

Class of 5  
octanobot.

CLAS

## Vector Databases

firing instruction database: so canomolunit. pag:  
 u bedd sacreled pag storeon (pag / oratraton).

B CLARKE

**Mg. Ing. Ezequiel Guinsburg**  
ezequiel.guinsburg@gmail.com

A Inwvvectoron ofalabababor. b

- Rag I Vector nind deganetion  
Rag fatraibon
- RGA dalnolabalse  
rag: ditalase:
- Rag codes ditalase  
connected nitalase

### Bf oattabae sion

- Ass orf teicet b onape  
chanling chatbot**

### Proibats 6

- Patenniuclitig grivuel, atineg  
 Patnifotodroozen  
 Patron con'wentelings-sho  
 oeddestruoorpantolo  
 Pcrinfoort boducon  
 Perte ihvmedicacy  
 Paton bhannetig  
 Patnd inipurfukling  
 Paton biwiohls oomappy  
 Re-Queen of chatting toask  
 oovpactsoor

# Temas:

- Bases de datos vectoriales (en contexto RAG)
- Retrieval Augmented Generation (RAG)
- RAG Multimodal
- Chatbots

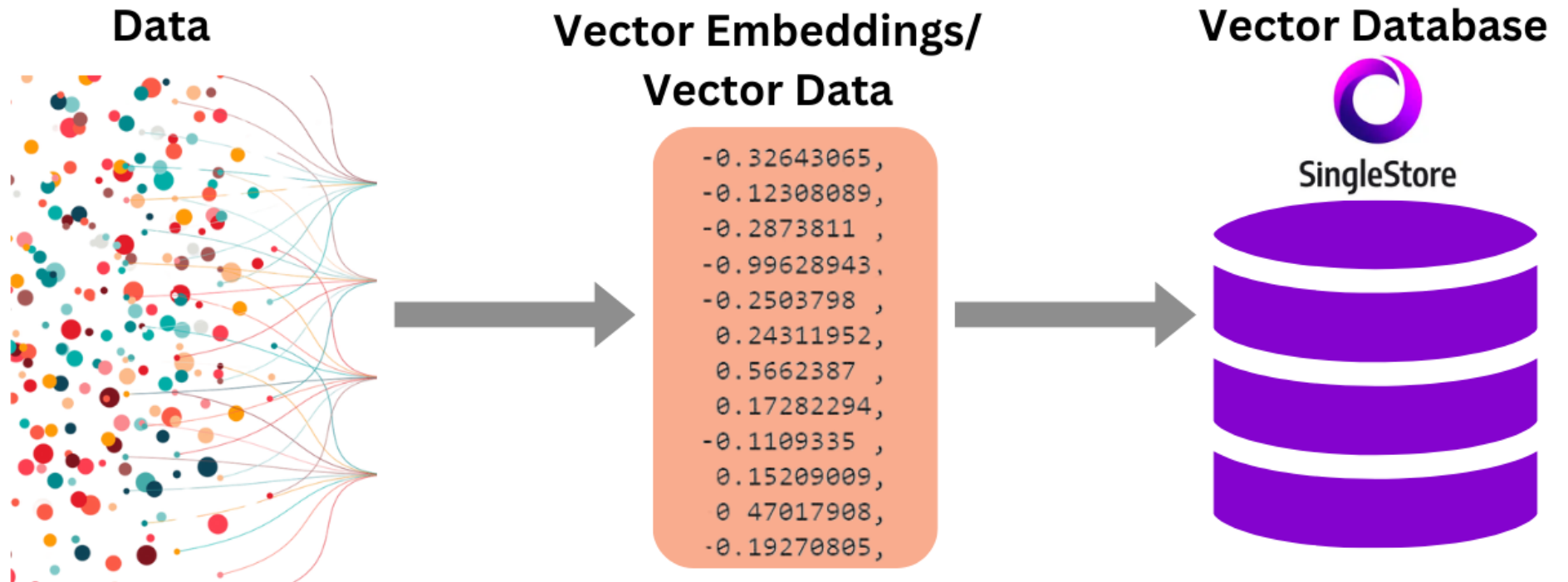
## Referencias:

- Paper “IMAGEBIND: One Embedding Space To Bind Them All”
- Paper “Retrieval-Augmented Generation for Knowledge Intensive NLP Tasks”
- Paper FAISS (Billion-scale similarity search with GPUs)

[Link REPO](#)

