In [1]:
```
# Copyright 2023 Google LLC
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     https://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
```

Getting Started with Text Embeddings + Vertex AI Vector Search

Run in Colab View on GitHub Open in Vertex AI Workbench

Author(s) Smitha Venkat, Kaz Sato

Introduction

In this tutorial, you learn how to use Google Cloud AI tools to quickly bring the power of Large Language Models to enterprise systems.

This tutorial covers the following -

- What are embeddings - what business challenges do they help solve ?
- Understanding Text with Vertex AI Text Embeddings
- Find Embeddings fast with Vertex AI Vector Search
- Grounding LLM outputs with Vector Search

This tutorial is based on the blog post, combined with sample code.

Prerequisites

This tutorial is designed for developers who has basic knowledge and experience with Python programming and machine learning.

If you are not reading this tutorial in Qwiklab, then you need to have a Google Cloud project that is linked to a billing account to run this. Please go through this document to

create a project and setup a billing account for it.

Choose the runtime environment

The notebook can be run on either Google Colab or Vertex AI Workbench.

- To use Colab: Click this link to open the tutorial in Colab.
- To use Workbench: If it is the first time to use Workbench in your Google Cloud project, open the Workbench console and click ENABLE button to enable Notebooks API. Then click this link, and select an existing notebook or create a new notebook.

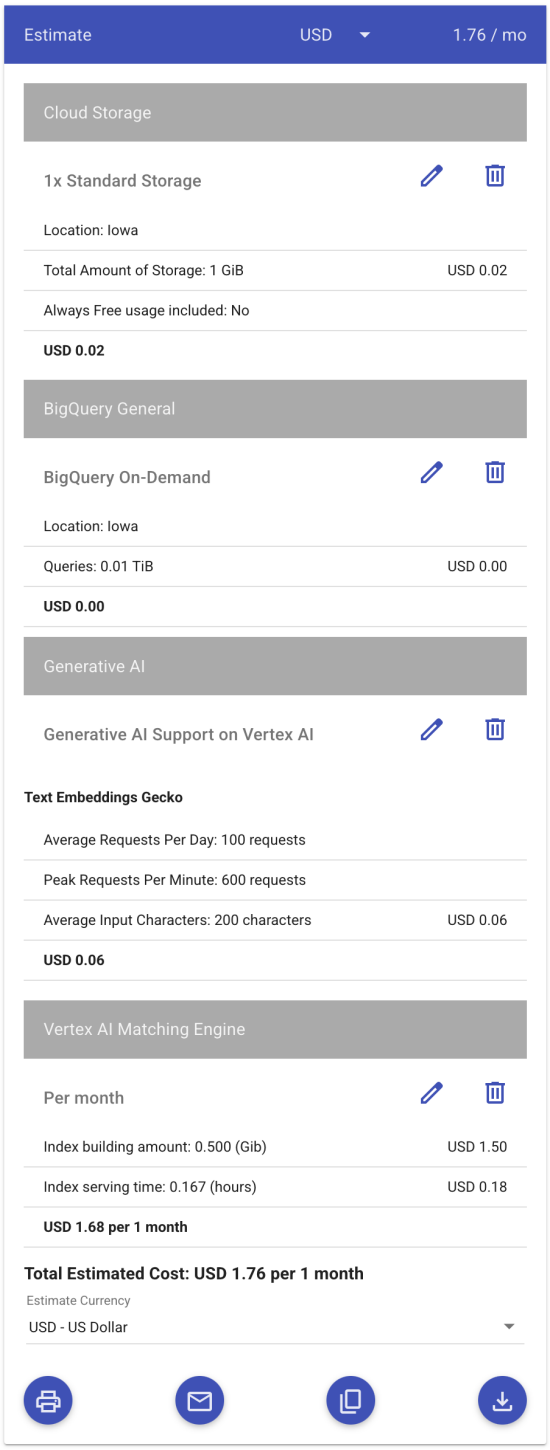
How much will this cost?

In case you are using your own Cloud project, not a temporary project on Qwiklab, you need to spend roughly a few US dollars to finish this tutorial.

The pricing of the Cloud services we will use in this tutorial are available in the following pages:

- Vertex AI Embeddings for Text
- Vertex AI Vector Search
- BigQuery
- Cloud Storage
- Vertex AI Workbench if you use one

You can use the Pricing Calculator to generate a cost estimate based on your projected usage. The following is an example of rough cost estimation with the calculator, assuming you will go through this tutorial a couple of time.



Warning: delete your objects after the tutorial

In case you are using your own Cloud project, please make sure to delete all the Indexes, Index Endpoints and Cloud Storage buckets (and the Workbench instance if you use one) after finishing this tutorial. Otherwise the remaining assets would incur unexpected costs.

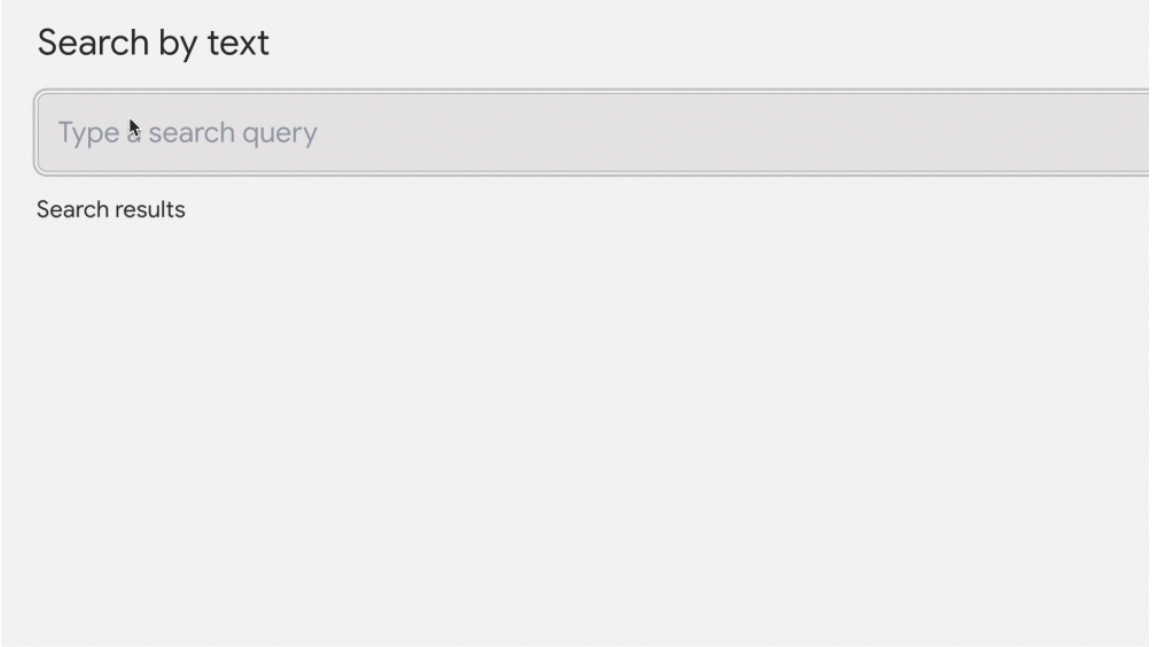
Bringing Gen AI and LLMs to production services

Many people are now starting to think about how to bring Gen AI and LLMs to production services, and facing with several challenges.

- "How to integrate LLMs or AI chatbots with existing IT systems, databases and business data?"
- "We have thousands of products. How can I let LLM memorize them all precisely?"
- "How to handle the hallucination issues in AI chatbots to build a reliable service?"

Here is a quick solution: **grounding** with **embeddings** and **vector search**.

What is grounding? What are embedding and vector search? In this tutorial, we will learn these crucial concepts to build reliable Gen AI services for enterprise use. But before we dive deeper, let's try the demo below.



Exercise: Try the Stack Overflow semantic search demo:

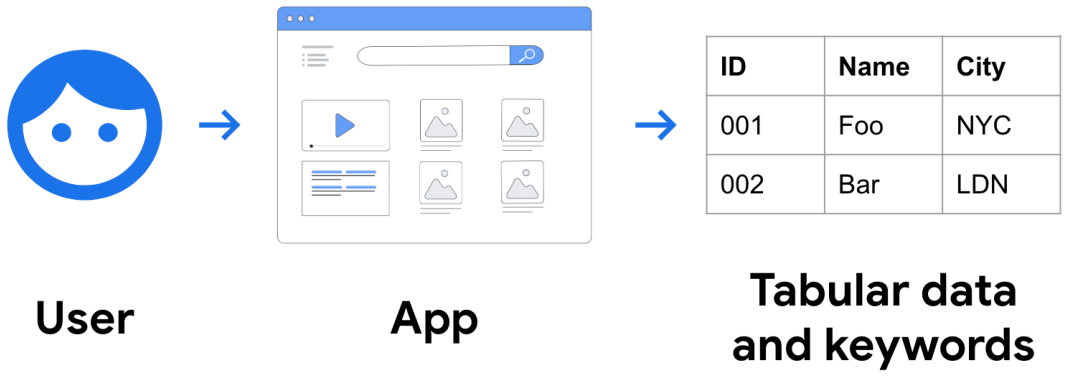
This demo is available as a public live demo. Select "STACKOVERFLOW" and enter any coding question as a query, so it runs a text search on **8 million** questions posted on Stack Overflow. Try the text semantic search with some queries like 'How to shuffle rows in SQL?' or arbitrary programming questions.

In this tutorial, we are going to see how to build a similar search experience - what is involved in building solutions like this using Vertex AI Embeddings API and Vector Search.

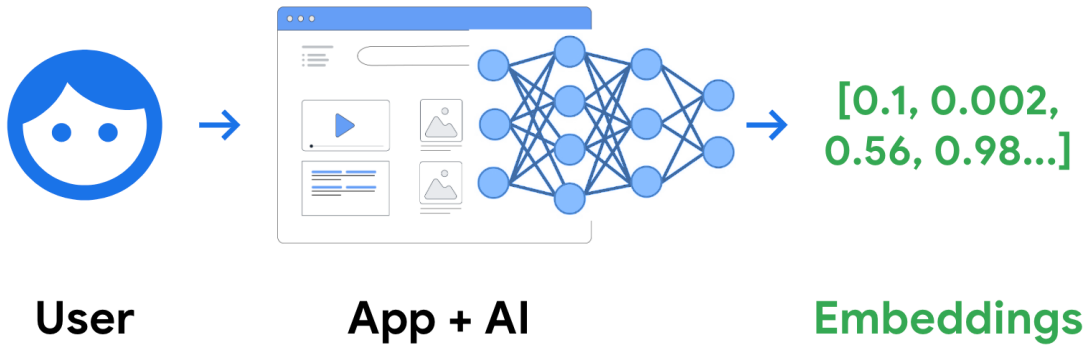
What is Embeddings?

With the rise of LLMs, why is it becoming important for IT engineers and ITDMs to understand how they work?

In traditional IT systems, most data is organized as structured or tabular data, using simple keywords, labels, and categories in databases and search engines.

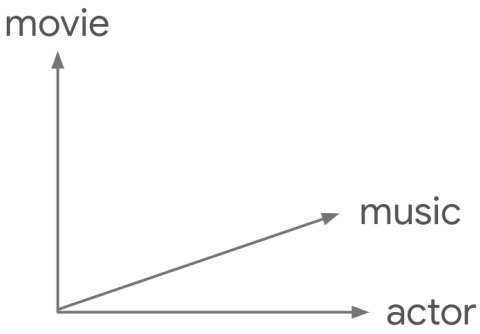


In contrast, AI-powered services arrange data into a simple data structure known as "embeddings."



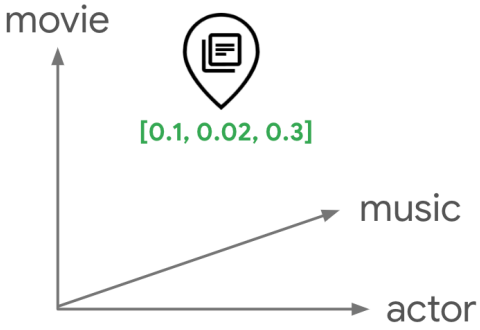
Once trained with specific content like text, images, or any content, AI creates a space called "embedding space", which is essentially a map of the content's meaning.

AI builds an embedding space as a map of meaning



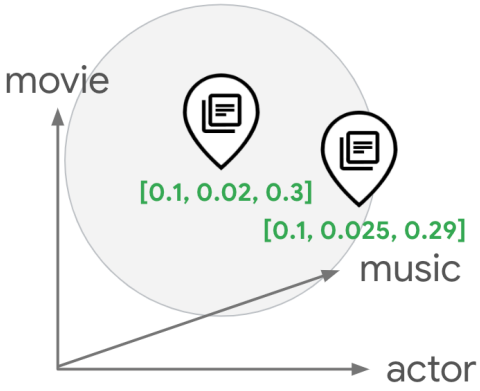
AI can identify the location of each content on the map, that's what embedding is.

AI finds an embedding from content, pointing a location in the map of meaning



Let's take an example where a text discusses movies, music, and actors, with a distribution of 10%, 2%, and 30%, respectively. In this case, the AI can create an embedding with three values: 0.1, 0.02, and 0.3, in 3 dimensional space.

AI puts contents with similar meaning close together in the space



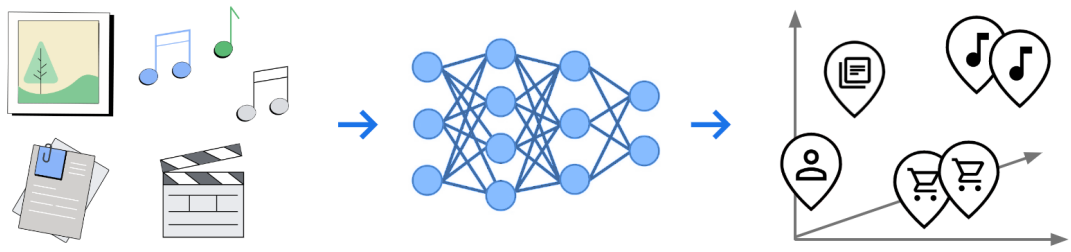
AI can put content with similar meanings closely together in the space.

This is how Google organizes data across various services like Google Search, YouTube, Play, and many others, to provide search results and recommendations with relevant content.

Embeddings can also be used to represent different types of things in businesses, such as products, users, user activities, conversations, music & videos, signals from IoT sensors, and so on.

