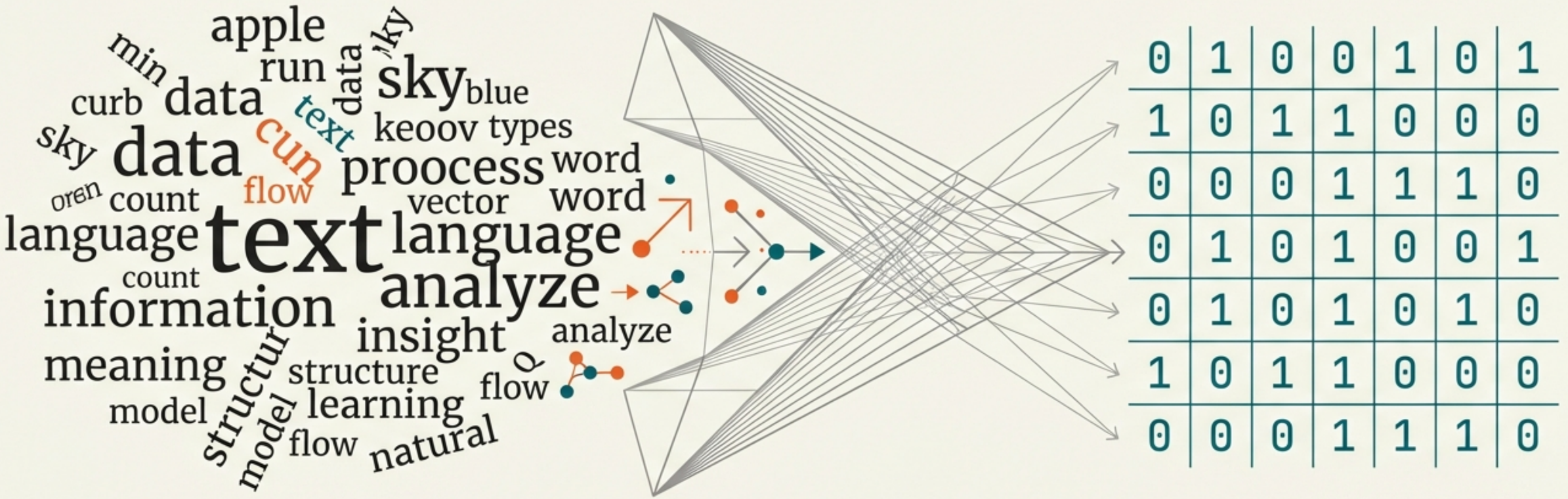
Unlocking Text Data: From Counts to Insights

A Foundational Guide to Bag of Words and TF-IDF Vectorization



FUNDAMENTAL NLP CONCEPTS

The Challenge of Speaking to Computers

Machines are mathematical engines, not linguistic ones. They cannot understand raw text or semantic meaning directly. Before any analysis can begin, we must bridge the gap between human language and machine logic.

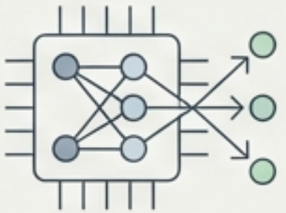
> **Vectorization**: The process of converting unstructured text into a numerical format (vectors) that a machine can process.

The Roadmap

1. **Bag of Words (BoW)**: Counting occurrences.
2. **TF-IDF**: Weighing importance.

