Domácí úkol 1

Ladislav Martínek a Richard Werner

April 20, 2019

1 Domácí úkol 1 (6 bodů)

1.1 Úkoly

1. (1b) Z obou datových souborů načtěte texty k analýze. Pro každý text zvlášť odhadněte základní charakteristiky délek slov, tj. střední hodnotu a rozptyl. Graficky znázorněte rozdělení délek slov.
2. (1b) Pro každý text zvlášť odhadněte pravděpodobnosti písmen (symbolů mimo mezery), které se v textech vyskytují. Výsledné pravděpodobnosti graficky znázorněte.
3. (1.5b) Na hladině významnosti 5% otestujte hypotézu, že rozdělení délek slov nezávisí na tom, o který jde text. Určete také p-hodnotu testu.
4. (1.5b) Na hladině významnosti 5% otestujte hypotézu, že se střední délky slov v obou textech rovnají. Určete také p-hodnotu testu.
5. (1b) Na hladině významnosti 5% otestujte hypotézu, že rozdělení písmen nezávisí na tom, o který jde text. Určete také p-hodnotu testu.

1.2 Řešení

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        from scipy import stats
        from scipy.optimize import minimize
```

1.2.1 Zpracování souborů

```
In [2]: K = 15
        L = len("Martínek")
        X = ((K*L*23) % (20)) + 1
        X_file = '0'*(3-len(str(X)))+str(X)+'.txt'
        Y = ((X + ((K*5 + L*7) % (19))) % (20)) + 1
        Y_file = '0'*(3-len(str(Y)))+str(Y)+'.txt'

In [3]: def read_whole_file(filename):
            with open(filename, 'r') as file:
                return file.read()

        xfile = read_whole_file(X_file)
        yfile = read_whole_file(Y_file)
```

1.2.2 Příklad 1

Výpočet výběrové střední hodnoty a výběrového rozptylu

- U rozptylu bylo nutné jako parametr předat hodnotu 1 jako odečtení stupně volnosti, aby byl rozptyl nestranný.

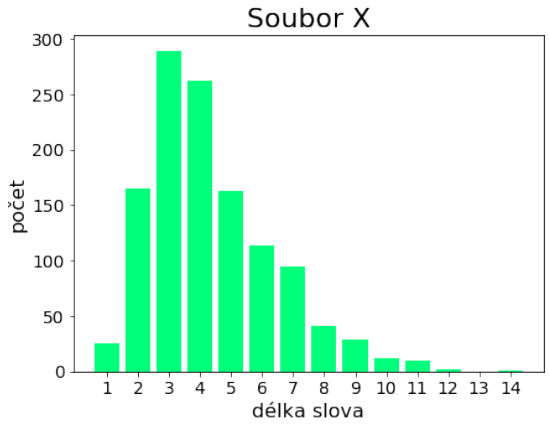
Grafické znázornění četností

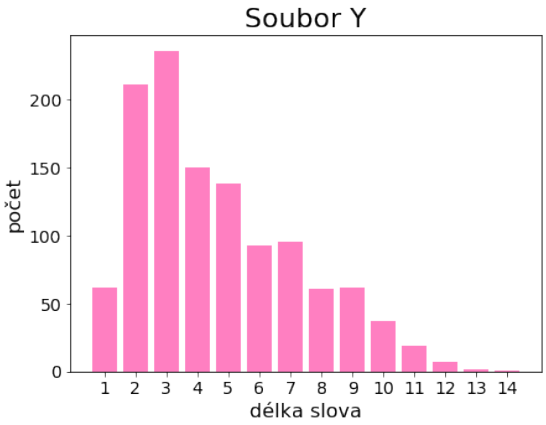
- Vytvořili jsme grafy jak pro každý soubor zvlášť, tak pro soubory jako korpus.
- Jelikož v zadání byla řečena délka slov, vyfiltrovali jsme tečky a čárky přilepené ke slovům.

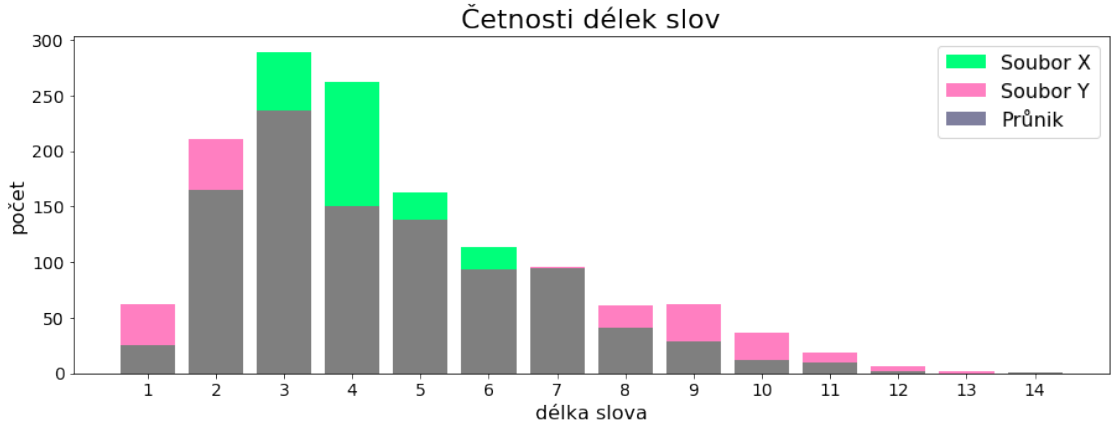
```
In [4]: def get_word_lengths(file_str):
            word_list = file_str.split()
            return [len(word.replace(',', '').replace('.', '')) for word in word_list]

