Facultad de Ciencias, UNAM
Redes Neuronales
Fake News Detection Dataset

Rubí Rojas Tania Michelle

12 de junio de 2020

1. Introducción

Si bien es cierto que internet ha permitido el intercambio de conocimiento a una escala con la que generaciones previas sólo podían soñar, también ha fundamentado lo que el ensayista *Jonathan Swift* escribió en 1710:

"La falsedad vuela y la verdad viene cojeando tras ella."

En Estados Unidos, por ejemplo, una investigación del *Pew Centre* reveló que el 62 % de los estadounidenses adultos reciben noticias a través de las redes sociales, de manera que es **cada vez más probable que más de nosotros estemos viendo -y creyendo- información que no sólo no es precisa, sino que a veces es totalmente inventada**.

Actualmente existen cientos de sitios web de noticias falsas, desde los que imitan diarios reales, hasta sitios de propraganda gubernamental, y otras que se mueven por la fina línea que divide la sátira con la desinformación.

Figura 1: National Report. Sitio web dedicado a la difusión de *fake news*

La principal razón por la que existen las noticias falsas es para desinformar a la población, ya sea para fines políticos o económicos. En el primer caso, las noticias intentan manipular el debate público a favor de determinados intereses políticos, mientras que en el segundo caso, se tratan de operaciones comerciales que buscan generar el tráfico a partir de contenidos falsos, y sobre todo, titulares sensacionalistas a los que la gente les da `clic`, pero cuya información relacionada no tiene sentido o relevancia alguna.

Ahora bien, ¿por qué es importante combatir las noticias falsas? Un motivo importante es que la información es esencial para las sociedades democráticas: difundir información falsa, imprecisa o errónea atenta contra ese derecho y afecta la toma de decisiones con fundamento por parte de los ciudadanos. En

otras palabras, afecta directamente la democracia. Por esta razón, es un deber de los ciudadanos revisar (o verificar) la calidad de los contenidos que consume y comparte.

Por otro lado, sabemos que hoy en día existe una cantidad exorbitante de información, por lo que puede ser complicado encontrar la *verdad*. También existe mucho sesgo de confirmación ya que mucha gente quiere probar que su visión del mundo es la apropiada y correcta; es decir, lo que para algunas personas es verdad, para otros no lo es. Esta cuestión abre un gran debate, por eso es importante que construyamos un criterio propio para así formar nuestro concepto de *verdad*.

> *La gente ya no sabe lo que es real y lo que es falso, muchos han dejado de creer en todo. Y eso es incluso más peligroso*
>
> *Olga Yurkova, fundadora de StopFake*

2. Objetivo

`"Dada una noticia, queremos saber si es real o falsa."`

3. Métodos

3.1. Conjunto de Datos

El conjunto de noticias que utilizaremos se obtuvo del sitio web *Kaggle*

`https://www.kaggle.com/c/fake-news/data`

el cual contiene dos archivos:

1. *train.csv*

   Contiene un conjunto de datos para entrenamiento con los siguientes atributos:

   - **id**: el *id* único de la noticia.
   - **title**: el título de la noticia.
   - **author**: el autor de la noticia.
   - **text**: el texto de la noticia (podría estar incompleto).
   - **label**: la etiqueta que clasifica la notica como REAL (0) o FAKE (1).

2. *test.csv*

   Contiene un conjunto de datos con los mismos atributos que el archivo anterior, pero sin *label*.

3.2. Preprocesamiento de Datos

Los datos contenidos en el archivo *train.csv* se pueden ver de la siguiente forma:

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1
5	5	Jackie Mason: Hollywood Would Love Trump if He...	Daniel Nussbaum	In these trying times, Jackie Mason is the Voi...	0
6	6	Life: Life Of Luxury: Elton John's 6 Favorite ...	NaN	Ever wonder how Britain's most iconic pop pian...	1
7	7	Benoît Hamon Wins French Socialist Party's Pre...	Alissa J. Rubin	PARIS — France chose an idealistic, traditi...	0
8	8	Excerpts From a Draft Script for Donald Trump'...	NaN	Donald J. Trump is scheduled to make a highly ...	0
9	9	A Back-Channel Plan for Ukraine and Russia, Co...	Megan Twohey and Scott Shane	A week before Michael T. Flynn resigned as nat...	0

