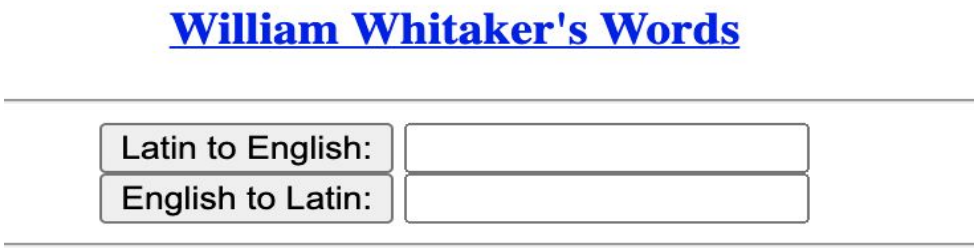
Latin Linguistic Analysis

Frances Harrington

What is it?

- My project takes a collection of Latin texts and examines them for instances of the words gladius, ferrum, mucro, ensis, and capulus (and their various forms)
- These are the five main words in Latin used for “sword”
- I collected, cleaned, and organized the data into dataframes
- I then analyzed the findings about how each of these words occurred
- With some time left I thought about doing some machine learning (like a basic Naive Bayes) mainly because I think it’s super cool; however, I didn’t do any before this presentation

Why this?



- I wanted to combine what we've learned in this class with what I do in my other major (classics)
- Specifically I knew that there were a lot of synonyms for the word "sword" and I wondered what the usage rates for each word was! (William Whitaker's Words run by Notre Dame is great and what I used to get basic grammatical info on the words!)
- I also wanted to use the Latin Library website

Resource

- The Latin Library is a website that collects Latin texts for free use by anyone
- It's a great resource and has been maintained by William L. Carey for more than 20 years (and it's always been completely free!)


THE LATIN LIBRARY

Alcuin GO!

Ammianus	Apuleius	Augustus	Aurelius Victor	Caesar	Cato
Catullus	Cicero	Claudian	Curtius Rufus	Ennius	Eutropius
Florus	Frontinus	Gellius	Historia Augusta	Horace	Justin
Juvenal	Livy	Lucan	Lucretius	Martial	Nepos
Ovid	Persius	Petronius	Phaedrus	Plautus	Pliny Maior
Pliny Minor	Propertius	Quintilian	Sallust	Seneca Maior	Seneca Minor
Silius Italicus	Statius	Suetonius	Sulpicia	Tacitus	Terence
Tibullus	Valerius Flaccus	Valerius Maximus	Varro	Velleius	Vergil
Vitruvius	Ius Romanum	Miscellany	Christian	Medieval	Neo-Latin

Credits About These Texts Technical Notes Index ePUBS

Data Collection and Organization

- I scraped my texts from the Latin Library website using a Spider in Jupyter Notebook
- The data I collected included the actual text, the title, author's name, and the era in which they lived
- This was by far the most difficult part of the project!
- Cleaning the data and getting it into the first dataframe was very tricky
- For scraped data I had two initial dataframes: one with the text and title and one with the author and era

Methods used to incorporate the authors and eras:

```
ne_df['name'] = [w[:-2] for w in ne_df.name]
```

```
def MatchName(l):
    for name in ne_df.name:
        last_word = name.split()[len(name.split()) - 1]
        if last_word in l:
            return name
            return "None"
```

```
latlib_df['author'] = latlib_df.text.map(MatchName)
```

```
def MatchEra(l):
    for x in ne_df.merged:
        last_word = x.split()[len(x.split()) - 1]
        if last_word in l:
            clean = re.compile(r'\)\w* \w* \w*|\)\w* \w*|\)\w+\. \w* \w*|\)\w+|\(')
            return(clean.sub(' ', x))
```

```
latlib_df['era'] = latlib_df.text.map(MatchEra)
```

What raw, uncleaned text looks like:

```
[['P. VERGILI MARONIS AENEIDOS LIBER DVODECIMVS\n\n',
  '\n\n',
  '\nTurnus ut infractos adverso Marte Latinos',
  '\ndefecisse videt, sua nunc promissa reposci,',
  '\nse signari oculis, ultro implacabilis ardet',
  '\nattollitque animos. Poenorum qualis in arvis',
  '\nsaucius ille gravi venantum vulnere pectus\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0',
  '\ntum demum movet arma leo, gaudetque comantis',
  '\nexcutiens cervice toros fixumque latronis',
  '\nimpavidus frangit telum et fremit ore cruento:',
  '\nhaud secus accenso gliscit violentia Turno.',
  '\ntum sic adfatur regem atque ita turbidus infit:\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0',
```

