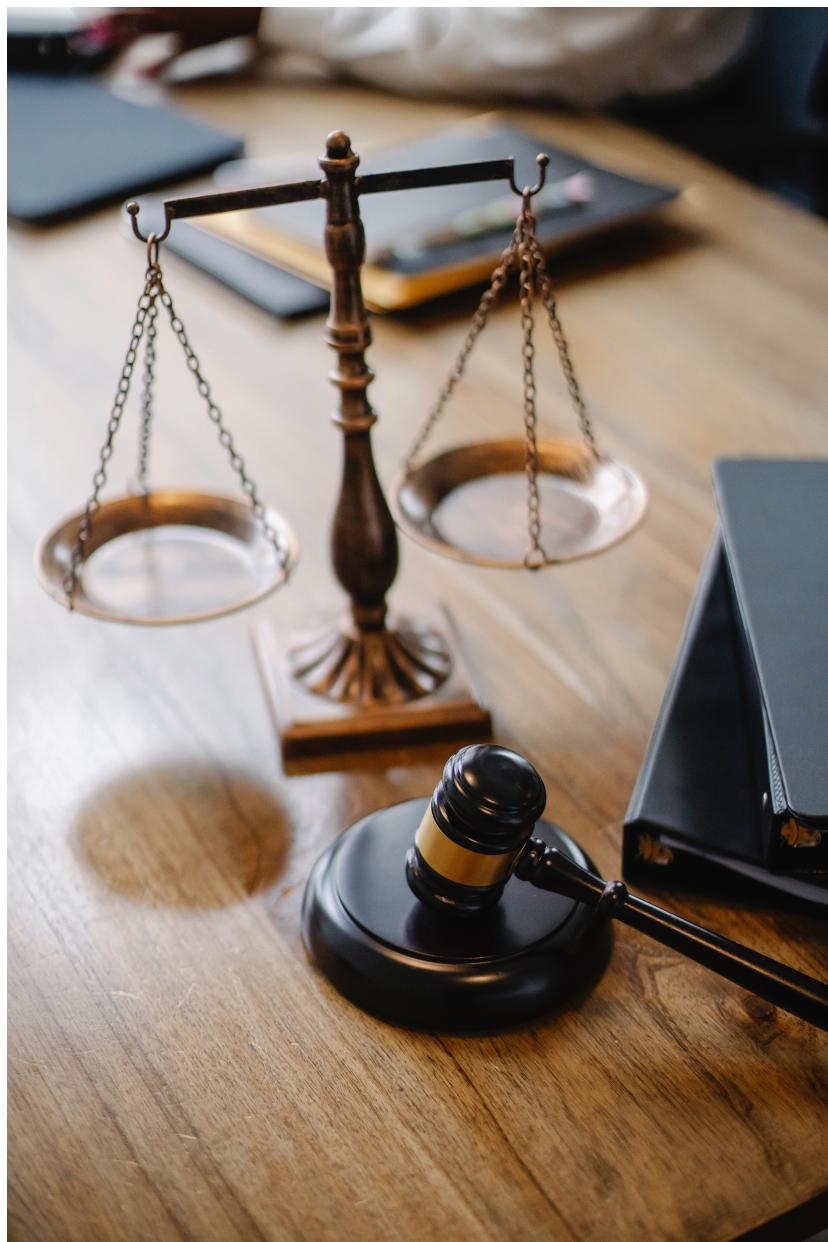


# Análisis en clústeres de los juicios de la Corte Suprema de EEUU

Gorka Cidoncha - Joel García - Marcos Martínez

1 de noviembre de 2022



# Índice

<b>1. Introducción</b>	<b>2</b>
1.1. Definición . . . . .	2
1.2. Objetivo . . . . .	2
<b>2. Algoritmo</b>	<b>3</b>
2.1. Pseudocódigo . . . . .	3
2.1.1. Preprocesamiento de los datos . . . . .	3
2.1.2. Creación del modelo . . . . .	3
<b>3. Diseño</b>	<b>4</b>
3.1. Clústering . . . . .	4
3.2. Inserción de nuevas instancias . . . . .	5
3.3. Mapa de diseño . . . . .	5
<b>4. Resultados experimentales</b>	<b>6</b>
4.1. Banco de pruebas . . . . .	6
4.2. Análisis crítico y discusión de resultados . . . . .	6
4.3. Rendimiento del software . . . . .	10
<b>5. Conclusiones</b>	<b>10</b>
5.1. Motivación para realizar clústering . . . . .	10
5.2. Conclusiones finales . . . . .	10
5.3. Propuestas . . . . .	10

