**Python Text Analysis: Preprocessing**

**Remove Extra Whitespace Characters**

Sometimes we might come across texts with extraneous whitespace, such as spaces, tabs, and newline characters, which is particularly common when the text is scrapped from web pages. Before we dive into the details, let's briefly introduce Regular Expressions (regex) and the re package.

Regular expressions are a powerful way of searching for specific string patterns in large corpora. They have an infamously steep learning curve, but they can be very efficient when we get a handle on them. Many NLP packages heavily rely on regex under the hood. Regex testers, such as [regex101](https://regex101.com/), are useful tools in both understanding and creating regex expressions.

Our goal in this workshop is not to provide a deep (or even shallow) dive into regex; instead, we want to expose you to them so that you are better prepared to do deep dives in the future!

The following example is a poem by William Wordsworth. Like many poems, the text may contain extra line breaks (i.e., newline characters, \n) that we want to remove.

*# File path to the poem*

text\_path = '../data/poem\_wordsworth.txt'

*# Read the poem in*

with open(text\_path, 'r') as file:

text = file.read()

file.close()

As you can see, the poem is formatted as a continuous string of text with line breaks placed at the end of each line, making it difficult to read.

*text*

"I wandered lonely as a cloud\n\n\nI wandered lonely as a cloud\nThat floats on high o'er vales and hills,\nWhen all at once I saw a crowd,\nA host, of golden daffodils;\nBeside the lake, beneath the trees,\nFluttering and dancing in the breeze.\n\nContinuous as the stars that shine\nAnd twinkle on the milky way,\nThey stretched in never-ending line\nAlong the margin of a bay:\nTen thousand saw I at a glance,\nTossing their heads in sprightly dance.\n\nThe waves beside them danced; but they\nOut-did the sparkling waves in glee:\nA poet could not but be gay,\nIn such a jocund company:\nI gazed—and gazed—but little thought\nWhat wealth the show to me had brought:\n\nFor oft, when on my couch I lie\nIn vacant or in pensive mood,\nThey flash upon that inward eye\nWhich is the bliss of solitude;\nAnd then my heart with pleasure fills,\nAnd dances with the daffodils."

One handy function we can use to display the poem properly is .splitlines(). As the name suggests, it splits a long text sequence into a list of lines whenever there is a newline character.

*# Split the single string into a list of lines*

text.splitlines()

['I wandered lonely as a cloud',

'',

'',

'I wandered lonely as a cloud',

"That floats on high o'er vales and hills,",

'When all at once I saw a crowd,',

'A host, of golden daffodils;',

'Beside the lake, beneath the trees,',

'Fluttering and dancing in the breeze.',

'',

'Continuous as the stars that shine',

'And twinkle on the milky way,',

'They stretched in never-ending line',

'Along the margin of a bay:',

'Ten thousand saw I at a glance,',

'Tossing their heads in sprightly dance.',

'',

'The waves beside them danced; but they',

'Out-did the sparkling waves in glee:',

'A poet could not but be gay,',

'In such a jocund company:',

'I gazed—and gazed—but little thought',

'What wealth the show to me had brought:',

'',

'For oft, when on my couch I lie',

'In vacant or in pensive mood,',

'They flash upon that inward eye',

'Which is the bliss of solitude;',

'And then my heart with pleasure fills,',

'And dances with the daffodils.']

Let's return to our tweet data for an example.

*# Print the second example*

*second\_example = tweets['text'][5]*

*second\_example*

*@VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing.\nit's really the only bad thing about flying VA"*

In this case, we don't really want to split the tweet into a list of strings. We still expect a single string of text but would like to remove the line break completely from the string.

The string method .strip() effectively does the job of stripping away spaces at both ends of the text. However, it won't work in our example as the newline character is in the middle of the string.

# Strip only removed blankspace at both ends

second\_example.strip()

*"@VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing.\nit's really the only bad thing about flying VA"*

This is where regex could be really helpful.

