FACULTAD DE INGENIERÍA | UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Informe Tarea 2

Clasificación Múltiple a partir de Lenguaje Natural

Curso: Introducción a la Ciencia de Datos

Autores: Enzo Faliveni y Mikaela Lezcano

Fecha: 06/06/2023

1. INTRODUCCIÓN Y ANÁLISIS EXPLORATORIO

El presente trabajo consta de analizar algunos personajes de la base de datos relacional abierta con la obra completa de William Shakespeare y poner en práctica conceptos clave de aprendizaje automático relacionados al procesamiento de lenguaje natural. Para ello, luego de cargar los datos, lo primero a realizar es un análisis exploratorio de las distintas tablas que componen la base.

La base presenta 4 tablas: "Characters", "Paragraphs", "Chapters" y "Works". La primera contiene información relacionada a los nombres de los personajes de todas las obras y se vincula con la tabla "Paragraphs" a través de la variable "id" y "character_id" respectivamente. A su vez, esta última ("Paragraphs") presenta información sobre el texto y el personaje asociado a los párrafos de los diferentes capítulos de las obras y se relaciona con la tabla "Chapters" a través de los campos "chapter_id" y "id" respectivamente. Por su lado, "Chapters" tiene información sobre los actos de las diferentes escenas en cada una de las obras y se vincula con la tabla "Works" mediante las variables "work_id" y "id". Finalmente "Works" posee información asociada a los títulos y el género de las diferentes obras que hizo Shakespeare en su carrera.

Las dimensiones de la tabla "Characters" son 1266 filas y 4 columnas, dentro de las cuales se encontraron valores faltantes en la variable "Abbrev" (5 casos) y en el 50% de los registros en "Description" (646 casos). Por otro lado, se observa que existen 957 nombres de personajes únicos (309 casos son nombres repetidos los cuales a su vez corresponden a 125 nombres diferentes). El hecho de que existan personajes de distintas obras que comparten el mismo nombre podría significar un problema de calidad de los datos.

Dentro de la tabla "Paragraphs" se hallan 35.465 filas y 5 columnas sin datos faltantes. En la tabla "Chapters" tampoco existen valores faltantes y sus dimensiones son 945 filas y 5 columnas.

Analizando la tabla "Works", se encuentran 43 filas y 5 columnas y ningún dato faltante. Al mismo tiempo, la misma contiene 43 nombres de obras diferentes, publicadas entre 1589 y 1612. Los géneros de las mismas varían entre comedia, tragedia, historia, poema y soneto.

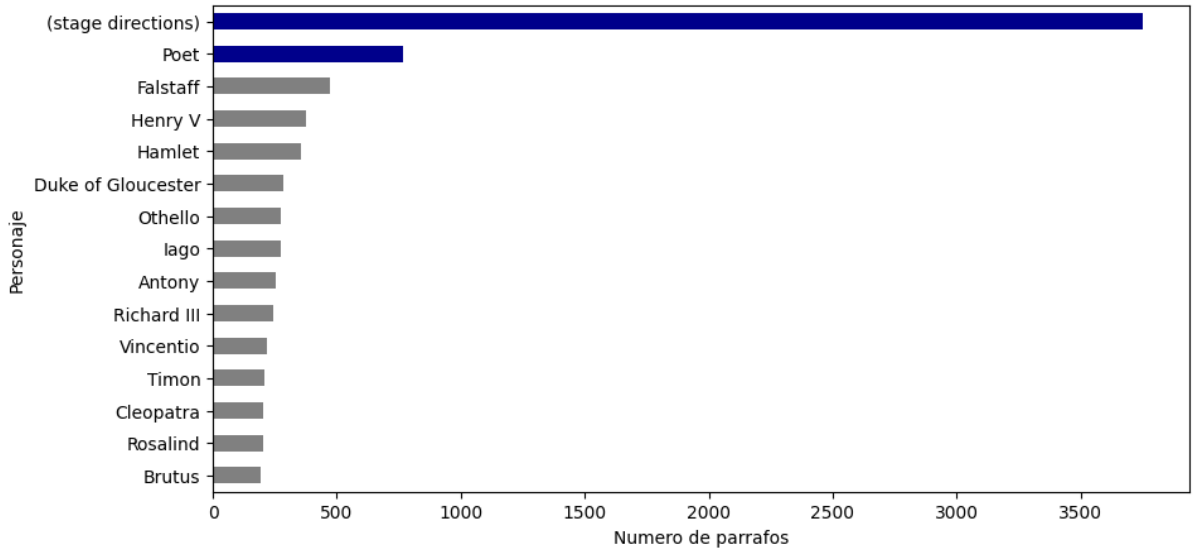
2. PÁRRAFOS POR PERSONAJE

En la actual sección se analiza y visualiza la cantidad de párrafos por personaje. Algo a tener en cuenta es que del total de 957 personajes diferentes, 925 tienen párrafos asociados, es decir que hay 32 personajes que no hablan. Como se puede ver en la Figura 1, el primer puesto lo ocupa “(stage directions)”, en español: direcciones del escenario o acotaciones, con 3.751 ocurrencias. Este término se usa en teatro y dramaturgia para describir una instrucción proporcionada por el dramaturgo o director que indica cómo se debe realizar una acción o movimiento en particular en el escenario. Las direcciones de escena generalmente se escriben en cursiva y se colocan entre corchetes o paréntesis para diferenciarlas del diálogo. Por lo tanto, se puede afirmar que no se trata de un personaje y se comprende porque ocupa el primer lugar.

En segundo lugar se observa “Poet”, el poeta (766 ocurrencias). En las obras de William Shakespeare, el personaje del poeta suele representar al propio dramaturgo. Por otro lado,

en varias de las obras de Shakespeare, hay personajes a los que se hace referencia como "poeta" o "el poeta", que a menudo sirven como observadores, comentaristas o participantes en los eventos dramáticos. Le siguen Falstaff, Henry V y Hamlet con 471, 377 y 358 ocurrencias respectivamente.