AI and Embeddings are now playing a crucial role in creating a new way of human-computer interaction.

Embeddings enables the new Human Computer Interaction Realizing the next-gen user experience



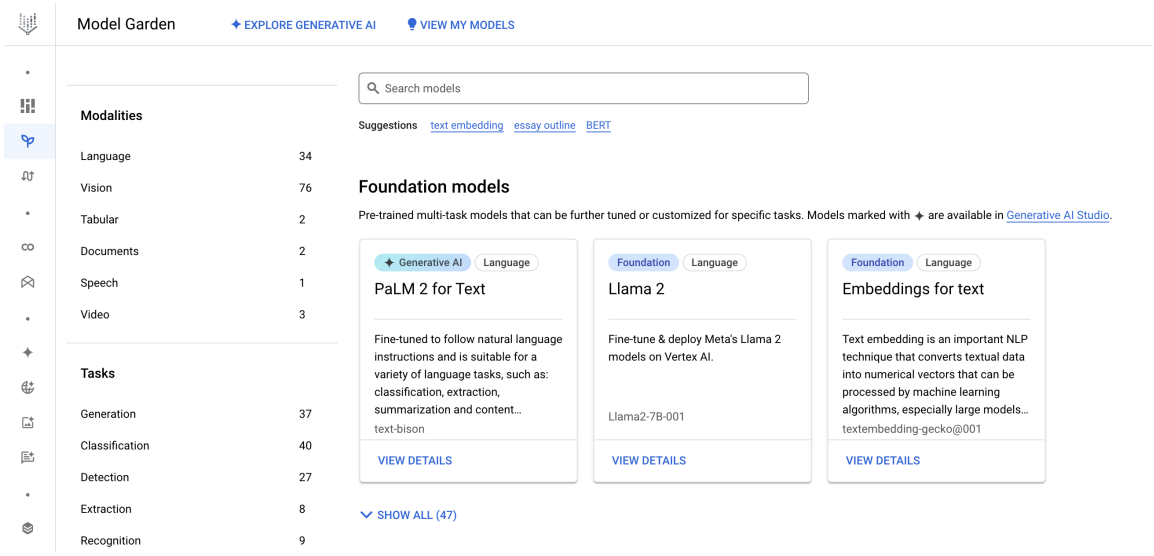
AI organizes data into embeddings, which represent what the user is looking for, the meaning of contents, or many other things you have in your business. This creates a

new level of user experience that is becoming the new standard.

To learn more about embeddings, Foundational courses: Embeddings on Google Machine Learning Crush Course and Meet AI's multitool: Vector embeddings by Dale Markowitz are great materials.

Vertex AI Embeddings for Text

With the Vertex AI Embeddings for Text, you can easily create a text embedding with LLM. The product is also available on Vertex AI Model Garden



This API is designed to extract embeddings from texts. It can take text input up to 2048 input tokens, and outputs 768 dimensional text embeddings.

LLM text embedding business use cases

With the embedding API, you can apply the innovation of embeddings, combined with the LLM capability, to various text processing tasks, such as:

LLM-enabled Semantic Search: text embeddings can be used to represent both the meaning and intent of a user's query and documents in the embedding space. Documents that have similar meaning to the user's query intent will be found fast with vector search technology. The model is capable of generating text embeddings that capture the subtle nuances of each sentence and paragraphs in the document.

LLM-enabled Text Classification: LLM text embeddings can be used for text classification with a deep understanding of different contexts without any training or fine-tuning (so-called zero-shot learning). This wasn't possible with the past language models without task-specific training.

LLM-enabled Recommendation: The text embedding can be used for recommendation systems as a strong feature for training recommendation models such as Two-Tower model. The model learns the relationship between the query and candidate embeddings, resulting in next-gen user experience with semantic product recommendation.

LLM-enabled Clustering, Anomaly Detection, Sentiment Analysis, and more, can be also handled with the LLM-level deep semantics understanding.

Sorting 8 million texts at "librarian-level" precision

Vertex AI Embeddings for Text has an embedding space with 768 dimensions. As explained earlier, the space represents a huge map of a wide variety of texts in the world, organized by their meanings. With each input text, the model can find a location (embedding) in the map.

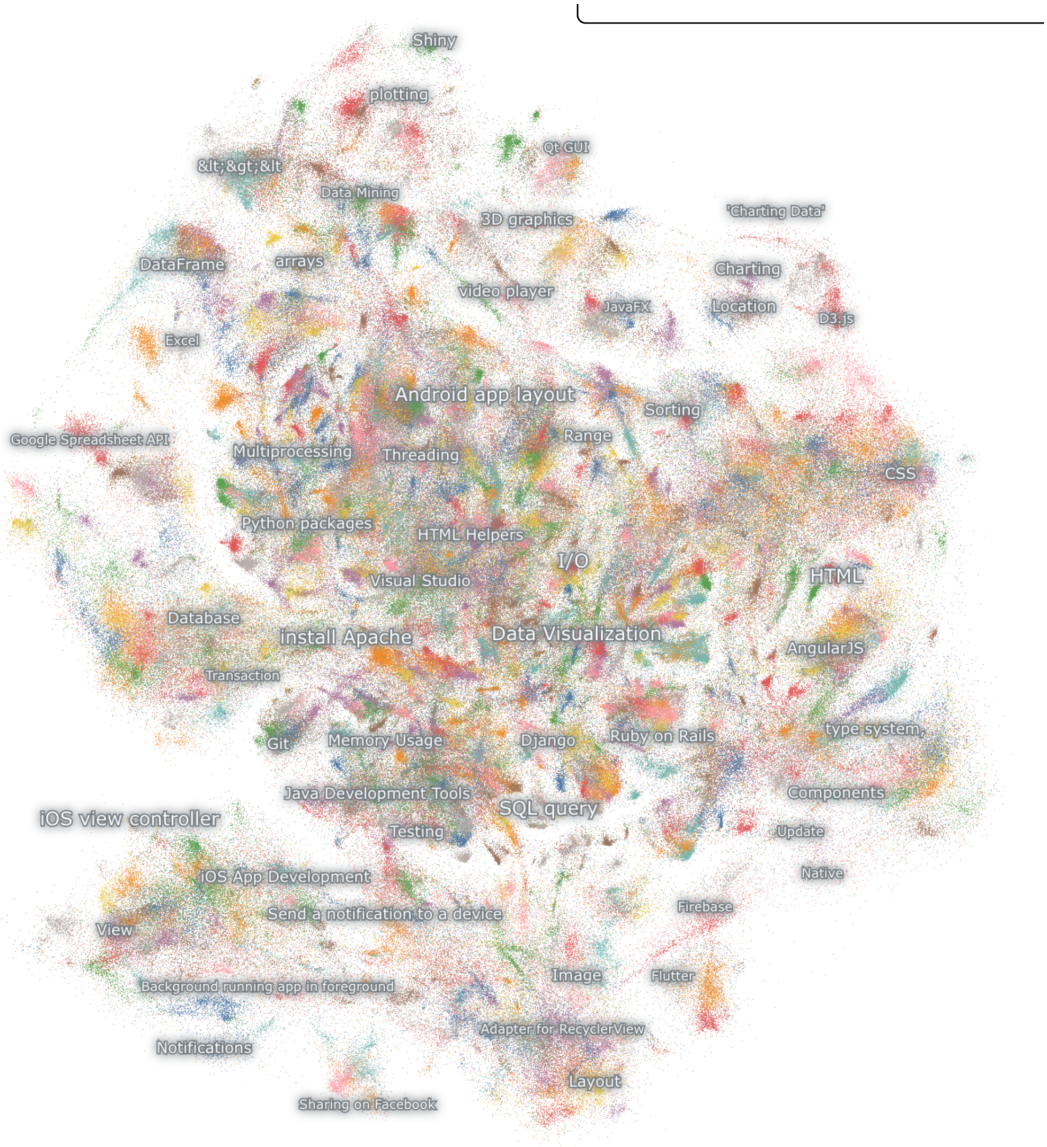
By visualizing the embedding space, you can actually observe how the model sorts the texts at the "librarian-level" precision.

Exercise: Try the Nomic AI Atlas

Nomic AI provides a platform called Atlas for storing, visualizing and interacting with embedding spaces with high scalability and in a smooth UI, and they worked with Google for visualizing the embedding space of the 8 million Stack Overflow questions. You can try exploring around the space, zooming in and out to each data point on your browser on this page, courtesy of Nomic AI.

The embedding space represents a huge map of texts, organized by their meanings With each input text, the model can find a location (embedding) in the map Like a librarian reading through millions of texts, sorting them with millions of nano-categories

Try exploring it here. Zoom into a few categories, point each dots, and see how the LLM is sorting similar questions close together in the space.



The librarian-level semantic understanding

Here are the examples of the librarian-level semantic understanding by Embeddings API with Stack Overflow questions.

Key questions	Questions close in the embedding space	What's the point?
How do I write a class that instantiates only once	How to create a singleton class based on its input?	The model knows "instantiating only once" and "singleton" mean the same.
Does moving the request line to a header frame require an app change?	Does an application developed on HTTP/1.x require modifications to run on HTTP/2?	The model knows both questions talk about what's the change required to support the HTTP/2 header frame.
How to increase IO speed with TensorFlow?	Tensorflow GPU/CPU Performance Suddenly Input Bound	The model knows both questions intent to improve TensorFlow IO performance

`for num in range(1,101):`
`  string = ""`
`  if num % 3 == 0:`
`  string = string + "A"`
`  if num % 4 == 0:`
`  string = string + "B"`
`  if num % 4 != 0 and num % 3 != 0:`
`  string = string + str(num)`
`  print(string)`
	Write a program that prints the integers from 1 to 100 (inclusive) in one line of code
	I am new to python and would like to write a program that prints the integers from 1 to 100 (inclusive) in 1 line using python:
	`for i in xrange(1, 101):`
	`  if i % 15 == 0:`
	`  print "shellfish"`
	`  elif i % 3 == 0:`
	`  print "shell"`
	`  elif i % 5 == 0:`
	`  print "fish"`
	`  else:`
	`  print i`
		The model thinks both Python code looks similar in what they are doing (printing fizzbuzz-like sequences), although they are not exactly the same.

For example, the model thinks the question "Does moving the request line to a header frame require an app change?" is similar to the question "Does an application developed on HTTP1x require modifications to run on HTTP2?". That is because The model knows both questions talk about what's the change required to support the HTTP2 header frame.

Note that this demo didn't require any training or fine-tuning with computer programming specific datasets. This is the innovative part of the zero-shot learning capability of the LLM. It can be applied to a wide variety of industries, including finance, healthcare, retail, manufacturing, construction, media, and more, for deep semantic search on the industry-focused business documents without spending time and cost for collecting industry specific datasets and training models.

Text Embeddings in Action

Lets try using Text Embeddings in action with actual sample code.

Setup

Before get started with the Vertex AI services, we need to setup the following.

- Install Python SDK
- Environment variables
- Authentication (Colab only)
- Enable APIs
- Set IAM permissions

Install Python SDK

Vertex AI, Cloud Storage and BigQuery APIs can be accessed with multiple ways including REST API and Python SDK. In this tutorial we will use the SDK.

