GUÍA DE LA PRACTICA 3. Minería de Texto con RapidMiner

RapidMiner es una herramienta que permite realizar diferentes tareas de procesado y minería de datos de manera intuitiva, sin escribir código, por medio de **operadores** que realizan funciones concretas y conexiones entre esos **operadores**.

El objetivo de esta práctica es realizar varias tareas básicas de minería de texto, como desarrollar un modelo predictivo o realizar tareas de modelado de topics.

1.Instalar RapidMiner

RapidMiner puede descargarse de: https://rapidminer.com

Comenzar por visualizar los tutoriales que muestra Rapid Miner en una ventana para entender el funcionamiento básico de la herramienta.

Extensiones

La funcionalidad básica de RapidMiner puede ampliarse fácilmente con el uso de extensiones, que son módulos externos que realizan tareas concretas. Estas extensiones se descargan a través de menú Extensions > Marketplace (updates and extensions). Las extensiones que necesitarás inicialmente para esta práctica son Text Processing (para el pre-procesado de texto), Operator Toolbox (para LDA y LSA), y word2vec. Text Processing y Operator Toolbox suelen estar incluidas en la instalación básica, puedes comprobar si ya están instaladas eligiendo Extensions > Manage Extensions.

RapidMiner establece tres áreas de trabajo, *Design*, *Results* and *Auto Model*. El proceso típico de trabajo es diseñar nuestro pipeline de minería de datos en *Design* enlazando operadores y visualizar los resultados en *Results*. Los tres primeros tutoriales que se abren por defecto al descargar RapidMiner explican claramente esta dinámica.

Parte 1. Clasificación de texto supervisado

1. Cargar el texto

El primer paso es cargar los datos del soporte en el que estén disponibles y prepararlos para trabajar con ellos. Los soportes más típicos son **ficheros de texto independientes** o **texto como campos en un fichero excel**. En función del formato que tengamos usaremos el operador de RapidMiner correspondiente (por ejemplo Read Excel o Process Documents from Files).

Una vez cargado el texto, RapidMiner puede manejarlo también de diferentes formas. Las principales son colecciones de documentos (documentos como objetos independientes dentro de un objeto "**document collection**"), y conjuntos de ejemplos (objeto "**exampleSet**", en formato tabla, que es el formato directo para texto leído de un fichero Excel. Como veremos más adelante algunos operadores están disponibles en dos versiones, cada una preparada para trabajar con uno de estos tipos de datos. También existe un operador que convierte un "exampleSet" en "document collection"y viceversa.

Leer texto de Excel (no aplica en esta práctica – todo el texto está en documentos ver más adelante cómo se hace con Process Documents from Files)

Usaremos el operador “Process Documents from Files” para leer los documentos de ficheros. Añada a tu proceso el operador: el camino más rápido para encontrar cualquier operador es teclear su nombre en la caja de texto search for operators que mostrará todos los operadores que coincidan con el texto buscado. Una vez añadido el operador, lo configuramos (se puede hacer usando el wizard “import configuration wizard”), estableciendo el directorio del que queremos los datos.

Generar la matriz de términos por documento

El siguiente paso es procesar el texto, siguiendo los pasos típicos del proceso de minería de texto, y crear la matriz de términos por documento en el formato que elijamos.

Usaremos el operador “Process documents from file”, que espera como entrada los documentos y da como salida el texto procesado

Una vez añadido el operador “*Process documents from File*”, lo desplegamos (doble click) para insertar los operadores correspondientes (es decir, se insertan subprocesos dentro del proceso. Estos son los pasos más comunes de pre-procesado del texto que puedes añadir (Figura 2)

1. Identificación de los términos individuales (tokens) en el texto. Para ello, se utiliza el operador *Tokenize*.
2. Filtro de palabras que no son de interés (*Filter Stopwords*). Utilizaremos el filtro por defecto para palabras en inglés
3. Poner todo en minúsculas: Operador *Transform cases*
4. Eliminar palabras de dos o menos letras: *Filter Tokens (By Length)* – el número de letras es configurable
5. Reducir las palabras a su raíz. Para ello añadimos un operador que ejecute un algoritmo de *stemming*, por ejemplo el operador *Stem* con el algoritmo Snowball configurado para inglés.

