DOLOVÁNÍ Z TEXTU – PROTOKOL (2024/2025)

Cíl projektu

Student má prokázat schopnost aplikace vybraných technik text miningu a zpracování přirozeného jazyka pro základní i pokročilou analýzu textových dat. Student si vybere z nabídky témat, která obsahuje minimální spektrum požadavků pro projekt. Daný projekt zpracuje student v jazyce Python s využitím principů strukturovaného programování a v libovolném vývojovém prostředí. Doporučovaným editorem je Spyder, Jupyter Notebook nebo PyCharm.

Doporučení a upozornění

Pokud pro zpracování projektu student použije programové kódy, které sám nevytvořil, musí jim rozumět, což prokazuje po odevzdání projektu u jeho obhajoby, důsledně je cituje! Necitování je důvodem neudělení zápočtu! Není tolerováno neetické chování. Neetické chování je zejména vydávání práce jiného autora za svou práci nebo její část (projekt, zkoušková/zápočtová písemná práce, bonusový úkol, atd.). Za neetické chování se též považuje podpora neetického chování, jako je propůjčení své práce jinému studentovi za účelem splnění studijní povinnosti (např. poskytnutí vypracovaného projektu) nebo dokonce vypracování úkolu místo jiného studenta za účelem splnění studijní povinnosti. Prohřešek s tímto spojeným bude postoupen disciplinární komisi, která navrhne postih. Konečné rozhodnutí učiní děkan. Reálně hrozí vyloučení ze studia (viz Vysokoškolský zákon).

Pro závěrečný projekt je vyžadováno, aby student využil data, která nejsou použita v rámci přednášek, cvičení, a která u sebe na webu nemají přilinkované zdrojové kódy, viz např. data ze serveru Kaggle. Zároveň je vyžadováno, aby věcné zaměření projektu bylo odlišné od zaměření zmíněné v přednáškách a cvičeních.

- Příklad: v rámci cvičení je demonstrována implementace detekce hamů a spamů. Student musí zvolit detekci dokumentů dle jiných kritérií.

Věcné zaměření tématu spolu s použitými daty student předem konzultuje s vyučujícím, přitom závazně téma projektu nahlašuje do termínu, které je předem určeno cvičícím. Přitom obhajuje uvažovaný způsob analýzy dat spolu s použitými daty.

Při použití nástrojů umělé inteligence se řídí ustanoveními, která jsou platná pro zpracovávání závěrečných bakalářských/diplomových prací, viz kapitola 5 (Výnos děkana FIM č. 6/2023).

Použití jiné šablony než platné pro daný rok zpracování projektu je důvodem pro jeho zamítnutí! Projekt je zasílán k ohodnocení jednou.

Hlavička projektu

Autor projektu:	Adéla Leppeltová
Přihlašovací jméno autora projektu:	leppead1
Téma projektu:	Analýza textu a práce s regulárními výrazy
Akademický rok zpracování:	2024/25
Studijní obor studenta a ročník:	Datová věda, 2. ročník

1. Obecný popis projektu

Charakterizujte v obecné rovině Vaši aplikaci, tj. k čemu slouží (co je jejím cílem):

Aplikace slouží k analýze obsahu nestrukturovaného textu. Cílem je provést předzpracování textu, provést jeho statistickou analýzu, aplikovat metody pro práci s regulárními výrazy, které vybrané části textu extrahují, výsledky těchto metod pak vhodně vizualizovat a interpretovat.

2. Charakteristika korpusu (analyzované texty)

Popište, jakými texty je tvořen Váš korpus, jaká doména je textem (texty) pokryta a uveďte zdroj(e), ze kterého jste text (texty) získali.