In [2]:
```
%pip install --upgrade --user google-cloud-aiplatform google-cloud-storage '
```

```
Requirement already satisfied: google-cloud-aiplatform in /opt/conda/lib/pyt
hon3.10/site-packages (1.70.0)
Requirement already satisfied: google-cloud-storage in /opt/conda/lib/python
3.10/site-packages (2.14.0)
Collecting google-cloud-storage
  Downloading google_cloud_storage-2.18.2-py2.py3-none-any.whl.metadata (9.1
kB)
Requirement already satisfied: google-cloud-bigquery[pandas] in /opt/conda/l
ib/python3.10/site-packages (3.25.0)
Collecting google-cloud-bigquery[pandas]
  Downloading google_cloud_bigquery-3.26.0-py2.py3-none-any.whl.metadata
(8.7 kB)
Requirement already satisfied: google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=
2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1 in /opt/conda/lib/p
ython3.10/site-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=2.2.*,!
=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1->google-cloud-aipl
atform) (1.34.1)
Requirement already satisfied: google-auth<3.0.0dev,>=2.14.1 in /opt/conda/l
ib/python3.10/site-packages (from google-cloud-aiplatform) (2.35.0)
Requirement already satisfied: proto-plus<2.0.0dev,>=1.22.3 in /opt/conda/li
b/python3.10/site-packages (from google-cloud-aiplatform) (1.24.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!
=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.2 in /opt/conda/lib/python3.10/site-packag
es (from google-cloud-aiplatform) (3.20.3)
Requirement already satisfied: packaging>=14.3 in /opt/conda/lib/python3.10/
site-packages (from google-cloud-aiplatform) (24.1)
Requirement already satisfied: google-cloud-resource-manager<3.0.0dev,>=
1.3.3 in /opt/conda/lib/python3.10/site-packages (from google-cloud-aiplatfo
rm) (1.12.5)
Requirement already satisfied: shapely<3.0.0dev in /opt/conda/lib/python3.1
0/site-packages (from google-cloud-aiplatform) (2.0.6)
Requirement already satisfied: pydantic<3 in /opt/conda/lib/python3.10/site-
packages (from google-cloud-aiplatform) (2.9.2)
Requirement already satisfied: docstring-parser<1 in /opt/conda/lib/python
3.10/site-packages (from google-cloud-aiplatform) (0.16)
Collecting google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=
2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1 (from google-api-core[grpc]!=2.0.*,!=
2.1.*,!=2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1->go
ogle-cloud-aiplatform)
  Downloading google_api_core-2.21.0-py3-none-any.whl.metadata (2.8 kB)
Requirement already satisfied: google-cloud-core<3.0dev,>=2.3.0 in /opt/cond
a/lib/python3.10/site-packages (from google-cloud-storage) (2.4.1)
Requirement already satisfied: google-resumable-media>=2.7.2 in /opt/conda/l
ib/python3.10/site-packages (from google-cloud-storage) (2.7.2)
Requirement already satisfied: requests<3.0.0dev,>=2.18.0 in /opt/conda/lib/
python3.10/site-packages (from google-cloud-storage) (2.32.3)
Requirement already satisfied: google-crc32c<2.0dev,>=1.0 in /opt/conda/lib/
python3.10/site-packages (from google-cloud-storage) (1.6.0)
Requirement already satisfied: python-dateutil<3.0dev,>=2.7.3 in /opt/conda/
lib/python3.10/site-packages (from google-cloud-bigquery[pandas]) (2.9.0.pos
t0)
Requirement already satisfied: pandas>=1.1.0 in /opt/conda/lib/python3.10/si
te-packages (from google-cloud-bigquery[pandas]) (2.2.3)
Requirement already satisfied: pyarrow>=3.0.0 in /opt/conda/lib/python3.10/s
ite-packages (from google-cloud-bigquery[pandas]) (15.0.2)
Requirement already satisfied: db-dtypes<2.0.0dev,>=0.3.0 in /opt/conda/lib/
```

```
python3.10/site-packages (from google-cloud-bigquery[pandas]) (1.3.0)
Requirement already satisfied: numpy>=1.16.6 in /opt/conda/lib/python3.10/si
te-packages (from db-dtypes<2.0.0dev,>=0.3.0->google-cloud-bigquery[pandas])
(1.26.4)
Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in
/opt/conda/lib/python3.10/site-packages (from google-api-core!=2.0.*,!=
2.1.*,!=2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1->go
ogle-api-core[grpc]!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!
=2.7.*,<3.0.0dev,>=1.34.1->google-cloud-aiplatform) (1.65.0)
Requirement already satisfied: grpcio<2.0dev,>=1.33.2 in /opt/conda/lib/pyth
on3.10/site-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=2.2.*,!=
2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1->google-cloud-aipla
tform) (1.66.2)
Requirement already satisfied: grpcio-status<2.0.dev0,>=1.33.2 in /opt/cond
a/lib/python3.10/site-packages (from google-api-core[grpc]!=2.0.*,!=2.1.*,!=
2.2.*,!=2.3.*,!=2.4.*,!=2.5.*,!=2.6.*,!=2.7.*,<3.0.0dev,>=1.34.1->google-clo
ud-aiplatform) (1.48.2)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /opt/conda/lib/pyth
on3.10/site-packages (from google-auth<3.0.0dev,>=2.14.1->google-cloud-aipla
tform) (5.5.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in /opt/conda/lib/pytho
n3.10/site-packages (from google-auth<3.0.0dev,>=2.14.1->google-cloud-aiplat
form) (0.4.1)
Requirement already satisfied: rsa<5,>=3.1.4 in /opt/conda/lib/python3.10/si
te-packages (from google-auth<3.0.0dev,>=2.14.1->google-cloud-aiplatform)
(4.9)
Requirement already satisfied: grpc-google-iam-v1<1.0.0dev,>=0.12.4 in /opt/
conda/lib/python3.10/site-packages (from google-cloud-resource-manager<3.0.0
dev,>=1.3.3->google-cloud-aiplatform) (0.13.1)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/sit
e-packages (from pandas>=1.1.0->google-cloud-bigquery[pandas]) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.10/s
ite-packages (from pandas>=1.1.0->google-cloud-bigquery[pandas]) (2024.2)
Requirement already satisfied: annotated-types>=0.6.0 in /opt/conda/lib/pyth
on3.10/site-packages (from pydantic<3->google-cloud-aiplatform) (0.7.0)
Requirement already satisfied: pydantic-core==2.23.4 in /opt/conda/lib/pytho
n3.10/site-packages (from pydantic<3->google-cloud-aiplatform) (2.23.4)
Requirement already satisfied: typing-extensions>=4.6.1 in /opt/conda/lib/py
thon3.10/site-packages (from pydantic<3->google-cloud-aiplatform) (4.12.2)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-pa
ckages (from python-dateutil<3.0dev,>=2.7.3->google-cloud-bigquery[pandas])
(1.16.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/py
thon3.10/site-packages (from requests<3.0.0dev,>=2.18.0->google-cloud-storag
e) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/sit
e-packages (from requests<3.0.0dev,>=2.18.0->google-cloud-storage) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python
3.10/site-packages (from requests<3.0.0dev,>=2.18.0->google-cloud-storage)
(1.26.20)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python
3.10/site-packages (from requests<3.0.0dev,>=2.18.0->google-cloud-storage)
(2024.8.30)
Requirement already satisfied: pyasn1<0.7.0,>=0.4.6 in /opt/conda/lib/python
3.10/site-packages (from pyasn1-modules>=0.2.1->google-auth<3.0.0dev,>=2.1
4.1->google-cloud-aiplatform) (0.6.1)
```

```
Downloading google_cloud_storage-2.18.2-py2.py3-none-any.whl (130 kB)
Downloading google_api_core-2.21.0-py3-none-any.whl (156 kB)
Downloading google_cloud_bigquery-3.26.0-py2.py3-none-any.whl (239 kB)
Installing collected packages: google-api-core, google-cloud-storage, googl
e-cloud-bigquery
ERROR: pip's dependency resolver does not currently take into account all th
e packages that are installed. This behaviour is the source of the following
dependency conflicts.
google-api-python-client 1.8.0 requires google-api-core<2dev,>=1.13.0, but y
ou have google-api-core 2.21.0 which is incompatible.
Successfully installed google-api-core-2.21.0 google-cloud-bigquery-3.26.0 g
oogle-cloud-storage-2.18.2
Note: you may need to restart the kernel to use updated packages.
```

Restart current runtime

To use the newly installed packages in this Jupyter runtime, you must restart the runtime. You can do this by running the cell below, which will restart the current kernel.

In [3]:
```
# Restart kernel after installs so that your environment can access the new
import IPython

app = IPython.Application.instance()
app.kernel.do_shutdown(True)
```

Out[3]:
```
{'status': 'ok', 'restart': True}
```

⚠️ **The kernel is going to restart. Please wait until it is finished before continuing to the next step.** ⚠️

Environment variables

Sets environment variables. If asked, please replace the following `[your-project-id]` with your project ID and run it.

In [5]:
```
# get project ID
PROJECT_ID = ! gcloud config get project
PROJECT_ID = PROJECT_ID[0]
LOCATION = "us-central1"
if PROJECT_ID == "(unset)":
    print(f"Please set the project ID manually below")
```

In [6]:
```
# define project information
if PROJECT_ID == "(unset)":
    PROJECT_ID = "[your-project-id]"  # @param {type:"string"}

# generate an unique id for this session
from datetime import datetime

UID = datetime.now().strftime("%m%d%H%M")
```

Authentication (Colab only)

If you are running this notebook on Colab, you will need to run the following cell authentication. This step is not required if you are using Vertex AI Workbench as it is pre-authenticated.

In [7]:
```
import sys

# if it's Colab runtime, authenticate the user with Google Cloud
if "google.colab" in sys.modules:
    from google.colab import auth

    auth.authenticate_user()
```

Enable APIs

Run the following to enable APIs for Compute Engine, Vertex AI, Cloud Storage and BigQuery with this Google Cloud project.

In [1]:
```
! gcloud services enable compute.googleapis.com aiplatform.googleapis.com st
```

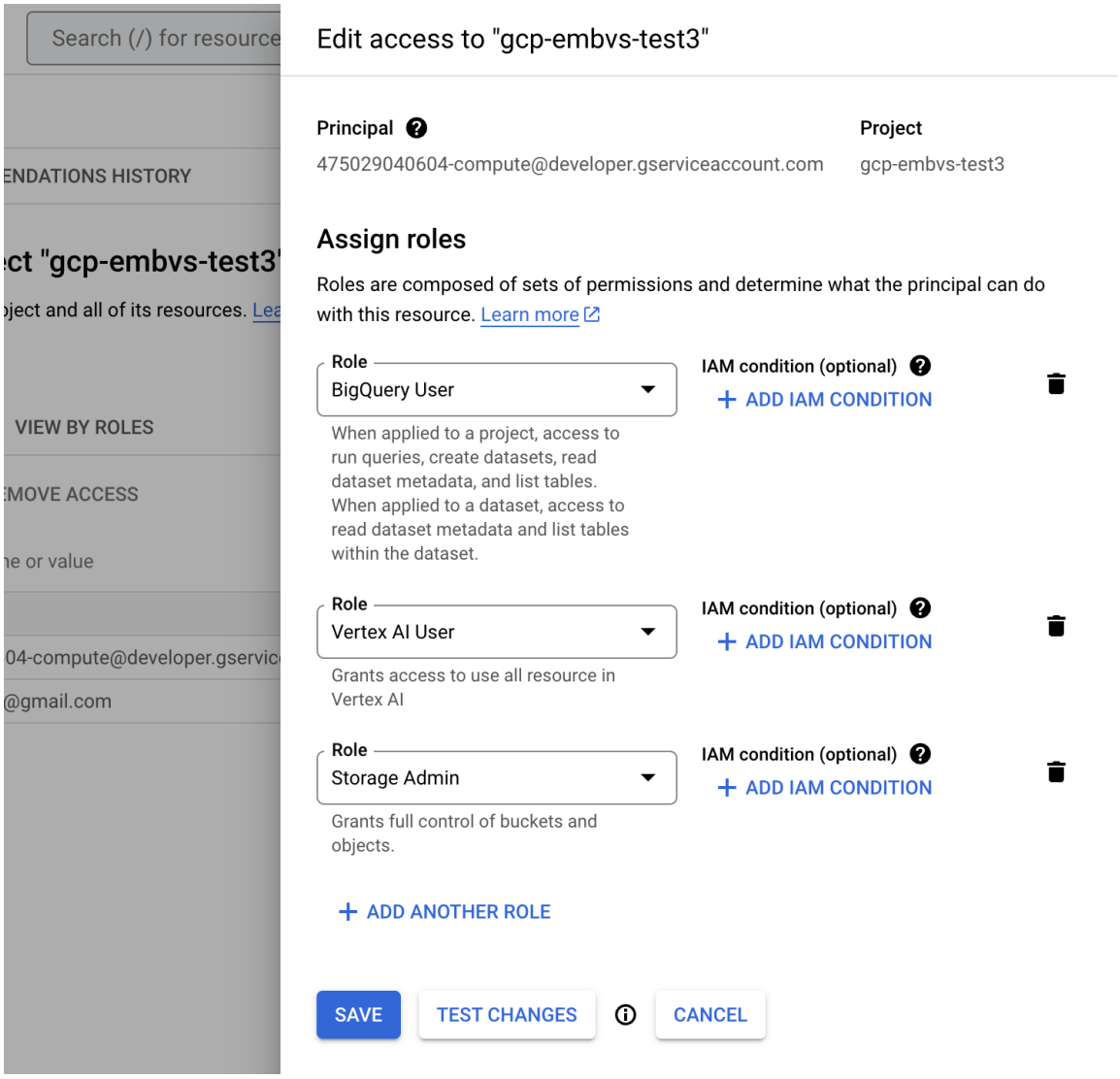
```
ERROR: (gcloud) The project property must be set to a valid project ID, [{PR
OJECT_ID}] is not a valid project ID.
To set your project, run:

  $ gcloud config set project PROJECT_ID

or to unset it, run:

  $ gcloud config unset project
```

Set IAM permissions

Also, we need to add access permissions to the default service account for using those services.

- Go to the IAM page in the Console
- Look for the principal for default compute service account. It should look like: `<project-number>-compute@developer.gserviceaccount.com`
- Click the edit button at right and click `ADD ANOTHER ROLE` to add `Vertex AI User`, `BigQuery User` and `Storage Admin` to the account.

This will look like this:



Getting Started with Vertex AI Embeddings for Text

Now it's ready to get started with embeddings!

Data Preparation

We will be using the Stack Overflow public dataset hosted on BigQuery table `bigquery-public-data.stackoverflow.posts_questions` . This is a very big dataset with 23 million rows that doesn't fit into the memory. We are going to limit it to 1000 rows for this tutorial.

In [8]:
```
# load the BQ Table into a Pandas DataFrame
from google.cloud import bigquery

QUESTIONS_SIZE = 1000

bq_client = bigquery.Client(project=PROJECT_ID)
QUERY_TEMPLATE = """
```

```
        SELECT distinct q.id, q.title
        FROM (SELECT * FROM `bigquery-public-data.stackoverflow.posts_questi
        where Score > 0 ORDER BY View_Count desc) AS q
        LIMIT {limit} ;
        """
query = QUERY_TEMPLATE.format(limit=QUESTIONS_SIZE)
query_job = bq_client.query(query)
rows = query_job.result()
df = rows.to_dataframe()

# examine the data
df.head()
```

Out[8]:

	id	title
0	73237542	How to get random single result from multiple ...
1	73398579	How should I increase the test coverage for th...
2	73327915	How do you add diagnostic information to iOS c...
3	73515603	[iOS/Safari15.5]border displayed in <video>tag
4	73265831	Compare two strings in GridDB FDW

Call the API to generate embeddings

With the Stack Overflow dataset, we will use the `title` column (the question title) and generate embedding for it with Embeddings for Text API. The API is available under the vertexai package of the SDK.

You may see some warning messages from the TensorFlow library but you can ignore them.

In [9]:
```
# init the vertexai package
import vertexai

vertexai.init(project=PROJECT_ID, location=LOCATION)
```

From the package, import TextEmbeddingModel and get a model.

In [13]:
```
# Load the text embeddings model
from vertexai.language_models import TextEmbeddingModel

model = TextEmbeddingModel.from_pretrained("textembedding-gecko@003")
```

In this tutorial we will use `textembedding-gecko@001` model for getting text embeddings. Please take a look at Supported models on the doc to see the list of supported models.

Once you get the model, you can call its get_embeddings function to get embeddings. You can pass up to 5 texts at once in a call. But there is a caveat. By default, the text

embeddings API has a "request per minute" quota set to 60 for new Cloud projects and 600 for projects with usage history (see Quotas and limits to check the latest quota value for `base_model:textembedding-gecko`). So, rather than using the function directly, you may want to define a wrapper like below to limit under 10 calls per second, and pass 5 texts each time.

In [14]:
```
import time

import tqdm  # to show a progress bar

# get embeddings for a list of texts
BATCH_SIZE = 5


def get_embeddings_wrapper(texts):
    embs = []
    for i in tqdm.tqdm(range(0, len(texts), BATCH_SIZE)):
        time.sleep(1)  # to avoid the quota error
        result = model.get_embeddings(texts[i : i + BATCH_SIZE])
        embs = embs + [e.values for e in result]
    return embs
```

The following code will get embedding for the question titles and add them as a new column `embedding` to the DataFrame. This will take a few minutes.