*import re*

Now, with regex, we are essentially calling it to match a pattern that we have identified in the text data, and we want to do some operations to the matched part—extract it, replace it with something else, or remove it completely. Therefore, the way regex works could be unpacked into the following steps:

* Identify and write the pattern in regex (r'PATTERN')
* Write the replacement for the pattern ('REPLACEMENT')
* Call the specific regex function (e.g., re.sub())

In our example, the pattern we are looking for is \s, which is the regex short name for any whitespace character (\n and \t included). We also add a quantifier + to the end: \s+. It means we'd like to capture one or more occurences of the whitespace character.

*# Write a pattern in regex*

*blankspace\_pattern = r'\s+'*

The replacement for one or more whitespace characters is exactly one single whitespace, which is the canonical word boundary in English. Any additional whitespace will be reduced to a single whitespace.

*# Write a replacement for the pattern identfied*

*blankspace\_repl = ' '*

Lastly, let's put everything together using the function [re.sub()](https://docs.python.org/3.11/library/re.html" \l "re.sub), which means we want to substitute a pattern with a replacement. The function takes in three arguments—the pattern, the replacement, and the string to which we want to apply the function.

*# Replace whitespace(s) with ' '*

*clean\_text = re.sub(pattern = blankspace\_pattern,*

*repl = blankspace\_repl,*

*string = second\_example)*

*print(clean\_text)*

*@VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing. it's really the only bad thing about flying VA*

Ta-da! The newline character is no longer there.

**Análisis de texto en Python: Preprocesamiento**

**Eliminar espacios en blanco innecesarios**

A veces nos encontramos con textos con espacios en blanco innecesarios, como espacios, tabulaciones y caracteres de nueva línea, lo cual es particularmente común cuando el texto se extrae de páginas web. Antes de profundizar en los detalles, presentemos brevemente las expresiones regulares (regex) y el paquete re.

Las expresiones regulares son una forma eficaz de buscar patrones de cadenas específicos en campos extensos. Su curva de aprendizaje es notablemente pronunciada, pero pueden ser muy eficientes una vez que las dominamos. Muchos paquetes de PLN dependen en gran medida de las expresiones regulares. Los evaluadores de expresiones regulares, como regex101, son herramientas útiles tanto para comprender como para crear expresiones regulares.

Nuestro objetivo en este taller no es ofrecer una introducción profunda (ni siquiera superficial) a las expresiones regulares; en cambio, queremos presentarles para que estén mejor preparados para profundizar en el futuro.

El siguiente ejemplo es un poema de William Wordsworth. Como muchos poemas, el texto puede contener saltos de línea adicionales (es decir, caracteres de nueva línea, \n) que queremos eliminar.

*# File path to the poem*

text\_path = '../data/poem\_wordsworth.txt'

*# Read the poem in*

with open(text\_path, 'r') as file:

text = file.read()

file.close()

Como puede ver, el poema está formateado como una cadena continua de texto con saltos de línea al final de cada línea, lo que dificulta su lectura.

*text*

*"Vagaba solo como una nube\n\n\nVagaba solo como una nube\nQue flota en lo alto sobre valles y colinas,\nCuando de repente vi una multitud,\nUna multitud de narcisos dorados;\nJunto al lago, bajo los árboles,\nRevoloteando y danzando con la brisa.\n\nContinuos como las estrellas que brillan\nY centellean en la vía láctea,\nSe extendían en una línea interminable\nA lo largo de la orilla de una bahía:\nDiez mil vi de un vistazo,\nSacudiendo sus cabezas en una danza alegre.\n\nLas olas a su lado danzaban; pero ellas\nSuperaban a las olas centelleantes en alegría:\nUn poeta no podía sino estar alegre,\nEn una compañía tan alegre:\nMiré, y miré, pero poco pensé\nEn la riqueza que me había traído el espectáculo:\n\nPorque a menudo, cuando yazgo en mi lecho\nEn vacío o en estado de ánimo pensativo,\nDestellan en ese ojo interior\nQue es la dicha de la soledad;\nY entonces mi corazón se llena de placer,\nY baila con los narcisos."*

Una función útil para mostrar el poema correctamente es .splitlines(). Como su nombre indica, divide una secuencia de texto larga en una lista de líneas cuando hay un salto de línea.