Pořadí	Název souboru (s příponou)	Doména	URL
1	bmu58.txt	www.swarthmore.edu	https://www.cs.swarthmore.edu/~bryce/cs35/lab07/test_data/large/bmu58.txt
2			
3			
4			
5			
6			
7			
8			
9			
10			

3. Informační zdroje (dodatečné zdroje, nepovinné)

Do této části projektu vložte zdroje, které jste příp. použili pro zpracování projektu, tj. např. další dodatečné informační zdroje k danému problému nebo zdroje zmiňující programové kódy, kterými jste se inspirovali.

Pořadí	Citace zdroje (alt. URL)	Pro jaký účel je zdroj použit
1	https://www.geeksforgeeks.org/generating-word-cloud-python/	Inspirace k vytvoření vizualizace obsahu textu
2	https://medium.com/@neri.vvo/stop-words-removal-explained-top-3-easy-ways-to-implement-in-python-9bd273bd9c95	Inspirace k odstranění stop slov z textu
3	https://www.nltk.org/	Informování se ohledně NLTK knihovny (import, příklady užití)
4		
5		
6		
7		
8		

4. Popis funkcionalit aplikace - detail

Charakterizujte účel jednotlivých funkcí (procedur) Vašeho projektu.

Pořadí	Název funkce s parametry	Účel funkce
1	numberOfSentences(text)	Vrací počet vět v textu.
2	longWords(text)	Vrací slova s více než dvanácti znaky.
3	names(text)	Vrací nalezené názvy (jména osob) v textu.
4	replace(text)	Vrací originální text a text s nahrazeným slovem.
5	vowels(text)	Vrací věty začínající samohláskou.
6	selectedLetter(text)	Vrací slova končící znakem „l“
7	doubleLetters(text)	Vrací slova, která obsahují dvě stejná písmena za sebou.
8	selectedWords(text)	Vrací věty, které končí řetězcem „disaster“ nebo „success“.
9	quotedPhrases(text)	Vrací věty v uvozovkách.
10	launchWords(text)	Vrací všechny různé tvary řetězce „launch“
11	numbers(text)	Vrací výčet nalezených čísel nebo hlášku „Nebyla nalezena žádná čísla pohromadě.“.
12	threeWords(text)	Vrací věty obsahující pouze tři slova.
13		

5. Vizualizace a interpretace výsledků

Zvolte vhodný vizualizační prostředek, který ukáže výsledky činnosti programu. Výsledky vizualizace vložte do protokolu níže ve formě obrázku/ů, které opatřete komentáři (např. se může jednat o vyhodnocení Word2Vec modelů, statistiky textu, výsledky NER tagování nebo klasifikace textů, výstupy analýzy modelování témat nebo použité regulární výrazy a výsledky extrakce).

Funkce vracející počet vět v textu.

```
# Kolik vět text obsahuje?
def numberOfSentences(text):
    return len(re.split(r"[.?!]", text))

print("Text obsahuje", numberOfSentences(raw_text), "vět.")
```

✓ 0.0s

```
Text obsahuje 160 slov.
```

Funkce vracející slova s více než dvanácti znaky.

```
# Slova s více než 12 znaky
def longWords(text):
    pattern = r'\b\w{12,}\b'
    return (re.findall(pattern, text))


print("Slova s více než deseti znaky:\n", longWords(text))
```