En función de la tarea a realizar, algunos de estos pasos son adecuados o no. Por ejemplo, *stem* puede no ser necesario y empeorar el resultado. Hay otras opciones de pre-procesado que pueden ser útiles, como por ejemplo eliminar palabras de un diccionario establecido por nosotros. Para ello habría que crear el diccionario y usar el operador Filter Stopwords (Dictionary).

Cuando terminemos esta parte puedes probar a añadir o quitar algunos pasos del procesado de los textos y ver el efecto.

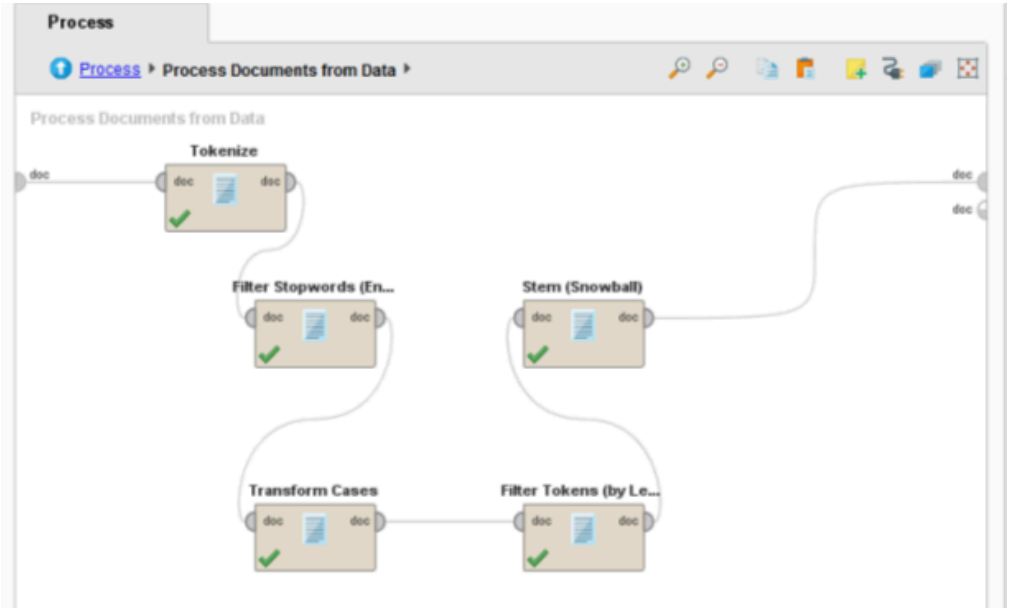


Figura 2. Sub-proceso englobado en el operador Process Documents from Data

Volvamos a la pantalla principal de diseño (pulsando sobre el nombre del proceso o sobre la flecha que está junto al nombre) para configurar el operador "*Process Documents from Data*" o "*Process Documents from File*" y ver el resultado del proceso. Como se puede ver en la Figura 1 el operador *Process Documents from Data* tiene dos puertos de salida exa y wor. En el primero se genera la tabla de términos por documento (tipo de datos Example Set) y en el segundo, una lista de las palabras del corpus con su frecuencia de aparición (tipo de datos Word List). Conectaremos ambas salidas a los puertos de resultados.

Antes de ejecutar el proceso, estableceremos la configuración del operador "*Process Documents from File*". El principal parámetro a configurar es la forma de crear la matriz de términos por documento, según como queramos que se represente la frecuencia de cada término: cuenta (term occurrence), frecuencia, frecuencia binaria (presencia vs ausencia) y tf-idf (frecuencia de término – frecuencia inversa del documento, term frecuency –inverse document frecuency). Estas opciones se establecen en el desplegable vector creation.

De momento vamos a explorar la opción básica (cuenta, term occurrence).

Además, el operador "*Process Documents from File*" permite establecer un método de poda (*prune*) que limite los términos que se utilizan para generar la matriz de términos por documento, eliminando los muy frecuentes y poco frecuentes. En función de la tarea a realizar, este paso será útil o no. Por ejemplo se puede establecer una poda ligada al porcentaje (*percentual*), haciendo que la cota inferior (*prune_below_percent*) sea de 10%, dejando la cota superior (*prune_above_percent*) al 100%.

