Master Oficial en Sistemas Ingeligentes y Aplicaciones Numéricas en la Ingeniería

Universidad de las Palmas de Gran Canaria

Trabajo final para la asignatura: Minería de Datos

Clasificación de revistas utilizando técnicas de Minería de Texto

Autores: Lukas Tajak y Sergio Marrero Marrero

Tutor de la asignatura:
Javier Lorenzo Navarro

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivos

El contexto de este documento es la asignatura Minería de Datos del Máster Universitario de Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería (MUSIANI)(año académico 2015/2016). Se ha realizado como tarea final para superar la asignatura un pequeño proyecto de minería de datos siguiendo la metodología CRISP-DM aplicada a Minería de Textos (Text Mining). Se concretan los objetivos a continuación:

- Obtención de datos
- Aplicación de técnicas de preprocesado que se consideren necesarias
- Obtención de un modelo predictivo (clasificación)
- Evaluación del resultado obtenido
- Redacción de memoria descriptiva

El trabajo de minería de texto propuesto consistirá en la clasificación de artículos de investigación proveniente de dos revistas de IEEE. El conjunto de documentos sobre el que se trabajará serán todos los artículos publicados en el 2015 en las revistas indicadas utilizando :

- Título
- Resumen(abstract)
- Palabras claves del autor(Keywords)

Las dos revistas sobre las que se realizará el trabajo son:

- IEEE Transactions on Image Processing
- IEEE Transactions on Medical Imaging

Una vez planteados los objetivos damos paso al desarrollo del trabajo

Capítulo 2

Metodología CRISP-DM y desarrollo del proyecto

Para implementar una tecnología en un negocio, se requiere de una metodología. La mayoría de las consultoras especializadas en alguna tecnología cuentan, con por lo menos, una metodología, según los tipos de proyectos que aborden. Para el caso de proyectos de implementación de minería de datos, hay una metodología en particular, CRISP-DM. El estándar incluye un modelo y una guía, estructurados en seis fases, algunas de estas fases son bidireccionales, lo que significa que algunas fases permitirán revisar parcial o totalmente las fases anteriores. En la imagen 2.1 se puede observar a simple vista las distintas fases y de esta metodología.

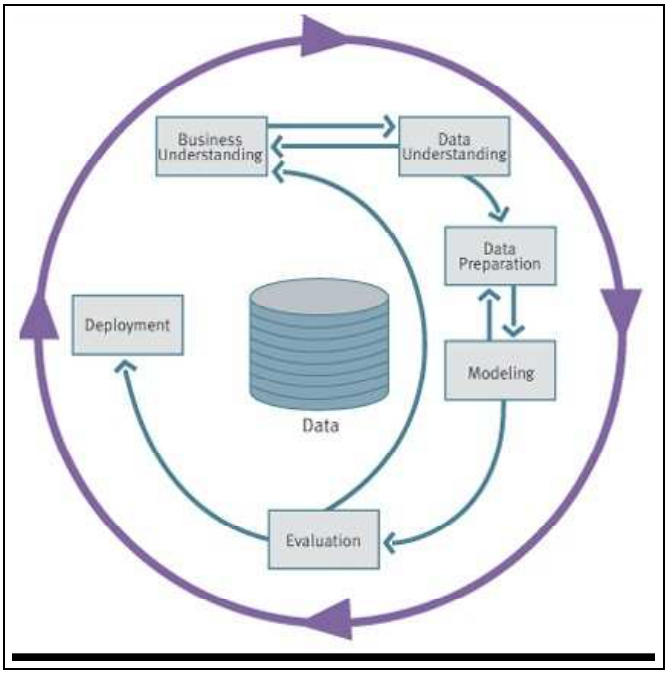


Figura 2.1: Metodología CRISP-DM

De ahora en adelante se emulará estar bajo la metodología CRISP-DM.

2.1. Business Understanding

2.1.1. Determinar los objetivos del negocio:

Se puede considerar que el apartado redactado en la Introducción y Objetivos se correspondería con esta fase de la metodología.

2.1.2. Evaluación de la situación

La situación en la que nos encontramos es la siguientes:

1. El día de comienzo del proyecto se fechó el 20/05/2016.El plazo máximo para entregar los resultados fechan el 05/06/2016. Se disponen 16 días para su elaboración.

2. Se deberá dividir el proyecto en varias fases. En la fase de obtención de los datos a analizar posiblemente haya que realizar algún programa que facilite la obtención y depuración de los datos.

2.1.3. Determinar los objetivos

Se diferencian dos tipos de objetivos:

- *Objetivos del grupo:* Conseguir destrezas en el área de la Minería de Datos. Conseguir la máxima puntuación en la asignatura.

- *Objetivos de la minería de datos:* Conseguir un clasificador que permita discernir entre las dos clases de documentos. En principio se pretende conseguir un *ratio de acierto* superior al 90 %.

2.1.4. Generar el plan del proyecto

Para realizar el proyecto se utilizará la herramienta minería de datos : *RapidMiner 7.1.* Además deberá ser descargado el paquete *Text Processing 7.1.1.* Se escogió esta herramienta de trabajo porque se consideró la más eficiente para realizar la tarea en cuestión. En la subimagen 2.2 se detalla el entorno de trabajo de *RapidMiner.*

