**Vector Database:**

When I search in google: 1. Calories in apple 2. Employees in apple

it figures out which apple I am talking about, is it product or the fruit

How it is done by google? Semantic search.

Not searching with exact keyword matching., rather using the context and intent of search

To do semantic search, google use word embeddings. (refer original doc)



Features or properties: here, related\_to \_phones, calorie, is\_fruit are the vectors. For each word which is being searched these features will be assigned with values.

Gets sequence of number which is nothing but vector.

In example: apple vs Orange: these has the values in is\_fruit and calories features where as apple(phone) and Samsung has the value 0 against these features.

Consider, apple vs Samsung. Both has different value assigned in revenue, these are not similar values. But when I compare this feature in apple(fruit) and orange, which has 0. It means Apple(company) and Samsung are related somehow, but apple(company) and apple(fruit) is not related, though the keyword is same.

It helps in finding similarities in word.

Storing:

Traditional DB:

Consider 2 articles: Apple company and Apple fruit. Generate embeddings. Store it in traditional DB

When I search in google: Calories in Apple: even here it generates the emberddngg for my search query.

These embeddings will be compared with articles embeddings in the DB using cosine similarity. Then it retrives the matching records.

Cosine similarity uses the linear search of embeddings vectors and compare.

This is not possible in real world data as we have so much of data and this serial search cant be used.

When DB has millions of records, which google has.

DB index: hashing function: hashing function creates buckets which has similar embeddings. This is called Locality sensitive hashing. (Type of Vector DB)

Advantages:

Fast Search

Optimal Storage

VSCode:

1. Given PDF of any size, load this PDF.
2. Convert this pdf into chunks as openai or huggingface has limited size. Text chunks
3. Using OpenAI embeddings to convert the text chunks into vectors. (numeric form for each word)
4. All these vectors are sent to vector search DB. Why? When user queries any input. Bot can apply similarity search to vector DB.

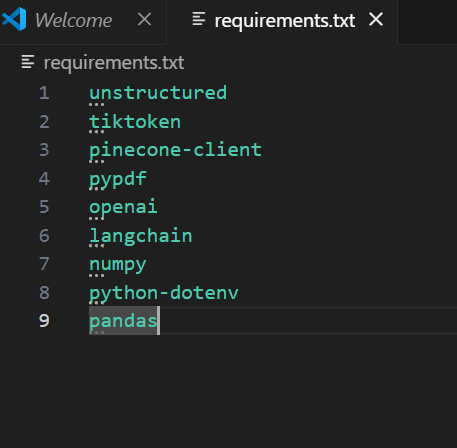
This Vector Search DB I am using in this called Pinecone.

This base can be used for text summarisation, Q&A

1. Create new environment
2. Activate environment



1. Write requirements



1. Install requirements



Pinecone:

Pinecone provides long-term memory for high-performance AI applications. It’s a managed, cloud-native [vector database](https://www.pinecone.io/learn/vector-database/) with a streamlined [API](https://docs.pinecone.io/reference/api) and no infrastructure hassles. Pinecone serves fresh, relevant query results with low latency at the scale of billions of vectors.

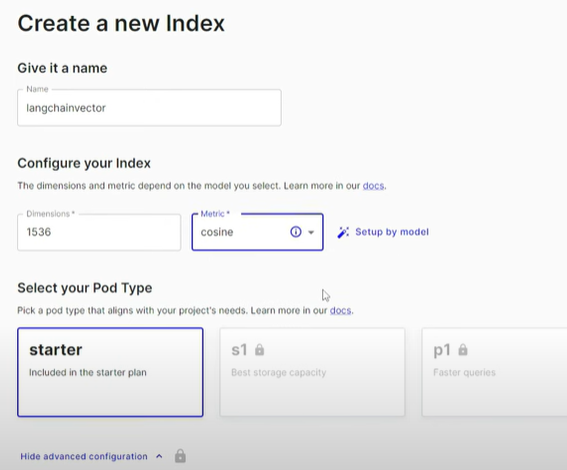
Create an account in pinecone

Login

API keys

Create new index: provide name,

Length was 1536 in my program., so mention dimesions as 1536. Use cosine similarity



Go back to indexes page

From this fetch 2 details

index name and environment (copy this)

Go to APi keys and create one: need to pass this as an argument for pinecone.init()