addr Introduction

addr contains interfaces and functionality to compute pair-wise document-dictionary similarities for a set of term-based dictionaries. The functions in addr can also be used to convert pre-processed text and term-dictionary files from a range of formats into aggregate vector representations.

Example

```
>>> import addr
```

Define file paths. These will be used later in function calls.

```
>>> documents_path=‘/../../documents.csv'
>>> dictionary_directory=‘/../../MFD-Seed-text-format'
>>> model_path=‘/../../GoogleNews-vectors-negative300.bin'
>>> dictionary_vector_path=‘/../../agg_dic_vectors.tsv’
>>> document_vector_path='/Users/joe/offline_research/addr-testing/agg_doc_vecs.tsv'
>>> document_loadings_out_path=‘/../../document_dictionary_loadings.tsv’
```

Load Word2Vec Model and specify model dimensionality and word index.

```
>>> model, num_features, model_word_set=addr.load_model(model_path)
```

Build Python dict of dictionary terms

```
>>> dic_terms=addr.terms_from_txt(input_path=dictionary_directory)
```

Make aggregate vector representations of dictionary dimensions

```
>>> agg_dic_vecs=addr.dic_vecs(dic_terms=dic_terms, model=model, num_features=num_featur
```

Write aggregate dictionary vectors to file

```
>>> addr.write_dic_vecs(dic_vecs=agg_dic_vecs, output_path=dictionary_vector_path)
```

Make aggregate vector representations of documents and write these to file

```
>>> addr.doc_vecs_from_csv(input_path=documents_path, output_path=document_vector_path,
                       model=model, num_features=num_features, model_word_set=model_word
                       text_col='text')
```

Calculate similarities between document and dictionary vectors

```
>>> addr.get_loadings(agg_doc_vecs_path=document_vector_path,
                  agg_dic_vecs_path=dictionary_vector_path,
                  out_path=‘document_loadings_out_path',
                  num_features=num_features)
```