## 1. Introducción

### 1.1. Definición

Para la realización de este proyecto hemos utilizado la clasificación no supervisada. Es una técnica exploratoria. Se disponen de un conjunto de instancias (que no han sido previamente clasificadas) y se desea agrupar las instancias en clústers (conjuntos) para que las instancias del mismo clúster presenten similitud interna y fuerte disimilitud entre las instancias de otros clústers (tienen gran parecido entre los del mismo clúster y a la vez, tengan gran diferencia entre los de otros clústers).

### 1.2. Objetivo

- Se dispone de un dataset en inglés que contiene 3304 juicios realizados por la Corte Suprema de los Estados Unidos desde 1955 hasta 2021.
- Cada juicio es una instancia del dataset y tiene 16 atributos<sup>[1]</sup>, entre los cuales se consideran las siguientes:
  - Los hechos del juicio, de media 200 palabras.
  - Un identificador único para cada juicio.
  - El tema sobre el que trata el juicio; derechos sociales, procedimientos criminales o otro campo
  - Un booleano que es True si el juicio lo ganó el denunciante
- Nuestro primer objetivo ha sido conseguir agrupar todas las instancias en diferentes clústeres según los tópicos a los que pertenecen tras analizar los hechos de cada juicio. De esta forma, cada hecho puede pertenecer a un sólo tópico (muy poco probable) con una probabilidad de 1 sobre 1 o puede tener distintas probabilidades de pertenecer a más de un tópico.

## 2. Algoritmo

### 2.1. Pseudocódigo

#### 2.1.1. Preprocesamiento de los datos

- Cargar los datos del csv
- Eliminar los signos de puntuación y pasar todo a minúsculas (sent\_to\_words)
- Eliminar las stop words<sup>1</sup> (remove\_stopwords)
- Lematizar (lemmatization)

#### 2.1.2. Creación del modelo

- Realizar el topic modeling
  - Conseguir para cada tópico, el porcentaje de aparición de cada palabra (suma total 1)
- Generar la matriz, en la que las filas son los documentos y las columnas los tópicos
  - Recorrer los documentos
    - Recorrer cada palabra del documento
      - ◊ Guardar en un vector la información
    - Guardar en un vector la información del documento
- Realizar el word cloud<sup>2</sup> de un documento y de los dos tópicos a los que más se parece
- Comparar las dimensiones de antes y después de realizar el PCA<sup>3</sup>
- Crear la función para calcular la distancia euclídea entre dos clústers
  - Recorrer todo el cluster(i)
    - Distancia = Distancia +  $\sqrt{((cluster1(i) - cluster2(i))^2)}$
- Crear la función que genera una matriz con las distancias entre cada clúster
  - Recorrer los clusters(i)
    - Llamar a la función de calcular distancias euclídeas con el cluster(i) con el resto de clusters
    - Introducir la información de las distancias de todos los clusters con el cluster(i) en un vector
- Agrupar la fila y columna de la matriz de distancias con menor distancia
  - Llamar a la función de generar la matriz de distancias
  - Mientras la matriz sea mayor que 1
  - Coger la fila y columna que tengan la menor distancia
  - Agruparlas en una nueva columna y fila y eliminar las anteriores
- Crear una función que dado el número de clústers, devuelve a qué clúster pertenece cada instancia, distancia y en cada clúster qué instancias hay
- Crear una función que dado un número de instancias máximo, devuelve a qué clúster pertenece cada instancia, el número de clusters y en cada clúster qué instancias hay

---

<sup>1</sup>Conjunciones, artículos, preposiciones y adverbios en su mayoría que no aportan información sobre el tema que trata el texto

<sup>2</sup>Representación visual de las palabras que aparecen en el texto, cambiando su tamaño en función de la frecuencia de aparición de la palabra en el texto

<sup>3</sup>Método estadístico que permite simplificar la complejidad de espacios muestrales con muchas dimensiones conservando su información