Ejecutemos el proceso para inspeccionar los dos resultados: matriz de términos por documento (de tipo *example set*) y lista de palabras (de tipo *word list*). Veamos primero la *word lis*t. Si ordenamos las palabras por el número de veces que aparecen, podemos ver los términos más comunes, y cómo de comunes son para cada una de las clases por separado. Si inspeccionamos el *example set*, veremos la matriz en la que para cada entrada y término tenemos el número de

veces que aparece. Podemos comprobar que es una matriz dispersa (la mayoría de los términos son 0).

Ahora, repite la ejecución del proceso con otras representaciones de los documentos como term frequency y TF-IDF, y quizá con otros valores y métodos de pruning.

Durante las pruebas, puede ser práctico desconectar algún operador, por ejemplo para conectar los resultados intermedios a la salida. Para desconectar un operador y que no genere errores en la ejecución es necesario deshabilitarlo. Para ello selecciona "*enable*" en el menú que aparece al hacer click-derecho sobre el operador.

Opcionalmente antes del operador process documents, puedes usar el operador Filter Examples, que sirve para filtrar los datos de entrada, y dejar solo los correspondientes a una etiqueta determinada y ver cómo cambian los términos identificados. Este filtro también se puede usar para trabajar con un subconjunto de todos los datos que contiene el fichero Excel, por ejemplo para hacer unas pruebas iniciales.

5. Construir y evaluar un clasificador

Ya tenemos los datos preparados para construir el clasificador. Empezaremos por un árbol de decisión, pero puedes probar otros tipos de modelos

Añade el operador decision tree y conecta el modelo generado a los resultados. Visualiza el árbol y mira si tiene sentido cómo se realiza la clasificación.

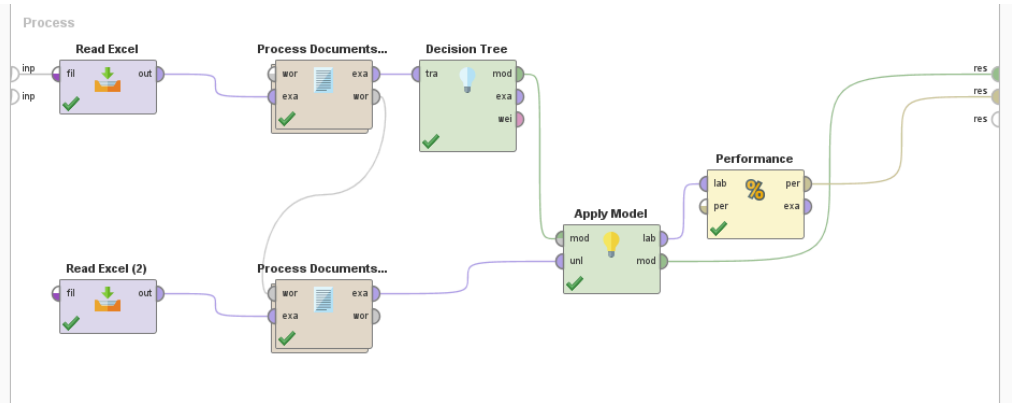
Para evaluar la calidad del modelo, tenemos dos opciones: hacer validación cruzada (cross validation) o usar un conjunto de test independiente.

Probemos con el conjunto de test independiente que se facilita para la práctica. Lo cargamos igual que el conjunto original, y generamos también la matriz tf-idf siguiendo los mismos pasos de pre-procesado, pero con una importante diferencia. Ahora el operador "process data" tiene como entrada la lista de palabras que deben figurar en la matriz, y que será la generada con el conjunto de entrenamiento. Esto significa que, si el conjunto de test tiene alguna palabra que no estaba en el de entrenamiento, esa palabra no aparecerá en la matriz tf-idf. Si no se utiliza la word list como entrada, se generará la matriz con los nuevos términos, y no será posible aplicar el modelo entrenado (las entradas serán distintas). En la figura 3 se muestra la configuración del proceso.

A continuación, incluir el operador Apply Model y unir a la entrada mod el modelo entrenado generado por Decision Tree. A la otra entrada unl (unlabeled), conectamos los datos de test.

