Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

ЗВІТ

Дисципліна «Розробка прикладних програм»

Робота №4

Тема «Оброблення природньої мови»

Виконав варіант 19

Студент КНТ-122 Онищенко О. А.

Прийняли

Викладач Дейнега Л. Ю.

2024

МЕТА РОБОТИ

Ознайомитись з основними інструментами оброблення природньої мови, які входять у склад бібліотеки NLTK мови програмування Python.

Навчитися розв’язувати актуальні практичні завдання у галузі оброблення природної мови за допомогою бібліотеки NLTK.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Реалізувати створення програмних відповідей Українською мовою на основі виконання SQL-запитів до бази даних, розробленої в лабораторній роботі №3.

ТЕКСТ ФАЙЛУ

```
NAME='data'
USER='root'
PASSWORD='1313'
HOST='localhost'
PORT='3306'

import mysql.connector

db=mysql.connector.connect(database=NAME,user=USER,password=PASSWORD,ho
st=HOST,port=PORT)
c=db.cursor()

CLIENTS_TABLE='finance_client'
PAYMENTS_TABLE='finance_payment'
PERIODIC_PAYMENTS_TABLE='finance_periodicpayment'

import nltk

while 1:
    HELP_MESSAGE='''Загальні команди:
- поможи АБО допомога: виведення цього повідомлення
- вийти АБО вихід: закриття програми
- користувачі: виведення списку користувачів
- періодичні платежі: виведення усіх запланованих платежів
- платежі: виведення історії усіх платежів
Користувацькі команди (потребують імені користувача у запиті):
- баланс: виведення балансу користувача
- ліміт АБО кредит: виведення кредитного ліміту користувача
```

```
- менеджер АБО адміністратор: виведення статусу користувача
- користувач: виведення усіх даних про користувача
'''.strip()

    user_query=input('> ')
    words=nltk.word_tokenize(user_query.lower())
    vocab=nltk.Text(words).vocab()

    def check_any(
        terms:list[str],
        words:list[str]=words,
    ):
        found=False
        for term in terms:
            for word in words:
                if term in word:
                    found=True
                    break
        return found

    def check_all(
        terms:list[str],
        words:list[str]=words,
    ):
        all_good=True
        for term in terms:
            found=False
            for word in words:
                if term in word: found=True
            if not found:
                all_good=False
                break
        return all_good

    def clean_query(
        query:list[str],
    ):
        return [
            word for word in query
            if 'буд' not in word
            and 'ласк' not in word
            and 'про' not in word
            and 'дан' not in word
            and 'інф' not in word
            and 'чи' not in word
        ]

    def execute_query(
        query:str,
    ):
        c.execute(query)
        rows=c.fetchall()
        return rows

    def get_client_by_id(
        client_id:int,
    ):
        q=f'SELECT id,name FROM {CLIENTS_TABLE} WHERE id={client_id}'
```

```
        return execute_query(q)[0]

    def get_client_by_name(
        user_name:str,
    ):
        q=f'SELECT name,balance,credit,manager FROM {CLIENTS_TABLE}
WHERE name="{user_name}"'
        return execute_query(q)[0]

    def get_balance(
        user_name:str,
    ):
        q=f'SELECT name,balance FROM {CLIENTS_TABLE} WHERE
name="{user_name}"'
        return execute_query(q)[0]

    def get_credit(
        user_name:str,
    ):
        q=f'SELECT name,credit FROM {CLIENTS_TABLE} WHERE
name="{user_name}"'
        return execute_query(q)[0]

    def get_manager(
        user_name:str,
    ):
        q=f'SELECT name,manager FROM {CLIENTS_TABLE} WHERE
