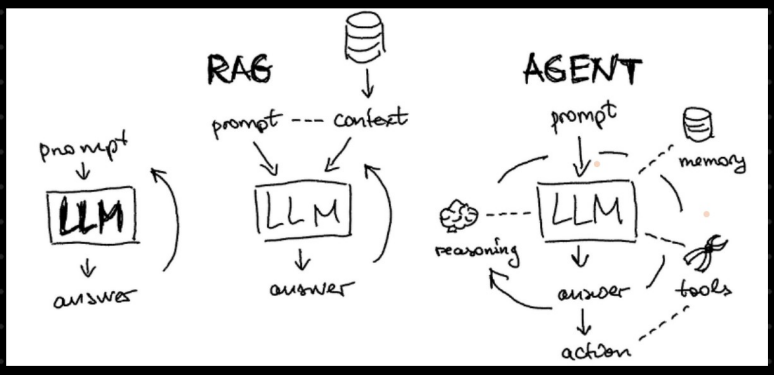
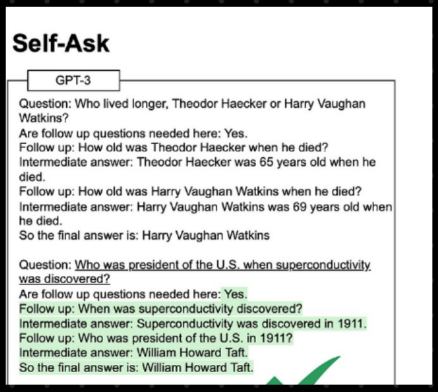
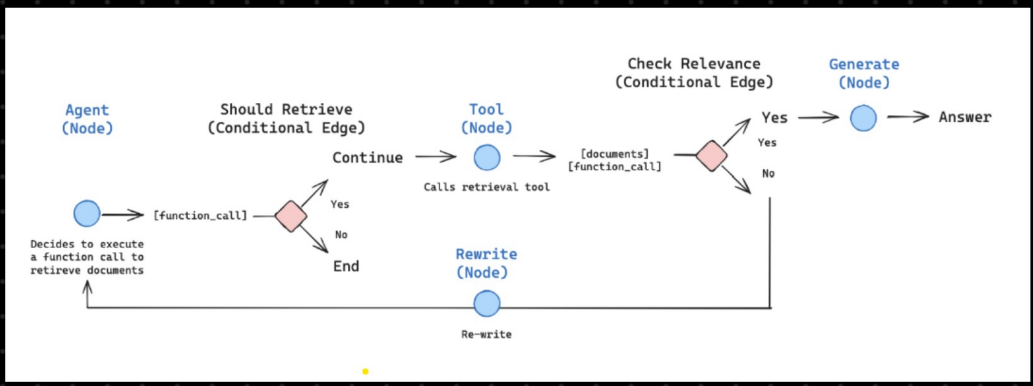
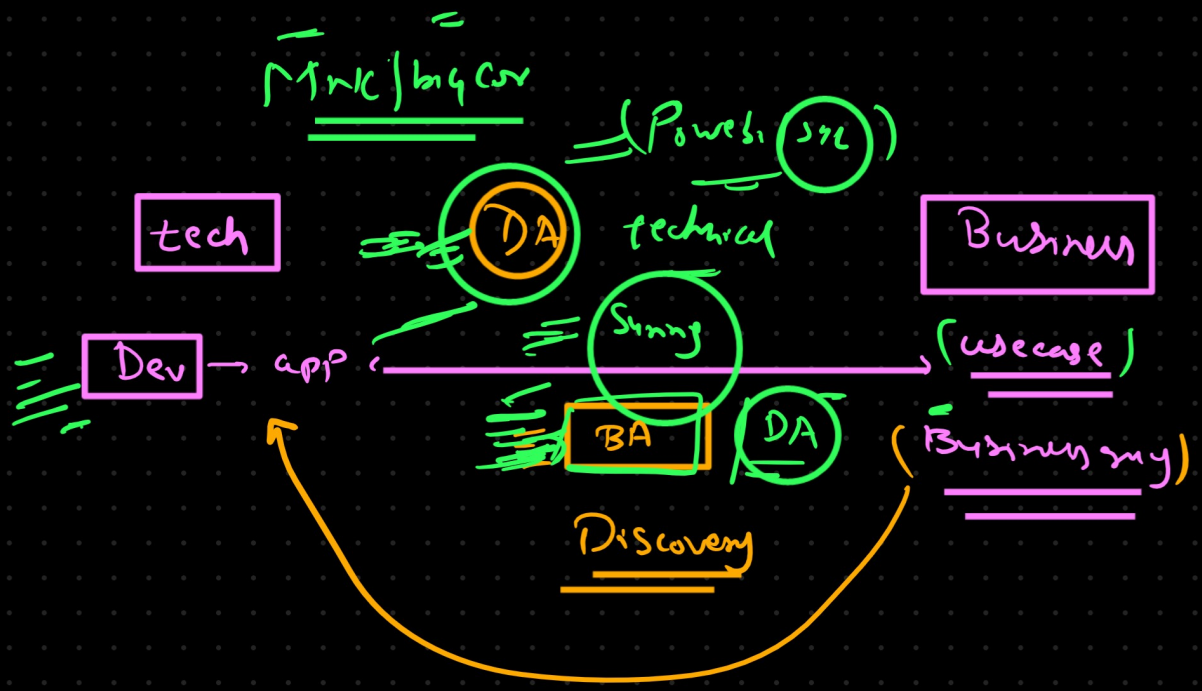

