NI-VSM-HW2

Members: Andrei Shchapaniak (shchaand), Maksim Spiridonov (spirimak)

`Preparation.`

```
'''
# Representative member: Andrei Shchapaniak, 14.05.2002
'''
K = 14
L = len('Shchapaniak')
X = ((K*L*23) % 20) + 1
Y = ((X + ((K*5 + L*7) % 19)) % 20) + 1
'''
file1 = 003.txt
file2 = 018.txt
'''
```

1. `Load texts for analysis from both data files. Estimate the probability of word lengths separately for each text and graphically illustrate the distribution of word lengths.`

To find the probability of each word length in a text, we calculate the relative frequency of each length. The probability $P(l_i)$ of a word having length l_i is then:

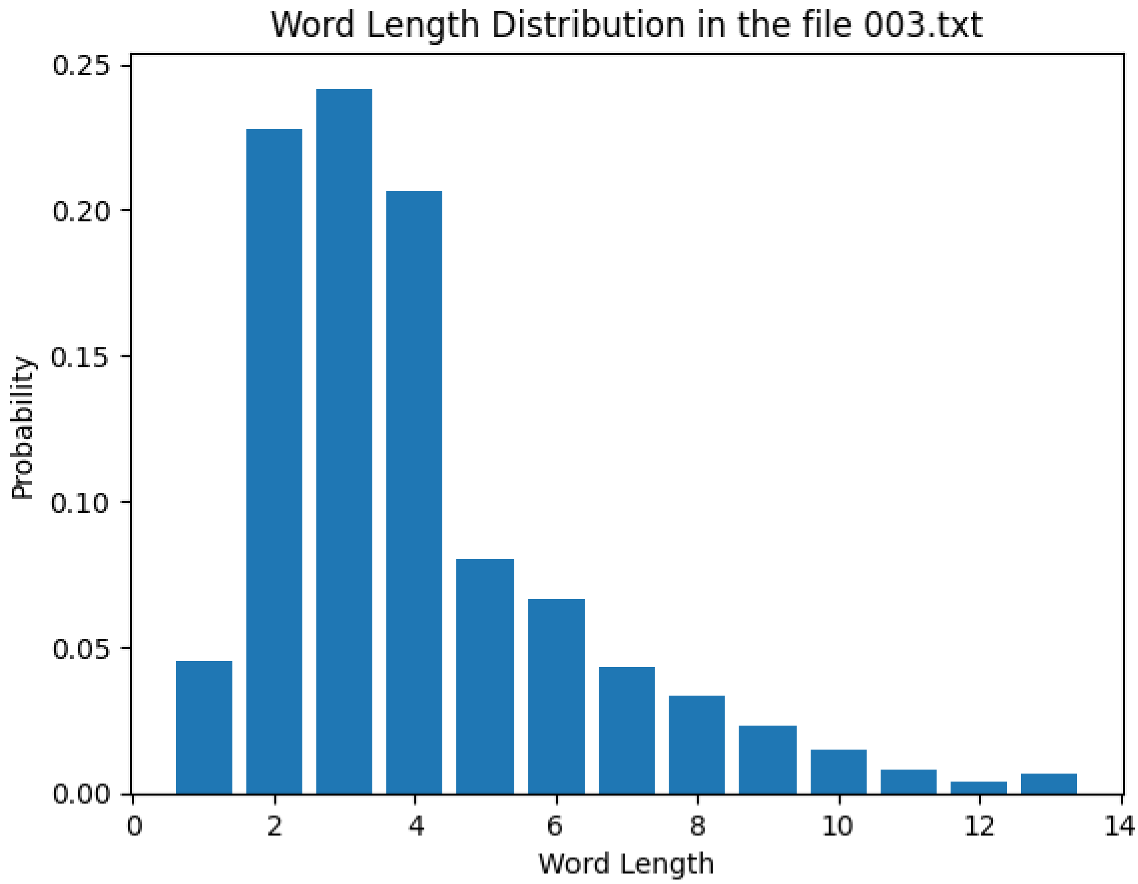
$$P(l_i) = \frac{f(l_i)}{\sum_j f(l_j)}$$

