Andrés Takashi Bravo Yaguchi – 201911714, Sara Plazas Copete – 201821723, Kevin Steven Gámez Abril – 201912514

PROYECTO I – ETAPA I

1. Entendimiento del negocio y enfoque analítico

1.1. Oportunidad/problema negocio

Para toda empresa e industria es de gran utilidad tener una buena comunicación y recibir retroalimentación del cliente, pues le permite entender la experiencia real del cliente y su perspectiva frente al producto/servicio ofrecido. Actualmente existen muchas facilidades para obtener grandes cantidades de datos acerca de la opinión de los clientes de cada empresa. No obstante, aunque la información que contienen estos datos puede ser de gran utilidad, el procesamiento de la retroalimentación recibida se convierte en todo un reto cuando se trata de una gran cantidad de comentarios, los cuales tomaría mucho tiempo leer y procesar uno a uno. A raíz de lo anterior, la utilización de la analítica de textos se convierte en una oportunidad muy grande para la empresa, pues le permite automatizar el procesamiento de grandes cantidades de comentarios para extraer información importante. Para este caso de estudio especifico, este campo ayudará a determinar si las reseñas de ciertas películas son positivas o negativas a través del análisis de sentimientos. Así pues, la implementación de estas herramientas permite a la empresa tener una entrada de información valiosa con rápida interpretación, posibilitando así que la empresa pueda tomar decisiones más fundamentadas y acertadas. Algunas de las oportunidades concretas que este desarrollo permitiría son: Identificación y gestión rápida de quejas, identificación de películas más importantes para mantener, identificación de productos o líneas de negocio que podrían dejar de brindarse sin afectar a los clientes.

1.2. Enfoque analítico

Con este proyecto se pretende realizar predicciones categóricas (positivo/negativo) mediante el procesamiento del lenguaje natural (reseñas). Lo anterior mediante el modelamiento de algoritmos de machine learning de tipo supervisado.

1.3. Organización y rol que se beneficiaría con la oportunidad

Está claro que este modelo podría beneficiar a un gran número de empresas pues, como se mencionó previamente, la retroalimentación del cliente puede llegar a ser una entrada de información muy valiosa para cualquier empresa de cualquier industria. Esto es importante pues así la empresa puede entender tanto aquello que está haciendo bien como lo que podría mejorar en su operación. Cabe resaltar que entre más grande sea la empresa (o entre más datos se deseen analizar), estas herramientas podrías representar una mayor ayuda por las ventajas de escalamiento que tiene el procesamiento automático de las reseñas. Para este caso en particular (reseñas de películas), se podrían ver mayormente beneficiadas las personas y entidades que trabajan en la industria del cine como productoras, directores, actores, empresas de exhibición y distribución de películas, etc. Por ejemplo, un canal de televisión podría analizar las reseñas de las películas y decidir transmitir la película con mayor cantidad de reseñas positivas, o tal vez, tras clasificar las opiniones del público, una productora identifique que las películas animadas suelen tener una mejor recepción y decida invertir más dinero en el departamento de animación. En realidad, los potenciales beneficiados son de múltiples naturalezas y en gran cantidad, pues el desarrollo da la

oportunidad de obtener información valiosa a todo aquel interesado en comprender mejor las películas o a las personas que consumen contenido. Finalmente, es importante resaltar que esta implementación también beneficiaría a los clientes de muchas empresas, pues las empresas los conocerían mejor y podrían ofrecerles un contenido más acorde a sus gustos.

1.4. Técnicas y algoritmos a utilizar

Como se mencionó previamente, se optó por construir un modelo de aprendizaje supervisado, específicamente, utilizar los algoritmos de Logistic Regression, Random Forest y Multinomial Naive Bayes. Esto se debe a que estos algoritmos nos permiten predecir la categoría a la que un elemento pertenecerá basándonos en una serie de entradas, las cuales en este caso son los textos de las reseñas.

