PONTIFICIA UNIVERSIDAD CATOLICA DE VALPARAISO
FACULTAD DE INGENIERIA
ESCUELA DE INGENIERIA INFORMATICA

DETERMINACIÓN DE AUTORÍA POR MEDIO DE REDES DE PALABRAS

Sebastián Andrés Canepa Labbe

Profesor Guía: **Rodrigo Alfaro Arancibia**

Profesor Co-referente: **Héctor Allende Cid**

Carrera: **Ingeniería Civil Informática**

Diciembre 2017

Índice

Lista de Figuras

Lista de Tablas

Resumen

Hoy en día la inmensa cantidad de información disponible a través de internet se encuentra en constante crecimiento. Gran parte de ésta es texto escrito por usuarios bajo distintos contextos, por ejemplo: redes sociales, foros, bitácoras, correos electrónicos, etc. En este sentido, surge la necesidad de contar con mecanismos automáticos para facilitar el análisis de dicha información.

Una de las situaciones que en recientes años ha estado ganando interés es la Atribución de Autoría. De forma general, esta consiste en lograr identificar automáticamente los documentos de uno o más autores. Por ejemplo, existe interés en el desarrollo de métodos para hacer frente a situaciones de: verificación de mensajes terroristas, filtrado de spam, disputas por derechos de autor, etc.

Dicho esto, se han propuesto diferentes algoritmos y estrategias para llevar a cabo la Atribución de Autoría, en especial enfoques de aprendizaje automático. Con este enfoque se pretende construir clasificadores utilizando un conjunto de documentos de entrenamiento. Desafortunadamente, no siempre se tiene disponible un conjunto de documentos ideal, es decir existen escenarios donde los datos son escasos o desbalanceados.

Palabras Claves: Grafo, Estilometría, Redes de palabras, Métricas descriptivas, Stopwords, Clasificación de texto, Pos Tagging, Aprendizaje automático.

1. Introducción

Desde el comienzo de la redacción de obras literarias y el posterior análisis de estas, han llevado a expertos de la comunidad lingüista a determinar la autoría para textos anónimos y confirmarla en textos donde el autor ponga en duda. Con el paso del tiempo, nace la estilometría como análisis lingüístico de textos que permite identificar patrones únicos, los cuales permiten atribuir autoría. Algunas de sus aplicaciones son determinar la autoría de una obra, la autenticidad, clasificación de textos, medición de frecuencia de palabras, identificación de lenguas [1].

En proyectos anteriores [2] las técnicas utilizadas como frecuencia de palabras, Pagerank y Simrank demostraron malos resultados ya que los porcentajes de atribución de autoría no eran suficientemente confiables. Esto se debe a que hay distintos rasgos que deben ser considerados para identificar el estilo del autor como lo son el uso de puntuación, n-gramas de palabras longitud de palabra y oraciones etc.

En función a lo anterior, es que se propuso un nuevo enfoque utilizando métricas que nos permitieron reconocer el estilo lingüístico del autor. Este método consistió en tomar de un corpus de textos de opinión, con el que se han trabajado otros temas similares de proyectos pasados, 5 textos por cada autor y crear un grafo de palabras co-ocurrentes para cada uno, luego se le aplicaron métricas para comparar cada texto y sacar conclusiones.

Como resultado del trabajo realizado anteriormente, se pudieron observar valores similares en los resultados obtenidos por el método utilizado siendo estos no muy contundentes, en esta instancia se propondrá un método en base a Pos Tagging para crear los grafos de palabras lo cual permitirá reflejar de mejor forma la manera en que el autor construye el texto. Además se tendrán grafos más pequeños debido a la menor cantidad de nodos.

Este documento corresponde al informe final de proyecto de título para determinar autoría por medio de redes de palabras. En primer lugar, se definen los objetivos y se explica el problema a abordar durante el trabajo. Posteriormente, se expone el desarrollo del marco teórico, en donde se presentan conceptos básicos y las métricas aplicadas a los grafos. Luego, se muestran detalles del experimento y la implementación del método. Para finalizar, se presentan los resultados obtenidos y se expone la conclusión respecto a resultados y al proyecto.

2. Objetivos

A continuación, se presenta el objetivo general y los objetivos específicos del proyecto

2.1. Objetivo General

Buscar una clasificación adecuada de textos mediante el cálculo de métricas de centralidad y uso de Pos-Tagging.

2.2. Objetivos específicos

- Definir métricas descriptivas aplicables a los grafos
- Determinar métricas que permitan identificar el estilo de escritura de los autores
- Aplicar las métricas a grafos de textos
- Realizar análisis mediante Pos Tagging
- Realizar clasificación con métricas de pos tag
- Buscar relaciones entre los nuevos valores obtenidos

3. Problema

En esta sección se da una definición del problema, además se describen algunos avances en cuanto a la atribución de autoría y la aplicación que tienen los grafos de textos en esta área.

3.1. Definición

El problema principal a evaluar en este proyecto es la búsqueda de una representación de texto mediante grafos que se espera tenga un buen rendimiento en la atribución de autoría. En el problema de la atribución de autoría, se busca determinar quién es el autor de un texto mediante el entendimiento, representación y comparación de patrones que cada autor tiene como características a la hora de escribir un texto. En este proceso se construye un perfil lingüístico del autor basándose en la forma que este escribe, la profundidad de su vocabulario, la omisión o la frecuencia en que utiliza ciertas palabras [2].

Esta problemática no es nueva y abarca diversas áreas, desde la literatura hasta la criminología pasando por análisis de comentario en el internet y por el análisis de autenticidad de escritos académicos [2].

En este proyecto se buscará la autoría de textos mediante la construcción de grafos dirigidos que conformen redes de palabras, luego se le aplicarán métricas descriptivas que permitan encontrar patrones y ayuden a la atribución de autoría [3].

3.2. Avances

La atribución de autoría de textos ha tenido un historial de aplicación tan largo como la existencia de la literatura. Desde la antigüedad, la autoría ha sido atribuida a compilaciones escritas de tradiciones históricas, como la épica homérica, los libros del Antiguo Testamento y las cartas y evangelios canónicos. Sin embargo, ha sido más recientemente cuando han comenzado a plantearse cuestiones tales como si esas atribuciones son válidas. El primer análisis de atribución de autoría registrado está relacionado con un documento de la Roma Imperial, la Donación de Constantino2, un testamento utilizado por el papado en la Alta Edad Media destinado a reclamar tierras para la Iglesia. Por aquel entonces ya se venía considerando una falsificación, pero no fue hasta el año 1.440 cuando Lorenzo Valla lo demostró en base a la existencia de anacronismos lingüísticos, estilísticos y de contenido [4].

Más recientemente, la atribución de autoría se ha aplicado a la resolución de problemas surgidos con la atribución literaria, tal es el caso de las obras de Shakespeare, los Documentos Federalistas y las Epístolas de San Pablo, con diferentes resultados [5].

Con la aparición de Internet y la comunicación mediante el uso de las nuevas tecnologías, la atribución de autoría se ha ampliado a correos electrónicos, blogs, foros de chat e incluso a mensajes de texto de teléfonos móviles. Además, trayendo con si un creciente desarrollo de sistemas totalmente automatizados para analizar el lenguaje en múltiples niveles de forma mucho más eficiente que cualquier intervención humana.

3.3. Aplicaciones

En la última década, el análisis de autoría totalmente automatizado y computarizado ha crecido a pasos agigantados, desarrollándose una amplia variedad de análisis que van desde un simple perfil lingüístico a unas complejas redes neuronales y algoritmos genéticos. Algunas de las aplicaciones más conocidas en la determinación de autoría, por ejemplo, en el ámbito académico donde se emplean técnicas para reconocer la existencia de plagios entre distintos autores, determinar si una persona posee más de un usuario en un sitio web o foro, detectar identidad de acosadores que se ocultan en redes sociales en base a su forma de escribir, entre otros. A continuación, se hará referencia a algunos trabajos que mencionen estas aplicaciones:

- Detección automática de plagios en textos. [6]: Hace uso de diversas técnicas de recuperación y extracción de características, las cuales son empleadas para la comparación de los textos. De acuerdo a esa comparación se puede establecer si dos textos o fragmentos de textos son procedentes del mismo autor.
- Reconocimiento de autoría mediante patrones de estilos de escritura. [7]: Determina la atribución de la autoría por medio del estilo del autor, estilometría. El texto es tratado como una red co-ocurrente, ya que es mejor para trabajar los atributos relacionados al estilo escrito de los textos. Cada palabra del texto es representada como un nodo y como medida de medición se necesita la información de los nodos vecinos hasta en segundo grado (los vecinos de los vecinos) del nodo que se está analizando.
- Reconocimiento de acuerdo a posts en foros web. [8]: Se determina la atribución de autoría de los distintos autores que escriben post en un foro web, la idea detrás del método es generar meta características que capturen la modalidad similar especifica de las relaciones entre los textos de los diferentes autores, empleando como medida de clasificación medidas sintácticas, léxicas y de estilo.
- Reconocimiento de autor de acuerdo a distintos tipos de atributos. [9]: Se hace uso de distintos tipos de atributos proporcionan distintas perspectivas del estilo de cada documento. Se espera utilizar un conjunto de atributos que pueden retener el estilo de los autores, obtener características que tengan una relación entre el documento y el autor, y que puedan ser utilizadas como un modelo de clasificación.

4. Marco Teórico

En esta sección se presentan conceptos a utilizar en este proyecto, además de los fundamentos básicos para la comprensión de grafos como, estilometría, métricas descriptivas, atribución de autoría y clasificación automática de textos. También se realiza una explicación breve de algunos clasificadores utilizados.

4.1. Planteamiento inicial

A continuación, se definirán términos básicos para comprender la teoría de grafos y el método utilizado en este proyecto

4.1.1. Estilometría

Es la aplicación del estudio de estilo lingüístico que analiza ciertos rasgos del estilo del autor y los utiliza para comparas dos o más textos, para determinar la autoría de documentos anónimos o en disputa. El punto base de la estilometría es que el estilo es algo propio de cada autor, ya que se encuentra dentro de su subconsciente, y por esta razón, cada quien tiene un estilo propio. Dentro de las aplicaciones de la estilometría se encuentran la determinación de autoría de una obra, la autenticidad, la clasificación de textos, la medición de frecuencias de palabras, la identificación de lenguas, entre otras [6].

Cada texto tiene marcadores lexicales de estilo, los cuales determinan las características propias del estilo de cada autor permitiendo una comparación para lograr determinar la autoría. Los marcadores lexicales de estilo se dividen en dos: la riqueza del vocabulario y la frecuencia de las palabras de función.

