Text analysis

Instructor: Davide Proserpio

Why Text Matters in Marketing

- 90%+ of consumer data is unstructured text: reviews, tweets, chats, survey comments.
- Text reveals **perception**, **experience**, and **emotion**.
- Quantifying text → insights for **brand monitoring**, **product design**, and **customer engagement**.

Core Steps in Text Analysis

- **Text cleaning & tokenization**
- **Feature extraction** (Bag-of-Words, TF-IDF, embeddings)
- **Modeling** (classification, clustering, topic modeling, etc.)
- **Interpretation & visualization**

Tokenization and Preprocessing

- Lowercasing, removing punctuation, stopwords.
- Tokenization = splitting text into words (or n-grams).
- N-grams capture context:
  - Unigram: “good”
  - Bigram: “not good” (captures negation!)

Representing Text Numerically

Method	What it Captures	Example
Bag-of-Words	word presence/frequency	“great”: 5, “bad”” 1
TF-IDF	importance weighted by rarity	down-weights popular words across docs
Word Embeddings	semantic meaning	“awesome” ≈ “great”

Bag-of-Words (BoW) Representation

- **Idea:** Treat a document as a **"bag" of words** — ignore order and grammar, count how often each word appears.
- **How it works**
  - Build a **vocabulary** of all unique words in the corpus.
  - For each document, record word counts or frequencies.
  - Represent each document as a **vector** of word counts:
    - $\text{Doc}_i = [\, f_{i,1}, f_{i,2}, \ldots f_{i,V} \,]$

      where $f_{i,j}$ =frequency of word j in document i, and V= vocabulary size.

Document	Text	Vector (great, bad, movie)
d_1	"great movie"	(1, 0, 1)
d_2	"bad movie"	(0, 1, 1)

Bag-of-Words (BoW) Representation

- **Pros**
  - Simple, fast, and easy to interpret.
  - Works well with linear models for classification (e.g., Logistic Regression, SVM).
- **Cons**
  - Ignores word order and semantics (“not good” ≈ “good”)
  - Produces large, sparse matrices for big vocabularies.

TF-IDF (Term Frequency – Inverse Document Frequency)

- **Goal:** Weight words by how important they are → *common in the document, rare in the corpus.*

1. Term Frequency (TF): Measures how often a term *t* appears in a document.

- $\mathrm{TF}(t, d) = \frac{f_{t,d}}{\sum_w f_{w,d}}$

frequency of term t divided by total words in document d

2. Inverse Document Frequency (IDF): Down-weights terms that appear in many documents.

- $\mathrm{IDF}(t) = \log \frac{N}{\mathrm{df}_t}$

N =total documents df_t =docs containing t

3. Combine: $\mathrm{TF{-}IDF}(t, d) = TF(t, d) \times IDF(t)$

TF-IDF (Term Frequency – Inverse Document Frequency)

Term	TF (doc)	IDF	TF-IDF
“great”	0.30	2.3	0.69
“movie”	0.20	0.8	0.16
“camera”	0.05	1.098	0.055

“great” scores higher because it appears often in the doc, but it is not common in all documents

TF-IDF (Term Frequency – Inverse Document Frequency)

- **Pros**
  - Highlights meaningful, unique terms.
  - Improves performance of linear models.
  - Easy to compute and interpret.
- **Cons**
  - Still ignores word order and context.
  - Large sparse matrices for big corpora.

Word Embeddings

- **Idea:** Map each word to a dense vector so that **semantic similarity ≈ geometric proximity**.
  - Similar words have vectors that point in similar directions
  - Capture semantic similarity: similarity(great, awesome) ≈ 1.
- Use pretrained models like Word2Vec, GloVe, BERT.
- **How they're learned (classic models)**
  - **Word2Vec:** predict a word from its context, or context from a word.

Word Embeddings

- **Pros**
  - Capture semantic information
  - Compact vector representation
  - Work with simple classifiers.
- **Cons**
  - Less interpretable
  - Quality depends on the pretraining corpus/model

- Example: https://devopedia.org/word-embedding

Understanding Sentiment

- **Sentiment = polarity of opinion (positive ↔ negative)**
- Useful for: customer satisfaction, campaign tracking, brand health.
- Sources of signal: adjectives, adverbs, modifiers, emojis, negations.

Lexicon-Based Sentiment (e.g., sentimentr)

- Uses predefined dictionaries of positive/negative words.
- Adjusts for valence shifters (“not good”, “really bad”).
- No training data required
  - Easy, interpretable
  - Domain sensitive, misses sarcasm

Supervised Sentiment Classification

- **Goal:** learn to predict sentiment from labeled data.
- Typical workflow:
  - Label reviews (e.g., using sentimentr or human tags).
  - Vectorize text (TF-IDF or embeddings).
  - Train a classifier (Logistic Regression, Random Forests, etc.).
  - Evaluate (Accuracy, Precision, Recall, AUC).

Beyond Sentiment: Emotion detection

- Often, human labelling or dictionary-based approaches

How it works

- Uses **emotion lexicons** — curated word–emotion mappings (e.g., *“joy”: happy, delighted, cheerful*; *“anger”: furious, hate, annoyed*).
- Common resources:
  - **NRC Emotion Lexicon** — maps ~14K English words to 8 basic emotions (Plutchik’s model).
  - **LIWC**, **ANEW**, and **WordNet-Affect** (psychological lexicons).

Beyond Sentiment: Topic Modelling

- Topic modelling
  - Latent Dirichlet Allocation (LDA)