Lab#4, NLP@CGU Spring 2023

This is due on 2023/04/20 16:00, commit to your github as a PDF (lab4.pdf) (File>Print>Save as PDF).

IMPORTANT: After copying this notebook to your Google Drive, please paste a link to it below. To get a publicly-accessible link, hit the *Share* button at the top right, then click "Get shareable link" and copy over the result. If you fail to do this, you will receive no credit for this lab!

LINK: paste your link here

https://colab.research.google.com/drive/1eUImdaTfOVK0krdGgBr8o-X_OXzt9_Sf?usp=sharing

---

Student ID:B0928013

Name:吳佳恩

Word Embeddings for text classification

請訓練一個 kNN或是SVM 分類器來和 Google's Universal Sentence Encoder (a fixed-length 512-dimension embedding) 的分類結果比較

```
!wget  -O  Dcard.db  https://github.com/cjwu/cjwu.github.io/raw/master/courses/nlp2023/lab4-Dcard-Dataset.db
```

```
--2023-04-24 05:36:46--  https://github.com/cjwu/cjwu.github.io/raw/master/courses/nlp2023/lab4-Dcard-Dataset.db
Resolving github.com (github.com)... 140.82.112.4
Connecting to github.com (github.com)|140.82.112.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/cjwu/cjwu.github.io/master/courses/nlp2023/lab4-Dcard-Dataset.db [following]
--2023-04-24 05:36:46--  https://raw.githubusercontent.com/cjwu/cjwu.github.io/master/courses/nlp2023/lab4-Dcard-Dataset.db
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 151552 (148K) [application/octet-stream]
Saving to: ‘Dcard.db’

Dcard.db            100%[===================>] 148.00K  --.-KB/s    in 0.02s

2023-04-24 05:36:47 (6.60 MB/s) - ‘Dcard.db’ saved [151552/151552]
```

```
import  sqlite3
import  pandas  as  pd

conn  =  sqlite3.connect("Dcard.db")
df  =  pd.read_sql("SELECT  *  FROM  Posts;",  conn)
df
```

	createdAt	title	excerpt	categories	topics	forum_en	forum_zh
0	2022-03-04T07:54:19.886Z	專題需要數據🥺🥺幫填～	希望各位能花個20秒幫我填一下			dressup	穿搭
1	2022-03-04T07:42:59.512Z	#詢問 找衣服🥺	想找這套衣服🥺，但發現不知道該用什麼關鍵字找，（圖是草屯囝仔的校園演	詢問	衣服 | 鞋子 | 衣物 | 男生穿搭 | 尋找	dressup	穿搭

```
!pip3 install -q tensorflow_text
!pip3 install -q faiss-cpu
```

```
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.0/6.0 MB 34.9 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 17.0/17.0 MB 76.2 MB/s eta 0:00:00
```

	04T06:39:13.017Z		人知道有類似的嗎		男生穿搭		

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss

embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")
```

355		開了新頻道	線條的動畫影片，新的頻道改成有線條			youtuber	YouTuber

```
docid = 355
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
texts[docid]
```

'[開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑] 昨天上了第一支影片，之前有發過沒有線條的動畫影片，新的頻道改成有線條的，感覺大家好像比較喜歡這種風格，試試看新的風格，影片內容主要是分享自己遇到的小故事，不知道這樣的頻道大家是否會想要看呢？喜歡的話也'

	02T21:25:51.080Z	廻什麼嗎？	基本沒關注過放火，但是最				

```
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values
topk = 10
# Step 1: Change data type
embeddings = embed_arrays.astype("float32")

# Step 2: Instantiate the index using a type of distance, which is L2 here
index = faiss.IndexFlatL2(embeddings.shape[1])

# Step 3: Pass the index to IndexIDMap
index = faiss.IndexIDMap(index)

# Step 4: Add vectors and their IDs
index.add_with_ids(embeddings, index_arrays)

D, I = index.search(np.array([embeddings[docid]]), topk)

plabel = df.iloc[docid]['forum_zh']

cols_to_show = ['title', 'excerpt', 'forum_zh']
plist = df.loc[I.flatten(), cols_to_show]

precision = 0
for index, row in plist.iterrows():
    if plabel == row["forum_zh"]:
        precision += 1