2. Entendimiento de los datos

Para comprender mejor los datos se evaluaron diferentes características de la tabla de datos, como su tamaño, disposición y valores, entre otros. En primera instancia, después de cargar los datos al proyecto de Python e imprimirlos se pudo notar que la primera columna, llamada *Unnamed: 0*, consistía en valores para enumerar cada fila, por lo cual se ajustó la carga de datos para ignorar dicha columna. Luego, se obtuvieron las dimensiones de la tabla de datos, tanto el número de filas como el de columnas, y los tipos de las variables ahí mostradas. Con respecto al tamaño de la tabla se obtuvo que esta consta de 5000 filas y 2 columnas, cosa que nos podría dar indicios de que la tabla fue cargada correctamente pues el número de columnas coincide con el número de variables incluidas en el diccionario suministrado. Posteriormente, se obtuvo el tipo de dato de cada columna, esta información se puede observar en la Ilustración 1 de la sección de anexos.

Ahora bien, para realizar la exploración del contenido de los datos de entrada, especialmente las reseñas de las películas, se hizo uso de la librería pandas_profiling, específicamente de su función ProfileReport(). Dicha herramienta permite obtener información estadística de los textos como la longitud de cada comentario (en caracteres), las palabras más frecuentes, la longitud de la palabra más larga y de la más corta del comentario. A partir de estas estadísticas vemos que la longitud de los comentarios parece seguir una distribución normal, donde el comentario más corto tiene 60 palabras; esto en principio es un buen indicador sobre la calidad de los comentarios obtenidos, pues da indicios de que cada reseña tiene suficiente información para que el modelo a implementar pueda inferir si la reseña es positiva o negativa. Así mismo, como puede observar en la ilustración 2 en la sección de Anexos, el reporte generado por la librería nos permite identificar una alta frecuencia de palabras que no agregan información como preposiciones y conectores, adicionalmente, detectamos la presencia de caracteres no alfanuméricos en las reseñas de las películas (véase las ilustraciones 3 y 4 en la sección de Anexos), esta información es de vital importancia para tomar decisiones en la fase de preparación y limpieza de los datos.

Por otra parte, con respecto a la calidad de los datos, nos damos cuenta de que no hay entradas nulas en el DataFrame y que los valores de la columna ‘Sentimiento’ coinciden con lo indicado por el diccionario. Más aun, como puede evidenciar en la ilustración 5 en la sección de Anexos, esta columna se encuentra bien distribuida, es decir, existen suficientes datos de ambas categorías para ser estudiados. Finalmente, aunque concluimos que la calidad de los datos en términos generales es buena, también detectamos dos entradas que

se encuentran duplicadas y la presencia de algunas reseñas escritas en inglés, de las cuales nos tendremos que encargar en la siguiente sección.

3. Preparación de los datos

En este caso, la preparación de los datos se centró en transformar el contenido de las reseñas de las películas para poder incorporarlos a un modelo de aprendizaje automático. A pesar de que se encontraron problemas de unicidad y entradas en otros idiomas, estos problemas son fáciles de corregir al compararlos con la dificultad de trabajar con datos no estructurados, como lo son las reseñas de los clientes en lenguaje natural. Así pues, para llevar a cabo la preparación de los textos con mayor facilidad, se realizaron los cambios necesarios para asegurar la calidad de los datos tomando como base todos los otros problemas que se detectaron en la fase de entendimiento de los datos.

Primero, se eliminaron dos entradas que se encontraban repetidas en el DataFrame, las entradas duplicadas no aportan información adicional al modelo y en este caso representan menos del 0.1% de los datos, por lo que podemos eliminarlas sin problema. Posteriormente, se eliminaron 197 reseñas que estaban escritas en inglés, esto representa el 0,039% de los datos. Nuevamente, esta eliminación se puede hacer sin problema, pues la cantidad total de datos eliminados no es representativa y, se nos comunicó explícitamente que el objetivo del proyecto analizar reseñas escritas en español, por lo que podemos afirmar que las reseñas en otros idiomas se tratan de un error. Finalmente, recuerde que no se encontraron errores ni anomalías en la columna ‘Sentimiento’, por lo tanto, no es necesario realizar ninguna transformación sobre ésta. Con estas correcciones, continuamos con la siguiente etapa de la preparación de los textos, es decir, su respectiva limpieza, tokenización, normalización y lematización.