Figura 2: DataFrame con las 10 primeras noticias

Renombraremos el atributo *label* (para evitar confusiones por mi parte), de manera que la etiqueta 0 la cambiamos por REAL y la etiqueta 1 la cambiamos por FAKE.

```
0    FAKE
1    REAL
2    FAKE
3    FAKE
4    FAKE
5    REAL
6    FAKE
7    REAL
8    REAL
9    REAL
Name: label, dtype: object
```

Figura 3: Etiquetas de las 10 primeras noticias

Dividimos nuestro conjunto de datos con un split del $75 - 25$ para entrenamiento y prueba.

```
X_train, X_test, y_train, y_test = train_test_split(df['text'],
                                                    labels,
                                                    test_size = 0.25,
                                                    random_state = 7)
```

Un `TfidVectorizer` se encarga de *vectorizar* un texto.

- `TF (Term Frequency)`: es el número de veces que una palabra aparece en un documento. Un valor más alto significa que un término aparece más a menudo que otros, y por lo tanto, el documento es un buen candidato cuando el término es parte de los términos de búsqueda.
- `idf (Inverse Document Frequency)`: las palabras que ocurren muchas veces en un documento, pero también muchas veces en muchos otros, pueden ser irrelevantes. Los `IDF` son una medida de la importancia de un término en todo el corpus.

El `TFidVectorizer` convierte una colección de documentos en bruto en una matriz de caracterísiticas `TF-IDF` (contiene las palabras más relevantes dentro de la colección de documentos).

Ahora bien, inicializamos un `TfidVectorizer` con palabras de stop del idioma inglés (ya que las noticias se encuentran escritas en este idioma) y una frecuencia de documento máxima de 0.7 (los términos con una frecuencia más alta serán descartados). Las palabras stop son las palabras más comúnes dentro del idioma que deben ser filtradas antes de procesar los datos del lenguaje natural.

```
# Inicializamos un TfidfVectorizer.
v = TfidfVectorizer(stop_words = 'english', max_df = 0.7)
```

Luego, entrenamos y transformamos el `vectorizer` en un conjunto de entrenamiento; y transformamos el `vectorizer` en un conjunto de prueba.

```
# Entrenamos y transformamos el conjunto de entrenamiento.
v_train = v.fit_transform(X_train.apply(lambda x: np.str_(x)))

# Transformamos el conjunto de prueba.
v_test = v.transform(X_test.apply(lambda x: np.str_(x)))
```

3.3. Plan de Acción

Para darle una solución a nuestro objetivo, utilizaremos dos *modelos* de clasificación diferentes: un perceptrón multicapa y el algoritmo de clasificación Pasivo-Agresivo.

3.3.1. Perceptrón Multicapa

3.3.1.1. Arquitectura

La arquitectura utilizada para nuestra red neuronal es:

- Capa de entrada: Tendremos tantas neuronas como entradas en el vector que le pasemos como entrada a nuestro perceptrón. En este caso, varía el valor entre el vector de entrenamiento y de testeo.
- Capa oculta: Tendremos cuatro capas ocultas con dos neuronas cada una. Esta elección la hice basándome en el método *prueba y error*, ya que estuve probando con varias arquitecturas para la capa oculta, y ésta en particular me pareció rápida en comparación a las otras arquitecturas que probé.
- Capa de salida: Tendremos dos neuronas, ya que se trata de una clasificación binaria.

