Method

- How you implement your indexes
  1. Kmean
     甲、資料結構:

     基於 HashIndex 製作 IvfflatIndex，主要維護兩種 table。

     index table:

     儲存每一個 cluster 的代表 vector 和 cluster 的 id。

     cluster table:

     儲存 cluster 中所有的 vector 值，和在原 table 的 record id，每個在 index table 的 id 都對應一個 cluster。

     乙、存取:

     insert:

     先從 index table 找到離要插入的 vector 最近的代表 vector，用 id 找到 cluster table 後插入。

     knnsort:

     先對 index table 做 sort，找到最近的代表 vector 的 cluster table，接著只對 cluster table 做 sort，不理其他 cluster。

     丙、訓練:

     初始化:

     讀取所有資料，讓每個 cluster table 依 clock 方式輪流 insert vector。

     優化:

     每次優化時，把 cluster table 的所以有 vector 取平均，修改 index table 的代表 vector。接著重新用存取中的 insert 方法放入所有 vector。重複直到時間到。
  2. Balance Kmean
     甲、理由:

     由於只對一個 cluster table 做 sort，為防止找不到足夠數量，也為了平衡每次 query 時間和 recall，也對之後找最佳解較不依靠運氣，想辦法平衡每個 cluster table 的 record 數量。

     乙、方法:

     在訓練初始化時取得每個 cluster table 的平均 record 數量，當優化

算平均時，如果有 cluster 超過平均數量，將超出的 vector 額外統計平均，並指定給數量太少的 cluster 當作其代表 vector。讓有更多 cluster 分擔數量。

3. Sample Training

甲、理由:

訓練時間不夠，所以利用 sample 小部分來訓練，依此加速。

乙、方法:

在初始化和優化時，每筆 vector 資料會按照一定機率被忽略，達到隨機的效果。

- How you implement SIMD

首先，初始化了一個長度為 256 位元的 VectorSpecies 對象，這指定了在 SIMD 操作中使用的向量長度，並使用迴圈按照向量長度處理向量的區塊。迴圈中，將需要計算距離的兩個向量加載到 SIMD 中的 FloatVector，再計算兩個 FloatVector 之間元素的差異平方，並儲存於一個陣列當中。最後再把陣列中的差異平方做加總且開平方，即使用 SIMD 完成計算兩個向量之間的歐幾里得距離。

- Other improvements you made to speed up the search

1. Sort plan 改成 Priority queue

對 cluster table 不用 sort，用在 executeCalculateRecall 裡的方法只記錄前 20 名，不用像 sort 和硬碟溝通。

Experiments

- Experiment environment
  - Intel Core i7-13700 @ 2.10 GHz, 32GB RAM, 512 GB SSD, Windows 11
- Benchmark parameters

對 Bucket number 和 Sample rate 進行參數分析，以找到最佳參數。下表為固定 Bucket number 對不同 Sample rate 之性能表現：

Bucket number	125		
Sample rate	0.25	0.5	0.75
Recall	0.5773	0.5804	0.5610
Throughput	657	619	622
Recall * Throughput	379.2861	359.2676	348.942

下表為固定 Sample rate 對不同 Bucket number 之性能表現：

Bucket number	32	64	125	250	500
Sample rate	0.5				
Recall	0.7111	0.6448	0.5804	0.5103	0.4611
Throughput	445	553	619	701	753
Recall * Throughput	316.4395	356.5744	359.2676	357.7203	347.2083

- Analysis on the results of the experiments

  從實驗結果可以看出，在固定 Bucket number 的情況下，Sample rate 設定為 0.25 會有較佳之表現，且隨著 Sample rate 增大，性能逐漸降低。在固定 Sample rate 的情況下，Bucket number 設定為 0.5 會有較佳之表現。而 Bucket number 設定為 125 且 Sample rate 設定為 0.25 是最佳之參數。