AMPLIACIÓN PROYECTO WORD COUNT

TFC Business Intelligence & Big Data

Descripción breve

Trabajo realizado por los integrantes del curso organizado por Empleo Digital – Talentum + Ayuntamiento de Vitoria Gasteiz.

Fecha: 17/05/2017

Versión: 1.0

Índice

Resumen ejecutivo

Información básica

Nombre del proyecto	Ampliación Word count
Localidad	Vitoria
Itinerario	Business Intelligence & Big Data
Github	https://github.com/GasteizTeEscucha/TFCWordCount
Equipo	Por orden alfabético - Daniel Álvarez López - Odei Barredo Díaz - Unai Barredo Díaz - Oscar Bartolomé Pato - Helton Borges Da Silva - Virginia Esquinas Martínez - Jesús Fuerte Fernández - Esperanza García Moreno - Miriam Insagurbe Davies - Jorge Íñiguez de Ciriano Alonso - Arkaitz Merino Daubagna - Mikel Ramos Berganzo - Daniel Redondo Iglesias - Alexander Somovilla de MigueL - Blanca Soto Salvador - Mónica Vázquez Muñoz

Introducción

Como parte final del curso sobre BI & Big Data impartido por Talemtum – Empleo Digital, los alumnos deben de realizar un proyecto final de curso.

En este caso el proyecto versa sobre una ampliación del proyecto Big Delfos para Diez Puntocero.

Con el siguiente trabajo, se pretende cumplir los siguientes objetivos

Objetivos básicos

- Crear un algoritmo que cuente el número de veces que se repite una palabra en un fichero txt.
- Crear un algoritmo que limpie los datos de este fichero eliminando: artículos (solo aquellos que no son necesarios para que la palabra tenga sentido), preposiciones, conjunciones, pronombres, números, signos de puntuación.
- Crear un algoritmo que identifique las letras mayúsculas y las cambie por las minúsculas exceptuando aquellas palabras propias que se escriben con mayúscula y que tienen un significado diferente si se escribe con minúscula.

Objetivo ampliado

- El objetivo principal de este proyecto es la realización del algoritmo “o”, dicho algoritmo selecciona grupos de palabras relacionadas (Casa de Campo, Casa Real, País Vasco)

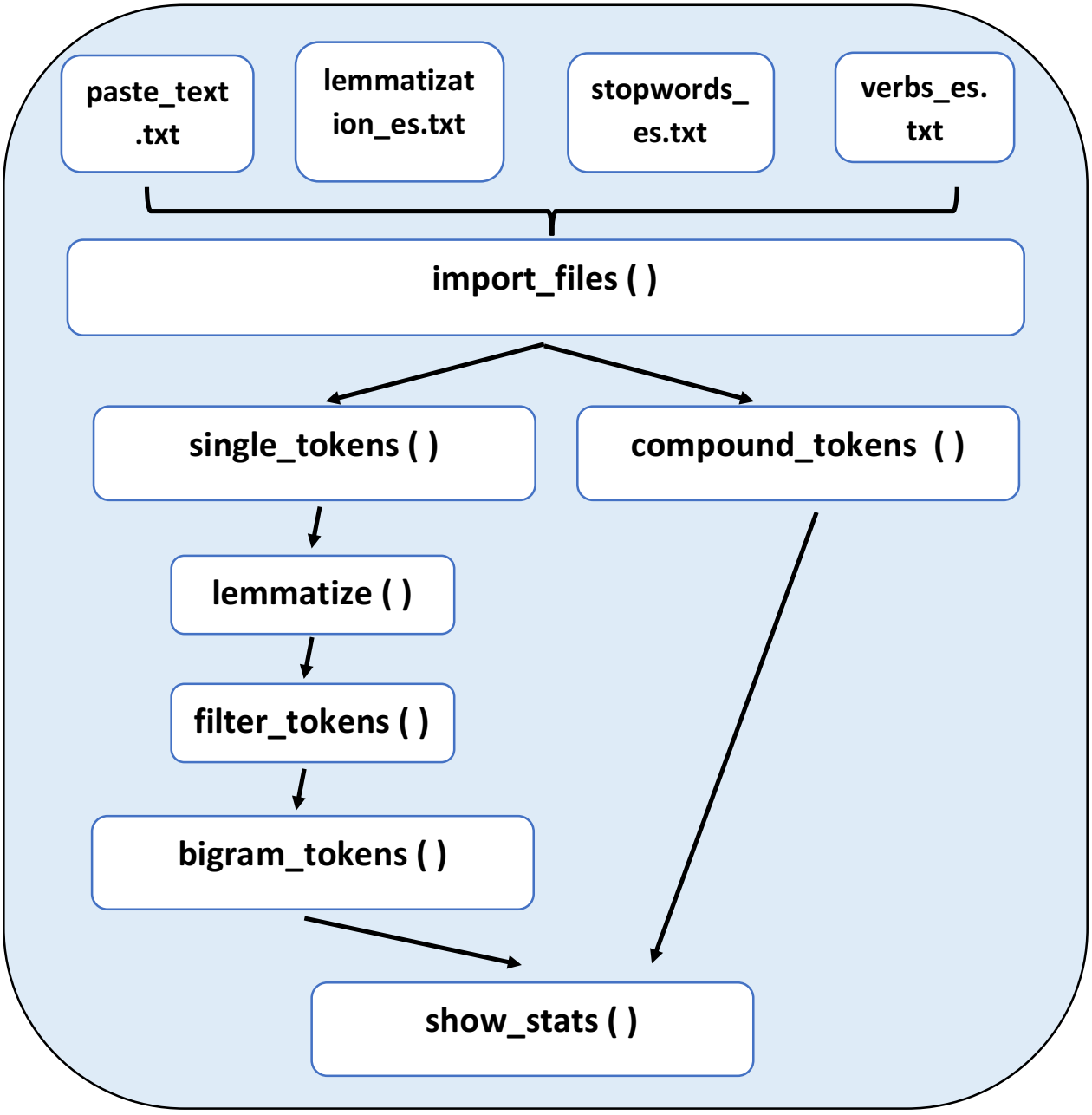
Esquema técnico

Diagrama de flujo del proyecto

Para que se entienda el esquema se procede a representar un diagrama de flujo sobre el desglose del proyecto y como son los procesos realizados.

