DESIGN DOCUMENTATION

This search engine helps you find page number of a particular dialogue from the book The Time Machine. Here each page of the book is taken a document in the corpus. There is a total of 156 documents in the corpus. Here cosine of tf-idf values are used to give scores. Top 10 documents with the highest scores will be given as output.

This program is divided into 6 subparts:

1. **tokenisation.py –** This program will tokenise all the documents in the corpus using inbuilt nltk tokenizer which are stored in a list.
2. **inverted_index.py –** An inverted index is built in this part of code.
3. **words.py** – Here this code creates a vocabulary for the corpus where every unique word in the corpus is given an id.
4. **tf-idf_values.py –** This part of the program generates tf-idf value of every word in the vocabulary with respect to all documents.
5. **scoring.py –** It gives score for all the documents based on the given query.
6. **trail.py –** it is basically a control for the scoring.py which takes input and passes it to it. And prints the output.

Database for this program has been stored in 5 json files (stored in savers):

1. **ii.json** – Stores the inverted index that is created by inverted_index.py. Here it is stored as a dictionary where words in the vocabulary are stored as key and value of the key is a list containing all the documents in which the key has occurred.
2. **tokens.json** – Stores the tokens (normalized words) of every document created by tokenisation.py . All the tokens are stored in a list
3. **words.json –** stores the vocabulary of the corpus with a id given to every word in the vocabulary in the form of a dictionary.
4. **dictionary.json –** stores tdf-idf value of every word in vocabulary corresponding to all the documents in a dictionary where key is a word in the vocabulary and its value is a dictionary which has document id as key and its value is a dictionary which contains tf, idf, tf*idf values with keys 1,2,3 respectively.
   Ex:{‘Time’:{‘0’: {‘1’:1 ,“1”:0.8, ‘3’ : 0.8} , ‘1’:{‘1’: 2 ,‘2’: 0.4, ‘3’:0.8}, ‘2’:{‘1’: 0 ,‘2’: 0.3,‘3’:0}} ,
   ‘Traveller’ :{‘0’: {‘1’:2 ,“2”:0.6, ‘3’ :1.2} , ‘1’:{‘1’: 0 ,‘2’: 0.4, ‘3’:0}, ‘2’:{ ‘1’: 1, ‘2’: 0.4, ‘3’:0.4}}}
5. **scores.json –** scores of all the documents will be stored corresponding to a query.

All the documents are stored in the corpus folder inside the assignment.