print("precision = ", precision/topk)
precision = 0

df.loc[I.flatten(), cols_to_show]
```

```
precision =  0.8
```

	title	excerpt	forum_zh
355	開了新頻道	昨天上了第一支影片，之前有發過沒有線條的動畫影片，新的頻道改成有線條的，感覺大家好像比較喜歡...	YouTuber
359	一個隨性系YouTube頻道	哈哈哈哈，沒錯我就是親友團來介紹一個我覺得很北七的頻道，現在觀看真的低的可憐，也沒事啦，就多...	YouTuber
330	《庫洛魔法使》（迷你）服裝製作	又來跟大家分享新的作品了~，頻道常常分享 {縫紉} {服裝製作} 等相關教學，大家對服裝製...	YouTuber
342	自己沒搞清楚狀況就不要亂黑勾惡	勾惡幫主在自己頻道簡介跟每部影片的下方都已經說明了，要分會會長以上才能看全部影片，這個說明已...	YouTuber
338	廚師系YouTuber	友人傳了這篇文給我，我一看，十大廚師系YouTuber，就猜一定有MASA，果不其然，榜上有...	YouTuber
243	毀我童年的家人	小時候都很喜歡看真珠美人魚和守護甜心，但是！！，每次晚餐看電視的時候，只要有播映到這種場景....	有趣
349	喜歡看寵物頻道的有嗎？🙋		YouTuber
332	#安利 翎週嗎 采翎	如題啦！ 最近突然超愛采翎，以前就很喜歡了，最近越來越愛~~，從之前的呱張新聞到新資料夾到翎...	YouTuber
340	超像Yoyo的啦～	先說，平常會看見習網美小吳的影片，但我不太會去追蹤YT的IG，然後在IG推薦的影片，就看到y...	YouTuber
263	大家熟悉的梗圖主角 昔與今	先貼幾張大家比較熟的，困惑的表情超傳神 🤣，用在比喻木頭男很適合🤣 跟貓咪一起用超好笑，生無...	有趣

Implemement Your kNN or SVM classifier Here!

請比較分類結果中選出 topk 相近的筆數，並計算 forum_zh 是否都有在 query text 的 forum_zh 中

> [開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑]

```
precision = 0
topk = 10

# YOUR CODE HERE!
# IMPLEMENTIG TRIE IN PYTHON









# # DO NOT MODIFY THE BELOW LINE!
print("precision = ", precision/topk)
```

```
docid = 355
texts = "[" + df['title'] + '] [' + df['topics'] + '] '
texts[docid]
```

```
'[開了新頻道] [Youtuber | 頻道 | 有趣 | 日常 | 搞笑] '
```

```
!pip install pytrie
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pytrie
  Downloading PyTrie-0.4.0.tar.gz (95 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.1/95.1 kB 3.5 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: sortedcontainers in /usr/local/lib/python3.9/dist-packages (from pytrie) (2.4.0)
Building wheels for collected packages: pytrie
  Building wheel for pytrie (setup.py) ... done
  Created wheel for pytrie: filename=PyTrie-0.4.0-py3-none-any.whl size=6104 sha256=a4068ef52cdb69c79423a1be13e846da602b4682a1b4bf61fc23b11d321c
  Stored in directory: /root/.cache/pip/wheels/47/4f/6b/8b01679863e246ca7b7e9e572ba9aef8cd7b76bc3c9fc0b9ba
Successfully built pytrie
Installing collected packages: pytrie
Successfully installed pytrie-0.4.0
```

```
import pandas as pd
import numpy as np
import tensorflow_hub as hub
import tensorflow_text
import faiss
from sklearn.svm import SVC
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

# 讀取資料
df = pd.read_csv("your_data.csv")

# 合併 title, topics, excerpt 為 texts
df['texts'] = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']

# 載入 Universal Sentence Encoder Multilingual 模型
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 將資料轉換為向量
embeddings = np.array(embed_model(df['texts']))

# 將向量規格化
embeddings_norm = embeddings / np.linalg.norm(embeddings, axis=1, keepdims=True)

# 將目標標籤進行 Label Encoding
le = LabelEncoder()
labels = le.fit_transform(df['forum_zh'])

# 切分訓練集和測試集
X_train, X_test, y_train, y_test = train_test_split(embeddings_norm, labels, test_size=0.2, random_state=42)
```

```
# 使用 SVM 建立模型
clf = SVC(kernel='linear', probability=True, random_state=42)

# 訓練模型
clf.fit(X_train, y_train)

# 找出最相似的 topk 筆資料
def find_topk_similar(docid, k):
    # 找出與 docid 最相似的前 k 筆資料
    similarity_scores = cosine_similarity(embeddings_norm[docid].reshape(1,-1), embeddings_norm)[0]
    topk_indices = np.argsort(similarity_scores)[-k-1:-1][::-1]

