UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Tarea 2 Introducción a la ciencia de datos

Análisis de las obras de William Shakespeare

Lucía Coudet - Grupo 2

Montevideo, Junio 2024

1 Introduccion

William Shakespeare fue un dramaturgo, poeta, y actor de origen inglés nacido en abril del año 1564 en *Warwickshire* al sureste de *Birmingham*, Inglaterra. Sus obras que incluyen tragedias como *Hamlet* y *Macbeth*, comedias como *El sueño de una noche de verano* y *La comedia de los errores* y obras históricas como *Enrique V* y *Ricardo III*, entre otras, son celebradas por su profundidad, ingenio, y habilidad para explorar la complejidad humana.

En este trabajo se utiliza la base de datos disponible en https://relational-data.org/dataset/Shakespeare. Se trata de una base de datos relacional que cuenta con información sobre las obras, los capítulos, los personajes y los párrafos para cada personaje de cada obra de Shakespeare.

2 Objetivo

El presente trabajo tiene por objetivo responder a las preguntas de la **Tarea 2** de la materia Introducción a la ciencia de Datos de la Facultad de Ingeniería (UdelaR) disponible en el siguiente repositorio de GitLab: https://gitlab.fing.edu.uy/maestria-cdaa/intro-cd/

El análisis exploratorio de datos ya ha sido realizado y se encuentra disponible en: https://github.com/lcoudet/Tarea1_introCD.

Este trabajose concentrará en el análisis de los párrafos de los siguientes personajes:

- **Antony** (de la obra *Antony and Cleopatra*)
- **Cleopatra** (de la obra *Antony and Cleopatra*)
- **Queen Margaret** (de la obra *Henry VI*)

3 Parte 1: Dataset y representación numérica de texto

3.1 Balance del dataset

En la tabla a continuación se presentan la cantidad de párrafos de cada uno de los personajes utilizados en el presente análisis. Según se observa en la misima, existe un desbalance en los datos ya que *Antony* tiene una proporción mayor que *Cleopatra* y *Queen Margaret*.

Table 1: Párrafos de cada personaje

Personaje	Cantidad	Proporcipn
Antony	253	0.4041534
Cleopatra	204	0.3258786
Queen Margaret	169	0.2699681

3.2 Conjunto de entrenamiento y conjunto de testeo

Para poder llevar a cabo la evaluación y validación de un modelo de aprendizaje estadístico supervisado es necesario partir el conjunto de datos que se tiene que se tiene en un conjunto de entrenamiento sobre el que se entrenan los modelos, y un conjunto de testeo sobre el cuál se mide el error.

Esto se hace ya que medir la performance predictiva del modelo utilizando los mismos datos que se usaron para entrenarlo puede llevar a una subestimación del error del mismo en particular en casos de sobreajuste.

Es decir, el modelo puede ser bueno prediciendo las observaciones utilizadas para ajustarlo pero malo prediciendo nuevas observaciones.

En este trabajo se propone utilizar 2/3 de la base de datos como conjunto de entrenamiento y 1/3 como conjunto de testeo.

Para ello, se utilizó la función *train_test_split* del paquete *scikit-learn* de Python, especificando el argumento *random state* en 42 para controlar la generación de números aleatorios y asegurar la reproducibilidad de los resultados y el argumento *stratify* = y para mantener las mismas proporciones de los valores de Y en el conjunto de entrenamiento y en el conjunto de prueba, evitando desbalances en uno y otro.

En la Tabla 2 se presentan las proporciones de párrafos de cada personaje en el conjunto de entrenamiento y en el conjunto de testeo. Como puede observarse en la misma, los párrafos de cada personaje se encuentran balanceados entre el conjunto de entrenamiento y el conjunto de testeo.

Table 2: Proporcion de parrafos de cada personaje en el conjunto de entrenamiento y de testeo

Personaje	Prop. entrenamiento	Prop. testeo
Antony	0.404	0.404
Cleopatra	0.326	0.324
Queen Margaret	0.269	0.271

3.3 Modelo Bag of Words (BoW)

El modelo *Bag of Words* (BoW) es un modelo de palabras de procesamiento de lenguaje natural (NLP) que tiene como concepto primario representar un texto como un grupo de palabras.

