cje3.py

```python
# cos類似度から、類似文書を表示する。知識情報演習3で作成。


from IPython.terminal.prompts import Token
import math
import pandas as pd
import re
from janome.tokenizer import Tokenizer

t = Tokenizer()

index_file = "index/index_nostop.txt"
idf_scores = {}
tfidf_scores = {}
query_tf = {}
query_tfidf = {}
stopwords = {}
query_words = {}
ranking_docs = {}

f = open(index_file, 'r')

# indexから文書のデータフレームを作成 ----------------
for line in f:
    line = line.rstrip()
    split_line = line.split('\t')
    word = split_line[0]
    doc = split_line[1]
    idf = float(split_line[2])
    tfidf = float(split_line[3])

    if word not in tfidf_scores:
        tfidf_scores[word] = {}
        tfidf_scores[word][doc] = tfidf
    else:
        tfidf_scores[word][doc] = tfidf

    if word not in idf_scores:
        idf_scores[word] = {}
        idf_scores[word] = idf
    else:
        idf_scores[word] = idf

tfidf_table = pd.DataFrame(tfidf_scores)
tfidf_table = tfidf_table.fillna(0)


# クエリのデータフレームを作成 ----------------------
#query = 'モラント 心配'
#query = 'ウォリアーズ 放出'
#query = 'プレイオフ ファイナル'
#query = 'W杯 日本'
query = 'オフシーズン ドラフト'

query_file = 'query'
argv = query.split()
```

```
argc = len(argv)
# クエリ表示
print(argv)

pattern = re.compile(r"^[ -—]$")
stopwords['という'] = 1
stopwords['にて'] = 1

tokens = []
for i in range(argc):
    list = t.tokenize(argv[i])
    tokens.append(list)
flat_t = [item for sublist in tokens for item in sublist]

flat_token = []
for token in flat_t:
    tmp = token.surface
    judge = pattern.match(tmp)
    flat_token.append(tmp)

    if judge:
        continue
    if tmp in stopwords:
        continue

    if tmp in query_words:
        query_words[tmp] += 1
    else:
        query_words[tmp] = 1


for index_word in idf_scores:
    query_tf[index_word] = {}
    query_tfidf[index_word] = {}
    query_tf[index_word][query_file] = 0
    query_tfidf[index_word][query_file] = 0

for query_word in query_words:
    if query_word in query_tf:
        query_tf[query_word][query_file] += 1

for query_word in query_words:
    if query_word in query_tf:
        query_tfidf[query_word][query_file] = float(query_tf[query_word][query_file]) * float(idf_scores[query_word])

query_table = pd.DataFrame(query_tfidf)


# 各データフレームからcos類似度を計算 ------------------
for query_word in query_words:
    if query_word in tfidf_scores:
        for doc in tfidf_scores[query_word]:
            ranking_docs[doc] = 1

for doc in ranking_docs:
    numerator = 0
    doc_vec = tfidf_table.loc[doc]
    query_vec = query_table.loc[query_file]
    for i in range(len(query_vec.values)):
```

```
116         i_value = query_vec.values[i] * doc_vec.values[i]
117         numerator = numerator + i_value
118 
119     denominator = 0
120     query_value = 0
121     doc_value = 0
122     for i in range(len(query_vec.values)):
123         query_value += query_vec.values[i]**2
124         doc_value += doc_vec.values[i]**2
125 
126     denominator = math.sqrt(query_value) * math.sqrt(doc_value)
127     cosine = numerator / denominator
128 
129     ranking_docs[doc] = cosine
130 
131 for i in sorted(ranking_docs.items(), key=lambda x:x[1], reverse=True):
132     print('cos類似度:',i)
```