    # 計算每筆資料的分類
    pred_labels = clf.predict(embeddings_norm[topk_indices])

    # 找出 query 的分類
    query_label = clf.predict(embeddings_norm[docid].reshape(1,-1))[0]

    # 計算 topk 中有多少筆資料屬於 query 的分類
    num_correct = sum(pred_labels == query_label)

    return num_correct, df.iloc[topk_indices]

# 測試
docid = 355
k = 10
num_correct, topk_df = find_topk_similar(docid, k)

print(f"Query 分類: {le.inverse_transform(clf.predict(embeddings_norm[docid].reshape(1,-1))[0]}")
print(f"Top-{k} 相似資料中與 Query 分類相同的筆數: {num_correct}")
print(topk_df[['title', 'excerpt', 'forum_zh']])
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-21-ca4ab9d9be1d> in <cell line: 12>()
     10
     11 # 讀取資料
---> 12 df = pd.read_csv("your_data.csv")
     13
     14 # 合併 title, topics, excerpt 為 texts

6 frames
/usr/local/lib/python3.9/dist-packages/pandas/io/common.py in get_handle(path_or_buf, mode, encoding, compression, memory_map,
is_text, errors, storage_options)
    854         if ioargs.encoding and "b" not in ioargs.mode:
    855             # Encoding
--> 856             handle = open(
    857                 handle,
    858                 ioargs.mode,

FileNotFoundError: [Errno 2] No such file or directory: 'your_data.csv'
```

SEARCH STACK OVERFLOW

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
from sklearn.cluster import KMeans

# 载入 Universal Sentence Encoder
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 将文本转换为向量
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)

# 运行 k-means
num_clusters = 10
kmeans = KMeans(n_clusters=num_clusters, random_state=42).fit(embed_arrays)

# 获取每个文本所属的类别
labels = kmeans.labels_

# 创建一个字典，将每个类别对应的文本索引存储在其中
clusters = {}
for i, label in enumerate(labels):
    if label not in clusters:
        clusters[label] = []
    clusters[label].append(i)
```

```
# 对每个查询进行处理
docid = 355
query = texts[docid]

# 获取与查询相似的文本
query_embedding = embed_model([query])[0]
query_embedding_array = np.array([query_embedding])

# 在 Faiss 中进行相似度搜索
index = faiss.IndexFlatIP(embeddings.shape[1])
index.add(embed_arrays)
D, I = index.search(query_embedding_array, k=10)

# 计算查询的类别
query_label = kmeans.predict(query_embedding_array)[0]

# 获取相似的文本并计算准确率
precision = 0
for i in range(10):
        index = I[0][i]
        label = kmeans.labels_[index]
        if label == query_label:
                precision += 1
print("precision = ", precision/10)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto
  warnings.warn(
precision = 0.9
```

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
from pytrie import SortedStringTrie

embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")
docid = 355
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
texts[docid]
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values
topk = 10

# Step 1: Change data type
embeddings = embed_arrays.astype("float32")

# Step 2: Instantiate the index using a type of distance, which is L2 here
index = faiss.IndexFlatL2(embeddings.shape[1])

# Step 3: Pass the index to IndexIDMap
index = faiss.IndexIDMap(index)

# Step 4: Add vectors and their IDs
index.add_with_ids(embeddings, index_arrays)

D, I = index.search(np.array([embeddings[docid]]), topk)

plabel = df.iloc[docid]['forum_zh']

cols_to_show = ['title', 'excerpt', 'forum_zh']
plist = df.loc[I.flatten(), cols_to_show]

precision = 0
forum_trie = SortedStringTrie()
for index, row in df.iterrows():
    forum_trie[row['forum_zh']] = index

for index, row in plist.iterrows():
    if plabel == row["forum_zh"]:
        precision += 1

query_forum = df.iloc[docid]['forum_zh']
results = {int(forum_trie[k]) for k in forum_trie.iterkeys(prefix=query_forum)}