Los puntos clave de los modelos BoW son:

- **Creación del vocabulario**: el primer paso es crear el vocabulario el cual es un set de palabras únicas en todo el documento o set de datos. Este vocabulario actua como la variable x del modelo.
- **Representación vectorial**: cada documento es representado como un vector. El largo del vector es igual al largo del vocabulario. Cada posición en el vector corresponde a una palabra, y el valor asociado a cada posición es el número de veces que la palabra aparece en el documento.
- **Conteo de frecuencias**: típicamente el modelo utiliza el número de ocurrencias de cada palabra.

Las ventajas del modelo BoW son la simplicidad ya que es fácil de entender e implementar y la efectividad para muchos problemas incluyendo clasificación y clustering.

La principal desventaja es que no toma en cuenta el contexto (orden de las palabras y estructura), y tiene una alta dimensionalidad para casos de vocabularios grandes.

Se necesita una limpieza adecuada de *stopwords* y *stemming* para mejorar la performance del modelo.

Es un modelo sencillo y utilizado como la base para la implementación de modelos más complejos.

3.4 Matriz dispersa (sparse matrix)

Una matriz dispersa o *sparse matrix* es una matriz donde la mayoría de sus elementos son 0. En general son utilizadas en problemas de bases de datos grandes.

En este trabajo, utilizando el conjunto de entrenamiento se generó una matriz dispersa utilizando el vocabulario del modelo BoW y las pálabras de cada personaje.

se quitaron del análisis las *stopwords* del idioma ingés debido a que no tienen contenido semántico y los signos de puntuación.

3.5 n-gramma

Un *n-gramma* es una secuencia contigua de elementos de un texto o un discurso. En particular en el procesamiento de lenguaje natural los n-grammas se utilizan para modelar

y analizar las propiedades estadísticas de un texto, así como para predecir el siguiente elemento en una secuencia.

Los elementos de un *n-gramma* pueden ser carácteres, sílabas, palabras, o cualquier otra unidad, dependiendo de la aplicación de que se trate. Son simples y fáciles de interpretar, y además son efectivos en algunas tareas como la predicción básica de texto y modelado de lenguaje.

Algunos de los problemas y limitaciones es que cuando n es grande tienen un requerimiento elevado en recursos computacionales y puedne ignorar las dependencias a largo plazo y el significado semántico de las palabras.

3.6 Representación numérica Term Frequency - Inverse Document Frequency

La representación numérica *Term Frequency - Inverse Document Frequency* (conocida por sus siglas **tf-idf**) es un sistema de valoración de palabras en función de las veces que se muestran en el texto para luego hacer un cálculo inverso y descartar aquellas excesivamente repetidas en un conjunto de textos.

Esto sirve para detectar *stopwords* o palabras muertas que se descartan del análisis y finalmente relaciona qué documentos están más relacionados a ciertas palabras.

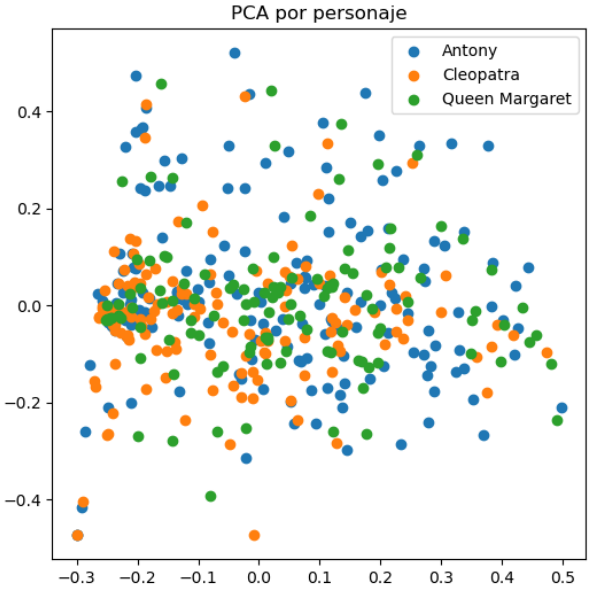
El objetivo de utilizar *tf-idf* en lugar de las frecuencias brutas de aparición de una palabra en un documento determinado es reducir el impacto de las palabras que ocurren con mucha frecuencia y que, por lo tanto, son menos informativos que las características que ocurren en un pequeña fracción.