3.3.1.2. Implementación

```
# Arquitectura.
mlpc = MLPClassifier(max_iter = 1000,
                     learning_rate_init = 0.0001,
                     hidden_layer_sizes = (2, 4))
```

```
# Entrenamos nuestro perceptron.
mlpc_entrenamiento = mlpc.fit(v_train, y_train)

# Predecimos en el conjunto de prueba.
mlpc_prediction = mlpc.predict(v_test)

# Obtenemos la precision.
score_mlp = accuracy_score(y_test, mlpc_prediction)
print(f'Precision: {round(score_mlp * 100, 2)}%')
```

```
Precisión: 95.77%
```

3.3.2. Algoritmo Pasivo-Agresivo

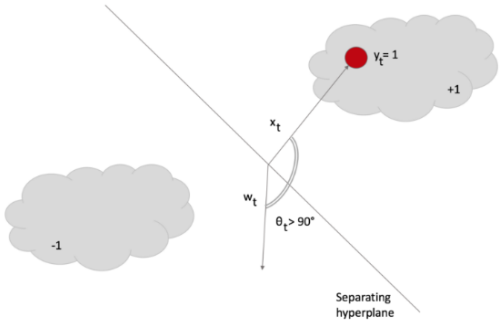
3.3.2.1. ¿Cómo funciona?

Los algoritmos Pasivo-Agresivo son una familia de algoritmos de aprendizaje en línea (tanto para la clasificación como para la regresión) propuestos por *Creammer at al.*

Un algoritmo Pasivo-Agresivo funciona genéricamente con esta regla de actualización:

$$\begin{cases} \overline{w_{t+1}} = argmin_{\bar{w}} \frac{1}{2} \|\overline{w} - \overline{w_t}\|^2 + C\xi^2 \\ L(\bar{w}; x_t, y_t) \leq \xi \end{cases}$$

Para comprender esta regla, supongamos que la variable $\xi = 0$ (y L está restringida a ser 0). Si se presenta una muestra $x(t)$, el clasificador usa el vector de peso actual para determinar el signo. Si el signo es correcto, la función de pérdida es 0 y el argumento es $w(t)$. Esto significa que el algoritmo es **pasivo** cuando se produce una clasificación correcta. Supongamos ahora que ocurrió una clasificación errónea:



El ángulo $\theta > 90°$, por lo tanto, el producto escalar es negativo y la muestra se clasifica como -1, sin embargo, su etiqueta es $+1$. En este caso, la regla de actualización se vuelve muy **agresiva** porque busca una nueva w que debe estar lo más cerca posible que la anterior (de lo contrario, el conocimiento existente se pierde inmediatamente), pero debe satisfacer $L = 0$ (en otras palabras, la clasificación debe ser correcta).

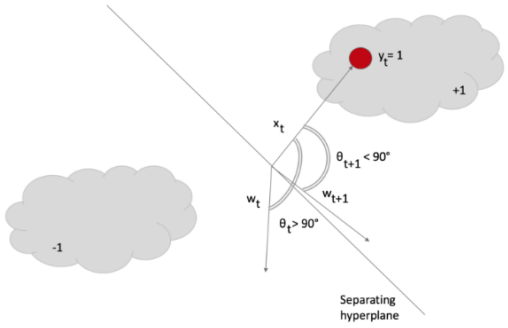
La introducción de la variable ξ permite tener márgenes blandos (como en SVM) y un grado de tolerancia controlado por el parámetro C. En particular, la función de pérdida debe ser $L \leq \xi$, lo que permite un error mayor. Los valores más altos de C producen una agresividad más fuerte (con el consiguiente mayor

riesgo de desestabilización en presencia de ruido), mientras que los valores más bajos permiten una mejor adaptación. De hecho, este tipo de algoritmos, cuando trabajan en línea, debe hacer frente a la presencia de muestras ruidosas (con etiquetas incorrectas). Es necesaria una buena robustez; de lo contrario, los cambios demasiado rápidos produce las consiguientes tasas de clasificació errónea más altas.

Después de resolver ambas condiciones de actualización, obtenemos la regla de actualización de manera cerrada:

$$\overline{w_{t+1}} = \overline{w_t} + \frac{max(0, 1 - y_t(\overline{w^T} \cdot \overline{x_t}))}{\|x_t\|^2 + \frac{1}{2C}} y_t \overline{x_t}$$

Esta regla confirma nuestras expectativas: el vector de peso se actualiza con un factor cuyo signo está determinado por $y(t)$ y cuya magnitud es proporcional al error. Tengamos en cuenta que si no hay una clasificación errónea, la fracción se hace cero, entonces $w(t+1) = w(t)$, mientras que, en caso de clasificación errónea, w rotará hacia $x(t)$ y se detendrá con una pérdida $L \leq \xi$. En la siguiente figura, el efecto se ha marcado para mostrar la rotación, sin embargo, normalmente es lo más pequeño posible.