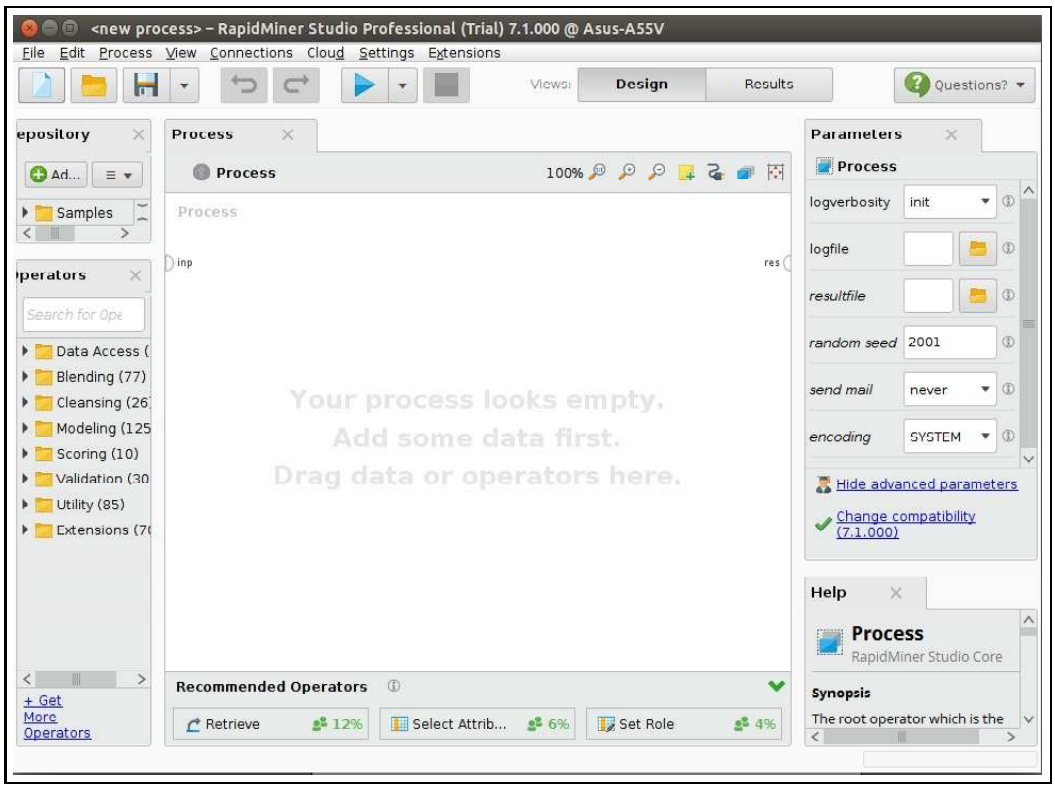
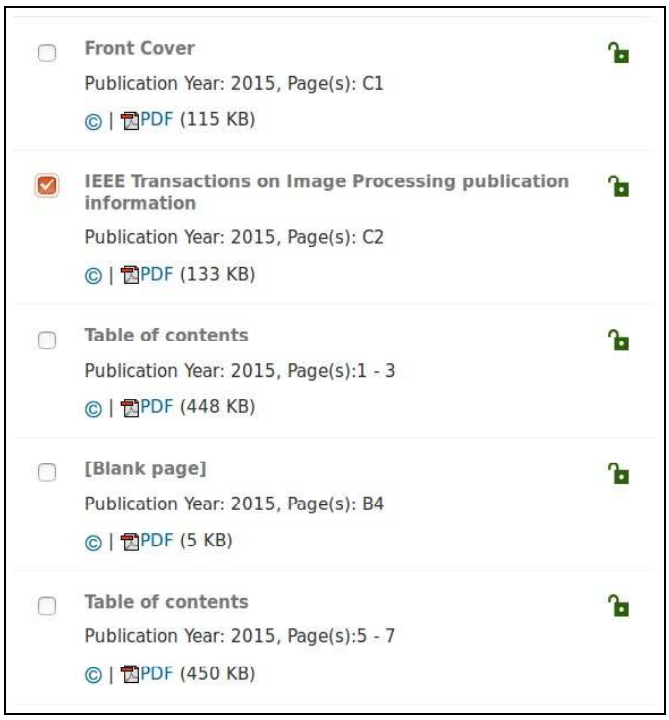


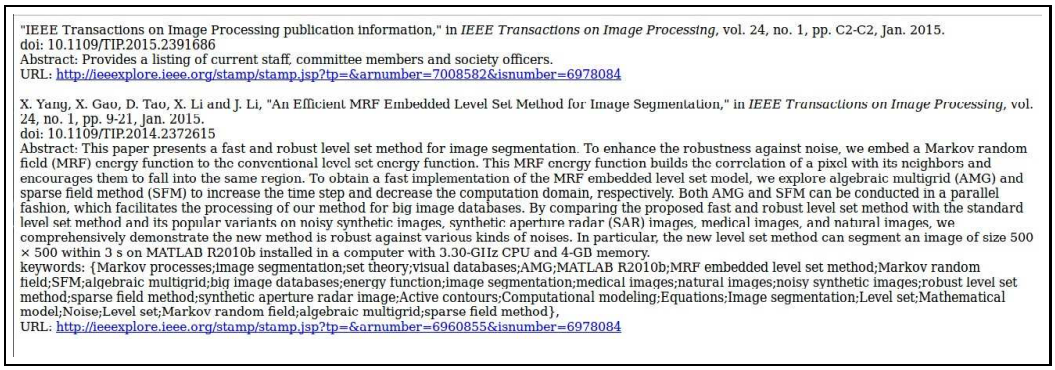
Figura 2.2: Entorno de un proyecto en Rapidminer

2.2. Data Understanding

Los datos se encuentran en la librería digital IEEE Xplore. Una vez accedemos a una de las revistas ya mencionadas, tenemos que elegir el año (2015), y finalmente la edición(Issue). Para acceder a los distintos textos, hay que seleccionarlos y descargarlos. En la subfigura 2.3a se observa la situación que acabamos de describir. Descargando el documento en formato *txt* accedemos al contenido de dicho documento (subfigura 2.3b).



