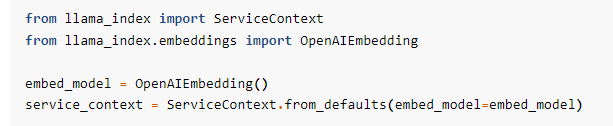
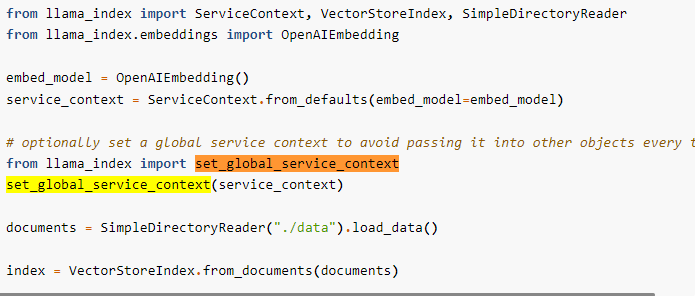
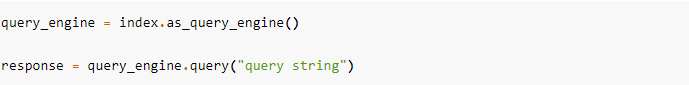
利用openai进行embeding



利用openai进行embeding，设置全局，这样子就不用了每次都传入模型了

https://gpt-index.readthedocs.io/en/stable/core\_modules/model\_modules/embeddings/usage\_pattern.html





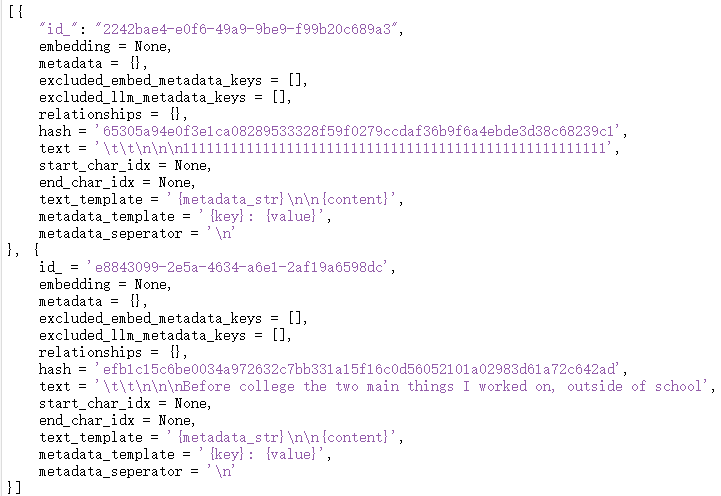
<https://github.com/jerryjliu/llama_index/blob/main/docs/examples/metadata_extraction/MetadataExtractionSEC.ipynb>

这篇文章旨在解决当我们有两篇相似的文章，可能导致提取了错误的文档的情况，举例了两家衣服销售商的财务报表，财务报表的内容有很高的相似的，a生产商和b生产商，我们要提取关键词去区分哪些片段是a生产商的文档片段，哪些是b生产商的文档片段。

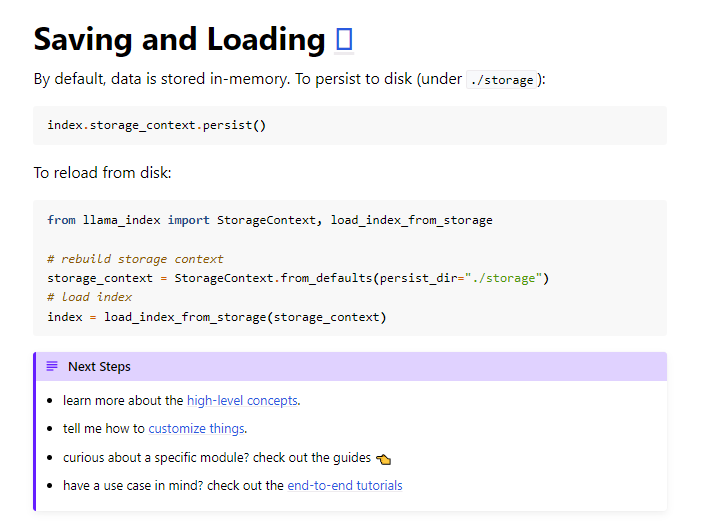
策略：

1. 获取文档的前半段然后总结，然后提取关键词。在后续文档拆分的时候给每一个片段都加上总结和关键词。
2. 每一个片段都进行总结以及提取关键词，然后将所有的关键词汇总，出现次数较高的关键词作为每个片段的前置词汇。