Figura 1: Cantidad de párrafos por personaje

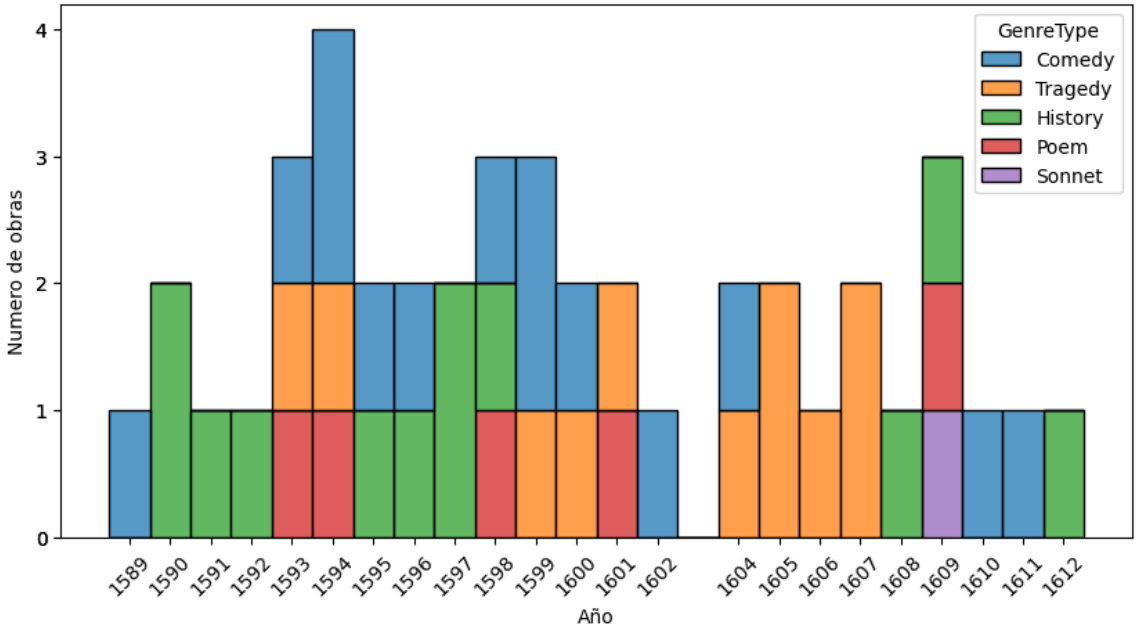


En conclusión, depende de si “poet” se considera un personaje o no, cuál es el personaje con mayor cantidad de párrafos en todas las obras de Shakespeare. En caso de considerarlo, él es el personaje con mayor cantidad de párrafos, de lo contrario es “Falstaff”.

3. OBRAS DE SHAKESPEARE EN EL TIEMPO

En esta sección se investiga cómo fue la trayectoria de trabajo de William Shakespeare a lo largo de los años. Como muestra la Figura 2, el período de mayor producción se da entre 1593 y 1601 aproximadamente. Durante el mismo priman las obras de tipo comedias (9) y en segundo lugar las tragedias e historias (5 de cada una).

Figura 2: Cantidad de obras por fecha y género



Resalta la baja presencia de poemas (5 en toda su carrera) y la producción de un único soneto en 1609 (ver Figura 3).

Figura 3: Cantidad de obras por género

GÉNERO	CANTIDAD
Comedia	14
Historia	12
Tragedia	11
Poema	5
Soneto	1

4. LIMPIEZA DE DATOS TIPO TEXTO

Para poder aplicar técnicas de Procesamiento de Lenguaje Natural (PLN o NLP en inglés) es necesario hacer una limpieza previa de los datos. La misma consta de normalizar el texto (pasarlo todo a minúsculas), eliminar todos los signos de puntuación y si el texto está en inglés, como es en este caso, expandir las contracciones. Estos pasos serán los que se llevarán a cabo en un principio.

La preparación de los datos puede ser realizada a distintos niveles de profundidad en función de los objetivos que se quieran lograr con ellos en etapas posteriores. También se pueden eliminar las “stopwords” o palabras vacías como pueden ser artículos, preposiciones, conjunciones y otros términos muy frecuentes en un idioma en particular, ya que son palabras muy comunes que no aportan un significado semántico importante para el

análisis de texto, sino que introducen ruido en el análisis. Eliminar las palabras vacías permite enfocarse en las palabras clave y mejora la eficacia computacional ya que se reduce el tamaño del conjunto de palabras y por ende disminuye la cantidad de datos a procesar, lo cual a su vez puede acelerar los algoritmos de procesamiento de texto y reducir los requisitos de memoria.

Siguiendo la idea de hacer el proceso más eficiente en términos de enfoque y computacionalmente, otra posibilidad podría ser aplicar técnicas de derivación y/o lematización.

La derivación, también conocida como stemming, consiste en eliminar los sufijos de las palabras con el objetivo de obtener su raíz. Es un enfoque basado en reglas heurísticas y no tiene en cuenta el contexto o la semántica de las palabras. Las palabras derivadas pueden no ser lexicográficamente válidas. Por ejemplo, las palabras "corriendo", "corre" y "corrió" se reducirían a la raíz "corr" mediante la derivación.

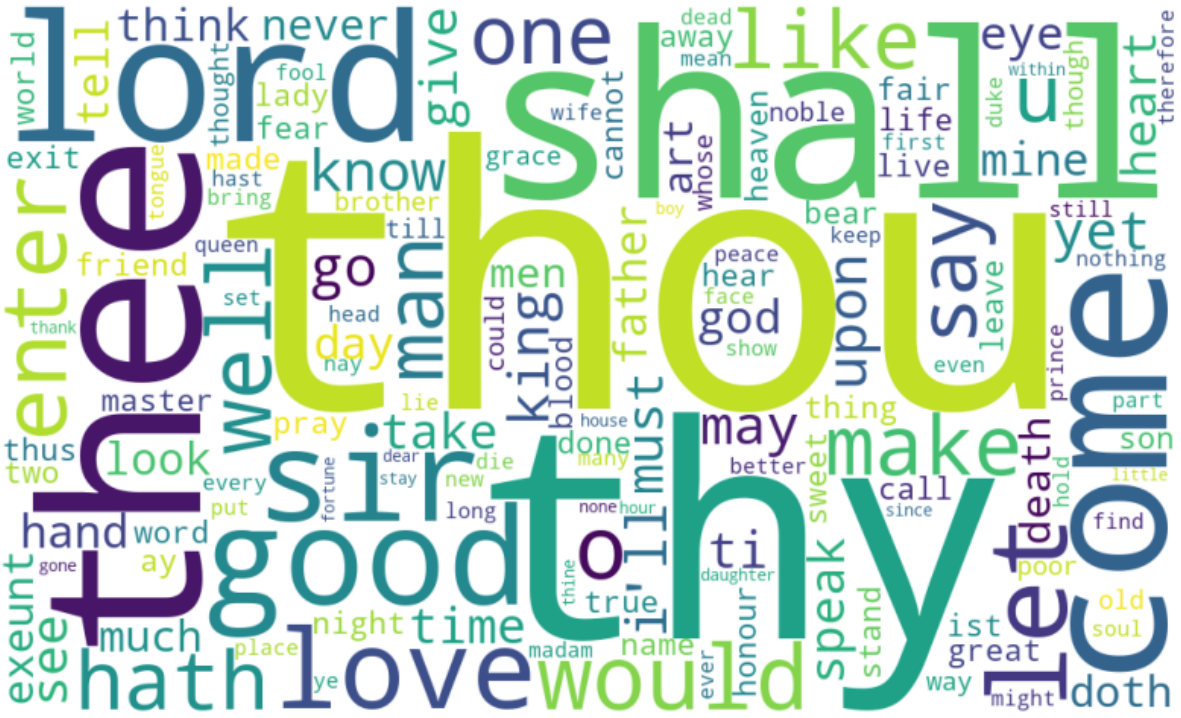
Por otro lado, la lematización implica determinar la forma canónica o lema de una palabra en función de su significado en el contexto. A diferencia de la derivación, la lematización tiene en cuenta el contexto y la categoría gramatical de las palabras. El resultado de la lematización es una palabra válida y reconocible en el idioma. Por ejemplo, las palabras "corriendo", "corre" y "corrió" se lematizarían todas a "correr".