(a) Panel de descarga de las distintas muestras


"IEEE Transactions on Image Processing publication information," in *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. C2-C2, Jan. 2015.
doi: 10.1109/TIP.2015.2391686
Abstract: Provides a listing of current staff, committee members and society officers.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7008582&isnumber=6978084

X. Yang, X. Gao, D. Tao, X. Li and J. Li, "An Efficient MRF Embedded Level Set Method for Image Segmentation," in *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 9-21, Jan. 2015.
doi: 10.1109/TIP.2014.2372615
Abstract: This paper presents a fast and robust level set method for image segmentation. To enhance the robustness against noise, we embed a Markov random field (MRF) energy function to the conventional level set energy function. This MRF energy function builds the correlation of a pixel with its neighbors and encourages them to fall into the same region. To obtain a fast implementation of the MRF embedded level set model, we explore algebraic multigrid (AMG) and sparse field method (SFM) to increase the time step and decrease the computation domain, respectively. Both AMG and SFM can be conducted in a parallel fashion, which facilitates the processing of our method for big image databases. By comparing the proposed fast and robust level set method with the standard level set method and its popular variants on noisy synthetic images, synthetic aperture radar (SAR) images, medical images, and natural images, we comprehensively demonstrate the new method is robust against various kinds of noises. In particular, the new level set method can segment an image of size 500 × 500 within 3 s on MATLAB R2010b installed in a computer with 3.30-GIIz CPU and 4-GB memory.
keywords: {Markov processes;Image segmentation;set theory;visual databases;AMG;MATLAB R2010b;MRF embedded level set method;Markov random field;SFM;algebraic multigrid;big image databases;energy function;image segmentation;medical images;natural images;noisy synthetic images;robust level set method;sparse field method;synthetic aperture radar image;Active contours;Computational modeling;Equations;Image segmentation;Level set;Mathematical model;Noise;Level set;Markov random field;algebraic multigrid;sparse field method},
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6960855&isnumber=6978084

(b) Formato inicial de cada muestra

Figura 2.3: Acceso a los datos

En este punto del proyecto no se pueden sacar más conclusiones acerca de los datos, ya que hará falta una etapa de preparación y de integración para poder sacar más aspectos relevantes acerca de ellos.

2.3. Data Preparation

Esta fase se dividirá en dos subetapas. La primera de estas será la integración de los datos en una carpeta de fácil acceso. La otra fase consistirá en tareas de transformación de los datos, específicas de un proyecto de minería de texto.

2.3.1. Selección y limpieza de datos

En esta fase se preparará el conjunto de muestras. En primer lugar se deberán descargar los distintos documentos a analizar, de tal forma que queden fácilmente accesibles y manipulables. Para ello se siguieron los siguientes pasos:

2.3.1.1. Descarga de los artículos de la web:

Para cada edición (issue) del año 2015 (en total 12 ediciones por año) se descargaron de una vez todos los documentos (año/edición/revista) en un solo documento de texto(.txt). Con lo cual, para cada revista se tendrán 12 documentos de texto, donde cada uno de ellos contiene todas las muestras relativas a esa edición. Finalmente se integraron las 12 ediciones de cada revista en un solo archivo.

2.3.1.2. Depurado e integración:

1. Una vez se tienen todas las muestras a analizar en un archivo de texto para cada clase (revista), se procedió a la limpieza de los datos. Esta limpieza consistió en extraer de cada artículo aquellas partes que nos interesan: *Título,Resumen (Abstract),Palabras Claves.* Esta tarea se llevó a cabo con el código adjunto en la documentación: *LimpiezaDeArticulos.py.*

2. Después de esta operación se decidió separar todos los artículos en ficheros independientes.Esta tarea se llevó a cabo con el código adjunto en la documentación: *SeparacionDeArticulos.py*

Finalmente, se tienen 215 artículos de la revista *Medical Imaging* y 461 artículos de la revista *Image Processing.*

2.3.1.3. Cargando el conjunto de datos en RapidMiner:

Para poder acceder a los datos desde la herramienta *RapidMiner*, simplemente debemos añadir al panel de *Process* el bloque: *Process Documents from files* . Una vez se ha realizado este paso, en el panel *Parameter* se darán las direcciones de la carpeta contenedora de los distintos artículos y además se tendrá que especificar el nombre de la clase de da cada dirección.

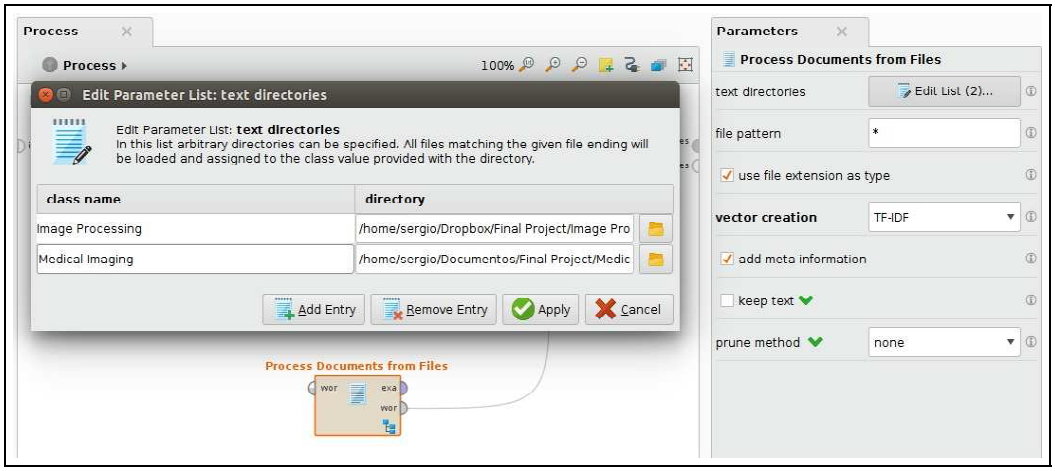


Figura 2.4: Panel: Process Documents from Files: Cargando conjunto de datos

2.3.2. Minería de Texto: construcción y formateo de datos

Las transformaciones que se realizarán a continuación son las propias de un proyecto de minería de texto. De forma simplificada se realizaron las siguientes:

1. Convertir todo el texto a minúscula/mayúscula. Se decidió pasarlo a minúscula.

2. Separación del texto en palabras(token). (Esto se hará de tal forma que no se distinguirá entre Abstract, Keywords y título).

