Trabajo final: Text Mining

Silvia García Hernández

Junio 2019

1. Objetivos

El objetivo de este trabajo de curso es la realización de un proyecto de Minería de Datos más concretamente de Minería de Textos (Text Mining). Para ello se deberán recolectar los datos necesarios para realizar el proceso, aplicar las técnicas de preprocesado que se consideren necesarias y finalmente aplicar un modelo descriptivo (clustering) así como obtener el mejor modelo predictivo (clasificación), evaluando los resultados obtenidos mediante las medidas correspondientes. Como resultado del trabajo de curso se realizará un documento explicativo del proceso realizado y analizando los resultados, así como los programas realizados en el lenguaje de programación Python durante la ejecución del trabajo.

2. Data Understanding

El trabajo de *Text Mining* que se ha realizado se ha hecho sobre artículos de revistas científicas, con el objetivo de poder determinar la revista de origen de cada artículo extraído. Se ha contado con un total de 781 artículos recolectados de 3 revistas diferentes, cuyos títulos son «*Journal of Visual Communication and Image Representation*», «*Data & Knowledge Engineering*» y «*Computer Vision and Image Understanding*». Estas revistas se han encontrado en el enlace: `https://www.sciencedirect.com/`, que ha proporcionado la base de este estudio.

Se ha añadido en la Figura 1 el aspecto de la información de los artículos descargados, con tal de que se puedan observar los diferentes campos que posee, así como el aspecto de cada uno de ellos.

3. Data Preparation

Dentro de estos 781 artículos, se han utilizado los campos relacionados con el «Título», «*Keywords*» y «*Abstract*» tanto paara el *Clustering* como para la clasificación, por lo que el resto de información se ha eliminado, al no ser de interés. Lo que se ha realizado en este apartado es la creación de un fichero

```
@article{MOEINI20171,
title = "Gender dictionary learning for gender classification",
journal = "Journal of Visual Communication and Image Representation",
volume = "42",
pages = "1 - 13",
year = "2017",
issn = "1047-3203",
doi = "https://doi.org/10.1016/j.jvcir.2016.11.002",
url = "http://www.sciencedirect.com/science/article/pii/S1047320316302255",
author = "Hossein Moeini and Saeed Mozaffari",
keywords = "Gender classification, Real-world face images, Dictionary learning, Probability decision
making, Sparse representation",
abstract = "Human gender is one of the important demographic distinctiveness for facial image
description. In this paper, a novel method is proposed for gender classification from real-world
images under wide ranges of pose, expression and so on. To this end, an automatic feature extraction
method is proposed by two types of features. Then, two separate dictionaries for male and female
genders are defined for representing the gender in facial images. Also, two dictionary learning
methods are proposed to learn the defined dictionaries in training process. Then, the Sparse
Representation Classification (SRC) is adopted for classification in the testing process. Finally, a
probability decision making approach is proposed to classify the gender from estimated values by SRC
and proposed gender formulation. Convincing results are obtained for gender classification on three
publicity databases including the FERET, LFW and Groups databases compared to several state-of-the-
arts."
}
```

Figura 1: Ejemplo de datos descargados

'.csv' donde las filas simbolizan los diferentes artículos y las columnas son las características relacionadas con cada uno de ellos. Estos atributos que conforman las diferentes columnas son los 3 apartados anteriormente citados, el año de publicación y la revista a la que pertenecen, ya que la etiqueta de cada muestra vendrá determinado por el valor de este último.

Se incluye también una muestra del aspecto de este archivo '.csv' que se ha denominado «*articles.csv*». También es importante reseñar la cantidad de muestras por clase que existen en este trabajo, la cual está bastante desbalanceada, encontrándose 433, 249 y 99 en cada una. En la Figura 3 se incluye el detalle de este dato.

	entry	title	keywords	abstract	year	journal
0	AZAZA20181	Context proposals for saliency detection	Computational saliency, Object segmentation, O...	One of the fundamental properties of a salient...	2018	Computer Vision and Image Understanding
1	BENSHABAT201812	Graph based over-segmentation methods for 3D p...	3D point cloud over-segmentation, 3D point clo...	Over-segmentation, or super-pixel generation, ...	2018	Computer Vision and Image Understanding
2	YANG201843	Text effects transfer via distribution-aware t...	Text effects, Texture synthesis, Spatial distr...	In this paper, we explore the problem of fanta...	2018	Computer Vision and Image Understanding
3	BARATH201870	Efficient energy-based topological outlier rej...	Stereo vision, Outlier filtering, Energy minim...	An approach is proposed for outlier rejection ...	2018	Computer Vision and Image Understanding
4	ARRIGONI201895	Robust synchronization in SO(3) and SE(3) via ...	Absolute rotations, Global rotations, Structur...	This paper deals with the synchronization prob...	2018	Computer Vision and Image Understanding

Figura 2: Ejemplo de muestras estructuradas

Una vez se tuvieron las muestras recopiladas y ordenadas, se pasó al proceso de limpieza. En este caso, la limpiza ha consistido en la eliminación de muestras con valores perdidos, así como otras técnicas de propias de la minería de texto que 'facilitan' el posterior procesamiento de las mismas, y que se detallarán más adelante.

