Music evokes autobiographical memories pipeline doc.

Dependencies: numpy, pandas, umap, sentence-transformers (SBERT), sklearn (sci-kit learn), skfuzzy

To download dependencies:

pip install pandas

pip install -U sentence-transformers

pip install -U scikit-learn (installs numpy)

pip install -U scikit-fuzzy

pip install umap==0.5.2

**Mpipe:**

The Mpipe object can be instantiated as follows:

Pipe = Mpipe(root, data\_file, output, custom\_fname , step\_attributes, safe\_run, verbose)

If the given output directory does not exist, a new one will be created with the name given.

Where:

root = path (string) to root working directory folder

data\_file = path (string) to file containing ONLY data to be analysed (.csv)

output = name (string) of folder (in root dir) for pipeline outputs (defaults to root dir)

custom\_fname = custom filename (string) beginning for saved outputs (default = “”)

*Optional (recommended):*

safe\_run = if true check pipeline step function has expected attributes before assigning them (default=True)

verbose = if true pipeline object will announce each step and the type/shape of data being passed through it. (default = True)

*Optional:*

step\_attributes = list of dictionaries containing parameters (arguments) to pass to step functions (default = [], empty list)

*Note. Only needed if execute function will be used*

**Execute:**

execute(pipe):

Runs pipeline objects and assigns step\_attributes at the same index to their functions

* saves the output of every step taken, with a filename that shows which steps were taken (and which order)
* Does not check whether transformations are repeated (make sure to check which steps are before others)

**Embedding:**

SBERT(text\_data, params=None):

self.stepadd="-SBERT"

Sentence Bert Sentence Transformer encoding

IN: dataframe containing text data

* **will vertically concatenate columns if cols > 1 and drops ALL rows containing NA**

OUT: data frame of shape (n\_observations, n\_features)

info: <https://sbert.net/docs/package_reference/sentence_transformer/SentenceTransformer.html>

"""

OPENAI : returns Open AI GPT word embeddings

**Dimensionality reduction:**

UMAP\_reduce(data, params=None):

self.stepadd="-UMAP"

UMAP dimentionality reduction

IN: dataframe containing only values to be reduced

* Scales data before transforming (sklearn - StandardScaler)

OUT: umap dimensional values (n\_observations, n\_features)

info:

<https://umap-learn.readthedocs.io/en/latest/parameters.html>

PCA : returns n principle components

**Clustering:**

KMEANS(data, params=None):

self.stepadd="-KMEANS"

KMEANS clustering algorithm

IN: pandas dataframe containing data to be clustered

* Scales data before transforming (sklearn - StandardScaler)

OUT:cluster labels (n\_observations, labels)

info: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

FUZZYCMEANS(data, params=None, score = False):

        """ FUZZY C Means (skfuzzy)

        IN: 2D array (n\_observations, n\_features)

        OUT: cluster membership (proportions) shape = (memberships, n\_observations), score (float)

        INFO: <https://scikit-fuzzy.github.io/scikit-fuzzy/auto_examples/plot_cmeans.html#example-plot-cmeans-py>

        """

Params : arguments to pass to fuzzy c means cluster function

Score = (True/False). Returns score of clustering if true.