3. Eliminación de *Stopwords*

4. *Lematización*: obtención de la raíz de las palabras. Después de numerosas pruebas se decidió utilizar el algoritmo *Snowball*, descartando los algoritmos *Porter*, *Lovins*.

5. Eliminiación de *token* demasiado pequeños: aquellos *token* con menos de dos caracteres fueron descartados.

6. Localización de palabras compuestas (*n-Grams*): esta etapa aumenta demasiado la dimensionalidad,y no siempre mejora los resultados, por ellos no siempre se llevó a cabo. Mas adelante se darán más especificaciones.

La figura 2.5 muestra los bloques que se han incorporado para realizar estas tareas. Estos bloques se corresponden de izquierda a derecha con la numeración asignada a los distintas etapas.Hay que tener en cuenta que estos bloques se han colocado en el interior del bloque *Process Documents From File*.

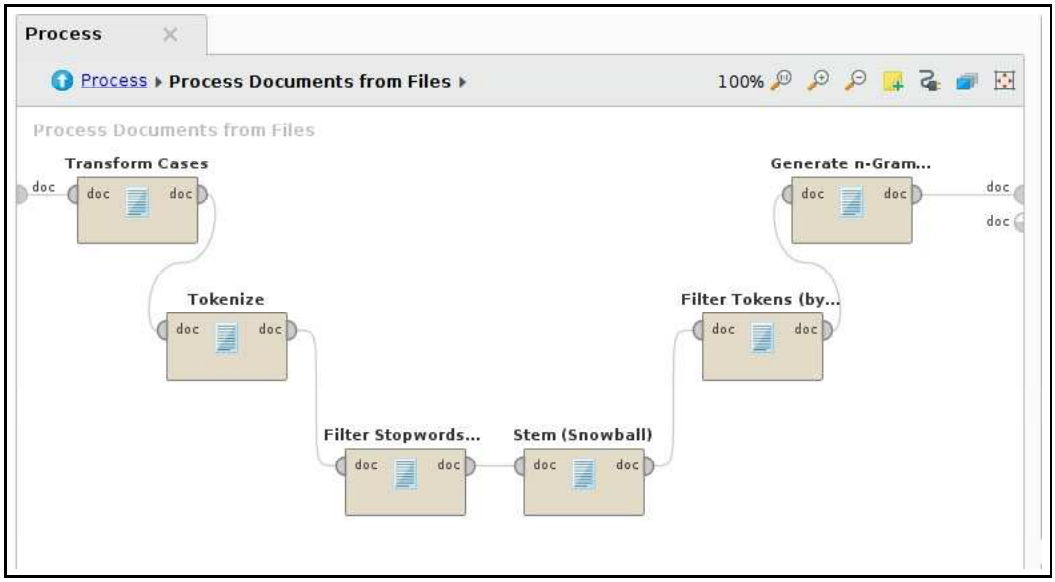


Figura 2.5: Preprocesado con *RapidMiner*

Una vez que hemos pasado por estos procesos, podemos convertir cada texto no estructurado en una estructura numérica, dando lugar así a un modelo de espacio vectorial, el cual sugiere que cada texto será representado como un vector cuyas componentes representan las distintas palabras del *vocabulario*, el valor asignado a dichas componentes será un número que representa la importancia de esa palabra en ese texto. De las distintas técnicas existentes para asignar este valor, **se usará el modelo TF-IDF**. Esta opción se pondrá en los parámetros del bloque *Process Document from Files*.

En la imagen 2.6 se puede observar el resultado del preproceso aplicando el modelo TF(Term Frecuency).(Este no es el modelo que se ha aplicado para realizar la minería de datos, solo se ha aplicado para visualizar los resultados y poder ver aquellas palabras que tenían mayor frecuencia.)

Word	Attribute Name	Total Occurences ↓	Document Occurenc...	ImageProcessing	MedicalImaging
imag	imag	6083	651	3863	2220
method	method	1797	511	1214	583
model	model	1583	347	1103	480
base	base	1182	419	839	343
propos	propos	1131	530	910	221
data	data	1111	356	704	407
featur	featur	1060	239	891	169
algorithm	algorithm	982	299	678	304
segment	segment	921	148	477	444
use	use	842	414	466	376
estim	estim	807	212	531	276
learn	learn	795	169	661	134
reconstruct	reconstruct	779	163	325	454
process	process	750	417	458	292
comput	comput	736	296	447	289
analysi	analysi	715	321	504	211

Figura 2.6: Visualización de los datos después del preproceso aplicando el modelo TF (Terms Frequency)

2.4. Modelling

2.4.1. Seleccionar la técnica de modelado

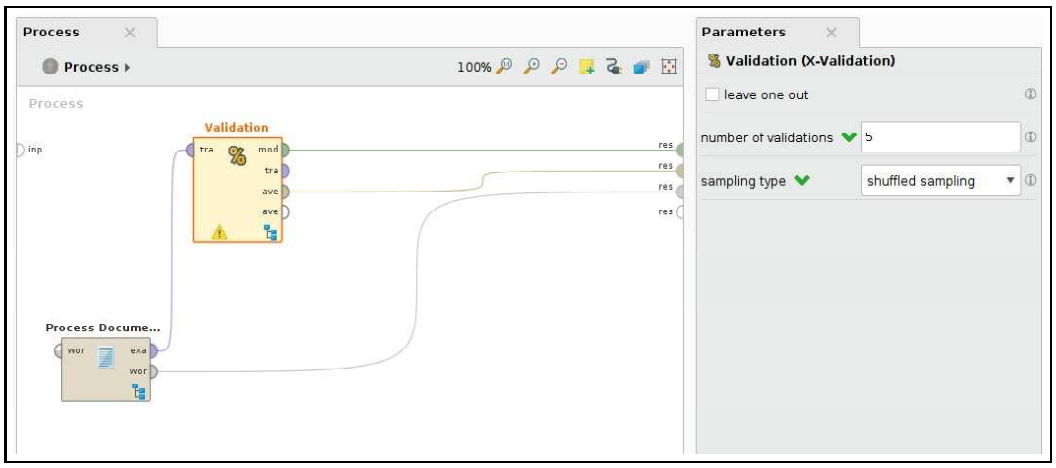
Se optó por utilizar los siguientes clasificadores:

1. K vecinos más cercanos (k-NN)
2. Arboles de decisión
3. Random Forest
4. Naives Bayes
5. SVM (Support Vector Machine)

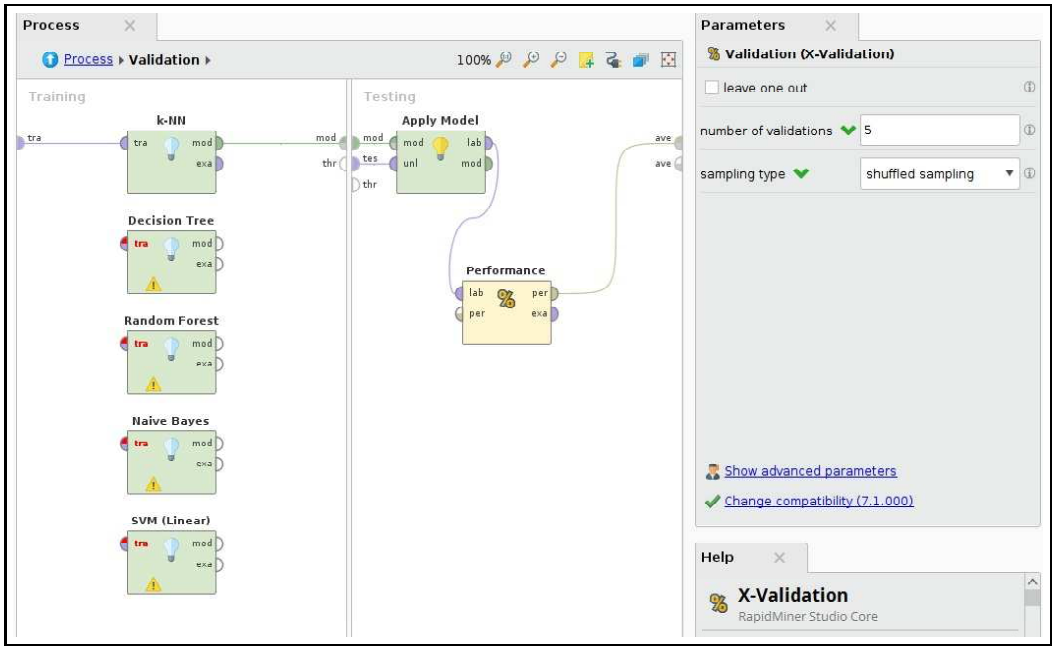
Para realizar esta tarea en *RapidMiner* habrá que insertar el bloque *Validation* (subfigura 2.7a). Dentro de este bloque se insertarán los distintos clasificadores. La subfigura 2.7b muestra el interior de este bloque. Se observa dos paneles, un subpanel es para añadir los distintos clasificadores y otro subpanel para añadir las tarea de *Testing*.

2.4.2. Generar el diseño de la prueba

El proceso de entrenamiento/validación se realizará con un *k-fold cross-validation*, particularmente se utilizó k=5. Es importante recalcar que las muestra se tienen que aleatorizar. Esto se conseguirá marcando la casilla *shuffled sampling*.



(a) Bloque: *Validation*



(b) Interior del bloque *Validation*

Figura 2.7: Generación de los modelos

2.4.3. Construcción y validación del modelo

Todos los modelos que se van a explicar a continuación se dividen en dos subetapas:

1. Se buscará el mejor funcionamiento del algoritmo variando sus parámetros característicos, **sin realizar el preprocesado n-Grams**.

2. Una vez se tienen el mejor modelo, se aplicará el preproceso con n-Grams(n=2) y se compararán los resultados.

2.4.3.1. k-Vecinos mas cercanos

1. **Mejor funcionamiento del algoritmo *sin n-Grams*:** Se probó este algoritmo con los siguientes vecinos: *k=1,3,5,7,10,15,20*. El que mejor resultados dio fue para k=7.

2. **Comparación con n-Grams (n=2):** A continuación se exponen las matrices de confusión obtenidas. La subimagen 2.8a muestra los resultados aplicando el preprocesado *n-Grams (n=2)* y la subimagen 2.8b sin usarlo.

accuracy: 91.12% +/- 2.20% (mikro: 91.12%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	443	42	91.34%
pred. MedicalImaging	18	173	90.58%
class recall	96.10%	80.47%	

(a) Clasificador: k-NN(k=7). Preprocesado: n-Grams (n=2)

accuracy: 91.27% +/- 2.17% (mikro: 91.27%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	440	38	92.05%
pred. MedicalImaging	21	177	89.39%
class recall	95.44%	82.33%	

(b) Preprocesado: Sin n-Grams

Figura 2.8: Matrices de confusión con algoritmo k-NN. *Parámetros:k=7*

Se observan mejores resultado sin la etapa de *n-Grams*. Se observa en ambos casos una buena tasa de aciertos con buenos resultados tanto en *recall* y en *precision*.

2.4.3.2. Árboles de decisión

1. **Mejor funcionamiento del algoritmo *sin n-Grams*:**

   Los parámetros que se probaron fueron los criterios para la construcción del árbol:

   - Gain ratio
   - Information gain
   - Gini index

   También se variaron los valores de *maximal depth y minimal gain*. Para la profundidad de los arboles de decisión sólo se usaron valores relativamente pequeños para que puedan ser interpretados visualmente con facilidad.