name="{user_name}"'
        return execute_query(q)[0]

    if check_any(['вих','вий']): break
    elif check_any(['пом','доп','ком']): print(HELP_MESSAGE)
    elif check_any(['прив','віт']): print(user_query)
    elif check_any(['користувачі']) or check_all(['всі','кор']) or
check_all(['усі','кор']) or (check_any(['кор']) and len(vocab)==1):
        q=f'SELECT name,balance,credit,manager FROM {CLIENTS_TABLE}'
        rows=execute_query(q)
        print(f'Користувачі ({len(rows)}):')
        for name,balance,credit,manager in rows:
            print(f'- {name} має {balance} на рахунку, {credit}
кредитного ліміту, та {"Є" if manager else "НЕ є"} менеджером')
    elif check_any(['пер']):
        q=f'SELECT amount,purpose,period,next_date,client_id FROM
{PERIODIC_PAYMENTS_TABLE}'
        rows=execute_query(q)
        print(f'Періодичні платежі ({len(rows)}):')
        for amount,purpose,period,next_date,client_id in rows:
            client_id,client_name=get_client_by_id(client_id)
            print(f'- {purpose} кожен {"день" if period=="Day" else
"місяць" if period=="Month" else "рік"}, наступний платіж
{next_date.strftime("%d.%m.%Y")} для {client_name}')
    elif check_any(['пл']):
        q=f'SELECT timestamp,purpose,amount,client_id,kind,operation
FROM {PAYMENTS_TABLE}'
        rows=execute_query(q)
        print(f'Платежі ({len(rows)}):')
        for timestamp,purpose,amount,client_id,kind,operation in rows:
            client_id,client_name=get_client_by_id(client_id)
```

```
            print(f'- {purpose} за {timestamp.strftime("%d.%m.%Y о
%H:%M:%S")} на {amount} від {client_name}, {"одноразове" if
kind=="Single" else "періодичне"} {"зняття" if operation=="Withdrawal"
else "внесення"}')
    elif check_any(['бал','гро']):
        stripped_words=clean_query([w for w in words if 'бал' not in w
and 'гро' not in w])
        found=False
        for word in stripped_words:
            try:
                name,balance=get_balance(word)
                print(f'{name} має {balance} на рахунку')
                found=True
                break
            except: continue
        if not found: print('користувача не знайдено')
    elif check_any(['кр','лім']):
        stripped_words=clean_query([w for w in words if 'кр' not in w
and 'лім' not in w])
        found=False
        for word in stripped_words:
            try:
                name,balance=get_credit(word)
                print(f'{name} має {balance} кредитного ліміту')
                found=True
                break
            except: continue
        if not found: print('користувача не знайдено')
    elif check_any(['мен','адм']):
        stripped_words=clean_query([w for w in words if 'мен' not in w
and 'адм' not in w])
        found=False
        for word in stripped_words:
            try:
                name,manager=get_manager(word)
                print(f'{name} {"Є" if manager else "НЕ є"}
менеджером')
                found=True
                break
            except: continue
        if not found: print('користувача не знайдено')
    else:
        stripped_words=clean_query([w for w in words if 'кор' not in
w])
        found=False
        for word in stripped_words:
            try:
                name,balance,credit,is_manager=get_client_by_name(word)
                print(f'{name} має {balance} на рахунку, {credit}
кредитного ліміту, та {"Є" if is_manager else "НЕ є"} менеджером')
                found=True
                break
            except: continue
        if not found: print('користувача не знайдено')
```

