WE1S “comparing” module

Included in the WE1S Workspace (see S-2), the “comparing” module allows you to compare two corpora (collections of texts and term frequency data about them) to one another to discover how, and how much, they differ. These two corpora may be texts from two different projects you are working on in the WE1S Workspace, or two sets of texts from within the same project (e.g., those classified as being about the humanities vs. those classified as being about science).

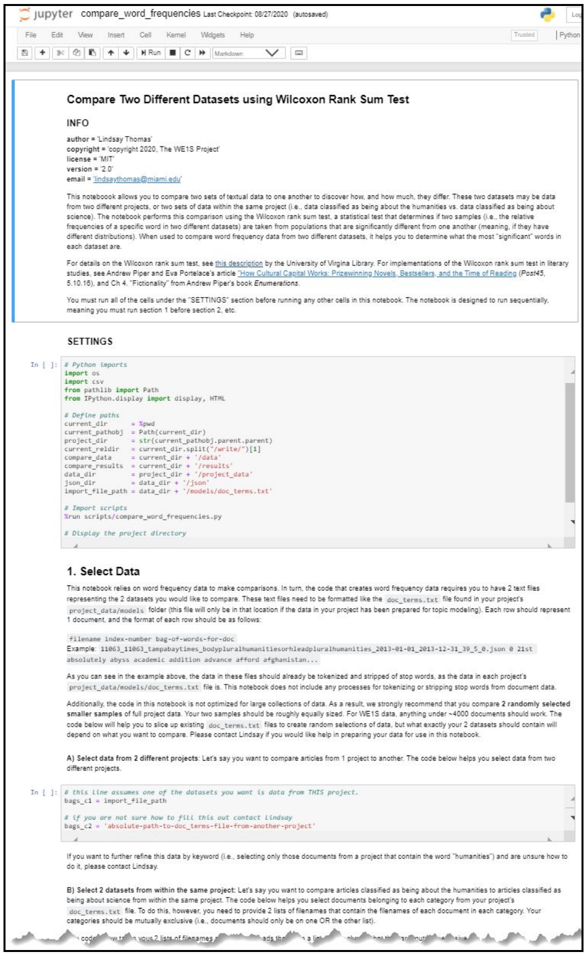
The module performs this comparison using the Wilcoxon rank sum test, a statistical test that determines if two samples (i.e., the relative frequencies of shared words in two different datasets) are taken from populations that are significantly different from one another (meaning, if they have different distributions). When used to compare word frequency data from two different corpora, it helps you to determine what the most "significant" words in each corpus are. For more on the Wilcoxon rank sum test in the context of corpus analytics, see these articles: a, b.

To run the code in this module, you must have previously used the “topic_modeling” module (see S-13) to create a bag of words plain-text file of all of your project data.

Module structure

```
comparing
 ├ data
 ├ results
 ├ scripts
 │ ├ compare_word_frequencies.py
 ├ compare-word-frequencies.ipynb
 └ README.md
```

Screenshot from
`compare_word_frequencies.ipynb` (larger)



jupyter compare_word_frequencies Last Checkpoint: 08/27/2020 (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python
Run
Markdown

Compare Two Different Datasets using Wilcoxon Rank Sum Test

INFO

author = 'Lindsay Thomas'
copyright = 'copyright 2020, The WE1S Project'
license = 'MIT'
version = '2.0'
email = 'lindsaythomas@miami.edu'

This noteboook allows you to compare two sets of textual data to one another to discover how, and how much, they differ. These two datasets may be data from two different projects, or two sets of data within the same project (i.e., data classified as being about the humanities vs. data classified as being about science). The notebook performs this comparison using the Wilcoxon rank sum test, a statistical test that determines if two samples (i.e., the relative frequencies of a specific word in two different datasets) are taken from populations that are significantly different from one another (meaning, if they have different distributions). When used to compare word frequency data from two different datasets, it helps you to determine what the most "significant" words in each dataset are.