   Los mejores resultados fueron: *Information gain, minimal gain= 0.1, maximal depth=5.*

2. **Comparación con n-Grams (n=2):**

   A continuación se exponen las matrices de confusión obtenidas. La subimagen 2.9a muestra los resultados aplicando el preprocesado *n-Grams (n=2)* y la subimagen 2.9b sin usarlo.

accuracy: 91.71% +/- 1.84% (mikro: 91.72%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	436	31	93.36%
pred. MedicalImaging	25	184	88.04%
class recall	94.58%	85.58%	

(a) Preprocesado: n-Grams (n=2)

accuracy: 90.24% +/- 2.17% (mikro: 90.24%)

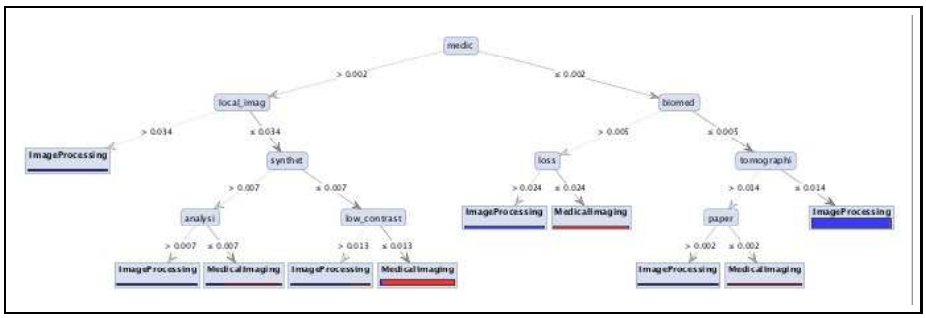
	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	433	38	91.93%
pred. MedicalImaging	28	177	86.34%
class recall	93.93%	82.33%	

93.93%

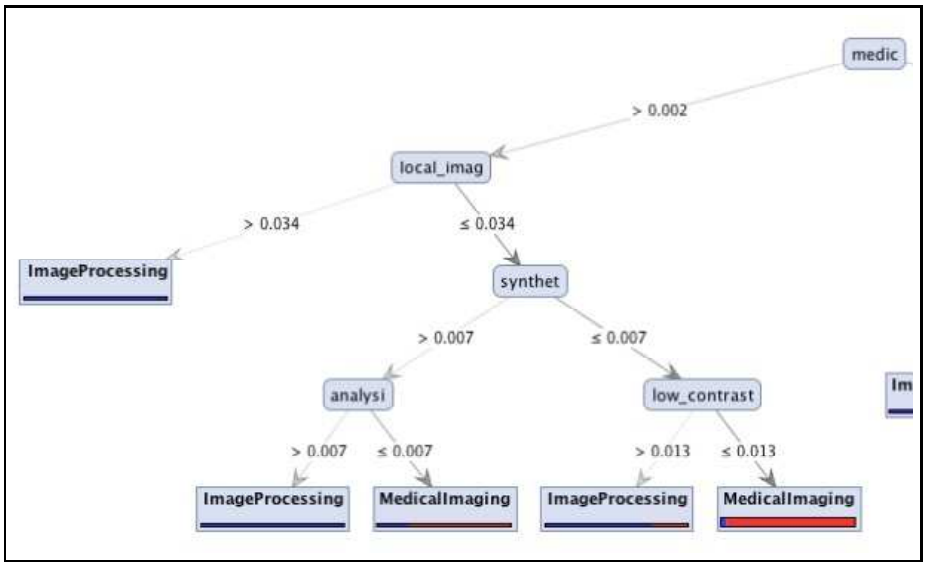
(b) Preprocesado: Sin n-Grams

Figura 2.9: Matrices de confusión con algoritmo Árbol de decisión. *Parámetros:Information Gain, minimal gain=0.1,maximal depeth=5.*

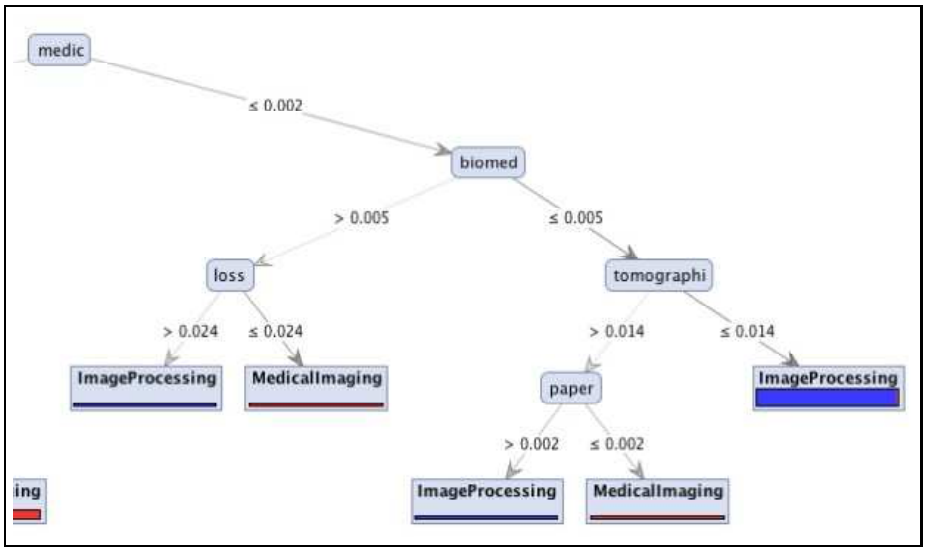
A diferencia del algoritmos anterior, se observan mejores resultados utilizando preprocesamiento con *n-Grams*. A continuación se muestra el árbol de decisión obtenido en el mejor de los dos (con *n-Grams*).



