addr API Reference

This is the function reference for addr.

Contents

1. **get_vecs**
   - get_files()
   - make_agg_vec()
   - doc_vecs_from_csv()
   - doc_vecs_from_txt()
   - terms_from_txt()
   - terms_from_liwc()
   - terms_from_csv()
   - terms_to_csv()
   - dic_vecs()
   - write_dic_vecs()
2. **get_loadings**
   - load_model()
   - get_loadings()
3. **file_length**
   - file_len()
4. **simple_progress_bar**
   - update_progress()

---

1. get_vecs

- **get_files(** input_path **)**

This function is used to store file names and paths for a single file or (recursively) for a directory of files in a Python dictionary object

Parameters:

input_path : string

Path to file or directory.

Returns:

path_info : Python dictionary object

Python dictionary object with file names as keys and corresponding file paths as values.

- **make_agg_vec(** words, model, num_features, model_word_set, filter_out = [] **)**

This function queries vector representations for each word in a list of words and returns the average of these vectors.

Parameters:

words : list

List of words that should be included in the returned vector representation.

model : Word2Vec model

Gensim.Word2Vec model containing vector representations to be used for aggregate vectors.

num_features : int/float

Dimensionality of the Word2Vec model being used.

model_word_set : list

List of words in vocabulary of Word2Vec model.

filter_out : list

Optional list of words to exclude from aggregation process.

Returns:

path_info : Numpy array object

Array containing the aggregate vector representation of the input words.

- **doc_vecs_from_csv(** input_path, output_path, model, num_features, model_word_set, text_col, filter_out = [], delimiter = “\t”, id_col = False **)**

This function reads a CSV file with documents as rows and generates a CSV file containing aggregate distributed representations of each document in rows. If unique document identifiers are contained in the input file, they can be carried through to the output file. Otherwise, a new set of unique identifiers is output along with the vector representations.

Parameters:

input_path : string

Path to file or directory of files containing text to be analyzed.

output_path : string

Path to file or directory of files containing text to be analyzed.

model : Word2Vec model

Gensim.Word2Vec model containing vector representations to be used for aggregate vectors.

num_features : int/float

Dimensionality of the Word2Vec model being used.

model_word_set : list

List of words in vocabulary of Word2Vec model.

text_col : string or int

Column name or index number for column containing text to be analyzed.

filter_out : list

Optional list of words to exclude from aggregation process.

delimiter : string

Delimiter to use for output CSV file. Default is tab delimited.

id_col : String, int, or False

Column name or index number for column containing unique identifier for documents. If input document does not contain a unique ID column, specify ‘False’ (which is the default value). If id_col is False, unique IDs will be automatically generated.

- **doc_vecs_from_txt(** input_path, output_path, model, num_features, model_word_set, text_col, filter_out = [], delimiter = “\t”, id_col = False **)**

This function reads a CSV file with documents as rows and generates a CSV file containing aggregate distributed representations of each document in rows. If unique document identifiers are contained in the input file, they can be carried through to the output file. Otherwise, a new set of unique identifiers is output along with the vector representations.

Parameters:

input_path : string

Path to file or directory of files containing text to be analyzed.

output_path : string

Path to file or directory of files containing text to be analyzed.

model : Word2Vec model

Gensim.Word2Vec model containing vector representations to be used for aggregate vectors.

num_features : int/float

Dimensionality of the Word2Vec model being used.

model_word_set : list

List of words in vocabulary of Word2Vec model.

text_col : string or int

Column name or index number for column containing text to be analyzed.

filter_out : list

Optional list of words to exclude from aggregation process.

delimiter : string

Delimiter to use for output CSV file. Default is tab delimited.

id_col : String, int, or False

Column name or index number for column containing unique identifier for documents. If input document does not contain a unique ID column, specify ‘False’ (which is the default value). If id_col is False, unique IDs will be automatically generated.

- **terms_from_txt(** input_path **)**

This function extracts terms from term dictionaries organized as individual text files. The name of each term dictionary file is used as the name of the dimension and is stored in a Python dictionary object. The words contained in each term dictionary are stored as values in the Python dictionary object. **NOTE:** Words should be delimited by single spaces and line breaks should not be used.