РЕЗУЛЬТАТИ ВИКОНАННЯ

Процес роботи з програмою наведено нижче:

```
> привіт
привіт
> поможи
Загальні команди:
- поможи АБО допомога: виведення цього повідомлення
- вийти АБО вихід: закриття програми
- користувачі: виведення списку користувачів
- періодичні платежі: виведення усіх запланованих платежів
- платежі: виведення історії усіх платежів
Користувацькі команди (потребують імені користувача у запиті):
- баланс: виведення балансу користувача
- ліміт АБО кредит: виведення кредитного ліміту користувача
- менеджер АБО адміністратор: виведення статусу користувача
- користувач: виведення усіх даних про користувача
> покажи список усіх користувачів будь ласка
Користувачі (6):
- seesmof має 7 на рахунку, 5 кредитного ліміту, та НЕ є менеджером
- seesm має 0 на рахунку, 0 кредитного ліміту, та НЕ є менеджером
- Oleg має 0 на рахунку, 9 кредитного ліміту, та НЕ є менеджером
- admin має 1168 на рахунку, 336 кредитного ліміту, та Є менеджером
- new має 0 на рахунку, 0 кредитного ліміту, та НЕ є менеджером
- newuser має 2 на рахунку, 1272 кредитного ліміту, та НЕ є менеджером
> виведи усю інформацію про користувача під назвою seesmof
seesmof має 7 на рахунку, 5 кредитного ліміту, та НЕ є менеджером
> покажи усі періодичні платежі будь ласочка
Періодичні платежі (5):
- taxes кожен день, наступний платіж 19.11.2024 для Oleg
- Monthly taxes JESUS THANK YOU LORD GOD ALMIGHTY ALLELUJAH AMEN кожен місяць, наступн
ий платіж 19.01.2026 для admin
- Daily taxes ALLELUJAH JESUS THANK YOU LORD GOD MOST HIGH ALLELUJAH AMEN кожен день,
наступний платіж 08.01.2025 для admin
- Yearly tax JESUS THANK  YOU LORD GOD ALMIGHTY ALLELUJAH AMEN кожен рік, наступний пл
атіж 19.11.2029 для admin
- Taxes кожен місяць, наступний платіж 19.12.2024 для newuser
```

```
> виведи детальну інформацію про усі платежі будь ласка
Платежі (138):
- ALLELUJAH за 19.11.2024 о 20:09:36 на 12 від Oleg, одноразове внесення
- reaping за 19.11.2024 о 20:11:26 на 12 від Oleg, одноразове зняття
- ALLELUUJAH за 19.11.2024 о 20:21:22 на 12 від Oleg, одноразове зняття
- AMEN GREAT JESUS IS my LORD GOD MOST HIGH за 19.11.2024 о 20:21:32 на 12 від Oleg, о
дноразове внесення
- ALLELUJAH PRAISE KING JESUS CHRIST our HOLY LORD GOD MOST HIGH AMEN за 19.11.2024 о
20:24:39 на 3 від Oleg, одноразове зняття
- ALLELUJAH PRAIES KING JESUS CHRIST our HOLY LORD GOD ALMIGHTY KING OF KINGS AND LORD
 OF LORDS AMEN за 19.11.2024 о 20:46:15 на 12 від admin, одноразове внесення
- AMEN AND AMEN ALLELUJAH JESUS THANK YOU LORD GOD MOST HIGH ALLELUJAH AMEN за 19.11.2
024 о 20:48:07 на 12 від admin, одноразове зняття
- ALLELUJAH прошу ПАНЕ ІСУСЕ ПОМОЖИ ГОСПОДИ ІСУСЕ ХРИСТЕ АМІНЬ за 19.11.2024 о 20:48:3
2 на 12 від admin, одноразове внесення
- AMEN AND AMEN PRAISE KING JESUS CHRIST our HOLY LORD GOD MOST HIGH ALLELUJAH AMEN за
 19.11.2024 о 20:49:32 на 12 від admin, одноразове зняття
- taxes за 19.11.2024 о 21:42:17 на 3 від Oleg, періодичне зняття
- taxes за 19.11.2024 о 21:47:21 на 3 від Oleg, періодичне зняття
- taxes за 19.11.2024 о 21:47:23 на 3 від Oleg, періодичне зняття
- AMEN AND ALLELUJAH за 19.11.2024 о 21:47:50 на 3 від Oleg, одноразове зняття
- For taxes JESUS THANK YOU LORD GOD ALMIGHTY ALLELUJAH AMEN за 19.11.2024 о 21:50:08
на 172 від admin, одноразове внесення
- Taxes за 19.11.2024 о 21:50:12 на 3 від admin, періодичне зняття
- Taxes за 19.11.2024 о 21:50:14 на 3 від admin, періодичне зняття
- Taxes за 19.11.2024 о 21:50:15 на 3 від admin, періодичне зняття
- Taxes за 19.11.2024 о 21:53:42 на 3 від admin, періодичне зняття
- Daily taxes ALLELUJAH JESUS THANK YOU LORD GOD MOST HIGH ALLELUJAH AMEN за 19.11.202
4 о 22:52:48 на 1 від admin, періодичне зняття
- Groceries shopping за 19.11.2024 о 23:16:38 на 12 від newuser, одноразове внесення
- Taxi за 19.11.2024 о 23:18:23 на 7 від newuser, одноразове зняття
- Taxes за 19.11.2024 о 23:22:03 на 3 від newuser, періодичне зняття
> який кредитний ліміт має користувач під назвою oleg?
Oleg має 9 кредитного ліміту
> а який баланс має користувач з ім'ям admin?
admin має 1168 на рахунку
```

```
> чи є користувач seesm менеджером?
seesm НЕ є менеджером
> а чи є користувач admin адміністратором?
admin Є менеджером
```