Hoy en día, la estilometría moderna se emplea en gran medida con ayuda de computadores para el análisis estadístico, la inteligencia artificial y el acceso a todos los textos disponibles en internet. Con el uso de computadores es más fácil la detección de patrones dentro de los textos, así como el almacenamiento de las secuencias para su posterior análisis cuantitativo e identificación de secuencias de palabras. Algunos parámetros a considerar para poder realizar el análisis son el número de textos, la cantidad de autores existentes, la extensión de la lista de palabras, la frecuencia de utilización de las palabras, entre otras.

4.1.2. Atribución de autoría

Es la ciencia de inferir características de un autor de las características que poseen los documentos escritos por este. La Atribución de Autoría es considerada como uno de los problemas más viejos y nuevo a la vez dentro de la recuperación de la información.

Los principales focos de la Atribución de Autoría están relacionados con los textos escritos, aunque también es posible determinar la Autoría de obras de artes como de música. Dentro de los textos escritos se consideran las estructuras de las sentencias y las elecciones léxicas; que a su vez se descomponen en más de mil opciones para seleccionar características posibles para determinar a un autor.

En términos amplios existen tres principales problemas con la Atribución de Autoría. El primero de ellos es dado una muestra en particular de texto y un conjunto conocido de autores, poder determinar a qué autor pertenece. El segundo, dado una muestra en particular de texto que se cree que pertenece a un grupo de autores, determinar cuál de ellos escribió que parte del texto. Y el tercer problema es poder determinar cualquiera de las propiedades relacionadas al autor de una muestra de texto, ya que puede existir mucha diversidad en la forma de escribir de un autor de acuerdo a un género literario frente a otro; como también en el cambio del público objetivo, entre otros [6].

4.1.3. Clasificación automática de textos

El proceso de crear una clasificación automática de textos consiste en descubrir que variables son útiles en la discriminación de los textos que pertenecen a clases prexistentes distintas. En particular, los clasificadores (programas que ejecutan algoritmos de clasificación) son entrenados con un grupo de documentos, previamente clasificados y etiquetados por un humano, acorde a algún criterio particular, conformando una clase. De esta manera, el objetivo de estos clasificadores es decidir en qué categoría debe ir cada nuevo texto, partiendo de un esquema de clasificación previo.

También se dice que la clasificación automática de documentos puede ser entendida como una tarea en la cual, en base a la identificación por medio matemático-estadístico, un nuevo documento es asignado a una clase particular de documentos pre-existentes.

4.1.4. Grafos

Un grafo es una representación de un conjunto de objetos donde algunos pares están conectados. La conexión entre grafos es conocida como vértice o nodo, estos a su vez están unidos por aristas. Estos pueden ser dirigidos o no.

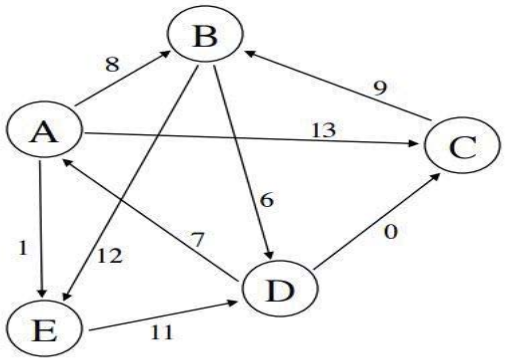


Figura 4.1.1 Representación de un Grafo

Un grafo es definido como G= (V, E) donde V es un conjunto finito V= {1, 2, 3, 4...N} de vértices y E es n conjunto de aristas [2].

4.1.5. Redes de palabras

Una red de palabras es la transformación de un texto, escrito en un determinado lenguaje, en un grafo G (V, E), donde G es el grafo, o red, compuesto por V palabras (o términos) distintas y E enlaces que las relacionan en un texto [10].

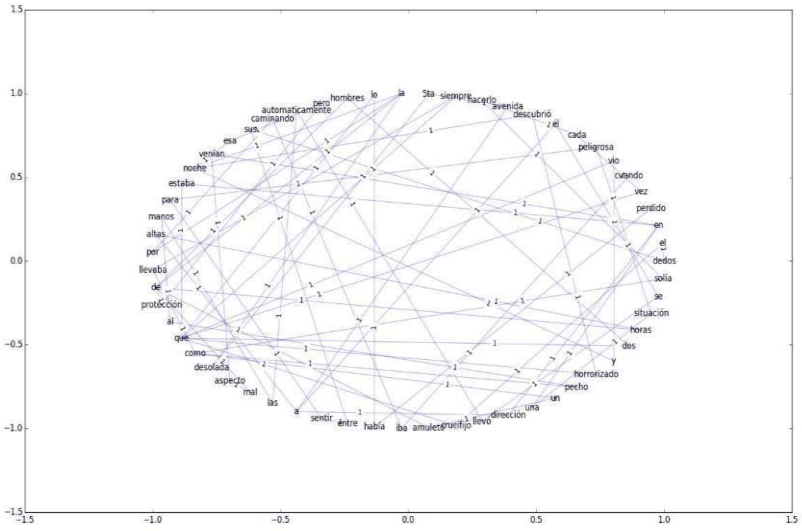


Figura 4.1.2 Red de palabras del micro cuento “Asesinato en la 5ta avenida”

Las redes de palabras [11] son objetos matemáticos del tipo G (V, E), donde G es el grafo compuesto por V palabras distintas y E enlaces que las relacionan por Co-ocurrencia en el texto. G_i Corresponde a la red de palabras Co-ocurrentes construida a partir de un conjunto de textos perteneciente a un autor i clasificado por un humano. De esta forma habrán tantos grafos G como autores de textos hayan sido entrenadas. Por otro lado, h corresponde a la red de palabras construida a partir de un nuevo texto sin clasificar.

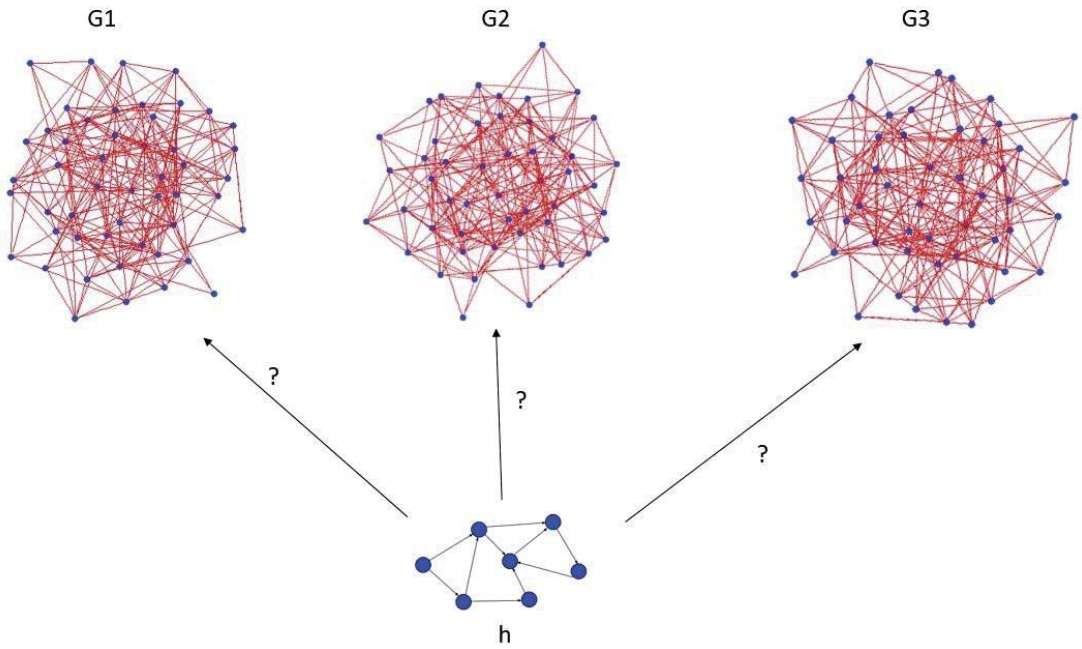


Figura 4.1.3 Grafo de palabras h y grafos de autores conocidos G1, G2 y G3

4.1.6. Stopwords

Las palabras que son demasiado frecuentes en los documentos de una determinada colección no aportan información relevante. De hecho, se considera que una palabra que aparece en al menos el 80% de los documentos de una determinada colección carece de utilidad, pues generalmente se trata de preposiciones, conjunciones, artículos, etc. Estas palabras se consideran vacías y normalmente se eliminan para evitar que puedan ser consideradas como potenciales [12].

Existe para cada idioma un conjunto de palabras vacías, comunes a todos los dominios, fácilmente identificable: artículos, preposiciones, conjunciones, etc., aunque también puede haber verbos, adverbios y adjetivos. Una herramienta de pre-procesamiento consiste en eliminar las stopwords o palabras vacías del texto. Sin embargo para esta etapa del proyecto se utilizaran y posiblemente compararan sin estas.

4.1.7. Técnicas utilizadas

4.1.7.1. PosTagging

El propósito del POS Tagging es asignar la categoría léxica correcta (Ej: sustantivo, verbo o articulo) a cada palabra del texto. La principal dificultad con POS Tagging es que la asignación de tipo a una palabra es casi siempre una tarea ambigua ya que la categoría léxica de una palabra usualmente depende del contexto en que es usada. Para manejar la ambigüedad, POS Taggers usualmente considera secuencias de n palabras y así derivar el contexto en el que la palabra es utilizada. Este enfoque evita el uso de conocimiento externo para producir un grafo con etiquetado [17].

La relación gramatical es usada para encontrar la relevancia de las etiquetas usando un modelo de grafos. El texto primero es etiquetado utilizando información de *part-of-speech* produciendo sustantivo, verbo, adjetivo o vértices de adverbios. El grafo generado toma lugar un pedazo de texto como input y genera un grafo como output. La estructura del grafo es definida utilizando un vector de relevancia el cual es creado usando coincidencias exactas de métricas (e), subcadenas (s), cadenas distintas (d), (o) no coincidencias, sinónimos (y), hiponimo (h) y dominio de palabras raras (r).

dorg(e,s,d,y,h,r)

Una estructura lingüística del párrafo de un texto de un documento es basada en *parse tree* para cada oración de un párrafo. Un *parse thicket* es un grafo que contiene parse tree por cada oración, como también arcos adicionales por cada relación de *parse tree nodes* para palabras como correferencias, relaciones taxonómicas como sub entidades, casos parciales y predicados de temas, relaciones de estructuras retoricas y actos de discurso.