In [15]:
```
# get embeddings for the question titles and add them as "embedding" column
df = df.assign(embedding=get_embeddings_wrapper(list(df.title)))
df.head()
```

```
100%|██████████| 200/200 [03:43<00:00,  1.12s/it]
```

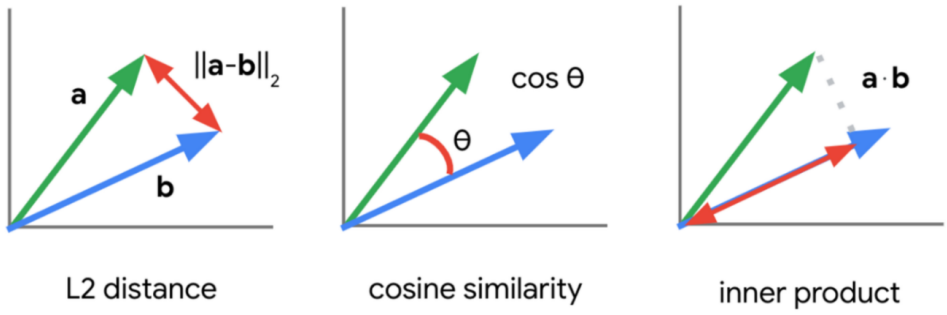
Out[15]:

	id	title	embedding
0	73237542	How to get random single result from multiple ...	[0.04007541015744209, -0.07926855236291885, -0...
1	73398579	How should I increase the test coverage for th...	[0.03479480743408203, -0.05112965777516365, -0...
2	73327915	How do you add diagnostic information to iOS c...	[0.07541406899690628, -0.0344245545566082, -0....
3	73515603	[iOS/Safari15.5]border displayed in <video>tag	[0.04759598150849342, -0.02628263086080551, -0...
4	73265831	Compare two strings in GridDB FDW	[-0.003980264533311129, 0.004829115699976683, ...

Look at the embedding similarities

Let's see how these embeddings are organized in the embedding space with their meanings by quickly calculating the similarities between them and sorting them.

As embeddings are vectors, you can calculate similarity between two embeddings by using one of the popular metrics like the followings:



Which metric should we use? Usually it depends on how each model is trained. In case of the model `textembedding-gecko@001`, we need to use inner product (dot product).

In the following code, it picks up one question randomly and uses the numpy `np.dot` function to calculate the similarities between the question and other questions.

In [16]:
```
import random

import numpy as np

# pick one of them as a key question
key = random.randint(0, len(df))

# calc dot product between the key and other questions
embs = np.array(df.embedding.to_list())
similarities = np.dot(embs[key], embs.T)

# print similarities for the first 5 questions
similarities[:5]
```

Out[16]:
```
array([0.66606262, 0.59874936, 0.58412744, 0.59255396, 0.66046554])
```

Finally, sort the questions with the similarities and print the list.

In [17]:
```
# print the question
print(f"Key question: {df.title[key]}\n")

# sort and print the questions by similarities
sorted_questions = sorted(
    zip(df.title, similarities), key=lambda x: x[1], reverse=True
)[:20]
for i, (question, similarity) in enumerate(sorted_questions):
    print(f"{similarity:.4f} {question}")
```

```
Key question: How to merge values in a dictionary depending on range

1.0000 How to merge values in a dictionary depending on range
0.7754 How to merge list of strings into list of lists?
0.7679 Merge to various type List in to new List Type
0.7588 make a join of all the data frames inside a list in R
0.7535 How to print the duplicates value while converting from Dictionary to
List in c#
0.7522 Creating specific Json with python dictionaries
0.7522 Split a vector into non-overlapping sub-list with increasing length
0.7517 How to convert nested dictionaries to dataframes in the below format
in python
0.7509 How to split list values in column into rows?
0.7506 Convert a list of "dictionary of dictionaries" to a dataframe
0.7496 Change structure of python dictionary using recursion
0.7412 Join dataframes and rename resulting columns with same names
0.7411 how to get value from the List of Map?
0.7399 How can I replace array item with matched object keys?
0.7391 How to append one dataframe into another dataframe as a column
0.7389 KQL - Remove Duplicates From A List And Sum Values
0.7383 Filter based on row repetition of a range of values
0.7368 find sum and minimum of nested list
0.7351 Create a dataframe with all combinations of two columns
0.7350 Add key-value pair to array only if variable is set
```

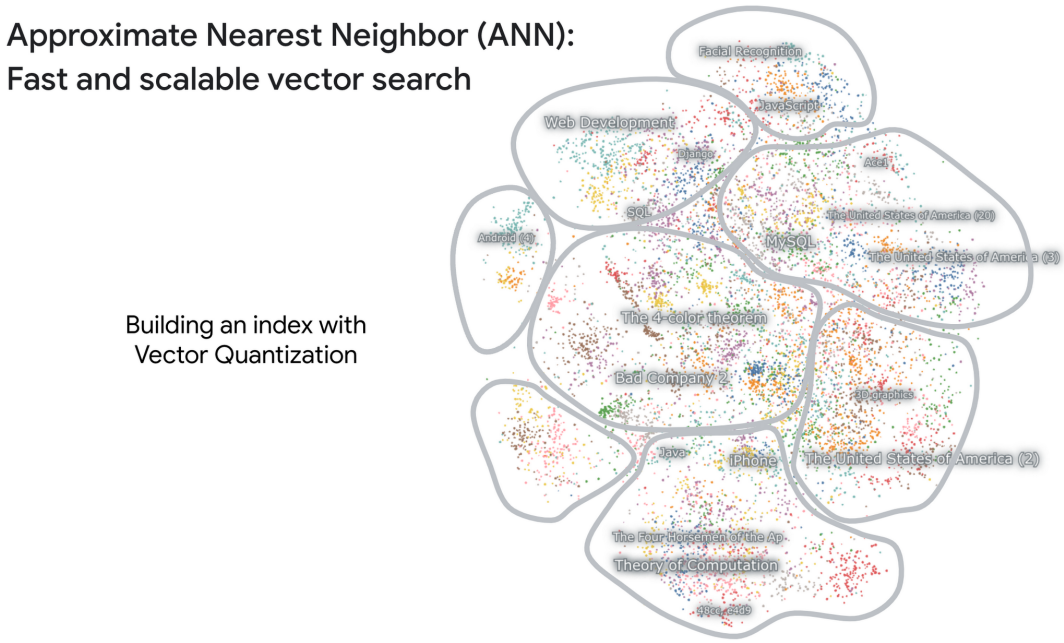
Find embeddings fast with Vertex AI Vector Search

As we have explained above, you can find similar embeddings by calculating the distance or similarity between the embeddings.

But this isn't easy when you have millions or billions of embeddings. For example, if you have 1 million embeddings with 768 dimensions, you need to repeat the distance calculations for 1 million x 768 times. This would take some seconds - too slow.

So the researchers have been studying a technique called Approximate Nearest Neighbor (ANN) for faster search. ANN uses "vector quantization" for separating the space into multiple spaces with a tree structure. This is similar to the index in relational databases for improving the query performance, enabling very fast and scalable search with billions of embeddings.

With the rise of LLMs, the ANN is getting popular quite rapidly, known as the Vector Search technology.



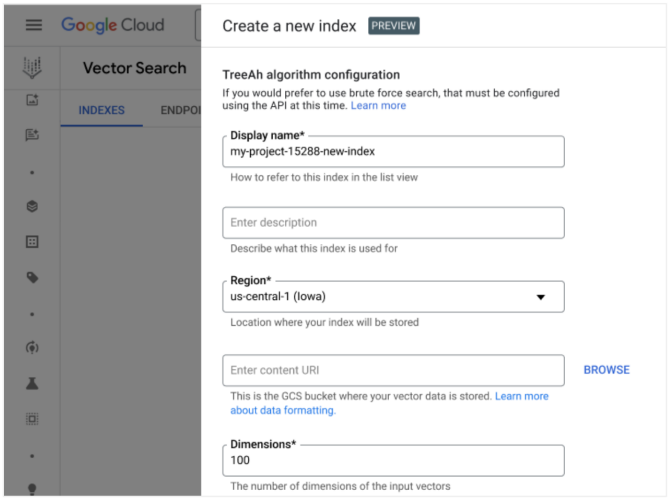
In 2020, Google Research published a new ANN algorithm called ScaNN. It is considered one of the best ANN algorithms in the industry, also the most important foundation for search and recommendation in major Google services such as Google Search, YouTube and many others.

What is Vertex AI Vector Search?

Google Cloud developers can take the full advantage of Google's vector search technology with Vertex AI Vector Search (previously called Matching Engine). With this fully managed service, developers can just add the embeddings to its index and issue a search query with a key embedding for the blazingly fast vector search. In the case of the Stack Overflow demo, Vector Search can find relevant questions from 8 million embeddings in tens of milliseconds.

Introducing
Vertex AI Vector Search:

Fully managed,
Google-scale
vector search engine
integrated with Vertex AI



With Vector Search, you don't need to spend much time and money building your own vector search service from scratch or using open source tools if your goal is high scalability, availability and maintainability for production systems.

Get Started with Vector Search

When you already have the embeddings, then getting started with Vector Search is pretty easy. In this section, we will follow the steps below.

Setting up Vector Search

- Save the embeddings in JSON files on Cloud Storage
- Build an Index
- Create an Index Endpoint
- Deploy the Index to the endpoint

Use Vector Search

- Query with the endpoint

Tip for Colab users

If you use Colab for this tutorial, you may lose your runtime while you are waiting for the Index building and deployment in the later sections as it takes tens of minutes. In that case, run the following sections again with the new instance to recover the runtime: Install Python SDK, Environment variables and Authentication.

Then, use the Utilities to recover the Index and Index Endpoint and continute with the rest.

Save the embeddings in a JSON file

To load the embeddings to Vector Search, we need to save them in JSON files with JSONL format. See more information in the docs at Input data format and structure.

First, export the `id` and `embedding` columns from the DataFrame in JSONL format, and save it.

In [18]:
```
# save id and embedding as a json file
jsonl_string = df[["id", "embedding"]].to_json(orient="records", lines=True)
with open("questions.json", "w") as f:
    f.write(jsonl_string)

# show the first few lines of the json file
! head -n 3 questions.json
```

{"id":73237542,"embedding":[0.0400754102,-0.0792685524,-0.0406071059,-0.0077
255806,0.0159847997,0.0246462859,0.0444029383,-0.0406927206,0.0108482894,0.0
751760006,0.015935095,0.005330848,-0.0231548175,-0.0341294259,0.000668057
2,-0.048980806,0.0198701136,0.0380846187,-0.0441499092,-0.0395904183,-0.0407
501236,0.0258147027,-0.0100368578,-0.0171885714,0.0150421653,0.012089705
1,0.0116401436,-0.0052855848,-0.004689577,0.0366029292,-0.0116050635,0.07253
99628,-0.0652003363,0.0231859572,-0.0238913726,-0.0196763184,0.008009445
9,0.0047453148,0.0057714786,0.037778493,-0.0084394757,-0.0081395693,0.015079
7321,-0.011390768,-0.0296219923,-0.0346656628,0.0024876781,0.0306280442,0.04
82395627,-0.0681847781,0.0089706481,0.0002384574,0.022981843,-0.07360070
2,-0.0059870076,-0.0555741638,-0.0295187924,0.0688899606,0.02492477,-0.02047
88614,-0.0095735146,0.0287025794,-0.0445167758,0.0316443667,-0.04998556
9,-0.030367475,-0.0050931796,0.0028927275,0.0562823005,0.0145156616,0.017597
0085,-0.0480866916,0.0936769098,-0.0002646345,-0.0768485889,-0.107579313
2,-0.0006867195,0.063502863,0.0221275296,0.0038091803,0.0245299675,-0.031666
3384,0.0125704156,-0.0055810381,-0.031292934,0.0522834398,-0.0107256146,0.00
52597634,-0.0027590059,0.0495103225,-0.0669060946,0.0020752454,0.003522073
9,-0.1068618223,0.0051637297,0.0709104314,-0.0211960338,-0.0032453879,0.0584
834218,0.008277989,0.014576369,-0.0158416517,-0.0355541445,0.0351953171,-0.0
194459446,0.0135626905,0.0195816215,0.0318109319,-0.0434262827,0.032805491
2,-0.0681191906,-0.0234332923,-0.038834095,-0.0171136782,0.0053354101,-0.044
091031,-0.0335241109,0.0403128937,0.018906638,0.0167272184,0.0657724664,0.07
34482259,0.0142334448,0.0088174706,-0.0150560699,0.042400986,-0.002394736
5,0.0073080454,0.050928954,0.0233421531,0.0189367011,-0.0054887789,0.0236266
032,0.030198846,0.0470846184,0.0376375355,0.0217108335,-0.0013098727,0.06090
05839,-0.0257540066,-0.0261804834,0.0245301835,-0.0104396185,0.043782923
4,-0.0021551675,0.0003718173,-0.0309753865,-0.0398573987,0.041588597,-0.0666
487142,-0.0089984173,0.0140253259,-0.0791328773,0.0124378512,0.032951977
1,0.000122587,-0.0482985042,0.0394497737,0.0104522733,0.0315688886,0.0613368
675,0.024741482,-0.0092539592,0.0422574803,-0.0355173275,0.0353547782,-0.065
8180416,0.0075247041,0.0024661596,-0.0233750325,0.0374295898,0.000412830
9,-0.0844341069,-0.0305695012,-0.0188626703,-0.0603383183,-0.0259377994,0.02
925626,0.0231859367,-0.026367737,0.0464493148,-0.0051136278,0.011911993
8,-0.0017478735,0.0453581698,-0.0159858018,0.0519455485,-0.0390562303,-0.033
2224183,-0.0000553847,0.00397242,-0.0021320346,-0.0033180849,0.002574797
5,0.0112235695,0.0071421452,0.0087148855,0.0249811709,0.0216662716,-0.040675
2937,-0.0102528539,0.0881056786,0.0147247994,-0.0092128506,0.0291780774,0.01
32854022,0.0240898523,-0.0602182858,-0.0237508994,0.0273713991,-0.005275642
5,0.0356232934,-0.0255496316,0.0170878042,0.0697567537,0.0164529737,0.033680
5731,0.0362959653,-0.001427464,-0.0387824215,-0.0232366696,0.0056378776,-0.0
235377643,0.0635907724,-0.0056017875,0.0386815742,0.0283262003,0.052283376
5,-0.0704227164,-0.0509318933,0.0395056121,0.0840069726,0.0639462471,-0.0362
685807,0.0338575467,0.0158253442,0.0247254502,-0.0104413321,0.001388790
8,-0.0090318192,-0.0191077981,-0.0033587175,0.0369602665,0.0413273685,-0.086
4023939,-0.0139877927,-0.0039916653,0.0026219448,-0.0004129765,0.048240695
1,-0.0063017616,-0.0005938114,0.0302601047,0.015771566,-0.0164038856,-0.0421
087146,-0.0603369176,0.0034957964,0.0137823345,0.0364098549,0.045124653
7,-0.0500077717,-0.0003850018,-0.0385361537,-0.0274492223,0.0008529177,0.013
2863661,-0.0938281491,-0.0113576818,0.0317397304,-0.0284685176,-0.037463113
7,0.0406914502,0.0115293097,-0.0103568053,0.0147358058,-0.0124567309,0.04053
8393,0.0529112369,-0.060818281,-0.0035232226,-0.0020722561,-0.005592626
1,-0.0880344138,-0.0243614502,0.0100119058,-0.0080812024,-0.0108016366,0.003
0900063,-0.035043221,-0.0139714489,-0.0136228362,-0.023408331,-0.048826046
3,-0.0295447074,-0.0004311003,-0.0532098338,0.0742487162,0.0498934984,-0.001
8313221,-0.017603064,-0.0141878147,-0.0225986447,-0.0537745766,0.042839482
4,0.0490866862,-0.0411246717,-0.0618346408,0.00882529,-0.0257049054,0.024287