КОНТРОЛЬНІ ПИТАННЯ

Яким чином виконується оброблення природної мови в бібліотеці NLTK?

Бібліотека NLTK для мови програмування Python впроваджує багато способів обробки природньої мови. Основні засоби це **токенизація**, **чанкування** та **виділення частин мови**.

Для **токенізації** можна застосувати розділ бібліотеки `nltk.tokenize` та вбудовані у неї функції `sent_tokenize()` для розділення тексту на речення або `word_tokenize()` для розділення заданого тексту на список слів.

Для **виділення частин мови** можна застосувати функцію бібліотеки `nltk` під назвою `pos_tag()` яка приймає список токенізованих слів тексту як параметр.

Для **чанкування** можна застосувати регулярні вирази для створення граматики як програмі потрібно виділяти чанки з тексту. Прикладом граматики чанку може бути `"NP: {<DT>?<JJ>*<NN>}"`. Після створення змінної граматики чанкування можна створити оброблювач чанків застосуванням функції бібліотеки `nltk` під назвою `RegexpParser()`, яка приймає граматику чанкування як аргумент. Для створення дерева чанків тепер необхідно використати метод prase об’єкту оброблювача чанків вигляду `chunk_parser.parse(text_with_pos_tags)`. При виведенні такого об’єкту через методом `draw()` на екрані має з’явитися дерево чанків.

Джерело

Які основні модулі входять до складу бібліотеки NLTK?

До складу бібліотеки оброблення природньої мови NLTK для мови програмування Python входять такі основні модулі:

- nltk.book
- nltk.chunk
- nltk.grammar
- nltk.metrics

- nltk.misc
- nltk.parse
- nltk

Джерело

Яким чином можна використати готові тексти для роботи програми на Python?

Для використання готових текстів з бібліотеки оброблення природньої мови NLTK для мови програмування Python потрібно спочатку завантажити їх викликом методу `download()` на об'єкт `nltk`. Після завантаження необхідних ресурсів можна імпортувати та використати їх включаючи відповідні назви до файлу:

```
from nltk.corpus import gutenberg
HOLY_BIBLE=gutenberg.words('Bible-KJV.txt')
print(HOLY_BIBLE)
```

Джерело

Яким чином розбити текст на лексеми?

Для розбиття заданого тексту на лексеми (окремі слова) використовуючи бібліотеку оброблення природньої мови NLTK для мови програмування Python можна використати вбудований метод частини бібліотеки nltk.tokenize під назвою word_tokenize(text: str):

```
REVELATION_22_UKRK='''
Одкриттє 22:1 І показав мені чисту ріку води життя, ясну як хришталь,
що виходила з престола Божого і Агнцевого.
Одкриттє 22:2 А посеред улиці його, та й по сей і по той бік ріки -
дерево життя, що родить овощі дванайцять (раз), і що місяця свій овощ
дає, а листє з дерева на сціленнє поган.
Одкриттє 22:3 І вже більш не буде жодного проклону; а престол Бога і
Агнця буде в ньому, і слуги Його служити муть Йому.
Одкриттє 22:4 І побачять лице Його, а імя Його на чолах їх.
Одкриттє 22:5 І ночі не буде там; і не потрібувати муть сьвічника і
сьвітла сонця, бо Господь Бог осьвічує їх; і царювати муть по вічні
віки.
```