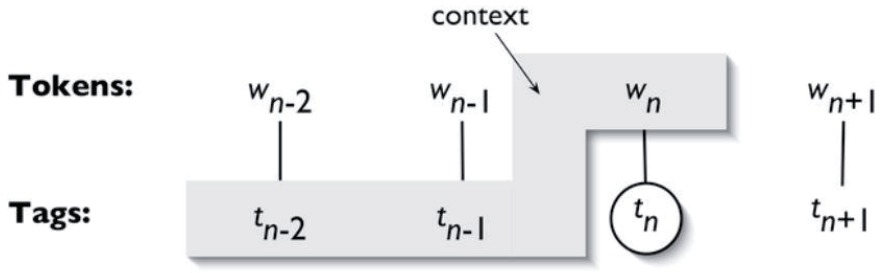


Figura 4.1.4 N-grama de Pos Tagging

Existen 2 POS Taggers principales:

- **DefaultTagger**: En este caso se utiliza un POS tag único para todas las palabras, por ejemplo asignar un POS tag pronombre a todas las palabras de una oración.

- **N-gramTagger:** N-gramas sobre cualquier secuencia dada pueden ser informalmente definidas como superposiciones de cada una de las longitudes de N. A continuación se muestran diferentes n-gramas para valores de N de la oración "Mi nombre es Sebastian Canepa".

    - N = 1 (1-grama or Unigramas): Mi, nombre, es, Sebastian, Canepa
    - N = 2 (2-grama or Bigramas): Mi nombre, nombre es, es Sebastian, Sebastian Canepa,
    - N = 3 (3-grama or Trigramas): Mi nombre es, nombre es Sebastian, es Sebastian Canepa
    - N = 4 (4-grama): Mi nombre es Sebastian, nombre es Sebastian Canepa
    - N = 5 (5-grama): Mi nombre es Sebastian Canepa

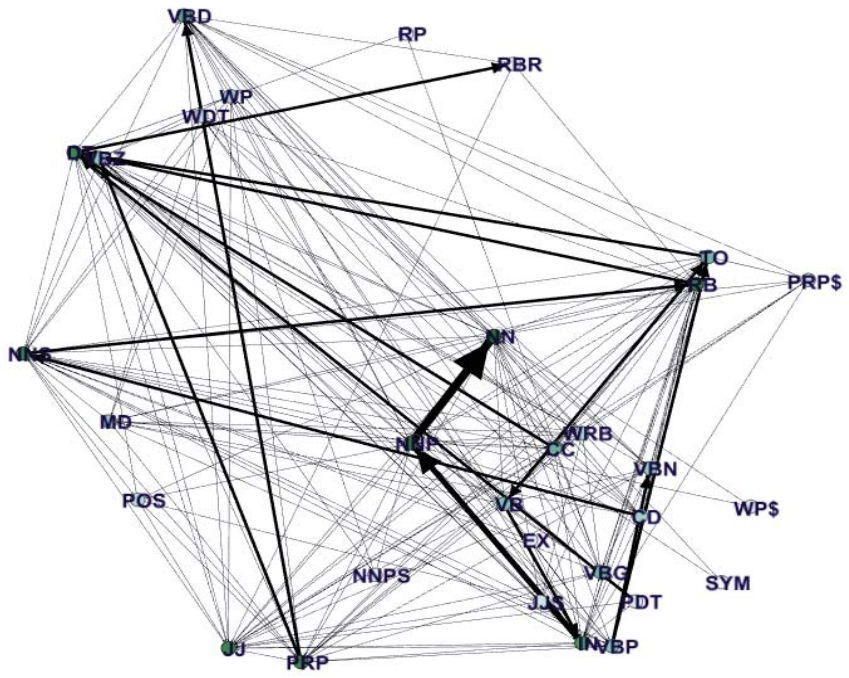


Figura 4.1.5 Grafo de palabras utilizando Pos-Tagg

4.1.7.2. Arboles de Decisión

Los árboles de decisión (DT) son un método de aprendizaje supervisado no paramétrico utilizado para la clasificación y la regresión. El objetivo es crear un modelo que prediga el valor de una variable objetivo mediante el aprendizaje de reglas simples de decisión inferidas a partir de las características de los datos.

CART (Árboles de Clasificación y Regresión), el algoritmo utilizado por scikit-learn es muy similar al algoritmo C4.5, pero difiere en que admite variables de objetivo numéricas (regresión) y no calcula conjuntos de reglas. CART construye árboles binarios usando la característica y el umbral que producen la mayor ganancia de información en cada nodo.

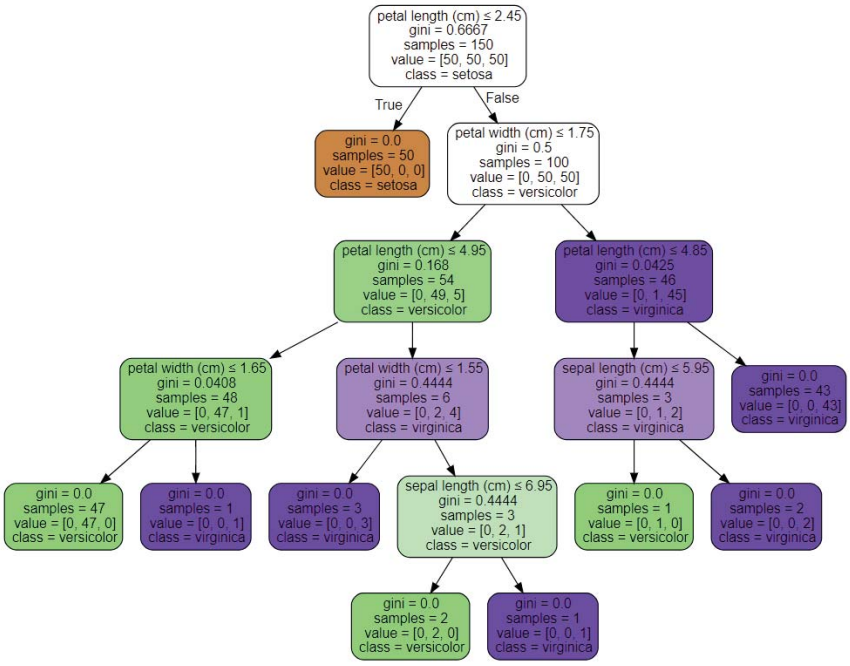


Figura 4.1.6 Algoritmo de clasificación de árbol CART

4.1.7.3. Máquinas de Soporte Vectorial

Máquinas de Soporte Vectorial han mostrado un buen desempeño en general en una gran variedad de problemas de clasificación, más recientemente en clasificación de textos.

En términos geométricos, el problema que resuelve las SVM (Support Vector Machine) es identificar una frontera de decisión linear entre dos clases, a través de una línea que los separe, maximizando el espacio del hiperplano. Sin embargo, las SVM incluyen una función llamada kernel, la cual permite realizar separaciones no lineales de los datos, proyectando la información a un espacio de características de mayor dimensión. Esto se logra cambiando la representación de la función, mapeando el espacio de entradas D a un nuevo espacio de características F = {φ(d)| d □ D}

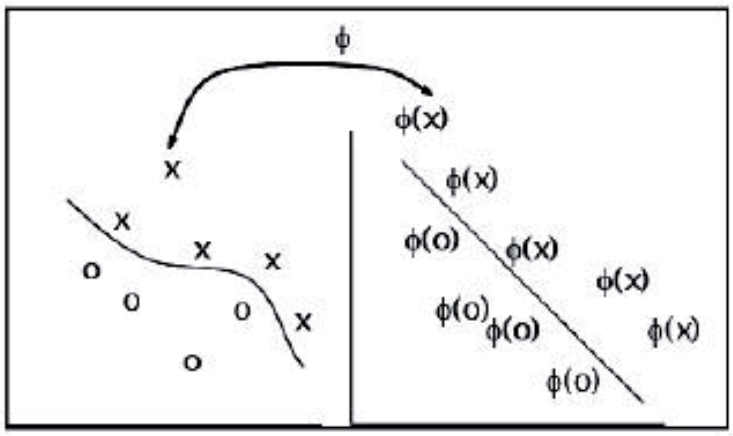


Figura 4.1.7 Mapeo de espacio en SVM

La función real φ no necesita ser conocida, es suficiente tener una función kernel k, la cual hace posible realizar el mapeo de la información de entrada al espacio de características de forma implícita y entrenar a la máquina lineal en dicho espacio.

4.1.7.4. K-NearestNeighbors

El método de los k vecinos más cercanos es un método de clasificación supervisada que sirve para estimar la función de densidad $F(x/Cj)$ de las predictoras **x** por cada clase C_j

Este es un método de clasificación no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad a posteriori de que un elemento **x** pertenezca a la clase C_j a partir de la información proporcionada por el conjunto de prototipos.

En el reconocimiento de patrones, el algoritmo Knn es usado como método de clasificación de objetos (elementos) basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos. Knn es un tipo de aprendizaje vago (lazy learning), donde la función se aproxima solo localmente y todo el cómputo es diferido a la clasificación.

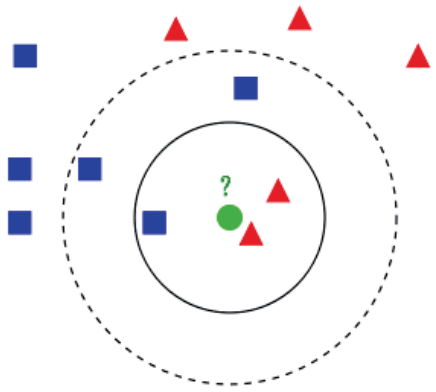


Figura 4.1.8 Algoritmo de clasificación knn

En la figura 7, se desea clasificar es el círculo verde. Para k = 3 este es clasificado con la clase triángulo, ya que hay solo un cuadrado y 2 triángulos, dentro del círculo que los contiene. Si k = 5 este es clasificado con la clase cuadrado, ya que hay 2 triángulos y 3 cuadrados, dentro del círculo externo.

4.1.8. Medidas de evaluación

4.1.8.1. Intermediación

Es una medida que cuantifica la frecuencia o el número de veces que un nodo actúa como un puente a lo largo del camino más corto entre otros dos nodos.

La medida fue introducida por Linton Freeman en 1977, como forma de cuantificar el control de un humano en la comunicación existente con otros humanos en una red social. La idea intuitiva es que si se eligen dos nodos al azar, y luego también al azar uno de los eventuales posibles caminos más cortos entre ellos, entonces los nodos con mayor intermediación serán aquellos que aparezcan con mayor probabilidad dentro de este camino [13].

El grado de Intermediación de un nodo *v* se define como:

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

Donde σ_{st} es el número de caminos más cortos desde el nodo *s* al nodo *t* y $\sigma_{st}(v)$ es el número de esos caminos que pasan por *v*.