✓ 0.0s Python

```
Slova s více než deseti znaky:
 ['organizations', 'organizational', 'organizational', 'organization', 'organization', 'organization', 'organizations', 'headquarters',
'organizations', 'organization', 'organization', 'organization', 'organization', 'organizations', 'organization', 'organization',
'organization', 'communication', 'organizations', 'organizations', 'organizations', 'organizations', 'organizations', 'organizations',
'organizations', 'organizations', 'organizational', 'organizations', 'construction', 'organization', 'organizations', 'organization',
'organization', 'organizations', 'organization', 'organizational', 'organization', 'organizations', 'organizations', 'organizations',
'organizations', 'organization', 'organizations', 'organization', 'alternatives', 'organizations', 'expectations', 'consultative',
'perspectives', 'nonconformity', 'whistleblowers', 'whistleblowers', 'whistleblowers', 'organizations', 'organization', 'significance',
'consequences', 'temperatures', 'organizational', 'organization', 'organization', 'organization', 'whistleblower', 'organizations',
'responsibility', 'organization', 'stakeholders', 'organization', 'responsibility', 'organization', 'organization', 'organizations',
'requirements', 'organization', 'organization', 'organizations', 'organization', 'responsibility', 'organization', 'relationship',
'relationship', 'relationship', 'organizations', 'cancellation', 'organizations', 'organization', 'relationship', 'relationship',
'organizational', 'communication', 'organization', 'incomprehensible', 'organization', 'organizations', 'organization', 'organization',
'communication', 'organization', 'organization', 'organization', 'organization', 'whistleblower', 'whistleblower', 'organization',
'organizations', 'organization']
```

Funkce vracející názvy (jména).

```
# Názvy, jména osob
def names(text):
    pattern = r'\b[A-Z][a-z]+\s[A-Z][a-z]+\b'
    return (re.findall(pattern, text))

print("Nalezené názvy:\n", names(raw_text))
```

✓ 0.0s Python

```
Nalezené názvy:
 ['The Space', 'Shuttle Challenger', 'Richard Cook', 'Roger Boisjoly', 'Challenger Space', 'Morton Thiokol', 'Marshall Space', 'Flight
Center', 'Johnson Space', 'At Morton', 'At Morton', 'Charles Locke', 'Jerry Mason', 'Neil Armstrong', 'When Thiokol', 'United States',
'The Challenger', 'Both Thiokol', 'In Challenger', 'If Thiokol', 'Richard Cook', 'Roger Boisjoly', 'Seals Engineer', 'Both Cook', 'In
Thiokol', 'United States', 'The Challenger']
```

Funkce nahrazující řetězec „NASA“ za řetězec „Space Agency“. Je vypsán originální text a text s nahrazeným řetězcem. Pro přehlednost je zobrazena jen první věta.

```python
# Nahrazení "NASA" za "Space Agency"
def replace(text):
    pattern = r'\bNASA\b'
    return (re.sub(pattern, "Space Agency", text))


print("Originální text:", raw_text[:130],"\n")
print("Text s nahrazeným řetězcem:", replace(raw_text)[:138])
```

✓ 0.0s Python

```
Originální text:
The Space Shuttle Challenger Disaster was a preventable disaster that NASA tried to cover up by calling it a mysterious accident.

Text s nahrazeným řetězcem:
The Space Shuttle Challenger Disaster was a preventable disaster that Space Agency tried to cover up by calling it a mysterious
accident.
```

Funkce vracející věty, které začínají na samohlásky.

```python
# Věty obsahující slova začínající na samohlásky
def vowels(text):
    pattern = r'\b[AEIOU]\w*\b'
    vowel_sentences = []

    for sentence in re.split(r'[.!?]', text):
        if re.search(pattern, sentence):
            vowel_sentences.append(sentence.strip())

    return vowel_sentences

# Zobrazení prvních pěti vět
print("Věty začínající na samohlásky:\n", vowels(raw_text)[:5])
```

✓ 0.0s Python

```
Věty začínající na samohlásky:
 ['One of the many key topics behind the Challenger disaster is the organizational culture', 'One of the aspects of an organizational
culture is the observable culture of an organization that is what one sees and hears when walking around an organization', 'In the
Challenger Space Shuttle incident there were mainly four organizations thrown together to form one, Morton Thiokol, Marshall Space
Flight Center, Johnson Space Center and NASA Headquarters', 'All of these organizations had the same type of stories to be told', 'At
Morton Thiokol, they talked about their product and their big deal, which they received from NASA']
```