Raw Text

Matrix / Vector



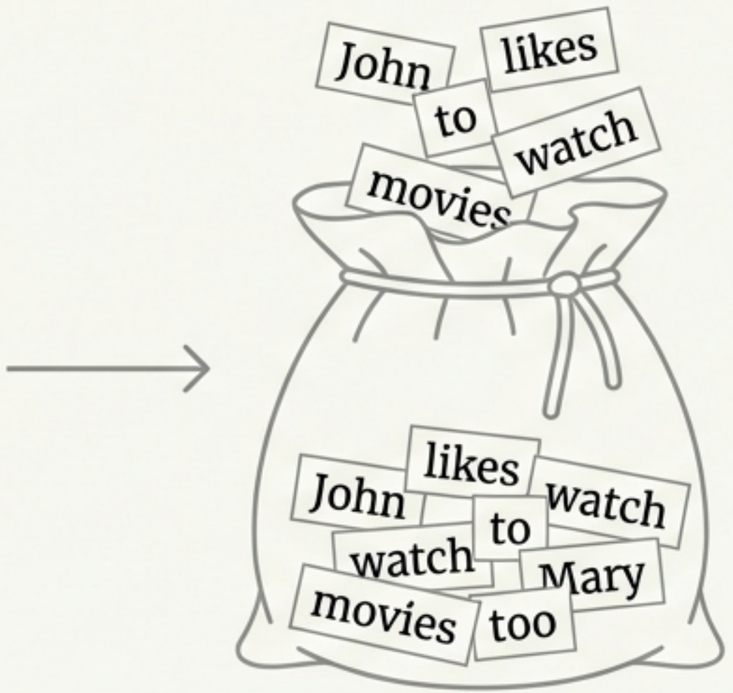
Machine Learning Model

The Bag of Words (BoW) Model

BoW is an algorithm that represents text by counting word frequency. It disregards grammar and word order, capturing only multiplicity. Imagine writing every word of a sentence on a slip of paper and shaking them inside a bag. You know what is inside, but the story is lost.

John likes to watch movies.

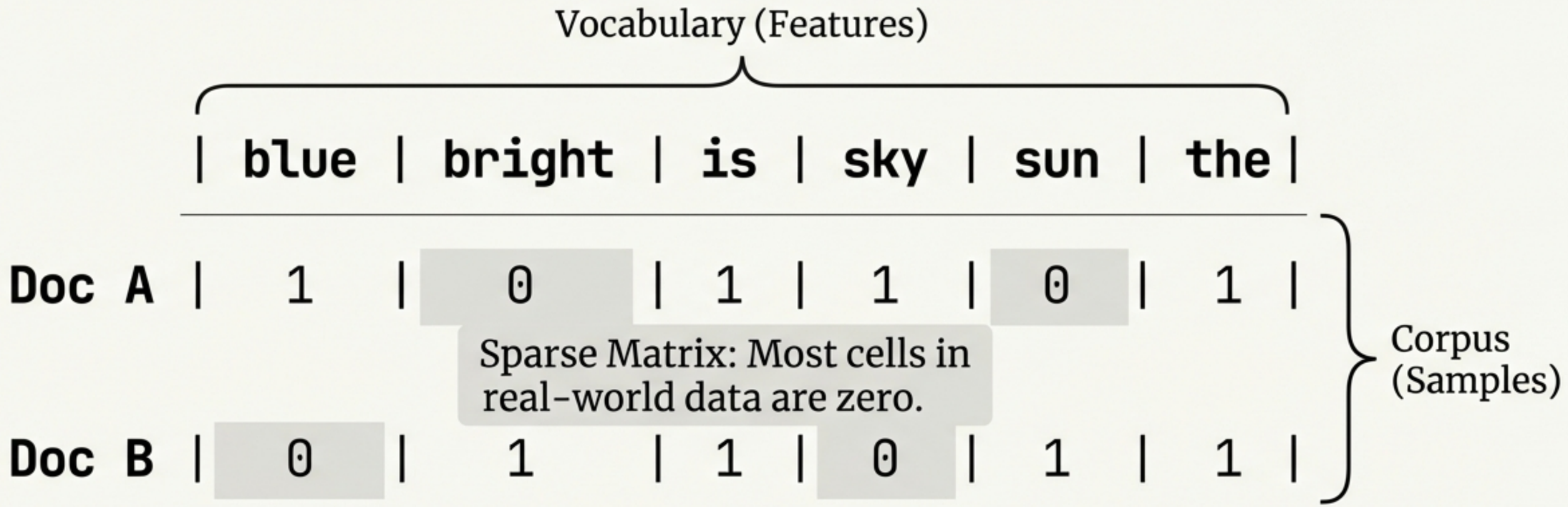
Mary likes movies too.




```
{
  'John': 1,
  'likes': 2,
  'to': 1,
  'watch': 1,
  'movies': 2,
  'Mary': 1,
  'too': 1
}
```