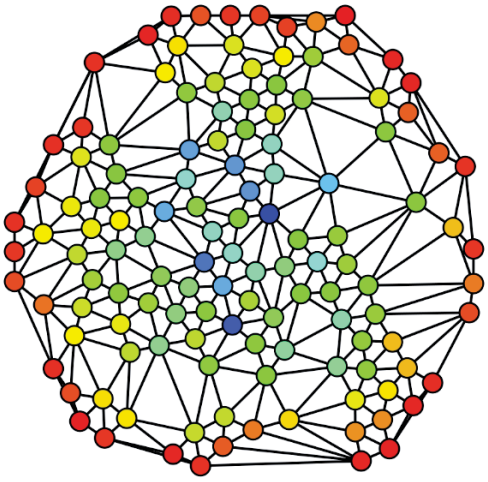


Figura 4.1.9 Intermediación de nodos en un grafo, rojo (valor 0) y azul (valor máximo).

4.1.8.2. Cercanía

La medida de cercanía, definida por el matemático Murray Beauchamp en 1965 [14] y luego popularizada por Freeman en 1979, es la más conocida y utilizada de las medidas radiales de longitud. Se basa en calcular la suma o bien el promedio de las distancias más cortas desde un nodo hacia todos los demás.

La cercanía de un nodo *i* se define como:

$$C_{CLO}(i) = e_i^T S\mathbf{1} = \sum_{j=1}^{n} (S)_{ij}$$

Donde *S* es la matriz de distancia de la red, es decir, aquella matriz cuyos elementos (i, j) corresponden a la distancia más corta desde el nodo i hasta el nodo j. Mientras menor sea el valor anterior, se puede decir que el nodo está más «cercano» al centro de la red. En tal caso definido así corresponde en realidad a una medida de lejanía. Por la misma razón, a veces la cercanía se define más bien como el valor recíproco de lo anterior, no cambiando por ello la idea del concepto.

$$C_{CLO}(i) = \frac{1}{e_i^T S\mathbf{1}} = \frac{1}{\sum_{j=1}^{n} (S)_{ij}}$$

	a	b	c	d	e	f	g		C
a	0	1	2	3	2	3	4		15
b	1	0	1	2	1	2	3		10
c	2	1	0	1	1	2	3		10
d	3	2	1	0	2	3	4		15
e	2	1	1	2	0	1	2		9
f	3	2	2	3	1	0	1		12
g	4	3	3	4	2	1	0		17

Figura 4.1.10 Matriz de distancia de una red

4.1.8.3. Centralidad de grado

La centralidad de grado, es la primera y más simple de las medidas de centralidad [15]. Corresponde al número de enlaces que posee un nodo con los demás. Formalmente, dado un grafo G:= (V, E) donde V es su conjunto de vértices o nodos y E su conjunto de aristas, entonces para cada nodo *v* que pertenece a V su centralidad de grado se define como:

$$C_{DEG}(v) = \text{grado}(v)$$

Si se tiene la matriz de adyacencia del grafo, donde cada posición a_{ij} asume el valor 1, si existe la arista (i, j) y el valor 0, si no existe, entonces la centralidad de grado de cada nodo j se puede definir como:

$$C_{DEG}(j) = \sum_i a_{ij}$$

Las interpretaciones de esta medida pueden ser múltiples. En una red social, puede ser el número de amistades o conexiones que posee cada persona, en cuyo caso cuantifica la conectividad o popularidad en la red. En una red de infección puede medir el grado de riesgo de ser contagiado o el índice de exposición. En la propagación de un rumor, puede medir la probabilidad de obtener la información a través de un rumor.

4.1.8.4. Centralidad de vector propio

La centralidad de vector propio mide la influencia de un nodo en la red. Corresponde al principal vector propio de la matriz de adyacencia. Se asignan puntuaciones relativas a todos los nodos de la red basándose en el concepto de que las conexiones con nodos de alta puntuación contribuyen más a la puntuación del nodo en cuestión que las conexiones iguales a los nodos de baja puntuación.

Dado un grafo G:= (V, E) con $|V|$ vértices y $A = (a_{v,t})$ como la matriz de adyacencia, si el vértice *v* está unido a *t* el valor será 1, de lo contrario 0. Se define la centralidad del vértice *v* como:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} a_{v,t} x_t$$

4.1.8.5. PageRank

Definido en el año 1998 por Sergei Brin y Lawrence Page, ambos fundadores del motor de búsqueda Google, corresponde a un algoritmo que describe una manera de calcular el ranking o la clasificación de páginas web que existen en el internet. Este se calcula como:

$$\mathrm{PR}(A) = (1-d) + d \sum_{i=1}^{n} \frac{\mathrm{PR}(i)}{C(i)}$$

Donde d es un factor de amortiguación que tiene un valor entre 0 y 1, PR (i) son los PageRank de las páginas que enlazan con PR(A) y C (i) es el número total de enlaces salientes de la página i (sean o no hacia A).

4.1.8.6. Coeficiente medio de Clustering

El coeficiente de agrupamiento (mencionado en la literatura también como clustering coefficient) de un vértice en un grafo cuantifica qué tanto está de agrupado (o interconectado) con sus vecinos. El coeficiente de agrupamiento de la red se calcula mediante Watts y Strogatz como la media de los coeficientes de agrupamiento de todos los vértices de la red:

$$\bar{C} = \frac{1}{n}\sum_{i=1}^{n} C_i$$

Para **grafos dirigidos** se tiene:

$$C_i = \frac{|\{e_{jk}\}|}{k_i(k_i-1)} : v_j, v_k \in N_i, e_{jk} \in E.$$

Para **grafos no dirigidos** se tiene:

$$C_i = \frac{2|\{e_{jk}\}|}{k_i(k_i-1)} : v_j, v_k \in N_i, e_{jk} \in E.$$

5. Experimentación

A continuación, se va a especificar todo lo que fue necesario para el desarrollo de la experimentación.

5.1. Set de datos

Los sets de textos a utilizar se componen de una colección de textos de opinión llamada Guardian Corpus, la cual cuenta con 844 textos de 13 autores distintos, cada autor posee 4 tópicos, Politics, Society, UK, World. Para la data de entrenamiento se estableció un 75% del total resultante de la clasificación estratificada de k-fold donde k=4 y un 25% para pruebas.

Tabla 5.1.1 Set de textos 1 para Autores de opinión

Autor	Cantidad de Textos	Textos entrenamiento	Textos prueba
Catherine Bennett	41	31	10
Polly Toynbee	70	53	17
Zoe Williams	38	28	10
George Mombiot	53	40	13
Jonathan Freeland	126	95	31
Peterson Preston	91	68	23
Hugo Young	54	41	13
Martin Kettle	46	35	11
Mary Riddell	67	50	17
Nick Cohen	48	36	12
Roy Hattersley	44	33	11
Simon Hoggart	116	87	29
Will Hutton	50	38	12

En el primer experimento se utilizaron la totalidad de los textos sin filtrado, en un segundo experimento se utilizó el total de los textos, pero las métricas se calcularon filtrando las aristas con peso mayor que 1 en los grafos creados. Como tercer experimento se utilizaron la misma cantidad de textos por autor, es decir 38 textos de cada autor, 494 en total, la data de entrenamiento y prueba fue dividida utilizando el mismo método anterior con k-fold, donde k=4.

El objetivo de esto es obtener resultados con la data mezclada pero estratificada, es decir manteniendo una misma proporción de clases tanto en entrenamiento como en pruebas para obtener conclusiones más acertadas. Como cuarto experimento, se realizó el tercer experimento, pero con aristas filtradas.

Tabla 5.1.2 Set de textos 2 con la misma cantidad de clases

Autor	Cantidad de Textos	Textos entrenamiento	Textos prueba
Catherine Bennett	38	29	9
Polly Toynbee	38	29	9
Zoe Williams	38	29	9
George Mombiot	38	29	9
Jonathan Freeland	38	29	9
Peterson Preston	38	29	9
Hugo Young	38	29	9
Martin Kettle	38	29	9
Mary Riddell	38	29	9
Nick Cohen	38	29	9
Roy Hattersley	38	29	9
Simon Hoggart	38	29	9
Will Hutton	38	29	9

Tabla 5.1.3 Set de datos C0 Autores con más textos

Autor	Cantidad de Textos	Textos entrenamiento	Textos prueba
Jonathan Freeland	126	94	32
Peterson Preston	91	68	23
Simon Hoggart	116	87	29

Tabla 5.1.4 Set de datos C1 Autores con menos textos

Autor	Cantidad de Textos	Textos entrenamiento	Textos prueba
Catherine Bennett	41	31	10
Zoe Williams	38	28	10
Roy Hattersley	44	33	11

A modo de experimentación, se crearon dos sets de datos C0 y C1, el primero con los tres autores con mayor cantidad de textos y el segundo con los tres autores con menor cantidad de textos. Se espera con estos sets de datos ver mejores resultados ya que se están clasificando 3 clases en vez de 13. Se hiso algo similar en el experimento 5.4.3 clasificando por tópico de autor.

5.2. Experimentos

La entrada de datos se definió como archivos de texto plano, en la cual cada autor puede tener varios textos. Los experimentos se realizaron utilizando la librería NLTK de Python para el pre procesado de textos y la creación de los POS Tagg mediante la sentencia PunkSentenceTokenizer, el cual entrega como output el texto con las palabras y su tag correspondiente.

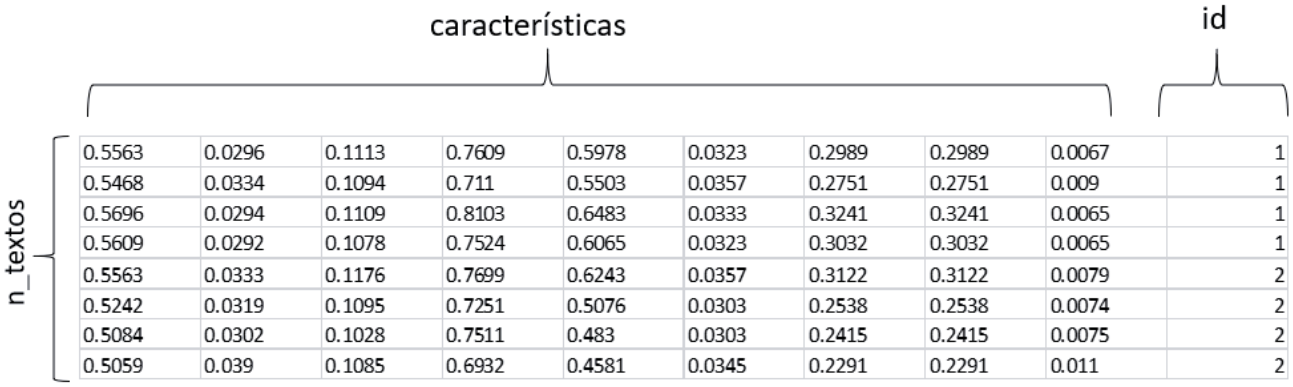
```
    train_text =gutenberg.raw("catherinebennett_World_001.txt")
    sample_text = gutenberg.raw(fileid)
    custom_sent_tokenizer = PunktSentenceTokenizer(train_text)
    tokenized = custom_sent_tokenizer.tokenize(sample_text)
    with open("C:/Users/Sebastian/Desktop/postagc1/%s" %fileid,
'w') as fichero:
    for i in tokenized:
          words = nltk.word_tokenize(i)
          palabra = [x for x in words if not re.fullmatch('[' +
          string.punctuation + ']+', x)]
          tagged = nltk.pos_tag(palabra)
                for number,w in enumerate(tagged):
                fichero.write("{}\n".format(w))
```

Posteriormente los archivos de Postag fueron procesados para crear los nodos y aristas, para cada arista además se calculó un valor peso según su ocurrencia en el texto [16]. Finalmente se calculan las 9 métricas de la librería de networkX y se utilizan como input en scikit-learn para realizar la clasificación.

```
cercania=nx.closeness_centrality(G,normalized=True)
bet_cen = nx.betweenness_centrality(G,normalized=True)
ed_cen = nx.edge_betweenness_centrality(G,normalized=True)
eig_cen = nx.eigenvector_centrality(G,max_iter=100)
centralidad_grado=nx.degree_centrality(G)
indegree = nx.in_degree_centrality(G)
out = nx.out_degree_centrality(G)
pr = nx.pagerank(G,alpha=0.85)
cluster=nx.average_clustering(G_ud)
```

5.2.1. Vector de Características

La siguiente figura 5.2.1 representa los vectores que se utilizan como input para los clasificadores, cada fila es un documento perteneciente a un autor. Se calcularon 9 métricas para cada texto y se le asigno la id de autor correspondiente. Los vectores fueron guardados en un archivo .csv que posteriormente se utilizara para entrenamiento y prueba.