Funkce vracející slova, která končí znakem „l“.

```python
# Slova, která končí znakem 'l'
def selectedLetter(text):
    # pattern = r'\b\w+\b'
    return ([word for word in re.split(r'\W+', text) if word.endswith('l')])


print("Slova končící znakem 'l':\n", selectedLetter(text))
```

✓ 0.0s Python

```
Slova končící znakem 'l':
 ['real', 'thankful', 'organizational', 'organizational', 'thiokol', 'marshall', 'all', 'thiokol', 'deal', 'thiokol', 'neil', 'all',
'goal', 'ritual', 'successful', 'ritual', 'special', 'all', 'thiokol', 'essential', 'organizational', 'thiokol', 'cancel', 'well',
'successful', 'thiokol', 'fall', 'ethical', 'thiokol', 'thiokol', 'thiokol', 'organizational', 'technical', 'rational', 'analytical',
'still', 'managerial', 'until', 'all', 'technical', 'thiokol', 'thiokol', 'final', 'logical', 'trial', 'thiokol', 'thiokol', 'thiokol',
'organizational', 'thiokol', 'ethical', 'ethical', 'ethical', 'social', 'social', 'successful', 'thiokol', 'thiokol', 'thiokol',
'ethical', 'unethical', 'thiokol', 'unethical', 'thiokol', 'thiokol', 'technical', 'total', 'thiokol', 'technical', 'thiokol',
'thiokol', 'technical', 'technical', 'technical', 'level', 'ethical', 'level', 'level', 'organizational', 'general', 'general', 'tell',
'illegal', 'unethical', 'unethical', 'will', 'all', 'feel', 'educational', 'feel', 'material']
```

Funkce vracející slova, která obsahují dvě stejná písmena za sebou. Je zobrazen počet těchto slov a pro přehlednost je vypsáno prvních 40 slov.

```python
# Hledání všech slov, která obsahují dvě stejná písmena za sebou
def doubleLetters(text):
    pattern = r'\b\w*(\w)\1\w*\b'
    return ([match.group(0) for match in re.finditer(pattern, text)])

print("Počet slov, která obsahují dvě stejná písmena za sebou:",len(doubleLetters(text)))
print("Slova obsahující dvě stejná písmena za sebou:\n", doubleLetters(text)[:40])
```

✓ 0.0s Python

```
Počet slov, která obsahují dvě stejná písmena za sebou: 215
Slova obsahující dvě stejná písmena za sebou:
 ['shuttle', 'challenger', 'calling', 'accident', 'cook', 'lessons', 'challenger', 'sees', 'challenger', 'shuttle', 'marshall', 'all',
'missions', 'moon', 'been', 'jerry', 'been', 'staff', 'all', 'follow', 'attain', 'successful', 'shuttle', 'finally', 'carry',
'communication', 'difference', 'between', 'shuttle', 'different', 'shuttle', 'all', 'challenger', 'mission', 'challenger', 'challenger',
'mission', 'employees', 'challenger', 'different']
```

Funkce vracející věty, které končí řetězcem „disaster“ a „success“.

```python
# Extrakce vět, které končí konkrétními řetězci (např. "disaster", "success")
def selectedWords(text):
    end_words = ["disaster", "success"]
    pattern = r'([^.]*\b(?:'  + '|'.join(end_words) + r')\b[^.]*\.)'

    matched_sentences = []
    for line in text.split('\n'):
        match = re.search(pattern, line)
        if match:
            matched_sentences.append(match.group(1))

    return (matched_sentences)

print("Věty končící řetězcem 'disaster' a 'success':\n", selectedWords(raw_text))
```