In [5]: Xlengths = get_word_lengths(xfile)
        Ylengths = get_word_lengths(yfile)

        for lengths, name in [(Xlengths, 'X'), (Ylengths, 'Y')]:
            print("Soubor {}\nVýběrový průměr: {}\nVýběrový rozptyl: {}\n"
                  .format(name, np.mean(lengths), np.var(lengths, ddof=1)))
```

```
Soubor X
Výběrový průměr: 4.379966887417218
Výběrový rozptyl: 4.1463091952572455

Soubor Y
Výběrový průměr: 4.6476595744680855
Výběrový rozptyl: 6.8297538874188986
```

```
In [6]: x_length_set, x_length_counts = np.unique(Xlengths, return_counts=True)
        y_length_set, y_length_counts = np.unique(Ylengths, return_counts=True)
```







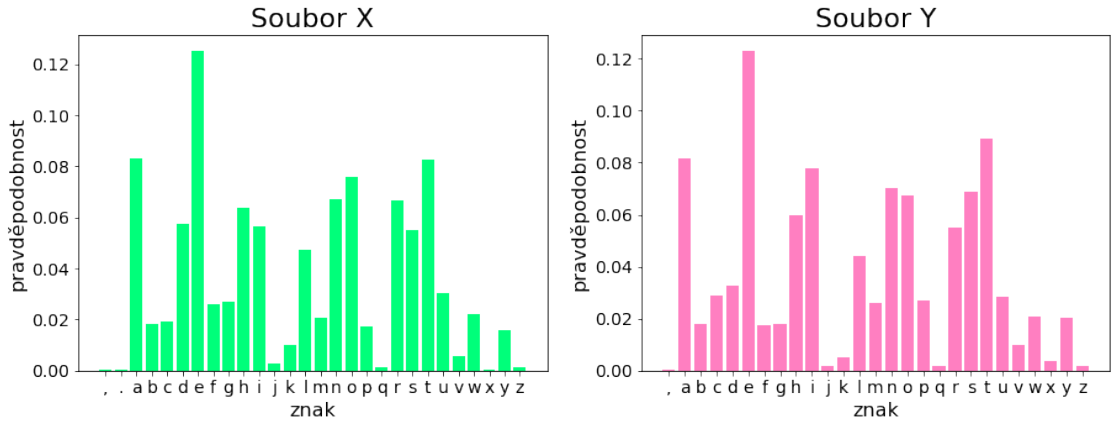
1.2.3 Příklad 2

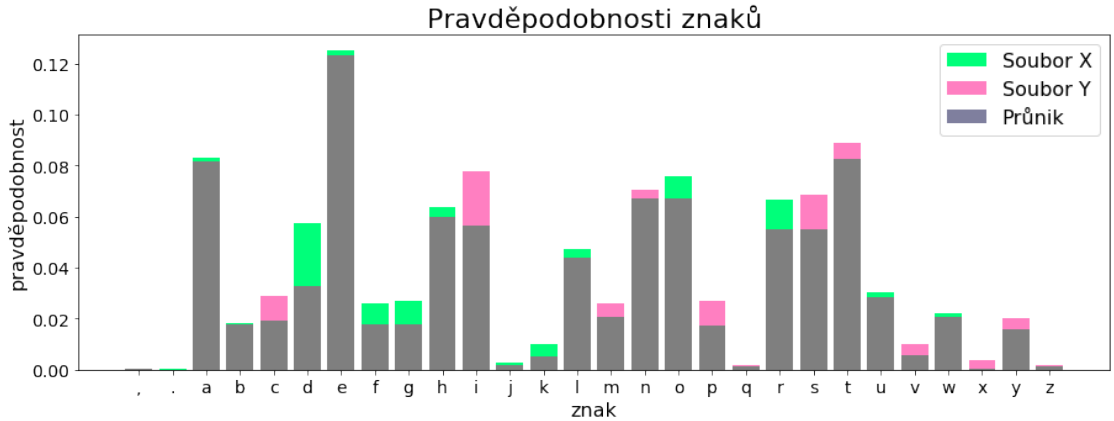
Grafické znázornění pravděpodobností symbolů

- Jelikož je v zadání řečeno "symboly mimo mezery", do statistik jsme započítali i znaky pro tečky a čárky.

```
In [9]: xletters, xletter_cnt = np.unique(list(xfile.replace(" ", "")
                                                    .replace("\n", "").lower()),
                                          return_counts=True)
        yletters, yletter_cnt = np.unique(list(yfile.replace(" ", "")
                                                    .replace("\n", "").lower()),
                                          return_counts=True)
```

```
In [10]: xletter_cnt_prob = xletter_cnt/xletter_cnt.sum()
         yletter_cnt_prob = yletter_cnt/yletter_cnt.sum()
```





1.2.4 Příklad 3

Vytvoření kontingenční tabulky

- V případě četností < 5 se v našem algoritmu podle potřeby automaticky spojí biny.

Test samotný

- Samotný test pak probíhá klasickým způsobem a jeho výstup je znázoněn níže.

```
In [14]: def contingency_table_test(x_labels, x_counts,
                                    y_labels, y_counts,
                                    a = 0.05):
             # merge the two series with letters and their counts
             # (convert to virtual contingency table)
             # then merge bins where (Ni. * N.j) / n < 5
             N, n, n_letters = merge_bins(*two_counts_to_one(x_labels,
                                                             x_counts,
                                                             y_labels,
                                                             y_counts))
             # theoretical frequency
             npp = np.matmul(np.sum(N, axis = 1)/n, np.sum(N, axis = 0)/n)*n
             # count test statictic
             chi, p, d, e = stats.chi2_contingency(N, correction = False)
             # critical value
             cval = stats.chi2.isf(0.05, n_letters)
             return n, npp, chi, cval, p, "Ano" if chi >= cval else "Ne"
```

```
In [15]: n, npp, chi, cval, p, refuse = contingency_table_test(x_length_set,
                                                               x_length_counts,
                                                               y_length_set,
                                                               y_length_counts,
                                                               0.05)
         print("Testová statistika:", chi)
         print("Kritická hodnota", cval)
         print("p-hodnota:", p)
         print("Zamítáme?", refuse)
```