Este proceso se inicia con la copia de las palabras a un .txt y tras su transformación y aplicación de diferentes funciones, se muestra el resultado final con las estadísticas.

Grafico 1: Diagrama de flujo



Fuente: elaboración propia

Desglose de funciones

En este apartado se indican las diferentes funciones realizadas para cumplir los diferentes objetivos, además se indican las variables que componen la función y las variables de salida.

TABLA 1: DESGLOSE DE FUNCIONES

( )	Nombre	Objetivo	Var entrada	Var salida
1º	**import_files**	Importa el texto objetivo del análisis, más los datos que ejercen de filtro para el análisis	tx path ; lm_path: sw_path ; vb_path	O_text ; lm_dict: sw_list ; vb_list
2º	**single_tokens**	Tokenizar el texto en palabras individuales	o_text	s_tokens
3º	**lemmatize**	Lemmatizar los tokens, esto consiste en dar uniformidad en la cuenta de resultados	s_tokens ; lm_dict	s_tokens (ya lematizada)
4º	**filter_tokens**	Se filtra las palabras de los tokens, eliminando la información definida como no relevante	s_tokens ; vb_list ; sw_list	s_tokens (ya lematizada y filtrada)
A partir de aquí, se desglosan las funciones propias para el objetivo ampliado				
5º	**bigram_tokens**	Obtener una lista de tokens emparejados en función de su proximidad	s_tokens	b_tokens
6º	**compound_tokens**	Definir una lista de tokens por nombres compuestos	o_text	c_tokens
7º	**show_stats**	Función que muestra por pantalla, la cantidad de los tokens Totales y únicos y un listado con los tokens más comunes	tokens	No tiene puesto que imprime en pantalla los resultados
8º	**main**	Es la función principal que define el flujo del algoritmo	tx_path ; lm_path; sw_path; vb_path	No tiene salida, es una llamada que los resultados se muestran en la f(show_stats)

Fuente: elaboración propia

Para la definición de estas funciones se establecieron diferentes grupos de trabajo, aunque la toma de decisión y transcripción a Python fue realizada únicamente por el responsable de esta tarea.

Descripción de variables

Se detalla el significado de cada una de las variables utilizadas.

- **tx_path:** fichero que contiene el texto a analizar.
- **lm_path:** fichero que contiene el diccionario lematizador
- **sw_path:** fichero que contiene las palabras sin significado
- **vb_path:** fichero con un listado de verbos
- **o_text:** es el texto original en formato string
- **lm_dict:** es el lematizador en formato de diccionario
- **sw_list:** es la lista de palabras sin significado
- **vb_list:** es la lista de verbos.
- **s_tokens**: lista de tokens individuales
- **b_tokens:** lista de tokens en bigramas
- **c_tokens:** lista de tokens por nombres compuestos

Tecnologías empleadas

Para la realización de este proyecto, se decide utilizar Python.

¿Qué es python?

Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y programación funcional.

Ejemplo práctico

No hay nada mejor que ver cómo funcionan las cosas, por ello, se presenta el siguiente ejemplo.

Texto a analizar

"Emmanuel Macron toma este domingo posesión como presidente de la **República de Francia**. El líder del movimiento **En Marche**! reemplaza en el cargo al socialista **François Hollande**, y a sus 39 años, se convierte en el jefe de Estado galo más joven desde Napoleón."

Resultados

```
<<Single-Tokens>>
Unique: 22
Total: 22
Ratio: 1.0
Most Common (10):
[('Emmanuel', 1), ('Macron', 1), ('domingo', 1), ('posesión',
1), ('presidente', 1), ('República', 1), ('Francia', 1),
('líder', 1), ('movimiento', 1), ('Marche', 1)]

<<Bigram-Tokens>>
Unique: 21
Total: 21
Ratio: 1.0
Most Common (10):
[(('Emmanuel', 'Macron'), 1), (('Macron', 'domingo'), 1),
(('domingo', 'posesión'), 1), (('posesión', 'presidente'), 1),
(('presidente', 'República'), 1), (('República', 'Francia'), 1),
(('Francia', 'líder'), 1), (('líder', 'movimiento'), 1),
(('movimiento', 'Marche'), 1), (('Marche', 'reemplazar'), 1)]

<<Compound-Tokens>>
Unique: 4
Total: 4
Ratio: 1.0
Most Common (10):
[('Emmanuel Macron', 1), ('República de Francia', 1), ('En
Marche', 1), ('François Hollande', 1)]
```

Análisis de los resultados

Obtenemos tres resultados

1. Tokenización por palabras individuales
2. Tokenización mediante bigramas
3. Tokenización de palabras compuestas

A partir de los Bigramas se puede crear una matriz de relación de palabras. Con la utilización de un algoritmo de clusterización, se podrían crear grupos de palabras para que, a la hora de realizar una búsqueda, se dispusiera de una lista de palabras relacionadas entre sí aunque no sea de forma directa.

Anexos

Disponibles en githup