

## Práctica 2

### **1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?**

En esta práctica vamos a continuar con el dataset generado en la práctica anterior. Como hemos explicado previamente, la idea de estas dos prácticas es automatizar la búsqueda y selección de artículos científicos de interés.

La automatización de búsqueda de información relevante para un investigador es un proceso que puede beneficiar muchísimo a un científico ya que le permitirá estar más tiempo haciendo su trabajo que una máquina no puede hacer.

En esta práctica vamos a mirar cual es el porcentaje de papers que han aplicado ML y AI en el ámbito de inmunología, qué algoritmos se han usado y si hay una tendencia de aumento de papers más computacionales en los últimos años.

### **2. Integración y selección de los datos de interés a analizar. Puede ser el resultado de adicionar diferentes datasets o una subselección útil de los datos originales, en base al objetivo que se quiera conseguir.**

En la anterior práctica sólo hemos trabajado con los artículos publicados en el año 2022, no obstante hemos pensado que sería más interesante aumentar este número y trabajar con papers en los últimos 5 años (2017-2022) con un total de 549 artículos.

Recordemos las columnas de nuestro dataset:

*Artículo:* Nombre del artículo científico.

*Summary:* Resumen general de la investigación que se ha llevado a cabo.

*Authors:* Nombre de los autores del artículo científico.

*Date:* Fecha de publicación.

*Access:* Si es de acceso público o se tiene que pagar para acceder al artículo.

*Figure:* Una imagen representativa del estudio en cuestión.

*Link paper:* Link del artículo científico, los artículos de acceso no público muestran el título del artículo y el resumen general del artículo.

TCR, BCR, T CELL, B CELL, NKC, CD4, CD8, DEEP LEARNING, MACHINE LEARNING ML DL CNN LSTM y HLA: palabras clave que nos interesan, en caso de que el estudio contenga esa palabra clave el valor va a ser 1 y en caso de no contenerla será 0.

De estas columnas, los keywords es la parte más interesante ya que nos permitirá filtrar el data por los diferentes tipos de células (T y B cells) y ver cuántos papers ML y AI para hacer el análisis.

Una vez tengamos los papers de interés podemos aplicar NLP para ver la similitud de los artículos. La idea que tenemos es a través del abstract crear un heatmap con las distancias entre los papers de las subcategorías como T-cell ML, B-cell ML por ejemplo para que nos sea más fácil ver papers parecidos a los papers que hemos detectado como papers de interés manualmente. Por ejemplo, hemos sacado todos los papers de interés y hemos leído el primero que va de clasificación de células T CD4+ utilizando ML, con el heatmap podemos ver que otros papers son parecidos a este para no tener que leerlos todos los papers de interés, sino solo los mas parecidos al nuestro paper de interés actual.

### **3. Limpieza de los datos.**

#### **3.1. ¿Los datos contienen ceros o elementos vacíos? Gestiona cada uno de estos casos.**

El primer paso en la limpieza de datos sería verificar que no hay missing data en nuestro dataset. A la hora de diseñar el código de web scraping, hemos intentado hacerlo de tal manera que el data sacado de la página web sea lo más completo posible.

Podemos ver que no hay missing data ya que todas las columnas tienen el máximo de non-null values. No obstante vemos que las columnas de keywords se identifican como integers, vamos a cambiarlas a string.

### **3.2. Identifica y gestiona los valores extremos.**

Ahora vemos que todas las variables que tenemos no son numéricas y la columna Date es de tipo date. Vemos que si a la hora de diseñar el código de web scraping se hace un esfuerzo para obtener el data de mayor calidad posible, la limpieza de datos será fácil.

Dado que no es un dataset numérico, no tenemos que revisar outliers ni ningún tipo de valores extremos.

## **4. Análisis de los datos.**

**4.1. Selección de los grupos de datos que se quieren analizar/comparar (p.ej., si se van a comparar grupos de datos, ¿cuáles son estos grupos y qué tipo de análisis se van a aplicar?)**

**4.2. Comprobación de la normalidad y homogeneidad de la varianza.**

**4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos.**

**En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.**

Como ya hemos dicho previamente, el análisis principal consistirá de análisis de similitud de artículos en función del abstract de los artículos. Para ello vamos a necesitar definir cómo vamos a calcular la similitud o la distancia entre los artículos.