- Hacer la representación gráfica en forma de árbol
- Hacer la representación gráfica en forma de puntos de la variable “first party winner” (True si gana el denunciante, False si no)
- Hacer la representación gráfica en forma de puntos de la variable “issue area”
- Crear una función para que calcule los centroides de los clústers
  - Recorrer todos los clústers (i)
    - Calcular la posición de todas las instancias de cada clúster
    - Calcular el centroide para el clúster (i)
- Representamos gráficamente los porcentajes obtenidos dependiendo de los parámetros utilizados
- Crear una función para añadir una nueva instancia y clasificarla
  - Eliminar los caracteres extraños
  - Eliminar las stop-words
  - Lematizar
  - Vectorizar
  - Realizar el PCA
  - Calcular los centroides
  - Calcular la distancia menor a los centroides
  - Imprimir la distancia seleccionada y el clúster asociado
- Generar la función para crear la matriz de confusión
  - Pasar los atributos del k-means a string
  - Obtener la matriz de confusión
  - Borrar las columnas vacías
  - Generar la matriz
- Generar la función para calcular cuántos ha acertado y con qué porcentaje de acierto
  - Imprimir el número total de instancias, los correctos y el porcentaje de acierto de los correctos
- Representar gráficamente la matriz de confusión de la variable “issue-area”
- Representar gráficamente la matriz de confusión de la variable “first party winner”
- Realizar una función que pruebe nuestro programa con distintos parámetros y guarde los datos obtenidos, para poder ver cuáles son los mejores parámetros a utilizar
- Finalmente, guardar los datos obtenidos anteriormente en un fichero

### 3. Diseño

#### 3.1. Clústering

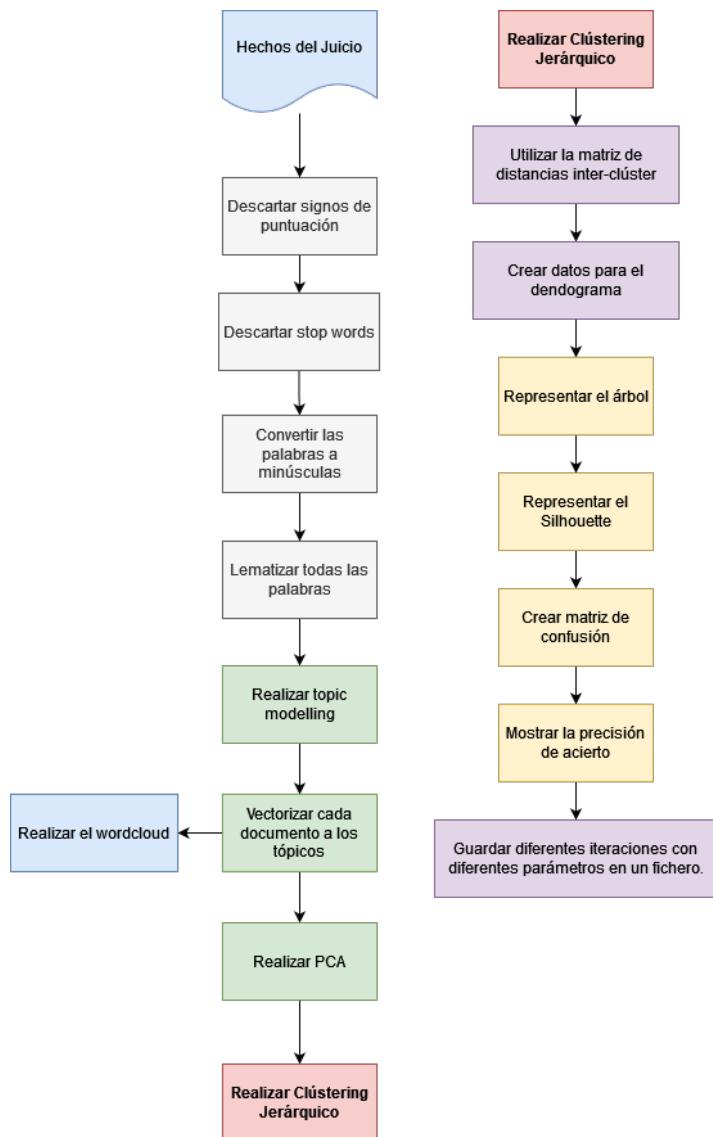
- Para conseguir los tópicos hemos utilizado la técnica no supervisada *Topic Modeling* junto con el algoritmo *Latent Dirichlet Allocation* (LDA), ya que ya aprendimos a utilizarla en otra asignatura y nos es familiar.
- Para conseguir los clústers nos ha tocado utilizar el clustering jerárquico. Trata de inicialmente tratar cada una de las instancias como un clúster y en cada iteración agrupar los clústers según su distancia inter-clúster. De esta forma se puede apreciar que clústeres se parecen más entre sí por la similitud de su conjunto de instancias.
- Hemos utilizado la distancia Single-Link<sup>[2]</sup>, la cual se define como la distancia menor entre pares de elementos de los dos clusters y la distancia de Euclídea para calcular las distancias entre clústers.

