Clasificación automática de reseñas de películas

Araceli Romero Zerpa
ENES-UNAM
Morelia, México
araceliromerozerpa@gmail.com

RESUMEN

El aprendizaje automático se utiliza en diversas áreas de la ciencia y la tecnología. Por ejemplo, en la clasificación automática de textos en mensajes, en imágenes y audios.

En este artículo se abordará el problema de clasificación automática de reseñas de películas para etiquetarlas como positivas o negativas. Para este propósito se utilizarán varios modelos de aprendizaje supervisado, utilizando como entrada una base de datos de reseñas de películas de la Universidad de Stanford de acceso libre.

Se presentan los resultados obtenidos utilizando cada modelo de aprendizaje y al final de realiza una comparación de los resultados.

KEYWORDS

Aprendizaje supervisado, clasificación automática de reseñas de películas.

ACM Reference Format:
Araceli Romero Zerpa. 2017. Clasificación automática de reseñas de películas. In *Proceedings of Conference Name.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/8888888.7777777

1. INTRODUCCIÓN

El análisis de sentimientos en reseñas de películas se ha convertido en una tarea importante en el procesamiento del lenguaje natural (NLP), con aplicaciones que van desde sistemas de recomendación hasta estudios de mercado. La capacidad de clasificar automáticamente las reseñas como positivas o negativas no solo ayuda a los usuarios a evaluar rápidamente la opinión pública, sino que también proporciona información para posteriores análisis estadísticos.

2. DATOS

Se utiliza la base de datos de acceso libre disponible en [Maas et al. 2011], la cual consta de 50,000 reseñas en inglés de películas, con 25,000 etiquetadas como positivas y 25,000 negativas, elaboradas por la Universidad de Stanford.

3. EXPLORACIÓN INICIAL DE LOS DATOS

Cada registro de datos tiene tres columnas: review (la revisión), sentiment (el sentimiento: 0-negativo ó 1-positivo) y dataset_set (tipo: train, test).

Adicionalmente se agregó una columna para la longitud de la reseña en cantidad de palabras. Con estos datos se efectuó un análisis estadístico cuyos resultados se muestran en la tabla 1 y también se calculó el histograma que se muestra en la figura 1.

Tabla 1: Estadísticas descriptivas de la longitud de las reseñas

Set	Mean	Median	Mode	S.D.	Var	Min	Max	Skew
Test	228.5	172	127	168.8	28521.7	4	2278	2.1
Train	233.7	174	123	173.7	30183.1	10	2470	2.1

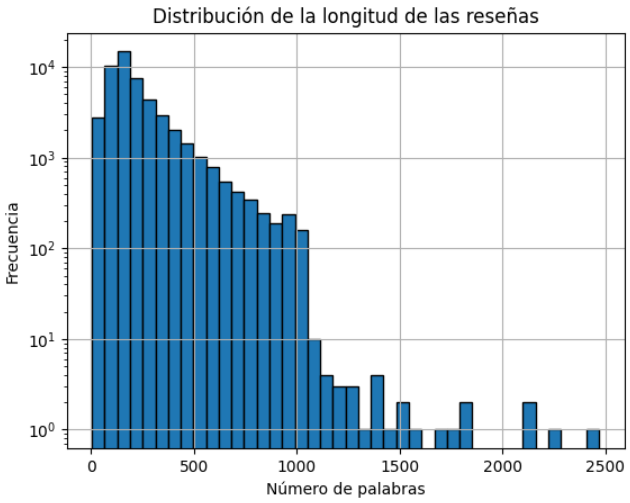

Figura 1: Histograma de la longitud de las reseñas.

3.1. Análisis de la Distribución

Los datos estadísticos revelan varias características importantes sobre la distribución de la longitud de las reseñas:

- **Distribución asimétrica positiva:** Tanto en el conjunto de entrenamiento como en el de prueba, la asimetría indica una distribución fuertemente sesgada a la derecha. Esto significa que hay muchas reseñas cortas y pocas reseñas extremadamente largas, como se verifica en la figura 1.
- **Discrepancia entre media y mediana:** La media es significativamente mayor que la mediana (228.5 vs 172 en test, 233.7 vs 174 en train), lo que confirma la presencia de reseñas muy largas.
- **Variabilidad considerable:** La alta desviación estándar (169-174 palabras) muestra una gran dispersión en las longitudes de las reseñas.
- **Consistencia entre conjuntos:** Las estadísticas son muy similares entre los conjuntos de entrenamiento y prueba, lo que indica que la división fue adecuada y representativa.

Conference Name, Conference Date and Year, Conference Location

ACM ISBN 978-1-4503-1234-5/17/07
https://doi.org/10.1145/8888888.7777777

- **Presencia de valores extremos:** El máximo de 2278-2470 palabras contrasta fuertemente con el mínimo de 4-10 palabras, mostrando un rango muy amplio en la longitud de las reseñas.

4. METODOLOGÍA

Este estudio aborda la pregunta fundamental: ¿Qué tan efectivos son diferentes algoritmos de aprendizaje automático supervisado para clasificar reseñas de películas como positivas o negativas basándose únicamente en su reseña? Específicamente, se comparó el rendimiento de cuatro enfoques clásicos: Naive Bayes Multinomial, Regresión Logística, Máquinas de Vectores de Soporte (SVM) y eXtreme Gradient Boosting (XGBoost).

4.1. Preprocesamiento de Texto

El preprocesamiento de texto es fundamental para mejorar la calidad de los datos antes de aplicar algoritmos de aprendizaje automático. Se implementaron las siguientes etapas:

- **Normalización de caracteres:** Se convirtió todo el texto a minúsculas para evitar duplicidad léxica (por ejemplo, "Good" y "good").
- **Eliminación de URLs y menciones:** Se eliminaron enlaces web, menciones a usuarios, ya que no aportaban información relevante al análisis de sentimientos.
- **Conservar "!" y "?"**: Estos signos de puntuación son indicadores importantes de sentimiento en reseñas.
- **Lemmatización**: Se eligió lemmatización mediante la herramienta `WordNetLemmatizer` de NLTK porque produce palabras válidas que preservan mejor el significado de la palabra. Este método reduce las palabras a su forma base (ej: "running" → "run"). Este enfoque se prefirió sobre el stemming.
- **Eliminación selectiva**: Mantuvimos números ya que en reseñas de películas pueden referirse a años o calificaciones.
- **Stop words**: Eliminamos palabras comunes del inglés (ej: "the", "is","and") utilizando la lista provista por NLTK, ya que generalmente no aportan información discriminante para la clasificación. Se conservaron las palabras que pudieran ser relevantes para el sentimiento (ej: "not", "very").

Con el texto ya preprocesado, se realizó un análisis de frecuencia de palabras, separando las reseñas positivas y negativas. Se contaron las apariciones de cada término utilizando la clase `Counter` de la biblioteca `collections`, y se graficaron las 40 palabras más frecuentes en cada grupo en las figuras 2 y 3. Este análisis permitió identificar patrones léxicos distintivos en ambos tipos de reseñas, lo cual puede facilitar la interpretación de los resultados obtenidos por los clasificadores.

4.2. Algoritmos utilizados

Se utilizaron los siguientes algoritmos de aprendizaje:

- **Naive Bayes Multinomial:** Se seleccionó este modelo por su eficacia comprobada en tareas de clasificación de texto, especialmente en escenarios con representaciones de tipo bag of words, donde se representa un documento como un conjunto de palabras, ignorando el orden y la estructura gramatical. Se enfoca en la frecuencia con la que aparecen las palabras en el documento.
- **Regresión Logística:** Este modelo lineal probabilístico se incluyó por su simplicidad y buen desempeño en clasificación binaria.
- **Máquinas de Vectores de Soporte (SVM):** Por ser un algoritmo robusto al overfitting en espacios de alta dimensión y por su efectividad con datos linealmente separables común en TF-IDF (Term Frequency-Inverse Document Frequency), que evalua la importancia de una palabra en un documento dentro de una colección de documentos.
- **eXtreme Gradient Boosting (XGBoost):** Este modelo de boosting basado en árboles fue incluido por su capacidad para capturar interacciones complejas entre las características y su rendimiento competitivo en múltiples tareas de clasificación.

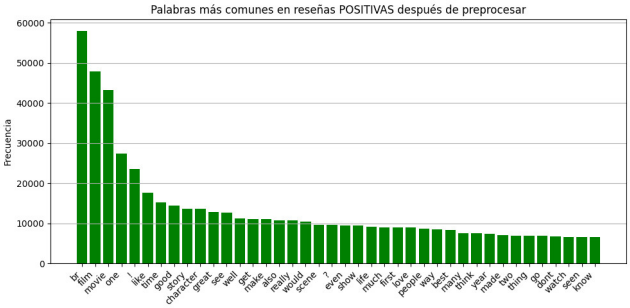

Figura 2: palabras más comunes positivas.

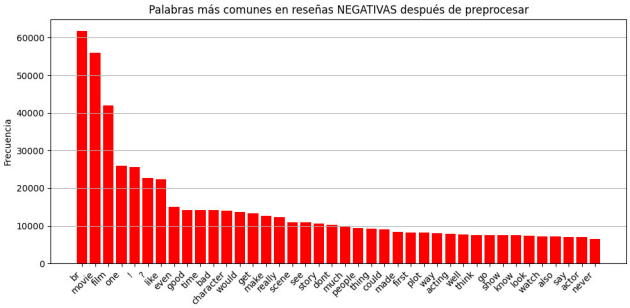

Figura 3: palabras más comunes negativas.

Todos los modelos se entrenaron sobre vectores generados mediante representaciones *TF-IDF* (la gráfica 4 muestra la distribución obtenida), con el objetivo de dar mayor peso a términos informativos que aparecen en pocas reseñas. Se empleó validación cruzada para ajustar los hiperparámetros clave (como el parámetro de regularización en SVM o el número de estimadores en XGBoost, entre otros hiperparámetros) y evaluar el rendimiento.

5. EXPERIMENTOS Y RESULTADOS

Para cada modelo se realizó una búsqueda de hiperparámetros (usando `GridSearchCV`), optimizando la métrica de `accuracy`. El mejor modelo encontrado se evaluó sobre el conjunto de validación,

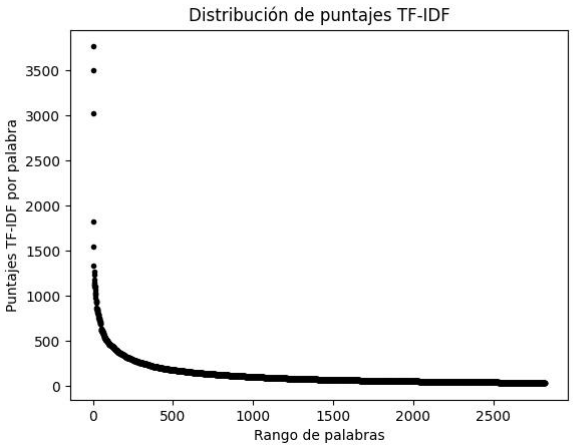

Figura 4: Distribución de puntajes TF-IDF

obteniendo la matriz de confusión y su respectivo informe de clasificación. Finalmente se evalúa el mejor modelo sobre el conjunto de prueba.

5.1. Naive Bayes Multinomial

Los mejores hiperparámetros encontrados se muestran en la tabla 2.

Tabla 2: Mejores hiperparámetros encontrados

Hiperparámetro	valor
`multinomialnb__alpha`	1.0
`multinomialnb__fit_prior`	True
`tfidfvectorizer__max_features`	10000
`tfidfvectorizer__ngram_range`	(1, 2)

El valor `multinomialnb__alpha` de 1.0 representa el suavizado de Laplace y evita probabilidades cero cuando una palabra no aparece en una clase. El valor 1.0 equivale a suavizado tipo "add-on" (conservador). Valores más bajos, como 0.1, pueden llevar a sobreajuste.

El valor `multinomialnb__fit_prior` True ajusta las probabilidades iniciales de clase según la frecuencia observada en los datos. Si se estableciera en `False`, asumiría distribución uniforme (50 %/50 %).

El valor `tfidfvectorizer__max_features` de 10000 limita el espacio vectorial a las 10,000 palabras más frecuentes, lo que ayuda a reducir la dimensionalidad.

El valor de `tfidfvectorizer__ngram_range` de (1, 2) incluye unigramas y bigramas. Esto permite capturar frases relevantes como "not good". Se observó mejora en precisión respecto a usar solo unigramas.

Los resultados del mejor modelo encontrado se muestran en la tabla 3.

La figura 5 muestra la matriz de confusión obtenida.

Tabla 3: Resultados del modelo Naive Bayes Multinomial después de GridSearch

Métrica	Valor
Precisión (Accuracy)	0.8695
Precisión (Clase 0)	0.88
Precisión (Clase 1)	0.86
Recall (Clase 0)	0.86
Recall (Clase 1)	0.88
F1-score (Macro Promedio)	0.87

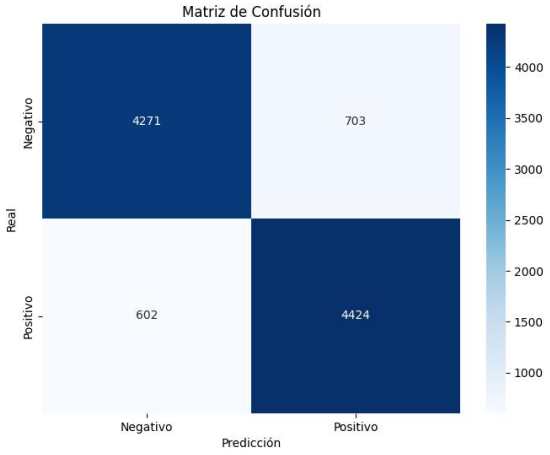

Figura 5: Matriz de confusión — Multinomial Naive Bayes

5.2. Regresión Logística

Los mejores hiperparámetros encontrados para el modelo de Regresión Logística se muestran en la tabla 4.

Tabla 4: Mejores hiperparámetros encontrados

Hiperparámetro	Valor
`logisticregression_C`	0.1624
`logisticregression_penalty`	l2
`tfidfvectorizer_ngram_range`	(1, 2)

El valor `logisticregression_C` controla la regularización del modelo. Un valor de 0.1624 indica una regularización moderada. Valores más bajos implican una mayor penalización, lo cual ayuda a prevenir el sobreajuste.

El valor `logisticregression_penalty` `l2` aplica la norma cuadrática como penalización, lo que tiende a generar coeficientes más pequeños pero no completamente cero.

El valor `tfidfvectorizer_ngram_range` de (1, 2) indica el uso combinado de unigramas y bigramas. Esta estrategia mejora la capacidad del modelo para identificar relaciones entre palabras, lo que contribuyó a mejorar su desempeño.

Los resultados obtenidos con el mejor modelo encontrado se resumen en la tabla 5.

Tabla 5: Resultados del modelo de Regresión Logística después de GridSearch

Métrica	Valor
Precisión (Accuracy)	0.8867
Precisión (Clase 0)	0.89
Precisión (Clase 1)	0.88
Recall (Clase 0)	0.88
Recall (Clase 1)	0.89
F1-score (Macro Promedio)	0.89

El modelo muestra un rendimiento equilibrado entre clases, con una precisión y recall muy similares (aproximadamente 89 % en ambas clases). Lo que muestra que el modelo está haciendo una buena generalización de los datos, por lo que no ve que haya un sesgo hacia alguna clase particular.

La figura 6 muestra la matriz de confusión obtenida.

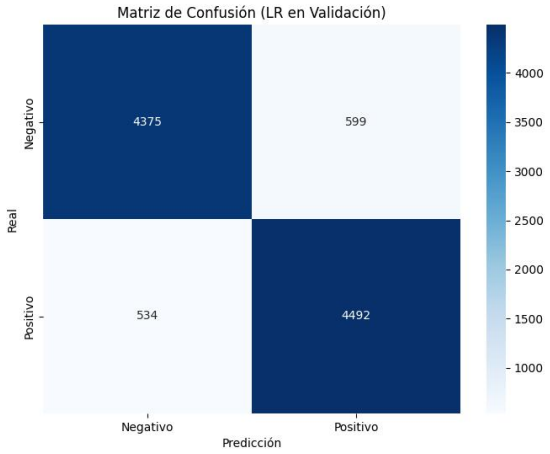

Figura 6: Matriz de confusión — Regresión logística

5.3. Máquinas de Soporte Vectorial

Los mejores hiperparámetros encontrados para el modelo SVM (Support Vector Machine) con kernel RBF se presentan en la tabla 6.

Tabla 6: Mejores hiperparámetros encontrados

Hiperparámetro	Valor
`svc_C`	1
`svc_gamma`	scale
`svc_kernel`	rbf

El parámetro `svc_C` controla el equilibrio entre maximizar el margen y minimizar el error de clasificación. Un valor de 1 representa una regularización estándar, balanceando bien la complejidad del modelo y el ajuste a los datos.

El parámetro `svc_gamma` con `scale`, ajusta automáticamente el valor de γ según la fórmula $1/(n_features \times \mathrm{Var}(X))$. Esto mejora la estabilidad del modelo en comparación con valores fijos como 'auto'.

El uso de `svc_kernel = rbf` (Radial Basis Function) permite modelar relaciones no lineales complejas entre las características. Es especialmente útil en tareas de clasificación de texto, donde los datos no son linealmente separables.

Los resultados del mejor modelo SVM tras la búsqueda de hiperparámetros se resumen en la tabla 7.

Tabla 7: Resultados del modelo SVM (RBF) después de GridSearch

Métrica	Valor
Precisión (Accuracy)	0.8900
Precisión (Clase 0)	0.89
Precisión (Clase 1)	0.89
Recall (Clase 0)	0.88
Recall (Clase 1)	0.90
F1-score (Macro Promedio)	0.89

El modelo SVM alcanza una precisión global de 89.00 %, con métricas de precisión, recall y F1-score equilibradas en ambas clases. El recall ligeramente superior en la clase 1 (90 %) indica una mayor sensibilidad al detectar reseñas positivas, lo cual puede ser valioso si el objetivo es identificar contenido favorable en análisis de sentimientos.

En conjunto, el modelo SVM demostró ser el de mejor desempeño general entre los evaluados, con una combinación adecuada de precisión y capacidad de generalización.

La figura 6 muestra la matriz de confusión obtenida.

5.4. eXtreme Gradient Boosting

Los mejores hiperparámetros encontrados para el modelo XGBoost se muestran en la tabla 8.

Tabla 8: Mejores hiperparámetros encontrados

Hiperparámetro	Valor
`xgbclassifier_learning_rate`	0.2
`xgbclassifier_max_depth`	7

El hiperparámetro `xgbclassifier_learning_rate` controla qué tan rápido el modelo se ajusta a los datos. Un valor de 0.2 representa una tasa de aprendizaje moderada, que permite una convergencia más rápida sin comprometer demasiado la estabilidad del entrenamiento.

El parámetro `xgbclassifier_max_depth` establece la profundidad máxima de los árboles de decisión individuales. Un valor de 7

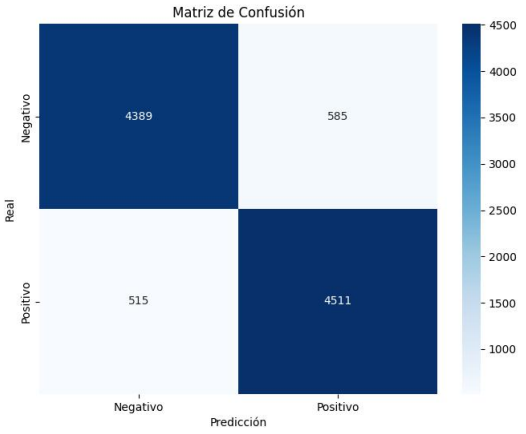

Figura 7: Matriz de confusión — Máquina de soporte vectorial (kernel RBF)

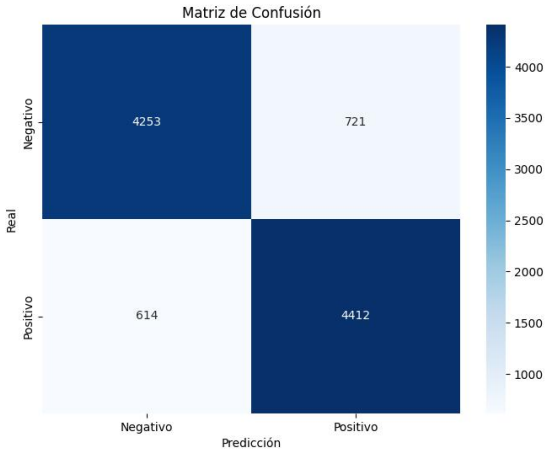

Figura 8: Matriz de confusión — eXtreme Gradient Boosting

permite al modelo capturar patrones más complejos sin sobreajustar excesivamente.

Los resultados del modelo XGBoost después de aplicar búsqueda de hiperparámetros se detallan en la tabla 9.

Tabla 9: Resultados del modelo XGBoost después de GridSearch

Métrica	Valor
Precisión (Accuracy)	0.8665
Precisión (Clase 0)	0.87
Precisión (Clase 1)	0.86
Recall (Clase 0)	0.86
Recall (Clase 1)	0.88
F1-score (Macro Promedio)	0.87

El modelo XGBoost logra una precisión global de 86.65 %, con un desempeño bastante equilibrado entre ambas clases. El recall superior en la clase 1 (88 %) indica buena capacidad para identificar correctamente las reseñas positivas.

La figura 8 muestra la matriz de confusión obtenida.

5.5. Análisis de resultados

En la tabla 10 se presenta una comparación entre los cuatro modelos evaluados: Naive Bayes Multinomial, Regresión Logística, SVM con kernel RBF y XGBoost. En este contexto, es importante lograr un balance entre precisión y recall en ambas clases para evitar sesgos hacia reseñas de un solo tipo, sean positivas o negativas.

SVM con kernel RBF. fue el modelo con mejor desempeño general, logrando una **accuracy de 89 %** y F1-score balanceado en ambas clases. Este modelo fue efectivo para capturar relaciones no lineales, lo que fue útil en la clasificación de reseñas ya que tienen bastante lenguaje subjetivo.

Tabla 10: Comparación de desempeño entre modelos

Modelo	Accuracy	Precision	Recall	F1-score
NBM	0.8695	0.87	0.87	0.87
RL	0.8867	0.89	0.89	0.89
SVM(RBF)	0.8900	0.89	0.89	0.89
XGBoost	0.8665	0.87	0.87	0.87

Regresión Logística. también mostró un rendimiento competitivo (**F1-score de 0.89**), y presenta la ventaja de ser más interpretable que SVM, lo que puede ser deseable si se requiere justificar decisiones del modelo ante usuarios finales.

Naive Bayes Multinomial. obtuvo un desempeño ligeramente inferior, aunque bastante aceptable (**F1-score de 0.87**). Su simplicidad y velocidad lo hacen en una buena opción, especialmente en aplicaciones con recursos computacionales limitados o que requieren respuestas en tiempo real.

XGBoost. , si bien alcanzó una buena precisión (86.65 %), se ubicó como el modelo de menor rendimiento relativo. A pesar de ser un algoritmo potente, en este caso no logró superar a los otros modelos, lo cual puede atribuirse a la naturaleza del problema y al preprocesamiento textual con TF-IDF, que favorece modelos lineales sobre árboles.

6. CONCLUSIONES

A lo largo de este análisis, se evaluaron cuatro modelos de aprendizaje supervisado para la clasificación de sentimientos en reseñas de películas: Naive Bayes Multinomial, Regresión Logística, SVM con kernel RBF y XGBoost. Cada modelo fue entrenado y optimizado usando GridSearch y validación cruzada, con preprocesamiento basado en TF-IDF y la incorporación de n-gramas.

De todos los modelos evaluados, el **SVM con kernel RBF** fue el que logró el mejor desempeño, alcanzando una **precisión del 89 %** con valores balanceados de precision, recall y F1-score entre

ambas clases. Este rendimiento superior nos dice que las fronteras de decisión no lineales resultaron efectivas para capturar patrones complejos en el lenguaje de las reseñas.

Regresión Logística también mostró un rendimiento muy competitivo, probablemente debido a la naturaleza lineal del espacio vectorial generado por TF-IDF, que favorece modelos lineales. Naive Bayes ofreció buenos resultados a pesar de su simplicidad, gracias a su adecuación a problemas de texto donde las características (palabras) son muchas y en su mayoría independientes. XGBoost, aunque potente en otros contextos, no logró destacar aquí; esto puede deberse a que su arquitectura basada en árboles no aprovecha tan bien la estructura dispersa y de alta dimensionalidad generada por TF-IDF.

Una limitación importante del análisis es la dependencia de representaciones basadas en bag of words y TF-IDF, que no capturan semántica ni contexto. Además, los modelos se entrenaron con una cantidad limitada de características (hasta 10,000), lo que podría haber reducido su capacidad para aprender matices más profundos.

Para mejorar el análisis, una posible dirección sería incorporar otras representaciones vectoriales del lenguaje, como *word embeddings* (Word2Vec, GloVe) o modelos de lenguaje contextualizados como BERT. Asimismo, se podría explorar el uso de técnicas de ensamble entre modelos para combinar sus fortalezas, así como evaluar el desempeño en subconjuntos de datos (por ejemplo, reseñas cortas vs. largas) para entender mejor sus límites. Además se podrían explorar algoritmos optimizados para buscar de manera más inteligente los hiperparámetros, especialmente para modelos como XG Boost que requieren de bastantes hiperparámetros para lograr un buen desempeño.

El análisis muestra que incluso modelos relativamente simples como Regresión Logística o Naive Bayes pueden ofrecer un buen rendimiento en tareas de análisis de sentimientos, siempre que el preprocesamiento sea adecuado. Sin embargo, SVM demostró ser la mejor opción en este caso.

REFERENCIAS

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015