Visualizing the Document-Term Matrix

The conceptual ‘bag’ becomes a structured grid.

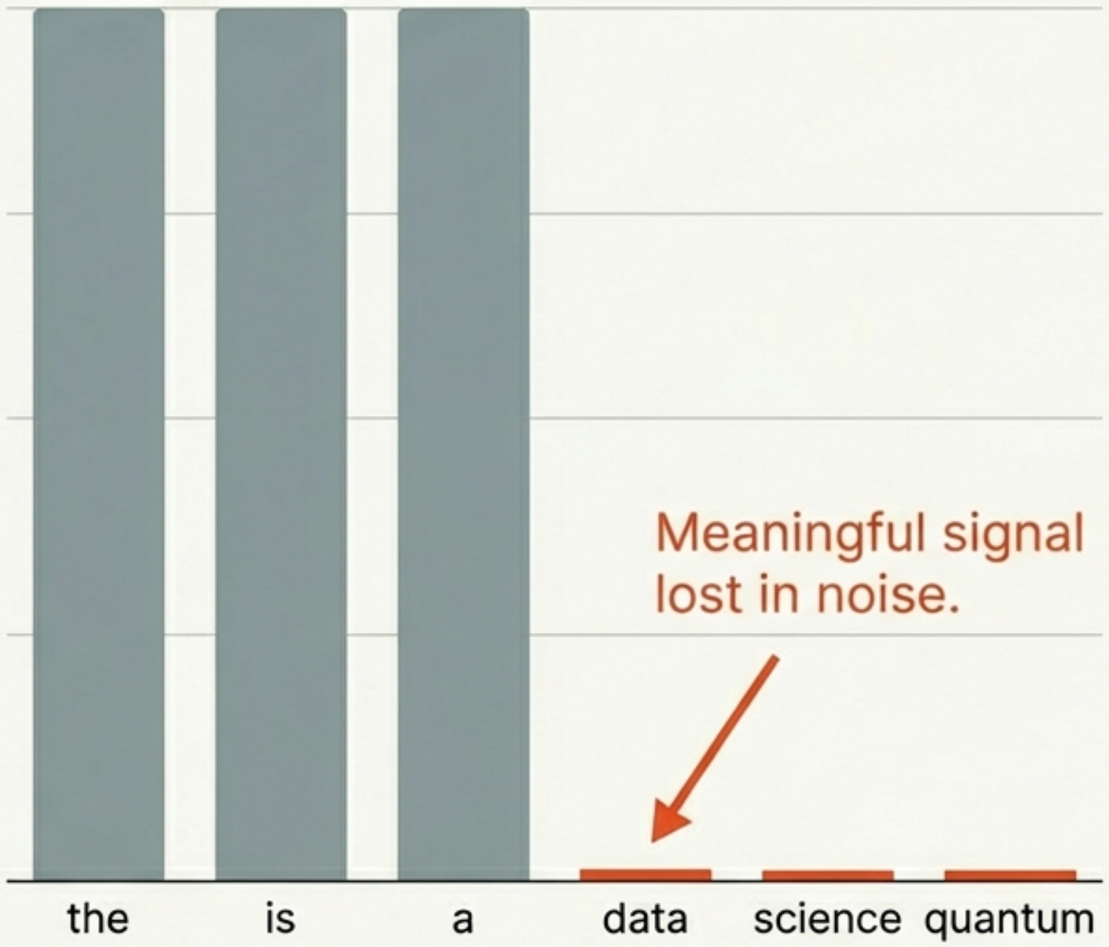


1. **Corpus**: The collection of documents.
2. **Vocabulary**: List of unique words.
3. **Vectorization**: Documents converted into fixed-length rows of counts.