print("precision = ", precision/topk)
print("Results:", results)
```

```
precision = 0.8
Results: {359}
```

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
from pytrie import SortedStringTrie

# load Universal Sentence Encoder model
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# concatenate title, topics, and excerpt into a single text field
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']

# compute embeddings for all texts using Universal Sentence Encoder
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values

# set the number of top similar documents to retrieve
topk = 10

# Step 1: Change data type
embeddings = embed_arrays.astype("float32")

# Step 2: Instantiate the index using a type of distance, which is L2 here
index = faiss.IndexFlatL2(embeddings.shape[1])

# Step 3: Pass the index to IndexIDMap
index = faiss.IndexIDMap(index)

# Step 4: Add vectors and their IDs
index.add_with_ids(embeddings, index_arrays)

# select a query document
docid = 355
query_text = df.iloc[docid]['forum_zh']

# find the topk most similar documents to the query document using KNN
D, I = index.search(np.array([embeddings[docid]]), topk)

# get the true label of the query document
true_label = df.iloc[docid]['forum_zh']

# create a trie of forum_zh labels
labels_trie = SortedStringTrie({label: True for label in df['forum_zh'].unique()})

# calculate the precision@k by counting the number of true labels in the topk most similar documents
precision = 0
for index, row in df.loc[I.flatten(), ['title', 'excerpt', 'forum_zh']].iterrows():
    label = row['forum_zh']
    if label == true_label:
        precision += 1
    else:
        # check if the query text contains the forum_zh label as a substring using the trie
        if labels_trie.has_subtrie(query_text):
            subtrie = labels_trie.get_subtrie(query_text)
            if subtrie.get(label) is not None:
                precision += 1

print("precision@{} = {}".format(topk, precision/topk))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-19-913bac3db589> in <cell line: 48>()
     52     else:
     53         # check if the query text contains the forum_zh label as a substring using the trie
---> 54         if labels_trie.has_subtrie(query_text):
     55             subtrie = labels_trie.get_subtrie(query_text)
     56             if subtrie.get(label) is not None:

AttributeError: 'SortedStringTrie' object has no attribute 'has_subtrie'
```

SEARCH STACK OVERFLOW

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
```

```
from pytrie import SortedStringTrie

embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")
docid = 355
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
texts[docid]
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values
topk = 10

# Step 1: Change data type
embeddings = embed_arrays.astype("float32")

# Step 2: Instantiate the index using a type of distance, which is L2 here
index = faiss.IndexFlatL2(embeddings.shape[1])

# Step 3: Pass the index to IndexIDMap
index = faiss.IndexIDMap(index)

# Step 4: Add vectors and their IDs
index.add_with_ids(embeddings, index_arrays)

# KNN search
k = 10
D, I = index.search(np.array([embeddings[docid]]), k)

# Step 5: Build a prefix trie to store forum_zh and its index
forum_trie = SortedStringTrie()
for index, row in df.iterrows():
    forum_trie[row['forum_zh']] = index

# Step 6: Find the precision and query results
precision = 0
query_forum = df.iloc[docid]['forum_zh']
results = set()
for i in range(k):
    forum_zh = df.iloc[I[0][i]]['forum_zh']
    if query_forum == forum_zh:
        precision += 1
    if forum_zh.startswith(query_forum):
        results.add(int(forum_trie[forum_zh]))

print("precision =", precision / k)
print("Results:", results)
```

```
precision = 0.8
Results: {359}
```

```
topk = 10    # 設定 topk 相近的筆數

# 計算 topk 的相似文檔
D, I = index.search(np.array([embeddings[docid]]), topk)

# 獲取 query text 的 forum_zh
plabel = df.iloc[docid]['forum_zh']

# 從 DataFrame 中選出相似文檔
cols_to_show = ['title', 'excerpt', 'forum_zh']
plist = df.loc[I.flatten(), cols_to_show]

# 計算在 topk 文檔中，有多少文檔的 forum_zh 在 query text 的 forum_zh 中
precision = 0
for index, row in plist.iterrows():
        if plabel == row["forum_zh"]:
                precision += 1

# 計算 topk 相近的筆數
num_neighbors = len(I.flatten())

print("precision = ", precision/topk)
print("num_neighbors = ", num_neighbors)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-7-377555a43b38> in <cell line: 4>()
      2
      3 # 計算 topk 的相似文檔
----> 4 D, I = index.search(np.array([embeddings[docid]]), topk)
```

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss

# 載入 Universal Sentence Encoder 多語言模型
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 計算文檔嵌入向量
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
embeddings = embed_model(texts)
embed_arrays = np.array(embeddings)
index_arrays = df.index.values

# 設定搜索參數
docid = 355   # 查詢文本的索引
topk = 10      # 選擇相似文檔的數量

# 將嵌入向量轉換為 float32 型別
embeddings = embed_arrays.astype("float32")

# 實例化 Faiss 索引
index_flat = faiss.IndexFlatL2(embeddings.shape[1])
index = faiss.IndexIDMap(index_flat)

# 添加向量和 ID
index.add_with_ids(embeddings, index_arrays)

# 查找最相似的文檔
D, I = index.search(np.array([embeddings[docid]]), topk)

# 獲取查詢文檔的 forum_zh
plabel = df.iloc[docid]['forum_zh']

# 從 DataFrame 中選出相似文檔
cols_to_show = ['title', 'excerpt', 'forum_zh']
plist = df.loc[I.flatten(), cols_to_show]

# 計算在 topk 文檔中，有多少文檔的 forum_zh 在 query text 的 forum_zh 中
precision = 0
for index, row in plist.iterrows():
    if plabel == row["forum_zh"]:
        precision += 1

# 計算 topk 相近的筆數
num_neighbors = len(I.flatten())

print("precision = ", precision/topk)
print("num_neighbors = ", num_neighbors)
```

```
precision =  0.8
num_neighbors =  10
```