Para evaluar el modelo, puedes añadir un operador Performance y unir la salida lab de Apply Model con la entrada lab de Performance, y la salida de este último operador con el nodo res. Puedes ver todo el proceso completo en la figura 3- Estudia la matriz de confusión ¿hemos construido un buen clasificador? Parece que no. Analiza las causas y haz los cambios necesarios para mejorarlo.

También se puede hacer validación cruzada con el operador cross validation, e iterar el experimento con diferentes parámetros.



~~Figura 3. Entrenar y evaluar el modelo~~

Representación con n-grams

Modificar el paso de procesamiento de texto para trabajar con 2-gramas (pares de dos palabras). Para ello usamos el operador generate n-grams dentro de Process Documents. Se puede eliminar el paso Stem o colocarlo a continuación.

Problema 2. Topic modeling

Ahora vamos a hacer un análisis no supervisado, aunque en el conjunto de datos original (colección BBC Full Text Document Classification[1] 2225 documentos del portal web BBC news de los años 2004 y 2005) los documentos están etiquetados en cinco clases: business, entertainment, politics, sport y tech

Utiliza la versión disponible en Aula Global. Contiene alrededor de 2200 documentos, con un volumen de 5Mb. Te recomiendo que para empezar trabajes solo con unos pocos documentos para que las pruebas vayan más rápido.

El objetivo de esta parte de la práctica es agrupar los documentos de la colección en temas e identificar las palabras más representativas de cada tema.

Puedes elegir un método como el visto al principio para representar los documentos como vectores y luego aplicar técnicas de clustering (como K medias), o alternativamente utilizar Latent Dirichlet Allocation o Latent Semantic Analysis (Singular Value Decomposition) como se han visto en clase de teoría. RapidMiner incluye operadores para todas estas opciones.

Como ahora se trabaja con documentos y no con datos de un fichero de Excel, usaremos el operador “Process Documents from Files”, en vez del operador usado en el primer apartado al para leer datos de Excel (“Process Documents from Data”). El funcionamiento es el mismo: para generar la matriz tf-idf, hay que incluir sub-operadores para pre-procesar el texto (tokenize, stopwords) y que la representación que tengamos sea la más adecuada. En este punto los datos están preparados para usar técnicas clásicas de clústering como k-medias.

[1] D. Greene and P. Cunningham. "Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering", Proc. ICML 2006
http://mlg.ucd.ie/datasets/bbc.html

Otra alternativa es usar el operador que implementa LDA (está en Operator Toolbox). Este operador no espera una matriz en la entrada, sino el documento completo, por lo que el preprocesado es ligeramente diferente. En la Figura 4, verás una configuración posible: usa "Process Documents from Files", marca "Keep text" y como subproceso, simplemente conecta directamente la entrada y la salida. Luego añade un operador "Data to Documents" que convierte el texto al documento que necesitamos. Este operador puede mezclar diferentes campos de texto en un documento, así que hay que elegir el texto como atributo y asignarle un peso uno (si hubiera varios campos de texto, cada uno tendría un peso). A continuación añade el un operador "Loop Collection", que itera las acciones que lo componen a lo largo de todos los documentos. Dentro de este operador es donde colocamos ahora los operadores de pre-procesado de texto que sean necesarios (tokenize, filter stopwords). Ya tenemos los datos preparados y conectamos su salida con el operador Extract Topics from Documents (LDA).

Si alguna palabra aparece en muchos temas o no tiene significado, puedes quitarla añadiendo un nuevo paso al filtrado de stopwords: eliminar las palabras de una lista (diccionario), el operador a utilizar es filter stopwords – dictonary)