5. ANÁLISIS DE PALABRAS

5.1. Palabras más frecuentes en el conjunto de todas las obras

A fin de investigar cuáles son las palabras que más se repiten en las obras de William Shakespeare se muestra la siguiente visualización:

Figura 4: Palabras más repetidas en todas las obras



La palabra "thou" es notoriamente la que alcanza un mayor número de repeticiones (5.768), seguida de "thy" con 4.262, "shall" con 3.721 y “thee” con 3.378.

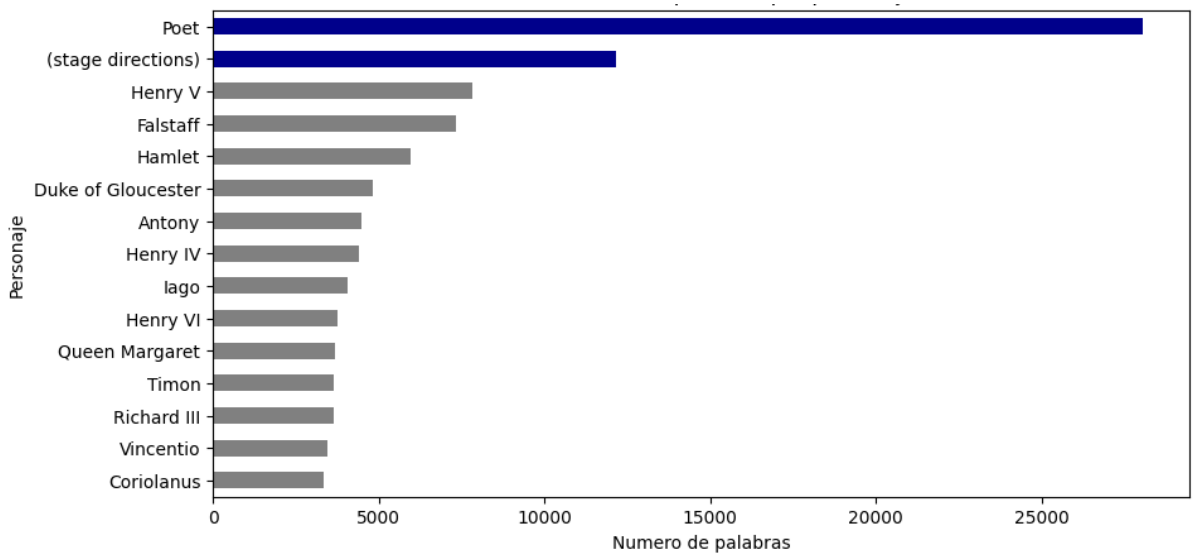
Podría ser interesante hacer este mismo análisis pero diferenciando por personaje y así conocer qué tipo de palabras priman en cada uno; lo mismo si se hace la diferenciación por género o incluso por ambas variables al mismo tiempo, es decir analizar las palabras más frecuentes de los personajes asociados a cada género. A modo de ejemplo, quizás en los personajes de las tragedias priman palabras asociadas a eventos y sentimientos negativos.

5.2. Personajes con mayor cantidad de palabras

A diferencia de lo que sucedía cuando se hizo el estudio de personajes con mayor cantidad de párrafos, en este caso se invierten los primeros dos lugares y el personaje “poet” pasa a ocupar el primer puesto y por notoria diferencia (ver Figura 5), es decir que alcanza una mayor cantidad de palabras a pesar de tener menos párrafos asociados en comparación a “(stage directions)”.

El tercer y cuarto lugar son ocupados por “Henry V” y “Falstaff” con 7.846 y 7.352 palabras respectivamente. Nuevamente se deja planteada la interrogante respecto a si “poet” se debería considerar un personaje o no.

Figura 5: Personajes con mayor cantidad de palabras en todas las obras



6. DATASET REDUCIDO DE SÓLO 3 PERSONAJES

Se trabajará con los personajes "Antony", "Cleopatra" y "Queen Margaret" y con los campos referentes al Título de la obra, el género, el texto del párrafo luego de aplicarle los procedimientos de limpieza explicados anteriormente y por supuesto el nombre del personaje. A este conjunto de datos se lo llamará Dataset reducido.

Una vez armado el dataframe con el cual trabajar, a través de un muestreo estratificado se dividen los datos en un conjunto de entrenamiento y otro de testeo, los cuales contienen el 70 y 30% del total de los datos respectivamente.

Figura 6: Párrafos por personaje

Personaje	Párrafos totales	% párrafos en data train	% párrafos en data test
Antony	253	40,41	40,42
Cleopatra	204	32,65	32,45
Queen Margaret	169	26,94	27,13

7. PROCESAMIENTO DE LENGUAJE NATURAL

7.1. Conteo de palabras o Bag of Words (BoW)

Cuando se trabaja en PLN, transformar el texto del conjunto de entrenamiento a una representación numérica es un paso clave. Una de las formas más comunes de hacerlo es utilizando la representación de conteo de palabras o bag of words (BoW).

La representación BoW implica convertir cada documento de texto (párrafo en este caso) en un vector que representa la frecuencia de aparición de las palabras en ese documento. El proceso se realiza en tres etapas:

Pre-procesamiento: consiste en dividir el texto en palabras o términos individuales, eliminando al menos los signos de puntuación y caracteres especiales, si se desea este paso se puede hacer más riguroso aplicando técnicas como las explicadas en la sección 2.

Construcción del vocabulario: se crea un vocabulario a partir de todas las palabras únicas presentes en los documentos del conjunto de entrenamiento. Cada palabra única en el vocabulario se asigna a una posición en el vector final.