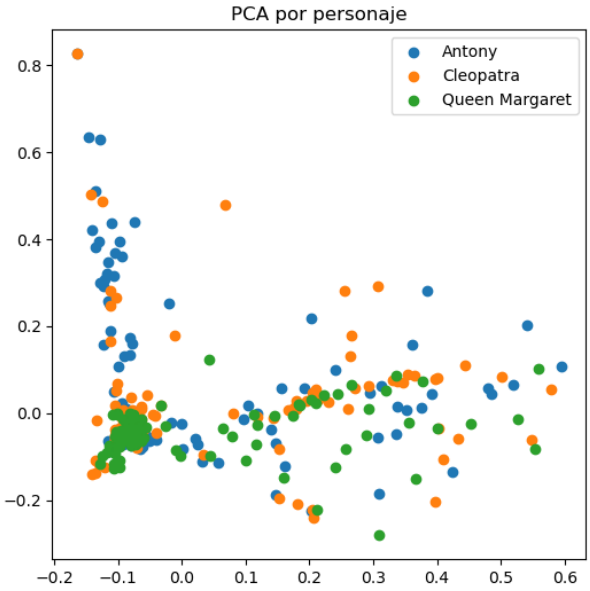
En este trabajo, se ajustaron los modelo con y sin utilización de *tf-idf*.

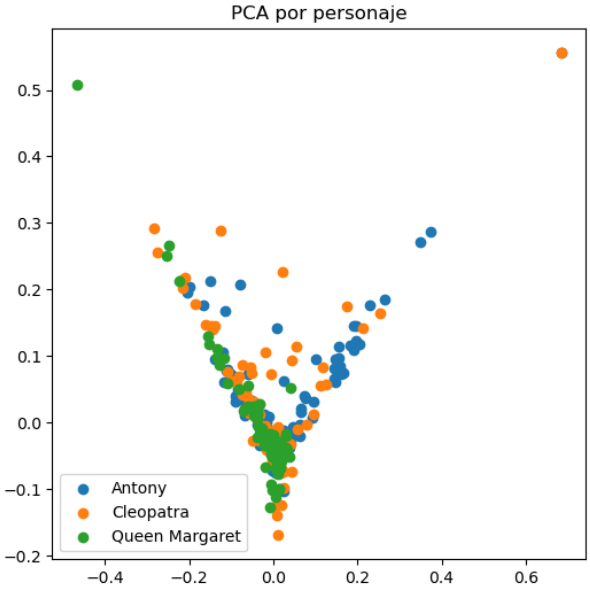
3.7 Análisis de componentes principales (PCA)

En este apartado se presentan los resultados utilizando las dos primeras PCA sobre los vectores *tf-idf*.

En la Figura 1 se presentan las dos primeras componentes principales sin hacer ningún filtrado adicional sobre el conjunto de entrenamiento y estableciendo *ngram_range=(1,1)*. En la Figura 2, se presentan los resultados filtrando las *stopwords* del idioma inglés y *ngram_range=(1,21)*. Por último, en la Figura 3 se presentan los resultados haciendo el filtrado de *stopwords*, *use_idf=True* y *ngram_range=(1,2)*.


PCA por personaje
Antony
Cleopatra
Queen Margaret
0.4
0.2
0.0
−0.2
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
0.5


PCA por personaje
Antony
Cleopatra
Queen Margaret
0.8
0.6
0.4
0.2
0.0
−0.2
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
0.5
0.6



Como vemos en las figuras anteriores, antes de filtrar las *stopwords* los puntos se encontraban mezclados unos con otros. Esto se relaciona con lo visto en la tarea 1 que las palabras más frecuentes eran las conectoras y no hay razones para pensar que un personaje contenga más palabras conectoras en sus párrafos que los otros.

Si se realiza el filtrado de las *stopwords* y además se considera un rango del n-gramma de (1,2) es posible identificar mas separadas las nubes de puntos. No obstante la nube de puntos mas concentrada puede indicar que se está perdiendo varianza explicada por las PCA.

3.7.1 Varianza explicada

A continuación se presenta la tabla con la varianza explicada por las componentes principales en cada uno de los casos presentados anteriormente.

De acuerdo a las Figuras observadas, las nubes de puntos se veían más concentradas en las figuras 2 y 3 respecto a como se veía en la figura 1, lo cual sugiere que las 2 componentes principales captaban cada vez menor varianza de los datos.