(a) Arbol de decisión completo



(b) Zoom en la parte izquierda del árbol de decisión



(c) Zoom en la parte derecha del árbol de decisión

Figura 2.10: Árbol de decisión con n-Grams(n=2)

En ambos casos, se observa una buena tasa de aciertos, con buenos resultados tanto en

recall y en *precision*.

2.4.3.3. Random Forest

1. **Mejor funcionamiento del algoritmo *sin n-Grams*:**

   Se utilizaron los mismos parámetros que en el apartado anterior para cada Árbol de Decisión, es decir: *Information gain, minimal gain= 0.1, maximal depth=5.* . Se varió el parámetro *number of trees* utilizando los siguientes valores $\{5, 10, 15, 20\}$. Los mejores resultados se obtuvieron con *number of trees*=10.

2. **Comparación con n-Grams (n=2):**

   A continuación se exponen las matrices de confusión obtenidas. La subimagen 2.11a muestra los resultados aplicando el preprocesado *n-Grams (n=2)* y la subimagen 2.11b sin usarlo.

accuracy: 68.19% +/- 3.58% (mikro: 68.20%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	461	215	68.20%
pred. MedicalImaging	0	0	0.00%
class recall	100.00%	0.00%	

(a) Preprocesado: n-Grams (n=2)

accuracy: 68.19% +/- 3.58% (mikro: 68.20%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	461	215	68.20%
pred. MedicalImaging	0	0	0.00%
class recall	100.00%	0.00%	

(b) Clasificador: Preprocesado: Sin n-Grams

Figura 2.11: Matrices de confusión con algoritmo Random Forest.*Parámetros: number of trees=10*

Se observa que la tasa de aciertos, no es muy buena. Tampoco lo son sus indicadores *recall* y *precision*. Se observa que este algoritmo solo predice la la clase *Image Processing*.

2.4.3.4. Naive Bayes

1. **Mejor funcionamiento del algoritmo *sin n-Grams*:**

   Este algoritmo no tiene parámetros.

2. **Comparación con n-Grams (n=2):** A continuación se exponen las matrices de confusión obtenidas. La subimagen 2.12a muestra los resultados aplicando el preprocesado *n-Grams (n=2)* y la subimagen 2.12b sin usarlo.

accuracy: 91.86% +/- 0.47% (mikro: 91.86%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	428	22	95.11%
pred. MedicalImaging	33	193	85.40%
class recall	92.84%	89.77%	

(a) Preprocesado: n-Grams (n=2)

accuracy: 87.27% +/- 2.68% (mikro: 87.28%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	403	28	93.50%
pred. MedicalImaging	58	187	76.33%
class recall	87.42%	86.98%	

(b) Preprocesado: Sin n-Grams

Figura 2.12: Matrices de confusión con algoritmo Naives Bayes

En ambos casos se observa una buena tasa de aciertos, con buenos resultados tanto en *recall* y en *precision*. Aunque ambos resultados son buenos, los resultados con n-Grams son mejores que sin n-Grams.

2.4.3.5. Support Vector Machines (SVM)

1. **Mejor funcionamiento del algoritmo *sin n-Grams*:**

   Para este algoritmo se probaron los siguientes kernel:

   - linear
   - radial
   - polynomial
   - neural
   - anova
   - multiquadratic

   Los experimentos se hicieron variando el parámetros *kernel degree*.Los mejores resultados se obtuvieron con : *polynomial, kernel degree=1.0*.

2. **Comparación con n-Grams (n=2):** A continuación se exponen las matrices de confusión obtenidas. La subimagen 2.13a muestra los resultados aplicando el preprocesado *n-Grams (n=2)* y la subimagen 2.13b sin usarlo.

   Los resultados fueron los siguientes:

accuracy: 68.63% +/- 3.34% (mikro: 68.64%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	461	212	68.50%
pred. MedicalImaging	0	3	100.00%
class recall	100.00%	1.40%	

(a) Preprocesado: n-Grams (n=2)

accuracy: 91.71% +/- 1.44% (mikro: 91.72%)

	true ImageProcessing	true MedicalImaging	class precision
pred. ImageProcessing	448	43	91.24%
pred. MedicalImaging	13	172	92.97%
class recall	97.18%	80.00%	

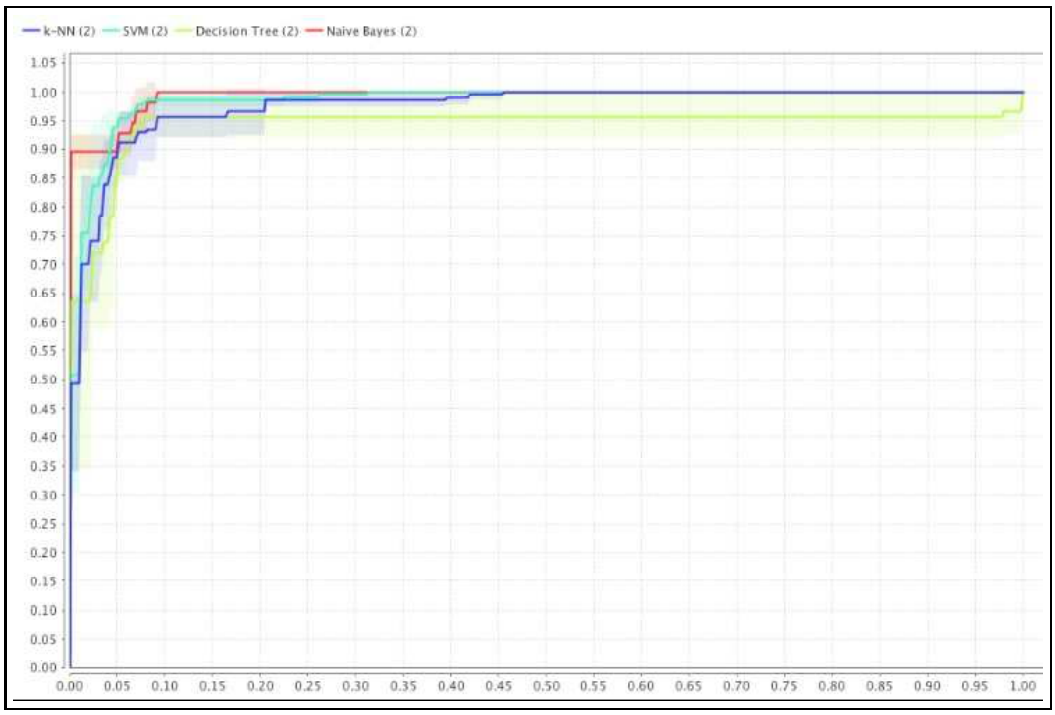
(b) Preprocesado: Sin n-Grams

Figura 2.13: Matrices de confusión con algoritmo SVM.*Parámetros:polynomial, kernel degree=1.0*