Creación de los vectores BoW: para cada documento del conjunto de entrenamiento, se cuenta la frecuencia de cada palabra del vocabulario en ese documento y se almacena en el vector BoW correspondiente. Notar que no pueden haber palabras de algún documento que no estén presentes en el vocabulario pero sí pueden existir palabras que no estén presentes en el documento que se está analizando.

Una vez que se ha creado el vector BoW para cada documento del conjunto de entrenamiento, estos vectores se utilizan para entrenar un modelo de aprendizaje automático. El conjunto de prueba también se transforma pero para ello se debe utilizar el mismo vocabulario, es decir que ahora sí pueden existir palabras en algún documento que no estén presentes en el vocabulario ya que este último se creó a partir de los documentos de entrenamiento. Luego se evalúa el rendimiento del modelo entrenado.

La representación BoW es simple pero efectiva para muchas tareas de NLP e Information Retrieval. Sin embargo, no captura el orden de las palabras ni considera el contexto semántico.

Ejemplo para mejor entendimiento:

Documento/oración 1: *This is the first document.*

Documento/oración 2: *This document is the second document.*

Documento/oración 3: *And this is the third one.*

Documento/oración 4: *Is this the four document?*

Si se utilizan los primeros 3 documentos (75% del total) para entrenar y el último para testear, el vocabulario sería el siguiente: {'and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this'}. Notar que la palabra 'four' no está presente en el vocabulario por pertenecer al cuarto documento, el cual no forma parte del conjunto de entrenamiento.

BoW doc1: [0 1 1 1 0 0 1 0 1]

BoW doc2: [0 2 0 1 0 1 1 0 1]

BoW doc3: [1 0 0 1 1 0 1 1 1]

BoW doc4: [0 1 0 1 0 0 1 0 1]

El tamaño de la matriz resultante es de 4 filas correspondiente a la cantidad de documentos totales y 9 columnas reflejando la cantidad de elementos/palabras únicas en el vocabulario creado. Dado que hay una gran proporción de valores iguales a cero, se refiere a la misma como una matriz sparse y esto se debe a que la mayoría de los documentos no contienen todas las palabras del vocabulario. Esto se ve acentuado a medida que aumentamos el conjunto de datos y por ende el tamaño del vocabulario.

Al trabajar con este tipo de matrices se deben emplear técnicas particulares para realizar cálculos con las mismas. De lo contrario debido a su gran dimensionalidad, se hará un uso ineficiente del consumo de memoria.

Una variación a este procedimiento puede estar asociado al n-grama. Un n-grama es una secuencia contigua de n elementos tomados de un texto o una cadena de caracteres. El valor de n en un n-grama indica la cantidad de elementos que se consideran juntos como una unidad. Por ejemplo:

Unigramas (n = 1): Cada elemento (por lo general, una palabra) se considera por separado.

Bigramas (n = 2): Se consideran pares consecutivos de elementos.

Trigramas (n = 3): Se consideran grupos de tres elementos consecutivos.

El concepto puede ser extendido a partir de definir dos valores, los cuales indican el límite inferior y superior del rango de valores n para diferentes n-gramas de palabras o n-gramas de caracteres que se van a extraer.

Siguiendo con el ejemplo presentado anteriormente, el vocabulario ante un n-grama(1,2) sería el siguiente: {'and', 'and this', 'document', 'document is', 'first', 'first document', 'is', 'is the', 'one', 'second', 'second document', 'the', 'the first', 'the second', 'the third', 'third', 'third one', 'this', 'this document', 'this is'}. Al usar este tipo de subsecuencias, se incrementa el tamaño del vocabulario lo que provoca un aumento de la cantidad de entradas de cada vector, es decir del número de features.

El uso de n-gramas es útil para capturar patrones en secuencia de palabras. Al considerar secuencias de elementos en lugar de unidades individuales, se puede capturar la estructura y el manejo de términos compuestos o expresiones idiomáticas.

7.2. Vectores de Term Frequency - Inverse Document Frequency (TF-IDF)

El conteo de ocurrencias es un buen comienzo, pero hay un problema: los documentos más largos tendrán valores de conteo promedio más altos que los documentos más cortos, aunque puedan hablar sobre los mismos temas. Para evitar estas potenciales discrepancias, se puede utilizar la representación numérica de Term Frequency - Inverse Document Frequency (TF-IDF).

La representación TF-IDF combina dos conceptos:

Term Frequency (Frecuencia del término): consiste en dividir el número de ocurrencias de cada término o palabra en un documento entre el número total en ese mismo documento. Es decir que se mide la frecuencia con la que aparece un término específico en un documento, cuantas más veces aparezca, mayor será su importancia relativa dentro de ese documento.

Inverse Document Frequency (Frecuencia inversa del documento): consta de reducir el peso de los términos o palabras que aparecen en muchos documentos del corpus y, por lo tanto, son menos informativas que las que aparecen en menor proporción. Mide la rareza o la importancia general de un término en una colección de documentos, los términos que aparecen en pocos documentos tienen una mayor importancia relativa, ya que pueden proporcionar más información distintiva sobre esos documentos en comparación con los términos comunes que aparecen en muchos documentos.

La fórmula general para calcular el valor TF-IDF de un término o palabra en un documento es:

$$\text{TF-IDF} = \text{Term Frequency (TF)} * \text{Inverse Document Frequency (IDF)}$$

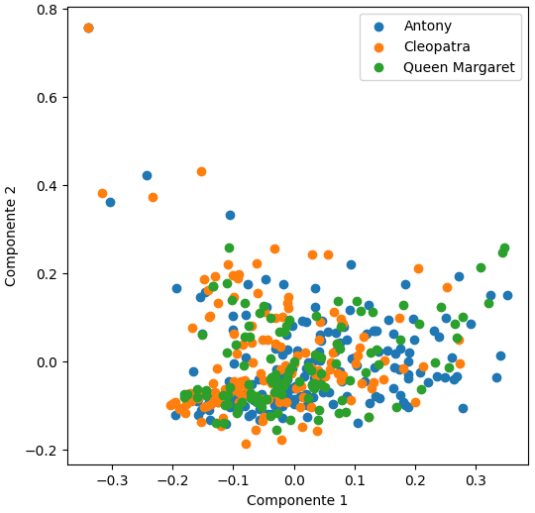
La idea general es asignar un valor numérico a cada término en un documento, que refleje tanto su frecuencia dentro del documento como su importancia relativa en la colección de documentos.