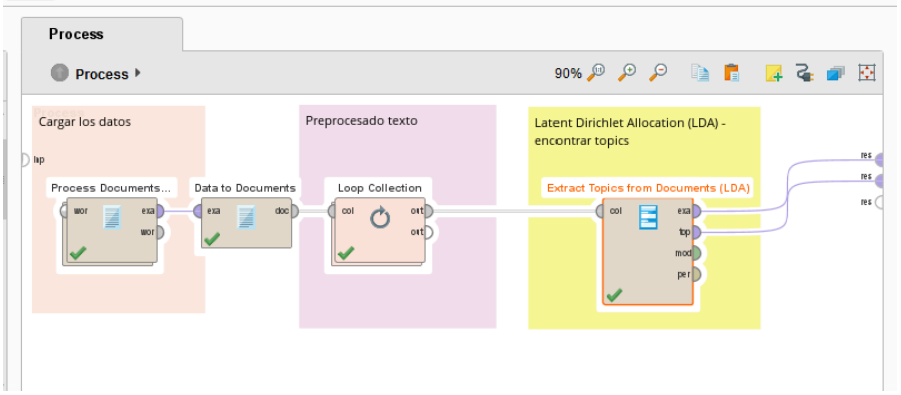


Figura 4. Extracción de topics con LDA

Problema 3. Generación de la representación con Word embeddings

El objetivo de este problema es experimentar con la representación con Word embeddings. Puedes usar como partida cualquiera de los dos datasets anteriores, la explicación que sigue se refiere a los datos cargados desde Excel, pero si usas los documentos del segundo problema, es más sencillo pues no tienes que hacer la conversión.

Si no lo hemos hecho ya, instalamos la extensión Word2vec que contiene los operadores necesarios para trabajar con word embedding. (Menu Extensions > Marketplace (updates and extensions).

El operador word2vec espera como entrada una colección de documentos tokenizada, por lo que debemos preparar los datos en ese formato: cargamos los datos de hipotecas con ReadExcel y los transformamos a una colección de documentos (Data to Documents). Para ello, colocaremos este operador tras el filter by attributes. Alternativamente se podría poner directamente tras ReadExcel y configurarlo para que tome el campo descripcion_usu como único parámetro: Show advanced parameters, Marcar select attributes and weights, asignar al atributo descripcion_usu el peso 1.

Con este pre-procesado hemos convertido las entradas en la tabla Excel a una colección de documentos (puedes comprobarlo conectando la salida a result). El siguiente paso es tokenizar estos documentos. Para aplicar el operador Tokenize sobre toda la colección la colocamos dentro del operador Loop Collection. Además de tokenizar se pueden aplicar los operadores que deseemos sobre los tokens (stem, filter by size, ..)

Comprobamos que la salida en este punto es una colección de documentos igual que antes, pero cada termino (token) está marcado con un color.

Ahora ya podemos usar el operador word2vec sobre la colección. El resultado es un modelo, que debemos guardar, ya sea como un archivo en disco con el operador Write Word2Vec o, preferiblemente, como un objeto en el repositorio con Store, para lo que debemos configurar un nombre.

Usamos el puerto thru de Store que proporciona a la salida el mismo modelo que a la entrada y lo conectamos con Extract Vocabulary que nos permite ver los términos del vocabulario y su codificación con el vector embedding

Codificar términos con el modelo word2vec creado

Si queremos algunos términos individuales (no documentos) podemos usar el operador Apply Word2Vec-Examples, que trabaja sobre términos individuales. Su entrada serán, por un lado esos términos (provenientes por ejemplo de un fichero Excel) y por otro el modelo word2vec creado. El fichero Excel lo podemos crear añadiendo los términos a mano o generarlo como una salida más de uno de los procesos anteriores (por ejemplo guardando la salida del fichero de términos por documento del primer apartado a un Excel con write Excel),

Normalmente querremos trabajar con documentos, para ello usamos el operador Apply Word2Vec-Documents. De nuevo tenemos que convertir los datos a documentos como hicimos en el punto anterior (data to documents, loop, tokenize). Cargamos el modelo word2vec desde el fichero o repositorio donde lo hubiésemos guardado (operador ReadWord2Vec o Retrieve) y usamos datos y modelo como entrada de Apply Word2Vec-Documents para generar la codificación.

Otra alternativa sería usar un modelo de Word embeddings ya entrenado. Puedes buscar uno de estos modelos siguiendo las referencias de la documentación del operador Apply Word2Vec-Documents

<u>Problema 4.</u> Extracción de entidades (voluntario)

Esta parte de la práctica es voluntaria, aunque se recomienda al menos leerla.

El Reconocimiento de entidades nombradas -NER por sus siglas en inglés Named entity extraction - también conocido como extracción de entidades, es una tarea de extracción de información que busca localizar y clasificar en categorías predefinidas, como personas, organizaciones, lugares, expresiones de tiempo y cantidades, las entidades nombradas encontradas en un texto.