```
Testová statistika: 97.14868027055996
Kritická hodnota 23.684791304840576
p-hodnota: 5.9076610786033936e-15
Zamítáme? Ano
```

Zamítli jsme hypotézu H_0 (tedy nezávislost četností slov jednotlivých délek) ve prospěch H_a (tedy rozdělení četností není nezávislé). p-hodnota je téměř nulová, tedy test je velmi silný.

Tento test jsme neočekávali s tak jasným výsledkem i z důvodu poměrně krátkého textu, ale rozdělení četností pro jednotlivé délky slov není nezávislé. Tedy četnosti délek slov spíš odpovídají konkrétnímu jazyku než konkrétnímu textu a nebo může jít o text od stejného autora například.

1.2.5 Příklad 4

Testování rovnosti středních hodnot

- Jelikož z první úlohy víme, že se rozptyly nerovnají, použili jsme test, který nepředpokládá roznost rozptylů.
- Protože je k výpočtu kritické hodnoty potřeba hodnota n_d jako stupně volnosti, nebylo možné jednoduše použít funkci, ale bylo potřeba hodnotu manuálně vypočítat.

```
In [16]: n = len(Xlengths)
         m = len(Ylengths)
         meanX = np.mean(Xlengths)
         meanY = np.mean(Ylengths)
         varX = np.var(Xlengths, ddof=1)
         varY = np.var(Ylengths, ddof=1)

         sd = np.sqrt(varX/n + varY/m)
         T = (meanX - meanY)/sd
         print("Testová statistika:", T)

         nd = (sd**4)/(1/(n-1)*(varX/n)**2 + 1/(m-1)*(varY/m)**2)

         p_val = 2*stats.t.sf(np.abs(T), df = nd)
         print("p-hodnota:", p_val)


         krit = stats.t.isf(0.05/2, df = nd)
         print("Kriticka hodnota:", krit)

         print("Zamítáme:", "Ano" if np.abs(T) >= krit else "Ne", "\n")

         # the easy way but without the critical value
         #print(stats.ttest_ind(Xlengths, Ylengths, equal_var = False))
```

```
Testová statistika: -2.784098872865385
p-hodnota: 0.005413212227311425
Kriticka hodnota: 1.9610342501606424
Zamítáme: Ano
```

Zamítli jsme hypotézu H_0 (rovnost středních hodnot) ve prospěch H_a (tedy, že se střední hodnoty nerovnají). p-hodnota vyšla 0,5 procenta, což lze také brát za poměrně silný test. Test jsme prováděli s předpokladen nerovnosti rozptylů.

Z vypočítaných výběrových středních hodnot jsme spíš očekávali rovnost středních hodnot, avšak test předpoklad vyvrátil, tedy se nerovnají.

1.2.6 Příklad 5

Vytvoření kontingenční tabulky

- V případě četností < 5 se v našem algoritmu podle potřeby automaticky spojí biny.

Test samotný

- Samotný test pak probíhá klasickým způsobem a jeho výstup je znázoněn níže.

```
In [17]: n, npp, chi, cval, p, refuse = contingency_table_test(xletters,
                                                               xletter_cnt,
                                                               yletters,
                                                               yletter_cnt,
                                                               0.05)
         print()
         print("Testová statistika:", chi)
         print("Kritická hodnota", cval)
         print("p-hodnota:", p)
         print("Zamítáme?", refuse)
```

```
Merging "." and ","
Merging "a" and "."

Testová statistika: 157.1058270988993
Kritická hodnota 38.88513865983007
p-hodnota: 4.0805403234958455e-21
Zamítáme? Ano
```

Zamítli jsme hypotézu H_0 (tedy nezávislost pravděpodobností písmen v textech) ve prospěch H_a (tedy pravděpodobnosti nejsou nezávislá). P-hodnota je téměř nulová, tedy test je velmi silný.

Výsledek testu odpovídá očekávání. Očekávali jsme, že rozdělení písmen bude odpovídat jazyku než jednotlivým textům nebo autorům ve stejném jazyce.