7.3. Aplicación - Análisis de Componente Principal (PCA por sus siglas en inglés) sobre los vectores de TF-IDF

Utilizando el dataset reducido y seleccionando un n-grama de (1,1), es decir seleccionando cada palabra por separado, se obtiene una matriz de entrenamiento de dimensión 438 por 2810, lo que implica que se utilizaron 438 documentos en el conjunto de entrenamiento y con los mismos se obtuvo un vocabulario con 2.810 palabras únicas.

Luego se aplicó a cada vector de la matriz la transformación TF-IDF explicada anteriormente. Recordando que cada vector posee 2.810 elementos, se decide hacer PCA a fin de reducir la dimensión de cada uno a 2. Es así que la nueva matriz de entrenamiento contiene 438 filas y 2 columnas, las resultantes de PCA.

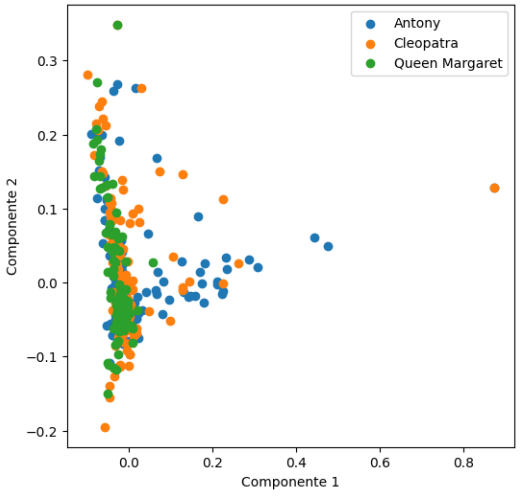
Figura 7: PCA de los vectores de tf-idf por personaje



A partir de la Figura 7 se puede afirmar que dos componentes no sirven para explicar los personajes, dado que no es posible identificar clusters bien definidos por personaje.

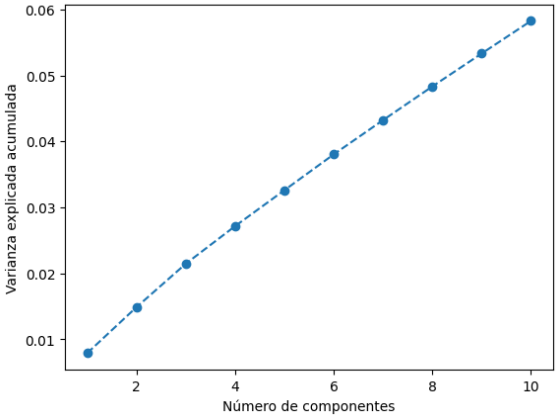
Si además se utiliza el filtrado de “stopwords” para idioma inglés y un n-grama(1,2) los resultados no difieren mucho (ver Figura 8), tampoco se logra separar los personajes utilizando sólo 2 componentes principales.

Figura 8: PCA de los vectores resultantes sin stopwords, n-grama(1,2), transformación tf-idf



A modo de entender cómo cambia la varianza explicada a medida que se agregan componentes, se muestra la siguiente figura en la cual, se observa un incremento proporcional según la cantidad de componentes. De todas formas, la varianza explicada alcanzada por 10 componentes (0,06) se considera muy baja, por lo que la cantidad de información de las variables originales que se conserva después de realizar la reducción de dimensionalidad es insuficiente para explicar los datos.

Figura 9: Varianza explicada acumulada por componentes

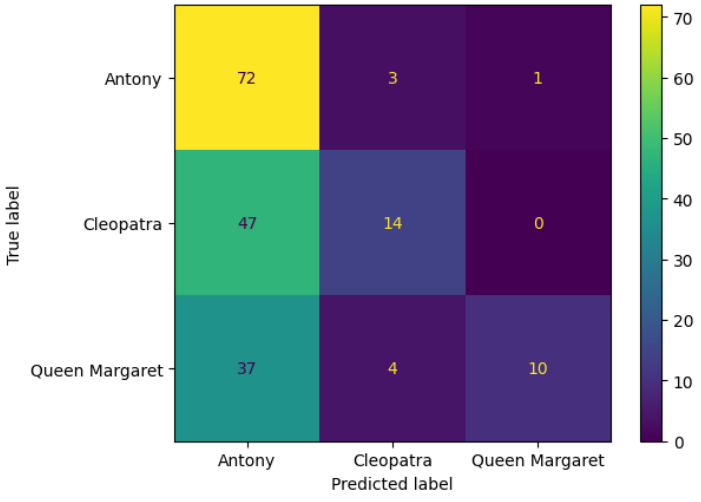


8. ENTRENAMIENTO Y EVALUACIÓN DE MODELOS

8.1. Modelo Multinomial Naive Bayes

Los métodos Naive Bayes son un conjunto de algoritmos de aprendizaje supervisado basados en la aplicación del teorema de Bayes con la suposición "ingenua" de independencia condicional entre cada par de características dado el valor de la variable de clase. En este caso, se entrena el modelo Multinomial Naive Bayes y luego se utiliza el mismo para predecir sobre el conjunto de test obteniéndose un valor de accuracy o precisión de 0,511 y la siguiente matriz de confusión.

Figura 10: Matriz de confusión a partir del modelo Multinomial Naive Bayes



Dado que la diagonal de la matriz de confusión corresponde a la cantidad de predicciones en las que el modelo acierta, se desprenden dos observaciones, el modelo tiene muy poca capacidad de detectar los personajes de Cleopatra y Queen Margaret y por ende, en la mayoría de los casos predice que se trata del personaje Antony.

Figura 11: Precisión y recall para cada personaje de la base de datos

	Precisión	Recall

Antony	0,46	0,95
Cleopatra	0,67	0,23
Queen Margaret	0,91	0,20

La precisión es una medida de qué tan preciso o exacto es un modelo al predecir la clase positiva (verdadero positivo) en comparación con todas las predicciones que hizo para esa clase (tanto verdaderos positivos como falsos positivos). Se calcula como:

Precisión = Verdaderos positivos / (Verdaderos positivos + Falsos positivos)

En resumen, la precisión mide la proporción de las predicciones positivas que son realmente correctas.

El recall, también conocido como sensibilidad, mide qué tan bien el modelo puede identificar correctamente los casos positivos en comparación con todos los casos positivos reales en los datos. Se calcula de la siguiente manera:

Recall = Verdaderos positivos / (Verdaderos positivos + Falsos negativos)

El recall representa la capacidad del modelo para encontrar y capturar todos los casos positivos existentes en los datos.