Table 3: Varianza explicada PCA

Primera componente	Segunda componente
0.0430826	0.0268973
0.0298589	0.0186309
0.0084401	0.0079330

Por lo tanto, utilizar solamente las dos primeras componentes principales para separar a los personajes resulta insuficiente, ya que la varianza explicada por las dos primeras componentes principales en todos casos es muy poca.

Por otro lado, si se incrementa la cantidad de componentes principales consideradas se incrementa la varianza explicada. Si se utilizan las 100 primeras componentes principales, la varianza explicada total por la suma de todas es de aproximadamente el 70%. No obstante utilizar tantas componentes no resulta práctico.

Si se miran solamente las 10 primeras componentes principales, la varinza explicada es de del 22.5%. Por su parte, las 5 primeras explican el 14%.

4 Entrenamiento y evaluación de modelos

4.1 Multinomial Naive Bayes

El modelo *Multinomial Naive Bayes* es un algoritmo de aprendizaje estadístico basado en el teorema de *Bayes*. Es comunmente utilizado en problemas de clasificación donde se tienen datos de tipo discreto, por ejemplo en el conteo de palabras.

El término *naive* refiere a que uno de los supuestos del modelo es la independencia entre las variables que representan un conteo o una frecuencia. Por su parte, el término *multinomial* refiere a la distribución de los datos asumida por el modelo: la distribución multinomial.

Aprendiendo de los datos, el modelo asigna una probabilidad a posteriori de pertenencia a una clase.

4.2 Medidas de la performance predictiva del modelo

Accuracy

Se define como la proporción de predicciones correctas realizadas por el modelo sobre la totalidad de las predicciones.

Si bien es una medida muy sencilla de entender, una de las principales desventajas es que en datos muy desbalanceados muchas veces se obtienen valores muy altos del *accuracy* cuando en realidad el modelo solamente es bueno prediciendo sobre la clase más frecuente en los datos.

En estos casos, el *recall* o el *F1 score* son métricas alternativas para obtener una mejor aproximación de la performance del modelo.

Recall

El *Recall* o *sensibilidad* se calcula como la tasa de verdaderos positivos. Se utiliza especialmente cuando hay un interes de minimizar falsos negativos. Es una aproximación a la capacidad del modelo para identificar correctamente a las instancias positivas.

F1 score

El *F1 score* es una medida que combina el *accuracy* y el *recall*. Se calcula como la media armónica de ambas.

Se utiliza la media armónica en lugar de la media arotmética ya que penaliza más los valores extremos.

De esta manera, un *F1 score* alto indica que tanto el *accuracy* como el *recall* son buenos y que el modelo tiene un buen equilibrio entre estos dos aspectos.

4.3 Validación cruzada (cross validation)

La técnica de *validación cruzada* (*cross validation*) es una técnica que se utiliza para medir la performance predictiva de un modelo de aprendizaje estadístico, ayuda a mitigar los problemas de sobreajuste y proporciona una evaluación mas precisa sobre el desempeño del modelo.

En términos generales el proceso de validación cruzada consiste en:

1. Dividir los datos datos de entrenamiento en k subconjuntos (folds) aproximadamente del mismo tamaño.
2. Se selecciona uno de los k subconjuntos como conjunto de prueba o validación y los restantes k-1 subconjuntos se utilizan para entrenar el modelo.
3. Se repite este proceso k veces para cada uno de los subconjuntos de datos.
4. Se promedian los resultados.

Existen diferentes variantes del procedimiento de validación cruzada:

- Validación cruzada k-folds
- LOOCV: leave one out cross validation (k = 1)
- Leave p out cross validation
- entre otras

Por otro lado, los algoritmos de aprendizaje estadístico tienen cantidades que deben ser fijadas previo a entrenar el modelo. A estas cantidades se les llama *hiperparámetros*. Un ejemplo es el número de árboles a ajustar en un algoritmo de *random forest*, o la cantidad de variables seleccionadas al azar en cada partición.

4.4 Ajuste del modelo

Se ajusto el modelo *Multinomial Naive Bayes* sobre los datos de entrenamiento. Para ello, se considero el filtro de las stopwords en ingles, tf-idf = True y un rango del n-gramma de 1,2.