For details on the Wilcoxon rank sum test, see this description by the University of Virgina Library. For implementations of the Wilcoxon rank sum test in literary studies, see Andrew Piper and Eva Portelace's article "How Cultural Capital Works: Prizewinning Novels, Bestsellers, and the Time of Reading (*Post45*, 5.10.16), and Ch 4. "Fictionality" from Andrew Piper's book *Enumerations*.

You must run all of the cells under the "SETTINGS" section before running any other cells in this notebook. The notebook is designed to run sequentially, meaning you must run section 1 before section 2, etc.

SETTINGS

```
In [ ]: # Python imports
import os
import csv
from pathlib import Path
from IPython.display import display, HTML

# Define paths
current_dir      = %pwd
current_pathobj  = Path(current_dir)
project_dir      = str(current_pathobj.parent.parent)
current_reldir   = current_dir.split("/write/")[1]
compare_data     = current_dir + '/data'
compare_results  = current_dir + '/results'
data_dir         = project_dir + '/project_data'
json_dir         = data_dir + '/json'
import_file_path = data_dir + '/models/doc_terms.txt'

# Import scripts
%run scripts/compare_word_frequencies.py

# Display the project directory
```

1. Select Data

This notebook relies on word frequency data to make comparisons. In turn, the code that creates word frequency data requires you to have 2 text files representing the 2 datasets you would like to compare. These text files need to be formatted like the `doc_terms.txt` file found in your project's `project_data/models` folder (this file will only be in that location if the data in your project has been prepared for topic modeling). Each row should represent 1 document, and the format of each row should be as follows:

`filename index-number bag-of-words-for-doc`
Example: `11063_11063_tampabaytimes_bodypluralhumanitiesorhleadpluralhumanities_2013-01-01_2013-12-31_39_5_0.json 0 21st absolutely abyss academic addition advance afford afghanistan...`

As you can see in the example above, the data in these files should already be tokenized and stripped of stop words, as the data in each project's `project_data/models/doc_terms.txt` file is. This notebook does not include any processes for tokenizing or stripping stop words from document data.

Additionally, the code in this notebook is not optimized for large collections of data. As a result, we strongly recommend that you compare **2 randomly selected smaller samples** of full project data. Your two samples should be roughly equally sized. For WE1S data, anything under ~4000 documents should work. The code below will help you to slice up existing `doc_terms.txt` files to create random selections of data, but what exactly your 2 datasets should contain will depend on what you want to compare. Please contact Lindsay if you would like help in preparing your data for use in this notebook.

A) Select data from 2 different projects: Let's say you want to compare articles from 1 project to another. The code below helps you select data from two different projects.

```
In [ ]: # this line assumes one of the datasets you want is data from THIS project.
bags_c1 = import_file_path

# if you are not sure how to fill this out contact Lindsay
bags_c2 = 'absolute-path-to-doc_terms-file-from-another-project'
```

If you want to further refine this data by keyword (i.e., selecting only those documents from a project that contain the word "humanities") and are unsure how to do it, please contact Lindsay.

B) Select 2 datasets from within the same project: Let's say you want to compare articles classified as being about the humanities to articles classified as being about science from within the same project. The code below helps you select documents belonging to each category from your project's `doc_terms.txt` file. To do this, however, you need to provide 2 lists of filenames that contain the filenames of each document in each category. Your categories should be mutually exclusive (i.e., documents should only be on one OR the other list).

Further Information:

* M-15 (about Jupyter notebooks)
* Clay Ford. “The Wilcoxon Rank Sum Test” (2017).
* Jeffrey Lijffijt, et al. “Significance Testing of Word Frequencies in Corpora” (2016).

Main Jupyter notebooks in this module:

* `compare_word_frequencies.ipynb`
* must previously have run the WE1S “topic_modeling” module (see S-13)

Code source: [TBD] (MIT License)

13 Sept. 2020; rev. 13 Sept. 2020 (Lindsay Thomas)