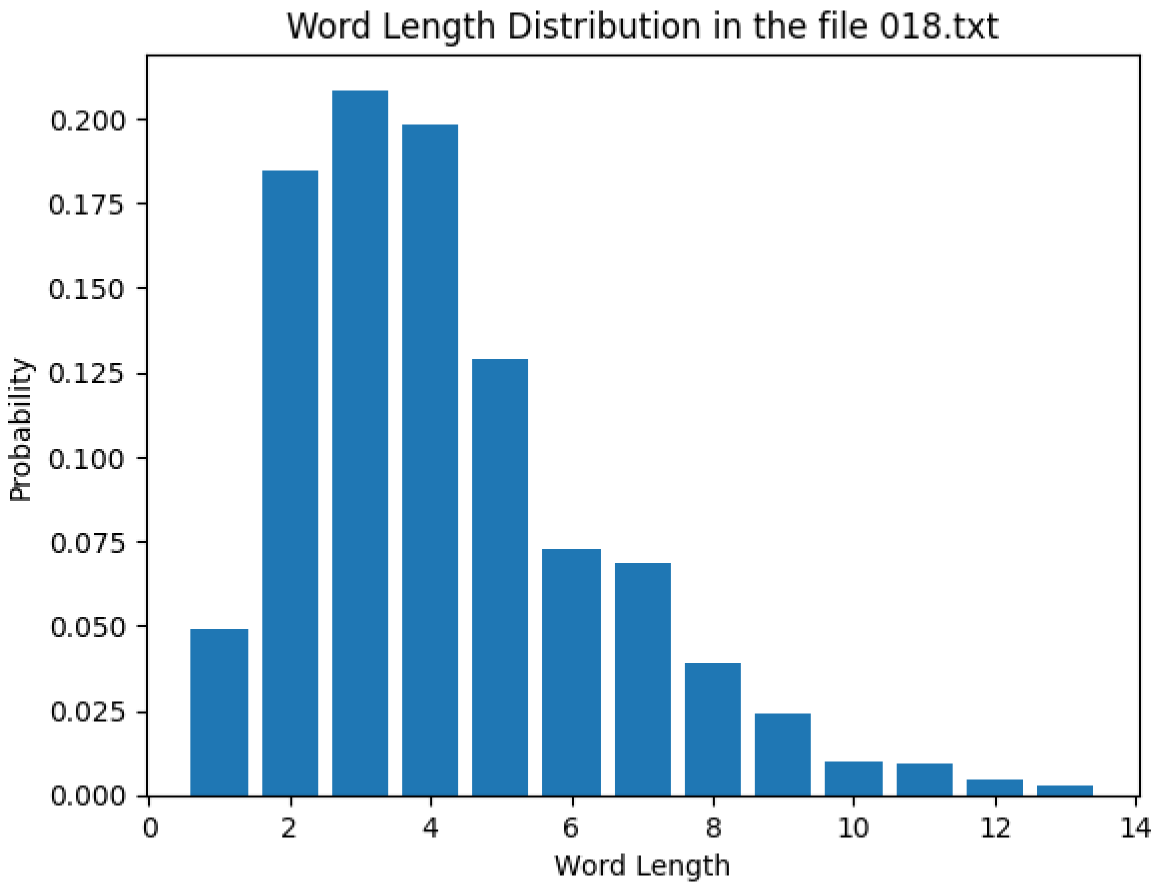
where $f(l_i)$ is the frequency of words of length l_i and $\sum_j f(l_j)$ is the sum of frequencies of all word lengths in the text.

```
from collections import Counter
import os

def read_text_calc_freq(filename):
    with open(filename, 'r', encoding='utf-8') as file:
        words = file.read().split()

    lens_map = Counter(list(map(len, words)))
    len_freq = {length: count / len(words) for length, count in
lens_map.items()}
    # call custom plot function with `len_freq` param
```





