Step-by-step guide

1. Choose a persona (below) and choose a practice dataset (below). Read through the Quickstart Guide with both the persona and dataset in mind. Take notes as you attempt to follow the guide. Given your chosen persona, what might not be clear or obvious? What might be particularly effective? If you were to begin attempting to following the guide using your chosen dataset, what challenges (if any) might your persona experience?
2. Choose one of the datasets and follow the steps in the Practice research task below. Collect notes about your experience. What was easily done using the GATK? What wasn't as easily done using the GATK? How did you leverage the GATK to complete the prompts? How did you discover where to go in the GATK to complete each prompt?

User personas

- Beginner: A user for whom working with the GATK is their first experience using computational techniques on a humanities dataset.
- Programming Humanist: A user who has used Python before and has engaged with humanist datasets, but would not self-identify as a developer. Example: a research librarian with access to a the Early modern mixed genres dataset described in Practice datasets below who has used Python's NLTK package in the past to perform basic tokenization and word frequency counting.
- Humanist Programmer: A user who would self-identify as a developer, is knowledgeable of Python and its ecosystem, and is interested in applying their experience with these tools to a humanist dataset.

Practice datasets

1. Reddit corpus: a sampling of top posts and comments from the `r/starwars` subreddit. The event of relevance for the Practice research task below is TBD.
2. Early modern mixed genres: a large selection of early modern English textual sources of a variety of genres. The event of relevance for the Practice research task below is TBD.

Practice research task

For your chosen dataset:

1. What's the frequency of binary gendered pronouns?
2. Introduce a non-binary gendered pronoun set.
3. What's the frequency of the above non-binary gendered pronoun set?

4. What're the most common nouns associated with each of the above genders (non-binary and binary set)?
5. What're the most common adjectives associated with each of the above genders?
6. TBD
7. TBD
8. What're the most common adjectives associated with each of the above genders before and after the event described in their respective entry in Practice dataset above?
9. Produce a data visualization for steps 3-8 above.

Init

- No guide on setting up a corpus/the path -> TODO: add an issue with rough instructions

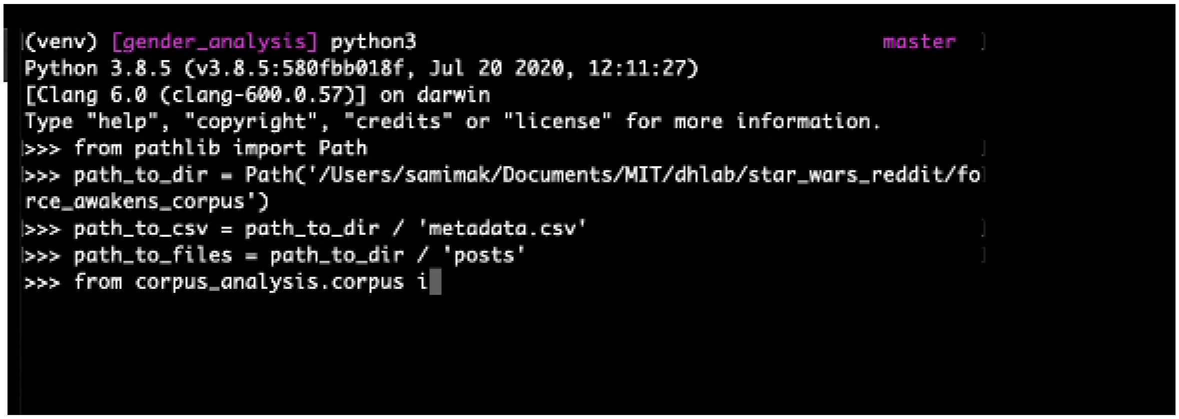
Issue:

Metadata instruction see:
https://docs.google.com/document/d/130pWbn734Bx2ZS314BQoBVm8n4M915wbr-k62YX9D2k/edit

Use case:
No metadata -> need a guide on how to do one
Metadata doesn’t match
Having extra metadata entries: i.e. multiple dates
Having less metadata entries: i.e. author gender missing

Group Testing session

Good habit to set up PATH variable



```
(venv) [gender_analysis] python3                                       master
Python 3.8.5 (v3.8.5:580fbb018f, Jul 20 2020, 12:11:27)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from pathlib import Path
>>> path_to_dir = Path('/Users/samimak/Documents/MIT/dhlab/star_wars_reddit/fo
rce_awakens_corpus')
>>> path_to_csv = path_to_dir / 'metadata.csv'
>>> path_to_files = path_to_dir / 'posts'
>>> from corpus_analysis.corpus i
```

