Module **corpus**

Functions

def BuildCorpusList(fileDf: pandas.core.frame.DataFrame, corpusIndexPathDict)

Description

Build the list of all file paths (split into parts) to use when constructing the corpus.

File paths are split into parts using the Path.parts attribute. The filename is further split into individual parts on delimiters such as _ , + & ; and CamelCase. All parts are then lowered for consistency.

Singular occuring tokens are ignored from the corpus, as well as purely numerical strings (excluding four digit strings which may correspond to year dates).

Each document (split file path) in the corpus is then stored in the corpus list as a list of token strings, in addition to a separate list containing the bag of words (bow) vector representation.

The dict corpusIndexPathDict stores the index (in the corpus list) -> file path

Args:

```
fileDf : pd.DataFrame
    #: The file dataframe to use to construct the corpus
corpusIndexPathDict : dict[int] = Path
    #: Dict to store list index -> file path
```

Outputs:

```
corpus : list[list[str]]
    #: The list of corpus documents
dictionary : corpora.dictionary
    #: The gensim corpus dictionary
bowVectors : list[(int, ... , int)]
    #: Bag of word (bow) vector representation of each document
```

def BuildLSIAfterTfidfModel(corpus: FileCorpus, numTopics=200)

Description

Trains an LSI model after the TfIdf model transformation has been applied.

Returns the model transformation and the similarity matrix index

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the model
```

Kwargs:

```
numTopics : int : default=200
    #: Number of topics used for the LSI decomposition
```

Outputs:

```
lsiModel : models.LsiModel
    #: LSI model transformation
index : similarities.Similarity
    #: Similarity matrix for the LSI model
```

def BuildLSIModel(corpus: FileCorpus, numTopics=200)

Description

Trains an LSI model with the specified number of topics.

Returns the model transformation and the similarity matrix index

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the model
```

Kwargs:

```
numTopics : int : default=200
    #: Number of topics used for the LSI decomposition
```

Outputs:

```
lsiModel : models.LsiModel
    #: LSI model transformation
index : similarities.Similarity
    #: Similarity matrix for the LSI model
```

def BuildPhraseModel(corpus: FileCorpus)

Description

Builds a phrase model and uses it to train a Word2Vec model.

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the model
```

Outputs:

```
phraseModel : models.Phrases
    #: Phrase model transformation
phraseVecModel : models.Word2Vec
    #: Word2Vec model on the transformed phrase corpus
```

def BuildTfIdfModel(corpus: FileCorpus)

Description

Trains a term frequency / inverse document frequency model.

Returns the model transformation and the similarity matrix index

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the model
```

Outputs:

```
tfidfModel : models.TfidfModel
    #: TfIdf model transformation
index : similarities.SparseMatrixSimilarity
    #: Similarity matrix for the TfIdf model
```

def BuildWord2VecModel(corpus: FileCorpus)

Description

Trains a Word2Vec model. Similarity method is contained within the w2vModel.wv wordvector attribute wv.most_similar.

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the model
```

Outputs:

```
w2vModel : models.Word2Vec
    #: LSI model transformation
```

def DocSimilarityQuery(queryBow, model, index, corpus: FileCorpus)

Description

Perform a similarity query from the specified model, using the query bow vector.

Returns a sorted list of scores of the form (similar document, score). The similar documents are converted from their bow vector representation back to the original file path using the corpusListIndexPathDict.

Args:

```
queryBow : list[(int, int)]
    #: bow vector representation of the query
model : models. ..
    #: Model to query
index : similarities.Similarity
    #: Similarity matrix of the model
corpus: FileCorpus
    #: File corpus used to train the model
```

Outputs:

```
documentScores : list[(Path, score)]
    #: Scores for each similar document (file path)
```

def RemoveNumericalStrings(stringList, includeYears=True)

Description

Removes any purely numerical strings from a list of strings. Ignores four character strings if includeYears=True.

Args:

```
stringList : list[str]
    #: String list to be processed
```

Kwargs:

```
includeYears : bool : default=True
    #: If true, ignore four character numerical strings that may correspond to a year.
```

Outputs:

```
characterStrings : list[str]
    #: The strings with purely numerical strings removed
```

def RemoveSingularOccurences(corpus)

Description

Removes any tokens that only occur once throughout the corpus.

Args:

```
corpus : list[list[str]]
    #: The list of corpus documents
```

Outputs:

```
processedCorpus : list[list[str]]
    #: The list of corpus documents with singular occurences removed
```

def SplitAndLowerString(string)

Description

Splits a string on delimiters and CamelCase and then lowers the string, returning the list of tokens from the string.

Args:

```
string : str
    #: String to be processed
```

Outputs:

```
processed : list[str]
    #: The list of tokens from the string
```

def SplitCamelCase(string)

Description

Splits a string on CamelCase

Args:

```
string : str
    #: String to be processed
```

Outputs:

```
words : list[str]
    #: The split string
```

def SplitOnDelimiters(string)

Description

Splits a string on delimiters

Args:

```
string : str
    #: String to be processed
```

Outputs:

```
words : list[str]
    #: The split string
```

def TrainModels(corpus: FileCorpus, models: ModelCollection)

Description

Trains a variety of gensim models on the specified FileCorpus and stores the trained models in the ModelCollection object.

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the models
models : ModelCollection
    #: To store the models
```

def Word2VecQuery(queryBow, w2vModel, corpus)

Description

Perform a similarity query from the word2Vec model, using the query bow vector.

Returns a dict of token -> similar tokens

Args:

```
queryBow : list[(int, int)]
    #: bow vector representation of the query
w2vModel : models.Word2Vec
    #: Model to query
corpus: FileCorpus
    #: File corpus used to train the model
```

Outputs:

```
partSimilarTokensDict : dict[str] =
    #: Dict with token -> similar tokens
```

Classes

class FileCorpus(fileDf)

Class:

Class that holds the corpus consisting of the list of all file paths, where each document in the corpus is an individual file path (split into parts).

`class ModelCollection (corpus: FileCorpus)`

Class:

Class that holds the various gensim models trained on the file corpus. Models are trained on construction.

Args:

```
corpus : FileCorpus
    #: The file corpus on which to train the models
```

`class ModelQuery (corpus: FileCorpus, models: ModelCollection, queryStr: str)`

Class:

Class that stores the result of querying multiple models stored in the ModelCollection.

The queryStr can contain multiple tokens and is processed into a tokenised document similarly to how the file paths were processed.

Index

Super-module

fileorganiser

Functions

Classes