RAG
prompt
LLM
answer
prompt --- context
LLM
answer
AGENT
prompt
memory
reasoning
LLM
answer
tools
action

TAO


Self-Ask
GPT-3
Question: Who lived longer, Theodor Haecker or Harry Vaughan Watkins?
Are follow up questions needed here: Yes.
Follow up: How old was Theodor Haecker when he died?
Intermediate answer: Theodor Haecker was 65 years old when he died.
Follow up: How old was Harry Vaughan Watkins when he died?
Intermediate answer: Harry Vaughan Watkins was 69 years old when he died.
So the final answer is: Harry Vaughan Watkins
Question: Who was president of the U.S. when superconductivity was discovered?
Are follow up questions needed here: Yes.
Follow up: When was superconductivity discovered?
Intermediate answer: Superconductivity was discovered in 1911.
Follow up: Who was president of the U.S. in 1911?
Intermediate answer: William Howard Taft.
So the final answer is: William Howard Taft.


Agent
(Node)
Should Retrieve
(Conditional Edge)
Tool
(Node)
Check Relevance
(Conditional Edge)
Generate
(Node)
Continue
Calls retrieval tool
[documents]
[function_call]
Yes
Answer
Yes
No
[function_call]
Yes
No
Decides to execute a function call to retireve documents
End
Rewrite
(Node)
Re-write


tech
DA
technical
Business
Dev
app
Sunny
usecase
BA
DA
Discovery

(Text Processing)

Analysis

{Punc, stopword, emoji}

Python script

Stopwords — making ↓ sence

~~I~~ ~~am~~ (Sunny Saurabh)

~~who~~ (teach genAI)

[I, am, who]

(Sentences)

NLP ⇒ Text

entire data

1. Paragraph → CORPUS
2. Sentence → Document ⇒ RAG (Doc)
3. words → tokens ⇒ token limit
4. character → character

Chunking

{gpt 3.5, gpt 4, gemini}

Vocabulary → Collection of unique words (tokens)

token ≠ word

Word, Sentence

↳ langchain, Llamaindex.

desired result

Whatever technique

Regex → optional

Custom

Split

NLTK or other libraray

word_tokenize, sent_tokenize

Stemming => root form → not accurate

Lemmetization => original word

Base Form (root)

↓

accurate

Stemming
Definition: Stemming is the process of reducing a word to its root form by removing suffixes or prefixes, often without considering the word's meaning.

Key Characteristics:

Usually rule-based or algorithmic.
The output may not always be a valid word.
Focuses on quick and computationally efficient reduction of words.

Example:

Words like running, runner, and ran might be reduced to the stem run.
Similarly, better might be stemmed to bett.
Common Stemming Algorithms:

Porter Stemmer
Lancaster Stemmer
Snowball Stemmer

Advantages:
Simple and fast.
Useful in scenarios where slight inaccuracies are acceptable, like search engines.
Disadvantages:

Often produces stems that aren't actual words (e.g., studies → studi).

Text Preprocessing

Huge data ⇒ messy, ambiguous data

↑

Your understanding

↑

text Preprocessing

Definition: Lemmatization reduces a word to its base form (lemma) while considering the context and the word's grammatical meaning (e.g., part of speech).