*# Split the single string into a list of lines*

text.splitlines()

['Vagaba solo como una nube',

'',

'',

'Vagaba solo como una nube',

"Que flota en lo alto sobre valles y colinas",

"Cuando de repente vi una multitud",

"Una multitud de narcisos dorados";

"Junto al lago, bajo los árboles",

"Revoloteando y danzando con la brisa".

'',

"Continuos como las estrellas que brillan",

"Y centellean en la Vía Láctea",

"Se extendían en una línea interminable",

"A lo largo de la orilla de una bahía":

"Diez mil vi de un vistazo",

"Agitando sus cabezas en una danza alegre".

'',

"Las olas a su lado danzaban; pero ellos,

'Superaron en alegría a las olas centelleantes':

'Un poeta no podía sino ser alegre',

'En tan alegre compañía':

'Miré, y miré, pero poco pensé',

'Qué riqueza me había traído el espectáculo':

',

'Porque a menudo, cuando yazgo en mi lecho',

'En estado vacío o pensativo',

'Destellan en ese ojo interior',

'Que es la dicha de la soledad';

'Y entonces mi corazón se llena de placer',

'Y baila con los narcisos'.]

Volvamos a nuestros datos de tweets para ver un ejemplo.

*# Print the second example*

*second\_example = tweets['text'][5]*

*second\_example*

*@VirginAmerica realmente pagaría $30 por vuelo por asientos que no tuvieran esta función. Es realmente lo único malo de volar en Virginia.*

En este caso, no queremos dividir el tuit en una lista de cadenas. Seguimos esperando una sola cadena de texto, pero queremos eliminar por completo el salto de línea.

El método de cadena .strip() elimina eficazmente los espacios en ambos extremos del texto. Sin embargo, no funcionará en nuestro ejemplo, ya que el carácter de nueva línea está en medio de la cadena.

# Strip only removed blankspace at both ends

second\_example.strip()

*"@VirginAmerica realmente pagaría $30 por un vuelo por asientos sin esta función. Es realmente lo único malo de volar con Virginia".*

Aquí es donde las expresiones regulares pueden ser realmente útiles.

*import re*

Ahora, con regex, básicamente la llamamos para que coincida con un patrón identificado en los datos de texto, y queremos realizar algunas operaciones con la parte coincidente: extraerla, reemplazarla o eliminarla por completo. Por lo tanto, el funcionamiento de regex se puede resumir en los siguientes pasos:

* Identificar y escribir el patrón en regex (r'PATTERN')
* Escribir el reemplazo para el patrón ('REPLACEMENT')
* Llamar a la función regex específica (p. ej., re.sub())

En nuestro ejemplo, el patrón que buscamos es \s, que es el nombre corto en regex para cualquier espacio en blanco (incluidos \n y \t). También añadimos un cuantificador + al final: \s+. Esto significa que queremos capturar una o más ocurrencias del espacio en blanco.

*# Write a pattern in regex*

*blankspace\_pattern = r'\s+'*

El reemplazo de uno o más espacios en blanco es exactamente un solo espacio, que es el límite canónico de una palabra en inglés. Cualquier espacio adicional se reducirá a un solo espacio.

*# Write a replacement for the pattern identfied*

*blankspace\_repl = ' '*

Finalmente, combinemos todo usando la función re.sub(), que significa que queremos sustituir un patrón por un reemplazo. La función acepta tres argumentos: el patrón, el reemplazo y la cadena a la que queremos aplicar la función.

*# Replace whitespace(s) with ' '*

*clean\_text = re.sub(pattern = blankspace\_pattern,*

*repl = blankspace\_repl,*

*string = second\_example)*

*print(clean\_text)*

*@VirginAmerica en serio pagaría $30 por vuelo por asientos que no tuvieran esta función. Es realmente lo único malo de volar con Virginia.*

¡Ta-da! El carácter de nueva línea ya no está.