La primera parte de este problema es la representación. ¿Cómo representamos el texto? Podríamos dejar el texto tal como está o convertirlo en vectores de características utilizando una técnica de incrustación de texto adecuada. Una vez que tenemos la representación del texto, podemos calcular la puntuación de similitud usando una de las muchas medidas de distancia/similitud.

## **Métricas de distancia**

### **a. Índice Jaccard**

El índice de Jaccard, también conocido como coeficiente de similitud de Jaccard, trata los objetos de datos como conjuntos. Se define como el tamaño de la intersección de dos conjuntos dividido por el tamaño de la unión.

Para calcular la similitud utilizando la similitud de Jaccard, primero realizaremos la normalización del texto para reducir las raíces/lemas de las palabras.

### **b. Distancia euclidiana**

La distancia euclidiana, o norma L2, es la forma más utilizada de la distancia de Minkowski. En términos generales, cuando la gente habla de distancia, se refiere a la distancia euclidiana. Utiliza el teorema de Pitágoras para calcular la distancia entre dos puntos.

Cuanto mayor sea la distancia entre dos vectores, menor será la puntuación de similitud y viceversa. Las distancias pueden variar de 0 a infinito, necesitamos usar alguna forma de normalizarlas al rango de 0 a 1.

Aunque tenemos nuestra fórmula de normalización típica que usa la media y la desviación estándar, es sensible a los valores atípicos. Eso significa que si hay algunas distancias extremadamente grandes, cualquier otra distancia se hará más pequeña como consecuencia de la operación de normalización. Usando la fórmula:  $1/\exp(\text{distance})$ .

### **c. Cosine Similarity**

Cosine Similarity calcula la similitud de dos vectores como el coseno del ángulo entre dos vectores. Determina si dos vectores apuntan aproximadamente en la misma dirección. Entonces, si el ángulo entre los vectores es 0 grados, entonces la similitud del coseno es 1.

### **d. ¿Qué métrica usar?**

La similitud de Jaccard tiene en cuenta solo el conjunto de palabras únicas para cada documento de texto. Esto lo convierte en el candidato probable para evaluar la similitud de los documentos cuando la repetición no es un problema. Un excelente ejemplo de una aplicación de este tipo es comparar descripciones de productos. Por ejemplo, si un término como "HD" o "eficiencia térmica" se usa varias veces en una descripción y solo una vez en otra, la distancia euclidiana y

la similitud del coseno disminuirían. Por otro lado, si el número total de palabras únicas permanece igual, la similitud de Jaccard permanecerá sin cambios.

Dicho esto, la similitud de Jaccard no se suele dar cuando se trabaja con datos de texto, ya que no funciona con embeddings de texto. Esto significa que se limita a evaluar la similitud léxica del texto, es decir, qué tan similares son los documentos a nivel de palabras.

En lo que respecta a las métricas de coseno y euclidianas, el factor diferenciador entre las dos es que la similitud del coseno no se ve afectada por la magnitud/longitud de los vectores de características. Digamos que estamos creando un algoritmo de etiquetado de temas. Si una palabra (por ejemplo, *senado*) aparece con más frecuencia en el documento 1 que en el documento 2, podríamos suponer que el documento 1 está más relacionado con el tema de la política. Sin embargo, también podría darse el caso de que estemos trabajando con artículos de noticias de diferente extensión. Entonces, la palabra '*senado*' probablemente apareció más en el documento 1 simplemente porque era mucho más larga. Como vimos anteriormente cuando se repitió la palabra "*vacío*", la similitud del coseno es menos sensible a la diferencia de longitudes.

Además de eso, la distancia euclidiana no funciona bien con los vectores dispersos de incrustaciones de texto. Por lo tanto, generalmente se prefiere la similitud del coseno a la distancia euclidiana cuando se trabaja con datos de texto.

En el siguiente apartado vamos a explorar como hacer los embeddings de palabras/textos.