Para realizar extracción de entidades podemos apoyarnos en distintas extensiones que proporcionan operadores para hacer estas tareas y que normalmente funcionan bajo licencias de

pago. Os recomiendo que uséis la extensión **Meaning Cloud** y como segunda opción Rosette. En el momento de escribir este documento Meaning Cloud ofrece una versión de prueba por tiempo ilimitado y Rosette da un periodo de prueba de 30 días.

Para instalar las extensiones buscamos en el market place (menú Extensions > Marketplace - updates and extensions).

El siguiente paso será configurar la conexión. Para que estas extensiones funcionen es necesario registrarse en su página web y solicitar una clave (api key) que se debe luego usar para configurar el operador. Para Rosette esta conexión no se crea en el repositorio como la de twitter sino en la propia configuración del operador: pinchamos sobre el icono de Rosette junto a connection y configuramos el nombre de la conexión y la api key que se nos proporcionó con el registro. El campo alternate URL debe dejarse vacío.

Hay que tener en cuenta que estas extensiones procesan nuestras peticiones como invitados, por lo que serán lentas y tienen diversas limitaciones como número de consultas al día. Os recomiendo que para probarlas uséis un número reducido de ejemplos (por ejemplo, dos o tres documentos, o si los datos están en un example set los podéis filtrar con un operador Filter Example Range). Unas 1000 filas se procesan en un par de minutos

Elige los datos que vas a usar para este problema: pueden ser los documentos del archivo de noticias o párrafos sueltos de esos documentos. También puedes usar cualquier conjunto de datos que sea de tu interés, por ejemplo información extraída de twitter (siguiente apartado).

Utiliza el operador Extract Entities para ver qué tipo de entidades se identifican.

Preguntas Problema 4

Documenta los elementos del proceso que has configurado, los resultados obtenidos, y tu interpretación de los resultados. ¿Es útil la información obtenida? ¿Se extraen correctamente las entidades?

Problema 5. Obteniendo datos de twitter (voluntario)

Ahora puedes experimentar con datos obtenidos de twitter. Solo podrás hacer esta parte de la práctica usando una cuenta de twitter, por lo que su carácter es voluntario.

El primer paso es configurar una conexión. En RapidMiner se usan las conexiones para muchas opciones diferentes de importar datos, como por ejemplo conectarse a una base de datos. Las conexiones se configuran en la pestaña del repositorio, que está en la parte superior del panel derecho (encima de la pestaña operadores).

Configurar la conexión (dentro del repositorio, panel derecho pestaña superior, sobre operadores). Crearemos la conexión en el repositorio local (Local Repository), eligiendo connections – create connection. En el desplegable connection type, elegimos twitter. Rapid miner nos redirige a twitter, donde tras identificarnos con nuestra cuenta obtendremos un código de autorización que luego debemos pegar en RapidMiner para darle permisos para acceder con

nuestra cuenta. En cualquier momento podemos abrir la connection para comprobar si está funcionando (test connection).

Una vez configurada la conexión podemos usar diferentes operadores para obtener tweets que luego procesaremos. Podemos buscar por palabras clave (SearchTwitter) u obtener tweets de un usuario concreto (Get Twitter User Status) u obtener información de un usuario (Get Twitter User Details). En todos los casos debemos configurar el operador poniendo connection source = repository y connection entry la que acabamos de configurar (hay otras opciones para configurar las conexiones que no veremos aquí). Cada conexión puede necesitar información adicional (usuario cuyos twitters queremos leer, palabra o tag a buscar).

Twitter no servirá volúmenes de datos muy grandes, se recomienda siempre poner un límite inferior a 1000 tweets.

Elige un usuario de twitter o una palabra clave que te resulte de interés y realiza un análisis con las técnicas que has aprendido durante la práctica. Documenta el proceso realizado y los resultados

Problema 6. Sentiment Analytics (voluntario)

Rosette y Meaning Cloud incluyen también una extensión para hacer minería de opiniones. Puedes usarla sobre los datos extraídos de twitter.

Documenta los elementos del proceso que has configurado y pon algunos ejemplos de frases y su resultado. ¿Cuántos grados de subjetividad proporciona la herramienta? ¿Y de polaridad? ¿Tienen sentido los resultados obtenidos?