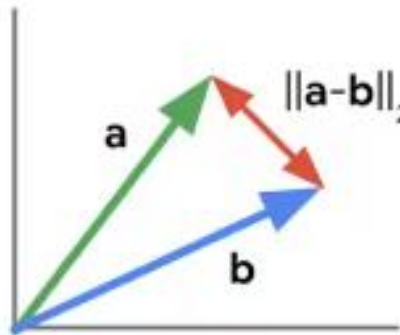
# Bases de datos vectoriales:

- Características

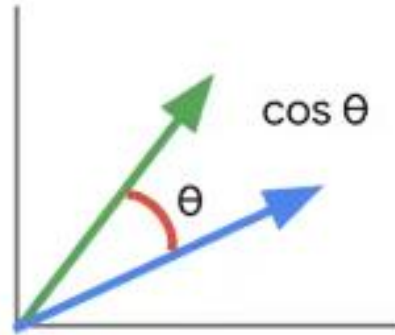


# Bases de datos vectoriales:

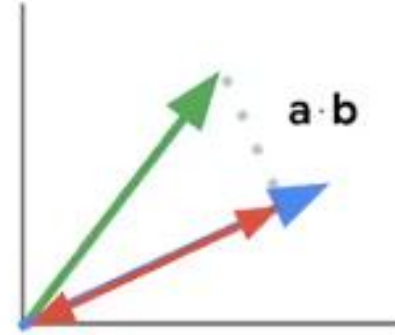
- Búsqueda por similitud
  - Distancia de Hamming( $A, B$ ) =  $\sum (A_i \neq B_i)$
  - $L1(A, B) = \sum |A_i - B_i|$



L2 distance



cosine similarity



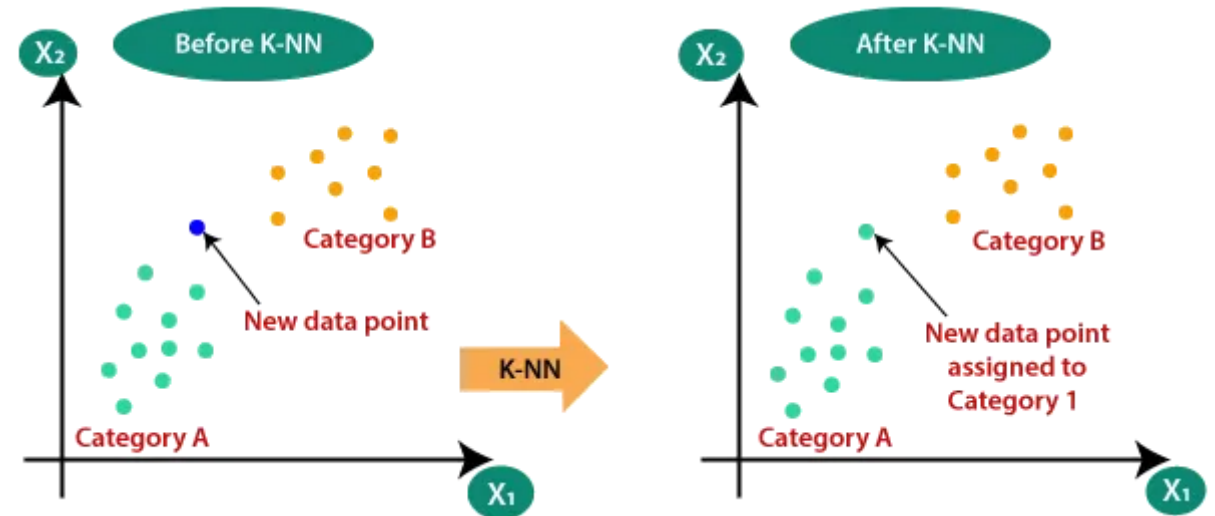
inner product

# Bases de datos vectoriales:

- Algoritmos de ordenamiento para búsquedas eficientes:
  - k-dimensional tree
  - Locality Sensitive Hashing (LSH)
  - Faiss (Facebook AI Similarity Search) ([link](#))

