Tarea 2

Introducción a la Ciencia de Datos
Grupo 16

María Luciana Martínez
4.421798-2
luchymd89@gmail.com

Lucía Lemes
5.127.425-6
lucia.lemes@fing.edu.uy

Introducción

En el presente trabajo se utilizó la colección completa de trabajos de William Shakespeare, recopilada sobre la base de datos relacional “Open Source Shakespeare” (OSS) [1]. La misma contiene información de cada título escrito por Shakespeare, su género y año de publicación. También, se tiene el detalle de los personajes, capítulos y escenas que componen la obra, así como los diálogos (cada párrafo y sus contenidos) y direcciones de escena.
El objetivo del trabajo fue estudiar y limpiar los datos de las obras de William Shakespeare, más específicamente los diálogos, así como aplicar métodos de procesamiento de lenguaje y modelos de aprendizaje supervisado, con el objetivo de identificar a qué personaje pertenecían ciertos diálogos, e interpretar los resultados obtenidos.
Los datos con los que se trabaja corresponden a la colección de obras de Shakespeare, modelada mediante las tablas personajes, párrafos y capítulos. Los detalles del esquema de la base de datos con las particularidades de cada tabla se encuentran en el Anexo 1.

Métodos

Se trabajó sobre un cuaderno de Jupyter Notebook [2] para descargar las tablas de la base de datos, aplicar transformaciones sobre las mismas, procesar el texto, entrenar modelos y extraer visualizaciones de los resultados. El mismo se encuentra disponible como parte de un repositorio de GitLab [3].
Como un primer paso los datos fueron transformados en dataframes con el fin de poder manipularlos, se realizó una exploración y posterior limpieza de los mismos, abarcando campos faltantes, normalización de textos, extracción de símbolos, signos de puntuación, abreviaciones y contracciones. En el Anexo 2 se puede ver en detalle las decisiones tomadas.

Extracción de características

Para el procesamiento de los diálogos y la posterior aplicación de métodos de aprendizaje supervisado, se seleccionó un subconjunto de diálogos perteneciente a tres personajes (Antony, Queen Margaret y Cleopatra). Posteriormente, se realizó un muestreo estratificado mediante la función *train_test_split* del paquete *sklearn.model_selection*, utilizando 30% de los datos como conjunto de testeo (*train*) y 70% para entrenamiento (*test*), con una semilla

(*random_state*) de valor 1. Esto garantizó que la proporción de cada personaje en los conjuntos de train y test fuera similar a la del conjunto inicial.
Para la extracción de características, se utilizó el modelo de bolsa de palabras (*bag of words*), que realizaba una simplificación del texto mediante la cuantificación de ocurrencias de cada palabra en el mismo, pero descartando la información de la posición relativa de la palabra así como de la gramática utilizada [4]. El modelo asigna un número identificador a cada palabra que aparezca en los textos y para cada línea de texto (*i*) cuenta la cantidad de ocurrencias (*n*) de cada palabra (*j*) y la guarda en una matriz *X[i,j] = n.*
Dado que las oraciones o diálogos no suelen contener el conjunto entero de palabras, el resultado de esta codificación era una matriz esparza (a las palabras sin usar se les asigna un cero). Las matrices esparza en scipy guardan únicamente los datos con valores distintos de cero, logrando reducir la cantidad de memoria utilizada.
El funcionamiento de este modelo puede apreciarse con un ejemplo de tres oraciones: “Esta es la primera oración del ejemplo”, “Esta es la la segunda”, “¿Es esta la primera oración del ejemplo?”, que da como resultado la matriz de *bag of words* de la Tabla 1*,* donde cada columna se corresponde con la aparición de una palabra. Como puede verse en ella, la primera y tercera oración son indistinguibles entre sí.

Tabla 1: Ejemplo de la salida de un modelo de bolsa de palabras para tres oraciones (“Esta es la primera oración del ejemplo”, “Esta es la la segunda”, “¿Es esta la primera oración del ejemplo?”). Las palabras se ordenan de forma alfabética.

del	ejemplo	es	esta	la	oración	primera	segunda
1	1	1	1	1	1	1	0
0	0	1	1	2	0	0	1
1	1	1	1	1	1	1	0

Estudiar las secuencias de palabras puede dar mucha información acerca de un texto. Estas secuencias lineales son conocidas como n-gramas (*n-gram*); un n-grama es un conjunto de *n* elementos consecutivos en un documento de texto. Se puede crear un modelo lingüístico que incorpore n-gramas contando la cantidad de veces que cada n-grama único aparece en un documento. Esto se conoce como modelo de bolsa de n-gramas. Al aumentar el rango del n-grama de una palabra a dos, por ejemplo, se incluirán columnas con combinaciones de dos palabras [4]. Continuando sobre el ejemplo de la Tabla 1, sería el equivalente a incluir las columnas “esta es”, “es esta”, “es la”, “la la”, “la segunda”, “la primera”, “primera oración”, “oración del”, “del ejemplo”, que para el ejemplo anterior permitiría diferencias entre sí las tres oraciones.
Una alternativa al n-grama es el uso de técnicas de word embedding, como Word2Vec. Word embedding trata de representar el significado de las palabras con vectores, usando los contextos de las palabras en el texto. Esta técnica es capaz de capturar el contexto, la similitud semántica y sintáctica (género, sinónimos, etc.) de una palabra reduciendo el tamaño. La idea es que estén colocados en el espacio vectorial de forma que las palabras que comparten contextos comunes en el corpus están localizadas cerca unas de otras en el espacio. Por ejemplo, cabría esperar que las palabras “*perro*” y “*gato*” estuvieran representadas por vectores relativamente cercanos en el espacio vectorial donde se definen esos vectores. Un modelo de este tipo suele dar mejores resultados que los enfoques

tradicionales, también permite reducir el tamaño del problema y, por tanto, la tarea de aprendizaje.
Retomando el modelo de bolsa de palabras, contar la cantidad de ocurrencias de una palabra puede generar problemas en textos largos donde los promedios pueden llegar a ser más grandes que para textos más chicos relacionados a los mismos temas. Una forma de mitigar este problema es dividir el número de ocurrencias de cada palabra en el texto sobre el total de palabras, esto se denomina *tf* o *Term Frequencies*. Para este trabajo se empleó la implementación dada por *TfidfTransformer* en el paquete *sklearn.feature_extraction.text*, normalizando la cantidad de ocurrencias a partir de la norma L2, asignada por defecto.
Adicionalmente, también se puede reducir el peso de las palabras que aparecen repetidas en muchos de los textos y por lo tanto son menos informativas que aquellas con menor frecuencia de aparición. Esta reducción de escala se conoce como *tf-idf* o *Frequency times Inverse Document Frequency*, se calcula multiplicando el resultado de *tf* por la frecuencia inversa del documento, calculada como el logaritmo de la relación entre la cantidad de documentos y el número de documentos donde aparece cada término. Este paso adicional se logró indicando *use_idf=True* en la implementación mencionada.
Una vez extraídas las características, resultó de interés aplicar un método de reducción de características, tal como el análisis de componentes principales (PCA).
PCA es un algoritmo de aprendizaje automático no supervisado que intenta reducir el número de características (la dimensión) del conjunto de datos tratando de mantener la mayor cantidad de información posible. Busca un nuevo conjunto (componentes) que componen las características originales correlativas entre sí. Las primeras componentes son las que tienen la mayor varianza. El método de *PCA* permite por lo tanto “condensar” la información aportada por múltiples variables en solo unas pocas componentes [5].
Se utilizó el objeto *PCA* de la librería *sklearn.decomposition*. Se observó el poder de representación de las primeras dos componentes, para luego evaluar la varianza explicada al tener 10 componentes. A modo de comparación, se variaron las características previas en cada grupo: aplicación de bag of words con y sin stopwords (incluyendo palabras del inglés moderno temprano), distintos n-gramas y parámetros de tf-idf; con el objetivo de diferenciar si alguno era mejor que el otro.

Modelos de aprendizaje supervisado

Dado que se deseaba utilizar las características determinadas antes para entrenar modelos que clasifiquen cada diálogo de acuerdo con el personaje que lo dice, se trabajó con cuatro modelos:

1. Multinomial Naive Bayes: implementado a partir de *MultinomialNB* del paquete *sklearn.naive_bayes*.
2. KNN: implementado mediante *KNeighborsClassifier* del paquete *sklearn.neighbors.*
3. SVM: implementado mediante *SGDClassifier* del paquete *sklearn.liinear_model*, que implementa un SVM que utiliza el método del gradiente descendiente estocástico.
4. Fastext.

Multinomial Naive Bayes es un clasificador adecuado para clasificar conjuntos con características discretas, como ser conteo de palabras en clasificación de textos; se basa en la probabilidad condicional (probabilidad de un evento dada la ocurrencia de otro) y el teorema de Bayes. Naive Bayes multinomial se basa en asumir una distribución Multinomial (la probabilidad de cada resultado es independiente y su suma siempre es uno). Algunas de las ventajas de este modelo es que es fácil de entrenar, de interpretar y trabaja bien para

conjuntos de datos con muchas variables. Como contra se da el tener que asumir que las variables son independientes, algo que pocas veces se da.
Se probó también el clasificador de K vecinos más cercanos (K-nearest neighbor, KNN); este clasificador utiliza la proximidad para hacer clasificaciones o predicciones, partiendo de la suposición de que se pueden encontrar puntos similares cerca uno del otro. El valor k en el algoritmo define cuántos vecinos se verificarán para determinar la clasificación de un punto específico [6].
El tercer clasificador utilizado fue SVM (Support Vector Machines) [7]. El SVM es un clasificador lineal, representa los puntos de una muestra en el espacio y separa las clases en dos espacios lo más amplios posibles, mediante un hiperplano de separación que se define por los dos puntos de las 2 clases más cercanas (vector de soporte). Al utilizar una nueva muestra sobre este modelo, se puede saber a qué clase pertenece según si queda clasificada en una u otra parte del hiperplano. La ventaja de SVM es que son efectivos para conjuntos con gran cantidad de dimensiones, así como para conjuntos donde la cantidad de dimensiones es mayor que la cantidad de ejemplos.
Por último se probó entrenar el modelo FastText con el fin de comparar resultados. FastText es un modelo que sirve para realizar word embedding y clasificación, se obtienen representaciones vectoriales de palabras [8]. A diferencia de Word2Vec, que trabaja a nivel de palabras, FastText trata de capturar la información morfológica de las palabras, trata a cada palabra como un compuesto de n-grams de las letras que las conforman y en skipgrams de palabras para detectar el contexto en que se usan; con esto logra capturar más información sobre la función de las palabras dentro de sus entornos. Otra de las ventajas de FastText es que puede manejar palabras nuevas, que no estaban en los datos de entrenamiento, ya que está basado en los caracteres y no en las palabras completas. Por lo tanto FastText resulta bastante sensible a la similitud de las palabras y también a la manera en que se usan en sus contextos.
Para el primer clasificador, se evaluó su desempeño al utilizar los datos de train y test sin mayores procesamientos (incluye stopwords, utiliza unigrama y TF), y se lo comparó con el desempeño de los demás modelos MNB entrenados variando las condiciones bajo las que se extrajeron las características (stopwords, n-gramas y normalizaciones TF-IDF). Se probaron en total ocho combinaciones de parámetros y se calcularon las métricas de exactitud (*accuracy*), precisión (*precision*) y sensibilidad (*recall*) para varios entrenamientos mediante validación cruzada (*cross validation*) con cuatro subconjuntos (*folds*). La misma consiste en separar el conjunto de entrenamiento en *k* conjuntos más pequeños, entrenar en *k-1* conjuntos y validar (evaluar) sobre el restante; luego, repetir este entrenamiento y evaluación variando en cada iteración sobre qué subconjunto se valida [9]. Al final de la ejecución, se tienen un total de *k* realizaciones de cada métrica, y se considera que el desempeño global del modelo está relacionado con los valores medios de cada métrica.
Finalmente, se observó los resultados de cada clasificación a partir de las matrices de confusión devueltas. Las métricas de exactitud, precisión y sensibilidad se calculan como en las Ecuaciones 1, 2 y 3 respectivamente, donde VP son los verdaderos positivos, FP los falsos positivos, VN los verdaderos negativos y FN los falsos negativos.

$$exactitud = \frac{(VP+VN)}{Total} \quad (1)$$

$$precisión = \frac{VP}{(VP+FP)} \quad (2)$$

$$sensibilidad = \frac{VP}{(VP+FN)} \quad (3)$$

Con la métrica de precisión se pudo medir la calidad del modelo en tareas de clasificación. La misma se calculó como las predicciones acertadas para cada personaje sobre el total de predicciones para cada personaje. Si la matriz de confusión era C, la precisión para el grupo de la columna i se podía calcular como $C[i,i]/\sum C[j,i]\ \forall j$. La sensibilidad, por otro lado, se midió como el valor de las predicciones acertadas para cada personaje sobre el total de valores de cada personaje en el conjunto de testeo. Si la matriz de confusión era C, la sensibilidad para el personaje *i* se podía calcular como $C[i,i]/\sum C[i,j]\forall j$.
Luego de calculadas las métricas, se eligió el mejor parámetro para la extracción de características y se comparó con éste el desempeño de los tres clasificadores mencionados antes.

Resultados y discusiones

Observación de las particularidades del conjunto de datos

Trabajando con el texto limpio (sin símbolos), se realizaron ciertas observaciones sobre el mismo: por ejemplo, la relación entre personajes y palabras asignadas, la cantidad de diálogos por personaje así como los personajes más importantes de cada obra. Adicionalmente, se observó la distribución temporal de la obra de William Shakespeare.
Los trabajos de este autor contienen 27083 palabras distintas. En la Figura 1 se muestran las 25 palabras con mayor cantidad de apariciones sobre el total de las obras. Entre las palabras más frecuentes se distingue una alta prevalencia de monosílabos, pronombres y verbos; a los cuales puede denominarse en conjunto como stopwords. Es interesante, por fuera de las stopwords, la alta prevalencia de las palabras lord y sir, que dan pauta del alto protagonismo que la aristocracia tiene en la obra de Shakespeare.

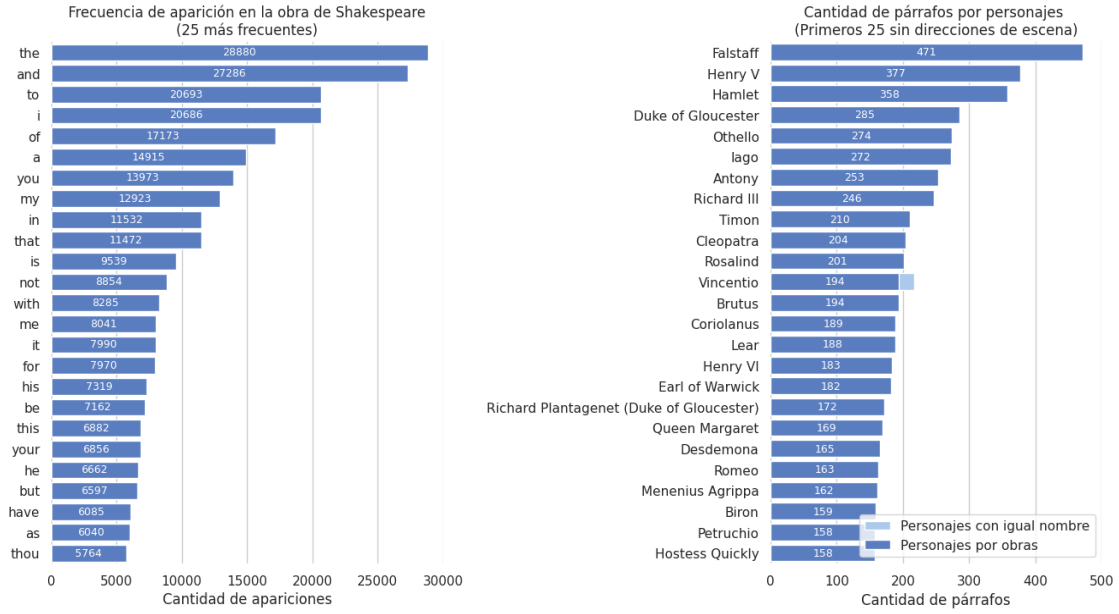


Figura 1: En la imagen de la izquierda puede observarse un gráfico de barras comparando la cantidad de veces que aparecen las 25 palabras más frecuentes (el largo de la barra se indica en el centro) en la obra de William Shakespeare. En la imagen de la derecha se observa un gráfico de barras comparativo de la cantidad de diálogos (párrafos) dichos por los 25 personajes con más intervenciones en la obra de William Shakespeare. En

azul oscuro se muestra la cantidad de párrafos dichos por un único personaje cuyo nombre es igual al que se indica a la izquierda (el largo de la barra se escribe a la derecha de la misma), mientras que en celeste claro la cantidad de párrafos asignados a todos los personajes que comparten dicho nombre (a modo de comparación).

Al observar cuáles fueron los personajes con más diálogo en el total de la obra, así como en cada publicación, se obtienen los resultados de la Figura 1.
En el Anexo 3 se detallan los resultados obtenidos en las diferentes agrupaciones.
Una forma de ver gráficamente (Figura 1) la relevancia de cada personaje es mostrando la cantidad de párrafos agrupados por personaje, pero aplicando las barras según si son párrafos de obras distintas. Esto permite distinguir el aporte a cada obra en particular.
Por último, la producción artística de William Shakespeare, visible en la Figura 2, osciló entre una y dos obras por año, desde que comenzó a publicar en 1589 hasta su última obra en 1612. Respecto a la producción de trabajos de acuerdo al género literario, parecería haber una mayor concentración de obras de historia y comedia al principio de su creación, con mayor abundancia de tragedias hacia el final. Sin embargo, estas tendencias no estaban claramente delineadas (no hay un quiebre entre una y otra). En el Anexo 4 se puede apreciar en más detalle la cronología de las obras por género.

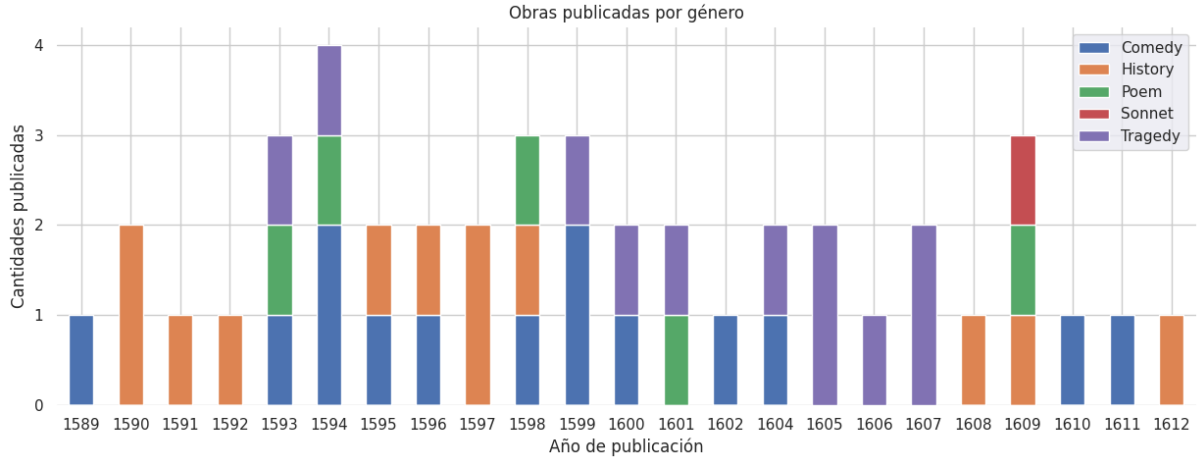


Figura 2: Gráfico de barras comparativo de la cantidad de obras publicadas por William Shakespeare entre 1589 y 1612 según el género literario al que pertenecen.

Clasificación de personajes a partir de párrafos

Partiendo del conjunto de datos limpio, se lo redujo a tres personajes: Antony, Cleopatra y Queen Margaret. El nuevo DataFrame contenía el texto de cada párrafo previamente limpiado, los nombres de personajes, título de la obra y género de la misma. Del mismo se recabaron un total de 626 datos, de los cuales 253 correspondían al personaje Antony, 204 a Cleopatra y 169 a Queen Margaret. Estos datos se emplearon para entrenar y evaluar distintos clasificadores mediante aprendizaje supervisado.

Definición de conjuntos de entrenamiento y test

Se generaron dos conjuntos a partir de los datos, utilizando muestreo estratificado tal como se describió en Métodos, con el fin de asegurar que se mantengan las proporciones en los datos relativos a cada personaje. Se obtuvo un conjunto de entrenamiento, con 70% del total de datos (438), el cual se utilizó para entrenar los diferentes modelos de aprendizaje supervisado, y un conjunto de test, con el 30% restante (188), utilizado para evaluar el

rendimiento de dichos modelos, las cantidades de párrafos y porcentajes resultantes para cada personaje pueden verse en la Tabla 2, o gráficamente en la Figura 3.
Para cada conjunto resultante de la división, se seleccionó como valor de entrada ***X*** el texto correspondiente a los párrafos y como valor de salida ***y*** el nombre del personaje que habla.

Tabla 2: En la tabla se recopilan la cantidad de párrafos asignada a un personaje para cada uno de los tres conjuntos de datos (conjunto madre o total, conjunto de entrenamiento o train y conjunto de evaluación o test). Adicionalmente, al lado de cada recuento de párrafos se especifica el porcentaje que dicho número representa sobre el total del conjunto.

Personajes	Total		Conjunto Train		Conjunto Test	
	Cantidad	Porcentaje	Cantidad	Porcentaje	Cantidad	Porcentaje
Cleopatra	204	32.59%	143	32.65%	61	32.45%
Antony	253	40.42%	177	40.41%	76	40.43%
Queen Margaret	169	27.00%	118	26.94%	51	27.13%

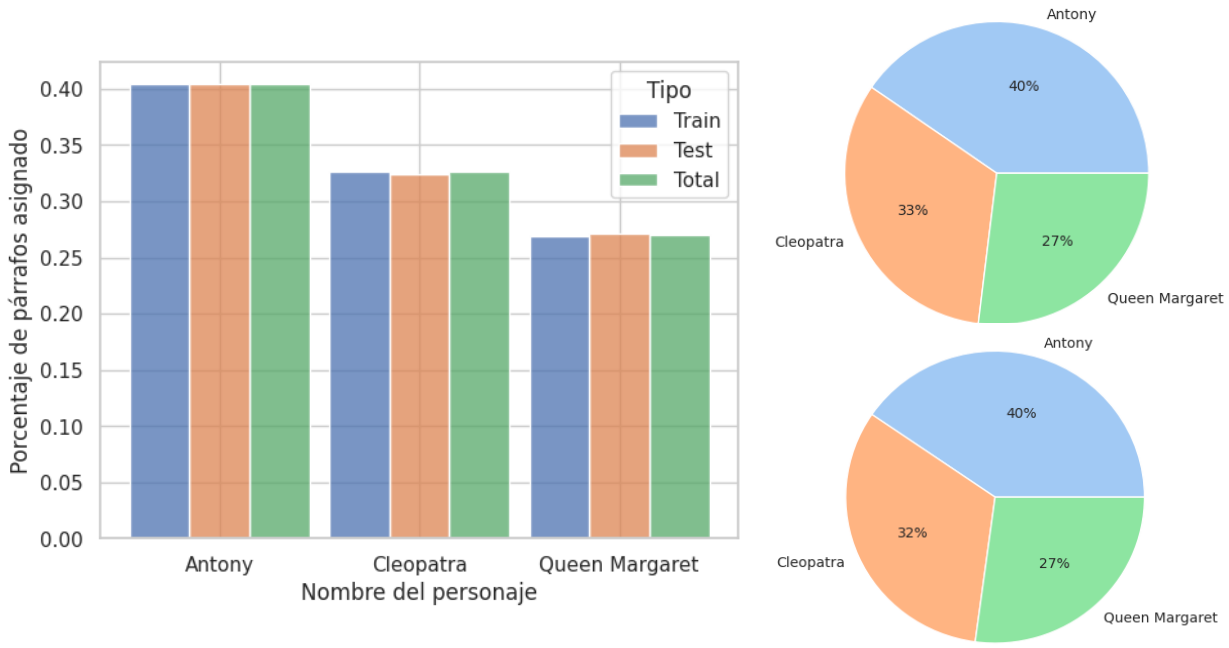


Figura 3: Distribución de los diálogos de cada personaje en los distintos conjuntos definidos. En la gráfica de barras de la izquierda se puede observar para cada personaje qué porcentaje de cada conjunto ocupan sus diálogos. Mientras que a la derecha, en el gráfico circular de arriba se representan estas distribuciones como áreas para el conjunto de entrenamiento (*train*) y en el gráfico de barras de abajo para el conjunto de evaluación (*test*).

El resultado obtenido fue acorde al esperado, las diferencias mínimas observadas en los porcentajes para un mismo personaje derivaron de que se estaba dividiendo un conjunto con valores discretos, por lo que era imposible alcanzar exactamente la misma proporción en train y test. Algo interesante a señalar fue el desbalance existente entre la cantidad de diálogos para cada personaje, estando Antony más representado que los demás y Queen Margaret menos representada (casi la mitad de datos que Antony), lo que pudo provocar la

aparición de sesgos en la clasificación (en el caso extremo, si se usara un clasificador que asigne todos las entradas al mismo personaje, acertaría más veces al devolver 'Antony').

Extracción de características y reducción de dimensionalidad

Como paso previo a aplicar algoritmos de aprendizaje automático, fue necesario pasar los textos a vectores numéricos. Utilizando la función *CountVectorizer* de Sklearn, se pasó de tener un conjunto de entrenamiento de 438 líneas y una columna (*CharName*), a un conjunto con 2800 columnas, la misma cantidad de filas y 10653 valores no nulos.
Para reducir la dimensionalidad se utiliza PCA con *n_components* = 5, en la Figura 4 se muestra gráficamente el conjunto de entrenamiento utilizando las dos primeras componentes de PCA, para el conjunto de diálogos con y sin stopwords. Como puede observarse, en las componentes del PCA con stopwords incluidas, los elementos de cada grupo estaban superpuestos y no existía una división aparente que delimitara el alcance de uno y otro. En la gráfica de PCA sin stopwords (tras filtrarlas) se notó que cada grupo ocupaba una región más compacta del plano, pero seguían existiendo superposiciones que impedían separar los grupos. En particular, la nube de puntos tomó una forma de "V" donde en ambos brazos existían puntos de los grupos Antony y Cleopatra, pero los puntos del grupo Queen Margaret se distribuían sobre un solo brazo

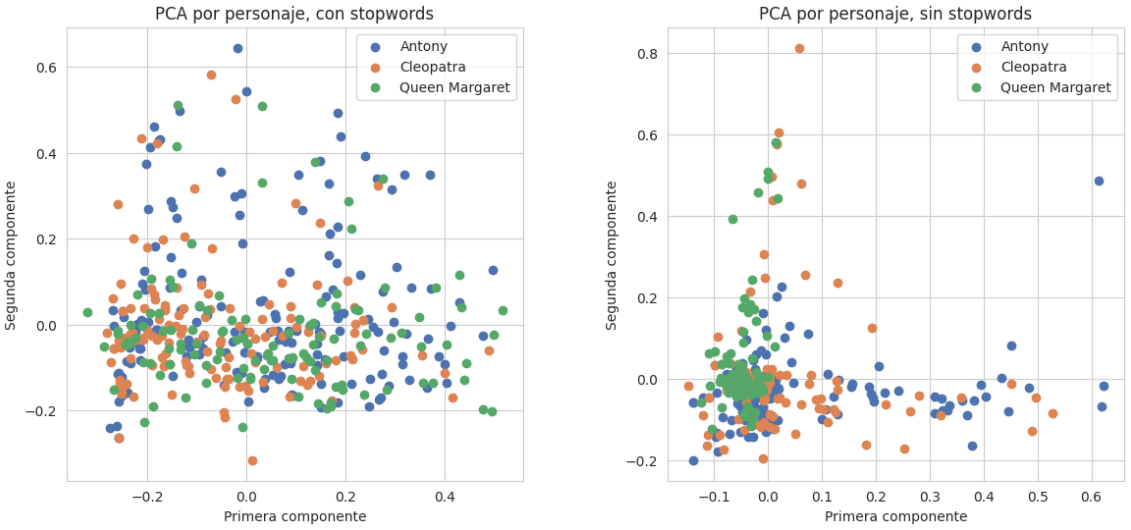


Figura 4: Agrupación de diálogos de personajes de acuerdo a las dos primeras componentes del PCA. La gráfica de la izquierda se correspondía con la reducción de dimensionalidad realizada sobre los diálogos con stopwords incluidas, mientras que el gráfico de la derecha se realizó tras eliminarlas. La varianza explicada para las dos primeras componentes de cada gráfica fue [0.040, 0.024] para la izquierda y [0.015, 0.014] para la de la derecha.

En el código de la tarea (ver [3]) se incluyó otra prueba de PCA con stopwords obligando a que se ejecutara full SVD. La varianza explicada obtenida fue muy similar a la anterior (con variaciones menores a 10^{-8}). Además, se ejecutó PCA para 10 componentes y se varió los parámetros utilizados en *CountVectorizer* para probar si existían mejoras:

1. Tal como en el ejemplo anterior, se preservaron las stopwords y se utilizó un unigrama. Además, se empleó tf para normalizar la matriz de bolsa de palabras.
2. Se quitaron las stop_words para el idioma inglés (*sklearn.feature_extraction.text.ENGLISH_STOP_WORDS*) además de *"'thou', 'thee', 'thy', 'ye', 'thine'"*. El resto de los parámetros se dejó como en el ítem anterior.

La varianza explicada obtenida en cada paso fue:

VE = [0.015264 , 0.01379047, 0.01284152, 0.01151207, 0.01126486, 0.01041234, 0.01008165, 0.00968897, 0.00938774, 0.00909716]

3. Se agregó IDF al transformer (*use_idf=True*), obteniendo como resultado:

VE = [0.01653643, 0.01346824, 0.01309251, 0.01176729, 0.01019829, 0.00909611, 0.00880945, 0.00862964, 0.00831319, 0.00796473]

4. Se probó con unigramas y bigramas (ngram_range=(1,2)), la varianza explicada fue:

VE = [0.02676528, 0.01561221, 0.01463297, 0.01381094, 0.0131081 , 0.01120605, 0.01015289, 0.00955951, 0.00933656, 0.00906481]

Para lograr una mejor interpretación de los resultados, en la Figura 5 se muestran gráficamente los valores de varianza explicada alcanzados para cada componente (hasta un máximo de diez). Además, se incluyó el valor acumulado de la varianza explicada al añadir componentes.

La varianza explicada se puede considerar una medida de cuánta información preserva cada componente añadida (cuánta variación del conjunto original es capaz de capturar cada una). En la Figura 5 puede verse que ninguno de los tipos de preprocesamiento utilizados logró capturar la información contenida en los diálogos con pocas componentes (para 10 componentes se está siempre por debajo del 20% de la varianza total), algo razonable si se considera que el lenguaje tiene muchas dimensiones y características que aportan información (no sólo las palabras utilizadas, sino sus sinónimos, el orden, la entonación, entre otros.). Adicionalmente, la varianza explicada fue mayor al utilizar parámetros más simples; sin embargo, al considerar que este valor se calcula sin tener en cuenta las etiquetas (no se sabe dónde está cada nube de puntos), es razonable que quede mayor si notamos que las nubes de puntos en la Figura 5 quedaron más compactas para el PCA sin stopwords.

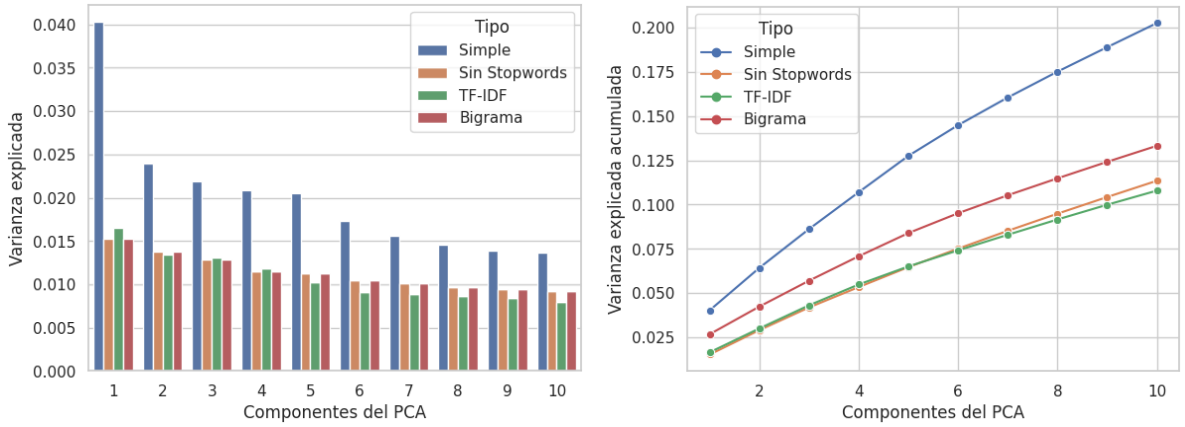


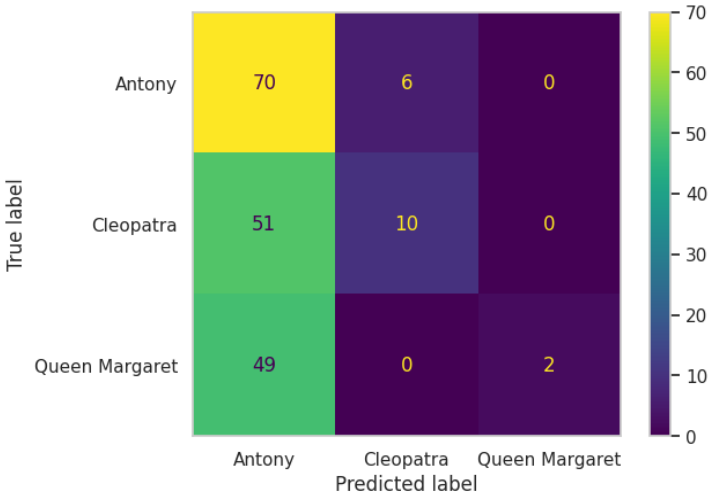
Figura 5: A la izquierda se grafica la varianza explicada para cada componente por tipo de parámetro utilizado. A la derecha se grafica el valor de varianza explicada acumulado para cada componente (contando toda la varianza explicada si se usa hasta ese número de componentes), por tipo.

Luego de procesar los datos y realizar las transformaciones necesarias para poder utilizarlos en modelos de aprendizaje supervisado, se procedió a entrenar diferentes modelos y analizar los resultados obtenidos.

Modelo de aprendizaje supervisado: Multinomial Naive Bayes

Se entrenó el modelo Multinomial Naive Bayes utilizando los datos de entrenamiento (*train*) con stop_words y un rango de n-grama = (1,2) (unigramas y bigramas). El conjunto de

entrenamiento resultante quedó con 13345 características. Se compararon los resultados predichos para el conjunto de entrenamiento con los valores reales y se obtuvo un valor de exactitud (*accuracy*) sobre el conjunto de entrenamiento de 0.794. En cambio, al utilizar el modelo entrenado para predecir con el conjunto de test se obtuvo una exactitud de 0.436.
La matriz de confusión resultante se muestra en la Figura 6, el conjunto de test tenía 76 datos correspondientes a Antony, 61 a Cleopatra y 51 a Queen Margaret. Para el personaje Antony, 70 se predijeron correctamente y 6 se predijeron como Cleopatra, para Cleopatra, 10 se predijeron correctamente y 51 se predijeron como Antony, por último, para Queen Margaret, únicamente 2 se predijeron correctamente y 49 se predijeron como Antony.



	precision	recall	f1-score	support
Antony	0.41	0.92	0.57	76
Cleopatra	0.62	0.16	0.26	61
Queen Margaret	1.00	0.04	0.08	51
accuracy			0.44	188

Figura 6: Matriz de confusión para el clasificador Multinomial Naive Bayes, para una bolsa de palabras con stopwords, unigrama y normalización a partir de tf. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto.

Como puede verse en la imagen, el modelo sobreestimó la cantidad de diálogos pertenecientes a Antony y tendió a asignarle la mayoría de ellos (muy similar al caso límite comentado antes). Esto provocó que, si bien parecía realizar buenas predicciones para Antony, las predicciones dadas para los demás casos eran mayormente malas. Además, parecían empeorar más cuanto menor era la cantidad de diálogos asignados al personaje.
Esta observación se confirmó al explorar las métricas de precisión y sensibilidad para cada categoría:

- Para Antony, el valor de precisión fue 0.41 y el de sensibilidad 0.92.
- Para Cleopatra la precisión tuvo un valor de 0.62 y la sensibilidad de 0.16.
- En el caso de la categoría Queen Margaret, el modelo tuvo una precisión de 1 y una sensibilidad de 0.04.

Si la exactitud es una medida de cuánto “acierta” el modelo, la precisión da una idea, para cierto grupo, de cuántos de los elementos que encuentra efectivamente pertenecen a ese grupo (si no hay falsos positivos, la precisión vale 1) y la sensibilidad cuantifica qué proporción de los elementos reales de un grupo fueron encontrados por el clasificador. Teniendo esto en cuenta, puede verse que si bien el clasificador no es muy preciso al

encontrar diálogos de Antony (porque le asigna la mayoría de los diálogos, generando muchos falsos positivos), dejó sólo un 8% de sus diálogos sin encontrar. Por otro lado, para Cleopatra, el clasificador generó menos falsos positivos (el 60% de los diálogos encontrados son efectivamente de este personaje), pero sólo encontró un 16% del total de diálogos. Esto se puede confirmar visualmente al ver la matriz de confusión, ya que la mayoría de los diálogos de Cleopatra están siendo asignados a Antony. Por último, para Queen Margaret, los dos únicos diálogos asignados a este personaje estaban bien clasificados (la precisión era 1), pero el 96% de sus diálogos se clasifican erróneamente como pertenecientes a Antony (por eso la sensibilidad era de 0.04).
Otra medida a tener en cuenta fue el F1-score, la cual combinaba las medidas de precisión y sensibilidad en un solo valor: $f1 = \frac{2(precision\ X\ recall)}{precision + recall}$. Ésta asumió que importaban de igual forma la precisión y la sensibilidad, y puede verse que los grupos con menor puntaje eran aquellos que estaban menos representados en el conjunto de datos. Todas estas métricas son buenos complementos de la exactitud, ya que ésta no tiene en cuenta cómo se distribuyen los datos y no funciona bien cuando las clases están desbalanceadas. Por ejemplo, si se sustituyeran los personajes Cleopatra y Queen Margaret por otros con menos líneas de diálogo, lo más probable es que al reentrenar, la exactitud aumentara para los datos de test, ya que el porcentaje de diálogos pertenecientes a Antony aumentaría, y el algoritmo acertaría más veces al clasificar todos los diálogos es su grupo.
Viendo los resultados obtenidos antes para este clasificador, se utilizó el método de validación cruzada para obtener valores de exactitud, precisión y sensibilidad más representativos de cada modelo, a la vez que se designaron ocho parámetros distintos bajo los que realizar la extracción de características (Tabla 3).

Tabla 3: Parámetros utilizados para generar las características

Parámetro	Stop Words	Ngram	idf
0	None	(1,1)	False
1	None	(1,1)	True
2	None	(1,2)	False
3	None	(1,2)	True
4	ENGLISH_STOP_WORDS	(1,1)	False
5	ENGLISH_STOP_WORDS	(1,2)	False
6	ENGLISH_STOP_WORDS	(1,1)	True
7	ENGLISH_STOP_WORDS	(1,2)	True

Los valores de las métricas obtenidas para cada modelo pueden observarse en la Figura 7. A partir de los resultados del gráfico de violín pudo verse que los primeros cuatro parámetros probados obtuvieron calificaciones bajas de exactitud (menor a 50%), precisión (fue alta sólo para Antony por los problemas ya expuestos) y sensibilidad (una media elevada para Queen Margaret, pero con alta dispersión). Estos parámetros tenían en común que ninguno de ellos filtraba las stopwords del conjunto de diálogos, lo que dio la pauta de que el quitar información repetitiva o presente en todos los personajes mejoraba las

estimaciones, porque probablemente fuera en las palabras poco comunes donde se encontraban las diferencias más ricas.

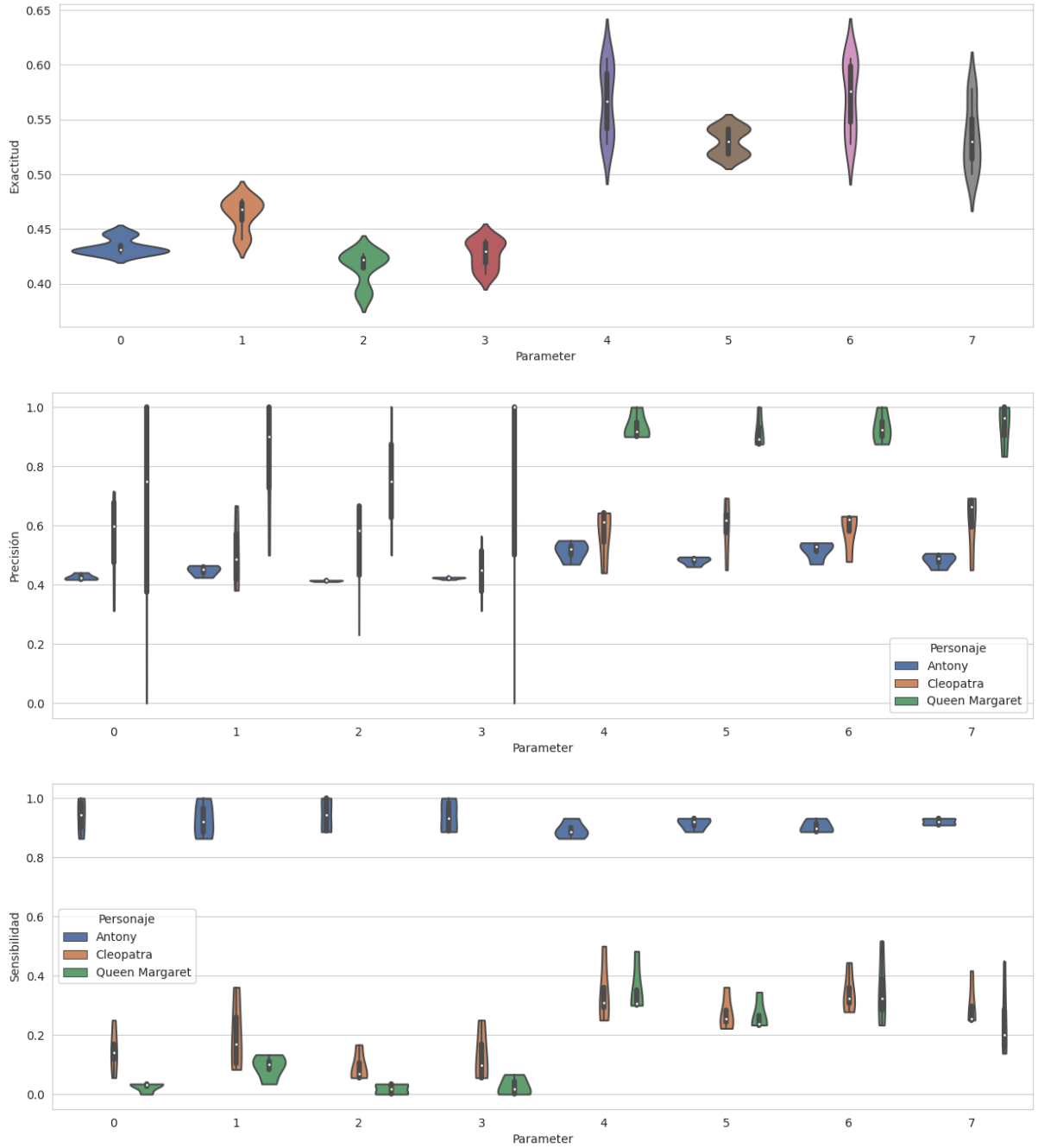
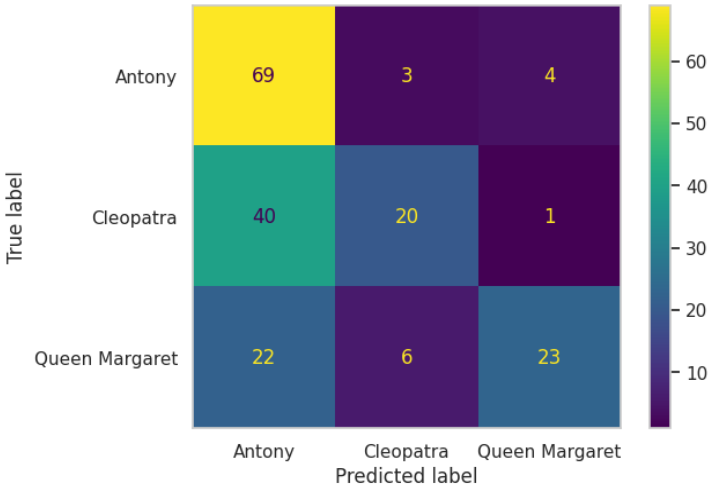


Figura 7: Por cada conjunto de métricas obtenidas en la validación cruzada se construye un gráfico de violín, a modo de identificar el rendimiento de los distintos conjuntos de parámetros. Arriba se presentan las métricas de exactitud, en el medio las de precisión (para cada grupo) y abajo las de sensibilidad (también separadas por parámetro y grupo).

Si se observan los últimos cuatro parámetros, para los que se filtraban las stopwords, se ve que aquellos con mayor exactitud fueron los grupos 4 y 6, donde se utilizaban stopwords con unigramas (la diferencia entre uno y otro estaba en si se usaba tf o tf-idf), a pesar de presentar una dispersión más alta que el parámetro 5 (unigrama y bigrama con tf). Respecto a sus desempeños frente a las métricas de precisión y sensibilidad, los mismos presentaron rendimientos similares para todos los grupos de personajes, observándose en el parámetro

6 una ligera ventaja en el valor medio de cada conjunto, y una menor dispersión. Por las razones ya mencionadas, se eligió reentrenar el modelo y evaluar su rendimiento usando el parámetro 6 en la extracción de características.
Los resultados de reentrenar con un conjunto ligeramente transformado se pueden observar en la matriz de confusión y sus métricas de la Figura 8. A simple vista se puede ver que la mejora en la estimación era evidente, ya que no solo la exactitud aumentaba en casi 0.2, sino que los valores de precisión y sensibilidad se mantenían mayores a 50% y 30% respectivamente. Se evidenció cierto compromiso entre el aumento de uno (precisión o sensibilidad) en detrimento del otro, aunque resultaba razonable que sucediera (por ejemplo, a menos diálogos del personaje dejaba fuera, aumentaba la probabilidad de que incluyera un diálogo de otro y se equivocara).



	precision	recall	f1-score	support
Antony	0.53	0.91	0.67	76
Cleopatra	0.69	0.33	0.44	61
Queen Margaret	0.82	0.45	0.58	51
accuracy			0.60	188

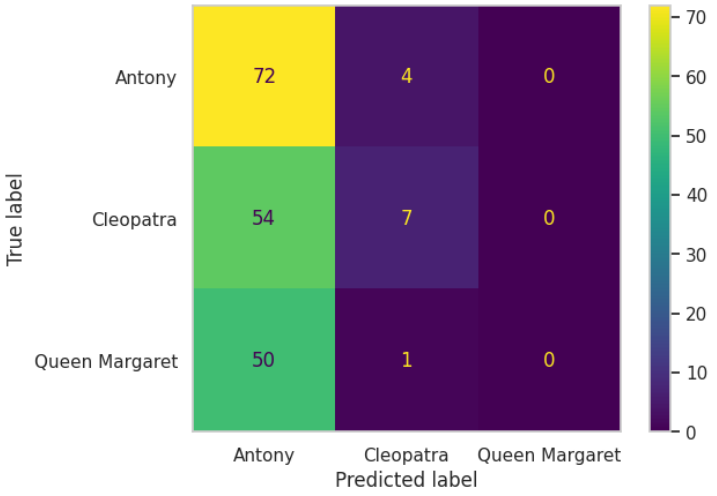
Figura 8: Matriz de confusión para el clasificador Multinomial Naive Bayes, para una bolsa de palabras sin stopwords, usando unigrama y normalización a partir de tf-idf. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto.

Si bien la mejora con respecto al primer entrenamiento fue evidente, los modelos basados en bolsas de palabras tienen sus limitaciones, ya que pierden información de estructura, semántica y gramatical de las oraciones, y se quedan sólo con frecuencias de palabras. A pesar de que tf-idf busca “paliar” la sobrerrepresentación de palabras que aportan poca información, su utilidad se vio disminuida al filtrar las stopwords (ya que son éstas las que solían introducir esa repetición o irrelevancia en el texto). Se vuelve necesario, para problemas donde la relación, interacción y cercanía de las palabras que forman oraciones es importante, utilizar otra forma de extraer características del texto (por ejemplo, word embeddings).

Otros modelos de aprendizaje supervisado: KNN, SVD y Fasttext

Se entrenó un modelo de K vecinos más cercanos utilizando tres vecinos (*n_neighbor = 3*) y los mismos parámetros para la extracción de características que en la parte anterior

(parámtero 6). Se obtuvo un valor de exactitud para el conjunto de entrenamiento de 0.64, mientras que al validar el conjunto de test dicha métrica descendió a 0.42. Respecto a los resultados de la clasificación y las métricas, los mismos pueden observarse en la Figura 9.



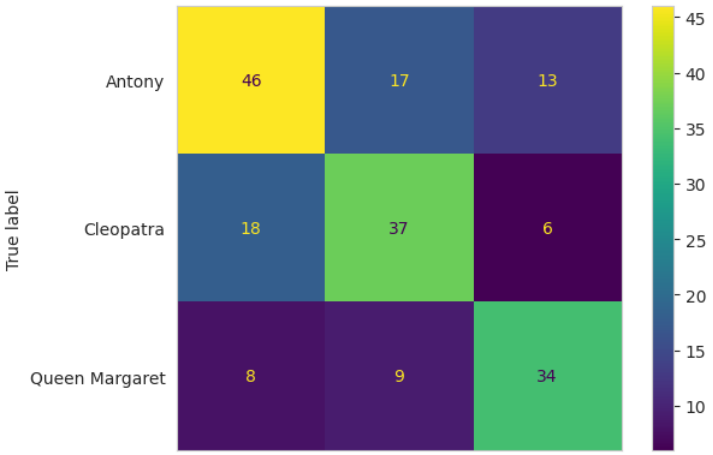
	precision	recall	f1-score	support
Antony	0.41	0.95	0.57	76
Cleopatra	0.58	0.11	0.19	61
Queen Margaret	0.00	0.00	0.00	51
accuracy			0.42	188

Figura 9: Matriz de confusión para el clasificador K vecinos más cercanos, para una bolsa de palabras sin stopwords, usando unigrama y normalización a partir de tf-idf. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto.

En este caso, los valores de sensibilidad (*recall*) para Queen Margaret se computaron como nulos, pero como en varios casos ocurrió que el denominador de la ecuación resultaba nulo, esto afectó también el cómputo de la métrica f1. Para este clasificador resultó evidente que se obtenían peores desempeños que para Multinomial Naive Bayes. Intentando encontrar un motivo de por qué sucedía esto, resultó llamativo observar que, mientras MNB calculaba la probabilidad condicional de que ocurriera cierto evento (para este trabajo, sería que X personaje esté hablando) en función de algo que ya había ocurrido (el diálogo menciona cierta/s palabra/s), mientras que K vecinos más cercanos consideraba una nueva ocurrencia (un diálogo) y observaba a qué grupo pertenecían sus K vecinos más cercanos. Considerando que no se utilizó un método de extracción de features que preservara relaciones semánticas, resultó posible que en la realidad la distancia entre algunas features fuera menor que otras (por ejemplo, 'rain' y 'wet' deberían estar más cerca entre sí que 'wet' y 'dry') pero ésto no se vió reflejado en el modelo (obteniendo una peor clasificación). Sería interesante reiterar el análisis cambiando el método de extracción de features a word embedding y entrenando nuevamente todos los modelos.

Por otro lado, con los mismos conjuntos de train y test se entrenó un modelo SVM utilizando la función *SGDClassifier* con el parámetro *loss = 'hinge'*, el cual devolvió un SVM lineal; además, se utilizó un máximo de cinco iteraciones. Se obtuvo una exactitud para el conjunto de entrenamiento de 0.98, mientras que dicha métrica en el conjunto de evaluación alcanzó 0.62. El desempeño de la clasificación puede observarse en la Figura 10. Lo primero que resulta de interés al observar los resultados devueltos por el clasificador, es que la matriz de confusión parece tener una diagonal mucho más marcada que los modelos anteriores

(aunque con un menor valor para Antony) dando cuenta de que existieron más aciertos para las clases subrepresentadas. Si se observan las métricas obtenidas para el conjunto de evaluación, puede verse que la exactitud aumentó ligeramente (en 2%), mientras que las métricas de f1 aumentaron para los dos personajes menos representados (en un 16% para Cleopatra y en un 7% para Queen Margaret). De cierta forma parece que se igualaron los rendimientos del modelo para los tres grupos, ya que mejoró levemente la precisión para Antony (en 11%) aunque empeorara su sensibilidad (en 30%), mientras que para Cleopatra y Queen Margaret empeoró la precisión (disminuyó en 10% y 18% respectivamente) en beneficio de una mejora en la sensibilidad (de 28% y 22% respectivamente).
Estos resultados llevaron a creer que SVM mejora, para este caso, la estimación dada por los modelos anteriores, pero no por un amplio margen. Si bien existen numerosos artículos que posicionan a SVM como un modelo excelente al trabajar con bolsas de palabras, otros autores matizan estas afirmaciones, mencionando que bajo ciertas condiciones no mejora las estimaciones dadas por naive Bayes o KNN [10]. En el ejemplo particular de este trabajo, se cuenta con una cantidad de características lo suficientemente grande como para que el desempeño de SVM dependa fuertemente de los parámetros de ajuste que se elijan (puntualmente el parámetro C), por lo que su no tan notable mejora podía estar asociado a esto.

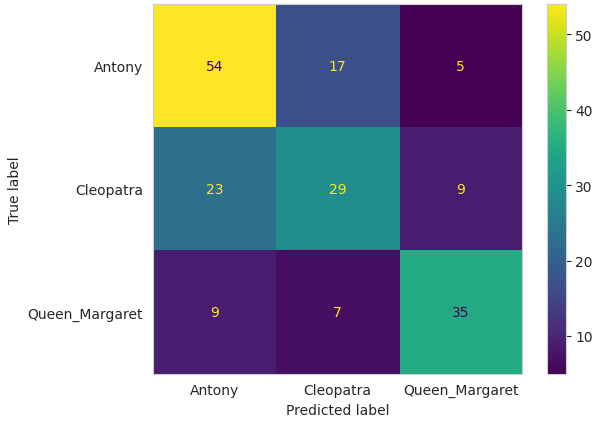


	precision	recall	f1-score	support
Antony	0.64	0.61	0.62	76
Cleopatra	0.59	0.61	0.60	61
Queen Margaret	0.64	0.67	0.65	51
accuracy			0.62	188

Figura 10: Matriz de confusión para el clasificador SVM con descenso por gradiente, para una bolsa de palabras sin stopwords, usando unigrama y normalización a partir de tf-idf. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto.

Por último, se decidió probar con el algoritmo *Fasttext* para verificar si se obtenían mejores resultados que los obtenidos al trabajar los textos como bolsa de palabras, se trabajó con *wordNGrams* = 2. La exactitud en este caso fue de 0.63. Como se puede apreciar en la Figura 11 el valor obtenido en f1 para los tres personajes aumentó en comparación a los modelos anteriores.

De los 76 diálogos correspondientes a Antony, 54 fueron clasificados correctamente; para Cleopatra se tenían 61 y se clasificaron correctamente 29 y para Queen Margaret de los 51 se encontraron 35. Al observar las métricas resultantes de esta clasificación, puede verse que la mejora en la exactitud no parecía demasiado distinta con respecto a MNB y SVM. Respecto a los valores de *f1-score* para cada clase, se observó una mejora para las clases Cleopatra y Queen Margaret (respecto a MNB), mientras que el valor para Antony se mantuvo. Comparado con SVD, se observó un peor rendimiento en la clasificación de los diálogos de Cleopatra, contrarrestado en parte por la mejora para los grupos restantes (tanto en precisión como en sensibilidad).



	precision	recall	f1-score	support
Antony	0.63	0.71	0.67	76
Cleopatra	0.55	0.48	0.51	61
Queen_Margaret	0.71	0.69	0.70	51
accuracy			0.63	188

Figura 11: Matriz de confusión para Fasttext. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto.

Clasificación con nuevos personajes

Se decidió entrenar y evaluar el desempeño de Multinomial Naive Bayes cambiando a uno de los personajes del conjunto. Se mantuvieron las proporciones para los conjuntos de entrenamiento y evaluación, y se utilizó el parámetro 6 para la extracción de características de texto.

De acuerdo a la cantidad de párrafos por personaje encontrados en la Figura 1, se eligió en primer lugar a Rosalind como sustituto de Antony, ya que tenía asignados 201 párrafos (similar a las cantidades de Cleopatra). Por otro lado, se eligió sustituir (en otra instancia de entrenamiento) a Cleopatra por Henry V, para acentuar los desbalances de representación de cada grupo en el conjunto. Como puede observarse en la Figura 12, para el primer caso se disminuye el desbalance de datos entre los grupos para cada uno de los conjuntos (se espera que esto mejore la estimación) pero la reducción de características siguió siendo inefectiva. Por otro lado, el segundo conjunto estaba notoriamente desbalanceado en cuanto a datos por clase y los resultados para su reducción de dimensionalidad mediante PCA se mostraron más superpuestos que para el conjunto previo. Si además se observaban los valores de varianza explicada de sus primeras dos componentes, se tenía que para el primer conjunto se estaba por encima de 0.01 en ambos casos (0.01075867 y

0.01001631) mientras que para el segundo conjunto la varianza explicada por las primeras dos componentes disminuyó a menos de 0.08 (0.00753763 y 0.00712269) lo que podría dar cuenta de una mayor dificultad para almacenar la información en un número reducido de componentes: seguramente para lograr explicar la misma proporción de información, el segundo conjunto necesitaría de más componentes.

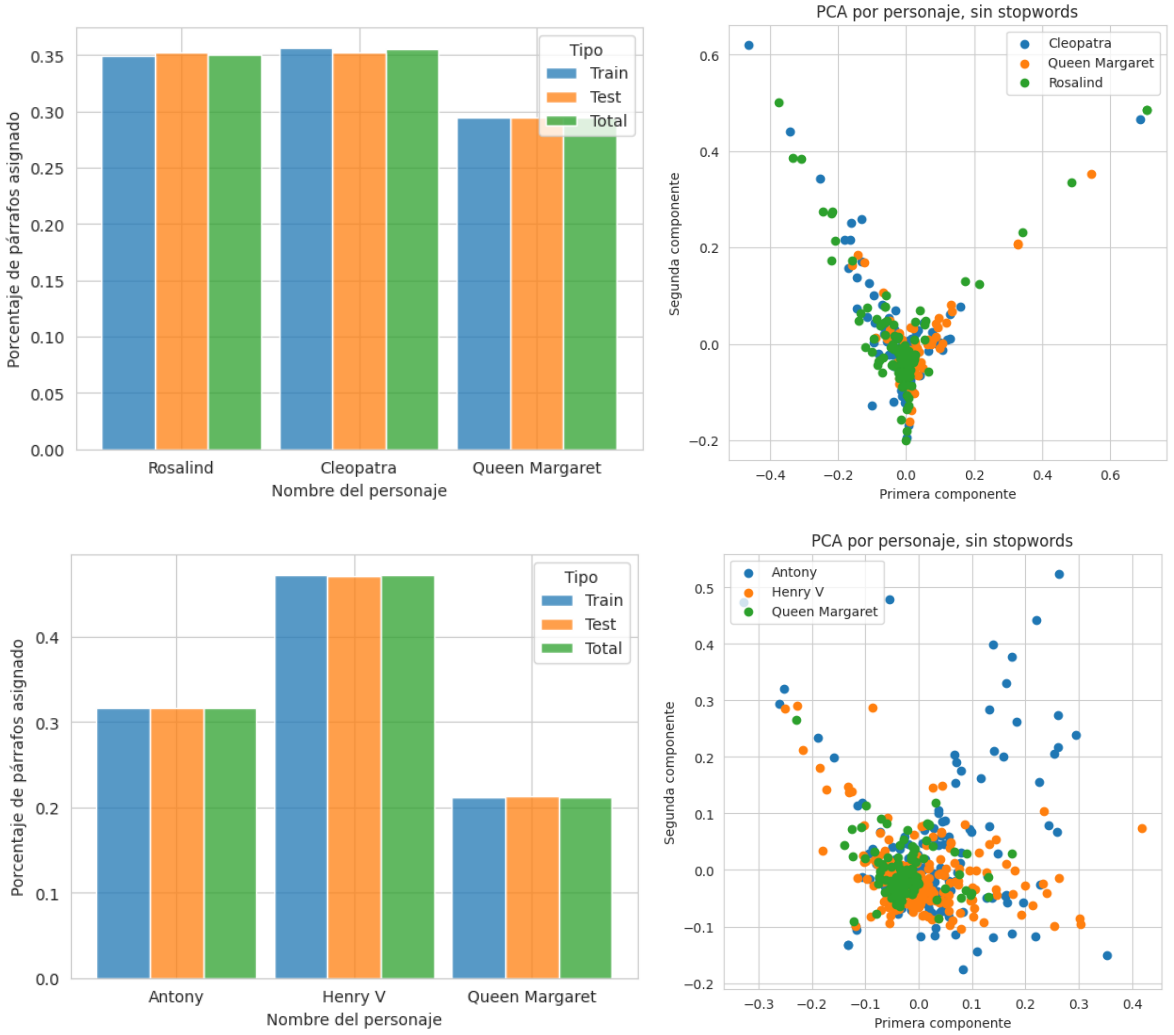
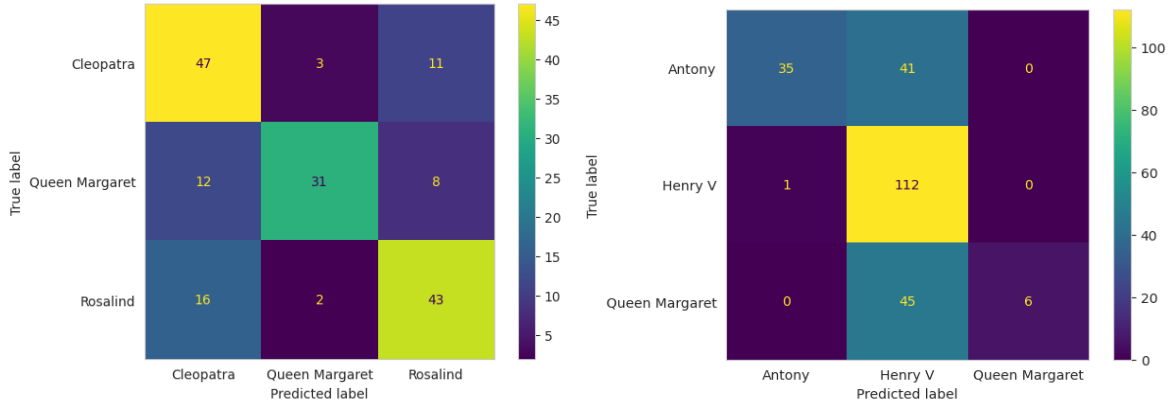


Figura 12: En la gráfica de barras de la izquierda se puede observar la distribución de los diálogos de cada personaje en los distintos conjuntos definidos. Por otro lado, a la derecha se muestra el resultado de la reducción de dimensionalidad mediante PCA. Arriba se tiene un ejemplo con un conjunto de datos más balanceado, mientras que debajo se acentúan las diferencias entre los grupos del conjunto.

El resultado de entrenar ambos modelos a partir de estos datos se muestra en la Figura 13. Tal como se esperaba, utilizar conjuntos desbalanceados empeoró el rendimiento del clasificador y generó un sesgo hacia los datos con mayor representación en el conjunto (se tendió a clasificar los diálogos como pertenecientes a la clase más representada).
Si se observan las métricas obtenidas para el conjunto balanceado, la exactitud aumentó hasta 70%, mientras que los valores de la métrica f1 se situaron por encima de 69% para todas las clases. Adicionalmente, la precisión y sensibilidad de los distintos grupos se mantuvo siempre por encima de 60%. En cambio, observando el conjunto desbalanceado, pudo verse que si bien la exactitud no disminuyó mucho con respecto al anterior, el valor de la métrica f1 para cada clase presentó una disminución notoria para aquellas menos representadas (Queen Margaret), desplomándose hasta 21%. Por otro lado, volvió a

observarse una precisión de 100% para la clase subrepresentada, nuevamente con una sensibilidad pequeña, de sólo 12%. Además, la clase con mayor representación sufrió el fenómeno inverso: su precisión fue relativamente baja (57%) pero su sensibilidad era muy alta (99%).



	precision	recall	f1-score	support
Cleopatra	0.63	0.77	0.69	61
Queen Margaret	0.86	0.61	0.71	51
Rosalind	0.69	0.70	0.70	61
accuracy			0.70	173

	precision	recall	f1-score	support
Antony	0.97	0.46	0.62	76
Henry V	0.57	0.99	0.72	113
Queen Margaret	1.00	0.12	0.21	51
accuracy			0.64	240

Figura 13: Matriz de confusión para el clasificador SVM con descenso por gradiente, para una bolsa de palabras sin stopwords, usando unigrama y normalización a partir de tf-idf. Las métricas del modelo se muestran debajo de la matriz de confusión. support hace alusión a la cantidad de elementos en cada conjunto. A la izquierda se tiene un ejemplo con un conjunto de datos más balanceado, mientras que a la derecha se acentúan las diferencias entre los grupos del conjunto al añadir grupos con gran diferencia de elementos.

Respecto a cómo afrontar problemas de clasificación en casos donde las clases estan desbalanceadas, podría resultar conveniente aplicar técnicas de submuestreo (se descartan datos de las clases con mayor representación en el conjunto) o sobremuestreo (se fabrican nuevos datos a añadir en las clases subrepresentadas, ya fuera utilizando los datos existentes y aplicando transformaciones, o generando nuevos datos). La primera es conveniente en casos donde se cuenta con datos de sobra para utilizar en el entrenamiento del modelo, por lo que las consecuencias de disminuir la cardinalidad del conjunto de entrenamiento no imposibilita el entrenamiento.
Cabe mencionar que este modelo recibía como parámetro un archivo que debía seguir cierto formato, por lo que era necesario generarlo a través del formateo de los datos. Esto podría generar ciertos inconvenientes dependiendo de qué caracteres están presentes en el texto y cómo se realizaba la limpieza de los mismos. Adicionalmente, cabe añadir que el entrenamiento de este modelo no se ejecutó por igual en todos los equipos del grupo: en algunos de ellos provocó el cierre del programa desde donde se ejecutaba el jupyter notebook, por lo que se puede suponer que el entrenamiento del mismo demandaba más recursos que otros modelos alternativos.

Conclusiones

Durante el trabajo se utilizaron los datos de la base de datos relacional Open Source Shakespeare para un primer análisis exploratorio de los datos y contenidos de las tablas. Se evaluó el estado de los datos, encontrando algunas entradas faltantes en la tabla de personajes, además de realizarse una limpieza de datos sobre el texto de los diálogos recopilados.

Posteriormente, se procedió al entrenamiento de varios modelos de clasificación mediante aprendizaje supervisado. Se observaron puntualmente los desempeños de cuatro clasificadores: Multinomial Naive Bayes, K Nearest Neighbor, Support Vector Machine y Fasttext; observando que el peor desempeño estaba dado por el segundo de ellos. Respecto a los restantes, Fasttext mejoraba el resultado dado por MNB (aunque no por mucho) y se situaba en un rendimiento similar al de SVM.

Durante las pruebas, se variaron las condiciones bajo las que se realizaba la extracción de características, para evaluar su incidencia sobre el rendimiento del modelo. Se aplicó el método de validación cruzada para encontrar el mejor conjunto de parámetros a aplicar en la extracción de características, que terminó siendo aquel que filtraba las stopwords y normalizaba la matriz esparza mediante tf-idf.

Por último, se evaluó el desempeño de MNB ante dos conjuntos con distintas proporciones de datos en cada clase, observando que el desbalance en los mismos afecta de manera negativa el desempeño del modelo.

Referencias

[1] George Mason University. *Download the source code and OSS database*. (n.d.). Open Source Shakespeare. Disponible en: www.opensourceshakespeare.org.

[2] Información del proyecto Jupyter, https://jupyter.org/.

[3] Repositorio en Gitlab del Notebook implementado. Disponible de forma pública en: https://gitlab.fing.edu.uy/lucia.lemes/introcd.

[4] 6.2. Feature extraction [Internet]. scikit-learn. Disponible en: https://scikit-learn.org/stable/modules/feature_extraction.html#

[5] 2.5.1. Principal component analysis (PCA) [Internet]. scikit-learn. Disponible en: https://scikit-learn.org/stable/modules/decomposition.html#pca

[6] 1.6. Nearest Neighbors [Internet]. scikit-learn. Disponible en: https://scikit-learn.org/stable/modules/neighbors.html

[7] 1.4. Support Vector Machines [Internet]. scikit-learn. Disponible en: https://scikit-learn.org/stable/modules/svm.html

[8] Fasttext [Internet]. scikit-learn https://fasttext.cc/

[9] 3.1 Cross-validation: evaluating estimator performance [Internet]. scikit-learn. Disponible en https://scikit-learn.org/stable/modules/cross_validation.html

[10] Colas, F., Paclík, P., Kok, J. N., & Brazdil, P. (2007). Does SVM really scale up to large bag of words feature spaces?. In *Advances in Intelligent Data Analysis VII: 7th International Symposium on Intelligent Data Analysis, IDA 2007, Ljubljana, Slovenia, September 6-8, 2007. Proceedings 7* (pp. 296-307). Springer Berlin Heidelberg.

Anexo

1- Esquema base de datos

El esquema de la base de datos utilizada se puede ver en la Figura 14. La misma cuenta con un total de 43 obras, divididas en 945 capítulos, donde existen 1266 personajes y 35465 párrafos.
Las particularidades de cada tabla se detallan a continuación.

- **works:** Contiene información de las obras publicadas, con un valor de id, año de publicación, género y título.
- **chapters:** Refiere a los capítulos de cada obra, se relaciona con la tabla anterior por medio del atributo *work_id*. Cada capítulo tiene asignado, además, un número de identificación (*id*), un número de acto, número de escena y una descripción del mismo.
- **characters:** Guarda los personajes que aparecen en cada uno de los trabajos escritos, cada personaje se encuentra vinculado a sus párrafos asignados a través de su identificador (*id* del personaje) en la tabla paragraphs. Como datos guardados en la tabla, se tiene un identificador único del personaje, su nombre, la abreviación del mismo y una breve descripción.
- **paragraphs:** Refiere a los párrafos de cada capítulo, vinculados con el capítulo y personaje correspondiente a través de los identificadores *chapter_id* y *character_id*. Cada párrafo tiene un identificador (*id*), un número de párrafo, el texto que contiene, y los identificadores al capítulo (*chapter_id*) y personaje (*character_id*) al que hace referencia.

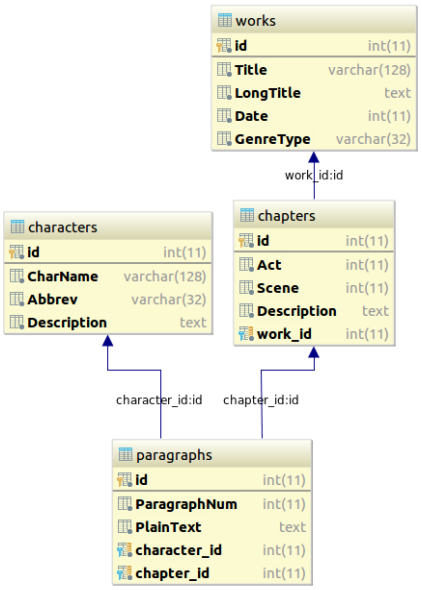


Figura 14: Modelo relacional de la base de datos “Open Source Shakespeare”. En el mismo se detallan los nombres de cada tabla con sus atributos (nombre y tipo).

2- Obtención y limpieza de datos

Se utilizó inicialmente el paquete de funciones *sqlalchemy* (en su versión anterior a 2.0) para descargar los datos de OSS en archivos .csv. Posteriormente, se utilizó el paquete *pandas* para la carga y manipulación de cada tabla mediante el uso de la clase *DataFrame.* En particular, esta clase permite identificar campos faltantes (espacios vacíos, null, etc) y extraer un informe inicial de cuántas ocurrencias hay por columna.
Para la limpieza de datos se utilizó la función *clean_text()*, usada para pasar el texto de los párrafos a minúscula, y eliminar la mayoría de los símbolos de puntuación. Cabe mencionar que se probó con el paquete *contractions* para eliminar las contracciones permitidas en el idioma inglés (por ejemplo, pasar de *It's →It is*) y así poder contar palabras individuales. Sin embargo, el uso de inglés antiguo en los textos impidió eliminar la mayoría de ellas (de 27819 se resolvieron menos de 3000), ya que muchas veces se utilizaba el apóstrofe para la conjugación de verbos en pasado (kill'd, dissever'd, perform'd, etc.). Considerando el desempeño observado, se decidió omitir este paso y conservar los apóstrofes (contando en cada caso una única palabra). Una vez realizado el procesamiento del texto se realizaron las visualizaciones a partir de los paquetes de *matplotlib* y *seaborn*.

Exploración de datos

La exploración de datos permitió identificar campos faltantes en algunas filas de la tabla *characters*, donde se vió que no todos los personajes contaban con una abreviación de su nombre (5 faltantes identificados) y más de la mitad tampoco contaba con una descripción (hay 646 sin ninguna descripción). Además, los campos de texto no estaban normalizados, habiendo varios que mezclaban mayúsculas con minúsculas.
También, en el texto correspondiente a los párrafos existían abreviaciones, símbolos, signos de puntuación y contracciones, además de mezclar mayúsculas y minúsculas. Para trabajar con el contenido de los párrafos (por ejemplo, agrupar por palabra y contar cuántas veces se usó cada una) fue necesario normalizar y corregir los valores de estos campos.

Limpieza de texto

Previo a aplicar transformaciones que permitiera modificar o quitar el contenido no deseado de la columna *PlainText*, fue necesario evaluar qué signos o caracteres se utilizaban en conjunto con el texto de interés. Para ello, se llevó el texto a minúsculas y se retiraron todos los caracteres del abecedario (de la "a" a la "z"), junto con aquellos que representaban números (del "0" al "9"). Luego, a partir del texto resultante, se extrajeron los símbolos que permanecían y que se deseaban eliminar en la etapa de limpieza.
En total, se obtuvieron 16 símbolos diferentes, incluyendo el apóstrofe (utilizado en las contracciones). De éstos se añadieron 15 en la función *clean_text*, donde se incluyen: la coma, el salto de línea (\n), punto, apóstrofe, punto y coma, dos puntos, signo de interrogación, signo de exclamación, guión, paréntesis rectos, paréntesis curvos, et (&), comillas doble y el salto de tabulación (\t).
Todas las ocurrencias de estos símbolos fueron sustituidas por espacios en blanco. Respecto del apóstrofe, se exploró una librería que permitiera resolver las contracciones, sin embargo, de las 27819 veces en que se utilizó este símbolo, un total de 19882 quedó sin resolver. Esto ocurrió debido a la aparición del apóstrofe utilizado en verbos con conjugación en pasado, además de otros casos particulares que la librería no logró resolver. Considerando que la misma podría arrojar resultados erróneos (por ejemplo, confundir usos

de “is” con “has”) se prefirió omitir este paso en la limpieza del texto, y conservar todas las contracciones.
Finalmente, se separaron las palabras individuales en listas (una por párrafo) y se creó otro DataFrame que guardara todas las ocurrencias de cada palabra existente en la obra de Shakespeare.

3- Personajes principales

Por cantidad de palabras

Fue de interés para el trabajo encontrar qué personajes tenían asignadas la mayor cantidad de palabras.
Como un primer acercamiento se agregó al conjunto de palabras los datos de los personajes: nombre e identificador, luego se los agrupó por nombre y se contó la cantidad de palabras que correspondían a cada nombre.

CharName	count
Poet	49730
(stage directions)	16408
Henry V	15223
Falstaff	14626
Hamlet	11961
Duke of Gloucester	9331
Antony	8632
Iago	8475
Henry IV	8251
Vincentio	6970
Henry VI	6907
Richard III	6872
Queen Margaret	6834
Coriolanus	6613
Timon	6478

(a)

CharName	count
All	23
Messenger	23
Servant	21
Lord	9
Page	8
First Lord	8
First Gentleman	8
Second Gentleman	8
Gentleman	7
First Servant	7
Both	7
Captain	7
First Citizen	6
Second Servant	6
Second Lord	6

(b)

character_id	count_x	id	CharName
894	48950	894	Poet
1261	16408	1261	(stage directions)
573	15223	573	Henry V
393	14626	393	Falstaff
559	11961	559	Hamlet
531	9331	531	Duke of Gloucester
120	8632	120	Antony
600	8475	600	Iago
572	8251	572	Henry IV
574	6907	574	Henry VI
945	6872	945	Richard III
736	6834	736	Queen Margaret
1236	6617	1236	Vincentio
283	6613	283	Coriolanus
1198	6478	1198	Timon

(c)

Figura 15: En (a) puede observarse una tabla con los nombres de los personajes que aparecen en la obra de Shakespeare, y la cantidad de palabras dichas por éste (en toda las obras). En (b) se indican los nombres que se repiten en las obras (el mismo nombre refiere a personajes distintos), junto con la cantidad de personajes que lo comparten. Por último, en (c) se muestra la cantidad de palabras por personaje, pero esta vez agrupado por identificador. Esto garantiza que se agrupan las palabras por un único personaje. Adicionalmente, se añade el nombre del personaje y cuántas palabras dijo.

Los primeros 15 resultados obtenidos se pueden observar en la tabla de la Figura 15a. Sin embargo, agrupar las palabras por el nombre del personaje que las dijo presenta problemas cuando éste no es único. Haciendo un breve análisis, se constató que los nombres se repetían en las diferentes obras, por lo que la agrupación por nombres no era correcta, sino que se debía agrupar por el identificador de personaje. En la Figura 15b se muestra una tabla con los primeros 15 nombres de personajes repetidos en las obras, sobre un total de 125 nombres repetidos.

Al agrupar por identificador de personaje se obtuvo una lista similar a la de la Figura 15a, pero era posible apreciar que, por ejemplo, “Poet” (el nombre de personaje con más palabras en toda las obras) tiene un total de 49730 palabras, pero sólo 48950 son las correspondientes al personaje con identificador 894 y nombre “Poet”. Los valores separados por identificador para los 15 personajes con más palabras pueden observarse en la Figura 15c.
Algo interesante a señalar, asociado a un problema ya mencionado en la sección de conteo de palabras, es que en esta base de datos se consideran las indicaciones para la puesta en escena como texto correspondiente al personaje “(stage directions)”, con un id distinto para cada obra. Se sabe que éstos no son personajes de las obras de Shakespeare, sino que recopila información necesaria para ejecutarlas; por eso, al momento de realizar una visualización comparativa entre personajes se decidió dejar fuera las indicaciones de escena.

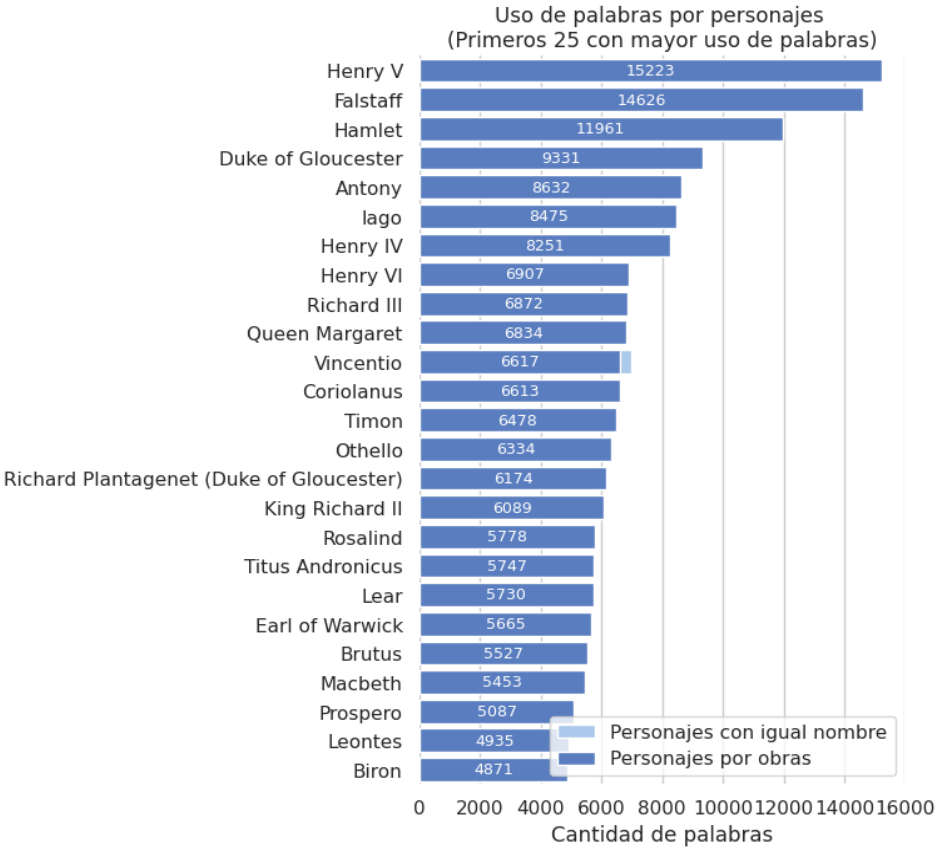


Figura 16: En la imagen puede observarse un gráfico de barras comparativo de la cantidad de palabras dichas por los 25 personajes con más palabras en la obra de William Shakespeare. En azul oscuro se muestra la cantidad de palabras dichas por un único personaje cuyo nombre es igual al que se indica a la izquierda (el largo de la barra se escribe en el centro de la misma), mientras que en celeste claro la cantidad de palabras dicha por todos los personajes que comparten dicho nombre (a modo de comparación).

En la Figura 16 se grafica de forma comparativa las cantidades de las Figuras 15a y 15c, para los 25 personajes con más palabras. Como añadido se incluye, para aquellos personajes con nombre repetido, la cantidad total de palabras dichas por personajes con ese nombre, y la cantidad total de palabras dicha por un único personaje con ese nombre (el de mayor cantidad de palabras). El número al final de cada barra indica la cantidad de palabras dichas por ese personaje, de acuerdo a lo mostrado en la Figura 15c.

A simple vista, entre los personajes con mayor cantidad de diálogo, “Vicentio” es quien tiene un nombre asignado a más de un personaje en distintas obras.

Por cantidad de párrafos

Se realizó el mismo procedimiento descrito anteriormente, pero esta vez utilizando la cantidad de párrafos de las obras en vez de la cantidad de palabras.
La cantidad de párrafos en una obra de teatro corresponden por lo general a la cantidad de diálogos que realiza el personaje. Al contar la cantidad de párrafos por personaje se tiene una idea un poco más certera de cuáles son los personajes principales. En la Figura 17 se muestran gráficamente los resultados obtenidos para los primeros 25 personajes (se quitó de la lista el primer personaje “stage directions” dado que corresponde a las direcciones de escena).

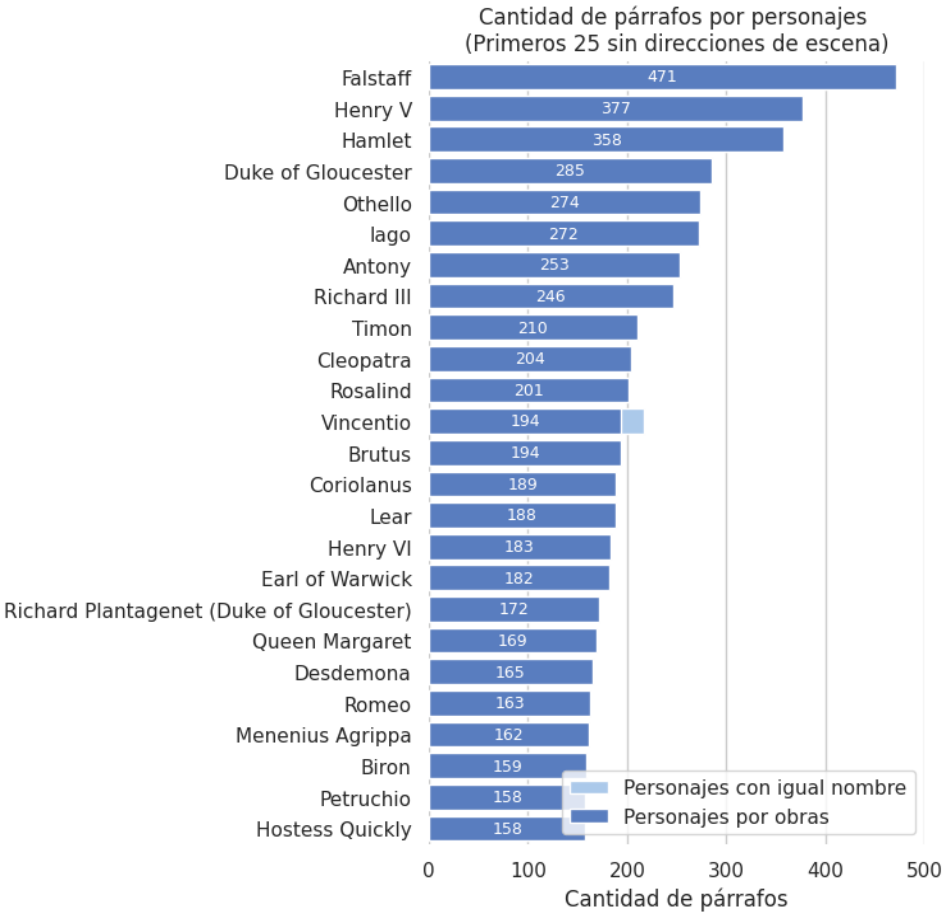


Figura 17: En la imagen puede observarse un gráfico de barras comparativo de la cantidad de diálogos (párrafos) dichos por los 25 personajes con más intervenciones en la obra de William Shakespeare. En azul oscuro se muestra la cantidad de párrafos dichos por un único personaje cuyo nombre es igual al que se indica a la izquierda (el largo de la barra se escribe a la derecha de la misma), mientras que en celeste claro la cantidad de párrafos asignados a todos los personajes que comparten dicho nombre (a modo de comparación).

Por cantidad de párrafos por Obra

Fue de interés explorar una visualización que extrajera el personaje con más líneas de diálogo en cada obra, y mostrarlo asociando dicha cantidad y la obra a su nombre.
Para obtener el personaje principal de cada obra se realizó un procedimiento similar a los comentados antes: a los datos de párrafos por personaje se le agregaron los capítulos correspondientes a cada párrafo y las obras correspondientes a cada capítulo; de esta

manera se tiene un conjunto de datos donde se puede agrupar los personajes con mayor cantidad de párrafos dentro de cada obra. Luego, se repitió esto pero contando la cantidad de palabras.

En las figuras 19 y 20 se muestran ambos resultados, y en la Figura 18 se propone una visualización alternativa para la Figura 19 (mediante one-hot encoding), apilando las barras correspondientes a las obras donde el mismo personaje es protagonista. La visualización alternativa para palabras no se realizó porque es análoga.

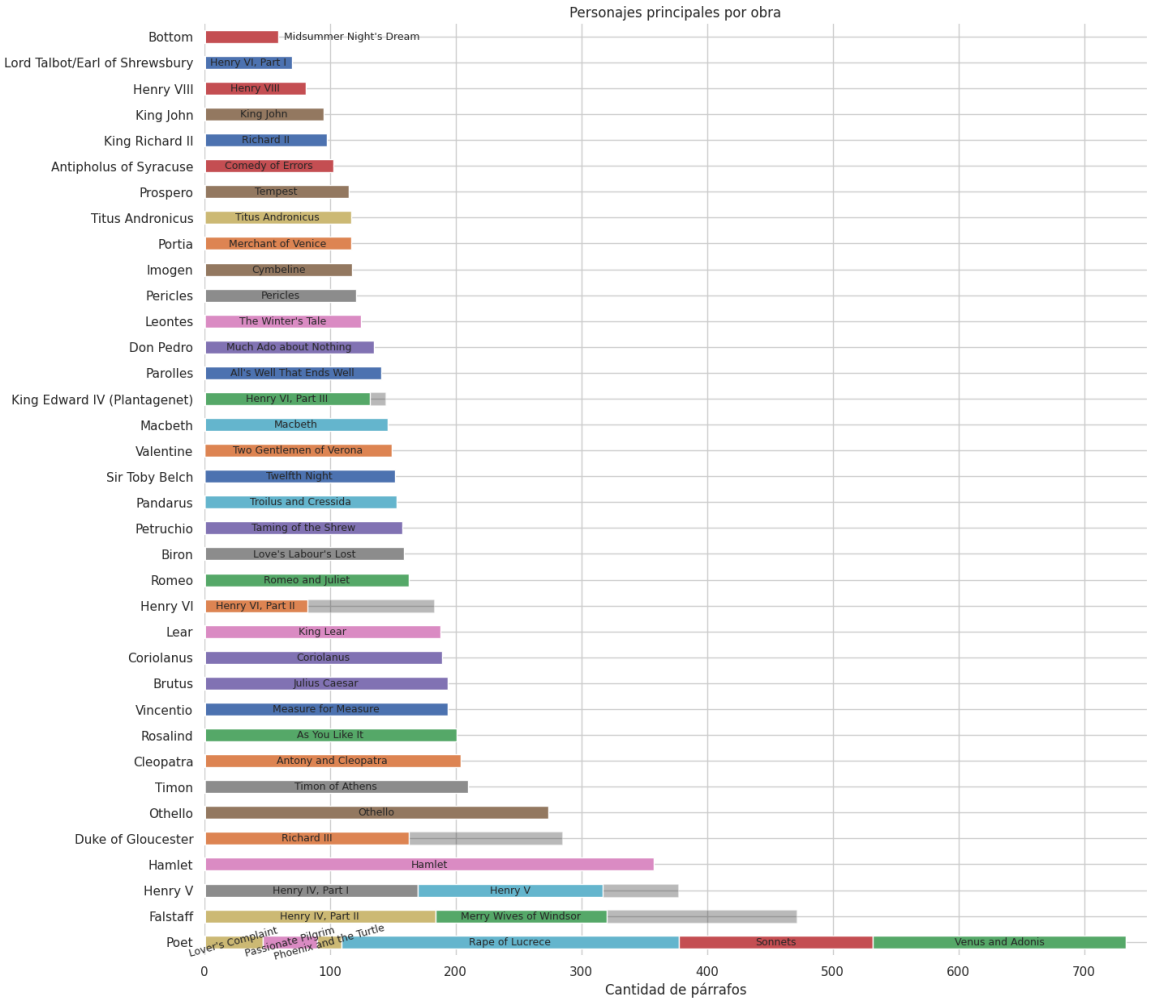


Figura 18: Gráfico de barras comparativo de la cantidad de diálogos (párrafos) dichos por los personajes con más intervenciones en cada obra de William Shakespeare. La cantidad de diálogos asignados al personaje de nombre que se indica a la izquierda se indica de acuerdo al largo de la barra asociada a cada obra (el título de la obra se escribe en el centro de su barra). Un personaje puede ser protagonista de más de una obra, en cuyo caso las barras se grafican apiladas. En gris se indicó el total de párrafos acumulados contando todos los trabajos de WS (fuera protagonista o no).

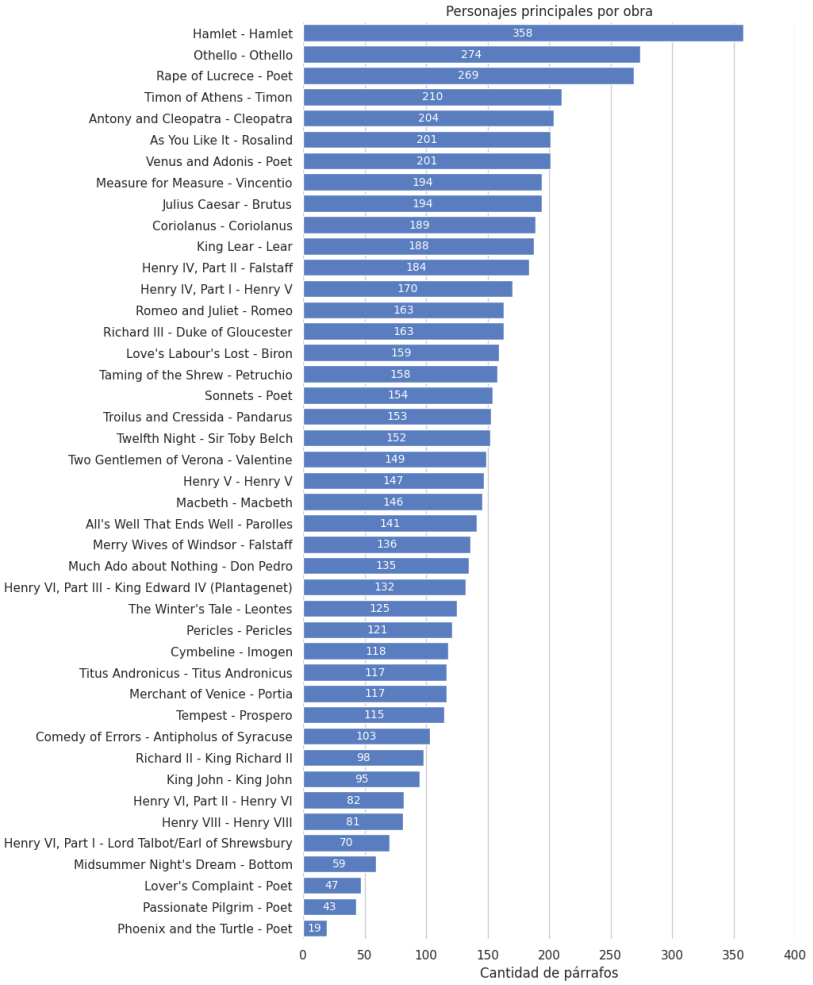


Figura 19: En la imagen puede observarse un gráfico de barras comparativo de la cantidad de diálogos (párrafos) dichos por los personajes con más intervenciones en cada obra de William Shakespeare. En azul oscuro se muestra la cantidad de diálogos asignados al personaje de nombre y obra que se indica a la izquierda (el largo de la barra se escribe en el centro de la misma).

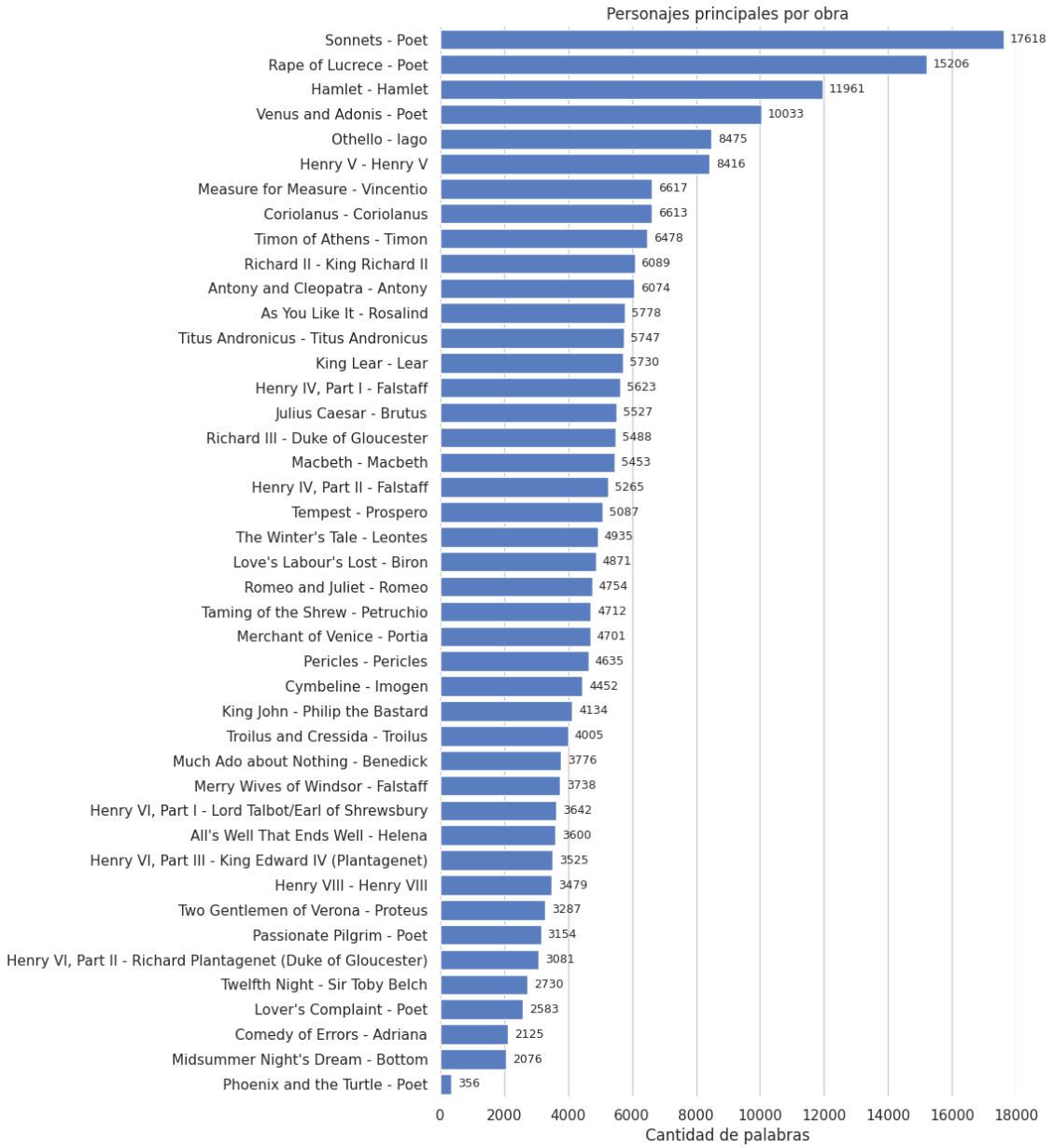


Figura 20: En la imagen se muestra un gráfico de barras comparativo de la cantidad de palabras asignadas a los personajes con más intervenciones más verbosas en cada obra de William Shakespeare. En azul oscuro se muestra la cantidad de palabras asignadas al personaje de nombre y obra que se indica a la izquierda (el largo de la barra se escribe a la derecha de la misma).

4- Cronología de obras

Con los datos de la tabla *works* se decidió contabilizar la cantidad de obras que fueron creadas por año y de esta manera ver si existía algún tipo de tendencia.

Al calcular la cantidad de obras por año se obtuvo que la máxima cantidad de obras publicadas fue de 4 en el año 1594. Al listar la cantidad de obras por año se encontró que en el año 1603 no se realizó ninguna obra.

En la Figura 21 se muestra gráficamente la cantidad de obras realizadas por año. A simple vista, parecería que su producción literaria se concentra mayormente en la primera mitad del período, tendiendo a decrecer hacia la segunda mitad (parece haber un quiebre en el ritmo de escritura luego de la pausa de 1603).

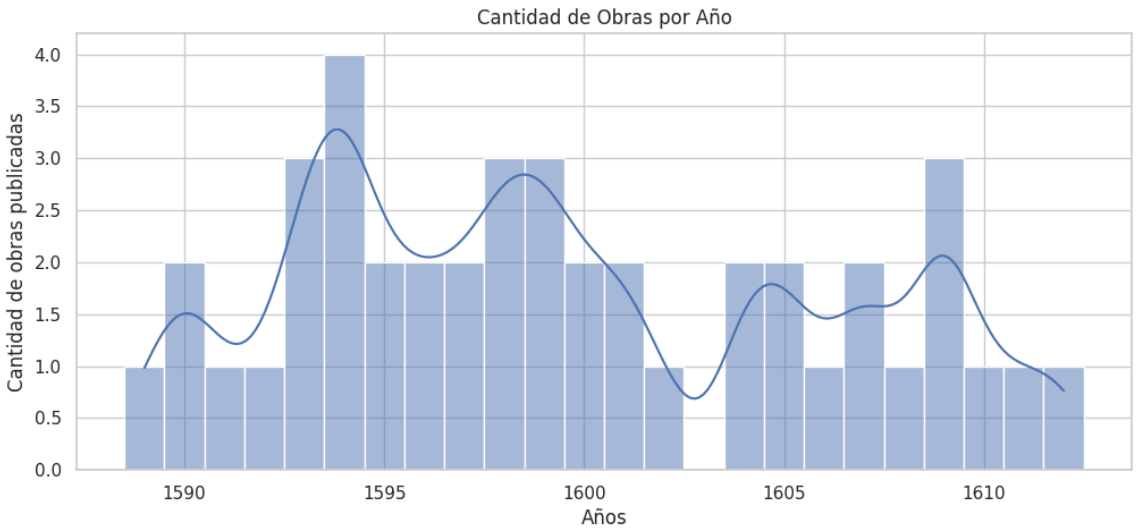


Figura 21: Gráfico de barras comparativo de la cantidad de obras publicadas por William Shakespeare entre 1589 y 1612. El máximo número de trabajos alcanzados fue 4 en 1594, mientras que el mínimo se dio en 1603, no habiendo publicado ninguno.

Otro aspecto interesante que se obtuvo de los datos fue el género literario de sus obras. William Shakespeare se dedicó a escribir en 5 géneros distintos: Comedia, Tragedia, Historia, Poemas y Sonetos. De la totalidad de sus obras conocidas 14 pertenecieron a la comedia, 12 fueron históricas, 11 tragedias, 5 poemas y 1 soneto.

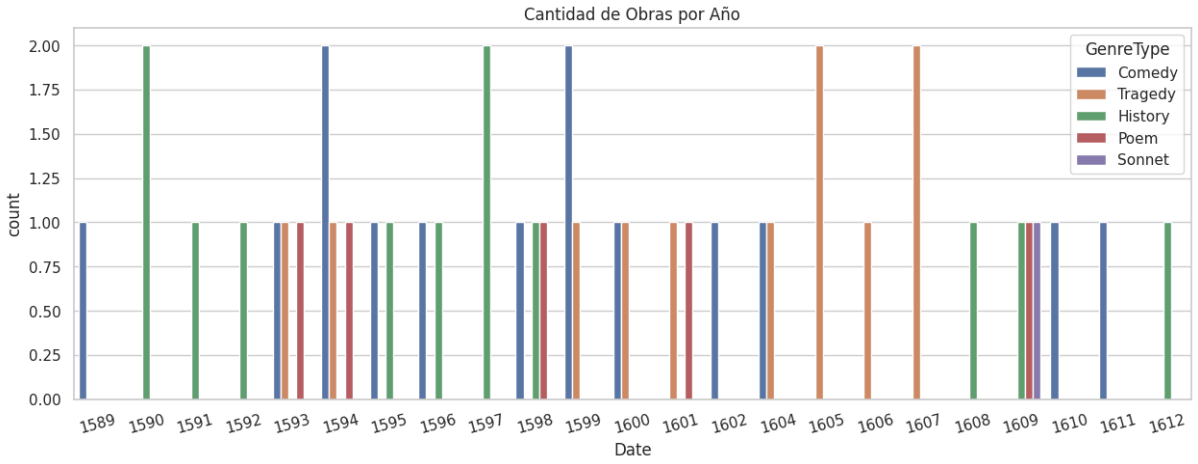
En la Figura 22 se pueden apreciar dos visualizaciones de las obras según su género a lo largo de los años.

En dicha imagen, se utilizaron dos abordajes distintos para explorar formas de representar la producción por género de Shakespeare: una que realizaba un histograma por cada género, reuniendo un máximo de 5 barras por año (1 por género), y otra que distinguía lo mismo pero aplicando horizontalmente las barras.

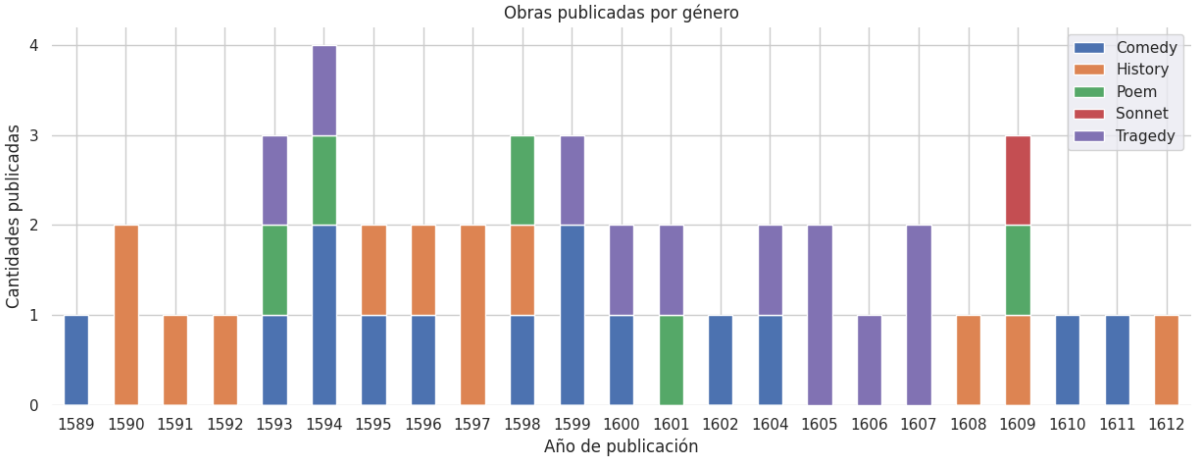
La primera visualización, de la Figura 22a, resulta ventajosa para comprender ya que se indica el género literario de acuerdo a un código de color y todas las barras comienzan desde el cero. Por eso, si una barra finaliza en la altura 2, significa que se publicaron 2 obras ese año bajo esa categoría. Sin embargo, dado que se tienen datos de 23 años de creación literaria, la cantidad de barras dibujadas es muy grande, y el ancho de cada barra, pequeño (no es una visualización escalable al aumentar los años). Adicionalmente, la ausencia de producción literaria para ciertos géneros en varios de los años aumenta la dificultad de separar visualmente a qué año corresponde cada barra.

En tanto, la segunda visualización, de la Figura 22b, es más “limpia” visualmente, ya que hay una sola barra por año, con colores cambiantes según a qué género pertenece cada creación. Además, esta forma de visualizar permite obtener fácilmente el valor total de obras publicadas para un año dado (se corresponde con la altura total de la barra). No obstante, hay algunos detalles que pueden hacer la lectura menos intuitiva: las barras indicando publicaciones en cierto año se “apilan” en lugar de superponerse, lo que implica que el “cero” de cada barra está por encima del límite de la barra anterior (para el mismo año), por lo que no es lineal extraer la cantidad de publicaciones para un género a simple vista. Por otro lado, puede resultar difícil agrupar visualmente las barras del mismo género, ya que para algunos años éstas comienzan desde el cero y para otros comienzan desde arriba de otras barras. Se considera que ésta no es una visualización que permita escalar en cantidad de géneros literarios.

En general, se cree que si bien ambas visualizaciones requerían de más trabajo, el uso de una u otra dependía en gran parte de a qué se quiere hacer referencia al mostrarlas y no había una que fuera intrínsecamente mejor que la otra.



(a)



(b)

Figura 22: Gráfico de barras comparativo de la cantidad de obras publicadas por William Shakespeare entre 1589 y 1612 según el género literario al que pertenecen. En (a) puede observarse una visualización a partir de un histograma con las barras de los géneros yuxtapuestas para un mismo año, mientras que en (b) puede verse lo mismo pero con las barras apiladas verticalmente.