Key Characteristics:

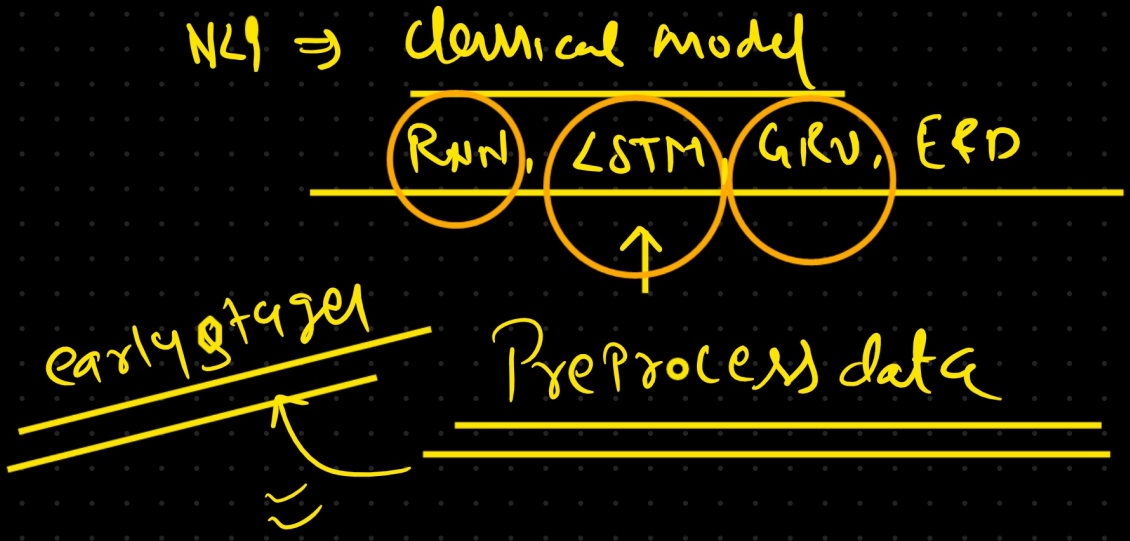
More sophisticated and linguistically informed.
The output is always a valid word.
Requires tools like a dictionary or vocabulary resource to find the base form.
Example:

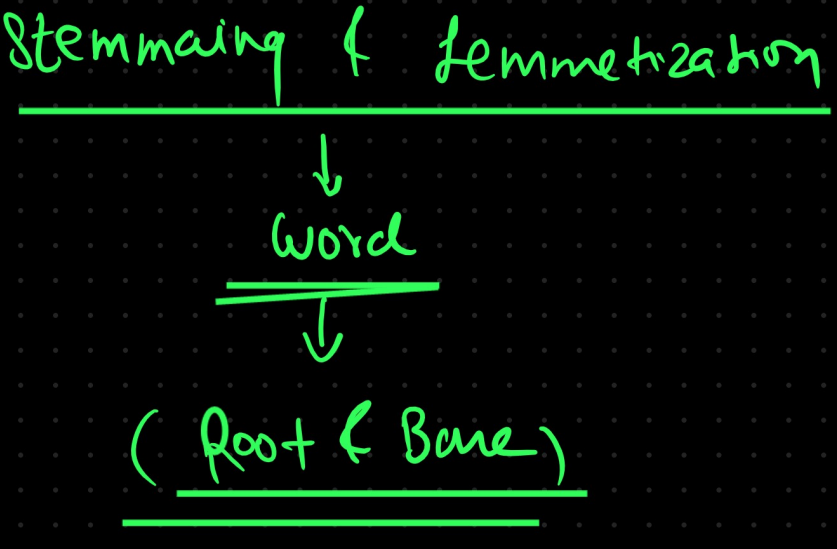
running → run
better → good (context-aware, since "better" is the comparative form of "good").
Common Lemmatization Tools:

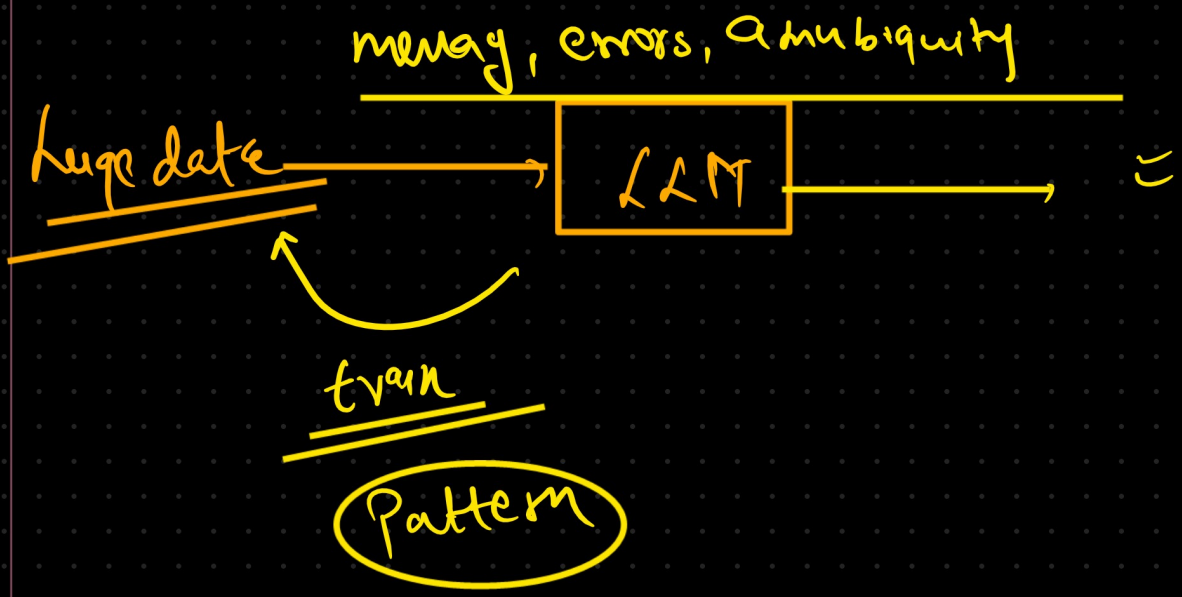
WordNet Lemmatizer (in NLTK)
spaCy (built-in lemmatizer)
Advantages:

Produces more accurate base forms.
Useful in linguistically rigorous tasks like machine translation or sentiment analysis.
Disadvantages:

Slower and computationally more expensive than stemming.
Requires additional information, such as part-of-speech tags.







Q. IS text Preprocessing req. for LLM?

Most of time → NO

but if data having lot's of error in that case YES

{ Punc, Stop word, ambiguous & redundant word, emoji }

tokenizer ⇒ Langchain

Encoding & Embedding

= vector

{ Numeric representation of text }

Text → Number

Encoding => frequency based method

Embedding => Neural network based method

=> text → Number → vector

= Set of number

Encoding (classical method) — OLD

lookup to SMP

1. OHE
2. BOW
3. TF-IDF
4. N-GRAMS
5. Custom method

Embedding (classical) = OLD

1. Word2vec

Word to vector → Neural network

Glove → Math (matrix factorization)

State of Art technique (New)