En este caso empezamos con la tokenizacion y lematización de los textos, que consiste en separar cada reseña en partes o tokens que guardan un sentido por sí mismas (en este caso palabras) y posteriormente agrupar las partes que tengan un significado similar bajo una misma etiqueta o lema’. Este proceso se realizó utilizando la librería Stanza, desarrollado por el grupo de procesamiento de lenguaje natural de la Universidad de Standford. Es importante mencionar que mientras se realizaba este procedimiento también se iban transformando las palabras para que solo contuvieran letras minúsculas, pues la librería de Stanza también nos permite hacer esto de forma fácil. Generalmente se eliminan las palabras que no aportan información adicional antes de realizar el proceso de lematización, sin embargo, debido a las funciones ofrecidas por la librería Stanza, es más fácil realizar el proceso de lematización y después de esto, eliminar los lemas que agrupan aquellas palabras que no aportan información.

Una vez hemos separado cada reseña en palabras y se han agrupado aquellas con significado similar, eliminamos todos los signos de puntuación y las palabras que no aportan información que hayan quedado. Es importante aclarar que, como se están trabajando reseñas escritas en español, en ningún momento se han eliminado las tildes de las palabras, pues estas determinan su significado. Por ejemplo, si tenemos las palabras ‘jugo’ y ‘jugó’ podría ser correcto asignarles los lemas ‘jugo’ y ‘jugar’ respectivamente, pero no sería correcto eliminar las tildes y agruparlas bajo la misma palabra ‘jugo’. Teniendo esto en cuenta, eliminamos todos los lemas correspondientes a categorías gramaticales que en el lenguaje natural ayudan a conectar las ideas, pero no aportan información valiosa para el

análisis de sentimientos, adicionalmente, eliminamos las palabras que son muy frecuentes en todas las reseñas y tampoco aportan información valiosa sobre cada reseña en particular. Para esto hacemos uso del paquete Stopwords de la librería Ntlk, el cual nos permite acceder a un listado de palabras que suelen ser comunes a todos textos escritos en español y que se puede considerar que no aportan información. Tras este paso, procedemos a eliminar todos los signos de puntuación y símbolos no alfanuméricos de las reseñas, para conservar únicamente los lemas (palabras) que tienen su propia interpretación semántica. Para terminar, utilizamos la librería num2words para remplazar los números por su representación en texto, esto facilitara nuestro trabajo en la fase de modelamiento.

Aun después de esto, la representación de los textos aun no es adecuada para entrenar un modelo de análisis de sentimientos. Consecuentemente, hacemos uso de vector-embedding para codificar cada palabra de las reseñas como un vector en un espacio vectorial. Específicamente usamos la librería sci-kit learn para implementar el modelo Tf-Idf y construir una representación de los textos en una matriz numérica que tiene en cuenta las frecuencias relativas con las que aparecen cada una de las palabras en las reseñas.

4. Modelamiento y evaluación

En esta fase se implementaron tres algoritmos diferentes para generar el modelo de clasificación. Estos tres algoritmos fueron medidos bajo las mismas métricas y se tomó el modelo con mejores resultados para presentar al cliente; las métricas utilizadas son las siguientes: accuracy, precision, recall y score f1. A continuación, se presenta una breve explicación de cada implementación:

4.1. Multinomial Naive Bayes (Sara Plazas)

Multinomial Naive Bayes es un método bayesiano de aprendizaje supervisado, el cual busca predecir la clasificación de una entrada determinada a través de la estimación de la probabilidad de ocurrencia basado en eventos pasados. Es un algoritmo altamente usado en el análisis de textos y procesamiento de lenguaje natural al recibir las palabras más útiles de un texto y con ayuda del teorema de Bayes calcular la probabilidad de que pertenezca a cada posible categoría. Adicionalmente, tiene la gran ventaja de ser de fácil implementación.

La implementación de Multinomial Naive Bayes se realizó utilizando la clase MultinomialNB de la librería SciKit-Learn en Python. Asimismo, con ayuda de la función train_test_split se dividió el conjunto de datos en dos subconjuntos correspondientes a entrenamiento y validación de forma aleatoria. El conjunto de entrenamiento contiene el 80% de los datos y hacen referencia a los “eventos pasados” los cuales serán tomados como base para predecir la clasificación de los datos del conjunto de validación. En este punto es importante recordar que es necesario realizar la búsqueda de hiper-parámetros, para lo cual se utilizó el método GridSearchCV de la librería de sklearn.model_selection pues permite probar varios parámetros y comparar los modelos resultantes en diferentes particiones del conjunto de entrenamiento. Después de ingresar los valores a probar y correr la comparación entre modelos se obtiene que el mejor modelo se alcanza con un Alpha de 1 y un valor de falso para el parámetro fit_prior. Asi pues, se corre una vez más el modelo final con dichos parámetros y se utiliza el conjunto de validación para el cálculo de las métricas de desempeño con el fin de estimar la capacidad real del modelo.

La Tabla 1 resume los resultados obtenidos para el modelo final. Como se puede ver, este método retorna una precisión del 85% en promedio, un recall de 84% en promedio y un score f1 de 84.5% en promedio, además, se obtiene una exactitud del 84%. Estas son métricas muy buenas que nos pueden dar indicios de que el modelo se realizó de forma apropiada y podrá hacer predicciones acertadas al ingresar nuevos datos de entrada.

4.2. Logistic Regression (Kevin Gamez)

La regresión logística es un algoritmo de aprendizaje supervisado usado para clasificar datos en dos o más categorías. El funcionamiento de esta se basa en una función logística, que también es conocida como sigmoidal. Esta función toma una entrada numérica y produce una salida en el rango 0 a 1. Esta salida es interpretada como la probabilidad que la entrada pertenezca a una de las categorías. Si la probabilidad es mayor que un umbral predefinido se clasifica la entrada en la categoría correspondiente. La regresión logística se entrena con un conjunto de datos de entrenamiento etiquetados. Durante el entrenamiento, el algoritmo ajusta los pesos y sesgos de la función para minimizar el error entre las probabilidades predichas y las categorías reales.

La implementación de este método se realizó usando la clase LogisticRegression de SciKit-Learn en Python. Así mismo, con ayuda de esta librería separamos los datos en dos subconjuntos de entrenamiento y un conjunto de prueba. Donde la proporción del conjunto de prueba es del 20% del conjunto original y se utiliza la semilla random_state=0 para garantizar reproducibilidad de los resultados. Luego se crea un objeto TfidfVectorizer que se encarga de convertir los textos de entrada en vectores numéricos para que puedan ser procesados por el modelo de regresión logística. Después, usamos un GridSearchCV para hacer validación cruzada y búsqueda de hiper-parámetros del modelo. Los hiperparámetros que se prueban son penalty, C y solver, que controlan la regularización, el parámetro de costo y el método de optimización, respectivamente.

La Tabla 2 resume los resultados obtenidos para el modelo final. Como se puede ver, este método retorna buenas métricas que nos hacen creer que el modelo tiene un buen desempeño. El algoritmo de regresión logística que se ha utilizado tiene una precisión de predicción del 85.74% en el conjunto de prueba, lo que indica que el modelo es capaz de clasificar correctamente alrededor del 85% de los comentarios. El Classification Report muestra que el modelo tiene un buen equilibrio entre la precisión y el recall para ambas clases (negativo y positivo), con un F1-score de 0.86 para ambas clases.

4.3. Random Forest (Andrés Bravo)

El método Random Forest, es un algoritmo de aprendizaje supervisado usado para tareas de clasificación basado en arboles de decisión. Un árbol de decisión es un modelo de aprendizaje supervisado que también es de gran utilidad para realizar procesos de clasificación, sin embargo, los modelos basados en árboles son más sensibles a los sesgos y a la selección inicial de los valores de entrenamiento, lo que también los hace más propensos a tener problemas de sobreajuste a los datos de entrenamiento. Así pues, Random Forest busca mejorar esto al construir múltiples arboles de decisión independientes y entrenarlos con un subconjunto de datos que también son aleatorios. En la fase de entrenamiento, cada árbol de decisión se ajusta a una muestra diferente de datos y produce una predicción de salida para cada conjunto de características de entrada. Finalmente, Random Forest promedia los árboles construidos teniendo en cuenta la información que

cada uno obtuvo, así se obtiene un modelo que explica mejor los datos y es más robusto. De hecho, al tener en cuenta múltiples arboles independientes, se tiene en cuenta posibles variaciones en los datos, lo que reduce el sesgo del modelo, así como los riesgos de sobreajuste, esto en general conduce a un modelo que genera mejores predicciones.

La implementación de este método se realizó usando la clase RandomForestClassifier de la librería SciKit-Learn en Python. Así mismo, con ayuda de esta librería se partió el conjunto de datos para crear un conjunto de entrenamiento con el 80% de los datos y un conjunto de prueba para posteriormente probar el modelo con el 20% de los datos restantes. Adicionalmente, se hizo uso de validación cruzada y búsqueda de hiper-parámetros con ayuda de los métodos KFold y GridSearchCV de la librería SciKit-Learn. Específicamente, se buscaron ajustar los parámetros que controlan la complejidad y precisión del modelo.

La Tabla 3 resume los resultados obtenidos para el modelo final. Como se puede ver, este método retorna una precisión del 83% en promedio, un recall de 83.5% en promedio y un score f1 de 83.5% en promedio, además, se obtiene una exactitud del 83%. Estas son métricas muy buenas que nos pueden dar indicios de que el modelo se realizó de forma apropiada y podrá hacer predicciones acertadas al ingresar nuevos datos de entrada.

5. Resultados

A continuación, se presenta un análisis de resultados de cada modelo implementado, en esta sección recibimos ayuda del experto en estadística Jhon Arciniegas, quien estuvo de acuerdo con la metodología seguida para obtener los resultados y nos comentó, que, en base a su experiencia, los resultados que obtuvimos son suficientes para sugerir implementar el proyecto en situaciones reales. La siguiente sección de resultados es elaborada en conjunto con el experto. Los resultados que se muestran son generados de la evaluación de clasificación binaria, es decir, (negativo y positivo) utilizando las métricas de precisión, sensibilidad (recall) y f1 score en cada algoritmo.

La precisión se calcula como la proporción de instancias clasificadas en cada categoría que realmente pertenecen a esta categoría. Por ejemplo, si la precisión conseguida para instancias clasificadas como negativas corresponde a un 83%, indica que del total de reseñas clasificadas como negativas, en realidad el 83% eran negativas. Por otra parte, el recall (sensibilidad) es la proporción de instancias en una categoría que fueron correctamente identificadas por el modelo. En este caso con la aplicación del algoritmo Naive Bayes, el recall para la clase negativa corresponde a un 88%, lo que permite entender que el modelo logró identificar correctamente el 88%; ésta es una métrica altamente utilizada cuando se tiene una categoría en la que es importante detectar todos los elementos que pertenecen a ella, aun si esto implica clasificar erróneamente otros elementos en esta categoría (por ejemplo, en detección de enfermedades es preferible tener algunos falsos positivos a omitir un caso grave de una enfermedad). Ahora, una forma de combinar las anteriores métricas es calculando el f1-score, que es una medida que combina la precisión y la sensibilidad (recall) generadas por el algoritmo en una sola métrica, esta medida permite evaluar de manera general la calidad del modelo.

Como se puede evidenciar en las tablas de los resúmenes con los mejores modelos en cada algoritmo, tenemos que los tres algoritmos tienen resultados muy similares en términos de precisión, recall y F1-score. En cuanto a la clasificación por sentimiento negativo, el algoritmo

LogisticRegression tiene la mayor precisión (0.87) y Random Forest tiene el mayor recall (0.83). En cuanto a la clasificación por sentimientos positivos, LogisticRegression y Native Bayes tienen la mayor precisión (0.85 y 0.87) respectivamente, mientras que Random Forest tiene el mayor recall (0.84). En cuanto al F1-score, todos los algoritmos tienen resultados muy similares, con valores que van desde 0.83 a 0.86, lo que indica que los tres algoritmos son igualmente eficaces en términos de la combinación de precisión y recall.

Dado que el objetivo del negocio es procesar grandes cantidades de textos para la extracción de conocimiento que apoye la toma de decisiones, es recomendable elegir un algoritmo que tenga un buen rendimiento en términos de precisión y recall. De los tres algoritmos implementados obtuvimos que la regresión logística tiene un mejor rendimiento en términos de precisión, recall y f1-score para ambas categorías, positivo y negativo, en comparación con Native Bayes y Random Forest. Además, la regresión logística es un modelo lineal que se puede interpretar fácilmente, lo que puede ser útil para la organización a la hora de extraer conocimiento.

Por lo tanto, se recomienda usar el modelo de regresión logística como algoritmo de clasificación para el análisis de sentimientos de comentarios de películas. Un caso en el que el algoritmo seleccionado podría beneficiar a una empresa de películas es en la clasificación de opiniones obtenida de usuarios sobre las películas ofrecidas. La empresa podría utilizar Logistic Regression para clasificar de manera automática las opiniones en positivas o negativas, esto permitiría identificar rápidamente las tendencias de opinión sobre sus películas lo que terminaría beneficiando a la empresa en:

Analizar las películas que tienen buena recepción al público: La empresa podría identificar rápidamente las películas que reciben críticas positivas y ajustar su estrategia de marketing

Ahorrar tiempo y recursos: Una vez se automatice la clasificación de opiniones, la empresa podría ahorrar tiempo y recursos en la clasificación manual de las opiniones, esto permitiría una respuesta más rápida a las tendencias del mercado.

6. Recomendaciones

1) **Utilizar el algoritmo de regresión logística:** Se recomienda utilizar dicho algoritmo pues fue el modelo que retorno las mejores métricas entre los tres algoritmos implementados. La implementación de este algoritmo puede representar una oportunidad muy valiosa para cualquier empresa pues ayuda a hacer una mejor toma de decisiones al tener un mayor sustento o fundamentación.
2) **Utilizar datos más amplios y diversos:** Fomentar la retroalimentación por parte de los espectadores para poder hacer uso de datos más amplios y diversos. Esto puede mejorar la capacidad del modelo para generalizar a nuevos datos y así conseguir una mejor precisión en las predicciones.
3) **Actualizar los datos:** Se recomienda estar actualizando constantemente los datos del proyecto, esto incluye tanto la base de datos de las reseñas como las condiciones bajo las cuales corre al algoritmo, es decir, los hiperparametros. Lo anterior para poder asegurar que el modelo se ajuste lo mejor posible a los nuevos datos de entrada y poder entrenar el algoritmo con nuevas palabras que se vayan popularizando y que se deberían tener en cuenta para futuras predicciones.

7. Trabajo en equipo

Integrante	Rol	Horas	Tareas realizadas	Desafíos	Puntaje
Andrés Bravo	Líder de proyecto	17	-Entendimiento del negocio y enfoque analítico -Entendimiento de los datos -Preparación de los datos -Algoritmo Random Forest -Informe	Realizar la preparación de los datos con herramientas que soporten el español	33.3
Kevin Gámez	Líder de analítica	17	-Entendimiento de los datos -Algoritmo Logistic Regression -Análisis del mejor modelo -Informe -Wiki	Seleccionar un algoritmo adecuado junto con la evaluación y la validación de este.	33.3
Sara Plazas	Líder del negocio	17	-Entendimiento del negocio y enfoque analítico -Entendimiento de los datos -Preparación de los datos -Algoritmo Naive Bayes -Informe	Aprender a manejar herramientas y conceptos nuevos para la preparación de los datos y obtención de las palabras más útiles	33.3

Después de una reflexión grupal se determinó que todos los integrantes del grupo aportaron al trabajo de su propia forma de manera equitativa y justa. Es por esto por lo que se tomó la decisión de repartir los puntos de forma igual entre los integrantes, es decir 1/3 para cada uno. Asimismo, creemos que es importante mantener las buenas prácticas que hemos adoptado como mantener una comunicación constante, entregar las tareas a tiempo y hacer una distribución razonable del trabajo. Con respecto a las oportunidades de mejora, se propone empezar los trabajos con un poco más de tiempo y así poder sacarles un mayor provecho a las herramientas y recursos del curso.

8. Anexos

```
review_es      object
sentimiento    object
dtype: object
```

Ilustración 1. Tipo de los datos

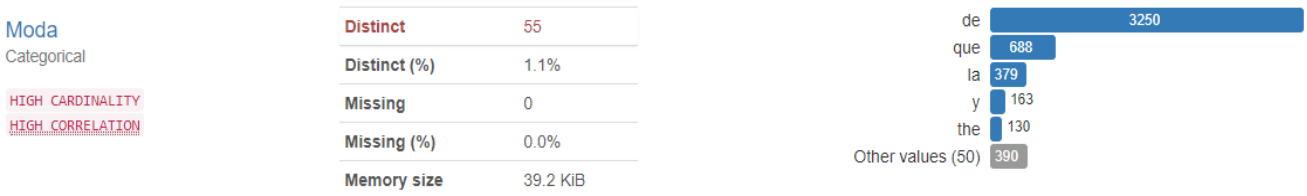

Ilustración 2. Palabras más frecuentes en las reseñas

Most occurring categories

Value	Count	Frequency (%)
Lowercase Letter	5398944	77.9%
Space Separator	1170568	16.9%
Other Punctuation	171201	2.5%
Uppercase Letter	154189	2.2%
Decimal Number	15562	0.2%
Close Punctuation	7434	0.1%
Open Punctuation	7293	0.1%
Dash Punctuation	3493	0.1%
Format	304	< 0.1%
Connector Punctuation	181	< 0.1%
Other values (7)	481	< 0.1%

Ilustración 3. Veces y frecuencia por tipo de carácter

Currency Symbol

Value	Count	Frequency (%)
$	162	94.2%
£	9	5.2%
€	1	0.6%

Modifier Symbol

Value	Count	Frequency (%)
`	58	85.3%
^	9	13.2%
´	1	1.5%

Other Letter

Value	Count	Frequency (%)
º	22	62.9%
ª	13	37.1%

Other Number

Value	Count	Frequency (%)
½	4	80.0%
¾	1	20.0%

Ilustración 4. Símbolos más frecuentes en las reseñas

sentimiento
Categorical

UNIFORM

Distinct	2
Distinct (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	39.2 KiB

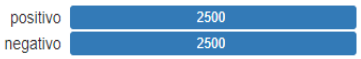

Ilustración 5. Proporción de los tipos de sentimientos

Classification Report

	precision	recall	f1-score	support
negativo	0.83	0.88	0.85	491
positivo	0.87	0.80	0.84	470
accuracy			0.84	961
macro avg	0.85	0.84	0.84	961
weighted avg	0.85	0.84	0.84	961

Tabla 1. Métricas del mejor modelo usando Multinomial Naive Bayes

Classification Report

	precision	recall	f1-score	support
negativo	0.87	0.85	0.86	491
positivo	0.85	0.86	0.86	470
accuracy			0.86	961
macro avg	0.86	0.86	0.86	961
weighted avg	0.86	0.86	0.86	961

Tabla 2. Métricas del mejor modelo usando Logistical Regression

{'criterion': 'entropy', 'max_features': 100, 'min_samples_split': 4}

Classification Report

	precision	recall	f1-score	support
negativo	0.84	0.83	0.84	491
positivo	0.82	0.84	0.83	470
accuracy			0.83	961
macro avg	0.83	0.83	0.83	961
weighted avg	0.83	0.83	0.83	961

Tabla 3. Métricas del mejor modelo usando Random Forest