0.5563	0.0296	0.1113	0.7609	0.5978	0.0323	0.2989	0.2989	0.0067	1
0.5468	0.0334	0.1094	0.711	0.5503	0.0357	0.2751	0.2751	0.009	1
0.5696	0.0294	0.1109	0.8103	0.6483	0.0333	0.3241	0.3241	0.0065	1
0.5609	0.0292	0.1078	0.7524	0.6065	0.0323	0.3032	0.3032	0.0065	1
0.5563	0.0333	0.1176	0.7699	0.6243	0.0357	0.3122	0.3122	0.0079	2
0.5242	0.0319	0.1095	0.7251	0.5076	0.0303	0.2538	0.2538	0.0074	2
0.5084	0.0302	0.1028	0.7511	0.483	0.0303	0.2415	0.2415	0.0075	2
0.5059	0.039	0.1085	0.6932	0.4581	0.0345	0.2291	0.2291	0.011	2

Figura 5.2.1 Vector Características para input de clasificadores

Luego el archivo que contiene los vectores de input se separa en características (X) e objetivo (Y) y se crea un numpy array con dichos datos, de tal manera que se tiene lo siguiente: X_train, X_test, y_train, y_test. A continuación, se muestra un ejemplo de cómo quedan los numpy array.

X =

0.5563	0.0296	0.1113	0.7609	0.5978	0.0323	0.2989	0.2989	0.0067
0.5468	0.0334	0.1094	0.711	0.5503	0.0357	0.2751	0.2751	0.009
0.5696	0.0294	0.1109	0.8103	0.6483	0.0333	0.3241	0.3241	0.0065

```
array([[  0.,   0.,   5.,  13.,   9.,   1.,   0.,   0.],
       [  0.,   0.,  13.,  15.,  10.,  15.,   5.,   0.],
       [  0.,   3.,  15.,   2.,   0.,  11.,   8.,   0.],
       [  0.,   4.,  12.,   0.,   0.,   8.,   8.,   0.],
       [  0.,   5.,   8.,   0.,   0.,   9.,   8.,   0.],
       [  0.,   4.,  11.,   0.,   1.,  12.,   7.,   0.],
       [  0.,   2.,  14.,   5.,  10.,  12.,   0.,   0.],
       [  0.,   0.,   6.,  13.,  10.,   0.,   0.,   0.]])
```

Y =

1
1
1

```
array([0, 1, 2, ..., 8, 9, 8])
```

5.2.2. Stratified kfold CV

Con el vector de características definido anteriormente, se correrán los algoritmos de clasificación dividiendo la data en entrenamiento y pruebas mediante la técnica de clasificación estratificada. Como se explicó anteriormente la idea es realizar el experimento con la data mezclada, pero manteniendo la proporción de las clases en los sets de entrenamiento y prueba. La siguiente figura 5.2.2 explica cómo funciona esta técnica.

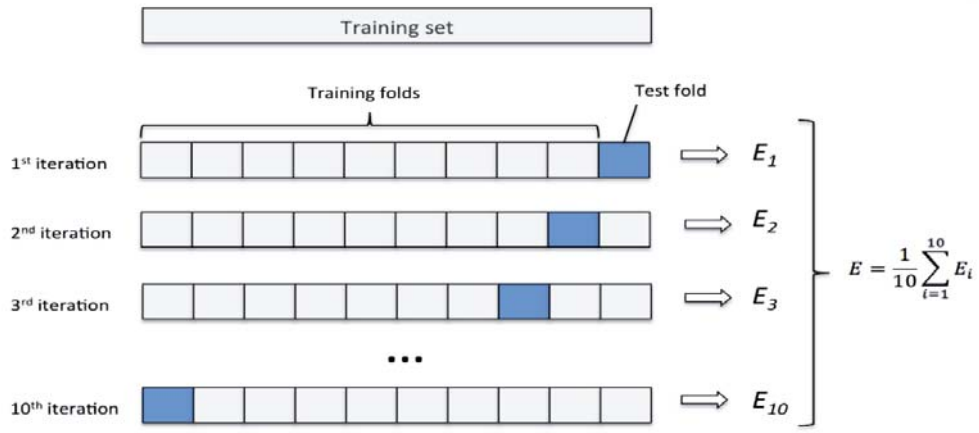


Figura 5.2.2 Ejemplo Clasificación estratificada con k=10

5.2.3. GridSearchCV

Dentro de los clasificadores y previo a aplicar la clasificación estratificada se usa la función GridSearchCV, esta es una búsqueda exhaustiva sobre valores de parámetros especificados para un estimador, es decir, este método nos da la mejor combinación de parámetros de los que probamos para el estimador y así poder usar aquellos para la clasificación. Por ejemplo, si se quiere probar 3 kernels distintos y un parámetro "p" de rango [1,20] GridSearchCV entrega la mejor combinación que optimice el estimador.

A continuación, se detallan las características de la maquina donde fueron procesados los datos y ejecutados los algoritmos.

- Procesador: Intel Core i7 4720HQ CPU @2.60Ghz
- Memoria Ram: 16GB DDR3 1600Mhz
- Disco Duro: 128GB SSD
- Sistema Operativo: Windows 10
- Versión Python: 3.6.0

5.3. Implementación de Grafos

5.3.1. Pre procesado

- En esta parte del proyecto las letras no fueron cambiadas a minúscula ya que al realizar POS Tagging se puede confundir un sustantivo singular con un sustantivo propio
- Se eliminan los caracteres que puedan generar error y que no tengan que ver con la palabra.
- Se eliminan signos de puntuación, comas y otros caracteres.
- Se mantuvieron las Stopwords en los textos de esta instancia.
- Se agrega el POS Tagg por palabra.
- Se exporta el archivo .csv

5.3.2. Pruebas

- Se probará el clasificador con las siguientes configuraciones.
- Set de textos completo
- Set de textos filtrando aristas
- Misma cantidad de textos por autor/filtrado
- Validación cruzada con 4-kfold y 10-kfold estratificada
- Se calculo el vector característico con valores TF-IDF para comparar

5.3.3. Parámetros y métricas

- Métricas de centralidad están normalizadas
- Las métricas fueron calculadas utilizando la opción de grafos dirigidos. Para el PageRank, se utilizó un alpha de 0.85. Vector propio se calcularon 100 iteraciones como opción, la mayoría de los parámetros fueron por default.
- Para los experimentos 2 y 4 se consideró filtrar por aristas con peso mayor que 1 y mayor que 20

5.4. Resultados

Para presentar los resultados se realizó una tabulación de estos mostrando el porcentaje de acierto por clasificador en los 4 sub set del k-fold. También se realizó una clasificación utilizando todos los textos, pero por tópico de autor.

5.4.1. Resultados experimento 1 y 2

Tabla 5.4.1 Resultados experimento 1 sin filtrar aristas

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio
KNeighborsClassifier	25,9%	24,7%	28,3%	21,3%	**25,05%**
RandomForestClassifier	32,8%	30,8%	28,8%	24,7%	**29,3%**
DecisionTreeClassifier	26,8%	21,9%	22,5%	19,4%	**22,7%**
GaussianNB	27,3%	25,7%	30,2%	16,5%	**24,9%**
OneVsRestClassifier	23,1%	16,3%	23,0%	16,0%	**19,6%**
LinearSVC	22,6%	23,3%	25%	24%	**23,8%**

Tabla 5.4.2 Resultados experimento 2 filtrando aristas con peso >1

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio
KNeighborsClassifier	26,8%	27,1%	25%	17,9%	**24,2%**
RandomForestClassifier	30%	31,7%	28,3%	20,8%	**27,7%**
DecisionTreeClassifier	21,7%	24,7%	19,2%	19,4%	**21,2%**
GaussianNB	25%	28,5%	27,4%	16,9%	**24,7%**
OneVsRestClassifier	20,8%	21,9%	23,0%	18,4%	**21%**
LinearSVC	24%	26,6%	25,4%	25,7%	**25,4%**

El siguiente grafico muestra el porcentaje de acierto del experimento 1 y 2 haciendo un filtrado del peso de las aristas.