Llama\_index的api可以读取文件夹下的所有的文本文件，读取出来的格式是这个样子，下面是两个文本文件，一个文本文件对应一个文档



如何加载缓存的index的文件

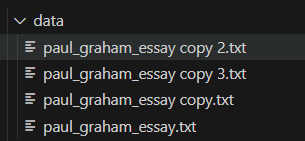


这里有很多index介绍

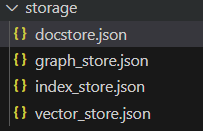
<https://gpt-index.readthedocs.io/en/stable/deprecated_terms.html#gptvectorstoreindex>

# Embedding文件分析

我们的目录下面一共有四个文档



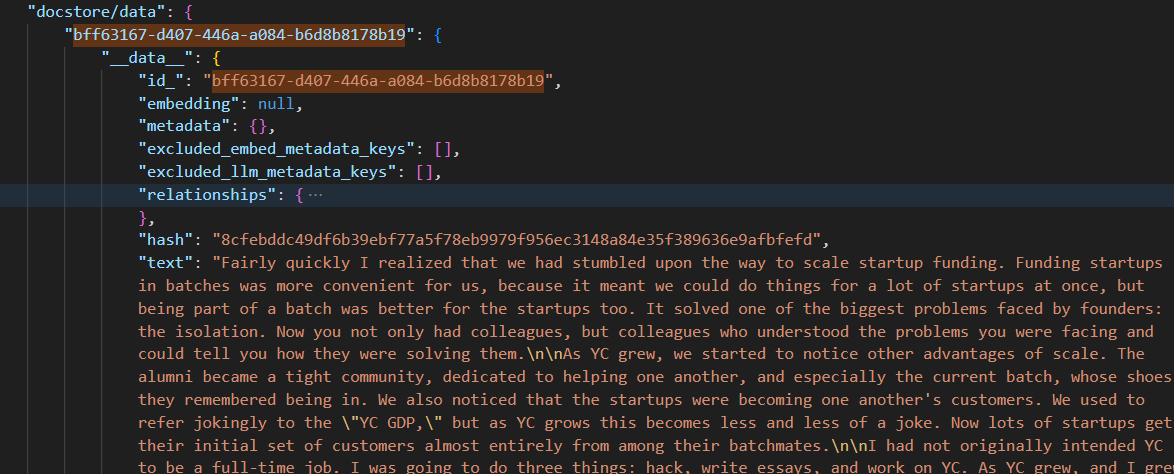
生成的文件



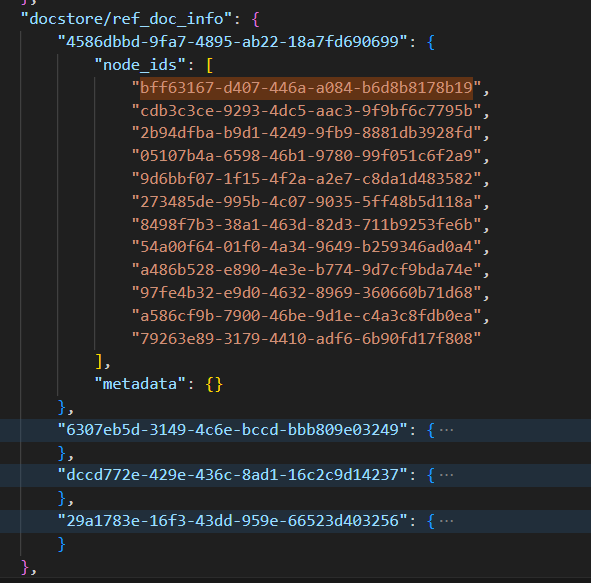
## Docstore.json节点分析，这里记录了节点和文档的关系

Node节点代表一个分割出来的文本段落。

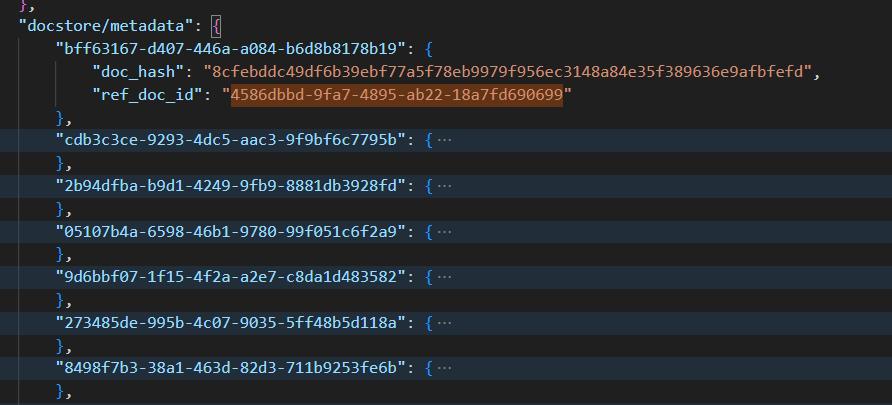
下面代表其中一个node节点