The Limitation: Frequency ≠ Importance

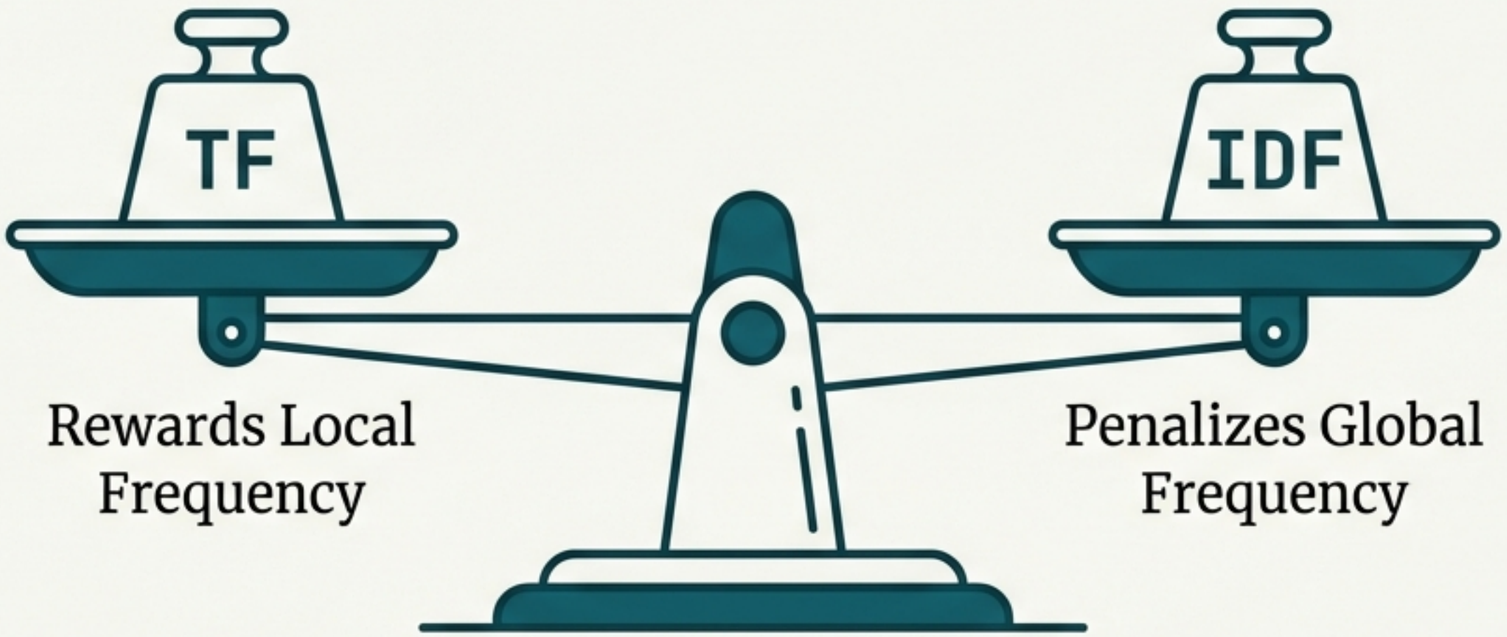
In a simple count model, the most frequent words are often the least informative. Stop words like ‘the’, ‘is’, and ‘and’ dominate the data, drowning out unique, meaningful keywords.

If we only count, a document about **‘The History of the Sun’** looks **identical** to **‘The Sun is Bright’** because the word **‘the’** overpowers everything else.