0133,0.0000041151,-0.0330579206,0.0110619366,-0.0016310412,0.0867833048,-0.0
377732813,-0.0153462263,-0.045176886,0.0483593792,-0.0504501462,0.066710218
8,0.0507157594,-0.0214109141,-0.0144990282,0.0595906787,-0.0369847082,-0.022
2639814,-0.0159568302,0.0506239273,0.0166427083,0.0510329641,-0.026014054
2,0.0093716579,-0.0049215881,0.0267637502,-0.0579008684,0.0103279119,-0.0588
222481,0.0044163414,0.0478421748,0.0297625996,-0.0107406434,-0.098456896
8,-0.0069087488,-0.0159442313,-0.0210836902,-0.016162565,0.0799386874,-0.025
3613517,0.0410504974,0.0388752222,0.0159835275,-0.0183919892,0.010110827
2,0.016431585,0.0144706862,-0.0812029168,0.0138578368,-0.0500751063,-0.04029
20768,-0.0257471893,-0.0099425744,0.0474777594,0.0119158868,-0.024733196
9,-0.0101272073,0.0307049677,-0.017270973,0.0162100121,0.0068897326,-0.00279
35433,0.0022693204,0.0061962181,-0.0182765163,-0.0608113408,-0.051813293
2,-0.0172305889,0.0782410353,-0.036509905,0.0223045032,0.0040078619,0.044926
174,0.0254573748,0.0379255973,-0.0308711976,0.037977919,0.0698476434,-0.0111
421896,0.0049602464,-0.0188961513,0.0249362215,0.0213437248,0.017995717,0.03
73571813,-0.010655283,0.0535632558,0.0035604101,0.0387318097,-0.022746693
3,-0.065291658,0.0147292046,-0.0608935021,-0.0134395808,0.0190870389,-0.0470
229574,-0.0136571042,0.0189618357,-0.0410975888,-0.0481867865,0.000843926
6,0.0447901264,-0.0363597199,-0.1094337702,-0.0000399615,0.0184936561,0.0694
119483,-0.0386126302,-0.005578462,0.0571017042,-0.0169715583,0.030290279
5,0.0187533367,0.0168543626,-0.0413488671,-0.048432976,0.0354357101,-0.01266
83144,-0.0063247979,0.0185201503,-0.0263422262,-0.0234892052,-0.020191656
4,-0.0156724397,-0.0167265628,-0.030575471,0.026522588,0.0549163595,-0.02832
96406,-0.0090253679,0.0086901728,-0.0492822491,-0.0079210037,-0.035718362
8,-0.0378033295,-0.0253920835,-0.008976954,-0.0550671443,0.0559181534,-0.061
6926849,-0.0427054651,-0.03831378,-0.0063040657,-0.0783542246,-0.111650519
1,-0.0146971224,0.0252771527,0.0300703011,-0.0034775715,-0.0037465633,-0.066
7494684,-0.0295872949,-0.0242638532,-0.0956521407,-0.0000597776,-0.02980196
1,0.0191671271,-0.044504486,0.0516240671,0.0416034907,0.0249767136,-0.028801
2307,0.0144216269,0.0451966375,-0.0094211036,-0.0398922972,-0.0712838471,0.0
04453599,-0.0381474681,-0.0141816596,0.024514582,0.0050070095,0.035500861
7,0.0005632785,-0.0388778821,0.0000201296,0.0033344566,-0.0170558151,-0.0078
364592,0.0230167918,-0.0114146452,-0.018564526,-0.0097861625,-0.073350496
6,0.0012674186,0.0180440992,-0.064810127,0.0112708416,0.0408299677,-0.011224
4133,0.0030843855,-0.008651237,-0.0188877769,-0.0395585783,0.0497056693,-0.0
376122855,-0.0056262105,-0.0359596014,-0.0362580419,0.0096146949,-0.00408593
24,0.0625722706,0.0108607905,-0.0086932061,0.0350363925,-0.014320665,-0.0271
379538,0.0611441135,0.0399884954,-0.0197227169,0.090089485,0.025146259,-0.05
88095337,-0.0204007719,0.0718474165,-0.0711568892,0.01406362,0.039384,-0.015
3427543,0.0371531695,0.0320254788,-0.0084373364,-0.0753650516,0.035515718
2,0.0119506605,0.0004035355,-0.0082440013,-0.0143139204,0.0523768291,-0.0504
153445,-0.0048793373,0.0284590218,-0.0357558727,0.0243486427,0.037916839
1,0.0191310253,0.0344840735,-0.1047798619,0.0174622256,0.006863805,-0.031293
0048,0.059266042,0.0204500388,-0.1048679277,0.0507952198,-0.0194242746,-0.01
87288765,0.010258954,0.0257593077,-0.031549532,-0.0380716771,0.013601103
8,-0.033562053,-0.0195613317,0.0718679652,0.0499287434,0.013227175,-0.003079
4346,0.0347005166,-0.0608209968,-0.0030127375,-0.0096188681,0.017505941
9,-0.0311978236,0.0604259223,0.0047816224,-0.0369175971,-0.0101023233,0.0082
815727,-0.0350982696,-0.0063312524,-0.0073843566,0.0041966331,0.02177924
1,-0.0193091705,0.035173703,0.0202876069,0.0231555849,0.0256433692,0.0261426
605,-0.0348945819,0.0295718722,-0.0261461008,-0.0251590721,0.0241272375,0.00
98537784,-0.0358795561,-0.0185927041,-0.0026512353,-0.0329578295,0.023239977
7,-0.026022248,0.0872602314,0.0189642385,0.0367767215,0.0323243774,0.0156558
007,-0.0120929237,0.012526067,-0.0092020212,-0.0061855637,0.0092804963,0.031
2112998,-0.0302573051,0.0120209083,-0.0146117089,0.0340454429,0.023626662
8,-0.009157748,-0.0057725171,0.0021196804,-0.0284296647,0.01844793,-0.010411

3007,0.0530002341,0.0326151848,−0.0141614238,−0.0075403401,−0.017710799
4,−0.0275518075,0.0277634095,0.0562775247,0.0064113336,−0.0164522026,0.02078
12749,0.0338893048,−0.0022566777,−0.0148888351,−0.0043458436,−0.004959836
1,−0.0534195565,−0.0057283654,0.0340512507,−0.0046384954,0.015531891,0.05211
03702,0.0486963205,−0.0569509566,−0.0033283832,0.0458781384,0.0016562773,0.0
06832445,−0.0005373621,−0.0165069476,0.0104681784,−0.0374002159,0.019601635
6,0.0084018828,0.0191872735,−0.039777793,−0.0214678217,0.0324389078,0.004350
9626,−0.0090619624,−0.0008633839,0.0164744984,−0.0272317193,−0.033949099
5,−0.0363581851,0.0451563261,−0.0486265086,0.0431297384,0.0236351248,−0.0350
395814,0.0279374067,0.0181749463,−0.00847957,0.0395078585,0.026341496,−0.025
2951384,0.0084275315,−0.0746877417,−0.0239242576,0.0167219285,−0.016891157
3,0.040121939,−0.0169998873,−0.0058204671,−0.0456035957,−0.0139606493,−0.002
1252336,−0.0194177441,−0.0780003443,0.0135836164,0.0332662798,0.022674251
3,0.0240054615,−0.0033238644,−0.0139610255,0.0756185055,−0.005544533,−0.0260
932259,0.0565012731,0.0272033699,0.0004985439,0.0363242961,0.0016512003,−0.0
297102258,0.0554665923,0.0124557512,−0.0145643353,−0.0979436189,−0.062244456
3,0.0315310098,0.0241697785,−0.0185025074,−0.0155994408,−0.0147026693,−0.021
4723423,0.0440181978,0.0335021503,0.0415125191,0.0226403251,−0.005252458
6,−0.0657926798,0.0592175461,−0.0229622982,0.0178597867,−0.0423047803,−0.025
8290116,0.0623879768,−0.0196865276,−0.0078718709,0.0500154532,−0.026186155
2,0.0624430962,−0.0038765389,0.0367766768,−0.0204350296,−0.0638898686,0.0107
926344,−0.0269380342,0.0359151736,0.0536299273,0.0520150736,−0.003134911
6,0.0217707884,−0.0141534135,−0.0387230292,−0.0355043672,−0.0791495144,0.000
8586664,0.0077624382,−0.0173203461,0.0343447067,−0.0214131791,0.004615220
2,0.00724344,0.0379197523,0.0436494909,−0.0493008234,−0.0041172854,0.0183390
081,0.0544132106,0.0130061759,−0.0197864063,−0.0175142512,0.0430385359]}
{"id":73398579,"embedding":[0.0347948074,−0.0511296578,−0.0554856621,0.01225
5949,0.0135175716,0.0322670378,0.0124115516,−0.0342019312,0.0174311064,0.049
8024449,0.0265598055,−0.0070108976,0.0272089671,−0.0415180922,0.091939739
9,−0.0416853949,0.0422644578,−0.033452712,−0.0003823944,0.0114464918,−0.0231
814943,−0.0295344815,−0.0370825417,−0.0094846953,−0.0013695449,−0.011739914
3,0.0408080854,−0.045050472,0.018981427,−0.019445939,−0.1154571995,0.0632482
171,−0.0972585976,0.048881188,0.0147540644,−0.0668385401,0.0048885276,−0.020
1849267,0.0059581865,−0.0040589464,−0.026554456,−0.0332063995,−0.000015882
2,−0.0296785235,0.0204509757,−0.0377286747,−0.045111414,0.0063857906,0.02373
65738,−0.0402623452,0.0272632241,0.0245300382,0.0276144445,−0.000358340
7,−0.0006580536,−0.068378441,0.0130565213,0.0079405336,0.0042503173,−0.01695
79908,−0.0106005361,0.0357444845,0.0061836797,0.0097310012,0.0090109734,−0.0
531702973,0.0192020033,−0.023886025,0.0388315655,0.00178127,0.008597364
6,−0.0235741064,0.0332110971,0.0004852306,−0.0410749204,−0.0933581367,−0.014
4872796,0.0480406508,0.0221369285,0.0267020892,0.0435552374,−0.014805592
6,−0.0234241057,−0.0257074181,−0.0590608008,0.0698712468,−0.0605910346,−0.00
63910591,−0.0216683429,0.0038960478,−0.015012932,−0.0278181508,0.046123672
3,−0.0448698811,0.0563204139,0.0464849174,−0.0150029249,−0.0321463123,−0.020
9824685,−0.0136958398,−0.0028476987,−0.0043990156,−0.0104220128,−0.039936073
1,0.0428867973,0.0160710365,−0.0016433824,0.0382012874,0.0075731273,0.021198
269,−0.0321084782,0.0249233935,−0.0535628423,−0.0667175874,0.0569121577,−0.0
224253628,0.0109313205,0.1103540286,0.0630063564,0.0217570066,0.076341986
7,0.0103420243,0.0142640229,0.0031453914,0.0046191094,−0.0030424166,0.048212
4388,0.008470038,0.0520702936,0.033473365,−0.0200656895,−0.0032687285,0.0294
897407,0.0050331508,0.0111274514,0.0618709363,0.0112666711,−0.0079621747,0.0
642926618,−0.0372790992,0.0041561858,−0.0170569364,−0.0168007892,0.01101223
1,−0.0293011982,0.0129668778,−0.0133043295,−0.0563077964,0.0471270643,−0.013
4005668,−0.0429586507,0.0045637893,−0.0402378254,−0.0416341387,−0.010966064
4,−0.0672314689,−0.0049526524,−0.0064578955,−0.0006479266,−0.0010833152,0.04
28623036,0.0201527849,0.0373042375,−0.0063103749,0.003757095,−0.00626133