### 3.2. Inserción de nuevas instancias

- De cara a añadir nuevas instancias y clasificarlas, hemos guardado los centroides de cada clúster después de realizar el clústering jerárquico para así no tener que realizar todo el proceso previo otra vez para clasificar una nueva instancia a un clúster.
- Cuando se quiere clasificar una nueva instancia, calculamos las distancias de la instancia a los centroides del resto de clusters, y la clasificamos en el clúster con más cercano.

### 3.3. Mapa de diseño

- El siguiente diagrama representa primero el proceso previo necesario para realizar el clústering jerárquico. Después se resumen brevemente los pasos realizados en el cuaderno de *Jupyter Notebook* para representar el dendograma con todos los clústeres, la representación del gráfico *Silhouette* utilizado para interpretar la coherencia de los clústeres, la matriz de confusión, la precisión de acierto y el guardado de los parámetros del clústering junto con su precisión para poder valorar cuáles son los mejores.



## 4. Resultados experimentales

### 4.1. Banco de pruebas

- Al principio hemos probado con pocos datos y tópicos para que el programa se ejecute más rápido y poder ir solucionando los problemas. Una vez que funcionaba con la muestra de prueba hemos realizado pruebas con todos los datos.
- Hemos probado con distintos números de clusters, quedándonos finalmente con el que mejor nos ha parecido en la métrica de evaluación de calidad.
- También hemos probado sin realizar algún preprocesso como el lematizar, es más rápido pero menos eficiente.

### 4.2. Análisis crítico y discusión de resultados

- Con los distintos resultados del accuracy<sup>4</sup> que hemos obtenido variando el número de clusters, el PCA y el número de tópicos los hemos representado gráficamente para predecir el ganador del juicio y así hemos decidido cuáles son los mejores parámetros para obtener un mayor porcentaje de accuracy.
- En la figura 1 podemos ver que el mejor número de tópicos que podemos utilizar es aproximadamente 200.

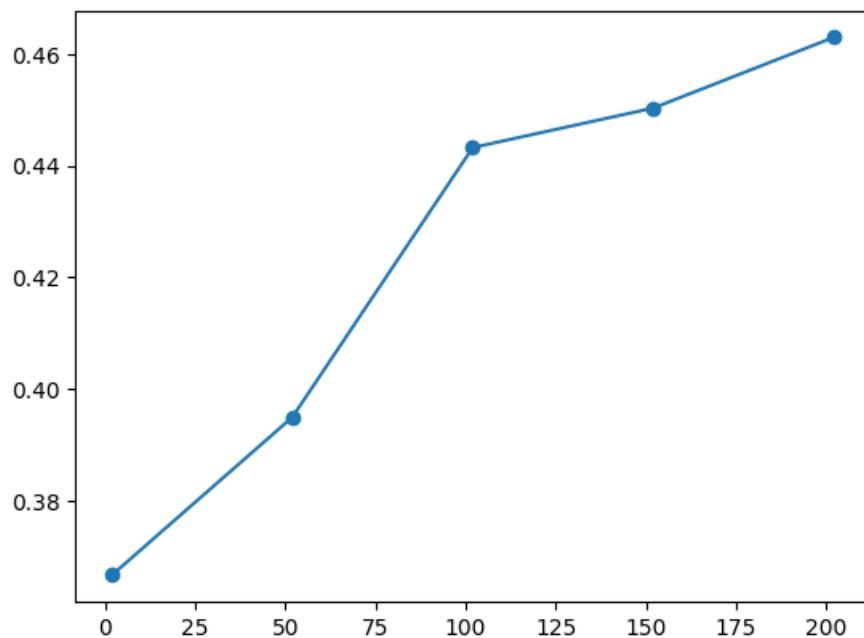


Figura 1: El eje X representa el número de tópicos y el eje Y el accuracy obtenido