---

2. `For each text separately, estimate the basic characteristics of word lengths, i.e., the mean value and variance. Explain thoroughly how you are estimating these characteristics!`

```python
import numpy as np

# arr_words is a param from the previous task which was named `words`
def estmt_basic_chrs(arr_words):
    lens = np.array([len(word) for word in arr_words])
    mean = lens.mean()
    variance = lens.var(ddof = 1) # / (n - 1), that is why ddof is set to 1 (Bessel's correction)
    deviation = np.sqrt(variance)
```

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i \quad \text{-} \quad \text{sample mean value}$$

$$s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X}_n)^2 \quad \text{-} \quad \text{sample variance}$$

$$s_n = \sqrt{s_n^2} \quad \text{-} \quad \text{sample standard deviation}$$

	003.txt	018.txt
Sample mean value	3.99318	4.21771
Sample variance	5.11898	4.93501
Sample standard deviation	2.26251	2.22149

---

3. `At a 5% significance level, test the hypothesis that the distribution of word lengths does not depend on which text it is. Also, determine the p-value of the test. Hint: This can be done using a chi-squared test of independence in a contingency table. Thoroughly describe the hypothesis you are testing!`

As was mentioned in a hint, the **Chi-squared test** will be used for this task. This test compares the observed frequencies of word lengths in each text with the expected frequencies if there were no association between text source and word length. Let's construct contingency table:

$\hat{p}_{i.} = \frac{N_{i.}}{n}$ and $\hat{p}_{.j} = \frac{N_{.j}}{n}$.

Data		
Length	**003.txt**	**018.txt**
1	46	53
2	234	200
3	248	226
4	212	215
5	82	140
6	68	79
7	44	74
8	34	42
9	24	26
10	15	11
11	8	10
12	4	5
13	7	3

Contingency Table	
003.txt	**018.txt**
48.13933	50.86066
211.03507	222.96492
230.48530	243.51469
207.631279	219.368720
107.94881	114.05118
71.47962	75.52037
57.37819	60.62180
36.95545	39.04454
24.31279	25.68720
12.64265	13.35734
8.75260	9.24739
4.37630	4.62369
4.86255	5.13744

In a contingency table used for a chi-squared test of independence, having values less than 5 can be problematic because the chi-squared test assumes that the sampling distribution of the test statistic approximates the chi-squared distribution.This issue is termed "small sample size problem" and it's "bad" because the chi-squared approximation to the true distribution may become poor.
To address this issue we combine categories.

Data		
Length	**003.txt**	**018.txt**
1	46	53
2	234	200
3	248	226
4	212	215
5	82	140
6	68	79
7	44	74
8	34	42
9	24	26
10	15	11
11	8	10
12	11	8

Contingency Table	
003.txt	**018.txt**
48.13933	50.86066
211.03507	222.96492
230.48530	243.51469
207.631279	219.368720
107.94881	114.05118
71.47962	75.52037
57.37819	60.62180
36.95545	39.04454
24.31279	25.68720
12.64265	13.35734
8.75260	9.24739
9.23886	9.76113

The null hypothesis for this test is that the distribution of word lengths is independent of the text source —meaning the text from which the words come has no effect on the distribution of their lengths. The alternative hypothesis is that there is a dependency, indicating the distribution of word lengths varies by text source.

H_0	H_A	**test statistic** χ^2	**critical region**
$p_{ij} = p_{i.}p_{.j}$	$p_{ij} \neq p_{i.}p_{.j}$	$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(N_{ij}-\frac{N_{i.}N_{.j}}{n})^2}{\frac{N_{i.}N_{.j}}{n}}$	$\chi^2 > \chi^2_{\alpha,(r-1)(c-1)}$