Одкриттє 22:6 І рече менї: Сї слова вірні і правдиві; і Господь, Бог сьвятих пророків, післав ангела свого, показати слугам своїм, що має незабаром бути.
Одкриттє 22:7 Ось прийду незабаром. Блаженний, хто хоронить слова пророцтва книги сієї.
Одкриттє 22:8 А я Йоан, що бачив се і чув; і коли чув я, і бачив, упав я поклонитись перед ногами ангела, що менї се показував.
Одкриттє 22:9 І каже менї: нї, глянь, я бо слуга-товариш твій, і братів твоїх пророків, і тих, що хоронять слова книги сієї: Богу поклони ся.
Одкриттє 22:10 І глаголе менї: Не печатай слів пророцтва книги сієї; час бо близько.
Одкриттє 22:11 Хто з'обіжає, нехай ще з'обіжає, і хто поганий, нехай ще опоганюєть ся; і хто праведний, нехай ще оправдуєть ся, і хто сьвятий, нехай ще осьвячуєть ся.
Одкриттє 22:12 І ось, я прийду хутко, і заплата моя зо мною, щоб віддати кожному, яко ж буде дїло його.
Одкриттє 22:13 Я Альфа і Омега, почин і конець, Первий і Останнїй.
Одкриттє 22:14 Блаженні, що творять заповідї Його, щоб мали власть до дерева життя, і увійшли ворітьми в город.
Одкриттє 22:15 А на дворі будуть пси, і чарівники, і перелюбники, і душегубцї, і ідолські служителї, і кожен, хто любить і робить лож.
Одкриттє 22:16 Я Ісус післав ангела мого, сьвідкувати вам усе по церквах. Я - корінь і рід Давидів, зоря ясна і рання.
Одкриттє 22:17 А Дух і невіста глаголють: Прийди! і хто чує, нехай каже: Прийди! Хто жадний, нехай прийде, а хто хоче, нехай приймає воду життя дармо.
Одкриттє 22:18 Сьвідкую ж також кожному, хто слухає словес пророцтва книги сієї: коли хто доложить до сього, доложить йому Бог і пораз, що написані в книзї сїй.
Одкриттє 22:19 Коли ж хто уйме від словес книги пророцтва сього, уйме Бог часть його з книги життя, і з города сьвятого, та й з того, що написано в книзї сїй.
Одкриттє 22:20 Сей, що про се сьвідкує, глаголе: Так, прийду хутко! Амінь. О, прийди, Господи Ісусе!
Одкриттє 22:21 Благодать Господа нашого Ісуса Христа з усїма вами. Амінь.
'''

```
import nltk
words=nltk.word_tokenize(REVELATION_22_UKRK.replace('\n',' ').strip())
print(words)
```

Яким чином побудувати граматику в NLTK?

Для побудови граматики використовуючи бібліотеку оброблення природньої мови NLTK для мови програмування Python можна використати вбудований метод `fromstring()` частини бібліотеки під назвою `nltk.CFG`:

```
import nltk.CFG
g=CFG.fromstring("""
S->NP VP
PP->P NP
```

```
NP->Det N | NP PP
VP->V NP | VP PP
Det -> 'a' | 'the'
""")
print(g, g.start(), g.productions())
```

Джерело

Що таке біграми та яким чином їх визначити у тексті?