1,0.0039218348,0.0082605025,-0.0666543916,-0.0433271863,0.0178635754,0.01854
22134,-0.0273699369,-0.031609349,0.0132011184,0.0118303467,0.0195724107,-0.0
072581735,-0.0525727756,0.0088742534,-0.0515095331,-0.0372220762,-0.04733262
58,0.0470884442,0.0375851654,0.0204948504,0.0558929928,-0.0286842342,-0.0431
043096,-0.018317312,-0.0120745879,-0.0223189648,-0.0111724725,-0.035657685
3,-0.0006356161,0.0233492702,0.0024443872,0.024153281,-0.0257864352,0.017628
1314,-0.0060215453,0.0756633058,0.002521703,0.0111796604,0.0122606717,-0.005
092599,0.0378126204,-0.0441299528,0.03493426,0.034792047,-0.015212466,0.0074
428059,-0.0718768016,-0.0264538117,0.0001302679,0.0420858972,0.015060975
2,0.0168306436,-0.0146631086,-0.0883900672,-0.0190759636,-0.0134729557,-0.00
87771704,0.0208789464,-0.044885233,0.0201296192,0.0346792638,0.01955552
4,-0.0204755161,-0.0639597923,0.024819348,0.0848714113,0.0032328852,0.016725
8456,0.0358159319,0.0018768394,-0.0233745985,-0.0146298287,-0.002495200
8,-0.0167165697,0.0145038478,0.0624648556,0.0703876689,0.0402430147,-0.08682
8135,-0.022863863,-0.0089568766,0.0056279041,-0.0396106616,-0.0282406714,0.0
029124303,-0.0112464344,0.0147071928,-0.0201610401,-0.0908167735,-0.00575286
34,-0.0744647905,0.0071259122,0.0553814285,0.0247332714,0.0670324042,-0.0081
194816,-0.0285497326,-0.0247543082,-0.0106861042,-0.0242830105,-0.015060892
3,-0.0505471006,-0.0276883692,-0.0019830519,-0.0089115268,-0.0389968418,0.07
14137331,0.0057519134,-0.0066149416,0.0340775698,-0.0098825144,0.065413691
1,0.061008513,-0.0586832128,-0.0134075135,0.0562846623,0.0367136225,-0.03178
46909,-0.0481961705,0.0015329903,-0.0694581047,0.0001967799,0.03074353,-0.02
23478898,-0.0453722812,0.0128244897,0.0008126254,-0.0341653936,-0.000674975
6,0.0262963846,-0.0483848155,0.0002108308,0.0038434442,-0.0210278369,0.04272
56562,0.0074925232,-0.0082666241,-0.0401596762,0.0461624525,0.048130907
1,-0.0251728632,-0.0030496886,-0.0129158003,-0.015655756,0.0088347811,-0.055
8250956,-0.0093234796,-0.0027295672,0.0536782034,0.0266477317,-0.01935270
8,-0.0493553691,0.0093463548,0.0571557619,-0.0156452246,0.0667723268,0.01343
35188,-0.0081324307,0.0227436069,0.0521050058,-0.0688745454,-0.008839979
8,-0.015164176,0.0269702021,-0.0249063112,0.0637155548,-0.0389110819,0.03892
58116,-0.0261735767,0.0339068063,-0.0618306696,0.0128670484,-0.050590507
7,-0.002721986,0.0480442941,0.0281090904,-0.0028700328,-0.0536891483,0.00549
56358,-0.0026309409,-0.0397192016,-0.0447318777,0.0494757406,0.032333008
9,0.0259053223,0.0929670632,-0.0295515563,-0.0235264227,0.0090606241,-0.0107
592959,-0.017231321,0.0197738875,0.0033077789,-0.0543156825,-0.047690514
5,0.0021648386,0.0175177306,-0.0058223614,-0.0196560416,-0.009198375,0.05500
61353,0.0227673054,-0.0306520686,0.0716030449,0.0411643535,-0.055023297
7,-0.0161119364,0.0547857694,0.0160441063,-0.0352882445,-0.0015436634,-0.023
9779241,0.0406555235,0.0038617079,-0.041896943,-0.0127281547,0.023094931
6,0.0199444238,-0.0075386791,0.0161178987,0.0086277863,0.0114953667,0.005734
5191,-0.0062919683,0.0043568457,0.0195062086,0.0624580197,0.0197663661,0.066
4491951,0.0009230238,0.0327940248,0.0299387574,0.0779216513,-0.037155196
1,0.0124577889,-0.0327001177,-0.0463274047,-0.026033124,0.0214834716,-0.0685
717836,-0.0050877272,-0.0238360818,0.0064770002,0.0016805759,-0.014010485
3,0.07531856,0.045619946,-0.0589902066,0.0003707073,-0.0021888106,0.01737772
86,-0.034283556,0.0221830066,0.0364745259,-0.0064829597,-0.0436779261,-0.038
9204063,-0.0425970368,-0.0332704559,-0.0399974361,-0.0028189167,0.018078239
6,-0.0008099452,0.0291259736,-0.0029149808,0.0491607934,0.0038605332,-0.0067
793601,0.0316342674,-0.0747226253,0.010491577,0.025117306,-0.0114292158,-0.0
133304233,0.0193296056,-0.0344868079,0.0561891496,0.0201287214,-0.031844280
7,-0.0438917801,0.0153256198,-0.0451969393,0.0583543777,-0.0866529197,0.0340
694822,-0.0773928612,-0.0499339513,-0.070778273,-0.0577972792,0.019594833
3,0.0267296415,0.0730974153,0.0287379213,-0.0240483396,-0.0019626506,-0.0317
781791,-0.027369367,-0.0787498802,-0.0225058887,-0.0087802643,0.036263365
3,-0.0019933251,0.0233351421,0.0520008467,0.0055651148,-0.0189081114,-0.0328
941084,0.0406036861,-0.0155531373,-0.0178763159,-0.0897395313,0.029215361

9,0.0132709332,-0.0045883418,0.0215732455,0.0038592548,0.0227726698,0.044094
8382,0.0253935978,-0.0220641904,0.0035650786,0.0133753661,0.0377138294,0.038
7166031,-0.0255935006,0.0186080746,0.0061261277,-0.0503460355,0.027992254
1,0.0273637846,-0.0137144187,0.0629460588,0.0548909381,0.0092025315,0.019047
6011,0.0108602084,0.0102045676,-0.0553058982,0.0364280269,0.0024036814,-0.02
55201794,0.0324015804,0.0266994666,0.0292204786,0.015494396,0.005585083
3,-0.0011340061,-0.0089273388,0.0417408943,-0.0226956308,0.0177166015,0.0092
701772,0.0026091053,0.0114931772,0.0261468869,0.0050089932,-0.0934821144,0.0
08182453,0.0160108954,-0.1124326959,-0.0106834145,0.0310967602,-0.005938417
7,0.0592273884,0.0103498986,0.0608088337,-0.0766836926,-0.0404612124,-0.0071
2557,-0.0037265762,0.0138926599,-0.0288243685,0.0548437946,-0.0345228612,0.0
387822725,-0.0489589423,-0.0634824783,-0.0081020277,-0.0115952175,0.02464984
92,0.0113900062,-0.0998153761,0.0121294446,0.0175945312,0.036677599,-0.01555
93688,0.0279113073,-0.0311089717,0.0486244708,-0.0296390075,-0.011377161
4,0.0032802655,0.0383770429,0.0151136452,-0.0199087616,-0.0186609607,0.00642
0854,-0.0365454033,-0.0052468823,0.0264917202,0.0350365415,-0.0268738735,0.0
516756959,-0.0262431167,-0.0199887417,0.0061279507,0.0037065975,0.009519344
2,0.0585568845,-0.0503190532,-0.0295248441,-0.0027437676,0.0091730263,-0.001
1861803,0.0175728127,-0.0105229719,0.0301239602,0.033174105,0.0004233483,0.0
65837808,0.0047172084,0.0395955481,0.0371558145,-0.0028413401,-0.042499110
1,0.0480613448,-0.0194673315,-0.0234299637,-0.0070342831,0.0442721397,0.0496
700034,-0.0541004501,-0.0058011552,-0.0185724553,0.013772591,-0.008866287
8,0.0902864859,0.0189512037,0.0105405422,0.0179324597,-0.0099890353,0.017646
227,-0.0171150099,0.0431456529,0.0290721785,0.0195296537,0.0120260082,-0.031
7893364,-0.0425233357,0.0029610905,0.0198555421,0.0302588921,0.021084837
6,-0.0465390794,0.0129986806,0.0242489558,-0.0322489291,-0.0475747846,0.0455
248244,0.0032983508,0.0034702781,-0.032658726,0.087853983,0.053858649,0.0223
465916,0.0712229982,0.004888,-0.0753872469,0.0011574343,0.0376602188,-0.0098
21998,-0.0352158211,0.0553436801,-0.0246598106,-0.0736191049,0.036756161
6,0.0910897329,0.0181416199,-0.0227895193,0.0134329237,0.025441017,-0.042945
5228,-0.0773490295,-0.0270636491,-0.0128344717,-0.0110838683,-0.003650606
1,0.0129330801,-0.009517408,-0.023568185,0.0089691905,-0.0073556001,0.008700
6465,0.0010708276,-0.0382597856,0.028777007,0.0240480434,0.0510234088,0.0416
004397,-0.0290064551,-0.04025295,-0.0452741534,-0.0241271872,0.035911213
6,-0.0783603489,0.0239061583,0.0255217273,-0.0268921275,0.0450253412,-0.0091
675492,-0.0166545603,0.0179021433,0.0118761286,0.0142783122,-0.007903867
4,-0.0214275699,-0.0482592173,0.0702252612,0.0162170753,-0.0201829411,0.0150
575051,0.0086476877,0.0018286464,-0.051388707,-0.0015249776,-0.024727771
1,-0.0910961181,0.0298211165,0.0669244677,-0.0086954217,0.0214916933,-0.0126
937982,-0.0111289388,0.0751579627,-0.0158291236,-0.0120425448,-0.016176817
9,0.0377090722,-0.0116715617,-0.022859985,0.0277934428,-0.0004091347,0.05662
36936,-0.0059747593,0.0240753107,-0.0668295473,0.0067333952,0.037080887
7,-0.0309754498,0.0491584092,0.008883303,-0.0177758131,0.0010658769,0.057520
8068,0.0267259758,-0.0194260646,0.0471782275,-0.0600472242,-0.014490569
9,-0.0092263743,-0.0199925024,0.0268446412,-0.0514241867,-0.0384148322,0.037
2209288,-0.0526666678,-0.0426664241,0.0678725466,-0.0659822673,0.09464250
5,0.0097955773,0.0147216907,-0.0260044225,-0.0602928251,0.0047598276,0.01808
44069,-0.0078883106,0.0228560083,0.0365793444,-0.0239058044,-0.016822140
7,-0.0342782475,-0.0083312821,-0.0243636798,-0.0499641635,0.0299963187,-0.00
77241426,0.0323708244,0.0607367158,-0.0158398822,-0.034892194,0.022255791
4,-0.0403763689,0.0281146728,-0.0349664092,0.0644345358,-0.0090374937,0.0330
020636,0.0119843567,-0.0175327566,0.019094605,0.0437422059]}
{"id":73327915,"embedding":[0.075414069,-0.0344245546,-0.0539125577,-0.02030
39385,0.0098982193,0.0307554081,0.0117364572,-0.0069823391,0.0221927669,0.05
12993783,0.044971928,-0.0125001194,0.0129903536,0.0072021331,0.022502135
5,0.0060420195,0.0267787203,-0.0226366185,-0.000392686,-0.0544092357,0.00081

24994,−0.0067854174,0.0023721452,−0.0251497664,0.0131236091,−0.02837193
2,−0.0016028861,−0.0479160622,−0.0072472701,0.0446752124,−0.0720009133,0.082
1264535,−0.0462338105,0.0323645622,0.0262354854,−0.0566928573,0.022150762
4,0.0059301481,0.0051082345,0.0273782238,−0.003369659,0.0048650545,−0.046183
0646,0.0119586727,−0.0095675327,−0.0121102361,0.0510081798,0.0349347703,−0.0
010106612,−0.02210203,−0.0155191198,−0.001406203,0.0615978278,−0.013897260
7,−0.0781367868,−0.0645616055,0.030741673,0.0066694403,0.018236693,0.0571120
605,−0.0152700972,0.0487475134,−0.0793202817,0.033422444,−0.0049528182,−0.05
34683764,−0.0051416573,−0.0144408504,0.0607302412,0.0266380198,0.003306166
3,0.021387348,0.0496433191,−0.0519526452,−0.0167464353,−0.1000095531,−0.0167
476814,0.0624853447,−0.0017406171,0.0205323882,−0.0334635563,−0.039392050
4,−0.0212754402,0.0390604511,−0.0115969339,0.0559972264,−0.0164989568,−0.009
8142847,−0.0145596424,0.0578153655,−0.0324543007,−0.018272711,0.03152468
8,−0.0562541597,0.0303131677,0.046551574,−0.0331530645,−0.0229616165,0.05209
71008,0.0031413205,−0.0072570457,−0.049369704,−0.0279942229,−0.003489379
3,0.040419843,0.0142639019,0.0099652614,0.0336816087,0.011750849,0.041420526
8,−0.0437359624,0.0115187513,0.0269702133,−0.0105187185,0.0347701572,−0.0399
428606,−0.0175378174,0.0527053848,0.0202110317,0.0116301244,0.0357996635,0.0
051744925,0.0633528158,0.0134612974,0.0033431051,0.0205549803,0.046258937
6,0.0301738586,0.0758080184,0.0345477164,−0.0037000277,−0.0430374891,0.01955
70234,−0.0159876496,0.0325545408,0.050522238,0.0315247923,−0.0126983635,0.00
40711705,0.0281610843,0.0393548645,−0.0083833337,−0.029910462,0.045841131
4,−0.0042435843,0.017927492,−0.0168174952,0.0027482447,0.0526623726,−0.00938
65078,−0.0328201279,−0.0267420169,−0.0063666347,−0.0170394573,0.009847435
2,−0.0050446657,0.000367116,0.006293437,0.0034414462,−0.0060996152,0.0609054
007,0.0185491852,−0.0128095727,0.0011517021,−0.0301907882,−0.0272203218,0.02
70147491,−0.0202808734,−0.0357338786,−0.004674104,−0.0063396082,0.013771386
8,−0.0603799783,−0.0487900116,−0.0277446751,−0.0738709718,0.0333957337,0.010
2670295,−0.0489488356,0.017028749,−0.0460819155,−0.0120298108,−0.023221358
7,0.085039176,0.0295955352,0.0307544526,0.0634101331,−0.0618282631,0.0056828
363,−0.0369515046,−0.0218073633,0.025246514,−0.0062321606,−0.0571246631,0.02
10108366,−0.0171552617,−0.0123425396,0.036775209,−0.0143130887,−0.024885596
7,0.022573363,0.0418979712,0.0077413628,0.0268669929,0.0564329326,−0.0152893
448,0.0485670343,−0.020164663,−0.0212592371,0.0342566781,−0.0453273728,0.034
901347,−0.0597589575,0.0089804288,0.0204684101,0.0425026901,0.0291977674,0.0
352058597,0.0121031767,−0.0506576635,−0.020753162,0.0052019139,0.001917059
9,−0.002540481,−0.0255327579,0.0198472552,−0.0173663702,0.0043540597,−0.0138
72222,−0.0365234874,−0.0106985513,0.0880684033,0.0199568737,0.0004767441,0.0
422571711,−0.0336901471,−0.0192919355,0.0221781656,−0.0110588372,−0.00599605
12,−0.0737432763,0.0018896272,0.0650891811,0.0154899415,−0.0634473488,0.0600
116588,0.0099573983,0.0252448861,−0.0168739446,−0.0069328835,−0.02579212,0.0
293264985,0.0148067279,0.0543016531,−0.0521508269,0.0360518359,−0.068806208
7,0.0148141393,−0.0041232514,0.0019113434,0.0551198609,−0.0253854562,−0.0184
076931,−0.0132358409,−0.0194051061,−0.0321767889,0.0341670364,−0.085154540
8,−0.0159634724,0.0121243959,0.0145893842,−0.0211053528,0.0894619972,0.04236
2228,−0.0092093823,0.0712616295,−0.0168433879,0.0050816899,0.0441521481,−0.0
416787453,−0.0107683297,0.0792052597,0.0071034078,−0.0812779218,−0.061767011
9,−0.0200847592,−0.0509822071,−0.0193420555,0.03569277,−0.0376436263,−0.0356
053747,0.038968645,−0.002695282,−0.0611955114,0.0083284788,0.0247329194,0.00
29349618,0.0591344833,0.0142543679,0.0187873486,−0.0341714211,−0.011597275
7,−0.0531016737,−0.0317720622,−0.0018326384,−0.0459940135,−0.0220446568,−0.0
41684404,−0.000496277,0.02873246,0.043845363,−0.0548030995,0.0127426526,−0.0
21339206,0.0175562482,0.0089848982,−0.0202203244,−0.0113815712,−0.02241634
2,0.0522015616,−0.0345671289,0.0295723956,0.0291620735,−0.0140700247,0.00257
00815,0.0042348723,−0.0254749805,0.0013872056,0.0200054478,0.0247689616,−0.0
733883604,0.0447113737,−0.015700534,−0.0072467588,−0.0245293379,0.096383765

