TP2: Críticas Cinematográficas - Grupo 24

Introducción

- El objetivo del trabajo práctico fue predecir críticas cinematográficas de un dataset como negativas o positivas mediante distintos modelos predictores, tales como Random Forest, XGBoost, Redes neuronales, Ensambles y Bayes Naive.

- El dataset de Train contiene 50000 registros y tres columnas: ID, Review y Sentimiento.
- El dataset de Test tiene 8599 registros y dos columnas: ID, Review.
  Se quiere predecir la columna sentimiento (positiva / negativa)

- **Preprocesamiento:** Para la limpieza del dataset primero identificamos los textos en español, luego con los textos ya separados creamos una función que se encarga de remover los acentos y convertir todas las letras a minúsculas, tokenizar el texto y eliminar puntuaciones y número y por último realizar una lematización con Spacy (reduce las palabras a su forma base o raíz). Una vez ya aplicada la limpieza son guardadas en una nueva columna: Review_es_clean.

  Modelos explorados: Los modelos utilizados en el trabajo práctico fueron Random Fores, XGboost, Ensambles como Voting y Stacking, Bayes Naive y Redes Neuronales con Keras y Tensor Flow o RNN. Algunos de estos modelos fueron optimizados a través de Bayes Search para lograr una mejor predicción.

- Desde un inicio creímos que las redes neuronales iban a ser un buen predictor por lo que decidimos dedicarle más tiempo y realizar más predicciones con estos modelos. Como resultado a esta hipótesis obtuvimos lo predicho y fue nuestro mejor resultado en Kaggle.

Cuadro de Resultados

Medidas de rendimiento en el conjunto de TEST:

- F1
- Precision
- Recall
- Accuracy
- Resultado obtenido en Kaggle.

Modelo	F1-Test	Presicion Test	Recall Test	Accuracy	Kaggle
Bayes Naive	0.8279	0.8596	0.7983	0.8339	0.7162
Random Forest	0.7291	0.6821	0.7831	0.7068	0.7059
XGBoost	0.7603	0.7338	0.7888	0.7494	0.7233
Red Neuronal (RNN)	0.8671	0.8432	0.8924	0.8708	0.7567
Ensamble Stacking	0.8671	0.8567	0.8778	0.8655	0.7292

Descripción de Modelos

- Bayes Naive: Este es un modelo de clasificación basado en el teorema de Bayes. Es especialmente conocido por su simplicidad y eficacia, siendo muy utilizado en tareas de clasificación de texto.

- Random Forest: Es un método de ensamble que opera construyendo múltiples árboles de decisión durante el entrenamiento y generando la salida basada en la mayoría de votos y es muy eficaz para evitar el sobreajuste.

- XGBoost: Es una implementación optimizada de los métodos de aumento de gradiente diseñada para ser altamente eficiente. XGBoost proporciona un algoritmo de aprendizaje supervisado que puede utilizarse para tareas de regresión, clasificación, ranking y predicción de usuarios.

- Red Neuronal - Red Neuronal Recurrente: Las RNN son un tipo de red neuronal artificial donde las conexiones entre los nodos forman un ciclo. Esto les permite exhibir un comportamiento dinámico temporal. A diferencia de las redes tradicionales, las RNN tienen “memoria” que captura información sobre lo que ha sido calculado anteriormente, es decir se retroalimentan a la hora de realizar el siguiente paso.

- Ensamble Stacking: Es una técnica de ensamble de modelos que consiste en combinar las predicciones de varios modelos de aprendizaje automático para hacer una predicción final.

- Arquitectura de de la RNN:

  - Epoch: 6
  - Test_size = 0.05
  - Early_stopping con patience 4
  - Validation split de 0.2

  - Entrada: es una secuencia de palabras representadas como índice enterados de longitud: ‘max_sequence_lenght’

  - Procesamiento: En la primera capa se transforma cada palabra en un vector de 100 dimensiones y en la segunda capa (GRU) se procesa la secuencia de vectores utilizando 64 unidades GRU con un 20% de dropout.

  - Salida: La salida de la red neuronal es un valor único entre 0 y 1 (debido a la función sigmoid).

- Intentamos implementar modelos como BERT, roBERTuito y roBERTa pero debido a su alto coste computacional y su tiempo de ejecución no logramos obtener un modelo que nos permita realizar las predicciones. Sin embargo, creemos que estos modelos serían buenos predictores para estos problemas y nos ayudarían a obtener un valor más elevado en Kaggle.

Conclusiones generales

- Creemos que el análisis exploratorio fue importante para obtener mejores resultados en la predicción de kaggle, sin embargo no encontramos formas de detectar outliers (como lo hicimos en el trabajo práctico 1) enfocamos la limpieza de los datos en eliminar palabras, caracteres y otras cosas que pueden afectar a la hora de predecir.

- Tanto en TEST como en Kaggle nuestro mejor predictor fue una Red Neuronal Recurrente con F1 en local de 0.8671 y 0.7567 en Kaggle

- El modelo más sencillo de entrenar y rápido fue el Bayes Naive ya que no requiere apenas tiempo de entrenamiento y obtiene resultados considerablemente buenos como puede ser en nuestro caso 0.7162 en Kaggle a comparación de otros modelos que requieren más tiempo como las redes neuronales llegando a tener un tiempo de ejecución de más de 30 minutos para obtener un 0.04 mas de score (0.7567 en Kaggle).

- Creemos que 30 minutos para entrenar el modelo no es un tiempo elevado, por lo que creemos que podría ser utilizado de forma productiva..

- Para mejorar nuestros resultados habría que realizar una limpieza más profunda y exhaustiva del dataset y buscar mejores hiperparametros en nuestra red neuronal o implementar algún modelo de la variante BERT por más que tarde mucho más tiempo en ejecutarse.

Tareas Realizadas

Teniendo en cuenta que el trabajo práctico tuvo una duración de 9 (nueve) semanas, le pedimos a cada integrante que indique cuántas horas (en promedio) considera que dedicó semanalmente al TP

Integrante	Promedio Semanal (hs)
Lautaro Torraca - 108813	10
Gianluca Negrotti - 108184	10
Marco Tosi - 107237	10