## **Embeddings de texto**

Hay mil y una manera de convertir palabras en números. Como ejemplo podemos poner One-hot embedding, Term Frequency-Inverse Document Frequency embedding, etc. Sequence embeddings es un ámbito en el que tenemos bastante experiencia y podemos decir que la metodología más robusta que suele funcionar mejor es algo como Word2Vec.

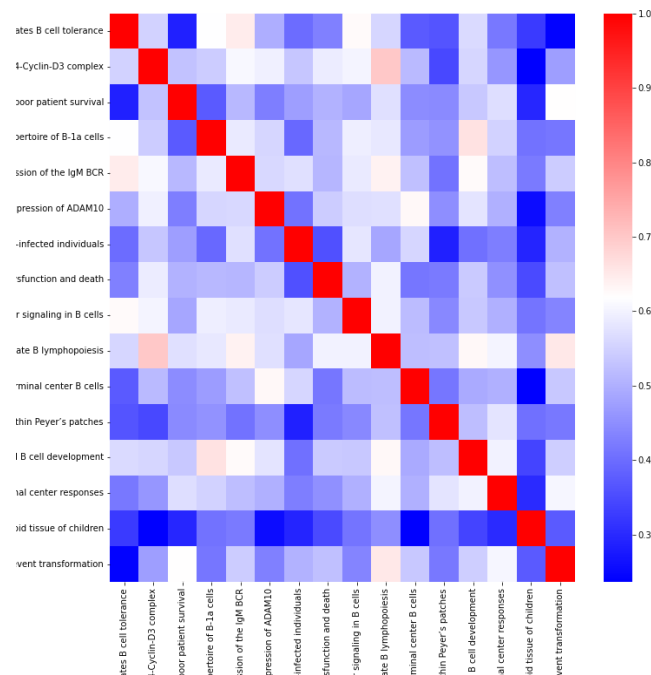
- a. Word2Vec: Word2Vec es un método predictivo para formar embeddings de palabras. Word2Vec es una red neuronal de dos capas pre entrenada. Toma como entrada el corpus de texto y genera un conjunto de vectores de características que representan palabras en ese corpus. Esto nos permite generar un vector numérico por palabra que podemos usar para calcular distancias euclidianas y cosenas. No obstante Word2Vec tiene la limitación de hacer embeddings solo por palabra y esto es una limitación ya que queremos contextualizar un embedding por todo el abstract. Para esto podemos usar otro modelo como SBET.
- b. Sentence-BERT (SBERT): SBERT es una red gemela que le permite procesar dos oraciones de la misma manera, simultáneamente. Estos dos gemelos son idénticos en todos los parámetros (su peso está vinculado), lo que nos permite pensar en esta arquitectura como un solo modelo utilizado varias veces. Es una extensión de BERT que está optimizada para frases en vez de palabras y añade el aspecto de contexto a la hora de hacer los embeddings. Para esta práctica vamos a usar SBERT.

Podemos ver que la mayoría de papers se centran en T cells, tanto en CD4 como en CD8. Hay papers de B cell y BCR, pero no son tan comunes. Sabiendo esto vamos a hacer 4 subdatasets para los heatmaps: T cell == 1, B cell ==1, TCR==1, BCR==1. Desafortunadamente hemos visto que no hay muchos papers sobre la parte mas computacional de inmunología, así que ML y DL serán keywords de menos importancia para este análisis pero podemos ver si los papers que mencionan los receptores (TCR y BCR) puede que tengan algunos otros keywords que nosotros no hemos puesto que están relacionados con aprendizaje automático ya que son las partes de células que más se usan en este ámbito o juntarlos en un grupo de ML and DL para ver si asi podemos sacar mas papers y hacer un heatmap más relevante.

- 5. Representación de los resultados a partir de tablas y gráficas. Este apartado se puede responder a lo largo de la práctica, sin necesidad de concentrar todas las representaciones en este punto de la práctica.**

Después de considerar todas las métricas, vamos a proceder con la cosine similarity ya que sería la métrica más adecuada para este problema ya que de entrada tiene límites en 0 y 1 y no necesita normalización, además es súper eficiente de calcular.