3,−0.1147417873,0.023120869,−0.0403214246,0.020445941,0.0487103127,0.0539496
578,−0.0103349462,−0.0462801568,0.0007969044,−0.0381563269,−0.026869911
7,−0.0091623804,0.0435522199,0.0163947362,−0.0016163903,0.041709654,−0.04277
75532,−0.0317739435,0.0376865044,0.0545457676,−0.0112211974,0.0400158614,0.0
223112907,−0.0275484659,−0.0447190553,0.0101300133,0.0219515879,−0.003718736
6,0.0138965342,−0.0368879251,0.0075315451,0.011962397,−0.0492267162,0.044497
5644,−0.005662085,−0.0319865122,−0.0182841234,−0.0141949607,−0.024404648
7,−0.0168251153,−0.0250175744,−0.0103184907,−0.0212502815,0.0180578604,0.033
2501568,−0.0082649784,0.0034479096,0.066655986,0.0026372743,0.0251481365,0.0
706725046,0.0694615915,0.0243647899,0.0380428582,0.0272612628,0.054145321
2,0.0573015101,−0.0188297909,0.0689861104,−0.0067009605,−0.0297109857,−0.008
1229955,0.0126254894,−0.0295597538,−0.0245347992,−0.0200936981,−0.047656472
8,−0.0008015644,0.0331115872,−0.0104803676,0.0043606218,−0.0458753742,−0.013
9830997,−0.0029952512,0.0200630426,0.0396644771,0.0152914152,−0.043122321
4,−0.0309076589,−0.0089381523,0.0913453102,−0.0157789066,0.0038208787,0.0729
248971,0.0142303351,−0.0099023934,0.0165794678,0.0422475636,−0.028096055
6,−0.0703010187,−0.0223158505,0.005072359,0.0345187895,0.0539987125,−0.00591
9768,0.0303612147,−0.0195170585,−0.0672300905,0.0014867462,−0.046968814,−0.0
039160387,0.0202672035,−0.0045328168,−0.0116649549,0.0045845988,−0.019661184
4,−0.000479098,0.0185858998,−0.0290821325,−0.0432731956,0.0079176761,−0.0555
472672,0.0485342182,−0.1040112153,0.0249223635,−0.0693835542,0.008328883
9,−0.0326963924,−0.0281802136,−0.058906842,0.084232159,0.016716443,−0.007783
8432,−0.0027969186,−0.005241238,−0.0094840024,−0.0424887352,−0.064282499
3,0.0391408056,0.0183708053,−0.0090369033,−0.0474776737,0.0946922973,0.03002
32153,−0.0149719361,0.0118109724,0.0238779671,0.0012582076,−0.0310533978,0.0
255749971,−0.0271577872,0.025791971,0.0215604585,−0.0359305777,0.014036809
1,0.0173427686,0.0726925954,0.0165873729,0.0062742601,0.0094242394,0.0351746
194,−0.0364881121,−0.0383503214,0.0195421334,−0.0458547734,−0.005626429
3,−0.0205675848,−0.0304918196,0.0023549586,0.050250113,−0.0775636807,0.00370
55579,0.0414554663,0.0141480519,−0.0317468122,−0.011900533,0.0288984124,−0.0
916069895,0.0654196888,−0.023870334,0.0363810025,−0.0196598098,−0.014377124
6,0.045047842,−0.0098303398,0.020766113,0.0030382266,−0.0062567079,0.0374899
916,0.0301935505,−0.036296647,0.0212481096,−0.0060201259,−0.0246502031,0.058
1530593,−0.0033352021,−0.0984618217,−0.0055043455,−0.0002567676,−0.098188094
8,0.0260610171,0.0426560156,−0.0593066812,−0.0400590822,−0.009508362,0.00780
62946,−0.0678725392,−0.0528077818,0.0197090227,−0.0380254164,0.059018123
9,−0.0053968686,0.0304792915,0.0235992372,0.0280020032,0.0001023485,−0.04287
11213,0.0199120548,−0.0006891226,−0.003276835,0.026062455,−0.0641255081,0.01
44926999,0.0428459942,−0.0240885541,−0.0232304577,−0.0085502118,−0.04952848
7,0.0509642437,−0.0049572345,−0.0252528153,−0.0100031085,0.03722528,0.000418
8125,−0.0039367601,0.022884354,0.0027193611,0.004912816,0.0785848051,0.00574
93714,0.0133816525,0.0012933919,0.0473028906,−0.0019906054,−0.061153262
9,−0.0431011096,0.018630974,0.0317591652,−0.0033661521,−0.0231325272,−0.0306
341369,−0.0300955735,−0.0020349366,−0.0008743384,0.0240043141,0.007857227
7,−0.0351116285,0.0508812256,−0.0377641842,0.0077841831,0.0213861857,0.01416
05418,0.0463785864,0.0427033938,−0.0668713674,0.0761618763,−0.005191778
8,−0.0155472904,0.0101401675,0.0044801165,0.0156602859,−0.0285437554,−0.0084
964996,−0.0070108846,0.0516727343,−0.0234931149,0.0940077677,−0.031788993
6,0.0507893227,0.0477987975,0.0096739288,−0.0020982516,0.0262101032,0.017989
4809,0.0205602739,0.0442986339,0.0157728158,−0.0331553891,−0.0259460919,−0.0
107381223,0.0342226662,−0.0035206561,0.0219117459,−0.0345894732,−0.005369731
7,0.0381090604,−0.0394515097,0.0102304043,0.0522392765,−0.0339088738,−0.0536
20439,−0.0427841768,0.0467144474,−0.00487269,−0.0142774256,0.1195062324,0.00
88427169,−0.0548100695,−0.039315585,−0.0065632053,0.0203854032,0.042495805
8,0.0142258657,−0.0022552579,−0.0496399067,0.0022382906,0.0497406311,−0.0060
742483,−0.047468029,0.076349102,0.0596321672,−0.054425098,−0.0450256504,−0.0

176661927,-0.0626949221,-0.0321074761,0.0532431938,0.0069885585,-0.0025023608,-0.0278226975,0.012137441,-0.0626647174,-0.0508361459,-0.0254623052,-0.0357661061,0.0325860716,-0.0223803669,-0.0069467407,-0.0064564333,-0.0488831513,-0.0076247035,-0.052731365,-0.0577694289,0.0480457097,-0.0527657233,0.037565127,0.026677778,0.0036674966,0.0235869121,0.0167714022,0.007084433,0.0339985341,0.0636064783,0.0106880143,-0.0145894159,-0.036188256,-0.0246377513,0.0106241843,-0.0007273352,0.0391281173,0.0104876105,-0.0035966153,-0.0442153327,-0.0168285184,0.0134935547,-0.0388396271,-0.0406457968,0.0219227597,0.0365763269,0.0126615148,0.0239277706,0.0722400844,0.0213607196,0.0167508442,-0.0243490431,-0.0245436467,-0.0156331956,0.0248535089,0.0430515669,-0.0210927911,-0.0160750709,-0.0466567166,0.0192894954,-0.0182032138,0.0618641376,-0.0328283571,-0.0253621396,0.0373378582,-0.0175270885,0.0046139401,-0.0186457913,0.0119397584,-0.0316541605,0.059680894,-0.00740259,0.0248539317,0.0033964238,-0.0571172982,-0.0342137963,0.0461037867,-0.038586136,0.0145851653,-0.0098661277,-0.0257777199,0.0677331164,-0.0074066813,-0.0192235615,0.0267544128,-0.0139606725,0.0523501597,-0.0030906596,0.021780327,-0.0730073005,-0.0361867957,0.0184862856,-0.0238689147,0.01322784,0.0719438642,0.0352646522,-0.0641366988,0.0514607839,-0.0263158642,-0.0160666686,-0.0148193743,-0.0502597839,0.0069855079,0.0288299322,0.0385488756,0.053073898,-0.0662418529,-0.0418642424,0.0414642654,0.0074641416,0.0084358733,-0.0356520526,0.0262771863,-0.0195083879,0.0534724705,0.0152129419,0.0284705367,-0.0067897094,-0.0183840618]}

Then, create a new Cloud Storage bucket and copy the file to it.

In [19]:
```
BUCKET_URI = f"gs://{PROJECT_ID}-embvs-tutorial-{UID}"
! gsutil mb -l $LOCATION -p {PROJECT_ID} {BUCKET_URI}
! gsutil cp questions.json {BUCKET_URI}
```

```
Creating gs://qwiklabs-gcp-03-8b8421a4ae80-embvs-tutorial-10211900/...
Copying file://questions.json [Content-Type=application/json]...
/ [1 files][  9.8 MiB/  9.8 MiB]
Operation completed over 1 objects/9.8 MiB.
```

Create an Index

Now it's ready to load the embeddings to Vector Search. Its APIs are available under the aiplatform package of the SDK.

In [20]:
```
# init the aiplatform package
from google.cloud import aiplatform

aiplatform.init(project=PROJECT_ID, location=LOCATION)
```

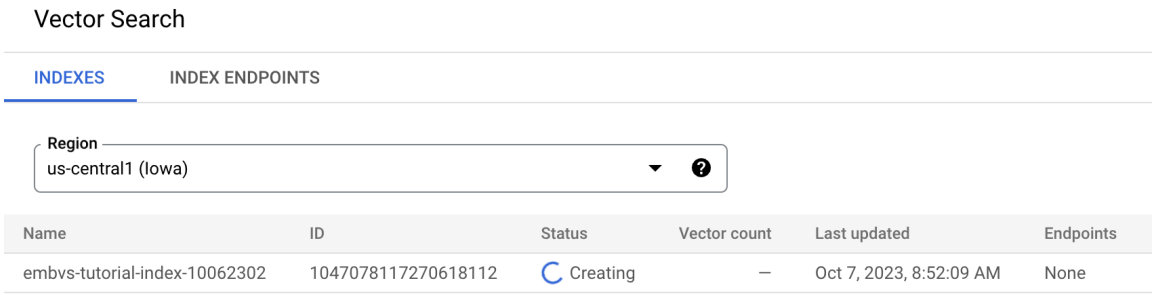
Create an MatchingEngineIndex with its `create_tree_ah_index` function (Matching Engine is the previous name of Vector Search).

In [21]:
```
# create index
my_index = aiplatform.MatchingEngineIndex.create_tree_ah_index(
    display_name=f"embvs-tutorial-index-{UID}",
    contents_delta_uri=BUCKET_URI,
    dimensions=768,
    approximate_neighbors_count=20,
    distance_measure_type="DOT_PRODUCT_DISTANCE",
```

```
)
```

```
Creating MatchingEngineIndex
Create MatchingEngineIndex backing LRO: projects/143428112324/locations/us-c
entral1/indexes/6315432062224433152/operations/4759476495917776896
MatchingEngineIndex created. Resource name: projects/143428112324/locations/
us-central1/indexes/6315432062224433152
To use this MatchingEngineIndex in another session:
index = aiplatform.MatchingEngineIndex('projects/143428112324/locations/us-c
entral1/indexes/6315432062224433152')
```

By calling the `create_tree_ah_index` function, it starts building an Index. This will take under a few minutes if the dataset is small, otherwise about 50 minutes or more depending on the size of the dataset. You can check status of the index creation on the Vector Search Console > INDEXES tab.



The parameters for creating index

- `contents_delta_uri` : The URI of Cloud Storage directory where you stored the embedding JSON files
- `dimensions` : Dimension size of each embedding. In this case, it is 768 as we are using the embeddings from the Text Embeddings API.
- `approximate_neighbors_count` : how many similar items we want to retrieve in typical cases
- `distance_measure_type` : what metrics to measure distance/similarity between embeddings. In this case it's `DOT_PRODUCT_DISTANCE`

See the document for more details on creating Index and the parameters.

Batch Update or Streaming Update?

There are two types of index: Index for *Batch Update* (used in this tutorial) and Index for *Streaming Updates*. The Batch Update index can be updated with a batch process whereas the Streaming Update index can be updated in real-time. The latter one is more suited for use cases where you want to add or update each embeddings in the index more often, and crucial to serve with the latest embeddings, such as e-commerce product search.

Create Index Endpoint and deploy the Index

To use the Index, you need to create an Index Endpoint. It works as a server instance accepting query requests for your Index.

