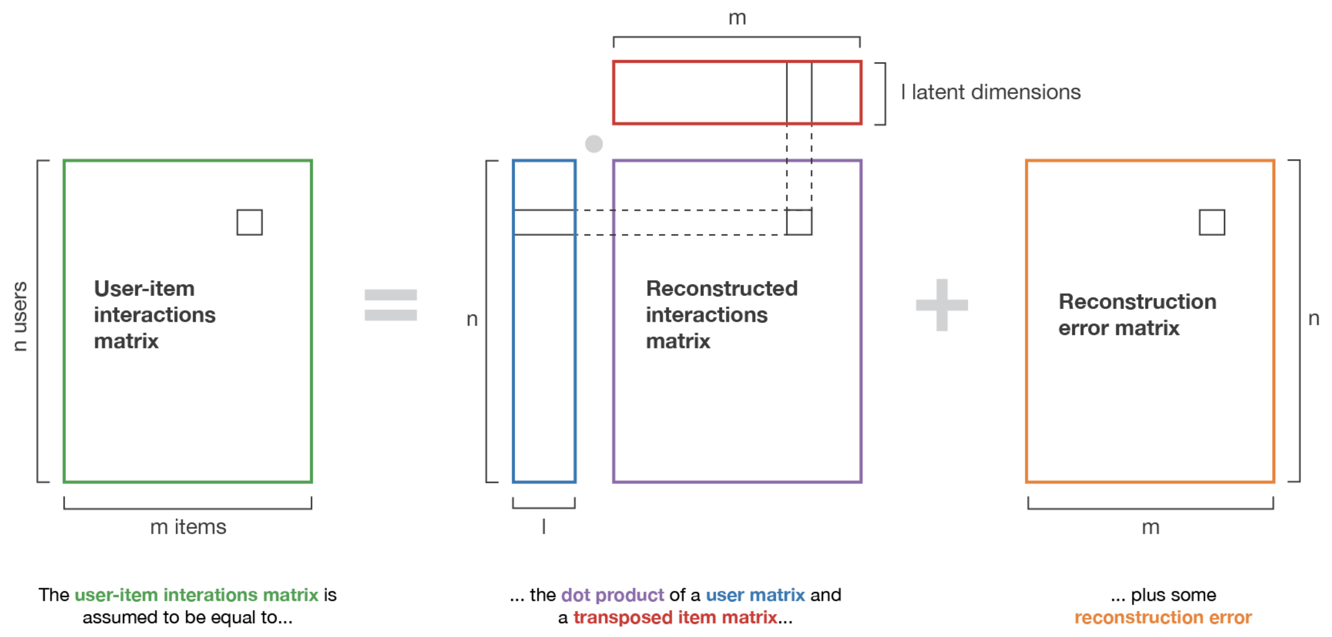
Advantages and Disadvantages of Both Types of Recommendation Systems

	Advantages	Disadvantages
Collaborative	• Requires no information about users or items • Increased interactions generate better recommendations • No latent model assumed • Low bias	• Difficult to recommend anything to new users (no experience) • Too few interactions may not provide reliable info on behavior patterns • Complexity increases with data volume • High variance
Content-Based	• Way to bypass 'cold start' or insufficient records of past behaviors • Makes association of users with identified classes with similar features • Latent interactions assumed • Low variance	• Model trained to reconstruct user-item interactions values from its own representation of users and items • Assumed user-item interactions are dynamic • High bias

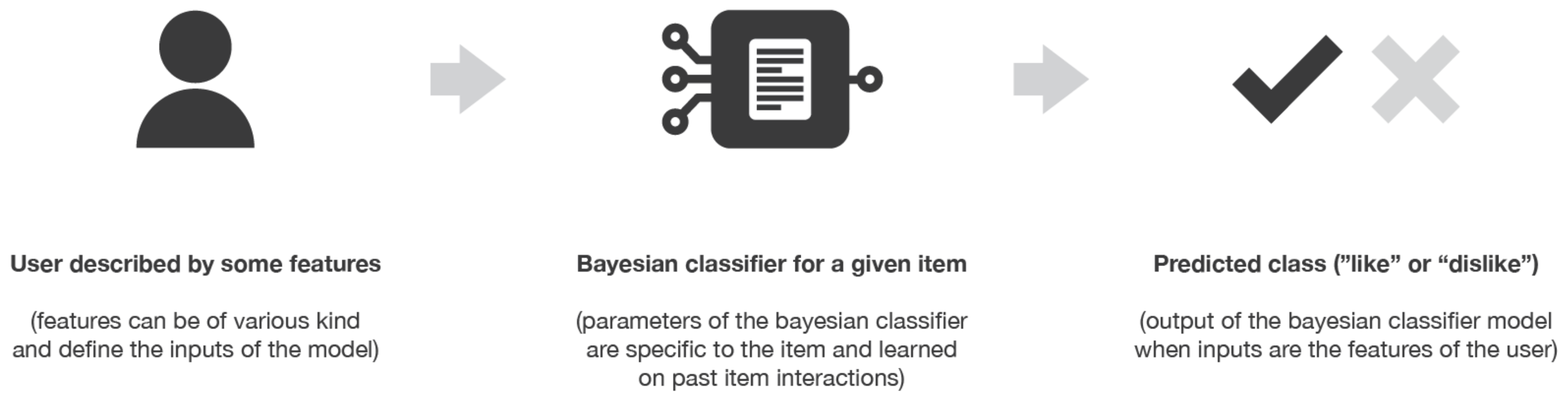
UMBC

Examples of Strategies for Recommendations

Collaborative Approach (NMF)


m
l latent dimensions
n users
User-item interactions matrix
m items
=
n
Reconstructed interactions matrix
l
+
Reconstruction error matrix
n
m
The user-item interations matrix is assumed to be equal to...
... the dot product of a user matrix and a transposed item matrix...
... plus some reconstruction error

Content-Based Approach (Naïve Bayes)


User described by some features
(features can be of various kind and define the inputs of the model)
Bayesian classifier for a given item
(parameters of the bayesian classifier are specific to the item and learned on past item interactions)
Predicted class ("like" or "dislike")
(output of the bayesian classifier model when inputs are the features of the user)

Source: Toward Data Science

Chatbots

- A **chatbot** is a program that communicates with a user
- **Chatbots** are not new. They were created in the 1960’s. In 1966, Weizenbaum invented a computer program called ELIZA which imitated the language of a psychotherapist from only 200 lines of code
- The first move away from text chatbots occurred in 1988 when Rollo Carpenter started the Jabberwacky project
- They are useful when we do not want to recreate the same task/process by going through the same set of procedures over again
- They are trained to provide answers to specific questions and interact with users (Siri and Alexa)
- One of the biggest successes ultimately came from the technology within smartphones
- In 2010, Apple launched Siri for iOS. Siri was the first multi-functional, voice-commanded bot. It paved the way for intelligent personal home assistants, such as Amazon’s Alexa
- Chatbots are difficult to create especially when they try to interpret natural language
- There are six types of **chatbots**:
  - **Rule-based**: They respond to very specific commands (for instance, weather forecast at a selected location)
  - **Machine-Learning-based**: They try to understand the sentiment and meaning of the language used and not to rely on predetermined command
  - **Text-based**: A bot answers the user’s questions via text interface
  - **Voice-based**: A bot answers the user’s questions via a human voice interface
  - **Retrieval-based:** They provide an answer from a predefined set of answers
  - **Generative-based:** They are most often based on basic probabilistic and machine learning models. They need to be trained to generate new content. Markov chains have originally been used for the task of text generation. Today, Recurrent Neural Networks (RNN) have gained more popularity