Inverted-index-walkthrough

September 3, 2020

In this example, instead of actual terms, we use letters to represent any arbitrary unique term.

```
In [1]: # Task: construct an inverted index from this
        # For simplicity: lets pretend that these letters are terms
        corpus = [
            "a a b c", # id 0
            "a b c d c", # id 1
            "b a a b d", # id 2
            "c c c b e f g" # id 3
        ]
```

```
In [2]: # We can use enumerate to see the ids with the corresponding documents
        for doc_id, document in enumerate(corpus):
            print(f"document {doc_id}: {document}")
```

```
document 0: a a b c
document 1: a b c d c
document 2: b a a b d
document 3: c c c b e f g
```

What we want to create from this corpus:
Assign each term a posting list of document ids

```
In [3]: # An inverted index could look like this:

        sample_index = {
            "a": [0, 1, 2],
            "b": [0, 1, 2, 3],
            "c": [0, 1, 3],
            "d": [1, 2],
            "e": [3],
            "f": [3],
            "g": [3]
        }
```

```
In [4]: # print every document containing c:
        for i in sample_index["c"]:
            print("Document", i, ": ", corpus[i])
```

```
Document 0 :  a a b c
Document 1 :  a b c d c
Document 3 :  c c c b e f g
```

How to build the index programatically

```
In [5]: index = {}
        # Code here
        for doc_id, document in enumerate(corpus):
            for token in set(document.split()):
                if token not in index:
                    index[token] = [doc_id]
                else:
                    index[token].append(doc_id)
        index

Out[5]: {'b': [0, 1, 2, 3],
         'a': [0, 1, 2],
         'c': [0, 1, 3],
         'd': [1, 2],
         'f': [3],
         'g': [3],
         'e': [3]}
```

Here is another way to do it

```
In [6]: terms = set(" ".join(corpus).split())
        terms

Out[6]: {'a', 'b', 'c', 'd', 'e', 'f', 'g'}

In [7]: index = {t: [i for i in range(len(corpus)) if t in corpus[i]] for t in terms}
        index

Out[7]: {'c': [0, 1, 3],
         'a': [0, 1, 2],
         'f': [3],
         'd': [1, 2],
         'g': [3],
         'b': [0, 1, 2, 3],
         'e': [3]}
```

0.1 Note:

in the mandatory assignment: * The inverted index is not implemented as a dictionary, like in this notebook * The postings are objects containing both document ids and term frequencies, instead of single integers denoting only document ids * The documents are divided into fields and have their own class