Tecnológico de Monterrey

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY CAMPUS ESTADO DE MÉXICO

Inteligencia artificial avanzada para la ciencia de datos I.

Grp. 101.

Alumno:

Leonardo Cossio Dinorin.

Profesor:

Jorge Adolfo Ramírez Uresti.

Módulo:

Aprendizaje máquina

Análisis y Reporte sobre el desempeño del modelo.

Fecha de entrega:

11 de septiembre del 2024.

Descripción de programa:

El programa elegido para hacer este análisis fue el clasificador de noticias falsas y verdaderas del segundo entregable de esta materia. En la entrega anterior se utilizó el algoritmo de *Random Forest* para clasificar las noticias, sin embargo, debido al poder computacional que requiere esa metodología se decidió cambiar por la técnica de Clasificación Lineal por Vectores de Soporte.

El algoritmo de aprendizaje máquina recibe un texto, el cual será la noticia a clasificar y utilizando la librería de *sklearn* con vectores de soporte podremos predecir si dicha noticia es falsa o verdadera.

Set de datos:

El dataset utilizado para este problema fue utilizado en las elecciones estadounidenses del 2017 por George McIntire, quien decidió obtener noticias de diferentes fuentes de información y construir un algoritmo que clasificara las que fueran falsas y verdaderas utilizando la técnica de *Naive Bayes*, con la cual obtuvo un 88% de exactitud. Se utilizó ese mismo dataset para esta implementación, dicho archivo contiene un total de 6335 noticias, de las cuales 3171 son noticias verdaderas y 3164 son falsas. Esta información revela que no existe un desbalanceo de clases, lo cual implica que, utilizando estos datos, nuestro algoritmo puede realizar un buen trabajo clasificándolas y generalizando.

División del set de datos:

Lo primero que se hizo fue dividir el set de datos en sets de entrenamiento, validación y prueba. Se decidió utilizar el 20% del dataset completo para el set de prueba, posteriormente se utilizó el 20% del set restante para validación y el resto para entrenamiento, quedando los siguientes tamaños:

```
********** DIVISIÓN DEL DATASET **********
Dimensiones del dataset COMPLETO: (6335,)
Dimensiones del dataset de entrenamiento: (4054,)
Dimensiones del dataset de validación: (1014,)
Dimensiones del dataset de prueba: (1267,)
**************************************************
```

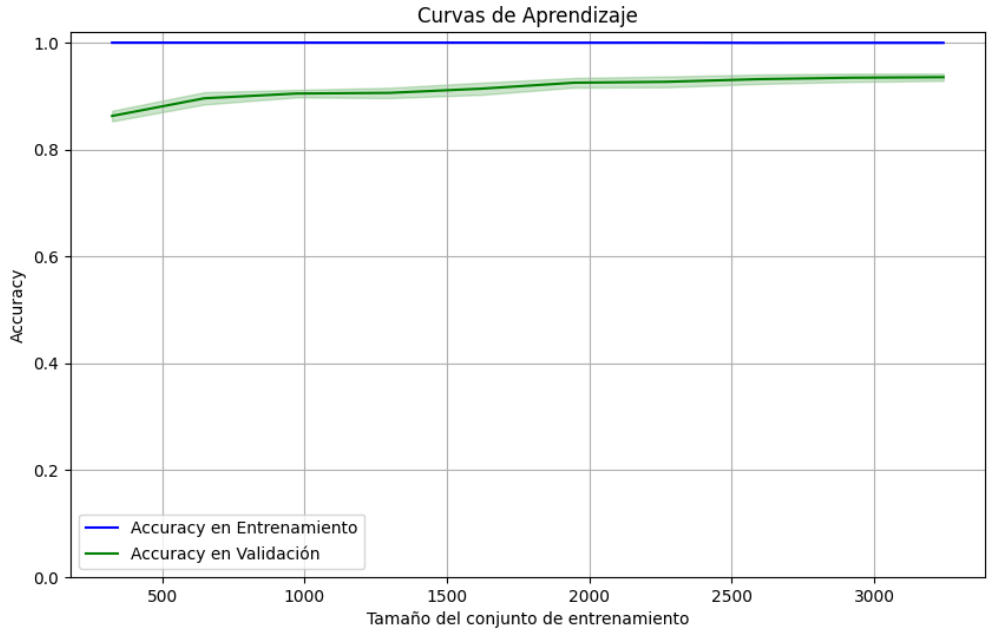
Como se puede observar, el set de entrenamiento equivale al 64% del dataset completo, el de validación al 16% y el de prueba al 20%. Para esta separación se utilizó la función del train_test_split proporcionada por la librería de sklearn.

Modelo utilizado:

Para esta clasificación se utilizó la Clasificación Lineal por Vectores de Soporte, esto debido a que este método es especialmente eficaz para trabajar en problemas de clasificación de texto, además que no es excesivamente tardado para hacer una búsqueda de parámetros con GridSearch como un Random Forest, lo que me permitió hacer un fine tunning de una manera más rápida y eficaz.

Después de hacer una búsqueda de parámetros, los mejores resultaron ser un valor de C = 1.0 y una penalización “L2”, con lo que el modelo conseguía un accuracy muy alto.

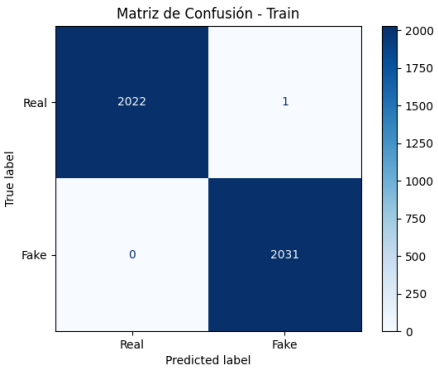
Resultados:

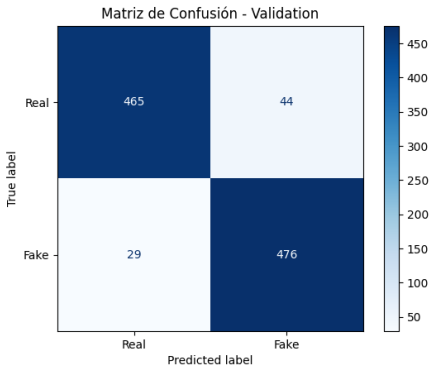


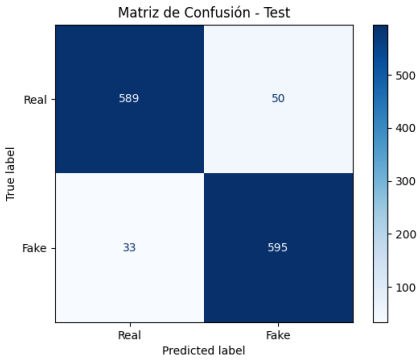
La curva de aprendizaje (Tomando en cuenta el accuracy) nos deja ver que el modelo utilizado se adaptó perfectamente y de una manera muy rápida a los datos.

```
****************** TRAIN SCORE ************************
Accuracy on train set: 1.00
Recall on train set: 1.00
Precision on train set: 1.00
F1 score on train set: 1.00
*******************************************************


****************** VALIDATION SCORE *******************
Accuracy on validation set: 0.93
Recall on validation set: 0.94
Precision on validation set: 0.92
F1 score on validation set: 0.93
*******************************************************


****************** TEST SCORE *************************
Accuracy on test set: 0.93
Recall on test set: 0.95
Precision on test set: 0.92
F1 score on test set: 0.93
*******************************************************
```

Los resultados de estas métricas coinciden de manera perfecta con la curva de aprendizaje obtenida anteriormente, donde de nuevo, podemos notar que en el entrenamiento obtiene calificaciones perfectas.







Por último analizaremos las gráficas de confusión, donde se puede observar claramente que lo hace perfecto con el set de entrenamiento y posteriormente baja ligeramente el rendimiento en el set de validación y prueba.

Gracias a estas métricas podemos analizar datos como el sesgo y la varianza resultantes del modelo.

Sesgo: El modelo muestra una pequeña diferencia en su rendimiento entre el conjunto de entrenamiento y el de validación, lo que sugiere un bajo sesgo. Si bien el modelo obtuvo una calificación perfecta en los datos de entrenamiento, lo cual podría indicar un ligero sobreajuste, el rendimiento en validación sigue siendo alto y las curvas de la curva de aprendizaje están cercanas, lo que implica que el modelo generaliza bien y no está sobreajustado de manera significativa.

Varianza: Analizando los resultados obtenidos, podemos darnos cuenta que existe una varianza baja en los las valores calculados, pues la diferencia entre las métricas obtenidas en el set de entrenamiento solo disminuyen en un 0.07.

Conclusión: Si bien el modelo muestra buenos resultados, el hecho de que se esté aprendiendo de manera perfecta los datos de entrenamiento indica un leve sobreajuste, lo que se traduce a que el modelo podría no generalizar de manera correcta con datos completamente nuevos (No disponibles en el dataset utilizado).

Este modelo podría ser utilizado sin problemas, sin embargo, como parte del entregable se pidió una mejora en el modelo utilizando técnicas de regularización, por lo que el siguiente paso es lograr que el modelo implementado sea capaz de generalizar un poco más (Disminuir el overfitting).

Modelo 2:

Como se explicó anteriormente, el primer modelo se aprende tan bien os datos de entrenamiento que posiblemente genere problemas a futuro si se intentan utilizar datos completamente nuevos, por lo que se utilizarán técnicas de regularización y manipulación de los datos para poder mejorar el balance de los resultados entre entrenamiento, validación y prueba.

Ajustes realizados:

```
# Aleatorizar los datos antes de entrenar y validar ---------- MEDIDA 1
X_train, y_train = shuffle(X_train, y_train, random_state=42)
X_valid, y_valid = shuffle(X_valid, y_valid, random_state=42)
```

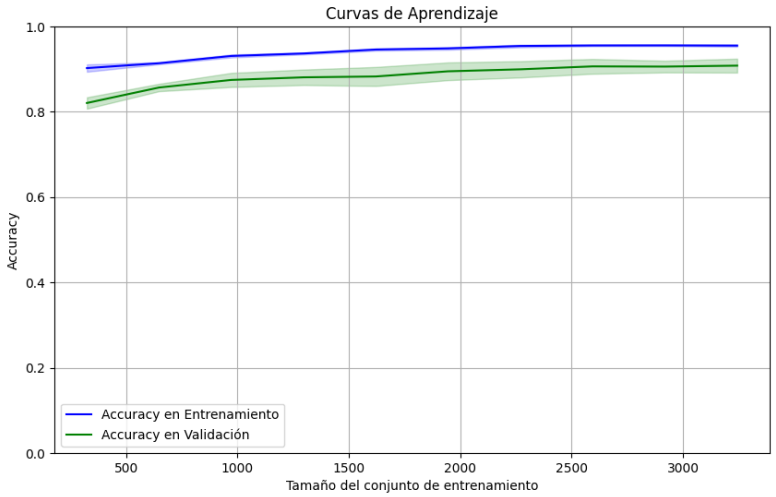
Lo primero que se hizo fue aleatorizar los datos, de esta manera nos aseguramos que el dataset no desbalancee las clases al momento de dividir el dataset en entrenamiento, validación y prueba.

```
('l_svc', LinearSVC(penalty="l1", C=0.7, random_state=42))  # Clasificador seleccionado
# MEDIDA 2: Regularizar utilizando L1 (Penaliza la suma de los valores absolutos de los coeficientes)
# MEDIDA 3: Disminuir C (Penalización de los errores)
```

Posteriormente se cambiaron 2 parámetros importantes en el clasificador: El primero fue la penalización utilizada, pues el modelo anterior utilizaba L2, en este caso se cambió a L1, el cual penaliza la suma de los valores absolutos de los coeficientes y los mantiene en valores bajos.

También se cambió el valor de C: Mientras el modelo anterior tenía un valor de 0.7, este tiene un valor de 0.7, lo que significa que penalizará menos los errores.

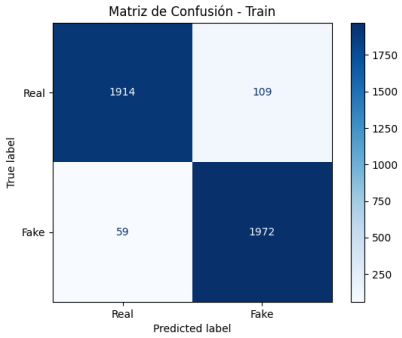
Resultados:

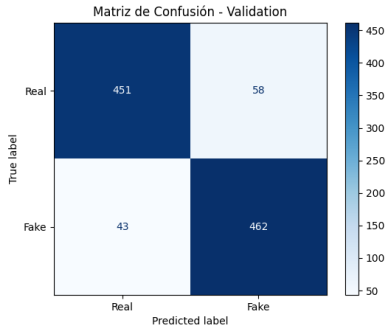


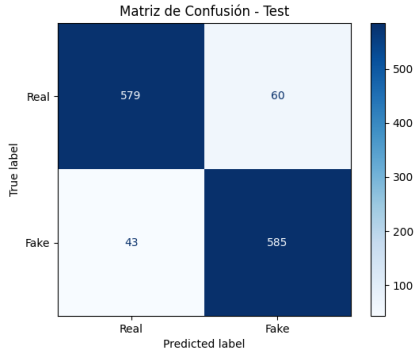
En el nuevo modelo se puede apreciar que las curvas de aprendizaje entre entrenamiento y validación tienen una diferencia mucho menos marcada, lo que nos indica un modelo más balanceado en cuanto al desempeño en los diferentes sets.

```
********** TRAIN SCORE **********
Accuracy on train set: 0.96
Recall on train set: 0.97
Precision on train set: 0.95
F1 score on train set: 0.96
**************************************************


********** VALIDATION SCORE **********
Accuracy on validation set: 0.90
Recall on validation set: 0.91
Precision on validation set: 0.89
F1 score on validation set: 0.90
**************************************************


********** TEST SCORE **********
Accuracy on test set: 0.92
Recall on test set: 0.93
Precision on test set: 0.91
F1 score on test set: 0.92
**************************************************
```

Analizando las métricas también podemos observar un mejor balance, pues ya no tiene ese ligero sobreajuste en los datos de entrenamiento y aún así los resultados de validación y prueba no se ven comprometidos.







Las matrices de confusión también concuerdan con los resultados obtenidos, pues si bien existen falsos positivos y falsos negativos, estos son mínimos para los tres sets.

Conclusión:

Se desarrolló un modelo capaz de clasificar noticias falsas y verdaderas. En un principio se utilizaron los mejores parámetros utilizando un GridSearch y midiendo el desempeño con “accuracy”, lo que nos llevó al primer modelo, cuyas calificaciones siempre fueron muy altas en el set de entrenamiento, sin embargo, disminuía un 7% en cuanto se predecían datos del conjunto de prueba, lo que nos indicaba un ligero sobreajuste y un pequeño problema al momento de generalizar los datos.

Para corregir esto se implementaron 3 cambios al modelo:

- Aleatorización de los datos
- Regularización L1

- Menor penalización en los errores (Valor C)

Estas medidas resultaron en un modelo mucho más equilibrado en cuanto al desempeño utilizando el conjunto de prueba, validación y entrenamiento, pues según las métricas obtenidas y las gráficas mostradas, la diferencia de accuracy entre el conjunto de entrenamiento y el conjunto de prueba solo es del 4%, lo que nos lleva a la conclusión de que el segundo modelo tiende a generalizar mejor la información, pues se disminuyó el sesgo que existía entre el conjunto de entrenamiento y validación, aunque la varianza aumentó ligeramente, no compromete los resultados obtenidos.