What my cleaned text looks like:

```
Out[10]: 'P. VERGILI MARONIS AENEIDOS LIBER OCTAVVS  Ut belli signum Laurenti Turnus ab arce extulit et rauco strepuerunt corn
ua cantu, utque acris concussit equos utque impulit arma, extemplo turbati animi, simul omne tumultu coniurat trepido
Latium saevitque iuventus effera. ductores primi Messapus et Ufens contemptorque deum Mezentius undique cogunt auxili
a et latos vastant cultoribus agros. mittitur et magni Venulus Diomedis ad urbem qui petat auxilium, et Latio consist
ere Teucros, advectum Aenean classi victosque penatis inferre et fatis regem se dicere posci edoceat, multasque viro
se adiungere gentis Dardanio et late Latio increbrescere nomen: quid struat his coeptis, quem, si fortuna sequatur, e
ventum pugnae cupiat, manifestius ipsi quam Turno regi aut regi apparere Latino. Talia per Latium. quae Laomedontius
heros cuncta videns magno curarum fluctuat aestu, atque animum nunc huc celerem nunc dividit illuc in partisque rapit
varias perque omnia versat, sicut aquae tremulum labris ubi lumen aenis sole repercussum aut radiantis imagine lunae
omnia pervolitat late loca, iamque sub auras erigitur summique ferit laquearia tecti. nox erat et terras animalia fes
sa per omnis alituum pecudumque genus sopor altus habebat, cum pater in ripa gelidique sub aetheris axe Aeneas, trist
```

Data Collection and Organization (continued)

- I tokenized the texts and included token count as well
- I also needed to split the texts between two categories: prose and verse
- This split was done by using a method to find the max line length
  - I picked works by two authors who I knew wrote poems and used that method to find the longest line and made that my threshold for what qualified as a poem
  - Any work with a maximum line length of 87 or shorter was categorized as “verse” and any longer as “prose”

```
def MaxLineLength(l):
    return(len(max(l, key=len)))
```

```
latlib_df['max_line_length'] = latlib_df.text_raw.map(MaxLineLength)
```

This is what my main DataFrame looks like now:

	title	text	text_raw	tokens	token_count	max_line_length	style	author	life_span	era	term
77	Tacitus: Histories I	P. CORNELI TACITI HISTORIARVM LIBER PRIMVS ...	[P. CORNELI TACITI HISTORIARVM LIBER PRIMVS\n,...	[P., CORNELI, TACITI, HISTORIARVM, LIBER, PRIM...	14024	1733	prose	P. CORNELIVS TACIT	c. 56 – c. 117 A.D.	Silver Age	
195	Justin XXXIV	HISTORIARUM PHILIPPICARUM IN EPITOMEN REDACTI ...	[HISTORIARUM PHILIPPICARUM IN EPITOMEN REDACTI...	[HISTORIARUM, PHILIPPICARUM, IN, EPITOMEN, RED...	753	303	prose	TITVS LVCRETIVS CAR	c. 94 – c. 49 B.C.	Golden Age	
545	Auli Gellii Noctes Atticae: Liber XII	AVLI GELLI NOCTES ATTICAE: LIBER XII I Dissert...	[AVLI GELLI NOCTES ATTICAE: LIBER XII\n, \nI D...	[AVLI, GELLI, NOCTES, ATTICAE, :, LIBER, XII, ...	5966	5302	prose	AVLVS GELLI	c. A.D. 125 – after 180	Later	

Data Collection and Organization: Specific Vocabulary

- Each vocabulary term I examined was filtered through and put into its own dataframe
- Then I added a new column: “term” to each df with the corresponding word in each row
- I created a new dataframe called “df_swords” that merged all of the individual dfs into one

```
#ensis:
ensis = latlib_df[(latlib_df.text.str.contains(pat = 'ensis', case=False)) | (latlib_df.text.str.contains(pat = 'ensi', case=False)) | (latlib_df.text.str.contains(pat = 'ensem', case=False)) | (latlib_df.text.str.contains(pat = 'ense', case=False)) | (latlib_df.text.str.contains(pat = 'enses', case=False)) | (latlib_df.text.str.contains(pat = 'ensium', case=False)) | (latlib_df.text.str.contains(pat = 'ensibus', case=False))]
```

