Taller 5

El objetivo de este taller es procesar un documento para obtener una lista de palabras diferentes y un recuento individual de las mismas

Para ello use los siguientes operadores

Read Documents: Este operador permitirá leer un documento el cual está grabado en el disco duro de la máquina. Se propone para este ejercicio utilizar el archivo el cual puede ser descargado desde la página virtual

Process Documents: Este operador permitirá procesar el documento para obtener una lista de sus diferentes palabras y un recuento individual. Este operador agrupará a los operadores que ejecutarán los subprocesos para dividir el documento en palabras. Los operadores a utilizar forman parte del componente Text Processing:

- Tokenization: permite crear una bolsa de palabras que está contenido en el documento. Esta operación permite filtrar aún más el documento.

- Filtering: Esta carpeta contiene diversos métodos de filtrado que se pueden aplicar al proceso. Para esta práctica la idea es filtrar aquellas palabras del documento que realmente no aportan ningún significado (como las palabras a, and, the, as, of, etc.). Este conjunto de palabras es conocido como stop words; por lo tanto seleccione el operador "Stopwords (inglés)" dado que el documento está en este idioma.

  De manera adicional se requiere filtrar las palabras restantes que tienen menos de tres caracteres. Seleccione "Filter Tokens by Length" y establezca los parámetros (en este caso, el número mínimo de caracteres es 3; mientras que el número máximo de caracteres puede ser un número arbitrariamente grande, ya que no me importa este valor en realidad. Conecte los nodos de cada operador.

- Transformation: Una vez filtrado las palabras vacías y aquellas de longitud menos a tres, es necesario transformar todas las palabras a minúsculas, ya que la misma palabra se contabilizaría de manera diferente si estuviera en mayúsculas y minúsculas. Para ello seleccione el operador "Transform Cases " y configúrelo.

  Otro operador dentro de la carpeta Transformation es “Generate n-grams (Terms)”. Un n-Gram es una combinación de 'n' términos consecutivos en una oración. En la oración "el zorro marrón rápido salta sobre el perro perezoso", "marrón rápido" se considera de 2 grams. Mientras que "zorro marrón rápido" es un 3 grams, y así sucesivamente. Al generar n-Grams, Rapidminer no filtra palabras sueltas, ya que una palabra es de 1 gram y, por lo tanto, sigue siendo n-Gram.

  Para esta práctica se pide agregar otro operador que permita filtrar todo lo que esté por debajo de 2 grams y, por lo tanto, mostrar datos más relevantes. Este operador es el operador "Filter Tokens (by content)". Como Rapidminer separó n-Grams con un guion bajo (_), configuramos el nuevo operador para almacenar cualquier cosa que contenga un guion bajo, que debería ser cualquier n-Gram con una n de dos (2) en este caso. Colocar esta información (_) en el campo string de la sección de parámetros y en el campo condition se puede seleccionar la opción contains.

Antes de ejecutar la práctica conecte los nodos de cada operador para ver los resultados.

Ejercicio Propuesto

- Agregue a este ejercicio la posibilidad de leer varios archivos y aplicar el proceso especificado en esta práctica
- Agregue la posibilidad de encontrar las raíces de las palabras usando el operador Stem (Porter)
- Incorpore la posibilidad de generar y ver solo 3-grams en el documento. Para ello es necesario ejecutar tres acciones

    -Cambiar la longitud máxima para que permita leer 3-grams

    -Cambiar la condición a matches

    - Usar una expresión regular que permita leer cualquier cosa que tenga dos guiones bajos. Ejemplo abandoned_idea_formed