transformer Arch

Sentence embedding

1. BERT
2. ELMo
3. Gemini
4. openAI
5. MPT

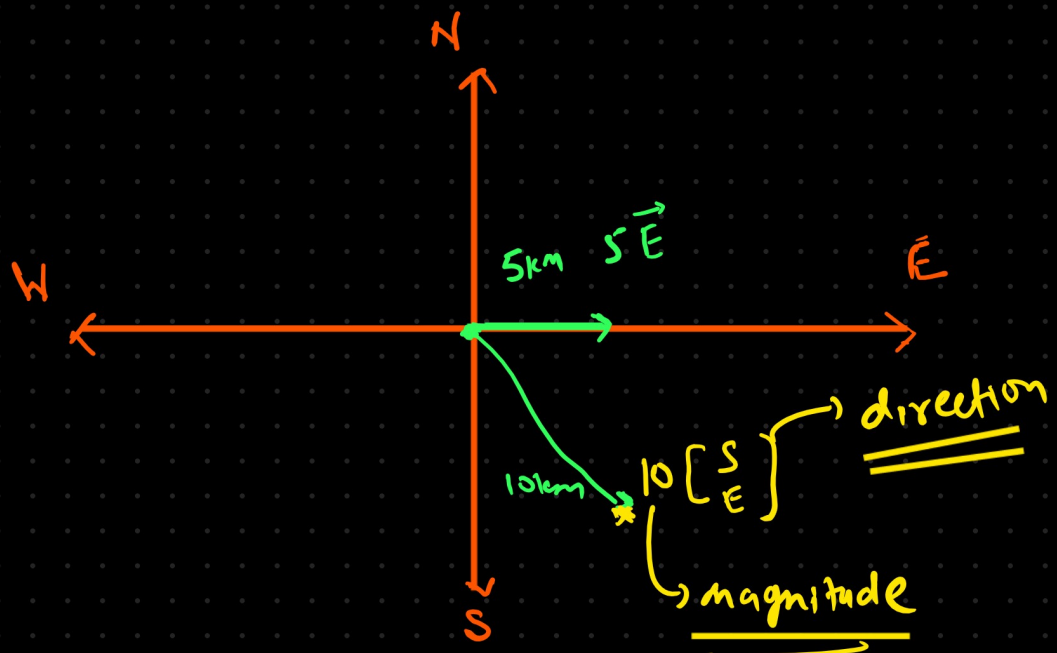
(Senfence transformer) ⇒ huggging

Sentence ⟶ embedding (Numric)

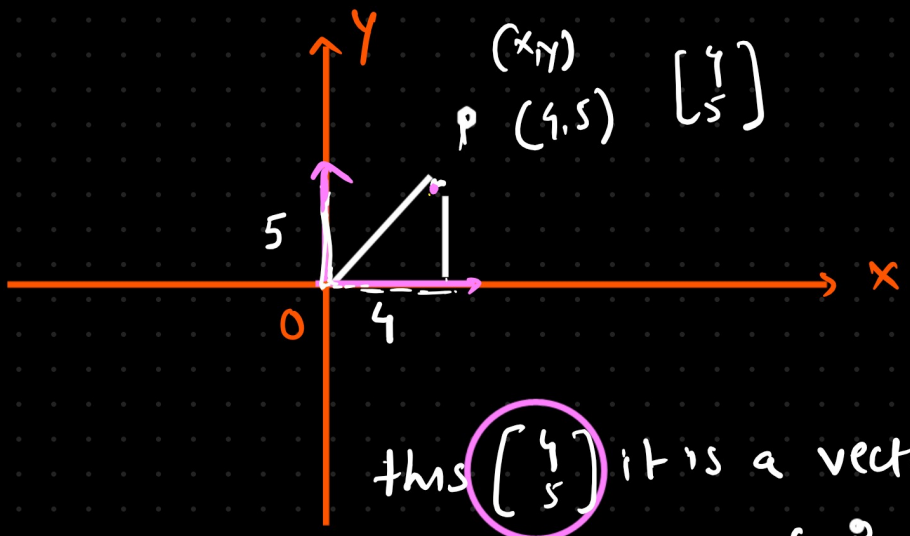
vector (set of Num)

Data → ST → Transformer → Sentence embedding

Vectors



Physics ⇒ M + D = V



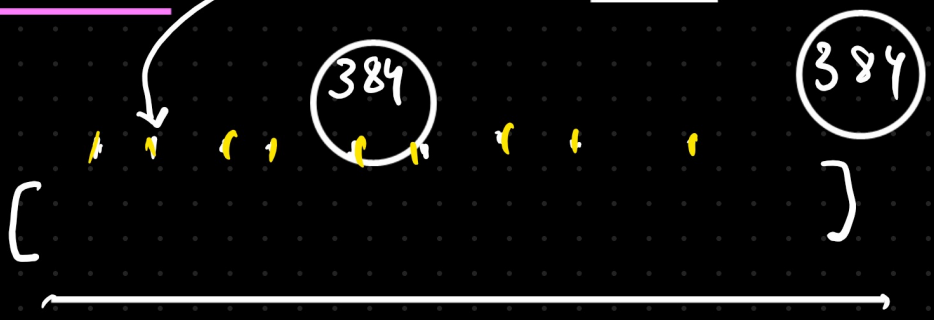
this $\begin{bmatrix} 4 \\ 5 \end{bmatrix}$ it is a vector representation of Point P

