Lecture 6

Word Embedding

Transforming Text Data

Quantifying text data

- How do we **quantify** the similarity of two text documents?
- How do we use a text document as input in a regression or classification model?
- **How do we turn a text document into a vector of numbers?**
  - A **design matrix** consists of one row per "data point", and one column per "feature".

	Employee Name	Job Title	Base Pay
0	Mara W Elliott	City Attorney	218759.0
1	Todd R Gloria	Mayor	218759.0
2	Elizabeth A Crisafi	Investment Officer	259732.0
3	Terence G Charlot	Police Officer	212837.0
4	Andrea H Tevlin	Independent Budget Analyst	224312.0

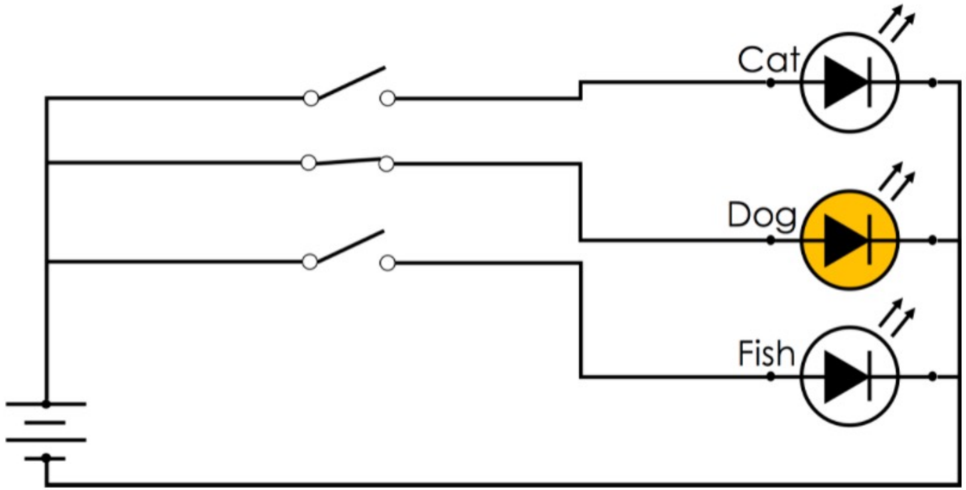
Salary data, python demo

Today’s Roadmap

- **Word Encoding**
  - **Bag of words**
  - TF-IDF
  - Word2Vector (Not covered)
  - BERT (Not covered)

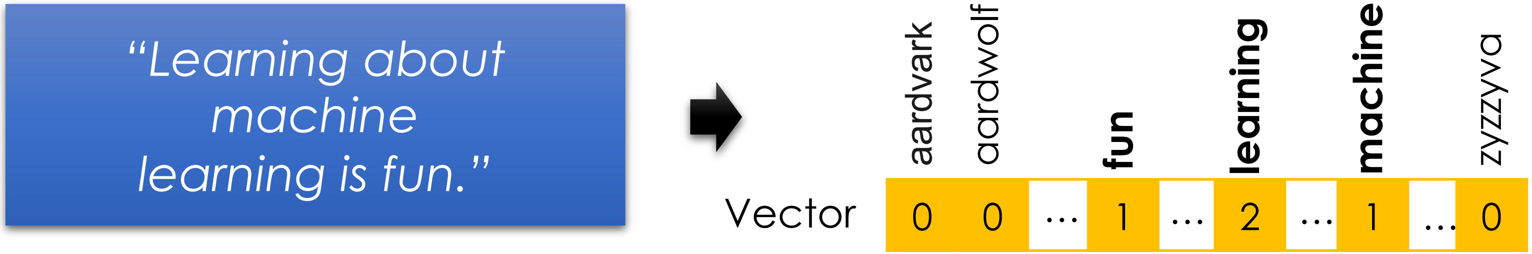
One-Hot Encoding

- One-Hot encoding, sometimes also called **dummy encoding**
- It is a simple mechanism to encode categorical data as real numbers such that the magnitude of each dimension is meaningful. Suppose a feature can take on k distinct values
- For each distinct *possible* value, a new feature (dimension) is created. For each record, all the new features are set to zero except the one corresponding to the value in the original feature.
- The term one-hot encoding comes from a digital circuit encoding of a categorical state as particular "hot" wire:



Bag-of-words Encoding

- Generalization of one-hot-encoding for a string of text:

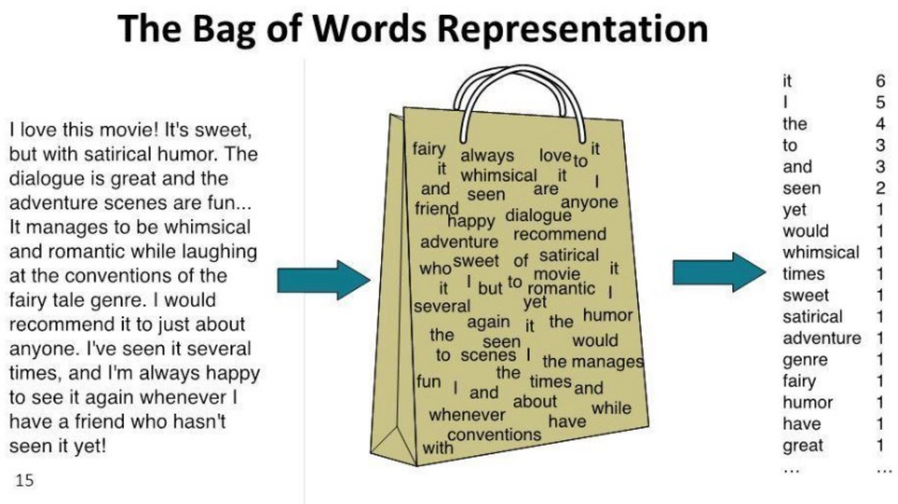


- A **bag** is another term for a multiset: *an unordered collection which may contain multiple instances of each element.*

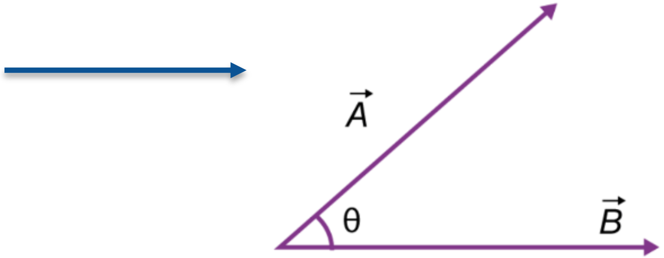
- **Stop words:** words that do not contain significant information
  - Examples: the, in, at, or, on, a, an, and …
  - Typically removed

From word vectors to similarity

Key idea: The more similar two unit vectors are, the larger their dot product is.



$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta$$



$$\cos\theta = \boxed{\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}}$$

$$\mathrm{dist}(\vec{a}, \vec{b}) = 1 - \cos\theta$$

Bag of words

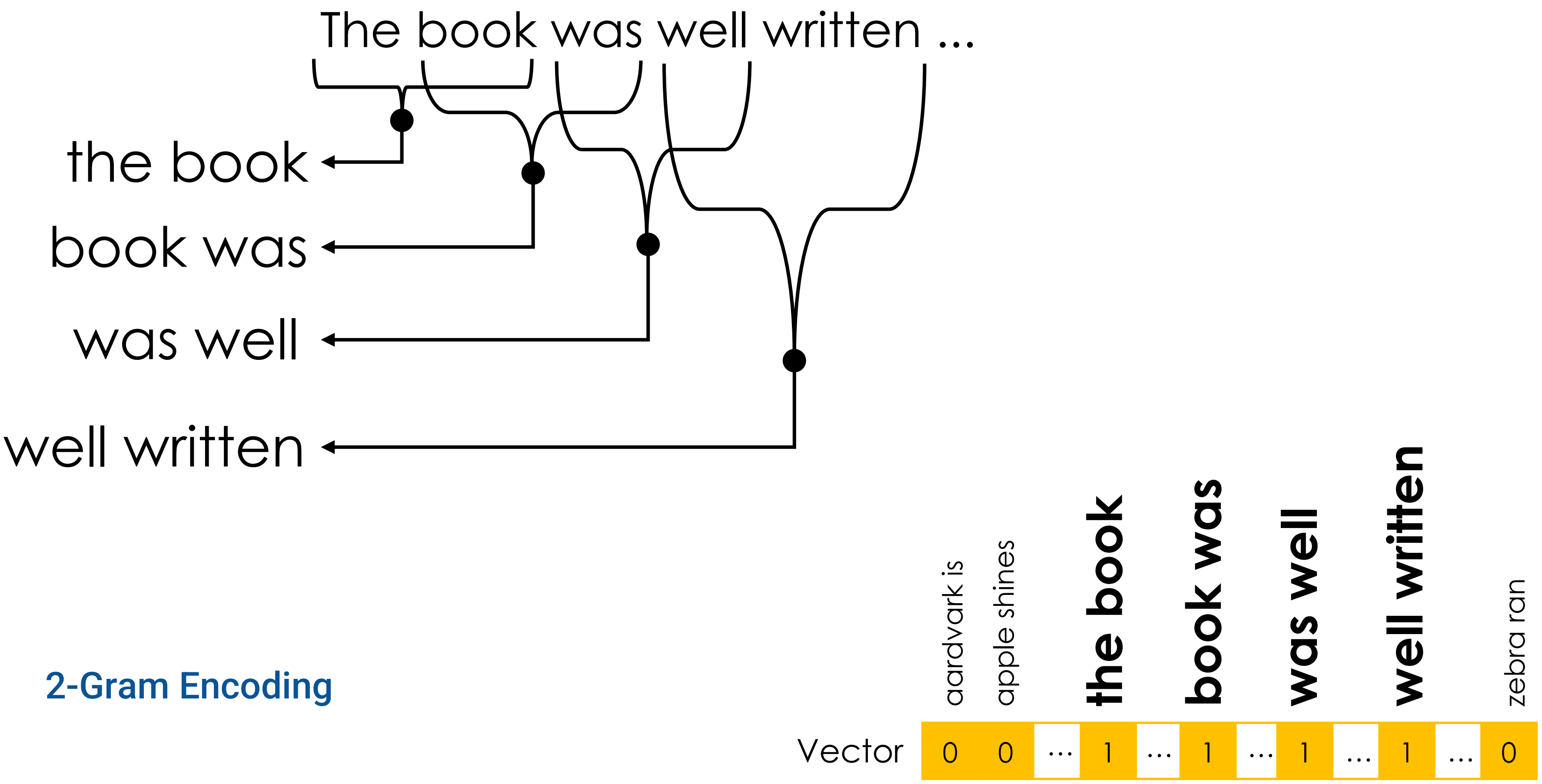
- Encode text as a long vector of word counts (Issues?)
  - Typically high dimensional (millions of columns) and very sparse
  - Word order information is lost… (is this an issue?)
  - What happens when you see a word not in the dictionary?