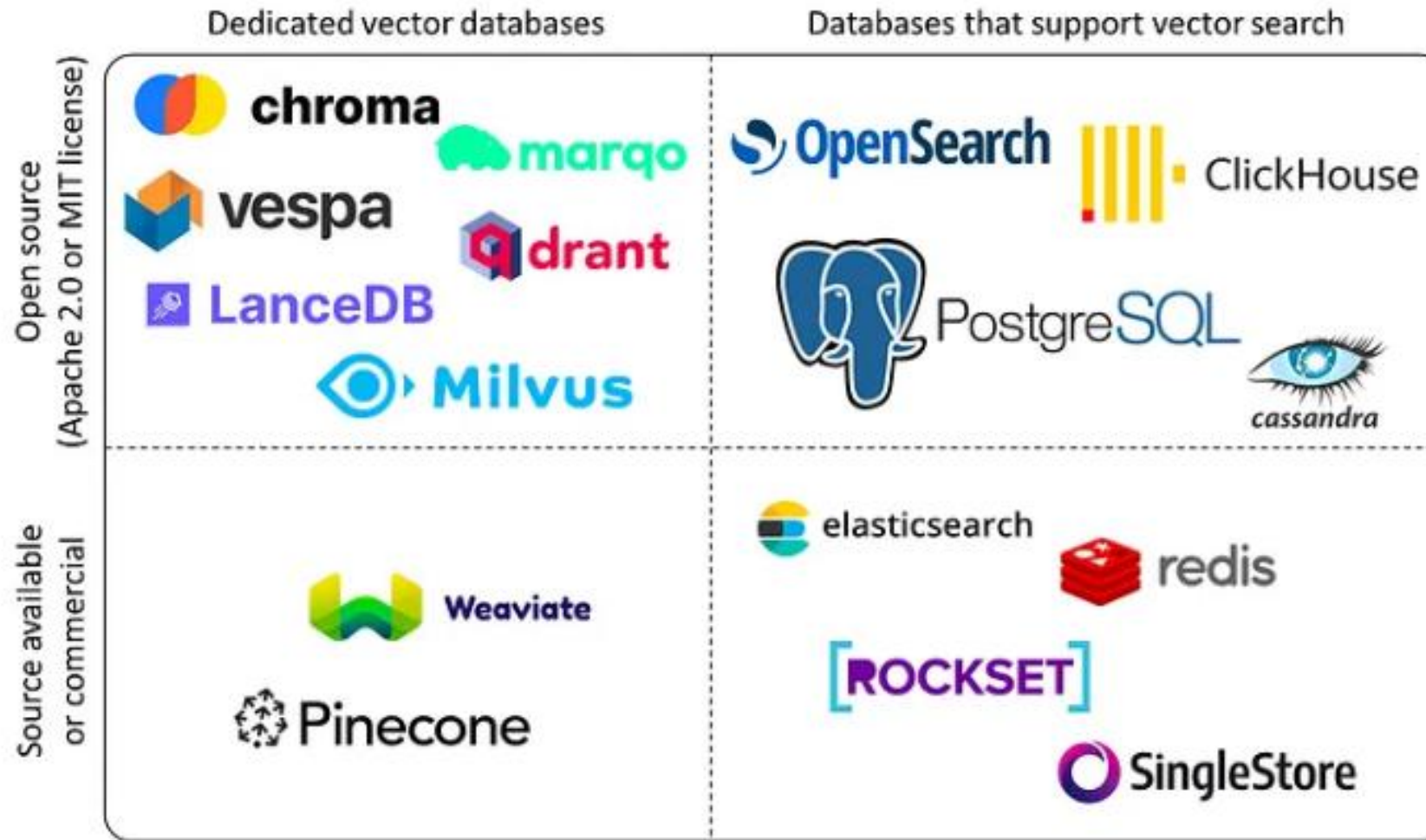


Source: [Medium](#)



# Bases de datos vectoriales:

- Panorama





# Bases de datos vectoriales:

- Ejemplo

<https://colab.research.google.com/drive/18eu4NyMFdJNho8qIUwqTNfvoERZvXqhX?usp=sharing>

- Re-ranking: Técnica utilizada para mejorar los resultados de una búsqueda o recuperación de información. En lugar de depender únicamente de la primera búsqueda para proporcionar los resultados más relevantes, el re-ranking ajusta el orden de esos resultados con la ayuda de un modelo adicional que evalúa la relevancia semántica de los resultados con respecto a la consulta inicial.