En un primer paso, se eliminaron los artículos que no poseían algunos de los tres atributos indispensables de este estudio, esto es, el título, las palabras clave

journal	entry	title	keywords	abstract	year
Computer Vision and Image Understanding	249	249	249	249	249
Data & Knowledge Engineering	99	99	99	99	99
Journal of Visual Communication and Image Representation	433	433	433	433	433

Figura 3: Cantidad de muestras por clase

y el resumen. Además, la lista inicial de muestras poseían también duplicaciones de algunas de las entradas, por lo que también se procedió a la eliminación de estas. En resumen, se detalla a continuación la cantidad de muestras descartadas y la razón de cada una:

- «*Journal of Visual Communication and Image Representation*»: se eliminaron 6 muestras que no contenían el campo *keywords*. También se eliminó una duplicación de entrada.
- «*Data & Knowledge Engineering*»: Se eliminaron 6 entradas por la misma razón anteriormente expuesta.
- «*Computer Vision and Image Understanding*»: 14 muestras eliminadas y una duplicación solventada.

Los tamaños de cada clase expuestos en la Figura 3, hacen referencia a la cantidad de muestras por revista una vez se ha limpiado el «*dataset*».

Para preparar los datos de entrada para las etapas de *Clustering* y Clasificación, hace falta convertir las diferentes palabras de cada muestra a valores numéricos que puedan interpretar los métodos de aprendizaje. Para ello, y para que sea clara y sencilla la explicación de los pasos seguidos, se va a dividir la teoría explicada a continuación en las siguientes etapas: Limpieza y procesado de muestras, Extracción de características y Selección de características. Dentro de cada una de estas etapas existen diferentes conceptos que se han utilizado con el objetivo de obtener el mejor rendimiento del sistema. A continuación se detallan los diferentes métodos y algoritmos utilizados en cada apartado.

se va a hacer uso de 3 conceptos básicos muy utilizados en la literatura de la minería de textos, los cuales son el algoritmo de Porter, el método Term frequency–Inverse document frequency (TF–IDF) y *Latent semantic analysis* (LSA). A continuación se explica en detalle el uso que se le ha dado a cada método, así como la utilidad que tienen en el preprocesamiento de datos del presente trabajo.

1. **Limpieza y procesado de texto**:
   a) **Eliminación de signos de puntuación**. Se han eliminado los signos de puntuación y se ha asegurado que las palabras unidas por

un guion se separen en dos palabras diferentes y no sean una única palabra.

	title	journal
0	Context proposals for saliency detection	Computer Vision and Image Understanding
1	Graph based over-segmentation methods for 3D p...	Computer Vision and Image Understanding
2	Text effects transfer via distribution-aware t...	Computer Vision and Image Understanding
3	Efficient energy-based topological outlier rej...	Computer Vision and Image Understanding
4	Robust synchronization in SO(3) and SE(3) via ...	Computer Vision and Image Understanding

Figura 4: Paso 1. Eliminación de signos de puntuación

b) **Lemmatización**. En esta pase se consigue agrupar las palabras según su significado, ya que son todas reunidas a una misma raíz semántica. Se ha utilizado '*WordNetLemmatizer()*' en el paquete NLTK de Python.

	0
0	Context proposal for saliency detection
1	Cascade residual guided nonlinear dictionary l...
2	Visual question answering Datasets algorithm a...
3	Recognizing semantic correlation in image text...
4	Human Attention in Visual Question Answering D...

Figura 5: Paso 2. Lemmatización de palabras

c) **Stemming**. Algoritmo de Porter: se encarga del «*Stemming*» que consiste en la transformación de las diferentes palabras (o tokens) a su raíz (lexema), eliminando los diferentes morfemas de las mismas (solamente los sufijos). Este algoritmo también normaliza mayúsculas y minúsculas, lo que significa que convierte las primeras en las segundas. Este método[1] se encuentra en el paquete Natural Language Tool Kit (NLTK) de Python.