En la Tabla 1 se presentan los valores de *accuracy* en el conjunto de entrenamiento y en el conjunto de testeo. Como puede apreciarse, el valor del *accuracy* utilizando los datos de testeo es sensiblemente inferior al valor en los datos de entrenamiento. Esto es un indicio de que el modelo podria estar sobreajustado a los datos de entrenamiento.

Table 4: Accuracy en datos de entrenamiento y datos de testeo

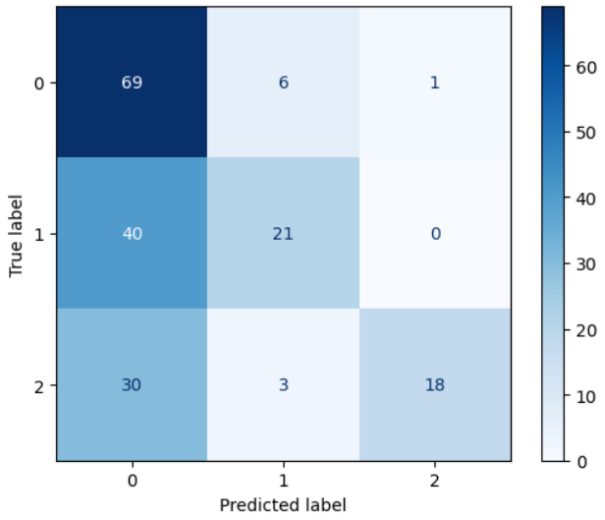
Accuracy entrenamiento	Accuracy testeo
0.96347	0.57447

Para poder tener una visión más integral de la performance predictiva del modelo, se utiliza una visualización a partir de la matriz de confusión, la cual es una tabla que compara los valores predichos por el modelo con los valores reales y es ampliamente utilizada en problemas de clasificación.

Para el caso de problemas binarios con etiquetas positivo y negativo, la matriz de confusión se compone de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN). A partir de ella se pueden calcular las medidas anteriormente mencionadas: accuracy y recall.

El accuracy se obtiene de la siguiente manera: (TP+TN) / (TP + TN + FP + FN). Por su parte, el recall se obtiene como sigue: TP / (TP + FN).

A continuación se presenta la matriz de confusion construida utilizando el conjunto de testeo. Como se observa en la misma, el modelo es mejor prediciendo en la clase con más frecuencia en los datos. Esto puede deberse al desbalance de cada categoría existente en los mismos.



4.5 Seleccion del modelo

Se implementó la técnica de validación cruzada para seleccionar al mejor modelo. Para ello se utilizó la función *StratifiedKFold* del paquete *sklearn*. Esta función permite realizar k-folds estratificados, manteniendo el porcentaje muestreado en cada clase.

En particular, se realizo validación cruzada con 4 folds y se especificó la semilla inicial en 42, de manera de mantener la reproducibilidad de los resultados.

Se ajustaron diferentes modelos para las siguientes combinaciones de los hiperparámetros:

- stopwords
- n-gramma
- tf-idf

A continuación se presentas los resultados obtenidos:

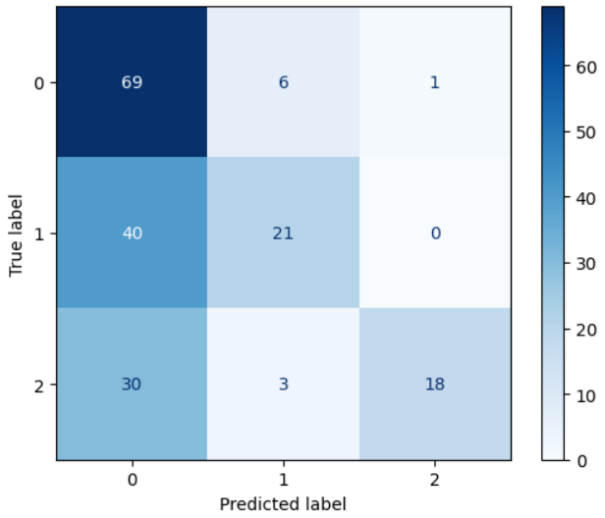
```
acc=0.4364 params={'stop_words': None, 'ngram': (1, 2), 'idf': True}
acc=0.4909 params={'stop_words': None, 'ngram': (1, 2), 'idf': True}
acc=0.5046 params={'stop_words': None, 'ngram': (1, 2), 'idf': True}
acc=0.4495 params={'stop_words': None, 'ngram': (1, 2), 'idf': True}
acc=0.4364 params={'stop_words': None, 'ngram': (1, 1), 'idf': False}
acc=0.4545 params={'stop_words': None, 'ngram': (1, 1), 'idf': False}
acc=0.4954 params={'stop_words': None, 'ngram': (1, 1), 'idf': False}
acc=0.4220 params={'stop_words': None, 'ngram': (1, 1), 'idf': False}
acc=0.5455 params={'stop_words': 'english', 'ngram': (1, 2), 'idf': True}
acc=0.5727 params={'stop_words': 'english', 'ngram': (1, 2), 'idf': True}
acc=0.5505 params={'stop_words': 'english', 'ngram': (1, 2), 'idf': True}
acc=0.5229 params={'stop_words': 'english', 'ngram': (1, 2), 'idf': True}
```