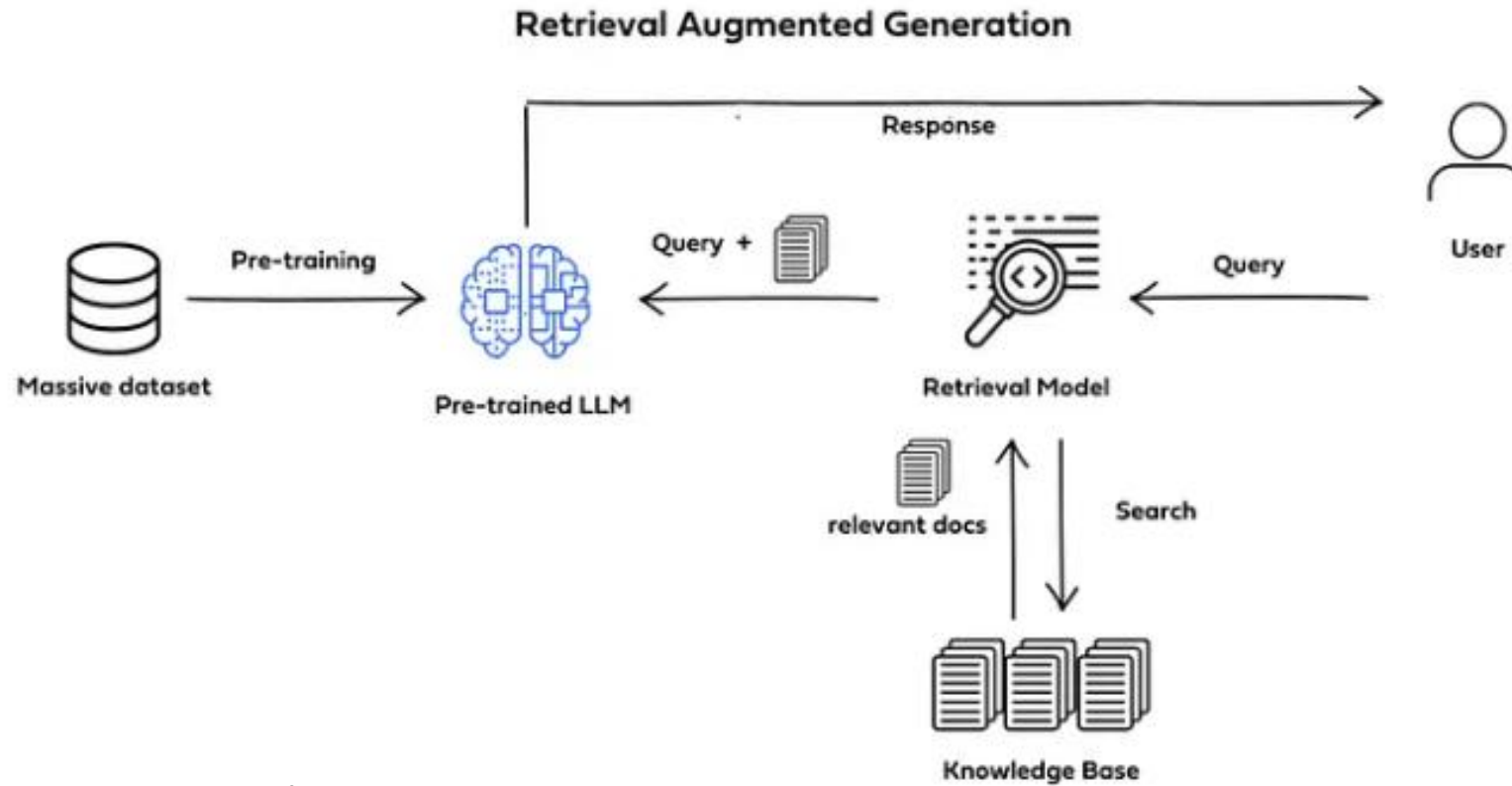
# Bases de datos vectoriales:

- Búsquedas Híbridas

<https://docs.pinecone.io/guides/data/understanding-hybrid-search>

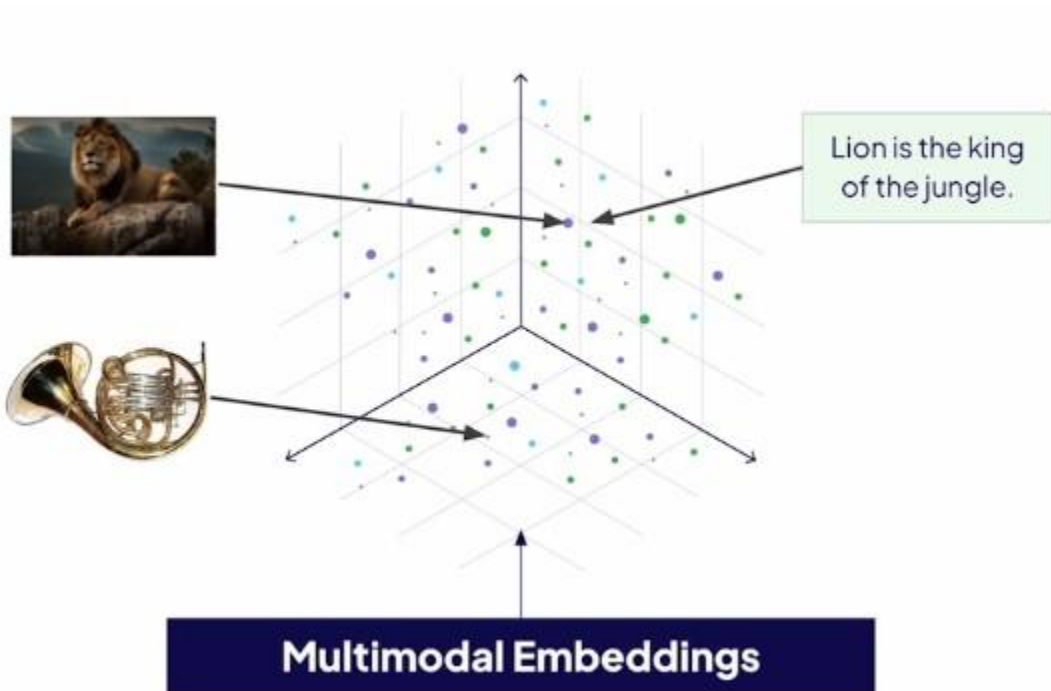
<https://github.com/pinecone-io/examples/blob/master/learn/search/hybrid-search/ecommerce-search/ecommerce-search.ipynb>

# Retrieval Augmented Generation (RAG):



Fuente: medium

# RAG Multimodal:



The lion is the king  
of the savannas.



Text Encoder

$[-0.10, 0.02, 0.14, \dots, 0.48]$



Image Encoder

$[0.66, 0.73, -0.42, \dots, 0.52]$



Audio Encoder

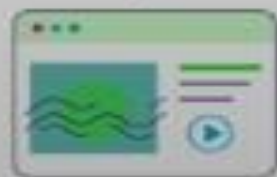
$[-0.61, 0.31, 0.88, \dots, 0.22]$



Video Encoder

$[-0.21, 0.15, =0.27, \dots, -0.59]$

Step 1



Customized  
Response

Step 2

# RAG Multimodal:

Ejemplo:

[https://colab.research.google.com/drive/1o\\_h\\_GNH\\_clwxe9G5ysvfKHGvpBZsXN8o?usp=sharing](https://colab.research.google.com/drive/1o_h_GNH_clwxe9G5ysvfKHGvpBZsXN8o?usp=sharing)

# Chatbots

- Simulación del contexto
- Memoria persistente gestionada programáticamente (Langchain)

Vemos el ejemplo de [la implementación de un chatbot paso a paso!](#)

## Ejercicio en clase:

- Implementar un sistema de generación de texto (chatbot) que utilice la técnica de **Retrieval-Augmented Generation (RAG)**. En este ejercicio, el chatbot será capaz de recuperar información de una base de datos (o un conjunto de documentos) y usarla para generar respuestas más completas, mejorando la calidad de las respuestas generadas.
- **Entregables:** Link a repo público y captura de video de chatbot consultando al CV del alumno. Utilizar librería como Streamlit (para capturar pantalla se puede utilizar [OBS](#) por ejemplo, es open source). Link entregar: <https://forms.gle/roAtRx2rXawaftjm7>

## Ejercicio en clase :

### Pasos

1. Preparación del entorno de trabajo: contar con IDE, cuenta de Pinecone (Starter), cuenta de Groq.
2. Cargar los CVs de los miembros del equipo y obtener los vectores de embeddings ([ejemplo de modelo de embeddings](#)).
3. Cargar los vectores a Pinecone.
4. Probar hacer una pregunta y, por medio de una comparación coseno, obtener el vector más cercano.
5. Implementar un simple chatbot para obtener respuestas sobre el documento cargado (CV del alumno).