2. **Extracción de características**:

a) **TF–IDF**. Con este método también se ha tratado la eliminación de «*stop–words*» y la «*tokenización*», además de realizar la asignación de valores numéricos a cada palabra para crear la matriz Término–Documento (TD).

[1] https://www.nltk.org/api/nltk.stem.html

	0
0	context propos for salienc detect
1	cascad residu guid nonlinear dictionari learn
2	visual question answer dataset algorithm and f...
3	recogn semant correl in imag text weibo via fe...
4	human attent in visual question answer do huma...

Figura 6: Paso 3. Stemming de palabras

Para calcular este valor numérico, –que se denomina 'peso' (W)–, se realiza el producto de la frecuencia de un término determinado (TF) en un cierto documento por la frecuencia inversa del término en los diferentes documentos, esto es:

$$W(t,d) = TF(t,d) * IDF(t,d) = TF(t,d) * \log(N/DF(t)) \qquad (1)$$

donde:

- TF(t,d) es el número de apariciones del término (t) en el documento (d).
- DF(t) es el número de documentos que contienen el término (t).
- N es el número total de documentos (d).

Con este método se busca penalizar las palabras que aparecen en todos los documentos, es decir, que no son específicas o propias de un conjunto determinado de documentos (que simbolizará una clase). Dentro de Python, la librería «*sklearn*» provee este método[2].

	abil	access	accord	account	accur	accuraci	achiev	achiev state	acquir	acquisit	...	weight	wide	wide rang	window	word	work	work propo	world
0	0.000000	0.0	0.0	0.0	0.000000	0.00000	0.000000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.000000	0.0	0.0	0.0	0.066209	0.00000	0.052469	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.117459	0.0	0.0	0.0	0.000000	0.00000	0.000000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.000000	0.0	0.0	0.0	0.000000	0.04208	0.039032	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.000000	0.0	0.0	0.0	0.000000	0.00000	0.000000	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figura 7: Paso 4. Construcción de matriz Término–Documento

b) **TF**. Este método es muy parecido al anterior, ya que realiza la primera parte de conteo de palabras por documento y construye la matriz Término–Documento en función de eso. Se ha utilizado la función *CountVectorizer()* de Python.

[2]https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

c) **Hashing**. El método de Hashing es otro algoritmo de creación de matrices Término–Documento. Asigna un valor numérico a cada palabra contenida en los documentos, que no está relacionado con la frecuencia de las mismas. En este caso se ha usado la función *HashingVectorizer()*.

3. **Reducción de características**:

   a) **LSA**. Dado que la salida de los métodos de extracción de características TF–IDF, TF y Hashing es muy grande, esto es, el número de atributos está sobredimensionado, hace falta utilizar un algoritmo que reduzca esta dimensionalidad, además de proporcionar más calidad a estos atributos. De esto se encarga el *Latent semantic analysis*.

   b) **LDA**. Latent Dirichlet Allocation es un modelo de conceptos que genera temas basados en la frecuencia de palabras de un conjunto de documentos. LDA es particularmente útil para encontrar mezclas razonablemente precisas de temas dentro de un conjunto de documentos dado.

Por lo que todos los tópicos importantes de tratar en la minería de textos quedan cubiertos con los tres puntos anteriores. Sacados de la literatura y con el fin de enumerarlos y tenerlos controlados, a continuación se añaden estos apartados a tratar: Separar en palabras (tokens), eliminar stopwords, Stemming/Lemmatization, normalizar palabras, corregir errores ortográficos, detectar los finales de frases y normalizar minúsculas/mayúsculas.

4. *Modeling*

Se ha dividido tanto este apartado como el siguiente en dos subapartados, ya que se han implementado tanto métodos de aprendizaje no supervisado (clustering) como métodos de aprendizaje basado en etiquetas (clasificación).

4.1. *Clustering*

Los métodos de aprendizaje no supervisado utilizados han sido K–means, MiniBatchKMeans y HAC (Hierarchical Clustering), concretamente el método *AgglomerativeClustering* de *sklearn*.

4.2. Clasificación

Por otro lado, para seleccionar los métodos de clasificación, se ha utilizado la bibliografía encontrada en *sklearn* que recomienda los mejores clasificadores para la minería de textos. En la Figura 8 se detallan estos métodos con el score de cada uno de ellos así como el tiempo de cómputo que conllevan.

Dentro de grupo de algoritmos utilizados se encuentran *Complement ND*, *PassiveAgressiveClassifier*, *SGD Classifier*, *SVC* y *RandomForest*.