## BCR dataset



## Query de los papers más parecidos

Dado que un heatmap de alrededor de 500 papers pensamos que la parte visual se puede perder. Para ello hemos añadido una función para poder sacar los N papers más parecido al paper de interés.

Para ello vamos a utilizar los abstracts como antes y ya que tenemos la matriz de las similitudes entre todos los papers y la vamos a reutilizar aquí.

La idea es sacar el título junto al índice del paper en el dataset para que sea fácil sacar el link del artículo en cuestión

Article of reference: The lung environment controls alveolar macrophage metabolism and responsiveness in type 2 inflammation

Idex in dataframe	Article
303	STEEP mediates STING ER exit and activation of signaling
472	Distinct changes in endosomal composition promote NLRP3 inflammasome activation
428	The tumor suppressor kinase DAPK3 drives tumor-intrinsic immunity through the STING–IFN- $\beta$ pathway
88	G3BP1 promotes DNA binding and activation of cGAS
95	A TCR mechanotransduction signaling loop induces negative selection in the thymus

**6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?**

En esta práctica hemos creado una manera bastante visual y rápida para facilitar la búsqueda de artículos de interés en el ámbito de inmunología de la página web Nature Immunology. Al empezar las dos prácticas teníamos en mente la idea de hacer la búsqueda de los artículos más automática, eficiente y útil.

La primera práctica de web scraping nos ha permitido sacar artículos, sus abstracts y más información relevante como los keywords. Los keywords en nuestro caso se basan en T y B cells y aprendizaje automático. En esta práctica nos hemos limitado a 5 subsets de data para hacer el análisis, pero pueden haber tantos como las combinaciones de keywords hay, también las keywords pueden cambiar en función de los intereses del investigador.

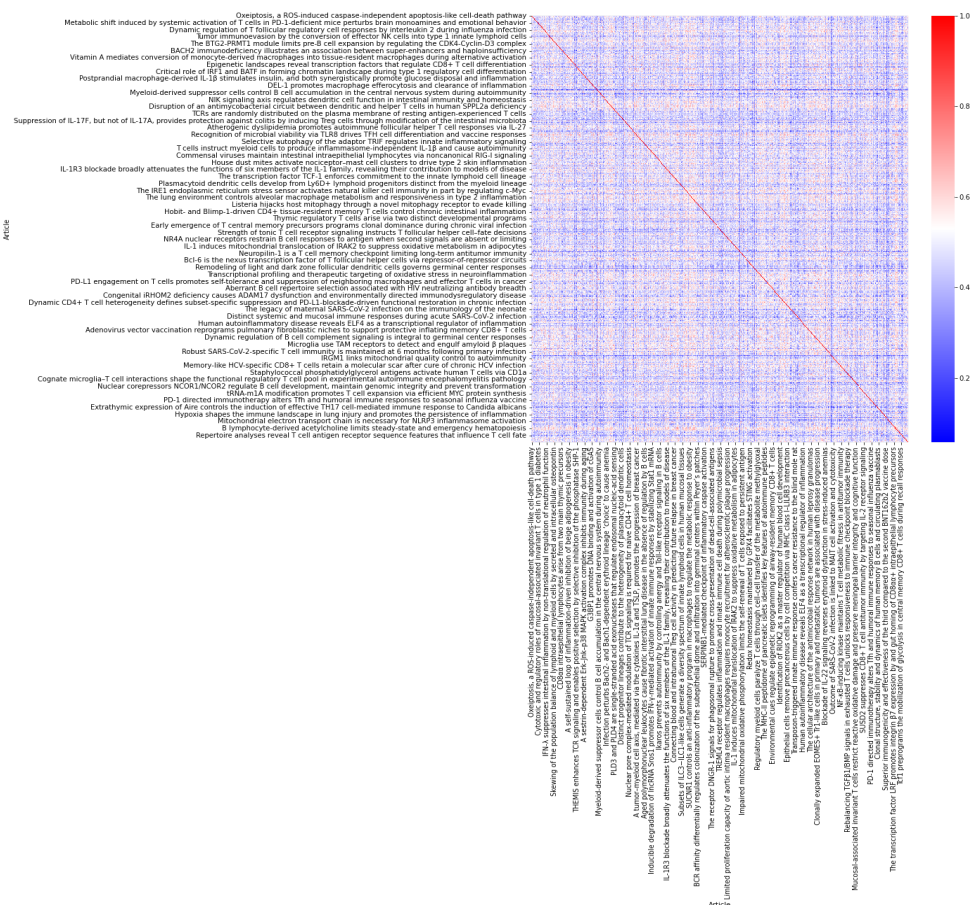
En esta práctica nos hemos centrado a ver cómo podemos usar text data que proviene de los artículos para hacer una búsqueda de los artículos más similares entre ellos. Para ello hemos explicado y aplicado las técnicas de sentence embedding y distance between vectors para poder crear un heatmap de