N-Gram Encoding

- Sometimes word order matters:

*The book was **not** well written but I did enjoy it.* ➡ *The book was well written but I did **not** enjoy it.*

- How do we capture word order in a "vector" model?
    - N-Gram: *"Bag-of- sequences-of-words"*



2-Gram Encoding

N-Gram Encoding

- Sometimes word order matters:

*The book was **not** well written but I did enjoy it.* ➡ *The book was well written but I did **not** enjoy it.*

- How do we capture word order in a "vector" model?
    - N-Gram: *"Bag-of- sequences-of-words"*
- Issues:
    - Can be very sparse (many combinations occur only once)
    - Many combinations will only occur at prediction time

Today's Roadmap

- Recap on text canonicalization
- **Word Encoding**
  - Bag of words
  - **TF-IDF**
  - Word2Vector (Not covered)
  - BERT (Not covered)

TF-IDF

- Motivation
  - The bag of words model doesn't know which words are "important" in a document.
  - How do we determine which words are important?
    - The most common words ("the", "has") often **don't** have much meaning!
    - The very rare words are also less important!

- TF (Term frequency)

The term frequency of a word (term) t in a document d, denoted tf(t,d), tf(t, d) is the proportion of words in document d that are equal to t.

$$\mathrm{tf}(t, d) = \frac{\#\text{ of occurrences of } t \text{ in } d}{\text{total }\#\text{ of words in } d}$$

Ex: What is the term frequency of "sam" in the following document?

- "my brother has a friend named sam who has an uncle named sam"

Issue with TF

- Intuition: Words that occur often within a document are important to the document's meaning.
  - If tf(t,d) is large, then word t occurs often in d.
  - If tf(t,d) is small, then word t does not occur often d.

- "my brother has a friend named sam who has an uncle named sam"
  - "Sam" and "has" have equal TF
  - Can we say they are both important?

IDF (Inverse document frequency)

- The **inverse document frequency** of a word t in a set of documents $d1$,$d2$,... Is

$$\text{idf}(t) = \log\left(\frac{\text{total \# of documents}}{\text{\# of documents in which } t \text{ appears}}\right)$$

Example: What is the inverse document frequency of "sam" in the following three documents?

- "my brother has a friend named sam who has an uncle named sam"
- "my favorite artist is named jilly boel"
- "why does he talk about someone named sam so often"

$$\log\left(\frac{3}{2}\right) \approx 0.4055.$$

- Intuition: If a word appears in every document (like "the" or "has"), it is probably not a good summary of any one document.
  - If idf(t) is large, then t is rarely found in documents.
  - If idf(t) is small, then t is commonly found in documents.
  - Think of idf(t) as the "rarity factor" of t across documents – the larger idf(t) is, the more rare t is.

TF-IDF

- **Goal:** Quantify how well word t **summarizes** document d.
  - If tf(t,d) is small, then t doesn't occur very often in d, so t can't be a good summary of d.
  - If idf(t) is small, then t occurs often amongst all documents, and so it is not a good summary of any one document.
  - If tf(t,d) and idf(t) are both large, then t **occurs often in** d **but rarely overall**. This makes t **a good summary** of document d.

$$
\begin{aligned}
\mathrm{tfidf}(t, d) &= \mathrm{tf}(t, d) \cdot \mathrm{idf}(t) \\
&= \frac{\text{\# of occurrences of } t \text{ in } d}{\text{total \# of words in } d} \cdot \log\left(\frac{\text{total \# of documents}}{\text{\# of documents in which } t \text{ appears}}\right)
\end{aligned}
$$

- (Demo)

Today’s Roadmap

- Recap on text canonicalization
- **Word Encoding**
  - Bag of words
  - TF-IDF
  - **Word2Vector (Not covered)**
  - **BERT (Not covered)**

Word2vec

- The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text
- It is vector based