Bag of Words (BoW) and **TF-IDF**

Why we need these methods

When we work with text (like sentences or documents) in machine learning, we must **convert words into numbers** because computers can only understand numbers. **Bag of Words (BoW)** and **TF-IDF** are two common ways to do that.

1. Bag-of-Words (BoW)

Idea: Count how many times each word appears in a sentence or document.

Example:
We have three short sentences:

1. Text processing is necessary.
2. Text processing is necessary and important.
3. Text processing is easy.

First, make a list of all unique words (the *vocabulary*):
{Text, processing, is, necessary, and, important, easy}

Now, for each sentence, count how many times each word appears:

Document	Text	processing	is	necessary	and	important	easy
1	1	1	1	1	0	0	0
2	1	1	1	1	1	1	0
3	1	1	1	0	0	0	1

Advantages:

- Simple to understand and implement.
- Works well for small datasets.

Limitations:

- Produces **large vectors** with many zeros (sparse data).
- **Ignores meaning and word order.**
  Example: “Text processing is easy but tedious” and “Text processing is tedious but easy” look the same to BoW.

2. TF-IDF (Term Frequency – Inverse Document Frequency)

Idea:
It improves on BoW by giving **more importance to rare but useful words** and **less importance to very common words**.

Term Frequency (TF)

How often a word appears in a document.

$$TF = \frac{\text{No. of times word appears in doc}}{\text{Total words in doc}}$$

Inverse Document Frequency (IDF)

Measures how rare a word is across all documents.

$$IDF = \log\left(\frac{\text{Total No. of documents}}{\text{No. of documents containing the word}}\right)$$

TF-IDF value

$$TFIDF = TF \times IDF$$

So, common words like *"is"*, *"the"*, *"and"* get **low scores**, while rare, meaningful words like *"important"* or *"easy"* get **higher scores**.

□ **Limitation:**
Like BoW, TF-IDF also **does not capture the context or meaning** — it only considers word frequency.

Summary

Feature	Bag-of-Words	TF-IDF
What it does	Counts words	Weighs words by importance
Common words	Treated equally	Given low importance
Context	Not captured	Not captured
Output	Word count vectors	Weighted frequency vectors

Example on TF-IDF

We have **two short documents**:

- **Doc 1:** "Text processing is necessary"
- **Doc 2:** "Text processing is necessary and important"

Step 1: Find all unique words (vocabulary)

Vocabulary = {text, processing, is, necessary, and, important}

There are 6 words in total.

Step 2: Calculate **Term Frequency (TF)**

TF = (Number of times word appears in the document) ÷ (Total words in the document)

Word	TF in Doc 1	TF in Doc 2
text	1/4 = 0.25	1/6 ≈ 0.17
processing	1/4 = 0.25	1/6 ≈ 0.17
is	1/4 = 0.25	1/6 ≈ 0.17
necessary	1/4 = 0.25	1/6 ≈ 0.17
and	0	1/6 ≈ 0.17
important	0	1/6 ≈ 0.17

Step 3: Calculate **Inverse Document Frequency (IDF)**

IDF = log(Total number of documents ÷ Number of documents containing the word)

We have **2 documents**, so:

Word	No. of Docs Containing the Word	IDF = log(2 ÷ count)
text	2	log(2/2) = 0
processing	2	0
is	2	0
necessary	2	0
and	1	log(2/1) = 0.301
important	1	log(2/1) = 0.301

(Using base 10 log for simplicity.)

Step 4: Calculate **TF-IDF = TF × IDF**

Word	TF-IDF in Doc 1	TF-IDF in Doc 2
text	0.25 × 0 = **0**	0.17 × 0 = **0**
processing	0.25 × 0 = **0**	0.17 × 0 = **0**
is	0.25 × 0 = **0**	0.17 × 0 = **0**
necessary	0.25 × 0 = **0**	0.17 × 0 = **0**
and	0 × 0.301 = **0**	0.17 × 0.301 ≈ **0.051**
important	0 × 0.301 = **0**	0.17 × 0.301 ≈ **0.051**

Step 5: Interpret the result

- Words like **“text”, “processing”, “is”, “necessary”** appear in **both** documents → **IDF = 0**, so TF-IDF = 0.
- Words like **“and”** and **“important”** appear in only **one document** → they get **higher TF-IDF scores**, meaning they are **more unique and informative** for that document.

Final Takeaway

TF-IDF helps identify **important and unique words** in a document by:

- rewarding frequent but **rare** words,
- and penalizing very **common** ones.