Después de la rotación, $\theta < 90°$ y el producto escalar se vuelve negativo, por lo que la muestra se clasifica correctamente como $+1$.

3.3.2.2. Implementación

```
# Inicializamos un PassiveAggressiveClassifier.
pac = PassiveAggressiveClassifier(max_iter = 100)

# Entrenamos nuestro clasificador.
entrenamiento = pac.fit(v_train, y_train)

# Predecimos en el conjunto de prueba.
prediction_pac = pac.predict(v_test)

# Obtenemos la precision.
score = accuracy_score(y_test, prediction_pac)
print(f'Precision: {round(score * 100, 2)}%')
```

```
Precisión: 96.4%
```

4. Resultados

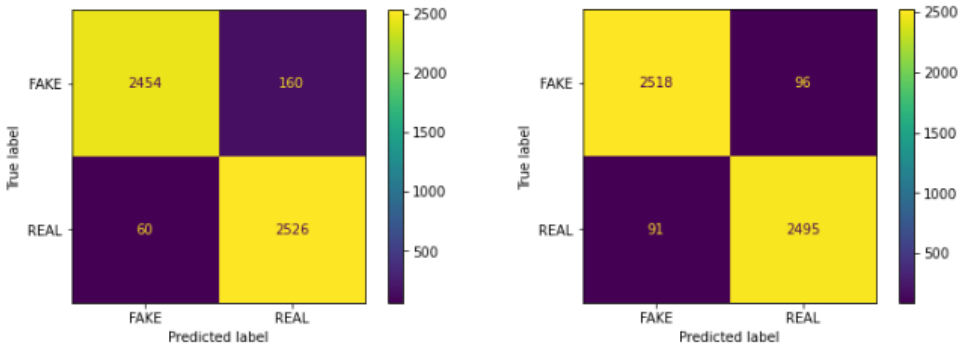
Partiéndo del hecho de que las precisiones obtenidas fueron

- Para el MLP: 95.77 %
- Para el algoritmo pasivo-agresivo: 96.4 %

compararemos las matrices de confusión y los reportes de clasificación de ambos algoritmos.

4.1. Matrices de confusión

La matriz de confusión nos dará una mejor idea de cómo está clasificando nuestro modelo, dándonos un conteo de aciertos y errores de cada una de las clases por las que estamos clasificando. Así podremos comprobar si nuestro modelo está confudiéndose entre clases, y en qué medida.



(a) MLP (b) Algoritmo Pasivo-Agresivo

Figura 4: Matrices de confusión

La diagonal principal contiene la suma de todas las predicciones correctas (es decir, las noticias que son falsas y reales), mientras que la otra diagonal refleja los errores del clasificador (falsos positivos y falsos negativos). Podemos notar que en este caso, el algoritmo pasivo-agresivo logró obtener más predicciones correctas y menos clasificaciones erróneas que el perceptrón multicapa.

4.2. Reportes de clasificación

Los reportes de clasificación muestran las principales métricas de clasificación, incluidas la precisión y la recuperación, la puntuación (media armónica de la precisión y la recuperación) y el soporte (número de observaciones de esa clase en el conjunto de entrenamiento).

	precision	recall	f1-score	support
FAKE	0.98	0.94	0.96	2614
REAL	0.94	0.98	0.96	2586
accuracy			0.96	5200
macro avg	0.96	0.96	0.96	5200
weighted avg	0.96	0.96	0.96	5200

(a) MLP

	precision	recall	f1-score	support
FAKE	0.97	0.96	0.96	2614
REAL	0.96	0.96	0.96	2586
accuracy			0.96	5200
macro avg	0.96	0.96	0.96	5200
weighted avg	0.96	0.96	0.96	5200

(b) Algoritmo Pasivo-Agresivo

Figura 5: Reportes de clasificación

Podemos notar en este caso que el perceptrón multicapa logró una mayor precisión al clasificar noticias falsas, mientras que el algoritmo pasivo-agresivo logró una mayor precisión al clasificar noticas reales.

