LLM 활용 인공지능 서비스 개발자 양성과정

도봉 SeSAC 캠퍼스 X Saltlux

강사 최동혁

파이썬: 정규표현식

정규 표현식 (Regular Expression, ‘regex’)

- 문자열(string)을 처리하기 위한 표현식
- 프로그래밍에서 규칙을 가진 텍스트 문자열을 처리하기 위함
- LLM을 위한 데이터들은 기본적으로 텍스트 데이터가 가장 많기 때문에 전처리(Preprocess) 과정에서 매우 유용하게 쓰임
- 예시 : 괄호에 들어 있는 문자만 삭제, 주민등록번호 뒷자리 마스킹, 전화번호만 추출 등
- 점프 투 파이썬 참고

정규표현식 기초 - 메타문자(1)

- **메타문자 : 특별한 용도로 사용되는 문자**
  - . ^ $ * + ? { } [ ] \ | ( )
- **[] brackets**
  - 의미: 문자집합. [ ] 사이의 문자들과 매치. 개별적으로 나열 할 수 있음(예: [abc]는 a, b, c 모두와 매치 됨).
  - 예: [a-zA-Z] : 모든 알파벳, [가-힣] : 모든 한글
  - 참고1: - 기호는 범위를 뜻함. 참고2: [] 안에 ^가 쓰이면, 그 문자가 아닌 것이라는 의미임.
- **. dot**
  - 의미 : \n 을 제외한 모든 문자
  - 예: “충.왕” - 충렬왕, 충선왕, 충숙왕, 충혜왕, 충목왕, 충정왕, ...
- *** asterisk**
  - 의미: 앞에 문자가 0부터 무한대로 반복될 수 있음.
  - 예: “ca*t” - ct, cat, caat, 모두 가능

정규표현식 기초 - 메타문자(2)

- **+ plus**
  - 의미: 최소 1번 이상 반복
  - 예: ca+t - cat, caat 매치 됨. ct 매치 안 됨.
- **{m, n} braces**
  - 의미 : 최소 m번, 최대 n번까지 반복(생략 가능)
  - 예: ca{2,5}t - caat, caaat, caaaaat 모두 매치 됨.
- **? Question mark**
  - 의미 : {0, 1}과 동일함. 있어도 되고 없어도 되고.
  - 예: ca?t - ct, cat 매치 됨. caat 매치 안 됨.

정규표현식 기초 - 메타문자(3)

- **| bar**
  - 의미: or 와 동일한 의미
  - 예: cat|dog - cat, dog 모두 매치 됨.
- **^ circumflex, carat**
  - 의미: 문자열의 맨 처음과 일치함.
  - 예: ^Life - ‘Life is too short’은 매치 됨. ‘My Life’는 매치 안 됨.
- **$ dollar sign**
  - 의미: 문자열의 맨 끝과 일치함.
  - 예: short$ - ‘Life is too short’은 short과 매치. ‘Life is short and art is long’은 매치 안 됨.

정규표현식 기초 - 특수기호

- **\d**
  - 의미: 모든 십진 숫자와 일치. [0-9]와 동일함.
- **\D**
  - 의미: 숫자가 아닌 모든 문자 [^0-9]와 동일함. *참고 [^문자들]와 ^문자들 은 다름.
- **\s**
  - 의미: 모든 공백 문자와 동일. [\t\n\r\f\v] 와 동일함.
- **\S**
  - 의미: 모든 공백이 아닌 문자와 동일함.
- **\w**
  - 의미: 모든 문자와 동일함.
- **\W**
  - 의미: 모든 문자가 아닌 것들.
- **\b**
  - 의미: 단어의 경계
- **\B**
  - 의미: 단어의 경계가 아닌 것

정규표현식 - ? 의 활용

•긍정형 전방탐색 (?=...)

- •...에 해당하는 정규표현식과 매치되어야 함. ...에 해당하는 것은 결과로 돌리지 않음.
- •예시1: .+(?=:) - “http:// google.com". http만 매치 됨.
- •예시2: .*[.].*$ - 확장자명이 있는 파일명 매치 되는 것으로 응용 가능.

Reg. Expression:

```
.+(?=원)
```

Text:

```
1000원
2000원
3000원
5000원
10000원
```

정규표현식 - ? 의 활용

•긍정형 후방탐색 (?<=...)

- 전방탐색과 동일하나 매치되는 것의 뒤를 찾음.

Reg. Expression:

```
(?<=\$)[0-9.]+
```

Text:

```
1: $600.4
2: $10.25
3: $47.33
4: $112.34
```

정규표현식 - ? 의 활용

- **부정형 전후방탐색 (?!...)**
  - ...이 아닌 것만 매치 된다는 뜻

```
'\b(?<!\$)\d+\b'
```

I paid $30 for 100 apples, 50 oranges, and 60 pea
rs. I saved $5 on this order.

정규표현식 - ? 의 활용

- **부정형 전방탐색 (?!...)**
  - ...이 아닌 것만 매치 된다는 뜻
- **탐욕 방지 문자 '?'**
  - 반복 뒤에 ?를 사용하면 횟수가 한 번으로 제한됨.
  - 예시: \(.*?\) - 한 번만 패턴이 일치하면 매치 됨.

정규표현식 기초 - 파이썬 re 모듈

- **re.compile**
  - 정규표현식을 컴파일하여 패턴 객체를 반환한다.
- **re.match**
  - 문자열의 처음부터 정규표현식과 매치되는지 조사

```python
import re

# 정규표현식을 컴파일하여 패턴 객체를 반환
pattern = re.compile(r'\b\w+\b')

# 문자열의 처음부터 정규표현식과 매치되는지 조사
match_result = pattern.match("Hello world")
print("Match:", match_result.group() if match_result else "No match")
```

정규표현식 기초 - 파이썬 re 모듈

- **re.search**
  - 문자열 전체를 검색하여 정규표현식과 매치되는지 조사

```
# 문자열 전체를 검색하여 정규표현식과 매치되는지 조사
search_result = pattern.search("Hello world")
print("Search:", search_result.group() if search_result else "No match")
```

정규표현식 기초 - 파이썬 re 모듈

- **re.findall**
  - 정규표현식과 매치되는 모든 문자열을 리스트로 돌려줌.
- **re.finditer**
  - 정규표현식과 매치되는 모든 문자열을 반복 가능한 객체로 돌려줌.

```python
# 정규표현식과 매치되는 모든 문자열을 리스트로 돌려줌
findall_result = pattern.findall("Hello world")
print("Find all:", findall_result)

# 정규표현식과 매치되는 모든 문자열을 반복 가능한 객체로 돌려줌
finditer_result = pattern.finditer("Hello world")
print("Find iter:")
for match in finditer_result:
    print(f" - {match.group()}")
```

정규표현식 기초 - 파이썬 re 모듈

- **re.sub**
  - 정규표현식과 매치되는 문자열을 다른 문자열로 바꿔줌.

```
# 정규표현식과 매치되는 문자열을 다른 문자열로 바꿔줌
sub_result = pattern.sub("REPLACED", "Hello world")
print("Substitution:", sub_result)
```