Week4-DH

October 24, 2021

0.1 Week 4 - Digital Humanities

[26]:
```
import re
import numpy as np
import pandas as pd
from scipy import stats
```

Replace "Ann" by "Alice"

[4]:
```
re.sub('^Ann', 'Alice','Ann plays the role with Mary and Annie')
```

[4]:
```
'Alice plays the role with Mary and Annie'
```

Remove the decimal part of prices

[5]:
```
aPat = '(?<=[\d+])\.\d+'
re.sub(aPat ,'.','$99.99 to $87.80 or Fr.75.50')
```

[5]:
```
'$99. to $87. or Fr.75.'
```

0.1.1 For the genre 'Comédie', extract (in a list) the number of word-tokens per play.

[6]:
```
import os
import lxml.etree
import tarfile
import collections
import matplotlib.pyplot as plt

tf = tarfile.open('./theatre-classique.tar.gz','r')
tf.extractall('data')
```

[7]:
```
subgenres = ('Comédie', 'Tragédie', 'Tragi-comédie')
print(subgenres)
plays, titles, genres = [], [], []
authors, years = [],[]
```

```
('Comédie', 'Tragédie', 'Tragi-comédie')
```

[8]:
```
for fn in os.scandir('data/theatre-classique'):
    # Only include XML files
    if not fn.name.endswith('.xml'):
        continue
    tree   = lxml.etree.parse(fn.path)
    genre  = tree.find('//genre')
    title  = tree.find('//title')
    author = tree.find('//author')
    year   = tree.find('//date')
    if genre is not None and genre.text in subgenres:
        lines = []
        for line in tree.xpath('//l|//p'):
            lines.append(' '.join(line.itertext()))
        text = '\n'.join(lines)
        plays.append(text)
        genres.append(genre.text)
        titles.append(title.text)
        authors.append(author.text)
        if year is not None:
            years.append(year.text)
```

[50]:
```
counts = collections.Counter(genres)
print("Total no. of plays per genres: ", counts)
```

```
Total no. of plays per genres:  Counter({'Comédie': 310, 'Tragédie': 150,
'Tragi-comédie': 38})
```

Extract mean word-tokens per genre

[80]:
```
word_token_all_comedie = []
word_token_all_tragi_comedie = []
word_token_all_tragedie = []

for play, genre in zip(plays, genres):

    if genre == 'Comédie':
        total_words = re.findall('[\w]+', play)
        word_token_all_comedie.append(len(total_words))

    elif genre =='Tragédie':
        total_words = re.findall('[\w]+', play)
        word_token_all_tragedie.append(len(total_words))

    else:
        total_words = re.findall('[\w]+', play)
        word_token_all_tragi_comedie.append(len(total_words))


print("Length: {:3d}, Mean word tokens for Comedie       : {:.2f}"
      .format(len(word_token_all_comedie), np.mean(np.
 ↪array(word_token_all_comedie))))

print("Length: {:3d}, Mean word tokens for Tragédie      : {:.2f}"
      .format(len(word_token_all_tragedie), np.mean(np.
 ↪array(word_token_all_tragedie))))

print("Length: {:3d}, Mean word tokens for Tragi-comédie : {:.2f}"
      .format(len(word_token_all_tragi_comedie), np.mean(np.
 ↪array(word_token_all_tragi_comedie))))
```

```
Length: 310, Mean word tokens for Comedie       : 10041.42
Length: 150, Mean word tokens for Tragédie      : 14325.02
Length:  38, Mean word tokens for Tragi-comédie : 16232.11
```

0.1.2 Perform t-tests on mean of word tokens for different genres

[60]:
```
result = stats.ttest_1samp(word_token_all_comedie, 10000)
print("Statistic: {:.2f}, p-Value: {:.2f} for Comedie with 10000 mean word␣
 ↪count"
      .format(result.statistic, result.pvalue))
```

```
Statistic: 0.14, p-Value: 0.89 for Comedie with 10000 mean word count
```

[64]:
```
result = stats.ttest_1samp(word_token_all_tragedie, 14000)
print("Statistic: {:.2f}, p-Value: {:.2f} for Tragedie with 14000 mean word␣
 ↪count"
      .format(result.statistic, result.pvalue))
```

```
Statistic: 1.17, p-Value: 0.24 for Tragedie with 14000 mean word count
```

[62]:
```
result = stats.ttest_1samp(word_token_all_tragedie, 15000)
print("Statistic: {:.2f}, p-Value: {:.2f} for Tragedie with 15000 mean word␣
 ↪count"
      .format(result.statistic, result.pvalue))
```

```
Statistic: -2.43, p-Value: 0.02 for Tragedie with 15000 mean word count
```

0.1.3 Analysis

1. The population mean for Comedie genre is 10041.42. Since, we acheived t-test with p-Value of 0.89 with 10,000 mean word token for comedie genre, so we can accept the Null hypothesis

that mean of word-tokens approximatly close to the 10000 mean token count.

2. Population mean of Tragedie genre is 14325.02. The t-test gives p-value for Tragedie genre of 0.24 for 14000 mean word count. We can take this into consideration of p-value and can therefore say the mean word token near to the 14000 mean word count.
3. But when comparing mean ttest with 15000 mean word token, the p-value is very small 0.02. Therefore, we can reject the null hypothesis.