En particular, el algoritmo pasivo-agresivo logró un mayor *balance* en cuanto a las precisiones al momento de realizar la clasificación binaria.

4.3. ¡Manos a la obra!

Ahora bien, veamos cómo funciona cada uno de nuestros modelos al momento de realizar la clasificación.

4.3.1. Conjunto de Datos

Los datos contenidos en el archivo *test.csv* se pueden ver de la siguiente forma:

	id	title	author	text
0	20800	Specter of Trump Loosens Tongues, if Not Purse...	David Streitfeld	PALO ALTO, Calif. — After years of scorning...
1	20801	Russian warships ready to strike terrorists ne...	NaN	Russian warships ready to strike terrorists ne...
2	20802	#NoDAPL: Native American Leaders Vow to Stay A...	Common Dreams	Videos #NoDAPL: Native American Leaders Vow to...
3	20803	Tim Tebow Will Attempt Another Comeback, This ...	Daniel Victor	If at first you don't succeed, try a different...
4	20804	Keiser Report: Meme Wars (E995)	Truth Broadcast Network	42 mins ago 1 Views 0 Comments 0 Likes 'For th...
5	20805	Trump is USA's antique hero. Clinton will be n...	NaN	Trump is USA's antique hero. Clinton will be n...
6	20806	Pelosi Calls for FBI Investigation to Find Out...	Pam Key	Sunday on NBC's "Meet the Press," House Minori...
7	20807	Weekly Featured Profile – Randy Shannon	Trevor Loudon	You are here: Home / *Articles of the Bound* /...
8	20808	Urban Population Booms Will Make Climate Chang...	NaN	Urban Population Booms Will Make Climate Chang...
9	20809	NaN	cognitive dissident	don't we have the receipt?

Figura 6: DataFrame con las 10 primeras noticias

4.3.2. Predicciones

Una vez que transformamos los datos (igual que transformamos el conjunto de prueba anteriormente) y predecimos sobre éstos, obtenemos los siguientes resultados:

	title	label
0	Specter of Trump Loosens Tongues, if Not Purse...	FAKE
1	Russian warships ready to strike terrorists ne...	FAKE
2	#NoDAPL: Native American Leaders Vow to Stay A...	REAL
3	Tim Tebow Will Attempt Another Comeback, This ...	REAL
4	Keiser Report: Meme Wars (E995)	REAL
5	Trump is USA's antique hero. Clinton will be n...	REAL
6	Pelosi Calls for FBI Investigation to Find Out...	FAKE
7	Weekly Featured Profile – Randy Shannon	FAKE
8	Urban Population Booms Will Make Climate Chang...	REAL
9	NaN	FAKE
10	184 U.S. generals and admirals endorse Trump f...	REAL
11	"Working Class Hero" by John Brennon	REAL
12	The Rise of Mandatory Vaccinations Means the E...	REAL
13	Communists Terrorize Small Business	REAL
14	Computer Programmer Comes Forward, Admits To B...	REAL
15	Thieves Take a Chunk of Change, All 221 Pounds...	FAKE
16	New England Patriots' Owner, Still Sore at N.F...	FAKE

(a) MLP

	title	label
0	Specter of Trump Loosens Tongues, if Not Purse...	FAKE
1	Russian warships ready to strike terrorists ne...	FAKE
2	#NoDAPL: Native American Leaders Vow to Stay A...	REAL
3	Tim Tebow Will Attempt Another Comeback, This ...	REAL
4	Keiser Report: Meme Wars (E995)	REAL
5	Trump is USA's antique hero. Clinton will be n...	REAL
6	Pelosi Calls for FBI Investigation to Find Out...	FAKE
7	Weekly Featured Profile – Randy Shannon	FAKE
8	Urban Population Booms Will Make Climate Chang...	REAL
9	NaN	FAKE
10	184 U.S. generals and admirals endorse Trump f...	REAL
11	"Working Class Hero" by John Brennon	REAL
12	The Rise of Mandatory Vaccinations Means the E...	REAL
13	Communists Terrorize Small Business	REAL
14	Computer Programmer Comes Forward, Admits To B...	REAL
15	Thieves Take a Chunk of Change, All 221 Pounds...	FAKE
16	New England Patriots' Owner, Still Sore at N.F...	FAKE