<sup>4</sup>Es una métrica que representa el porcentaje total de valores correctamente clasificados, tanto positivos como negativos

- En la figura 2 podemos observar que el mejor PCA que podemos utilizar es 150, pero la silueta de la figura 3 es muy pequeña con 150, por lo que tras realizar varias pruebas es mejor utilizar 2 dimensiones para el PCA.

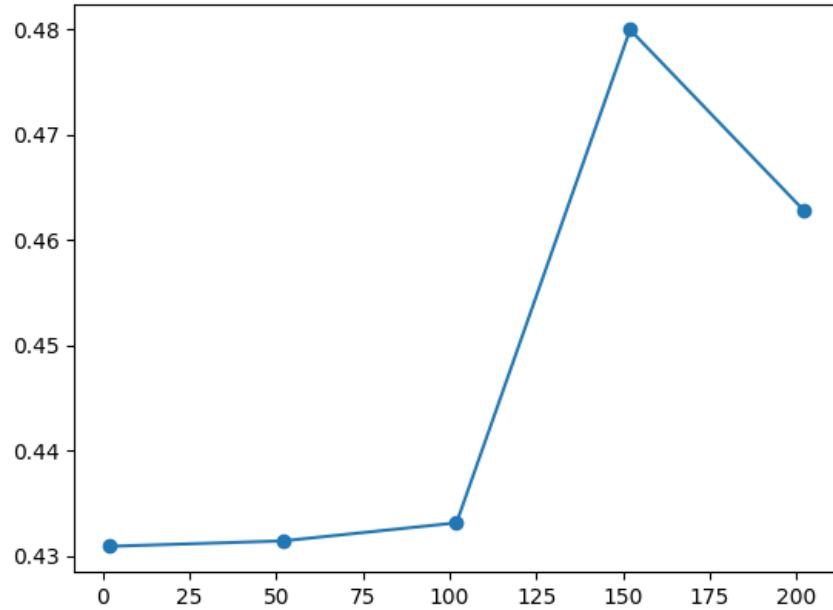


Figura 2: El eje X representa el número de PCA y el eje Y el accuracy obtenido

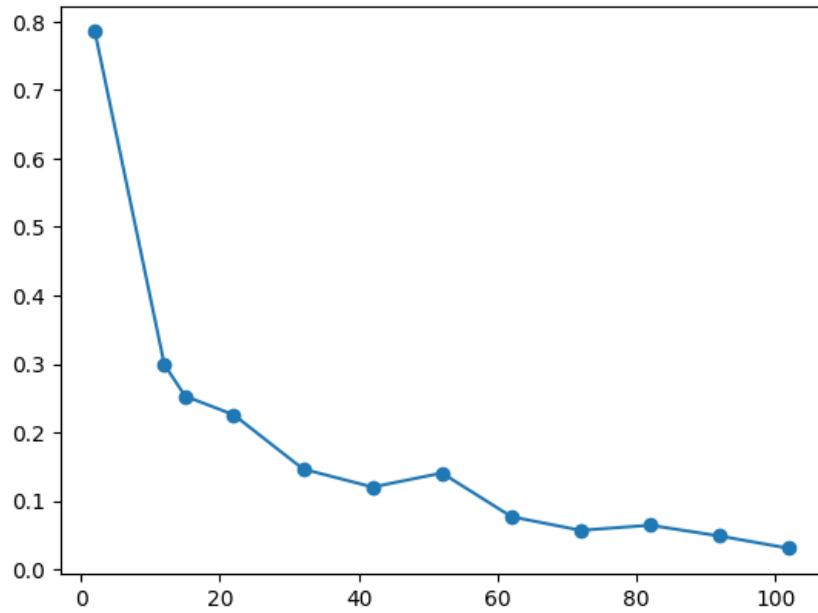


Figura 3: El eje X representa el número de PCA y el eje Y la silueta obtenida

- En la figura 4 podemos apreciar que el mejor número de clusters a realizar es aproximadamente 2.