distancias entre los artículos. Esto nos da una herramienta visual que enseña cómo de similares son los artículos.

¿Por qué es útil? Vamos a poner un ejemplo. Supongamos que el investigador decide leer un artículo de los que hemos sacado con web scraping. Le parece interesante y quiere ver que otros artículos van sobre un tema parecido, para ello puede consultar el heatmap y ver el abstract de los papers con menor distancia a nivel de los embeddings. De esta manera no tendrá que leer manualmente abstracts de decenas de artículos, sino de una subselección de 5 por ejemplo y estos ya seguramente serán de gran interés. Si el heatmap se vuelve más difícil de ver y comprender, usando las distancias cosas, se pueden sacar top N artículos más similares al artículo de interés si es necesario.

Para acabar queríamos poner un heatmap de todos los papers para ver como de diversos han sido los últimos 5 años de Nature Immunology a nivel de contenido de los artículos.



- 7. Código.** Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y representación de los datos. Si lo preferís, también podéis trabajar en Python.

El código lo podemos encontrar dentro del Jupyter Notebook en el github.


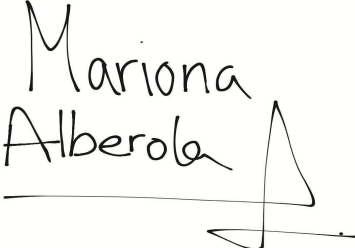
Link al github: [https://github.com/mariona9906/practica1\\_tipologia](https://github.com/mariona9906/practica1_tipologia) (el mismo que en la práctica 1 ya que es un proyecto y no dos separados)


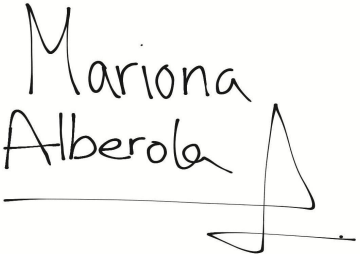

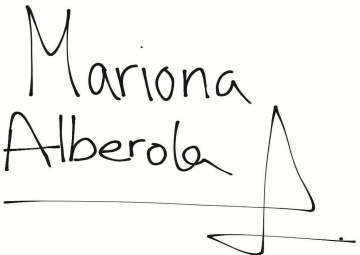

Hemos utilizado esta herramienta ya que pensamos que para un análisis y una práctica como esta ya que hace el seguimiento de los pasos de la limpieza y el análisis de los datos más simple.

- 8. Vídeo.** Realizar un breve vídeo explicativo de la práctica (máximo 10 minutos), donde ambos integrantes del equipo expliquen con sus propias palabras el desarrollo de la práctica, basándose en las preguntas del enunciado para justificar y explicar el código desarrollado. Este vídeo se deberá entregar a través de un enlace al Google Drive de la UOC (<https://drive.google.com/>...), junto con enlace al repositorio Git entregado.

Link:

<https://drive.google.com/file/d/1RqDAMAYbifH4oiGpjD9nyb3iQgQMzZi7/view?usp=sharing>

Contribuciones	Firma Dmytro	Firma Mariona
Investigación previa		

<b>Redacción de las respuestas</b>		<p>Mariona Alberola</p> 
<b>Desarrollo del código</b>		<p>Mariona Alberola</p> 
<b>Participación en el video</b>		<p>Mariona Alberola</p> 