(b) Algoritmo Pasivo-Agresivo

Figura 7: DataFrame con las primeras 16 noticias

Podemos notar que no existe alguna discrepancia entre ambas clasificaciones, al menos para los primeros datos (se hizo la prueba con un conjunto de 50 noticias y aún no encontrábamos diferencias). Para mayor simplicidad sólo se mostraron 16 noticias clasificadas, pero los demás se pueden comprobar en el *notebook* que contiene todo el código.

5. Conclusiones

Ambos *modelos de clasificación* son bastante buenos. La precisón que obtuvo cada uno de ellos superó mis expectativas, ya que no creí que fuera a superar el 90 %. Como no hay que volver a implementar ya sea el perceptrón multicapa o el algoritmo de clasificación pasivo-agresivo, me parecen modelos bastantes buenos para clasificar grandes cantidades de datos. Personalmente preferiría el algoritmo pasivo-agresivo si nuestro conjunto de datos fuera mucho más grande, ya que este algoritmo es bastante rápido (frente al perceptrón) y nos podría dar una excelente clasificación *eficientemente*.

Mi principal problema al momento de elaborar el proyecto fue que no sabía cómo preprocesar los datos, pero después de leer un poco sobre el manejo de textos en PYTHON logré tener un panorama más amplio y me apoyé de las herramientas que nos proporciona el lenguaje para *vectorizar* la información.

Respecto al proyecto en general, siento que la elección del tema fue bastante acertada. Leer sobre las *fake news* me dio mucho qué pensar, es decir, sabía que existían pero no sabía a ciencia cierta por qué existían (había escuchado brevemente sobre la repercusión que tiene a nivel político). Encontré mucho material que aborda el por qué son malas las noticias falsas, en que nos pueden afectar y cómo identificarlas. Creo que esto es un tema que nos debería preocupar bastante ya que a veces con tanta información ya no sabemos lo que es verdad o no (en mi caso, a veces me siento confundida con tantos datos). Es importante que entre tanta información forjemos un criterio propio, y defendamos lo que para nosotros es *la verdad*, sin olvidar tener fuentes confiables con las que sustentar la información que compartimos y respetando siempre a las demás personas.

Referencias

[1] BB Noticias: Fake News
https://www.bbc.com/mundo/noticias-37910450

[2] ColombiaCheck
https://colombiacheck.com/investigaciones/explicador-que-son-las-noticias-falsas

[3] Xataka
https://www.xataka.com/otros/13-noticias-falsas-que-hemos-ayudado-a-difundir-por-internet-en-20

[4] Kaggle: Fake News Dataset
https://www.kaggle.com/c/fake-news/data

[5] Passive-Aggressive Algorithm
https://www.bonaccorso.eu/2017/10/06/ml-algorithms-addendum-passive-aggressive-algorithms/?subscribe=success#blog_subscription-2

[6] Text classification with Python and Scikit-Learn
https://stackabuse.com/text-classification-with-python-and-scikit-learn/

[7] Scikit-Learn: Working with text data
https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

[8] Kavita Ganesan: Tfidtransformer and TfidVectorizer: usage and differences
https://kavita-ganesan.com/tfidftransformer-tfidfvectorizer-usage-differences/

[9] LUCA: Matriz de confusión
https://empresas.blogthinkbig.com/ml-a-tu-alcance-matriz-confusion/

[10] Koldopina: Matriz de confusión
https://koldopina.com/matriz-de-confusion/

[11] Wikipedia: Tf-idf
https://es.wikipedia.org/wiki/Tf-idf