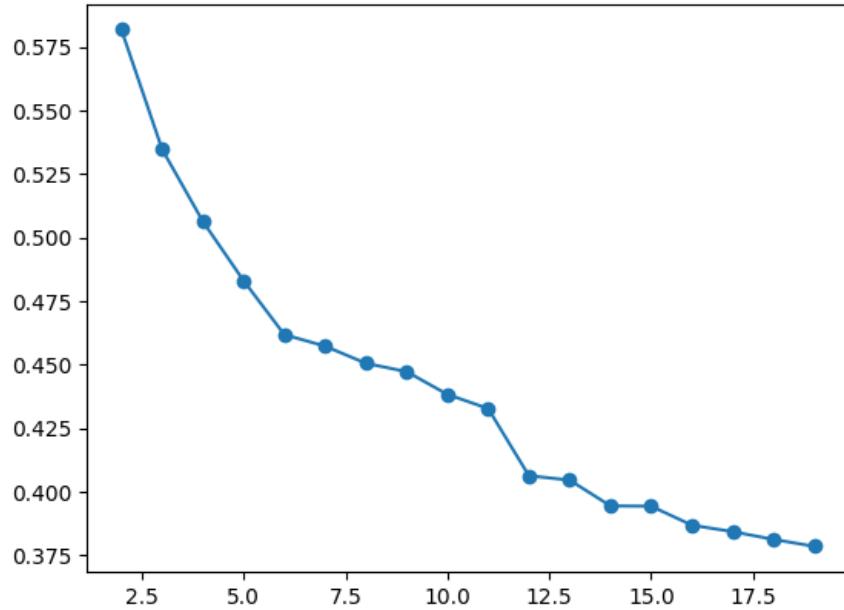


Figura 4: El eje X representa el número de clústers y el eje Y el accuracy obtenido

- Por lo que podemos ver en los gráficos, los mejores valores a utilizar son los siguientes:
  - Número de clusters → 2
  - PCA → 2
  - Número de tópicos → 202
  - Conseguimos un accuracy para el valor de “firts party winner” → 0.7659
- Con todos estos datos, vemos en la figura 5 que obtenemos la siguiente silueta:

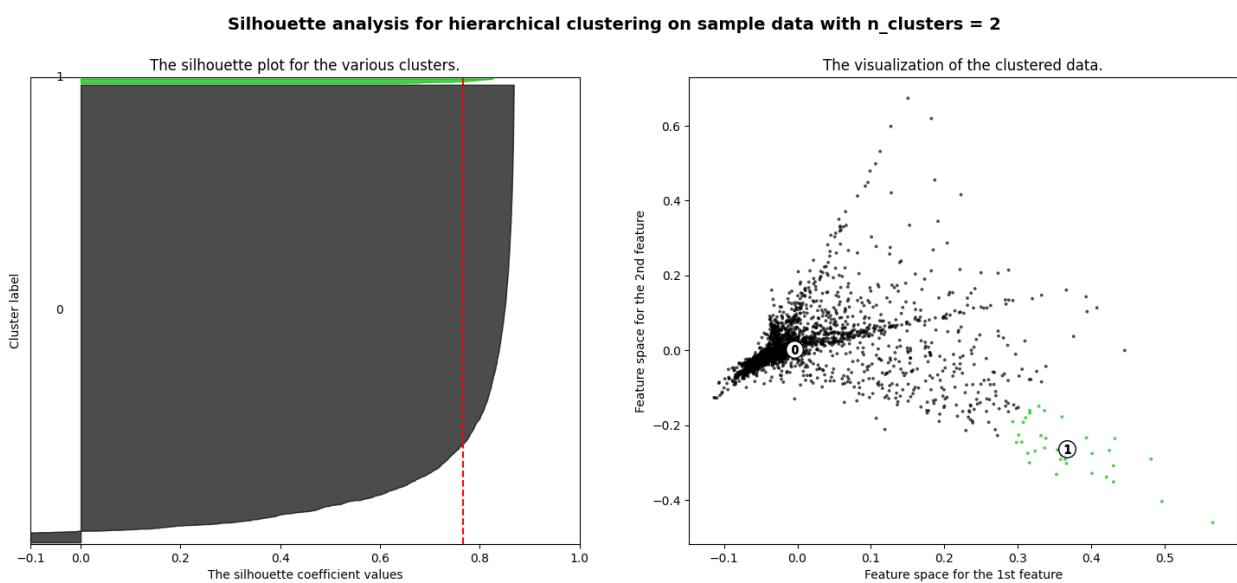


Figura 5: PCA 2, NClusters 2, NTopics 202

- Utilizando los valores obtenidos para las variables de nuestro programa,
- También, hemos comparado nuestros resultados con los obtenidos con un algoritmo de clasificación aleatoria. En la figura 6 se muestra la matriz de confusión y acierta con un accuracy del **0.5122**. No sería aceptable que un algoritmo aleatorio fuese igual o mejor que el nuestro

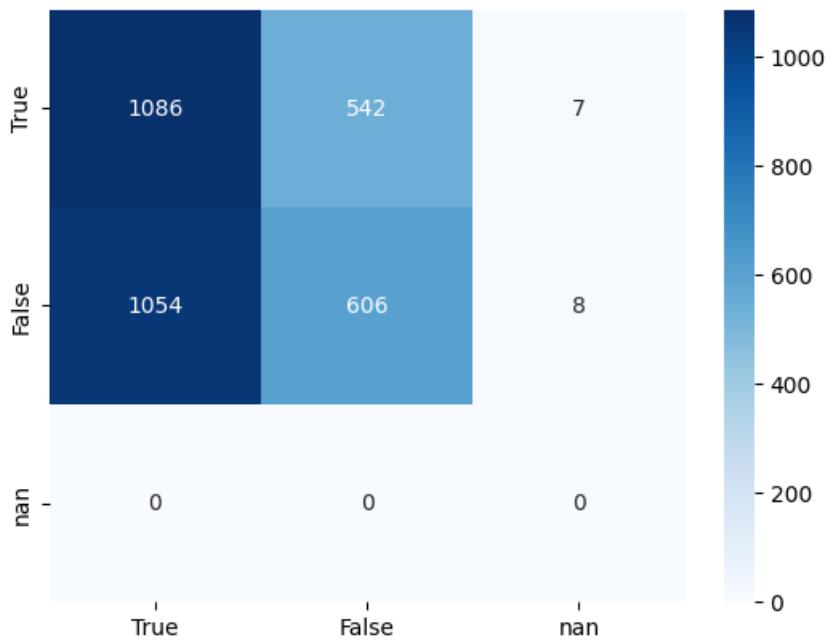


Figura 6: Matriz de confusión utilizando un algoritmo de clasificación aleatoria. Score: 0.5122

- La figura 7 es la matriz de confusión de nuestro algoritmo, que tiene un accuracy del **0.7659** frente al **0.5122** del aleatorio, por lo que podemos decir que nuestro programa ha aprendido, ya que es mucho mejor que uno aleatorio

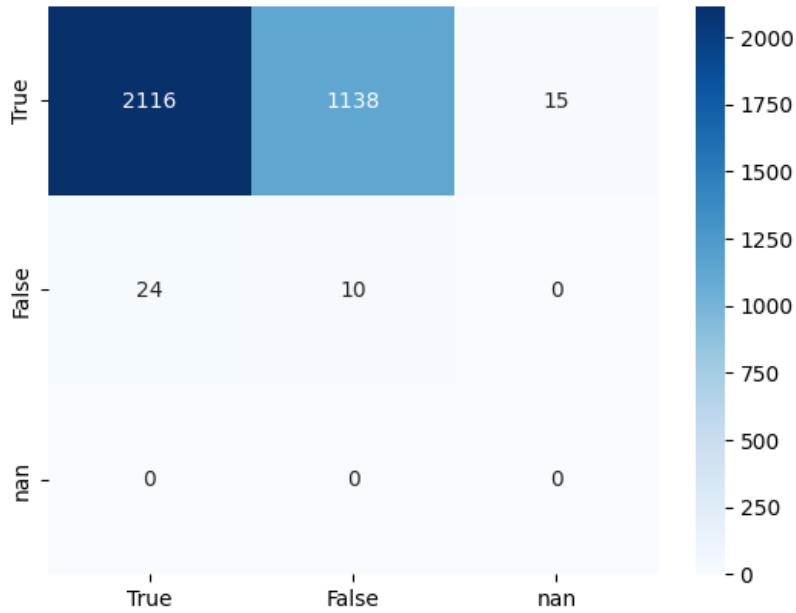


Figura 7: Matriz de confusión utilizando nuestro algoritmo de clasificación. Score: 0.7659