Por lo tanto, el modelo con mayor accuracy en el set de validación es el ajustado el inicialmente y tiene los siguientes hiperparámetros:

- stopwords = english
- rango del n-gramma = 1,2
- idf = True

El valor del accuracy en el conjunto de entrenamiento (incluyendo los datos para la validación) es de 0.96347 mientras que en el conjunto de testeo es de: 0.5744. Dado que el accuracy en el conjunto de testeo es muy inferior al del conjunto de entrenamiento, el modelo podría estar sobreajustado.

La matriz de confusion es:



Como comentado anteriormente, el modelo seleccionado tiene un accuracy muy bajo, esto puede deberse a la limpieza de la base de datos y tambien al modelo seleccionado el cual tiene algunas limitaciones.

En primer instancia, el modelo es muy sensible a la calidad de los datos.Si los datos son ruidosos, incompletos, desequilibrados o contienen características irrelevantes, el algoritmo Bayes ingenuo puede producir resultados inexactos o sesgados.

En segundo lugar, como fue comentado anteriormente el modelo tiene en su construcción el supuesto de que las características son condicionalmente independientes dada la etiqueta de la clase. Esto significa que la presencia o ausencia de una característica no afecta la probabilidad de otra característica dada la clase. Sin embargo, esta suposición a menudo se viola en los datos del mundo real, donde las características pueden estar correlacionadas o depender unas de otras. Por ejemplo, en la clasificación de textos, la aparición de ciertas palabras puede depender del contexto o del orden de las palabras anteriores. Para superar esta limitación, debe usar modelos más complejos que puedan capturar las dependencias entre las características

En tercer lugar, lo que se conoce como el problema de frecuencia 0, que se da cuando una característica no aparece en los datos de entrenamiento. Esto puede dar lugar a una

probabilidad cero para ese valor de entidad o etiqueta de clase, lo que puede afectar a la predicción o clasificación. Por ejemplo, si está utilizando Bayes ingenuo para clasificar los correos electrónicos como spam o no spam, y encuentra una palabra nueva que no está en su vocabulario, el algoritmo Bayes ingenuo asignará una probabilidad cero a esa palabra, lo que puede sesgar los resultados.

Por último, el modelo *naive bayes* no es adecuado para tratar características contínuas como los son los números, solo puede trabajar con características discretas para asignar una probabilidad. Si las características son continuas, es posible que el algoritmo Bayes ingenuo no pueda estimar la distribución de probabilidad de manera precisa o eficiente.