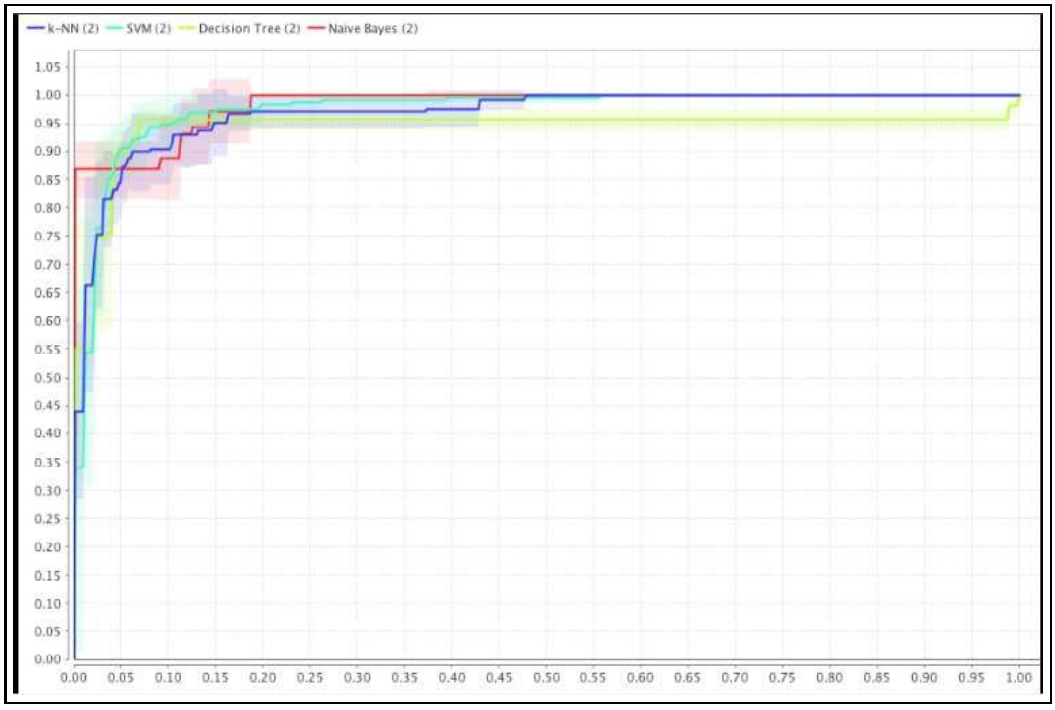
Se aprecia que los resultados son mejores sin *n-Grams*. El clasificador con *n-Grams* solo predice la clase *ImageProcessing*, por ello no tiene buenos resultados en *recall* y *precision*.

2.4.3.6. Curvas ROC

A continuación se mostrarán las curvas ROC para todos los clasificadores. La subfigura 2.14a muestra la curva ROC obtenida con los distintos clasificadores incorporando en el preproceso *n-Grams(n=2)*. La subfigura 2.14b muestra la curva ROC obtenida con los distintos clasificadores sin incorporar *n-Grams*. Se ha omitido incorporar el clasificador Random Forest, debido a que sus resultados estan muy por debajo de los conseguidos con el resto. Los colores se corresponden con los clasificadores de la siguientes forma: *rojo - Naive Bayes, azul - vecino más cercano, verde - SVM, gris - Árbol de decisión*



(a) Preprocesado: n-Grams (n=2)



(b) Preprocesado: Sin n-Grams

Figura 2.14: Curvas ROC.

2.4.3.7. Comparación de las tasas de acierto

La tabla 2.1 muestra una comparación de las mejores tasas de acierto que se han obtenido:

Clasificador	n-Grams (n=2)	Ratio de aciertos
Naive Bayes	si	91.86
Árbol de decisión	si	91.71
SVM	no	91.71
k-NN (k=2)	no	91.27
RF	-	68.2

Tabla 2.1: Mejores resultados obtenidos

2.5. Evaluation

2.5.1. Evaluar los resultados con respecto a los factores de éxito

Se estableció a priori que quedaríamos satisfechos con el trabajo si conseguíamos una tasa de éxito superior al 90 %. En la tabla 2.1 se puede ver que se ha conseguido superar ese umbral con varios clasificadores. Sin embargo el clasificador Random Forest se debe descartar por no llegar al umbral de aciertos.

2.5.2. Revisión del proceso

Al comenzar el proyecto, los clasificadores fueron entrenados con el preproceso *n-Grams*. Mas adelante se cayó en la cuenta que no siempre mejoraban los resultados. Con lo cual se hicieron nuevamente los experimentos, comparando los distintos resultados, tal y como se ha podido observar en el desarrollo de esta memoria. Se considera que este hecho forma parte de esta etapa de la metodología CRISP-DM.

2.5.3. Determinar los siguientes pasos a seguir

El proyecto ha terminado con éxito. Se procede a ordenar todas las tareas realizadas y ponerlas en formato apto para realizar la entrega.

2.6. Deployment

Se considera que esta etapa no es posible abarcarla con este proyecto,.