Estas dos métricas, están relacionadas ya que no se puede obtener un valor alto en ambos, se debe priorizar y lograr un equilibrio. Un modelo puede tener una alta precisión pero un recall bajo, lo que significa que es preciso en las predicciones positivas, pero puede haber muchos casos positivos que se están pasando por alto. Por otro lado, un modelo con un alto recall puede tener una precisión más baja, lo que significa que captura la mayoría de los casos positivos, pero también puede incluir falsos positivos.

Dependiendo del problema y las necesidades del contexto, la precisión y el recall pueden ser más o menos importantes. En algunos casos, es necesario alcanzar un equilibrio entre ambas métricas, mientras que en otros se puede priorizar una sobre la otra, según los requisitos específicos de la aplicación.

El hecho de evaluar el modelo solamente a través del valor de accuracy o precisión puede generar un sesgo hacia las clases dominantes. Supongamos el caso en que el conjunto de datos es desequilibrado y por lo tanto hay una clase que es mucho más frecuente que las demás, el clasificador puede tener un alto valor de accuracy simplemente prediciendo siempre la clase dominante. Esto puede llevar a una evaluación engañosa del modelo, ya que no está capturando el rendimiento real en las clases minoritarias.

8.2. Cross-validation o validación cruzada

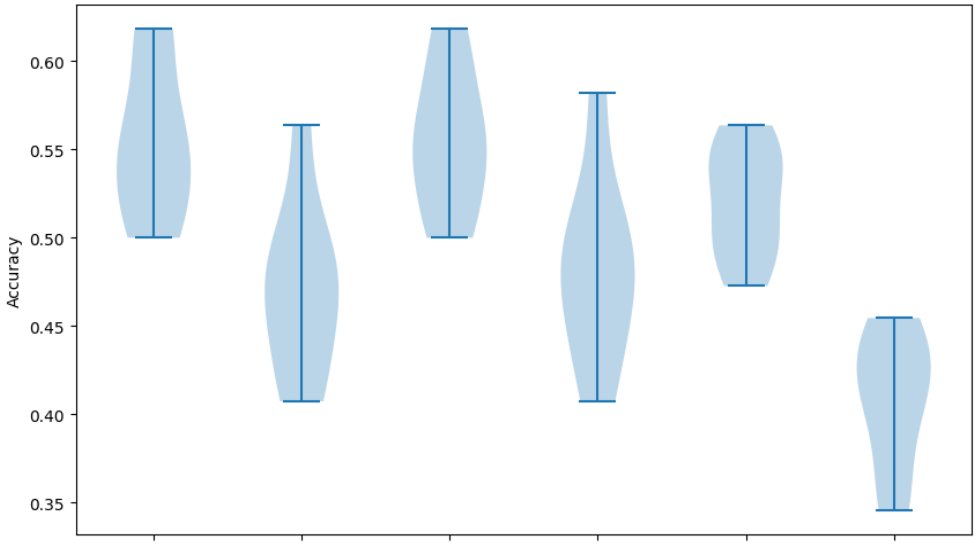
Cross-validation o validación cruzada es una estrategia comúnmente utilizada en aprendizaje automático para evaluar y hallar hiper-parámetros del modelo en cuestión. La idea consiste en dividir los datos en múltiples conjuntos de entrenamiento y prueba, realizando varias iteraciones de entrenamiento y evaluación, en lugar de tener un solo conjunto de entrenamiento y un solo conjunto de prueba.

Existen variaciones de esta técnica, como validación cruzada aleatoria, validación cruzada dejando uno fuera (leave-one-out) o de K-iteraciones. A modo de ejemplo, la validación cruzada de K iteraciones o K-fold cross-validation los datos de muestra se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante K iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado.

A continuación se muestran los resultados de implementar este método con K=8 (donde los datos de entrenamiento representan el 87,5% de los datos totales y los datos de prueba el 12,5%) y 6 configuraciones distintas de los parámetros:

1- Eliminando stopwords, n-grama(1,1) y sin transformación tf-idf.
2- Sin tratamiento de stopwords, n-grama(1,1) y sin transformación tf-idf.
3- Eliminando stopwords, n-grama(1,1) y con transformación tf-idf.
4- Sin tratamiento de stopwords, n-grama(1,1) y con transformación tf-idf.
5- Eliminando stopwords, n-grama(1,2) y con transformación tf-idf.
6- Eliminando stopwords, n-grama(2,2) y con transformación tf-idf.

Figura 12: Precisión en cada iteración de validación cruzada

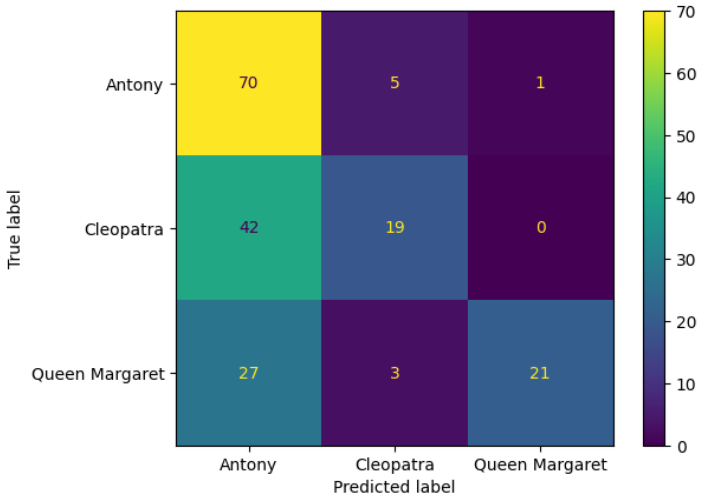


Se puede ver que sacar las stopwords y aplicar la transformación tf-idf aumenta la precisión. Por otro lado, complejizar el n-grama disminuye la precisión, por lo que la mejor definición de hiper-parámetros en términos de precisión se alcanza en la opción 1 y 3. Finalmente se opta por la 3 ya que la media de las iteraciones parece ser más alta.

8.3. Aplicación de los hiper parámetros de cross-validation

Al estimar nuevamente el modelo Multinomial Naive Bayes filtrando las stopwords, se alcanza un valor de accuracy de 0,585 y la siguiente matriz de confusión:

Figura 13: Matriz de confusión a partir del modelo Multinomial Naive Bayes sin stopwords.



Tal como era de esperar, las predicciones del modelo mejoran levemente ya que ahora presenta una mayor capacidad predictiva hacia las clases minoritarias (aumento de recall), no obstante el sesgo de predicción sobre la clase dominante sigue prevaleciendo. A su vez, se nota una mejora en la precisión de los 3 personajes (ver Figura 14).

