2. Regex

March 29, 2022

[1]:
```
import re            # re = regular expression
# re.sub(r"old", r"new",str) will replace/substitute oldpattern with newpattern␣
 ↪(all instances) in string str
# re.sub(r'old', r'new',str) can also be used if the old/new pattern contains "␣
 ↪inside them
```

[2]:
```
t = "i'm fine "
t = re.sub(r"i'm",r'i am ',t)  # sub = substitute
t
```

[2]:
```
'i am  fine '
```

[3]:
```
t = "i'm fi'ne"
t=re.sub(r"'",r' ',t)
t
```

[3]:
```
'i m fi ne'
```

[4]:
```
#(?P<name>substring) : substring is assigned a symbolic name
# \g<name> : global
# \w : word, \d : digits/number
# \b : beginning; finds/matches the pattern at the beginning or end of each␣
 ↪word.
# \< : in beginning only
# \> : in end only
```

[5]:
```
# (?P<f>\w) : Each word is treated as string and assigned symbolic name f
# r'\g : replace globally

t=re.sub(r'(?P<f>\w),'   ,   r'\g<f> , '  ,    t)       # a, b --> a , b
t=re.sub(r',(?P<f>\w)'   ,   r', \g<f>'   ,    t)       # a ,b --> a , b
t=re.sub(r'(?P<f>\w)\?'  ,   r'\g<f> ?'   ,    t)       # f? --> f ?
t=re.sub(r'\?(?P<f>\w)'  ,   r'? \g<f> '  ,    t)       # ?f --> ? f
t=re.sub(r'(?P<f>\w) \. com' , r'\g<f>.com ', t)       # h . com->h.com
t=re.sub(r'(?P<f>\d) , (?P<s>\d)',  r'\g<f>,\g<s>', t) # 45 , 25-->45,25
```

[6]:
```
# . refers to any character
# a*  means 0 or more continuous occurrence of a
# a+  means 1 or more continuous occurrence of a
# a?  means 0 or 1 occurrence of a
# a{2,5} means 2 to 5 continuous occurrence of a
```

[7]:
```
# [] specifies a set of characters you wish to match
# e.g. [a-zA-Z] refers to string of lower and uppercase alphabets
# [^abc] refers to any set of characters except a, b and c
# \1 means first paranethetic expression
# \ is for escaping i.e. use a regex operator symbol (like .) as a normal␣
 ↪character

t = re.sub(r'[\.]{1,}'    ,    r'.'    , t)     # ....  -->   .
t = re.sub(r'[\?]{1,}'    ,    r'?'    , t)     # ????  -->   ?
# Replace one or more characters inside square brackets with blank
t = re.sub(r'\[.{1,}\]'    ,    r' '    , t)

t=re.sub(r"[-\"\@\\#=><\+%‘\^/&’*_~\»;!]",' ',t) # remove all symbols
# The symbols which need escaping are  "  \  ^  .  @  +

 # remove any word (of length 1-30) enclosed inside parnthesis
t=re.sub('\(\w{1,30}\)',' ',t)
 # remove opening prenthesis, closing parenthesis, and vertical bar
t=re.sub('[\(|\)]',' ',t)
```

[8]:
```
# \1 means first parenthetic expression;
# this notation is used in second part of regex and refers to first part of␣
 ↪regex

t=re.sub(r'(\w):',r'\1 :',t) #c:-->c :
t=re.sub(r':(\w)',r': \1 :',t) #:c-->: c
# The first parenthetic expression above is \w i.e. a word
t=re.sub(r"(\w)'s",r"\1 's",t)#franci's-->franci 's
t=re.sub(r"'\s",r" ",t)#francis'-->francis

t=re.sub(r'\.(\b)',r'. \1',t) #.he-->. he
t=re.sub(r'(\b)\.',r'\1 .',t) #he.-->he .
# The first parenthetic expression above is \b
# i.e. any word beginning or ending with the given symbol

t=re.sub(r'\b([a-z]{1,2})\.',r'\1 ',t)#l. st. --> l st
# The first parenthetic expression above is any 1-2 letter word
# beginning or ending with the given symbol
```