Parameters:

input_path : string

Path to file or directory.

Returns:

dic_terms_out : Python dictionary object

Python dictionary object with dimension names as keys and dimension words as values.

- **terms_from_liwc(** input_path **)**

This function extracts terms from a LIWC dictionary file. The name of each dictionary dimension is stored and the words associated with each dimension are stored as values in the Python dictionary object.

Parameters:

input_path : string

Path to file or directory.

Returns:

dic_terms_out : Python dictionary object

Python dictionary object with dimension names as keys and dimension words as values.

- **terms_from_csv(** input_path **)**

This function extracts terms from term dictionaries contained in a CSV file with dimensions as columns and words as column cells. The name of each column is stored and the words contained in each term dictionary are stored as values in the Python dictionary object.

Parameters:

input_path : string

Path to file or directory.

Returns:

dic_terms_out : Python dictionary object

Python dictionary object with dimension names as keys and dimension words as values.

- **terms_to_csv(** terms_dic, output_path, delimiter=“”\t”” **)**

This function writes an addr term-dictionary object to CSV file with columns as dimensions.

Parameters:

terms_dic : Python dictionary object

Dictionary object containing dimension names as keys and words as values.

output_path : string

Path for output file.

- **dic_vecs(** dic_terms, model, num_features, model_word_set, filter_out = [] **)**

This function reads a CSV file with documents as rows and generates a CSV file containing aggregate distributed representations of each document in rows. If unique document identifiers are contained in the input file, they can be carried through to the output file. Otherwise, a new set of unique identifiers is output along with the vector representations.

Parameters:

dic_terms : Python dictionary object

Python dictionary object with dimension names as keys and words as values

model : Word2Vec model

Gensim.Word2Vec model containing vector representations to be used for aggregate vectors.

num_features : int/float

Dimensionality of the Word2Vec model being used.

model_word_set : list

List of words in vocabulary of Word2Vec model.

filter_out : list

Optional list of words to exclude from aggregation process.

Returns:

agg_dic_vecs : Python dictionary object

Python dictionary object with dimension names as keys and aggregate distributed dictionary representation vectors as values.

- **terms_to_csv(** dic_vecs, output_path, delimiter="”t”" **)**

This function writes a Python dictionary with dimension names as keys and aggregate distributed dictionary representation vectors as value to CSV file with columns as dimensions.

Parameters:

terms_dic : Python dictionary object

Dictionary object containing dimension names as keys and words as values.

output_path : string

Path to file or directory of files containing text to be analyzed.

delimiter : string

Delimiter to use for output CSV file. Default is tab delimited.

---

2. **get_loadings**

- **load_model(** model_path **)**

This function loads a Word2Vec model and returns the model object, model dimensionality, and word index.

Parameters:

model_path : string

Path to Word2Vec model.

Returns:

model : gensim.Word2Vec model object

A gensim.Word2Vec model object.

num_features : int

Model dimensionality.

model_word_set : list

List containing model vocabulary.

- **get_loadings(** agg_doc_vecs_path, agg_dic_vecs_path, out_path, num_features, delimiter = “\t” **)**

This function calculates the similarities between aggregate document vectors and aggregate dictionary vectors and returns a CSV with rows as documents and columns as dictionary dimensions. The unique IDs contained in the document CSV input file are transferred to the output file.

Parameters:

agg_doc_vecs_path : string

Path to CSV file with aggregate document vectors as rows and a unique ID as the first column.

agg_dic_vecs_path : string

Path to CSV file with aggregated dictionary vectors as columns.

output_path : string

Path for output.

num_features : int/float

Dimensionality of the Word2Vec model used to generate vector representations.

delimiter : string

Delimiter to use for output CSV file. Default is tab delimited.

---

2. **file_length**

- **file_len(** fname **)**

This is a helper function that returns the number of rows in a file.

Parameters:

fname : string

Path to file for which rows should be counted.

Returns:

nrow : Number of rows in file.

---

2. **simple_progress_bar**

- **update_progress(** progress **)**

This is a simple helper function that prints a progress bar that represents ration of completed iterations to total iterations (e.g. (lines_completed)/(total_lines))

Parameters:

progress : float

Ratio representing progress.