Figura 14: Precisión y recall para cada personaje de la base de datos

	Precisión		Recall	
	Sin remover stopwords	Removiendo stopwords	Sin remover stopwords	Removiendo stopwords
Antony	0,46	0,50	0,95	0,92
Cleopatra	0,67	0,70	0,23	0,31
Queen Margaret	0,91	0,95	0,20	0,41

Utilizar un modelo basado en bag-of-words (BOW) o tf-idf (frecuencia de término - frecuencia inversa de documento) tiene varias limitaciones en cuanto al análisis de texto. A continuación, se presentan algunas de las principales.

Pérdida de información de secuencia: no tienen en cuenta el orden o la estructura de las palabras en un texto. Sólo consideran la frecuencia de ocurrencia de las palabras individuales. Esto significa que se pierde la información sobre la secuencia y la relación entre las palabras, lo que puede ser crítico en tareas como análisis de sentimiento o comprensión del lenguaje natural.

Ignorancia del significado contextual: no capturan el significado contextual de las palabras. Tratan cada palabra como una unidad independiente y no consideran el contexto en el que aparecen. Esto puede llevar a una interpretación errónea o inexacta de las palabras, especialmente en casos de palabras con múltiples significados.

Sensibilidad al ruido: consideran todas las palabras por igual y asignan un peso en función de su frecuencia. Esto puede llevar a que palabras irrelevantes o ruido léxico, como las stop words, tengan una influencia desproporcionada en la representación del texto.

Dimensionalidad alta y dispersa: pueden generar representaciones de alta dimensionalidad debido al gran número de palabras distintas presentes en un corpus de texto. Además, la matriz resultante suele ser dispersa, ya que muchos documentos sólo contienen una fracción de todas las palabras posibles.

8.4. Modelo Random Forest o Bosques aleatorios

La metodología Random Forest o bosques aleatorios es un algoritmo de aprendizaje automático que combina múltiples árboles de decisión para realizar tareas de clasificación o regresión.

Para ello se crea una colección de árboles de decisión independientes donde cada árbol se entrena en una muestra aleatoria (con reemplazo) de los datos de entrenamiento. Esta selección aleatoria garantiza la diversidad entre los árboles y evita el sobreajuste (overfitting).

El proceso se lleva a cabo dividiendo los nodos hasta que se alcanza un criterio de parada, como una profundidad máxima o un número mínimo de muestras en un nodo. Una vez que los árboles de decisión están entrenados, se realiza una predicción combinada. Para la clasificación, cada árbol emite una votación y la clase con más votos se selecciona como la predicción final. En el caso de la regresión, se toma el promedio de las predicciones de todos los árboles.

Al utilizar esta metodología se obtuvo un accuracy de 0,559 y la siguiente matriz de confusión junto con las correspondientes medidas de precisión y recall:

Figura 15: Matriz de confusión a partir del modelo Random Forest

Figura 16: Precisión y recall para cada personaje de la base de datos

	Precisión			Recall		
	Sin Remover stopwords	Removiendo stopwords	Random Forest	Sin Remover stopwords	Removiendo stopwords	Random Forest
Antony	0,46	0,50	0,50	0,95	0,92	0,88
Cleopatra	0,67	0,70	0,60	0,23	0,31	0,34
Queen Margaret	0,91	0,95	0,89	0,20	0,41	0,33

Comparando las matrices de confusión, se puede afirmar que los resultados son muy similares y en términos de precisión y recall no se presentan grandes variaciones.

Esto es un indicador de que para mejorar las métricas analizadas se debe de poner un mayor foco en la representación y la forma de uso de los datos y no en la optimización y explotación de nuevos modelos. (ver sección 5.3. sobre las limitaciones relacionadas al análisis de texto utilizado).

9. MODIFICACIÓN DEL DATASET REDUCIDO

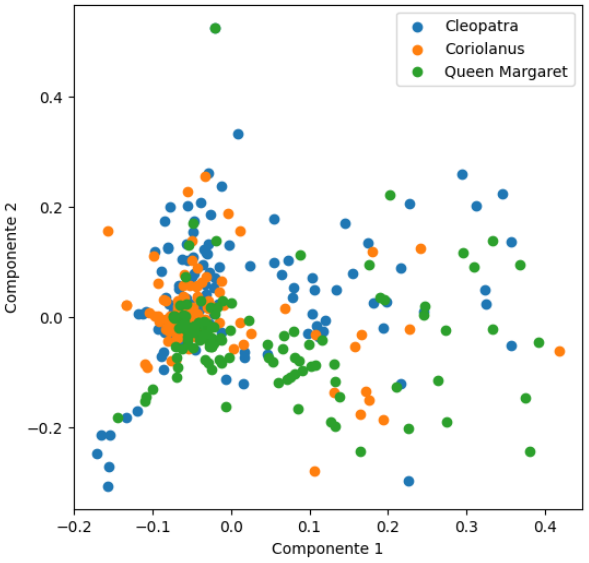
A fin de investigar qué sucedería si se mejora el desbalance de los datos, se decide cambiar uno de los tres personajes seleccionados por otro, más específicamente a Antony ya que es el que contiene la mayor cantidad de párrafos y además el personaje que alcanza una mayor proporción de predicciones positivas que son realmente correctas (recall). En su lugar se elige a Coriolanus, de esta forma la nueva distribución de párrafos es la siguiente:

Figura 17: Párrafos por personaje en el nuevo dataset

Personaje	Párrafos totales	% del total
Coriolanus	189	33,63
Cleopatra	204	36,30
Queen Margaret	169	30,07

Se utilizó la misma configuración de parámetros detallados anteriormente (filtrando las stopwords, n-grama(1,1) y transformación tf-idf) y los resultados preliminares obtenidos fueron muy similares. Se muestra únicamente el resultado asociado a la reducción de dimensionalidad ya que el mismo refleja que nuevamente no es posible diferenciar los personajes utilizando sólo dos componentes principales y da un indicio de que los resultados mediante los modelos estudiados con el nuevo personaje no serán muy diferentes a los obtenidos anteriormente en presencia de Antony.

Figura 18: PCA de los vectores de tf-idf por personaje



El personaje Coriolanus fue seleccionado para alcanzar un mejor balance en los datos, ya que tiene una cantidad de párrafos similar a los otros personajes, por lo que no es necesario realizar sobre-muestreo y submuestreo.

Es importante destacar que si se tiene un dataset desbalanceado, el modelo puede presentar un sesgo hacia la clase mayoritaria (únicamente clasificando esta de forma correcta) y clasificar incorrectamente la clase minoritaria. Se debe exigir al modelo una alto nivel de precisión para poder clasificar correctamente la clase minoritaria, esto resulta una tarea difícil cuando la clase minoritaria tiene a su vez una cantidad de datos reducida.