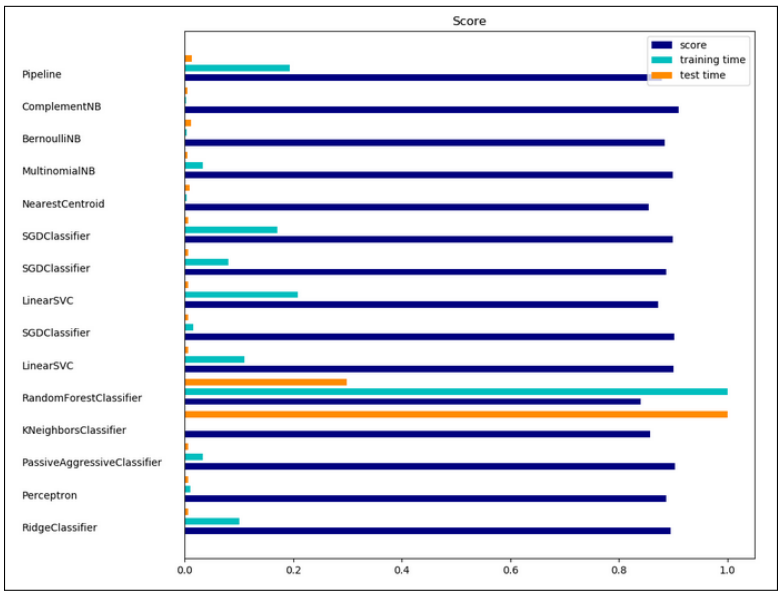

Figura 8: Ranking de clasificadores en la literatura de la minería de texto [3]

5. Evaluación

Dado que se han utilizado métodos tanto de *Clustering* como de Clasificación, primero se va a detallar las medidas empleadas para evaluar ambos tipos de métodos y, seguidamente, se expondrán y comentarán los resultados obtenidos.

Para el *Clustering* se ha calculado cuatro medidas; tres basadas en las etiquetas reales y una quinta que valora la separación y compactación de los cluster generados.

1. **Homogeneity**: Esta medida está relacionada con que cada cluster contenga solo muestras de una misma clase.

2. **Completeness**: Cuenta la cantidad de miembos de una misma clase que han sido asignados a un cluster determinado.

3. **V−measure**: Media de las dos medidas anteriores.

4. **Silhouette**: Mide la buena definición de un cluster, esto, la distancia entre muestras de un mismo cluster así como distancias entre clusters diferentes.

Por otro lado, para los métodos de claificación, se ha usado la medida *Accuracy*, ya que lo importante es determinar la cantidad de etiquetas previtas que coinciden con las etiquetas reales.

Para obtener los diferentes resultados expuestos en las diferentes tablas posteriormente visibles, se han ido alternando los posibles métodos a utilizar en

cada campo tratado y anteriormente explicado, esto es, en los apartados de 'Extracción de características' y 'Reducción de características'. Concretamente se han hecho todas las combinaciones posibles entre los 3 primeros y 2 últimos, a excepción de 'Hashing' con 'LDA', ya que no son compatibles por poseer, la salida del primero, valores negativos.

De esta forma, en cada tabla se han reflejado los resultados obtenidos para cada método de *Clustering* (Tablas 1 a 4), mientras que en la última se encuentran los valores de *Accuracy* de los métodos de clasificación.

Para construir estas tablas, se han fijado los parámetros de los distintos métodos de extracción y selección de características y se han realizado las diferentes combinaciones para comparar resultados de unos métodos y otros. Todos los resultados están referidos al campo *Abstract*, ya que es el que mejores resultados ha obtenido. Añadir que estos no son los resultados óptimos posibles, y que se comentarán posteriormente los mejores obtenidos fuera de estas tablas.

Parámetros	Comb. 1	Comb. 2	Comb. 3	Comb. 4	Comb. 5
TF–IDF	X	X			
TF			X	X	
Hashing					X
LSA	X		X		X
LDA		X		X	
Homogeneity	0.131	0.035	0.095	0.056	0.088
Completeness	0.120	0.033	0.095	0.061	0.085
V–measure	0.125	0.034	0.095	0.059	0.086
Silhouette	0.012	0.555	0.031	0.303	0.028

Cuadro 1: Método K–means

Por ejemplo, en el K–means se puede observar que los mejores resultados en cuanto a etiquetas correctas adjudicas se obtiene con TF–IDF como método de extracción de características y LSA como reducción de las mismas. Por otro lado, LDA está relacionado con buenos resultados en cuanto a compactación de clusters.