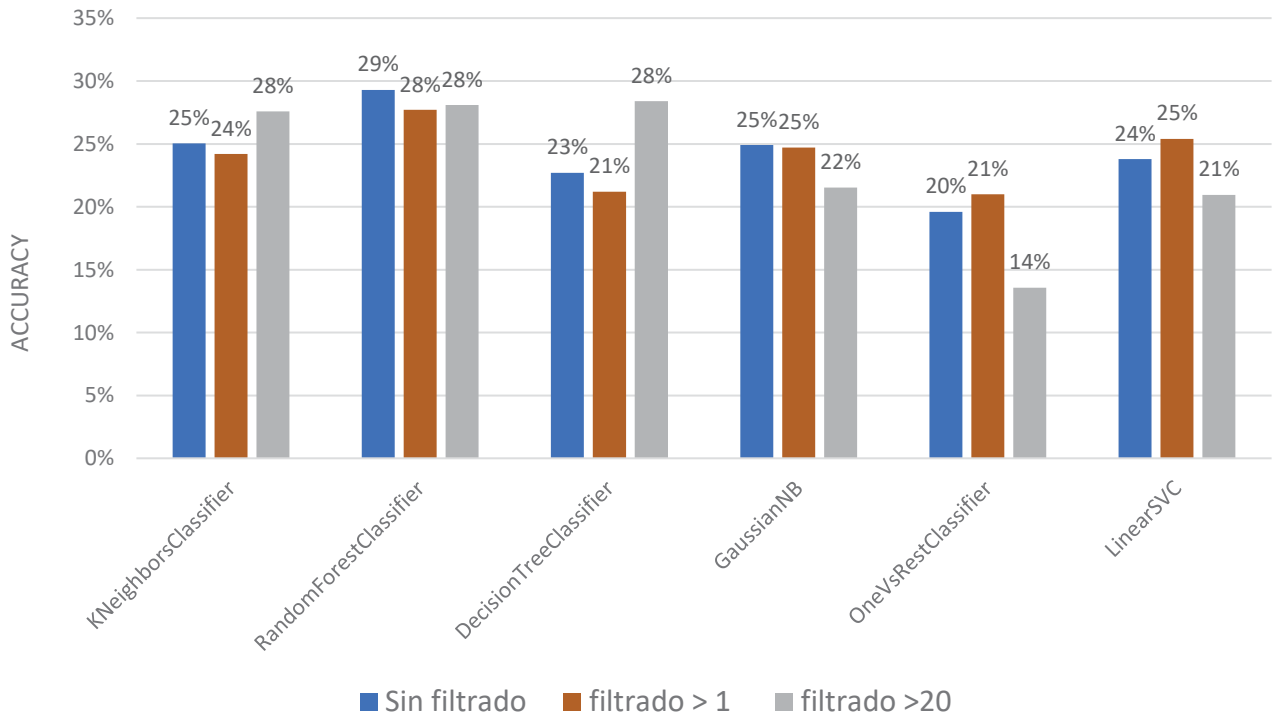


Figura 5.4.1 Grafico de aciertos por clasificador experimento 1 y 2

Como se puede apreciar, los resultados entre todos los clasificadores fueron bastante similares, destacando por poco el RandomForestClassifier en los 3 experimentos. Filtrar aristas no mostro resultados relevantes ya que en algunos clasificadores mejora la precisión y en otros disminuye. Se cree que los resultados fueron bajo lo esperado debido a la complejidad de clasificar 13 clases distintas.

5.4.2. Resultados experimentos 3 y 4

Tabla 5.4.3 Resultados experimento 3 sin filtrar

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio
KNeighborsClassifier	13,8%	16,9%	17,0%	18,8%	**16,6%**
RandomForestClassifier	21,5%	20%	22,2%	11,9%	**18,9%**
DecisionTreeClassifier	22,3%	16,1%	24,7%	11,1%	**18,5%**
GaussianNB	20%	17,6%	17%	13,6%	**17,1%**
OneVsRestClassifier	8%	11,5%	14,5%	9%	**10,9%**
LinearSVC	17,6%	16,9%	13,6%	10,2%	**14,6%**

Tabla 5.4.4 Resultado experimento 4 con filtrado peso >1

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio
KNeighborsClassifier	10%	13,8%	12,8%	11%	**11,9%**
RandomForestClassifier	21,5%	21,5%	16,2%	17,0%	**19,1%**
DecisionTreeClassifier	15,3%	13,8%	14,5%	11,9%	**13,9%**
GaussianNB	18,4%	22,3%	15,3%	17,9%	**18,5%**
OneVsRestClassifier	15,3%	8%	9%	13,6%	**11,7%**
LinearSVC	18,4%	15,3%	11%	11%	**14%**

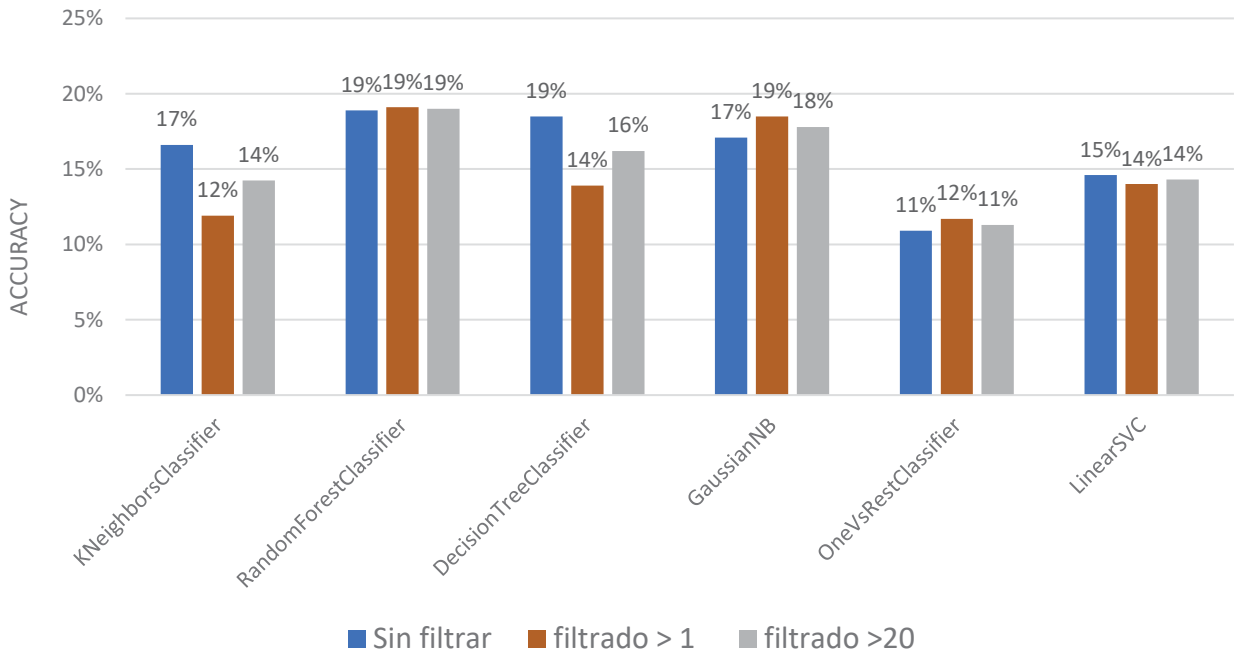


Figura 5.4.2 Resultado de aciertos por clasificador experimentos 3 y 4

Los resultados del experimento 3 y 4 mostraron peores resultados que los anteriores, posiblemente debido a la menor cantidad de textos por autor, en este caso eran 38 textos por autor de los cuales 29 eran de entrenamiento y 9 de pruebas. El filtrado tampoco fue relevante en estos resultados ya que son irregulares los porcentajes.

5.4.3. Resultados de experimento por tópico de autor

Tabla 5.4.5 Resultado de experimentos por tópicos

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio
KNeighborsClassifier	41,3%	47,1%	51,9%	48,3%	**47,1%**
RandomForestClassifier	37,5%	39%	50%	43%	**42,4%**
DecisionTreeClassifier	34,2%	31,6%	41,4%	35,8%	**35,7%**
GaussianNB	38%	46,2%	59,5%	44%	**46,9%**
OneVsRestClassifier	39,9%	45,7%	57,1%	49,7%	**48,1%**
LinearSVC	38,9%	46,6%	53,3%	51,1%	**47,5%**

Como se puede observar en la tabla 5.4.5, los resultados fueron bastante mejores que el resto de los experimentos, esto posiblemente a que el número de clases que se utilizaron para clasificar era mucho menor, 4 en este caso ya que hay 4 tópicos distintos. El mejor resultado se obtuvo con el clasificador OneVsRestClassifier, esta es una estrategia popular cuando hay que clasificar múltiples clases.

También conocida como uno contra todos, esta estrategia consiste en ajustar un clasificador por clase. Para cada clasificador, la clase se ajusta a todas las otras clases. Además de su eficiencia computacional (solo se necesitan clasificadores n_classes), una de las ventajas de este enfoque es su interpretabilidad. Como cada clase está representada por uno y un clasificador solamente, es posible obtener conocimiento sobre la clase inspeccionando su clasificador correspondiente [18].

5.4.4. Resultado experimento Set de datos C0

Tabla 5.4.6 Resultado de experimentos Set de datos C0

Clasificador	1-fold	2-fold	3-fold	4fold	Promedio	DESV
KNeighborsClassifier	60%	62%	59%	52%	**58%**	4%
RandomForestClassifier	70%	67%	66%	67%	**68%**	2%
DecisionTreeClassifier	64%	58%	65%	62%	**62%**	3%
GaussianNB	71%	64%	73%	72%	**70%**	4%
OneVsRestClassifier	50%	49%	55%	52%	**52%**	3%
LinearSVC	58%	58%	61%	61%	**60%**	2%