下面外部有四个文档数字，里面的node\_ids代表该文档被分割成的node节点

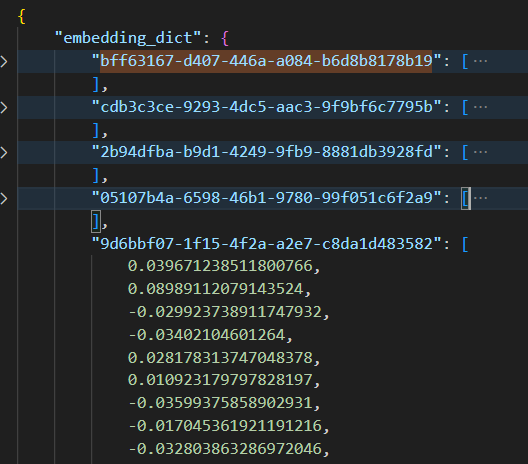


下面代表所有文档的所有node节点所占文本的哈希，ref\_doc\_id表示它属于哪个文档，这里是不按照文档进行分类了，不同的文档的node节点混在一起。

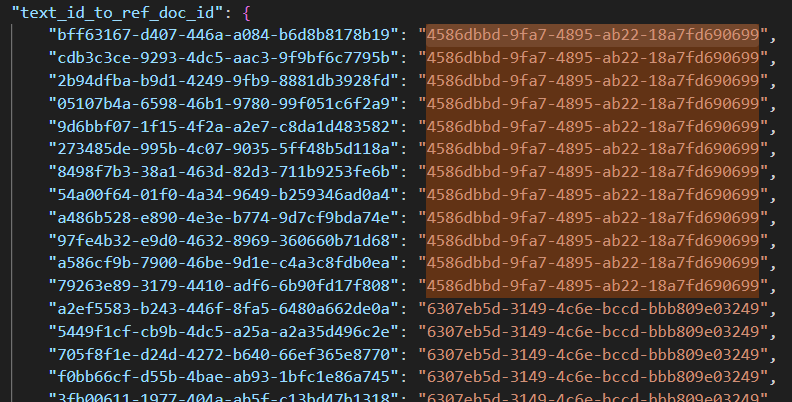


## vector\_store.json

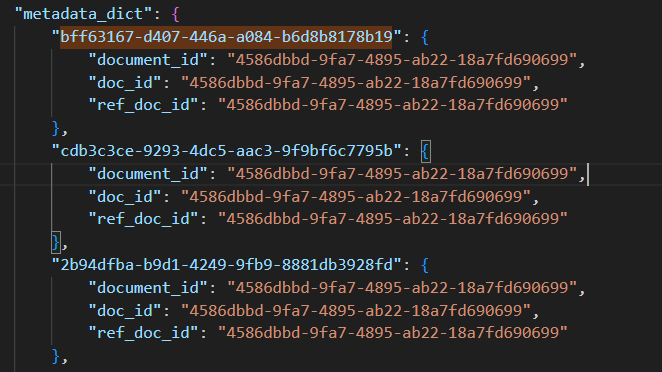
记录了每个node节点文本的向量数组



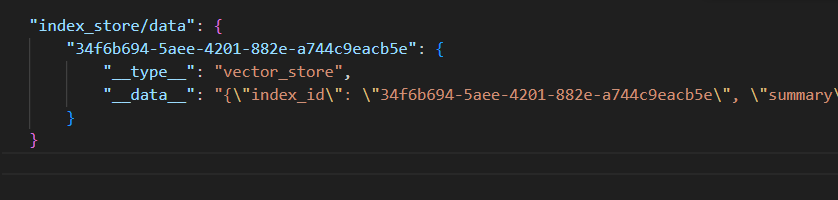
左边是node节点id，右边是归属的哪一个文档id



Node节点id对象里面三个属性值都代表文档id，就是不知道为什么用三个属性名记录一个值。



## index\_store.json



里面的字符展开如下

Node\_dict里面都是node节点id，不知道为什么要这样子记录，左边是属性，右边是属性的字符形式