En MiniBacthKMeans en cambio, la buena asignación de etiquetas está relacionada con TF como método de extracción de características y LSA, mientras que se obtiene la mejor calidad de los clusters también con LDA, siendo mejor TF–IDF que TF para este otro campo.

En el tercer algoritmo de *Clustering* se obtienen resultados generalmente peores en cuanto a etiquetas bien asignadas, pero ampliamente superiores, también en términos globales, en cuanto a propiedades de los clusters obtenidos, ya que en todas las combinaciones se obtiene un porcentaje mayor al 50 %. No se ha utilizado el método LDA porque son incompatibles.

Por último, K–means con medida basada en la distancia coseno obtiene los mejores resultados de asignación de etiquetas con la combinación TF–IDF + LSA, aunque tampoco son despreciables los obtenidos con TF + LSA. Se puede concluir también que este algoritmo de *Clustering* no es bueno, en este contexto,

Parámetros	Comb. 1	Comb. 2	Comb. 3	Comb. 4	Comb. 5
TF–IDF	X	X			
TF			X	X	
Hashing					X
LSA	X		X		X
LDA		X		X	
Homogeneity	0.020	0.023	0.110	0.020	0.068
Completeness	0.101	0.030	0.151	0.025	0.094
V–measure	0.034	0.026	0.127	0.022	0.079
Silhouette	-0.010	0.549	0.027	0.133	0.079

Cuadro 2: Método MiniBatchKMeans

Parámetros	Comb. 1	Comb. 2	Comb. 3
TF–IDF	X		
TF		X	
Hashing			X
LSA	X	X	X
LDA			
Homogeneity	0.016	0.003	0.003
Completeness	0.023	0.150	0.150
V–measure	0.019	0.006	0.006
Silhouette	0.548	0.523	0.523

Cuadro 3: Método HAC

para generar cluster con buenas propiedades internas.

Con ciertas combinaciones de los parámetros de los diferentes métodos, se han alcanzado valores de 22 % tanto en *Homogeneity* como en *Completeness*, no habiéndose conseguido superar estas marcas. Esto seguramente sea debido a la dificultad de diferenciar las diferentes muestras de las 3 revistas, ya que dos de ellas relatan de conceptos relacionados con 'imagen' o 'visión' y juntas suman apróximadamente el 87 % del total de las muestras, por lo que de la tercera revista se tiene poca representación. Esto también se usará de argumento posteriormente para justificar los valores máximos de *Accuracy* obtenidos.

Dentro de la evaluación de los métodos de clasificación se han realizado 3 combinaciones de métodos de extracción de características y reducción de las mismas basadas en los mejores resultados obtenidos en la evaluación anterior. Es por esto que solo se ha usado TF–IDF como extractor de características, mientras que como reductores están LSA y LDA.

En términos generales, las combinaciones de TF–IDF solo y TF–IDF combinado con LSA, son mejores que la tercera combinación (TF–IDF + LDA). Entre estos dos primeros no hay un claro ganador con estos parámetros, pero Complement ND, SVC y LogicticRegresion son los métodos de clasificación que mejores resultados obtienen.

Parámetros	Comb. 1	Comb. 2	Comb. 3
TF–IDF	X		
TF		X	
Hashing			X
LSA	X	X	X
LDA			
Homogeneity	0.185	0.128	0.077
Completeness	0.168	0.111	0.067
V–measure	0.176	0.119	0.072
Silhouette	0.012	0.029	0.024

Cuadro 4: Método K–means con medida de distancia coseno

Como en el caso anterior, los resultados más altos obtenidos rondan los expuestos en la Tabla 5, esto es, no se llega al 80 %. También se achaca este hecho al argumento anteriormente expuesto, relacionado con la cantidad de muestras y la distribución de las mismas en las revistas. Cabe añadir para terminar la argumentación de los resultados, que se han probado los métodos implementados con otros dataset, es decir, otras combinaciones de revistas, y se han obtenidos resultados de un 40 % de *Homogeneity* y *Completeness* apróximadamente en *Clustering* y 82 % de *Accuracy* en clasificación.

Parámetros	TF–IDF	TF–IDF + LSA	TF–IDF + LDA
Complement ND	0.77		
PassiveAggressiveClassifier	0.719	0.719	0.676
SGDClassifier	0.727	0.736	0.646
SVC	0.74	0.719	0.608
RandomForest	0.685	0.638	
LogisticRegresion	0.74	0.744	

Cuadro 5: Resultados de *Accuracy* en Clasificación