Por otro lado, el dataset original ya se encontraba medianamente balanceado, por lo que no es necesario realizar sobre-muestreo o submuestreo, es decir se encuentra suficientemente equilibrado como para generar un modelo que pueda clasificar correctamente a los personajes. El problema de clasificar los personajes mediante frecuencias de palabras es dificultoso y engañoso, ya que las palabras más utilizadas parecen no variar mucho entre los personajes estudiados.

La representación del lenguaje natural adoptada no permite reconocer sentimientos o emociones en los diálogos de los personajes. Una posible solución al problema sin alterar la representación de los datos es seleccionar un personaje que tenga diálogos más distintivos, o utilice con frecuencia palabras que no se encuentren en los diálogos de los demás personajes.

10. LANGUAGE MODEL

Una alternativa interesante a partir de la base de datos disponible para extraer features de texto, consiste en utilizar los personajes como etiquetas categóricas y el texto procesado, para determinar cuál es el personaje más probable que haya pronunciado un texto dado. La propuesta se fundamenta en el uso de técnicas más avanzadas de PLN. Es relevante destacar que existen múltiples alternativas para alcanzar este objetivo. A modo de ejemplo, se plantea un enfoque específico que involucra el preprocesamiento del texto previamente etiquetado. A través de la tokenización del texto utilizando el tamaño total del vocabulario y

estrategias adecuadas de “padding”, se generan vectores (secuencias de tokens) de longitud fija, con el objetivo de ser utilizados posteriormente como insumos del modelo.

Luego, se procede a la creación del conjunto de entrenamiento y prueba. Es importante mencionar que el problema se tratará como un problema de clasificación múltiple, por lo tanto, se realiza la codificación one-hot en las etiquetas. Esto implica que el modelo generará un vector de probabilidades cuya longitud coincide con la cantidad de categorías, es decir, los personajes considerados en el estudio.

Por último, se procede con la implementación del modelo. Este paso puede ser realizado de diversas formas, pero como propuesta concreta, es posible entrenar internamente los embeddings y utilizar capas de Long Short Term Memory (LSTM) en forma secuencial. No obstante, también se considera como opción válida el uso de embeddings pre-entrenados en un conjunto de datos más amplio y variado.

11. MODELO DE FASTTEXT

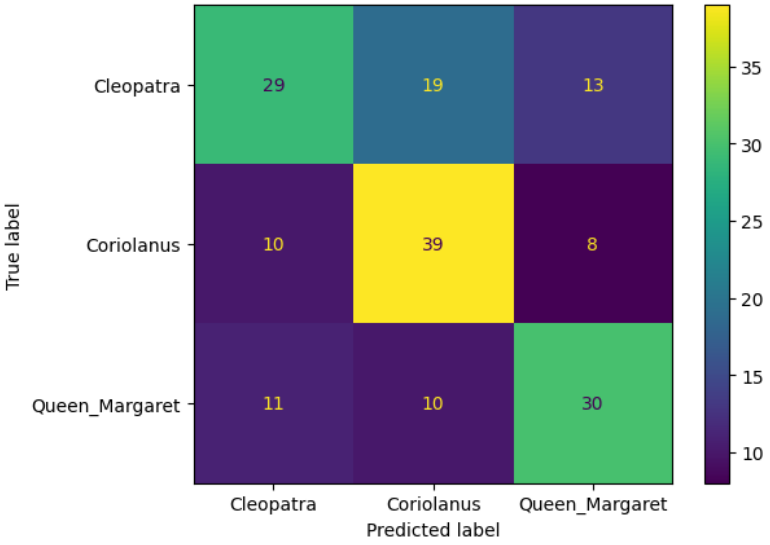
Finalmente, se evalúa el problema utilizando un modelo adicional, se entrena el modelo de FastText desarrollado por Facebook AI Research.

FastText se basa en la idea de representar palabras como secuencias de n-gramas (subsecuencias de caracteres) y utiliza estas representaciones para entrenar modelos de lenguaje y realizar tareas de clasificación de textos. Es decir que en lugar de representar las palabras como unidades indivisibles, descompone las palabras en subsecuencias más pequeñas (n-gramas). Esto permite capturar información tanto de las palabras completas como de las partes más pequeñas que las componen. Por ejemplo, la palabra "uno" se descompone en los n-gramas "u", "n", "o".

En el proceso de entrenamiento de FastText, se generan embeddings tanto para las palabras como para los n-gramas. Cada palabra y n-grama tiene su propio embedding asociado. Estos capturan información semántica y sintáctica de las palabras y sus partes constituyentes. El proceso es el siguiente, una vez que se han obtenido los embeddings de los n-gramas de una palabra, se promedian para obtener así una representación global de la palabra, este luego se utiliza como entrada para las capas ocultas y la capa de salida del modelo.

Una vez entrenado este modelo para el dataset reducido modificado, se obtiene un valor de accuracy o precisión de 0,580 y la siguiente matriz de confusión.

Figura 19: Matriz de confusión a partir del modelo FastText



La implementación del modelo conlleva una leve mejora en las métricas planteadas (ver Figura 20), no obstante los valores siguen siendo muy similares a los presentados con anterioridad, considerando a su vez la modificación del dataset.

Figura 20: Precisión y recall para cada personaje de la base de datos

	Precisión	Recall
Cleopatra	0,58	0,48
Coriolanus	0,57	0,68
Queen Margaret	0,59	0,59

12. COMENTARIOS Y REFLEXIONES FINALES

Tras llevar a cabo el análisis del presente informe, se puede afirmar que se ha logrado realizar una evaluación precisa de la calidad de los datos disponibles, así como interpretar su naturaleza y distribución mediante la generación de visualizaciones comprensibles. A su vez, se ha podido explorar exitosamente técnicas de aprendizaje automático con el fin de generar detección de personajes a partir de texto.

Sin embargo, la variedad de modelos explorados en este estudio, permite afirmar que para lograr una mejora considerable en las métricas el foco debería estar en cambiar la forma en que se representa el texto. Como fue dicho anteriormente, bajo la representación actual, no se es capaz de reconocer sentimientos y expresiones complejas en los diálogos de los personajes. Por lo que no es cuestión de optimizar el modelo o plantear arquitecturas más complejas, sino de usar los datos de forma tal que se pueda contemplar dicha falta de información.

13. REFERENCIAS BIBLIOGRÁFICAS

- Text Normalization for Natural Language Processing (NLP)
- Multiclass Text Classification Using Deep Learning
- Stemming
- Lemmatisation
- Bag of Words Model
- Working With Text Data
- Text feature extraction
- TF-IDF
- Naive Bayes
- Cross-validation
- Random Forest
- FastText