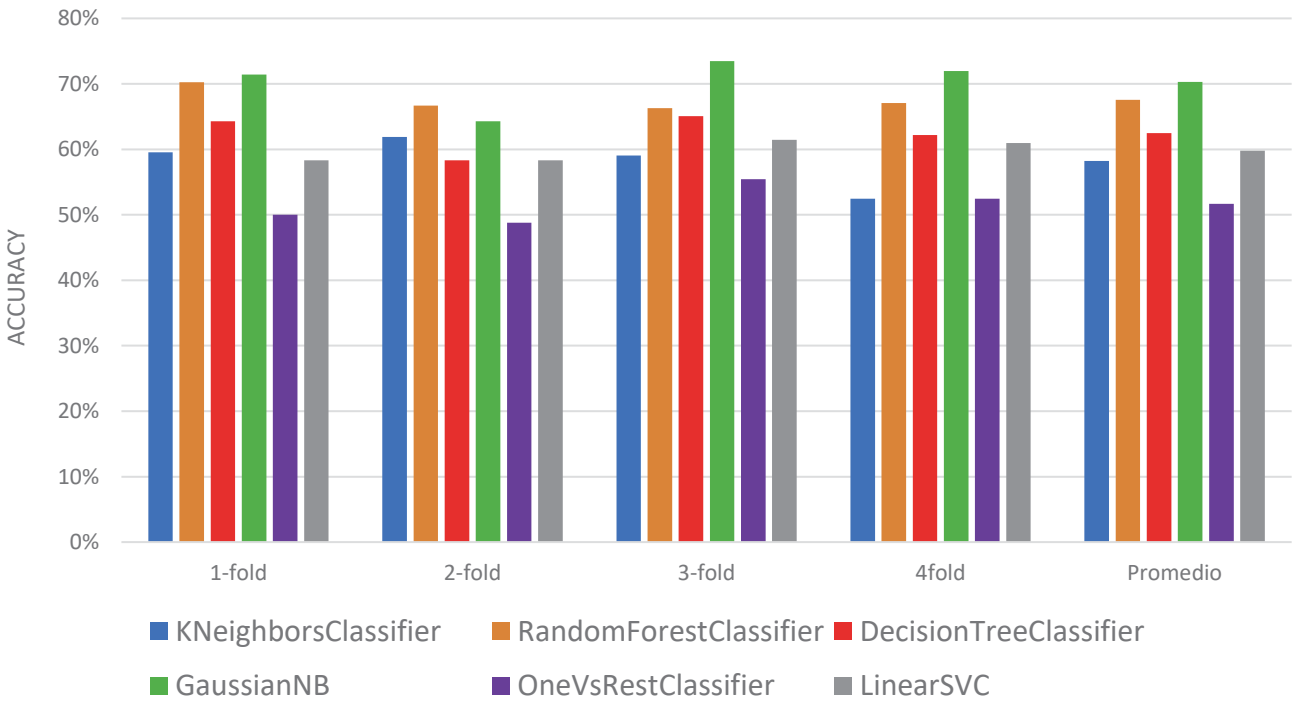


Figura 5.4.3 Resultados Set C0, Porcentaje de acierto Promedio

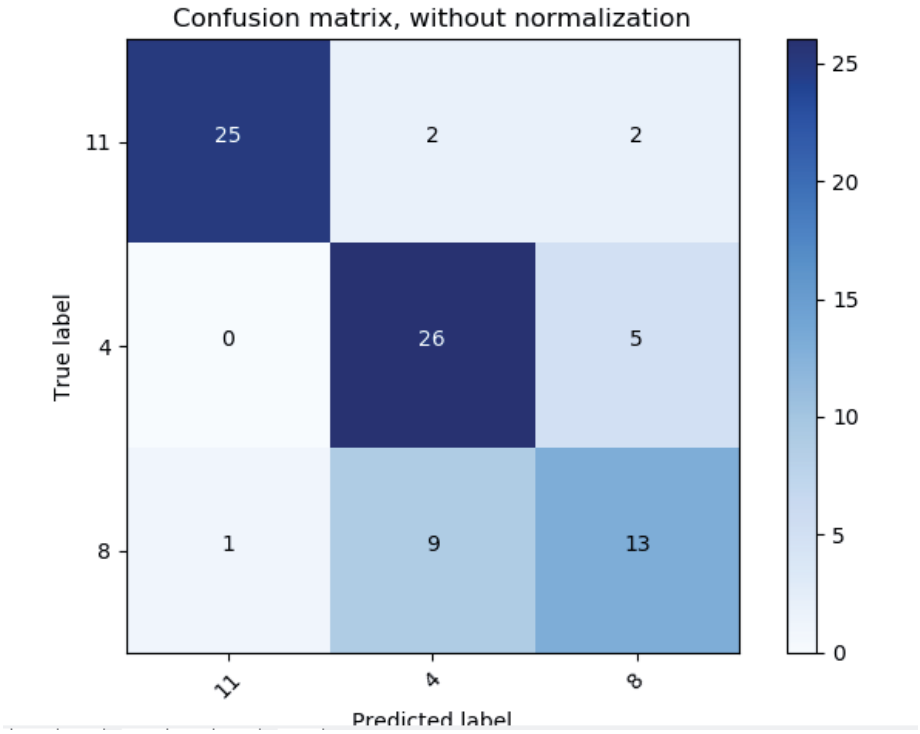


Figura 5.4.4 Matriz de confusión de clasificador GausianoNB en set C0

LABEL	PRECISION	RECALL	F1-SCORE	SUPPORT
11	0.96	0.86	0.91	29
4	0.70	0.84	0.76	31
8	0.65	0.57	0.60	23
PROMEDIO	0.78	0.77	0.77	83

Como se puede apreciar, los resultados del set C0 mejoraron considerablemente en comparación a los primeros experimentos. La complejidad de la clasificación es menor ya que se utilizaron solo 3 clases. El clasificador GaussianNB estuvo a la par con RandomForestClassifier en cuanto al promedio superando el 60% de asertividad.

5.4.5. Resultados experimento Set de datos C1

Tabla 5.4.7 Resultados experimentos Set de datos C1

Clasificador	1-fold	2-fold	3-fold	4fold	Promedio	DESV
KNeighborsClassifier	61%	50%	52%	42%	**51%**	8%
RandomForestClassifier	70%	56%	68%	65%	**65%**	6%
DecisionTreeClassifier	58%	53%	61%	55%	**57%**	4%
GaussianNB	45%	72%	77%	61%	**64%**	14%
OneVsRestClassifier	36%	34%	32%	48%	**38%**	7%
LinearSVC	30%	34%	35%	35%	**34%**	2%

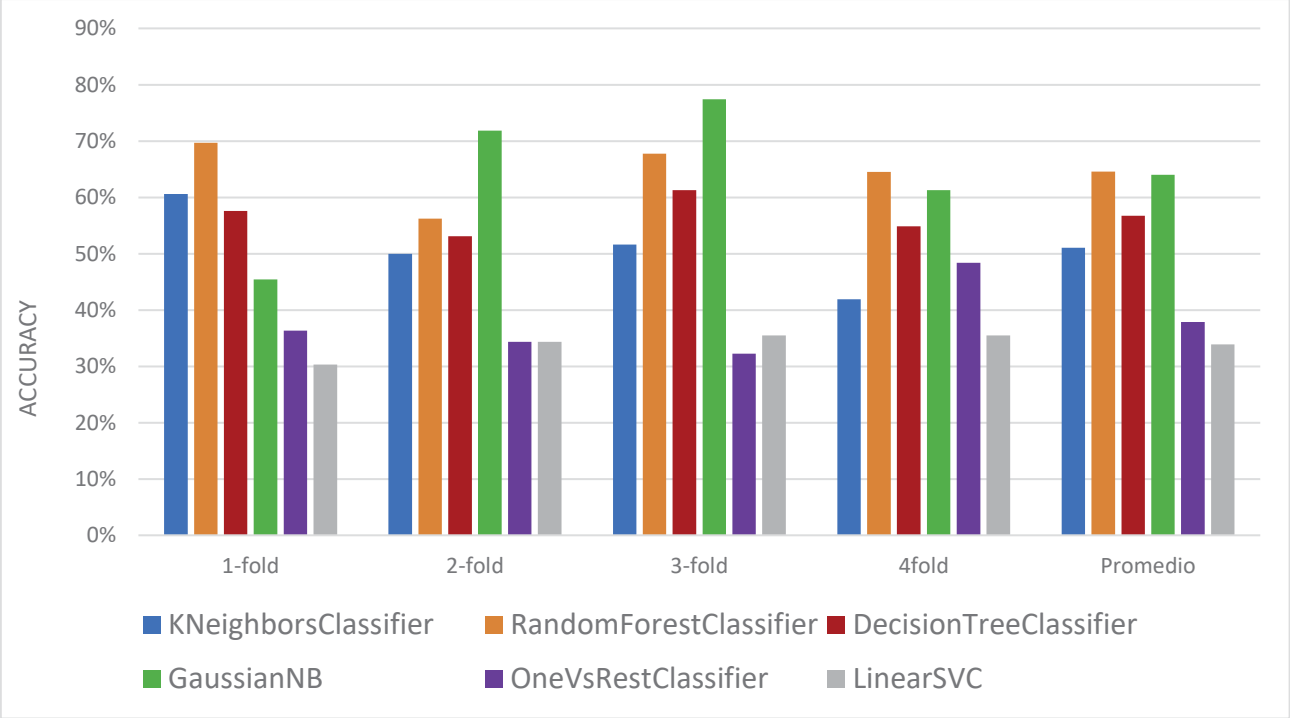


Figura 5.4.5 Resultados Set C1, Porcentaje de acierto Promedio

5.4.6. Resultados utilizando vector TF-IDF

Tabla 5.4.8 Tabla con resultados utilizando vector de TF-IDF

Clasificadores	1-fold	2-fold	3-fold	4-fold	Promedio	DESV
KNeighborsClassifier	20%	23%	21%	23%	**21,93%**	2%
RandomForestClassifier	20%	28%	24%	19%	**22,61%**	4%
DecisionTreeClassifier	19%	16%	19%	15%	**17,29%**	2%
GaussianNB	19%	21%	32%	15%	**21,81%**	7%
OneVsRestClassifier	19%	21%	28%	19%	**21,71%**	4%
LinearSVC	22%	30%	35%	24%	**27,52%**	6%

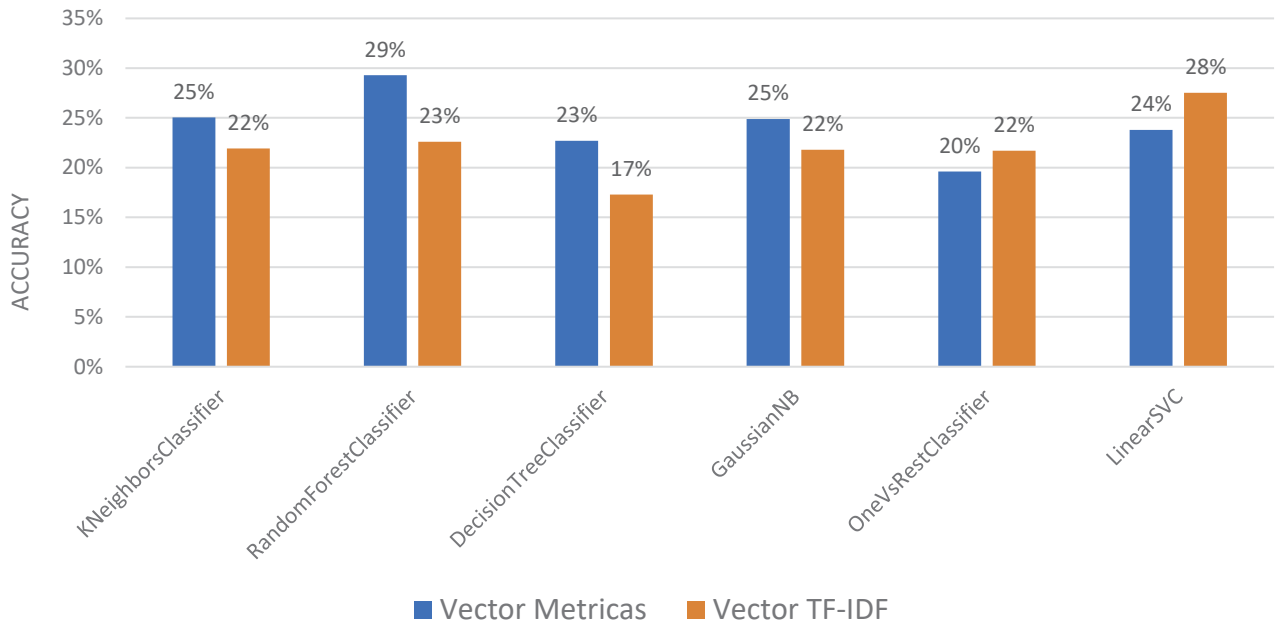


Figura 5.4.6 Resultados vector métricas y vector TF-IDF

Para el cálculo de TF-IDF se eliminaron las stopword y se normalizaron los datos utilizando *TfidfVectorizer* en scikit-learn. Para no sobrecargar el experimento no se calculó el TF-IDF para cada termino de los documentos ya que se crearían vectores muy grandes, sino que, se calculó con las top N características ordenadas por frecuencia de termino de estos.