Біграми у бібліотеці оброблення природньої мови NLTK для мови програмування Python є парами послідовних слів. Для створення списку біграм потрібно спочатку створити список токенів з вхідного рядку. Далі застосувати метод `bigrams()` з частини бібліотеки `nltk.util` таким чином:

```
from nltk.tokenize import word_tokenize
from nltk.util import bigrams
text="ІСУС ХРИСТОС ГОСПОДЬ БОГ ВСЕМОГУТНІЙ ВСЕВИШНІЙ ВСЕСИЛЬНИЙ АМІНЬ
СЛАВА ГОСПОДУ ІСУСУ ХРИСТУ НАВІКИ ВІЧНІ АМІНЬ"
tokens=word_tokenize(text)
bigrams_list=list(bigrams(tokens))
for b in bigrams_list:
	print(b)
```

Джерело

Що таке чанкінг та яким чином він виконується?

Чанкінг або чанкування у бібліотеці оброблення природньої мови NLTK для мови програмування Python є процесом розділення тексту на чанкки або частини в залежності від їх відповідної синтаксичної форми.

```
import nltk
from nltk.chunk import RegexpParser
from nltk.tokenize import word_tokenize
text='Some example text hopefully this will work LORD JESUS please help
me FATHER GOD in JESUS HOLY NAME AMEN.'
tokens=word_tokenize(text)
pos_tags=nltk.pos_tag(tokens)
chunk_patterns=r'''
    NP: {<DT>?<JJ>*<NN>}
    VP: {<VB.*><NP|PP>}
'''
chunk_parser=RegexpParser(chunk_patterns)
res=chunk_parser.parse(pos_tags)
print(res)
```

Джерело

Яким чином та за допомогою яких засобів виконується семантична інтерпретація?

Семантична інтерпретація заданого тексту за допомогою бібліотеки оброблення природньої мови NLTK для мови програмування Python може бути здійснена шляхом використання вбудованої граматики та оброблювача природньої мови `sql0.fcfg` за допомогою використання завантажувача обробників `nltk.load_parser` таким чином:

```
from nltk import load_parser
p=load_parser('grammars/book_grammars/sql0.fcfg')
q='What cities are located in Greece'
t=list(p.parse(q.split()))
a=t[0].label()['SEM']
a=[s for s in a if s]
res=' '.join(a)
print(res)
from nltk.sem import chat80
nr=chat80.sql_query('corpora/city_database/city.db',res)
print(', '.join([w[0] for w in nr]))
```

Джерело

Що таке контекстно-вільна граматика? Наведіть приклади.

Контекстно-вільна граматика (Англійською **Context-Free Grammar** або **CFG**) визначає формальну мову. У формальній природній мові граматика визначається за чіткими правилами. Зазвичай у природній мови елементи складають речення, які не залежать від контексту. Звідси і походить ім’я *контекстно-вільна*. Джерело

Прикладом визначення та використання контекстно-вільною граматики у мові програмування Python через використання бібліотеки оброблення природної мови NLTK може бути наступний застосунок:

```
import nltk
grammar_string='''
```

```
S -> NP VP
VP -> V NP | V NP PP
PP -> P NP
V -> "saw" | "ate" | "walked"
NP -> "John" | "Mary" | "Bob" | Det N | Det N PP
Det -> "a" | "an" | "the" | "my"
N -> "man" | "dog" | "cat" | "telescope" | "park"
P -> "in" | "on" | "by" | "with"
'''
grammar=nltk.CFG.fromstring(grammar_string)
```

Джерело

Яким чином визначити частину мови слів тексту?

Для визначення частини мови слова тексту за допомогою бібліотеки опрацювання природної мови NLTK для мови програмування Python можна використати вбудований метод бібліотеки `nltk` під назвою `pos_tag()`. Цей метод приймає токенізований текст як параметр. Приклад використання методу нижче:

```
import nltk
text='This is an example text of usage of a tokenizer function and then
processing those words into parts of speech using a built-in method
from a language processing library called nltk for a Python programming
language.'
words=nltk.tokenize.word_tokenize(text)
parts_of_speech=nltk.pos_tag(words)
print(parts_of_speech)
```

Джерело