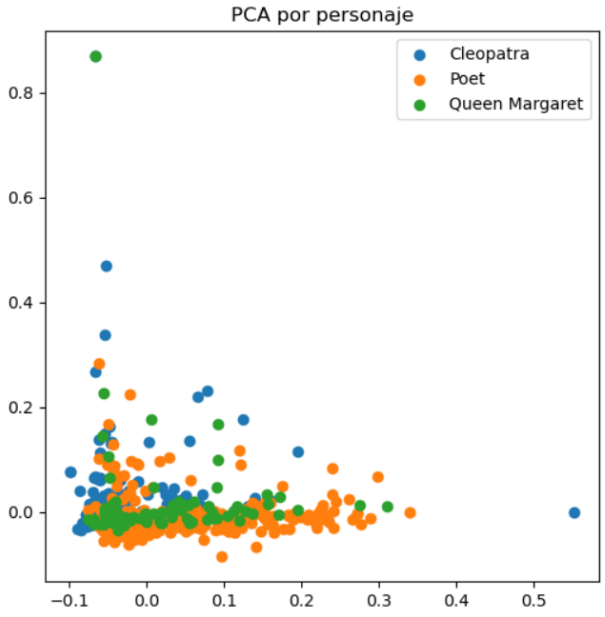
5 Cambio de personaje

Para poder observar los efectos del desbalance en los datos en un modelo de clasificación como es el *Multinomial Naive Bayes*, se realizó el análisis anterior utilizando al personaje Poet en lugar de Antony. Como fue mostrado en la Tarea 1, Poet es el personaje con más párrafos en los datos. En la tabla a continuación se presentan la cantidad de párrafos de cada personaje.

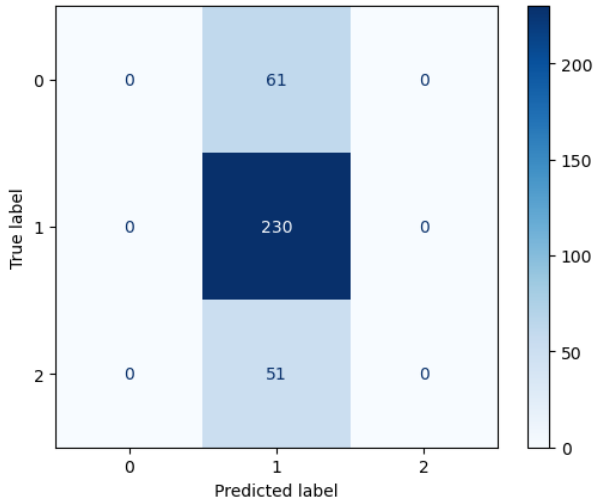
Table 5: Cantidad de párrafos de cada personaje

Personaje	Cantidad
Poet	766
Cleopatra	204
Queen Margaret	169

En la figura a continuación se presentan los resultados PCA haciendo el filtrado de *stopwords*, *use_idf=True* y *ngram_range=(1,2)*.



Y la matriz de confusión



Como vemos, el modelo predice solamente a la categoría Poet. Esto se debe al gran desbalance en los datos siendo Poet, la categoría con mayor frecuencia en los datos.

6 Otras técnicas para extraer características de un texto

La extracción de las características de un texto es un aspecto crucial del procesamiento de lenguaje natural. Existen diversas técnicas para convertir el texto en una repre-

sentación numérica.

Una alternativa a las utilizadas en el presente trabajo es el **embedding**.

En términos generales, esta técnica permite convertir palabras o frases en vectores de números en un espacio de alta dimensión. Dichos vectores son diseñados de tal manera que capturan las relaciones semánticas y contextuales entre las palabras.

Uno de los tipos más utilizados de *embedding* es el *fasttext*, el cual fue desarrollado por *Facebook* y reliza predicciones tanto de una palabra según el contexto como del contexto según la palabra. Como considera sub-palabras, suele ser bueno con palabras poco frecuentes.

7 Comentarios finales

El modelo *Multinomial Naive Bayes* ajustado luego de realizar el procedimiento de validación cruzada arroja un valor del accuracy de 0.57744 sobre el conjunto de testeo. No obstante, dentro del conjunto de entrenamiento este valor asciende a 0.96347. Como fue mencionado, esto puede deberse a un sobreajuste en los datos.

Por otro lado, este modelo tiene el supuesto de independencia entre las características, lo cual no suele cumplirse en particular para el caso del análisis de texto, ya que la aparición de ciertas palabras depende de la palabra anterior.

Cuando se analiza el cambio de un personaje, en particular *Antony* por *Poet*, el desbalance en los datos se hace mayor y el modelo no logra hacer ninguna predicción que no sea *Poet*. Si bien esto da un *accuracy* muy elevado, se sugiere en ests casos utilizar otras medidas como el *recall* o el *F1 scrore* para evaluar la bondad de ajuste.

Por último, para superar las limitaciones encontradas en este trabajo, se sugiere para futuras investigaciones la implementación de modelos más complejos que puedan capturar las dependencias entre las características y que no sean tan sensibles al desbalance en los datos. Asimismo, se sugiere utilizar una representación numérica que considere el contexto y no las palabras de manera aislada.