Corpus

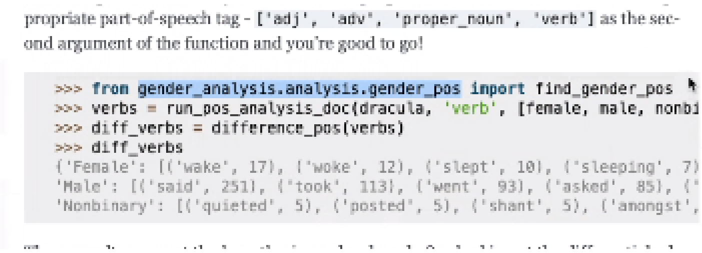
- Have a summary for metadata/corpus to help with validation
- Labels with filenames, etc.
- Navigation design => how to make users to navigate more easily?
- Docs
    - Have attributes

Word_count: maybe have a config for people to set to get rid of stopwords?
And maybe it’s unnecessary to have the instructions with for-loops to instruct people to get rid of stopwords themselves

Word_associated function with weird behaviors: maybe utilizing POS or other libraries for more useful info?

Get_part_of_speech_tags: maybe we should do a sanity check before return (i.e. for special chars like , and * being tagged, etc.) => should also have a config in the preproce pipeline to sanitize the input text/maintain raw text

Run_adj_analysis_doc isn’t on master (currently in the Part of Speech analysis section, but maybe referenced multiple times)

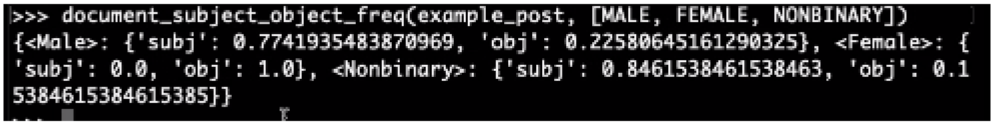


propriate part-of-speech tag - ['adj', 'adv', 'proper_noun', 'verb'] as the second argument of the function and you're good to go!

```
>>> from gender_analysis.analysis.gender_pos import find_gender_pos
>>> verbs = run_pos_analysis_doc(dracula, 'verb', [female, male, nonbi
>>> diff_verbs = difference_pos(verbs)
>>> diff_verbs
{'Female': [('wake', 17), ('woke', 12), ('slept', 10), ('sleeping', 7)
'Male': [('said', 251), ('took', 113), ('went', 93), ('asked', 85), ('
'Nonbinary': [('quieted', 5), ('posted', 5), ('shant', 5), ('amongst',
```

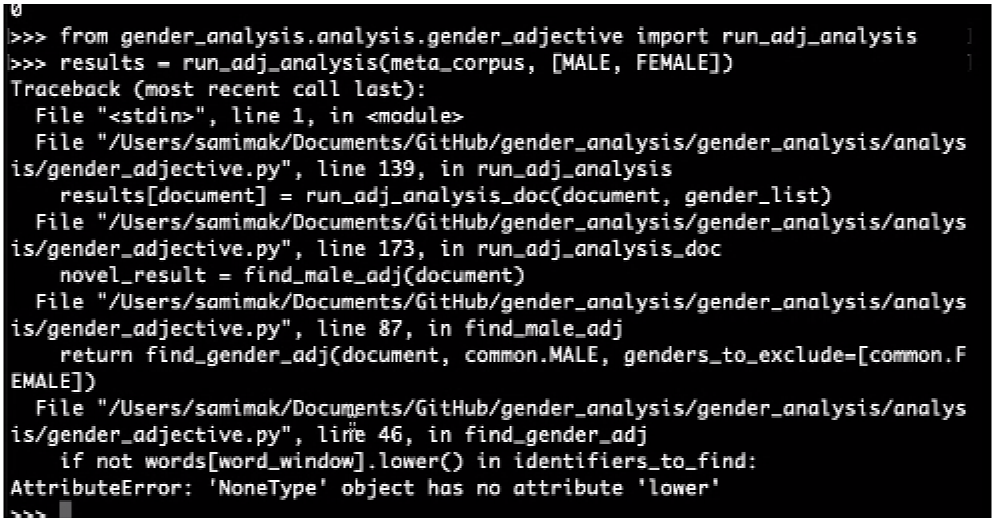
Doc_prounoun_freq: need to revise the approach!

Interesting data:



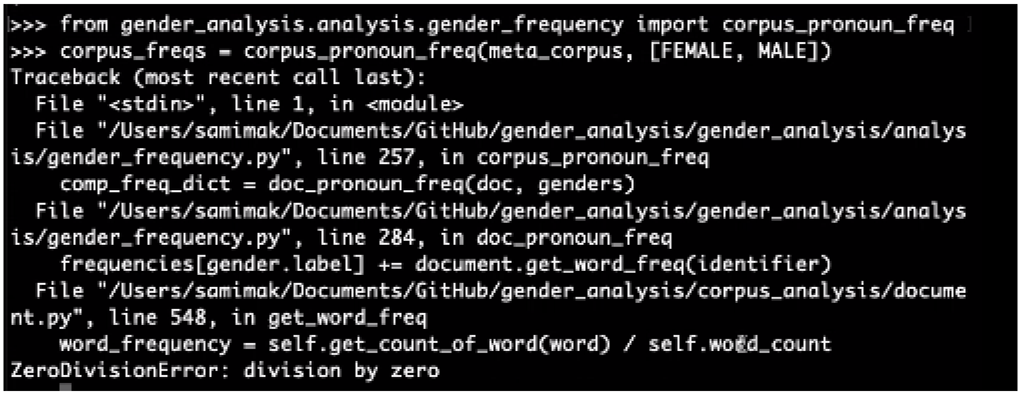
```
>>> document_subject_object_freq(example_post, [MALE, FEMALE, NONBINARY])
{<Male>: {'subj': 0.7741935483870969, 'obj': 0.22580645161290325}, <Female>: {
'subj': 0.0, 'obj': 1.0}, <Nonbinary>: {'subj': 0.8461538461538463, 'obj': 0.1
5384615384615385}}
```

Error due to empty “files” in run_adj_analysis:



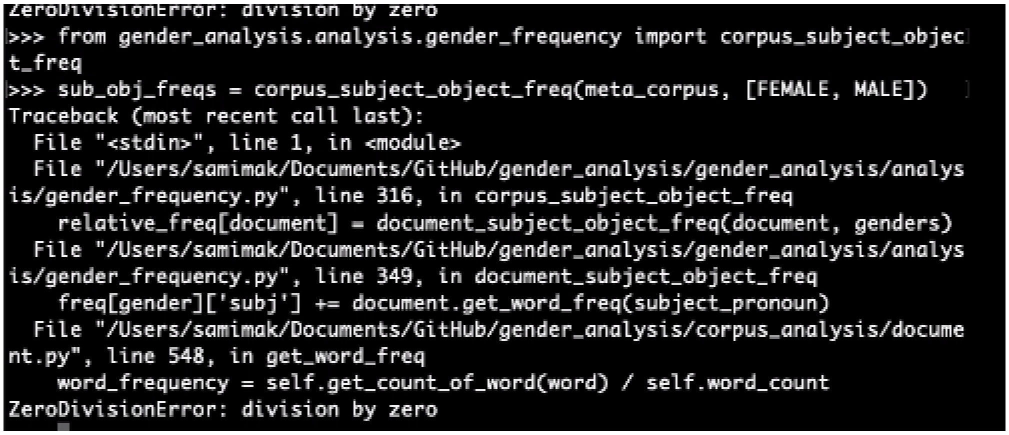
```
0
>>> from gender_analysis.analysis.gender_adjective import run_adj_analysis
>>> results = run_adj_analysis(meta_corpus, [MALE, FEMALE])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_adjective.py", line 139, in run_adj_analysis
    results[document] = run_adj_analysis_doc(document, gender_list)
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_adjective.py", line 173, in run_adj_analysis_doc
    novel_result = find_male_adj(document)
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_adjective.py", line 87, in find_male_adj
    return find_gender_adj(document, common.MALE, genders_to_exclude=[common.F
EMALE])
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_adjective.py", line 46, in find_gender_adj
    if not words[word_window].lower() in identifiers_to_find:
AttributeError: 'NoneType' object has no attribute 'lower'
>>>
```

Corpus_pronoun_freq also not working



```
>>> from gender_analysis.analysis.gender_frequency import corpus_pronoun_freq
>>> corpus_freqs = corpus_pronoun_freq(meta_corpus, [FEMALE, MALE])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_frequency.py", line 257, in corpus_pronoun_freq
    comp_freq_dict = doc_pronoun_freq(doc, genders)
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_frequency.py", line 284, in doc_pronoun_freq
    frequencies[gender.label] += document.get_word_freq(identifier)
  File "/Users/samimak/Documents/GitHub/gender_analysis/corpus_analysis/docume
nt.py", line 548, in get_word_freq
    word_frequency = self.get_count_of_word(word) / self.word_count
ZeroDivisionError: division by zero
```

Corpus_subject_object_freq also broken:



```
ZeroDivisionError: division by zero
>>> from gender_analysis.analysis.gender_frequency import corpus_subject_objec
t_freq
>>> sub_obj_freqs = corpus_subject_object_freq(meta_corpus, [FEMALE, MALE])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_frequency.py", line 316, in corpus_subject_object_freq
    relative_freq[document] = document_subject_object_freq(document, genders)
  File "/Users/samimak/Documents/GitHub/gender_analysis/gender_analysis/analys
is/gender_frequency.py", line 349, in document_subject_object_freq
    freq[gender]['subj'] += document.get_word_freq(subject_pronoun)
  File "/Users/samimak/Documents/GitHub/gender_analysis/corpus_analysis/docume
nt.py", line 548, in get_word_freq
    word_frequency = self.get_count_of_word(word) / self.word_count
ZeroDivisionError: division by zero
```