In [22]:
```
# create IndexEndpoint
my_index_endpoint = aiplatform.MatchingEngineIndexEndpoint.create(
    display_name=f"embvs-tutorial-index-endpoint-{UID}",
    public_endpoint_enabled=True,
)
```

```
Creating MatchingEngineIndexEndpoint
Create MatchingEngineIndexEndpoint backing LRO: projects/143428112324/locati
ons/us-central1/indexEndpoints/8574761331536691200/operations/59123980005246
23872
MatchingEngineIndexEndpoint created. Resource name: projects/143428112324/lo
cations/us-central1/indexEndpoints/8574761331536691200
To use this MatchingEngineIndexEndpoint in another session:
index_endpoint = aiplatform.MatchingEngineIndexEndpoint('projects/1434281123
24/locations/us-central1/indexEndpoints/8574761331536691200')
```

This tutorial utilizes a Public Endpoint and does not support Virtual Private Cloud (VPC). Unless you have a specific requirement for VPC, we recommend using a Public Endpoint. Despite the term "public" in its name, it does not imply open access to the public internet. Rather, it functions like other endpoints in Vertex AI services, which are secured by default through IAM. Without explicit IAM permissions, as we have previously established, no one can access the endpoint.

With the Index Endpoint, deploy the Index by specifying an unique deployed index ID.

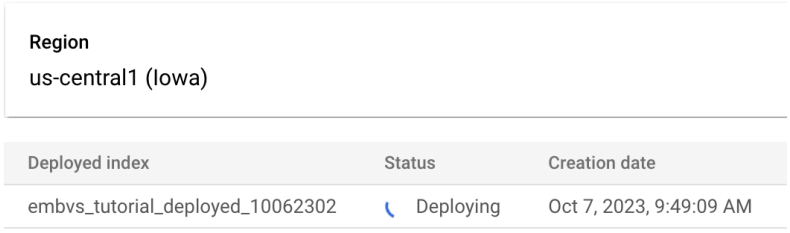
In [23]:
```
DEPLOYED_INDEX_ID = f"embvs_tutorial_deployed_{UID}"
```

In [24]:
```
# deploy the Index to the Index Endpoint
my_index_endpoint.deploy_index(index=my_index, deployed_index_id=DEPLOYED_IN
```

```
Deploying index MatchingEngineIndexEndpoint index_endpoint: projects/1434281
12324/locations/us-central1/indexEndpoints/8574761331536691200
Deploy index MatchingEngineIndexEndpoint index_endpoint backing LRO: project
s/143428112324/locations/us-central1/indexEndpoints/8574761331536691200/oper
ations/8388251895671554048
MatchingEngineIndexEndpoint index_endpoint Deployed index. Resource name: pr
ojects/143428112324/locations/us-central1/indexEndpoints/8574761331536691200
```

Out[24]:
```
<google.cloud.aiplatform.matching_engine.matching_engine_index_endpoint.Mat
chingEngineIndexEndpoint object at 0x7f1fc64c12a0>
resource name: projects/143428112324/locations/us-central1/indexEndpoints/8
574761331536691200
```

If it is the first time to deploy an Index to an Index Endpoint, it will take around 25 minutes to automatically build and initiate the backend for it. After the first deployment, it will finish in seconds. To see the status of the index deployment, open the Vector Search Console > INDEX ENDPOINTS tab and click the Index Endpoint.



Run Query

Finally it's ready to use Vector Search. In the following code, it creates an embedding for a test question, and find similar question with the Vector Search.

In [25]:
```
test_embeddings = get_embeddings_wrapper(["How to read JSON?"])
```

```
100%|██████████| 1/1 [00:01<00:00,  1.14s/it]
```

In [26]:
```
# Test query
response = my_index_endpoint.find_neighbors(
    deployed_index_id=DEPLOYED_INDEX_ID,
    queries=test_embeddings,
    num_neighbors=20,
)

# show the result
import numpy as np

for idx, neighbor in enumerate(response[0]):
    id = np.int64(neighbor.id)
    similar = df.query("id == @id", engine="python")
    print(f"{neighbor.distance:.4f} {similar.title.values[0]}")
```

```
0.7944 How can I display 100 json data in HTML?
0.7473 How to Insert array inside JSON object?
0.7463 While reading json data from my document DB, I want to add some confi
guration using which part of json structure would be transformed into a pojo
0.7457 How to convert JsonElement to a Int or other primitive types
0.7384 How to filter an array of json with jq in linux?
0.7226 Creating specific Json with python dictionaries
0.7076 Circe: decoding Json field with alias, how to do it?
0.7068 convert json list to a data frame in R
0.7026 How to base64 decode value of dynamic json array in NiFi?
0.6970 Redshift : loading & storing JSON (IoT) data
0.6964 How I use python get opendata(xml) data
0.6922 Can JSON strings be converted to fractions for equations when buildin
g REST API in Node JS?
0.6917 XML Config, How can I read a configuration file with a list/array of
values rather than a single value
0.6912 Read audio file from s3 directly in python
0.6831 How do I show API response data in one component by calling it from a
nother component?
0.6819 How do you receive a status code from HTTP using libogc?
0.6797 TJsonSerializer with arrays
0.6790 Jest how to test a map contents i.e. key and value
0.6775 Access a nested array of objects in nodejs, mysql and json
0.6771 How I can select object by name with JMESPath?
```

The `find_neighbors` function only takes milliseconds to fetch the similar items even when you have billions of items on the Index, thanks to the ScaNN algorithm. Vector Search also supports autoscaling which can automatically resize the number of nodes based on the demands of your workloads.

IMPORTANT: Cleaning Up

In case you are using your own Cloud project, not a temporary project on Qwiklab, please make sure to delete all the Indexes, Index Endpoints and Cloud Storage buckets after finishing this tutorial. Otherwise the remaining objects would **incur unexpected costs**.

If you used Workbench, you may also need to delete the Notebooks from the console.

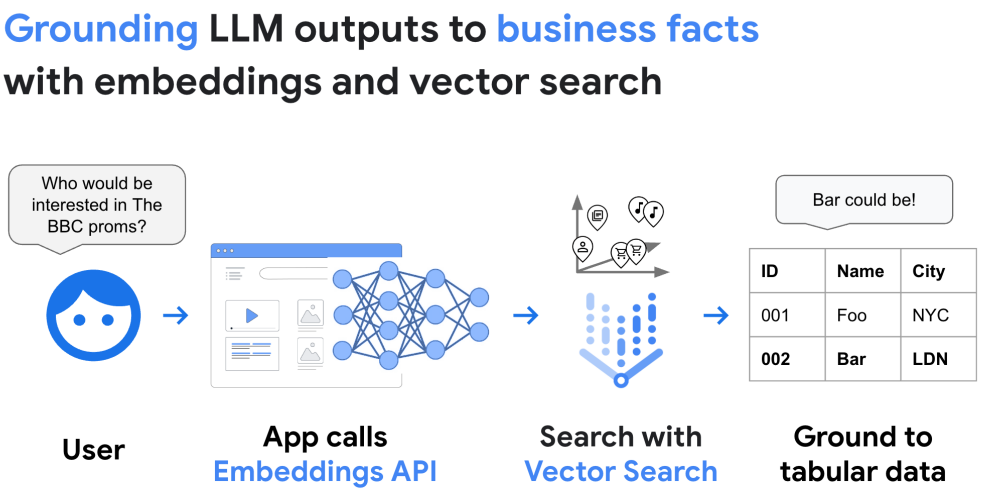
In [ ]:
```
# wait for a confirmation
input("Press Enter to delete Index Endpoint, Index and Cloud Storage bucket:

# delete Index Endpoint
my_index_endpoint.undeploy_all()
my_index_endpoint.delete(force=True)

# delete Index
my_index.delete()

# delete Cloud Storage bucket
! gsutil rm -r {BUCKET_URI}
```

Summary

Grounding LLM outputs with Vertex AI Vector Search

As we have seen, by combining the Embeddings API and Vector Search, you can use the embeddings to "ground" LLM outputs to real business data with low latency.

For example, if an user asks a question, Embeddings API can convert it to an embedding, and issue a query on Vector Search to find similar embeddings in its index. Those embeddings represent the actual business data in the databases. As we are just retrieving the business data and not generating any artificial texts, there is no risk of having hallucinations in the result.



The difference between the questions and answers

In this tutorial, we have used the Stack Overflow dataset. There is a reason why we had to use it; As the dataset has many pairs of **questions and answers**, so you can just find questions similar to your question to find answers to it.

In many business use cases, the semantics (meaning) of questions and answers are different. Also, there could be cases where you would want to add variety of recommended or personalized items to the results, like product search on e-commerce sites.

In these cases, the simple semantics search don't work well. It's more like a recommendation system problem where you may want to train a model (e.g. Two-Tower model) to learn the relationship between the question embedding space and answer

embedding space. Also, many production systems adds reranking phase after the semantic search to achieve higher search quality. Please see Scaling deep retrieval with TensorFlow Recommenders and Vertex AI Matching Engine to learn more.

Hybrid of semantic + keyword search

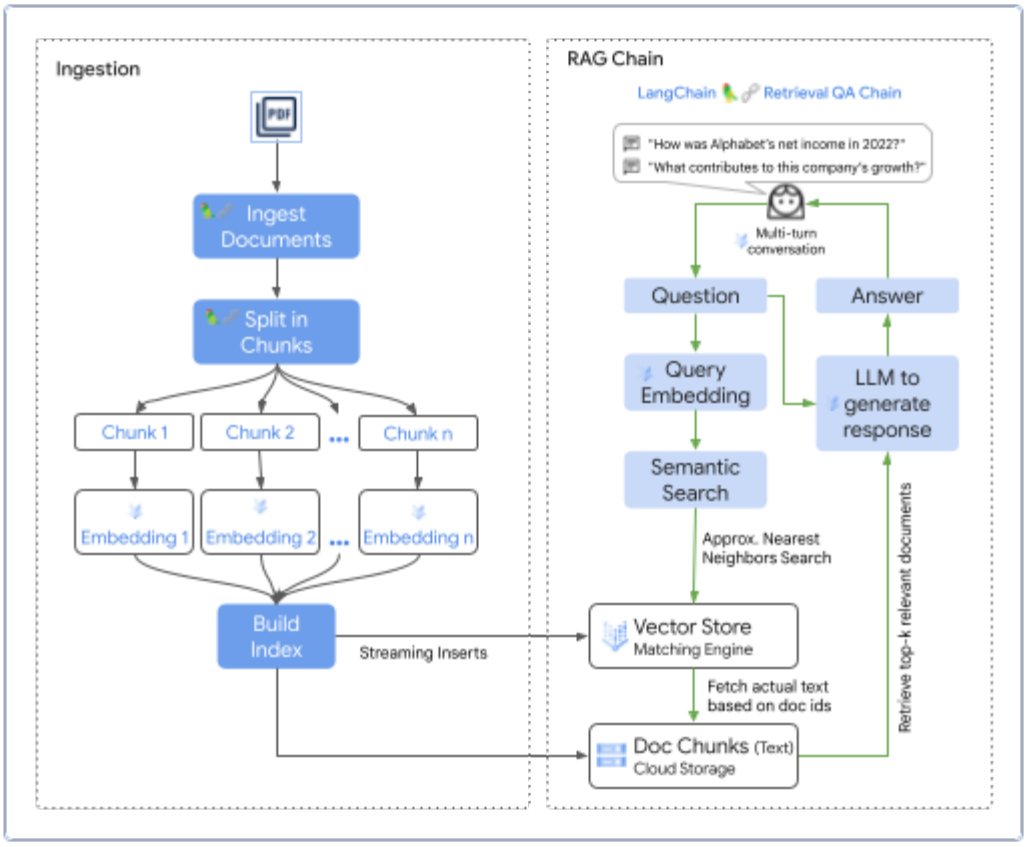
Another typical challenge you will face in production system is to support keyword search combined with the semantic search. For example, for e-commerce product search, you may want to let users find product by entering its product name or model number. As LLM doesn't memorize those product names or model numbers, semantic search can't handle those "usual" search functionalities.

Vertex AI Search is another product you may consider for those requirements. While Vector Search provides a simple semantic search capability only, Search provides a integrated search solution that combines semantic search, keyword search, reranking and filtering, available as an out-of-the-box tool.

What about Retrieval Augmented Generation (RAG)?

In this tutorial, we have looked at the simple combination of LLM embeddings and vector search. From this starting point, you may also extend the design to Retrieval Augmented Generation (RAG)&oq=Retrieval+Augmented+Generation+(RAG)).

RAG is a popular architecture pattern of implementing grounding with LLM with text chat UI. The idea is to have the LLM text chat UI as a frontend for the document retrieval with vector search and summarization of the result.



There are some pros and cons between the two solutions.

	Embeddings + vector search	RAG
Design	simple	complex
UI	Text search UI	Text chat UI
Summarization of result	No	Yes
Multi-turn (Context aware)	No	Yes
Latency	millisecs	seconds
Cost	lower	higher
Hallucinations	No risk	Some risk

The Embedding + vector search pattern we have looked at with this tutorial provides simple, fast and low cost semantic search functionality with the LLM intelligence. RAG adds context-aware text chat experience and result summarization to it. While RAG provides the more "Gen AI-ish" experience, it also adds a risk of hallucination and higher cost and time for the text generation.

To learn more about how to build a RAG solution, you may look at Building Generative AI applications made easy with Vertex AI PaLM API and LangChain.

Resources

To learn more, please check out the following resources:

Documentations

Vertex AI Embeddings for Text API documentation

Vector Search documentation

Vector Search blog posts

Vertex Matching Engine: Blazing fast and massively scalable nearest neighbor search

Find anything blazingly fast with Google's vector search technology

Enabling real-time AI with Streaming Ingestion in Vertex AI

Mercari leverages Google's vector search technology to create a new marketplace

Recommending news articles using Vertex AI Matching Engine

What is Multimodal Search: "LLMs with vision" change businesses

Utilities

Sometimes it takes tens of minutes to create or deploy Indexes and you would lose connection with the Colab runtime. In that case, instead of creating or deploying new Index again, you can check the Vector Search Console and get the existing ones to continue.

Get an existing Index

To get an Index object that already exists, replace the following `[your-index-id]` with the index ID and run the cell. You can check the ID on the Vector Search Console > INDEXES tab.

In [ ]:
```
my_index_id = "[your-index-id]"  # @param {type:"string"}
my_index = aiplatform.MatchingEngineIndex(my_index_id)
```

Get an existing Index Endpoint

To get an Index Endpoint object that already exists, replace the following `[your-index-endpoint-id]` with the Index Endpoint ID and run the cell. You can check the ID on the Vector Search Console > INDEX ENDPOINTS tab.

In [ ]:
```
my_index_endpoint_id = "[your-index-endpoint-id]"  # @param {type:"string"}
my_index_endpoint = aiplatform.MatchingEngineIndexEndpoint(my_index_endpoint
```