APC - Informe Caso Kaggle

June Gorostidi Elizetxea(1566312)

1 Introduction

En este trabajo nos centraremos en clasificar los titulares de noticias en clickbaits y no-clickbaits. Para ello, primero de todo haremos un análisis de los atributos y nos familiarizaremos con el dataset. Después, aplicaremos dos métodos de aprendizaje para poder hacer esta clasificación. Para finalizar, compararemos los métodos utilizados y los resultados obtenidos.

2 Análisis de los atributos

Para empezar, usando diferentes funciones observamos como es el dataset, los atributos que tiene... para así poder trabajar mejor el más adelante.

La siguiente tabla nos da información sobre nuestro dataset: de que tipo es cada atributo, el número de entradas, memoria usada... Como se puede apreciar, en nuestro dataset no hay ningún valor nulo. Gracias a esto no necesitaremos modificarlo para trabajar con él.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32000 entries, 0 to 31999
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   headline   32000 non-null  object
 1   clickbait  32000 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 500.1+ KB
```

Por otro lado, en esta tabla aparecen las estadísticas de los atributos numéricos del dataset; entre ellos la media, la desviación estándar y los cuartiles. Para acabar, usamos la función countplot() para visualizar los datos en un gráfico.

	clickbait
count	32000.000
mean	0.500
std	0.500
min	0.000
25%	0.000
50%	0.000
75%	1.000
max	1.000

Figure 1: describe()

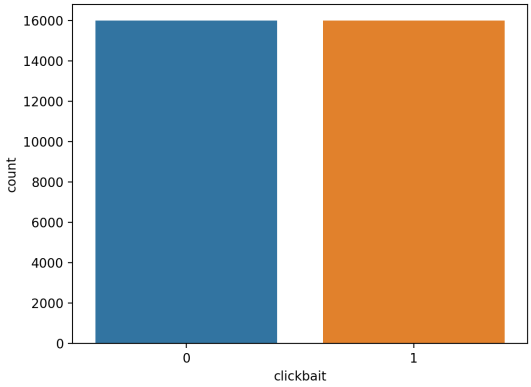


Figure 2: countplot

Nuestro dataset solo consta de dos columnas: la primera contiene titulares de sitios de noticias en formato de texto y la segunda contiene etiquetas numéricas de las cuales 1 representa es clickbait y 0 representa no-lickbait. Por lo tanto, tanto el gráfico como la tabla de estadísticas no nos dan mucha informacion relevante.

En conlsusión, este conjunto de datos contiene los titulares de varios sitios de noticias como 'WikiNews', 'New York Times', 'The Guardian', 'The Hindu', 'BuzzFeed', 'Upworthy', 'ViralNova', 'Thatscoop', 'Scoopwhoop' y 'ViralStories'. Tiene dos columnas la primera contiene titulares y la segunda tiene etiquetas numéricas de clickbait en el que 1 representa que es clickbait y 0 representa que es título no-clickbait. El conjunto de datos contiene un total de 32000 filas, de las cuales el 50% son clickbait y el otro 50% no son clickbait.

3 Aplicación de diversos métodos de aprendizaje

3.1 Detector clickbait con vectorizador de recuento

Primero de todo, con el fin de utilizar datos textuales para el modelado predictivo, el texto debe ser analizado para eliminar ciertas palabras - este proceso se llama **tokenización**. Estas palabras necesitan ser codificadas como enteros, o valores de coma flotante, para su uso como entradas en algoritmos de aprendizaje automático. Este proceso se denomina extracción de características (o vectorización).

El CountVectorizer de Scikit-learn se utiliza para convertir una colección de documentos de texto a un vector de conteos de términos/tokens. También permite el pre-procesamiento de datos de texto antes de generar la representación vectorial. Esta funcionalidad hace que sea un módulo de representación de características altamente flexible para el texto.

Una vez hecha toda la limpieza, hemos utilizado un algoritmo de clasificación Bayes Naive Multinomial para la clasificación de texto del conjunto de datos.

¿Qué es el algoritmo Bayes Naive Multinomial?

El algoritmo Bayes Naive Multinomial es un método de aprendizaje probabilístico que se utiliza principalmente en el Procesamiento del Lenguaje Natural (PNL). El algoritmo se basa en el teorema de Bayes y predice la etiqueta de un texto como una pieza de correo electrónico o artículo de periódico. Calcula la probabilidad de cada etiqueta para una muestra dada y luego da la etiqueta con la probabilidad más alta como salida.

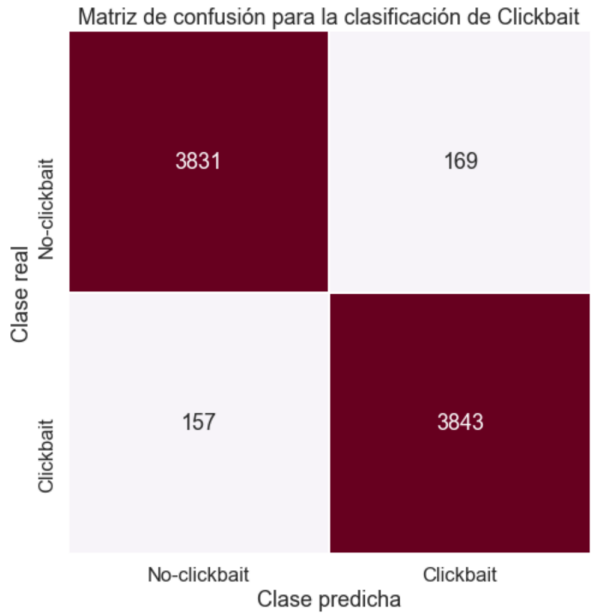
El clasificador Bayes es una colección de muchos algoritmos donde todos los comparten un principio común, que cada característica que se clasifica no está relacionado con ninguna otra característica. La presencia o ausencia de una característica no afecta a la presencia o ausencia de otra característica.

Resultados obtenidos:

```
Puntuación F1 del modelo
0.9593110334498253
Precisión del modelo
0.95925
Precisión del modelo en porcentaje
95.92500000000001 %
Informe de clasificación
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      4000
           1       0.96      0.96      0.96      4000

    accuracy                           0.96      8000
   macro avg       0.96      0.96      0.96      8000
weighted avg       0.96      0.96      0.96      8000
```

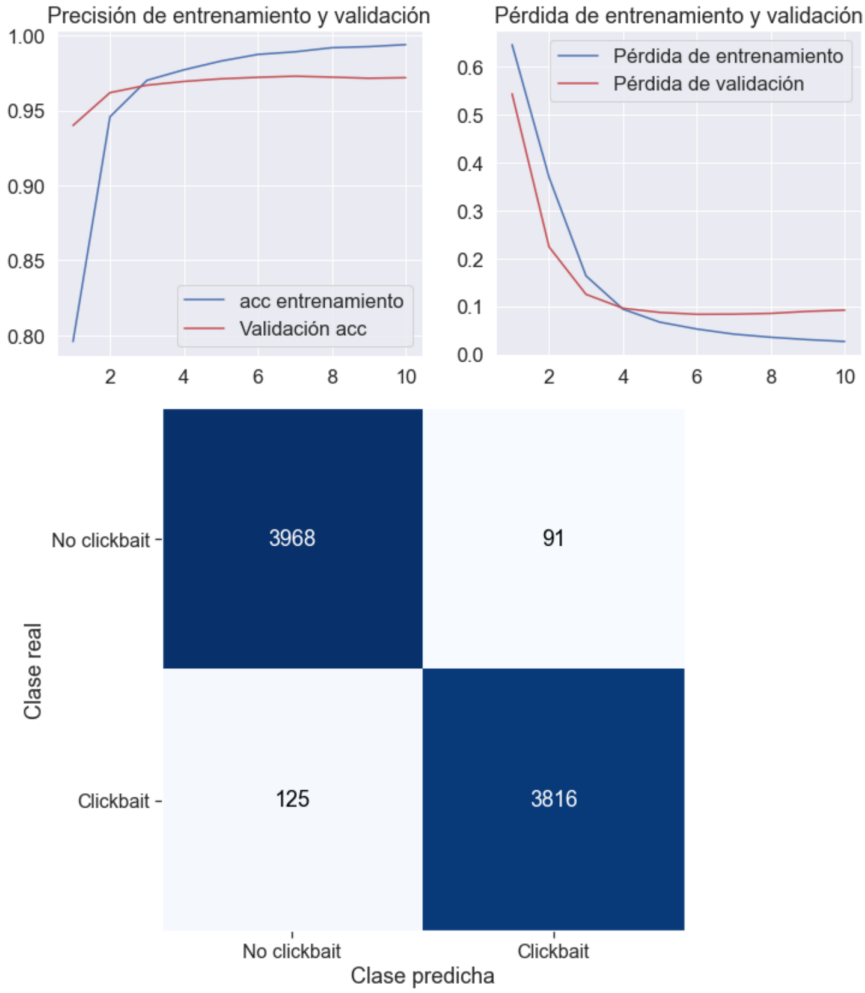


3.2 Clasificador Clickbait usando LSTM

Por otro lado, ahora clasificaremos usando LSTM, pero **¿qué es el LSTM?**

LSTM significa memoria a largo y corto plazo. LSTM es un tipo de red neuronal recurrente, pero es mejor que las redes neuronales recurrentes tradicionales en términos de memoria. Tener un buen control sobre la memorización de ciertos patrones LSTMs funciona bastante mejor. Al igual que con cualquier otro NN, LSTM puede tener múltiples capas ocultas y a medida que pasa a través de cada capa, la información relevante se mantiene y toda la información irrelevante se descarta en cada celda.

Hay muchos algoritmos de clasificación clásicos como árboles de Decisión, RFR, SVM, que pueden hacer un buen trabajo, entonces **¿por qué utilizar LSTM para la clasificación?** Una buena razón para utilizar LSTM es que es eficaz en la memorización de información importante. Si nos fijamos y otras técnicas de clasificación de red no neural que están entrenados en múltiples palabras como entradas separadas que son sólo palabras que no tienen un significado real como una frase, y al predecir la clase dará la salida según las estadísticas y no según el significado. Eso significa que cada palabra se clasifica en una de las categorías. Esto no es lo mismo en LSTM. En LSTM podemos usar una cadena de palabras múltiple para averiguar la clase a la que pertenece. Esto es muy útil mientras se trabaja con el procesamiento de lenguaje natural.



La exhaustividad (Recall) del modelo es 0.97
La precisión del modelo es 0.98

4 Conculsiones

Se pude observar que todos los métodos obtienen buenos resultados. Para empezar, para el primer método obtenemos una precisión del 95.92% (figura 3). En la matriz de confusión también se puede observar que el número de aciertos es mucho mayor que la de los fallos.

Por otro lado, los resultados obtenidos en el segundo método también son positivos. Se puede observar que hemos conseguido una precisión del 97% y una exhaustividad del 97%. Una vez más lo que se observa en la matriz de confusión coincide con lo anteriormente mencionado, ya que el número de clasificados correctamente es mucho mayor a los clasificados erróneamente.

En conclusión, los dos métodos utilizados para clasificar los titulares de las noticias tienen una precisión grande. Aun que la diferencia entre las dos sea pequeña, podemos decir que el método LSTM es mejor en este caso.