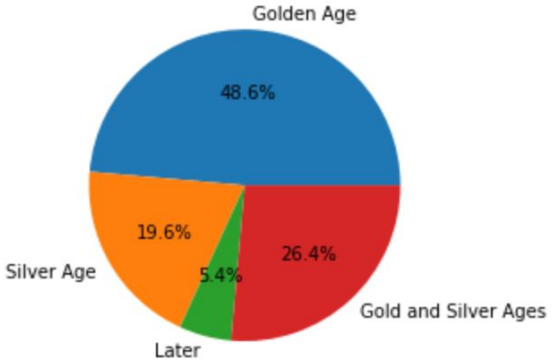
```
ensis.term = ['ensis' for x in ensis.term]
```

	title	text	text_raw	tokens	token_count	max_line_length	style	author	life_span	era	term
0	Ammianus: Liber XXXI	AMMIANI MARCELLINI HISTORIAE LIBER XXXI ...	[AMMIANI MARCELLINI HISTORIAE LIBER XXXI\n, \n...	[AMMIANI, MARCELLINI, HISTORIAE, LIBER, XXXI, ...	11030	774	prose	AMMIANVS MARCELLIN	4th century A.D.	Later	ensis

Analysis

Overall data stats

- 31 different authors
- 672 different works
- About 70% of those works are prose, 30% are written in verse



	token_count	max_line_length
count	610.000000	610.000000
mean	6975.355738	1315.316393
std	5442.246454	1847.014180
min	94.000000	51.000000
25%	2771.000000	77.000000
50%	6032.000000	646.000000
75%	9302.750000	2001.000000
max	38414.000000	20845.000000

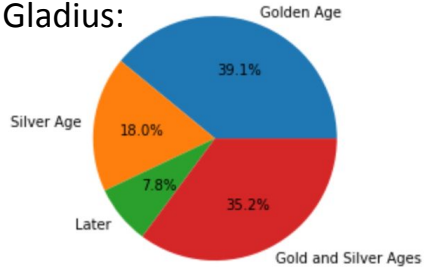
Specific Vocabulary

- My hypothesis was that “gladius” would be the most used word out of the five
- The occurrences of each word actually went like this:
  - gladius: 277
  - ferrum: 462
  - mucro: 89
  - ensis: 573
  - capulus: 48
- So, ensis is the top result! Odd since Logeion (great Latin/Ancient Greek vocabulary source run by UChicago) lists it as the 1131st most frequent word…
  - Contrary to my hypothesis, gladius was actually the 3rd most frequent word
- For the prose/verse split, most words lean heavily in favor of prose except “capulus” which skews almost exactly inversely towards verse

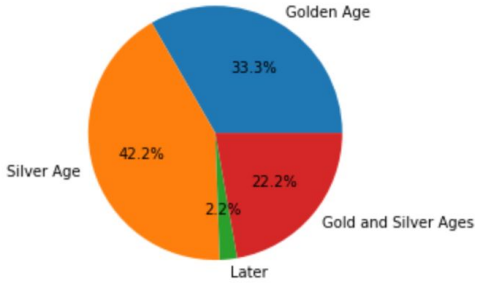
Specific Vocabulary (continued)

- I also took a look at era variance (how many different eras each term appeared in)
  - Gladius had the most variation while capulus had the least
  - Taking a look at capulus, there's pretty large span of time, from the 2nd century BC to the 4th century AD!
  - However, the majority of usage comes from after the year 0 so while there is a large span, it's mostly concentrated on one end!

Gladius:



Capulus:




```
gladius era variance:  22
ferrum era variance:  26
mucro era variance:  16
ensis era variance:  27
capulus era variance:  12
```

Improvements

- df_swords could use some reorganization
  - df_swords only has one “term” word per row, so some works repeat if they contain multiple terms, I want to figure out how to make it so there can be multiple terms per row
- Work on reducing my spider output
  - Right now the spider generates a lot of text which is very annoying!
- Generate more visuals and save them to their own folder
  - Could use data_samples/ but a separate area just for charts/visuals could be really helpful
- Change the way my terms are filtered (from .contains() to a regex)