✓ 0.0s Python

```
Věty končící řetězcem 'disaster' a 'success':
 ['The Space Shuttle Challenger Disaster was a preventable disaster that NASA tried to cover up by calling it a mysterious accident.',
'One of the many key topics behind the Challenger disaster is the organizational culture.', " When Thiokol and NASA first started to
plan for Challenger's mission, it was part of their core culture, which ultimately caused the Challenger disaster.", ' This was also
very important in trying to prevent the Challenger disaster.', ' Each time they lowered their expectations of the weather and
conditions, this eventually led to the disaster.', ' All of the previous missions were a success, but from a technical standpoint each
mission was a more and more devastating disaster.', " Sixth, don't be afraid of error; let trial and error be a path of success, if
lives are not at stake.", ' In the Challenger disaster there where two main whistleblowers, Richard Cook who worked for NASA and Roger
Boisjoly who was the SRM Seals Engineer with Thiokol.', ' In the case of the Challenger disaster, NASA and Thiokol assumed the role of
the defensive strategy.', ' In the Challenger disaster this relationship was thrown into total chaos.', "In the case of the Challenger
disaster, NASA's matrix organizational structure was not in perfect alignment.", 'The structure system of this organization is not to
blame; it is top executives of the organization who are at fault for this preventable disaster.', 'From watching the Challenger
disaster, I mainly have learned that I would not want to be apart of an organization that practices unethical behaviors.', 'The
Challenger disaster was probably one of the most preventable disasters that our nation has ever faced in dealing with an organization.',
'I feel as though that the movie of the Challenger disaster was very interesting and educational to watch.']
```

Funkce vracející věty v uvozovkách.

```python
# Nalezení vět v uvozovkách
def quotedPhrases(text):
    pattern = r'"(.*?)"'
    return (re.findall(pattern, text))

print("Věta (fráze) v uvozovkách:\n", quotedPhrases(raw_text))
```

✓ 0.0s

```
Věta (fráze) v uvozovkách:
 ['do as I say and not what I do.']
```

Funkce vracející všechny tvary vybraného řetězce.

```
# Nalezení různých tvarů řetězce 'launch'
def launchWords(text):
    return (re.findall(r"launch\w+", text))

print("Tvary řetězce 'launch':\n",launchWords(text))
```

✓ 0.0s

```
Tvary řetězce 'launch':
 ['launches', 'launches', 'launches', 'launching']
```

Funkce hledající dvě a více čísel pohromadě. Vrací výčet nalezených čísel nebo hlášku „Nebyla nalezena žádná čísla pohromadě.“.

```
# Nalezení dvou a více čísel v textu pohromadě
def numbers(text):
    pattern = r'\d{2,}'
    matches = re.findall(pattern, text)

    if matches:
        result = f"Nalezená čísla: {', '.join(matches)}"
    else:
        result = "Nebyla nalezena žádná čísla pohromadě."

    return(result)

print(numbers(text))
```

✓ 0.0s

```
Nebyla nalezena žádná čísla pohromadě.
```

Funkce vracející věty, které obsahují přesně tři slova.

```
# Extrakce vět obsahujících tři slova
def threeWords(text):
    three_word_sentence = r'\b([A-Z][a-z]*\s+\w+\s+\w+[.!?])'
    three_word_sentences = re.findall(three_word_sentence, text)
    return (sorted(set(three_word_sentences)))

print(threeWords(raw_text))
```

✓ 0.0s

```
['Engineer with Thiokol.', 'Thiokol and NASA.']
```

6. Deklarace k použití nástrojů umělé inteligence (UI)

Deklarujte, zda jste pro zpracování projektu použili nějaký nástroj umělé inteligence (ANO/~~NE~~). Pokud jste nástroj UI použili, uveďte, o který nástroj se jedná, kde ho lze stáhnout, jakým způsobem ho získat (URL adresa), a jakým způsobem přesně jste ho použili (např. screenshoty ze zadaných dotazů do aplikace UI).

Ano, využila jsem nástroj ChatGPT (je dostupný na URL adrese: https://chatgpt.com/).

Použila jsem ho pro některé extrakce pomocí regulárních výrazů a pro ověření, zda je regulární výraz správně.

Screenshoty zadaných dotazů: