TRABAJO PRÁCTICO Nº 2

Críticas Cinematográficas

Juan Duzac
Ignacio de Matías Pose
Carolina Pico
Pierre Bigey

Introducción:

Durante el desarrollo del proyecto, se llevó a cabo un análisis exhaustivo de un conjunto de datos que consistía en reseñas de películas en español. El objetivo principal fue construir y comparar diferentes modelos para predecir si las reseñas eran positivas o negativas. A continuación, se presenta un informe detallado de todo lo realizado.

Desarrollo:

Análisis de la estructura del dataset: Se comenzó examinando la estructura del conjunto de datos, el cual constaba de tres columnas: ID, la reseña en español y el sentimiento asociado, que podía ser positivo o negativo.

Luego se procedió a la verificación de valores nulos: Se llevó a cabo una revisión en busca de valores nulos en el conjunto de datos. Afortunadamente, no se encontraron registros vacíos, lo que permitió continuar con las siguientes etapas del análisis.

Se realizó el split de datos y su transformación: Con el objetivo de realizar el entrenamiento y la evaluación de los modelos, se dividió el conjunto de datos en una proporción del 70% para entrenamiento (train) y el 30% restante para pruebas (test). Además, se aplicó la técnica CountVectorizer para transformar las reseñas en vectores numéricos, lo cual es un requisito para aplicar los modelos de análisis. También se definieron los *custom laber encoders* para pipelines

Modelos:

<u>Implementación del modelo Bayes Naïve:</u>

Se procedió a implementar el modelo de Bayes Naïve, un algoritmo de clasificación probabilístico. Para encontrar el mejor hiperparámetro Alpha, se utilizó la técnica GridSearchCV. Los resultados arrojaron un valor óptimo de Alpha igual a 0.44. Luego, se evaluó el modelo utilizando el conjunto de entrenamiento (train), obteniendo los siguientes resultados:

- F1-Score: 0.829473393989523
- Precision-Score:0.8585901826484018
- Recall-Score: 0.8022666666666667

<u>Implementación del modelo Random Forest:</u>

A continuación, se procedió a implementar el modelo de Random Forest, utilizando nuevamente la técnica CountVectorizer al inicio del proceso. Para encontrar los mejores hiperparámetros, se empleó GridSearch, que resultó en los siguientes valores óptimos: {'max_depth': 18, 'min_samples_split': 6, 'n_estimators': 150}.

Al evaluar el modelo con el conjunto de entrenamiento (train), se obtuvieron los siguientes resultados:

- F1-Score: 0.8304771612410016
- Precision-Score:0.7951689642552153
- Recall-Score: 0.8690666666666667

Implementación del modelo XGBoost:

Posteriormente, se implementó el modelo XGBoost, un algoritmo de aprendizaje basado en árboles de decisión potenciados. También se utilizó la técnica CountVectorizer al principio del proceso y se realizaron búsquedas de hiperparámetros. Los valores óptimos encontrados fueron: {'colsample_bytree': 0.5, 'gamma': 0.08, 'learning_rate': 0.1, 'max_depth': 13, 'min_child_weight': 1, 'n_estimators': 110, 'subsample': 1}.

Al evaluar el modelo con el conjunto de entrenamiento (train), se obtuvieron los siguientes resultados:

- F1-Score: 0.8361243239721118
- Precision-Score: 0.8176373136230407
- Recall-Score: 0.8554666666666667.

Implementación de la red neuronal:

En esta etapa, se implementó un modelo de red neuronal utilizando técnicas de vectorización en x_train y convirtiendo las categorías binarias de 'positivo' y 'negativo' en valores 0 y 1. También se definió la función de pérdida (loss) como 'bce' y el optimizador como 'adam'.

Para regularizar el modelo, se aplicaron técnicas de Dropout y EarlyStopping.

El modelo se entrenó durante 600 épocas y se obtuvieron los siguientes resultados en el conjunto de entrenamiento (train):

- F1-Score: 0.8731185635067336
- Precision-Score: 0.8646705020920502
- Recall-Score: 0.8817333333333334.

Implementación del ensamble de modelos:

Por último, se implementó un ensamble de modelos de tipo Stacking. Se utilizaron los modelos base Random Forest, SVM y KNN, y como meta modelo se utilizó XGBoost. Después de probar diferentes combinaciones de modelos, estos proporcionaron los mejores resultados.

Al evaluar el ensamblado de modelos con el conjunto de entrenamiento (train), se obtuvieron los siguientes resultados:

- F1-Score: 0.8761980830670927
- Precision-Score: 0.8748006379585327
- Recall-Score: 0.8776

Detalle adicional: Durante el análisis, se encontraron reseñas en inglés, lo cual fue inesperado. Se realizaron intentos para eliminar estas reseñas y ver si los resultados mejoraban, pero estos no presentaron cambios significativos. Incluso a veces, empeoraban. También se probó eliminar las mayúsculas de las reseñas, pero tampoco se lograron mejoras considerables en los resultados.

Los resultados obtenidos en Kaggle fueron:

Bayes Naive: 0.690

Random Forest: 0.726

Xgboost: 0.691

Red Neuronal: 0.694

Stacking: 0.742

Conclusiones:

En este informe se presentaron todas las acciones realizadas durante el trabajo de construcción de modelos para predecir si las reseñas de películas eran positivas o negativas. Se implementaron diversos modelos, como Naive Bayes, Random Forest, XGBoost, redes neuronales y un ensamble de modelos tipo Stacking. Cada modelo fue evaluado utilizando métricas como F1-Score, Precision-Score y Recall-Score en el conjunto de entrenamiento (train). Allí se observaron resultados prometedores en todos los modelos, lo que indica que todos ellos tienen el potencial de ser utilizados para clasificar reseñas de películas en función de su sentimiento.

Al probar los modelos en el conjunto de datos de prueba, se observó que los resultados fueron considerablemente peores en comparación con los obtenidos en el conjunto de entrenamiento. Esta discrepancia entre los conjuntos de entrenamiento y prueba puede ser indicativa de un problema de overfitting. Es decir, que seguramente los modelos se ajustaron demasiado a los detalles y patrones específicos del conjunto de entrenamiento, lo que redujo su capacidad para generalizar y realizar predicciones precisas en nuevos datos. Intentamos entrenar menos los modelos, usar validación cruzada y utilizar técnicas de regularización para así sobreajustar lo menos posible.

En conclusión, los resultados más bajos obtenidos en el conjunto de datos de prueba en comparación con el conjunto de entrenamiento indican la presencia de overfitting en los modelos implementados. Esto destaca la importancia de evaluar el rendimiento de los modelos en conjuntos de datos independientes. Creemos que el hecho de que el dataset de test y el de train no pertenezcan al mismo dataset puede tener una implicancia también en la diferencia en los resultados. Quizás la forma de expresarse de las reseñas en los distintos datasets puede provocar la discordancia de resultados. Igualmente, para concluir decidimos que el modelo de Stacking es el mejor para predecir ya que con él obtuvimos muy buenos resultados en nuestro dataset y el mejor resultado en Kaggle.