這次作業總共測試了152次改了很多個版本，但是前面沒有發現有description，所以沒有紀錄
這次修改模型的順序為
1. HW1的td,idf寫法+BM25公式(最高0.78上下)
2.修改TF程式碼改成更快速

```
def tf(dict1):
    td={}
    for word in dict1:
        td[word]=dict1.count(word)/len(dict1)
    return td
```

3.修改BM25改成BM25L(突破0.85):把IDF
idf = math.log((5000+DF[i]) / (DF[i] + 0.5))改成
idf = math.log((5000+1) / (DF[i] + 0.5))
4.發現K3好像沒有那麼重要(調多少都沒有影響結果)把BM25L有K3的部分拿掉但分數上不去
5.參考https://www.cnblogs.com/geeks-reign/p/Okapi_BM25.html
改成BM25+公式，最高0.84多也上不去
6.最後改回BM25L並將K3拿掉，由於b的影響也沒有很大最後b設置為0
7.最終參數k1=0.0001,k3=1,b=0

```
def bm25(s1_cut,s2_cut,DF,co,k1,k3,b,dk,avg_doclen):
    wSum=0
    for i in s1_cut:
        try:
            tda=k3
            tfverse=s2_cut[i]/(1+(b * (dk/avg_doclen)))
            upper1 = (k1 + 1) * s2_cut[i]
            down1 = (s2_cut[i] + k1 * ((1 - b)+ b * dk / avg_doclen))
            idf = math.log((5000+1) / (DF[i] + 0.5))
            wSum += (((upper1 / down1))  * idf)
        except:
            wSum +=0

    if co%1000==0:
        gc.collect()
    return wSum


def tf(dict1):
    td={}
    for word in dict1:
        td[word]=dict1.count(word)/len(dict1)
    return td
```

本次bm25L程式碼