$$(r-1)(c-1) \quad - \quad \text{degrees of freedom}$$

where **r** is the number of rows, and **c** is the number of columns in contingency table.

```
class ChiSquaredTest:
    def __init__(self, text1, text2):
        self.text1 = text1
        self.text2 = text2

    def build_in_test(self):
        test_val, p_val, df, _ = st.chi2_contingency(self.Nij)
        print(f'Build-in test: test value: {test_val}, p-value {p_val}. df:
{df}')
```

```
    def create_contingency_table(self):
        all_lengths = sorted(set(self.text1.lens_map) |
set(self.text2.lens_map))
        table = [[self.text1.lens_map[length], self.text2.lens_map[length]]
for length in all_lengths]
        return np.matrix(table)

    def normalize_table(self):
        # not general function, works only for the 003.txt and 018.txt files
        self.Nij[-2] += self.Nij[-1]
        self.Nij = self.Nij[:-1]

    def test(self, alpha, normalize_flag):
        self.Nij = self.create_contingency_table()
        if normalize_flag:
            self.normalize_table()

        n = np.sum(self.Nij)
        pi_, p_j = np.sum(self.Nij, axis = 1)/n, np.sum(self.Nij, axis =
0)/n
        npipj = n * np.matmul(pi_, p_j)

        Chi2  = np.sum(np.square(self.Nij - npipj)/npipj)
        df = (np.size(self.Nij, axis=0) - 1)*(np.size(self.Nij, axis=1) - 1)
        chi2 = st.chi2.isf(alpha, df)
        p_val = st.chi2.sf(Chi2, df)
        print(f'Manual test: critical value: {chi2}, test value: {Chi2}, p-
value {p_val}, df: {df}')
        self.build_in_test()
```

χ^2	$\chi^2_{0.05;\,11}$	p - **value**
28.46471	19.67514	0.00275

Based on the results from the table provided, **we reject the null hypothesis** that the distribution of word lengths is independent of the text source.

---

4. `At a 5% significance level, test the hypothesis that the mean lengths of words in both texts are equal. Also, determine the p-value of the test. Thoroughly justify why you are using the chosen test!`

To assess the hypothesis that the mean word lengths in two texts are equal, we utilize the **two-sample t-test for unequal variances**, often referred to as **Welch's t-test**. This statistical test is appropriate when comparing the means from two independent samples, which in our case are the two different texts. Our samples may have different variances as they are from distinct texts, possibly with different styles or content, leading to a different spread of word lengths.

$$s_d = \sqrt{\frac{s_X^2}{n} + \frac{s_Y^2}{m}} \quad \text{-} \quad \text{standard error}$$

$$T = \frac{\bar{X}_n - \bar{Y}_m}{s_d} \quad \text{-} \quad \text{test statistic}$$

$$n_d = \frac{s_d^4}{\frac{1}{n-1}\left(\frac{s_X^2}{n}\right)^2 + \frac{1}{m-1}\left(\frac{s_Y^2}{m}\right)^2} \quad \text{-} \quad \text{degrees of freedom}$$

H_0	H_A	**critical region**
$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$\lvert T \rvert > t_{\alpha/2, n_d}$

```
from scipy import stats as st
import numpy as np

class TwoSampleTTest:
    # text is object with necessary data like mean value, variance and so on
    def __init__(self, text1, text2):
        self.text1 = text1
        self.text2 = text2

    def build_in_test(self):
        test_val, p_val = st.ttest_ind([len(word) for word in
self.text1.words], [len(word) for word in self.text2.words], alternative =
'two-sided', equal_var=False)
        print(f'Build-in test: test value: {test_val}, p-value {p_val}')

    def calculate_data(self):
        sd2 = self.text1.variance/self.text1.length +
self.text2.variance/self.text2.length
        self.nd =
sd2**2/((self.text1.variance/self.text1.length)**2/(self.text1.length - 1) +
(self.text2.variance/self.text2.length)**2/(self.text2.length - 1))
        self.test = (self.text1.mean - self.text2.mean)/np.sqrt(sd2)

    def test(self, alpha):
        self.calculate_data()
        crit_val = st.t.isf(alpha/2, self.nd)
        p_val = st.t.sf(np.abs(self.test), self.nd) * 2
        print(f'Manual test: critical value: {crit_val}, test value:
```

```
{self.test}, p-value {p_val}')
        self.build_in_test()
```

$\lvert T \rvert$	$t_{0.025;\ 2096}$	p - **value**
2.29862	1.96109	0.021624

Based on the results from the table provided, **we reject the null hypothesis** that the mean lengths of words in both texts are equal.