	title	excerpt	forum_zh
355	開了新頻道	昨天上了第一支影片，之前有發過沒有線條的動畫影片，新的頻道改成有線條的，感覺大家好像比較喜歡...	YouTuber
359	一個隨性系YouTube頻道	哈哈哈哈，沒錯我就是親友團來介紹一個我覺得很北七的頻道，現在觀看真的低的可憐，也沒事啦，就多...	YouTuber
330	《庫洛魔法使》（迷你）服裝製作	又來跟大家分享新的作品了~，頻道常常分享 {縫紉} {服裝製作} 等相關教學，大家對服裝製...	YouTuber
342	自己沒搞清楚狀況就不要亂黑勾惡	勾惡幫主在自己頻道簡介跟每部影片的下方都已經說明了，要分會會長以上才能看全部影片，這個說明已...	YouTuber
338	廚師系YouTuber	友人傳了這篇文給我，我一看，十大廚師系YouTuber，就猜一定有MASA，果不其然，榜上有...	YouTuber
243	毀我童年的家人	小時候都很喜歡看真珠美人魚和守護甜心，但是！！，每次晚餐看電視的時候，只要有播映到這種場景....	有趣
349	喜歡看寵物頻道的有嗎？🙋‍♂️		YouTuber
332	#安利 翎週嗎 采翎	如題啦！ 最近突然超愛采翎，以前就很喜歡了，最近越來越愛~~，從之前的呱張新聞到新資料夾到翎...	YouTuber
340	超像Yoyo的啦～	先說，平常會看見習網美小吳的影片，但我不太會去追蹤YT的IG，然後在IG推薦的影片，就看到y...	YouTuber
263	大家熟悉的梗圖主角 昔與今	先貼幾張大家比較熟的，困惑的表情超傳神 🤣，用在比喻木頭男很適合🤣 跟貓咪一起用超好笑，生無...	有趣

```
import tensorflow_hub as hub
import numpy as np
import tensorflow_text
import faiss
```

```
# 載入文本嵌入模型
embed_model = hub.load("https://tfhub.dev/google/universal-sentence-encoder-multilingual/3")

# 創建文本嵌入向量
texts = "[" + df['title'] + '] [' + df['topics'] + '] ' + df['excerpt']
embeddings = embed_model(texts).numpy()

# 將文本嵌入向量轉換為浮點數陣列
embeddings = embeddings.astype("float32")

# 獲取文本數據的索引陣列
index_arrays = df.index.values

# 指定需要返回的前 k 個最相似的文本
topk = 10

# 實例化 Faiss 搜索索引並添加文本向量和對應的 ID
index = faiss.IndexFlatL2(embeddings.shape[1])
index = faiss.IndexIDMap(index)
index.add_with_ids(embeddings, index_arrays)

# 指定要計算相似度的文本的 ID
docid = 355

# 使用 Faiss 索引查找前 k 個最相似的文本的索引和距離
D, I = index.search(np.array([embeddings[docid]]), topk)

# 獲取查詢文本的論壇標籤
plabel = df.iloc[docid]['forum_zh']

# 指定要顯示的列
cols_to_show = ['title', 'excerpt', 'forum_zh']

# 獲取前 k 個最相似的文本的標題、內容和論壇標籤
plist = df.loc[I.flatten(), cols_to_show]

# 計算在前 k 個最相似的文本中，有多少文本的論壇標籤和查詢文本的論壇標籤相同
precision = 0
for index, row in plist.iterrows():
    if plabel == row["forum_zh"]:
        precision += 1

# 計算 precision
precision = precision / topk

# 在控制台中打印 precision
print("precision = ", precision)
```

```
precision = 0.8
```


0 秒 完成時間：下午3:23