Enter TF-IDF: Weighting for Relevance

Term Frequency - Inverse Document Frequency



TF-IDF is a statistical measure that evaluates how relevant a word is to a document. It balances two opposing forces to highlight words that are **frequent locally** in a specific document but **rare globally** across the entire collection.

Deconstructing the Formula

$$\text{TF-IDF} = \text{TF} \times \text{IDF}$$

Component 1: Term Frequency (TF)

How often does the word (t) appear in this document (d)?

$$\text{TF}(t,d) = \frac{\text{Count of word t in doc d}}{\text{Total words in doc d}}$$

Component 2: Inverse Document Frequency (IDF)

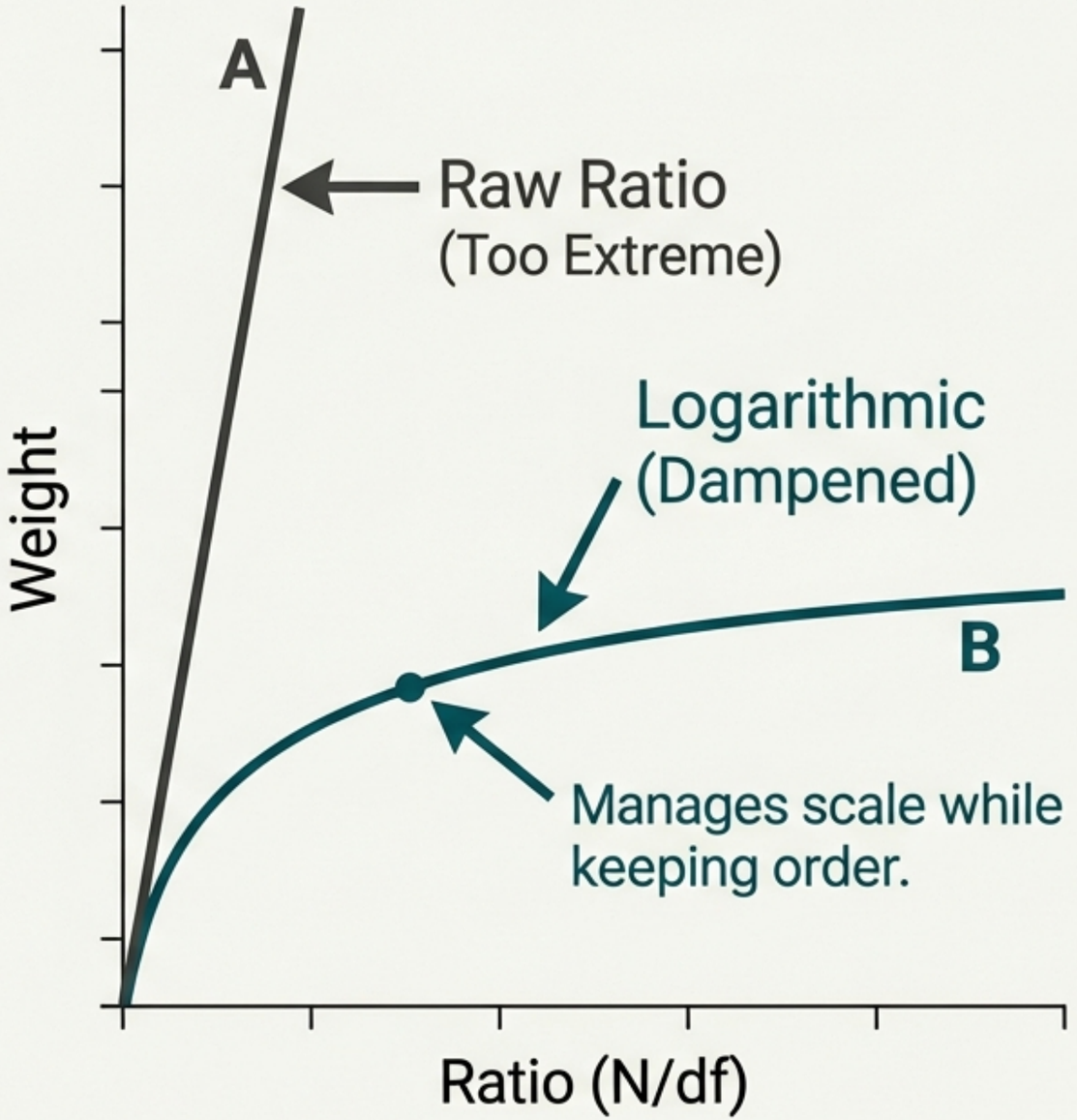
How rare is the word (t) across the entire corpus?