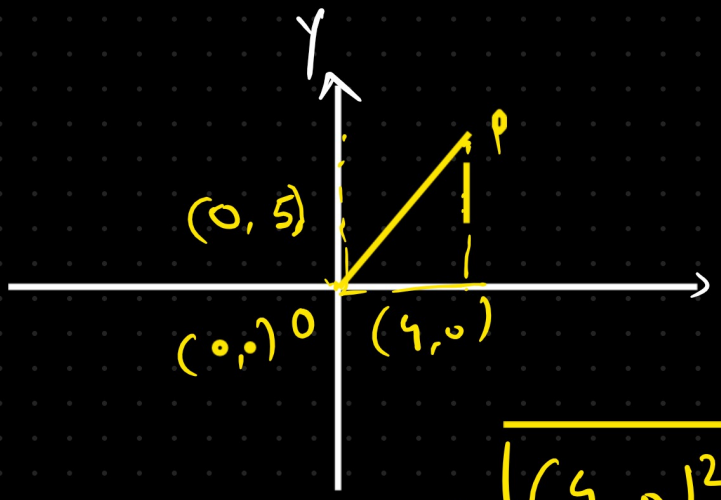
? → (P → [4,5])

2D

2dim → x, y

Word 2 vec ⇒ Sunny ⇒ 384 ↗ by default





$\overrightarrow{OP}$ = M + D

=

$\sqrt{(4-0)^2 + (5-0)^2}$

$\sqrt{16+25}$

$\sqrt{41}$

$\overrightarrow{OP} = \sqrt{41}\begin{bmatrix} x \\ y \end{bmatrix}$

$\overrightarrow{OP} = \sqrt{41}\begin{bmatrix} 4 \\ 5 \end{bmatrix}$

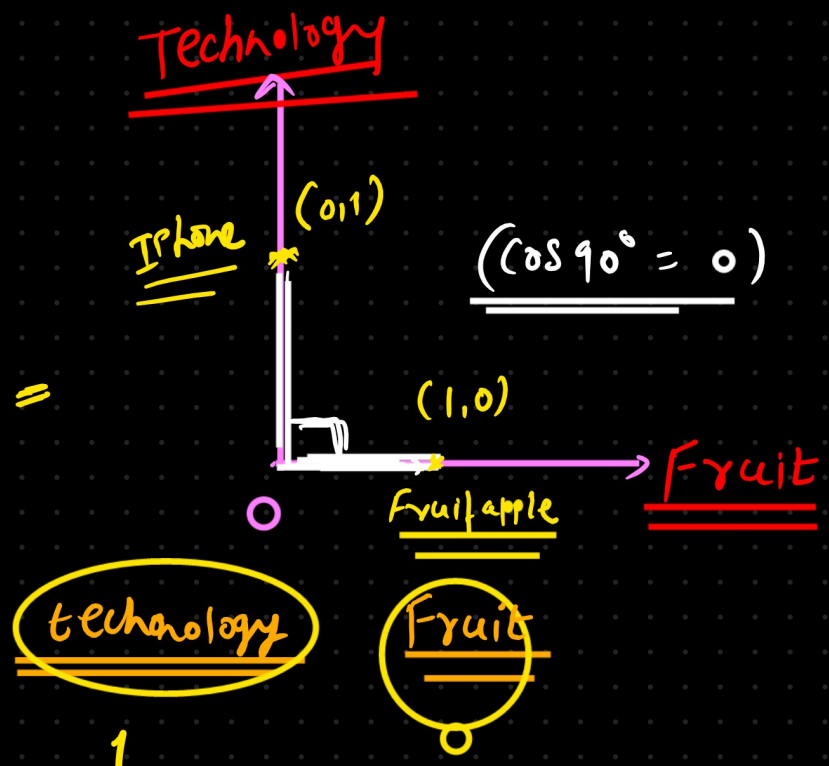
$OP = \sqrt{41}$ = $OP = \begin{bmatrix} 4 \\ 5 \end{bmatrix}$

Dimension

OP or P $\begin{bmatrix} 4, & 5 \\ x & y \end{bmatrix}$

Apple

1. Iphone
2. Fruit Apple =



Iphone

1

Iphone → [1,0] ← vector

Apple → [0,1] ← vector

Similimty Search

1. Dot Product
2. Cosine Simi.

[1,0]
[0,1]

$(x_1 y_1)\ (x_2 y_2)$

$(x_1 * x_2) + (y_1 * y_2)$

$(1 * 0) + (0 * 1)$

$0 + 0 = 0$

RAG => Semantic Search