- Finalmente, hemos comparado leyendo la información de dos documentos que pertenecen al mismo clúster y dos de distintos clústers para ver si podemos observar a simple vista que la clasificación tiene sentido o no y hemos observado que visualmente tampoco se puede clasificar de manera firme una instancia en un clúster.

#### 4.3. Rendimiento del software

- La parte más lenta y más costosa de nuestro software es la parte de realizar el clústering jerárquico ya que es la parte que hemos realizado nosotros y no hemos utilizado librerías externas. Como lo hemos implementado nosotros, tiene margen de mejora y se puede optimizar más su funcionamiento y coste, ya que además es una parte importante del proyecto.
- Con los resultados obtenidos podemos observar que el porcentaje de accuracy que conseguimos es 0.7659 para predecir quién ha ganado el juicio y se puede mejorar, ya que no es un porcentaje muy alto y realizando algunas mejoras al software el porcentaje puede aumentar.

### 5. Conclusiones

#### 5.1. Motivación para realizar clústering

- Nuestra mayor motivación para realizar el proyecto ha sido poder aplicar la teoría aprendida en clase, creando nosotros el algoritmo para poder realizar el clústering jerárquico.
- Por otra parte, nos ha gustado el reto de ponernos a prueba y poder sacar adelante un proyecto bastante completo, desde nuestro punto de vista, a demás, con la dificultad añadida de tener distintas disponibilidades horarias entre los miembros del grupo.

#### 5.2. Conclusiones finales

- Se ha implementado un algoritmo capaz de agrupar más de 3000 juicios de la Corte Suprema de Estados Unidos analizando los hechos del juicio y buscando similitudes y diferencias entre ellos. Se han conseguido unos resultados satisfactorios, ya que se ha obtenido un accuracy de 0.7659 a la hora de predecir quién ha ganado el juicio. Sin embargo, se puede mejorar, tampoco es un porcentaje muy bajo por lo que estamos satisfechos con el trabajo realizado.
- Nos ha parecido un proyecto muy completo y útil para complementar lo aprendido en teoría de una manera práctica.
- Además hemos podido entender mucho mejor todo el proceso que tienen que realizar los datos a través del software, para finalmente, el propio software sea capaz clasificar una nueva instancia en un clúster.

#### 5.3. Propuestas

- Nuestro algoritmo se puede mejorar de distintas maneras que no hemos tenido tiempo de probar, por ejemplo:
  - Guardando todas las instancias de los clústers para poder utilizar otros criterios a la hora de añadir una nueva instancia y clasificarla en un clúster.
  - Utilizando la distancia a la instancia más cercana de cada clúster o utilizando la distancia a la instancia más lejana de cada clúster.
  - Utilizando distintas inicializaciones, como la de dividir el espacio (depende de las dimensiones del espacio puede ser muy costoso) o empezando inicializando aleatoriamente la matriz de bits de pertenencia.
  - Utilizando distintas métricas, como por ejemplo, la distancia Manhattan o la distancia Minkowski.

- Utilizar Complete-Link<sup>5</sup> o Average-Link<sup>6</sup> para calcular las distancias entre los elementos de dos clusters a la hora de hacer el clústering jerárquico.

## Referencias

- [1] Supreme Court Judgment Prediction. (s. f.). Kaggle. Recuperado 13 de octubre de 2022, de <https://www.kaggle.com/datasets/deepcontractor/supreme-court-judgment-prediction>
- [2] Hierarchical Agglomerative Clustering Algorithm Example In Python .. (s. f.). towardsdatascience.Com. Recuperado 28 de octubre de 2022, de <https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglomerative-clustering-example-in-python-1e18e0075019>

---

<sup>5</sup>La distancia entre dos clusters, se define como la distancia mayor entre pares de elementos de los dos clusters

<sup>6</sup>La distancia entre dos clusters, se define como la distancia entre sus centroides