Como se puede ver en la figura 5.4.6, utilizando los vectores de input con los valores TF-IDF de los términos se lograron peores resultados que el experimento principal. Esto puede deberse a que el cálculo de los valores TF-IDF no sea el indicado para diferenciar entre autores ya que palabras con valores similares que sean de un mismo tópico pueden pertenecer a autores diferentes. En este sentido, utilizar TF-IDF podría dar mejor resultado clasificando tópicos que clasificando autores.

5.4.7. Análisis y comparación de resultados generales

Analizando los resultados obtenidos anteriormente, se esperaba en los experimentos 3 y 4 tener un mejor resultado que en los experimentos 1 y 2, debido a que se utilizó la misma cantidad de textos por autor, esto puede deberse a que en los primeros experimentos se utilizaron todos los textos, es decir había más textos para entrenamiento lo que llevo a mejores resultados.

Se puede observar en el experimento 2, donde se utilizaron métricas calculadas con aristas filtradas, que no se obtuvieron mejoras en comparación con el experimento 1. La idea de filtrar aristas por peso nos daría distintos resultados en el cálculo de métricas que se utilizarían como input en los clasificadores. Utilizando este enfoque, se está de cierta forma eliminando aristas en los grafos que no son relevante ya que ocurren solo una vez.

Uno de los motivos por el cual se cree que hubo pocos aciertos en los resultados, tanto para el set de texto 1 como para el set de texto 2, es debido a la cantidad de clases que se está intentando clasificar. Como se comprobó con el set de texto utilizando 3 autores distintos, los resultados fueron bastante mejor, además, varios clasificadores tienden a ser solo clasificación binaria o clasificación multiclase lo que puede traer variabilidad en los resultados. Otra posible causa de la poca precisión en los aciertos puede ser las métricas calculadas a los grafos, de tal manera que no representen las características de los autores.

Cabe destacar que los resultados obtenidos son mucho mejores en comparación con investigaciones anteriores [2] donde se utilizaron medidas de evaluación similares como pagerank y clasificadores como KNN, Naive Bayes entre otros

6. Trabajos Futuros

Probar con las mismas medidas de evaluación en otros procesadores de textos como WEKA para eliminar diferencias de algoritmos. Mejorar la optimización de parámetros de cada clasificador y buscar los óptimos, para estos es necesario tener un corpus más grande en algunos casos.

Otro tema importante es probar nuevos corpus, por ejemplo, en español y usar dichas etiquetas provistas por Spacy o Freeling las cuales son más específicas en cuanto el significado de la palabra.

7. Conclusión

En la actualidad la determinación de Autoría ha ganado cada vez mayor importancia debido a que soluciona los problemas de propiedad intelectual, como los derechos de autor, la detección de plagio, ver a quién pertenecen textos anónimos, entre otros. Una detección automática y certera puede ayudar a evitar que las situaciones de acoso, mensajes terroristas, plagios escalen a mayores niveles, castigando de manera más rápida a los responsables.

La resolución de la Atribución de Autoría mediante un algoritmo que utilice las Redes de palabras abre un nuevo enfoque a la hora de realizar caracterizaciones de textos; ya que hasta el momento no se ha realizado. Una aproximación para realizar la determinación de Autoría es a través de las características que ofrece la Estilometría, añadiendo las características que ofrecen las Redes de Palabras, como cálculo de caminos más cortos, costo de reconstrucción de una red, ver la conectividad de los vértices, entre otros, permitiendo generar un perfil más amplio de un autor.

Como propuesta a esta parte del proyecto, en primer lugar, se crearon los grafos de palabras utilizando las etiquetas gramaticales de palabras en idioma inglés, en segundo lugar, se calcularon métricas de centralidad usadas en Python que sirvieron para crear vectores de input para los clasificadores.

De los resultados obtenidos se pudo observar poca precisión al tener un número de clases muy alto siendo este el principal problema en clasificación multiclase. Como se realizó en el experimento clasificando tópicos, estos arrojaron mejores resultados que clasificando por autor debido a que se clasifican menos clases. De igual modo con el set de datos C0 se lograron mejores resultados. Con los resultados obtenidos no se puede determinar con certeza la autoría de un documento, pero siendo un enfoque nuevo, se puede mejorar.

8. Referencias

[1] Aplicación de estilometría para la atribución de autorías en e-mails y documentos informáticos, Jorge Roberto Maldonado Galiano. Universidad San Francisco de Quito USFQ.

[2] Determinación de autoría por medio de redes de palabras- Nicolás Ignacio Zárate Guzmán/ Fernando Javier Püschel ArayaPuchel PUCV.

[3] Networks, Crowds, and Markets: Reasoning About a Highly Connected World, David Easley, Jon Kleinberg

[4] Frontini, F., Lynch, G. y Vogel, C. (2008): "Revisiting the Donation of Constantine". AISB Proceedings 7: 1-9.

[5] Zipf, G.K. (1932): Selected Studies of the Principle of Relative Frequency in Language. Harvard University Press.

Mosteller, F. y Wallace, D.L. (1964): Inference and Disputed Authorship: the Federalist. Addisson Wesley.

[6] L. a. B. Cedeño, «Detección automática de plagio en texto,» Valencia, 2008.

[7] D. R. Amancio, «Autorship recognition via fluctuation analysis of network topology and word intermittency» Sao Paulo, 2015.

[8] T. Solorio, S. Pillay, S. Raghavan y M. Montes y Gómez, «Modality Specific

Meta Features of Authorship Attribution in Web Forums Post» de *Proceeding of the 5th International Joint conference of natural language processing*, Chiang,Mai, Thailand, 2011.

[9] A. P. L. Monroy, «Atribución de Autoría utilizando Distintos tipo de

Características A través de una Nueva Representación,» 2012.

[10] Cárdenas, J. P., Losada, J. C., Moreira, A., Torre, I. G. & Benito, R. M. (2011). Topological complexity in natural and formal languages. Int. J. Complex Systems in Science, 1(2), 221-225.

[11] J. M. A. T. I. &. b. R. cárdemas, Topological Cpmplexity in Natural and Formal Languages., 2011.

[12] [Moreiro, 2002], Aplicaciones al análisis automático del contenido provenientes de la teoría matemática de la información.

[13] Freeman, L. (1977). «A set of measures of centrality based upon betweenness». Sociometry 40 (1): 35-41.

[14] Beauchamp, M. A. (1965). «An improved index of centrality». Systems Research and Behavioral Science 10 (2): 161-163

[15] Jimeng, Sun; Jie, Tang (2011). «A survey of models and algorithms for social influence analysis». En Charu C. Aggarwal. Social network data analytics.

[16]. Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.

[17]. Atribución de autoría mediante grafos, Andrés Vega, Iván Zamorano, PUCV Diciembre 2016

[18]. Hua Wang, Chris Ding and Heng Huang (2007), Multi-Label Classification: Inconsistency and Class Balanced K-Nearest Neighbors

Anexo

Lista de Stop Words obtenidas de Ranks.nl, la lista empleada es la default del habla inglesa.

"a"	"did"	"herself"	"not"	"the"	"we've"
"about"	"didn't"	"him"	"of"	"their"	"were"
"above"	"do"	"himself"	"off"	"theirs"	"weren't"
"after"	"does"	"his"	"on"	"them"	"what"
"again"	"doesn't"	"how"	"once"	"themselves"	"what's"
"against"	"doing"	"how's"	"only"	"then"	"when"
"all"	"don't"	"i"	"or"	"there"	"when's"
"am"	"down"	"i'd"	"other"	"there's"	"where"
"an"	"during"	"i'll"	"ought"	"these"	"where's"
"and"	"each"	"i'm"	"our"	"they"	"which"
"any"	"few"	"i've"	"ours"	"they'd"	"while"
"are"	"for"	"if"	"ourselves"	"they'll"	"who"
"aren't"	"from"	"in"	"out"	"they're"	"who's"
"as"	"further"	"into"	"over"	"they've"	"whom"
"at"	"had"	"is"	"own"	"this"	"why"
"be"	"hadn't"	"isn't"	"same"	"those"	"why's"
"because"	"has"	"it"	"shan't"	"through"	"with"
"been"	"hasn't"	"it's"	"she"	"to"	"won't"
"before"	"have"	"its"	"she'd"	"too"	"would"
"being"	"haven't"	"itself"	"she'll"	"under"	"wouldn't"
"below"	"having"	"let's"	"she's"	"until"	"you"
"between"	"he"	"me"	"should"	"up"	"you'd"
"both"	"he'd"	"more"	"shouldn't"	"very"	"you'll"
"but"	"he'll"	"most"	"so"	"was"	"you're"
"by"	"he's"	"mustn't"	"some"	"wasn't"	"you've"
"can't"	"her"	"my"	"such"	"we"	"your"
"cannot"	"here"	"myself"	"than"	"we'd"	"yours"
"could"	"here's"	"no"	"that"	"we'll"	"yourself"
"couldn't"	"hers"	"nor"	"that's"	"we're"	"yourselves"

Lista de **POS Tagg** utilizada de idioma ingles

CC	*coordinating conjunction*
CD	*cardinal digit*
DT	*determiner*
EX	*existential there (like: "there is" ... think of it like "there exists")*
FW	*foreign word*
IN	*preposition/subordinating conjunction*
JJ	*adjective 'big'*
JJR	*adjective, comparative 'bigger'*
JJS	*adjective, superlative 'biggest'*
LS	*list marker 1)*
MD	*modal could, will*
NN	*noun, singular 'desk'*
NNS	*noun plural 'desks'*
NNP	*proper noun, singular 'Harrison'*
NNPS	*proper noun, plural 'Americans'*
PDT	*predeterminer 'all the kids'*
POS	*possessive ending parent's*
PRP	*personal pronoun I, he, she*
PRP$	*possessive pronoun my, his, hers*
RB	*adverb very, silently,*
RBR	*adverb, comparative better*
RBS	*adverb, superlative best*
RP	*particle give up*
TO	*to go 'to' the store.*
UH	*interjection errrrrrrrm*
VB	*verb, base form take*
VBD	*verb, past tense took*
VBG	*verb, gerund/present participle taking*
VBN	*verb, past participle taken*
VBP	*verb, sing. present, non-3d take*
VBZ	*verb, 3rd person sing. present takes*
WDT	*wh-determinerwhich*
WP	*wh-pronoun who, what*
WP$	*possessive wh-pronoun whose*
WRB	*wh-abverb where, when*