$$\text{IDF}(t) = \log\left(\frac{\text{Total Documents}}{\text{Documents with word t}}\right)$$

The Math Sidebar: Why Use a Logarithm?

Without a logarithm, the weight for rare words would explode. If we have 1,000,000 documents and a word appears in only 1, the raw ratio is 1,000,000. This is too extreme for a model to handle.

The `log()` function dampens this effect, scaling the number down to a manageable size (e.g., 6).



Step-by-Step Calculation: The Setup

Document A

The sky is blue

Document B

The sky is bright

Mission Parameters

Goal: Calculate TF-IDF for the words **‘blue’** and **‘sky’** in **Document A**.

Total Documents (N): 2
Total Words in Doc A: 4

Step 1: Calculate Term Frequency (TF)

Focus: Document A (‘The sky is blue’) in Merriweather

Word: ‘blue’
Count in Doc A: 1
Total Words in Doc A: 4
Calculation: 1 / 4

TF = 0.25

Word: ‘sky’
Count in Doc A: 1
Total Words in Doc A: 4
Calculation: 1 / 4

TF = 0.25

Based on TF alone, “blue” and “sky” appear equally important.

Step 2: Calculate Inverse Document Frequency (IDF)

Focus: Entire Corpus (Doc A + Doc B). N = 2.

Word: ‘blue’
Appears in: 1 Document (Doc A
A)
Formula: log(2 / 1) = log(2)

IDF ≈ 0.301

Word: ‘sky’
Appears in: 2 Documents (Doc
(Doc A, Doc B)
Formula: log(2 / 2) = log(1)

IDF = 0

‘Sky’ is penalized for being too common. Its weight drops to zero.

Step 3: The Final Score

TF × IDF = Score

`0.25 (TF) × 0.301 (IDF)`

→ **0.075**

High Relevance

`0.25 (TF) × 0 (IDF)`

→ **0.0**

Filtered Out (Noise)

The model successfully identifies ‘blue’ as the defining feature of Document A.

Summary: BoW vs. TF-IDF

Feature	Bag of Words (BoW)	TF-IDF
Metric	Merriweather Word Counts (Integer)	Merriweather Weighted Scores (Decimal)
Common Words	Dominate the vector	Penalized (approaches 0)
Complexity	Low (Simple Counting)	Moderate (Logarithms)
Interpretability	High	Moderate

When to Use Which Model?

Bag of Words

- Simple spam filtering
- Basic topic modeling
- When document length varies wildly
- When speed and simplicity are priority

TF-IDF

- Search engines (Relevance ranking)
- Keyword extraction
- User interest profiling
- When identifying unique content is critical

Key Takeaways

- Both models convert text into fixed-length vectors, allowing machines to process language mathematically.
- TF-IDF improves upon Bag of Words by adding “common sense” weighting—penalizing words that appear everywhere.
- The Logarithm is the mathematical tool that prevents rare word weights from exploding.
- Limitations: Both models ignore word order. “Dog bites man” and “Man bites dog” look identical to these algorithms.

Next steps in NLP: Word Embeddings (Word2Vec) and Transformers (BERT/GPT).