

# Text Analytics

Data Science Dojo

# Overview

- What is text analytics?
- Fundamentals
  - Tokenization
  - Stemming and Lemmatization
  - Document Vectors
- Term Frequency (TF) and Inverse Document Frequency (IDF)
  - Creating and inverted index and retrieving documents from a query

# Structured vs. Unstructured Data

- Structured – Tabular data
- Semi-structured – Non-tabular data with some meta-data
  - Ex: JSON, XML
- Unstructured – Non-tabular data with no meta-data

# Text Analytics

- **Information Retrieval (IR)**

- Find documents which match a query
- Can support binary or full text queries

- **Sentiment Analysis**

- Determine "emotion" of document
- Classification task

- **Recommendation Engines**

# Text Analytics

- How do we turn unstructured data into structured data?
  - Create columns based on document content
  - Each **term** in document creates a column
    - Column types: binary, word count, TF-IDF
  - Do we want to count every word?
    - Stop words

# Text Analytics

- **Token:** A specific word in the document
- **Term:** The version of a word set that is in the dictionary
- What do we do about word variations?
  - is, are, am, be
  - run, running, ran, runs

# Stemming & Lemmatization

- **Stemming:** Convert tokens to terms by removing letters via heuristic
  - Both simple (Levins) and complex (Porter)
- **Lemmatization:** Classify tokens into terms using a linguistic analysis
  - **Lemma:** the base (dictionary) form of a word
  - Can be done using machine learning

# Stemming Example

## Rules

- am, are, is => be
- car, cars, car's, cars' => car

## Sentence

The boy's cars are different colors.

=> the boy car be differ color



# Document Vectors

- Each document becomes a vector
- Store in an "inverted index"
- Allows use of numeric analysis

	Team	Coach	Play	Ball	Score	Game	Win	Lost	Timeout	Season
$d_1$	3	0	5	0	2	6	0	2	0	2
$d_2$	0	7	0	2	1	0	0	3	0	0
$d_3$	0	1	0	0	1	2	2	0	3	0

# Document Vectorization

## ■ Binary approach

- Each document has a 1 if the word occurs in the document and a 0 if not

	Team	Coach	Play	Ball	Score	Game	Win	Lost	Timeout	Season
$d_1$	1	0	1	0	1	1	0	1	0	1
$d_2$	0	1	0	1	1	0	0	1	0	0
$d_3$	0	1	0	0	1	1	1	0	1	0

# Binary approach: drawbacks

- Not every word has similar importance
- Longer documents have a higher chance to have random unimportant words
- Automated extraction particularly problematic

# TF-IDF

- Common Solution: TF-IDF
  - **Term Frequency:** Measures how often a term appears (density in a document)
    - Assumes important terms appear more often
    - Normalized to account for document length
  - **Inverse Document Frequency:** Aims to reduce the weight of terms that appear in all documents
    - Assumes terms that appear in many documents are less important

# Term Frequency

- **Term frequency (TF)**

- Let  $freq(t,d)$  number of occurrences of keyword  $t$  in document  $d$
- Let  $\max\{freq(w,d)\}$  denote the highest number of occurrences of another keyword of  $d$
- $TF(t, d) = \frac{freq(t,d)}{\max\{freq(w,d):w \in d\}}$

# Inverse Document Frequency

- **Inverse Document Frequency (IDF)**

- N: number of all recommendable documents
- $n(t)$ : number of documents in which keyword  $t$  appears
- $IDF(t) = \log \frac{N}{n(t)}$

# TF-IDF

- Compute the overall importance of keywords
  - Given a keyword  $t$  and a document  $d$

$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

# TF-IDF Exercise

- **D1** = "If it walks like a duck and quacks like a duck, it must be a duck."
- **D2** = "Beijing Duck is mostly prized for the thin, crispy duck skin with authentic versions of the dish serving mostly the skin."
- **D3** = "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."
- **D4** = "6:25 PM 1/7/2007 blog entry: I found this great recipe for Rabbit Braised in Wine on cookingforengineers.com."
- **D5** = "Last week Li has shown you how to make the Sechuan duck. Today we'll be making Chinese dumplings (Jiaozi), a popular dish that I had a chance to try last summer in Beijing. There are many recipies for Jiaozi."
- **Dictionary:** {beijing, dish, duck, rabbit, recipe}



# Creating the TF Matrix

	Beijing	Dish	Duck	Rabbit	Recipe
D1	0	0	3 / 3	0	0
D2	1 / 2	1 / 2	2 / 2	0	0
D3	0	0	2 / 2	1 / 2	1 / 2
D4	0	0	0	1 / 1	1 / 1
D5	1 / 1	1 / 1	1 / 1	0	1 / 1

# Creating the IDF Vector

	Beijing	Dish	Duck	Rabbit	Recipe
D1	0	0	1	0	0
D2	0.5	0.5	1	0	0
D3	0	0	1	0.5	0.5
D4	0	0	0	1	1
D5	1	1	1	0	1

Word	IDF
Beijing	$\log(5/2)$
Dish	$\log(5/2)$
Duck	$\log(5/4)$
Rabbit	$\log(5/2)$
Recipe	$\log(5/3)$

# TF-IDF Matrix

	Beijing	Dish	Duck	Rabbit	Recipe
D1	0	0	0.097	0	0
D2	0.199	0.199	0.097	0	0
D3	0	0	0.097	0.199	0.111
D4	0	0	0	0.398	0.222
D5	0.398	0.398	0.097	0	0.222

# TF-IDF Search Example

- User searches our document set
- **Query:** "Beijing duck recipe"
- Calculate TF-IDF of query

	Beijing	Dish	Duck	Rabbit	Recipe
Query	0.398	0	0.097	0	0.222

Word	IDF
Beijing	$\log(5/2)$
Dish	$\log(5/2)$
Duck	$\log(5/4)$
Rabbit	$\log(5/2)$
Recipe	$\log(5/3)$

# TF-IDF Search Exercise

- Cosine similarity of query and each doc
- $D1 = [0, 0, 0.097, 0, 0]$
- $Q = [0.398, 0, 0.097, 0, 0.222]$
- $$\cos(D1, Q) = \frac{0*0.398+0*0+0.097*0.097+0*0+0*0.222}{\sqrt{0.097^2}*\sqrt{0.398^2+0.097^2+0.222^2}}$$
- $$\cos(D1, Q) = \frac{0.00941}{0.0452} = 0.208$$

# Cosine similarities

	Beijing	Dish	Duck	Rabbit	Recipe	Cos(D,Q)
D1	0	0	0.097	0	0	0.208
D2	0.199	0.199	0.097	0	0	0.639
D3	0	0	0.097	0.199	0.111	0.256
D4	0	0	0	0.398	0.222	0.232
D5	0.398	0.398	0.097	0	0.222	0.760
Query	.398	0	.097	0	.222	1

# Final ordered list

- **D5** = "Last week Li has shown you how to make the Sechuan duck. Today we'll be making Chinese dumplings (Jiaozi), a popular dish that I had a chance to try last summer in Beijing. There are many recipes for Jiaozi."
- **D2** = "Beijing Duck is mostly prized for the thin, crispy duck skin with authentic versions of the dish serving mostly the skin."
- **D3** = "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."
- **D4** = "6:25 PM 1/7/2007 blog entry: I found this great recipe for Rabbit Braised in Wine on [cookingforengineers.com](http://cookingforengineers.com)."
- **D1** = "If it walks like a duck and quacks like a duck, it must be a duck."

# Text Analytics Tools

- R – tm, Rstem, openNLP
- Python – NLTK
- Azure – Feature Hashing module



# QUESTIONS