PREDICCIÓN DEL COMPORTAMIENTO DE UN STOCK FINANCIERO EN FUNCIÓN DEL LENGUAJE UTILIZADO EN TWITTER

Trabajo Fin de Máster

Máster Oficial en Big Data Science

Curso académico 2019-2020

Pablo Jordán de la Fuente

Madrid, 20 de julio de 2020

INDICE

1.- RESUMEN

En el presente trabajo se aborda la hipótesis de existencia de relación entre el lenguaje utilizado por los usuarios de la red social Twitter en sus publicaciones acerca de una empresa y el comportamiento del precio de las acciones de esta. Concretamente, el estudio toma como referencia la empresa estadounidense de vehículos eléctricos y energías limpias Tesla, de la cual se ha hecho una escucha en Twitter durante un periodo de 8 meses, y se pretende relacionar el cambio de precio en espacios temporales de diez minutos.

Para ello, se ha realizado en primer lugar un análisis exploratorio del texto obtenido de publicaciones de la red social que contengan la palabra *Tesla*, seguido de la implementación de técnicas tanto de aprendizaje supervisado, como no supervisado. Cabe destacar que el presente proyecto no tiene un objetivo inversor, sino meramente de investigación en el cual se propone una hipótesis y se estudia su veracidad.

2.- ESTADO DEL ARTE EN NLP

El Procesamiento de Lenguaje Natural (NLP) se trata de una rama del Deep Learning en la cual se persigue extraer información a partir de texto analizando el lenguaje, de forma que se puedan realizar tareas de forma automática como la generación, clasificación y resumen del texto, entre muchas otras, a través de la unión de la ciencia de datos y la lingüística. Para ello, se implementan técnicas de aprendizaje supervisado y no supervisado, como la clasificación de texto y el *topic modeling*, respectivamente, implementados en este trabajo.

En el paradigma actual dentro del Procesamiento de Lenguaje Natural está habiendo grandes avances en períodos de tiempo muy cortos, siendo una de las áreas de mayor evolución en los últimos tiempos debido a su gran interés tanto científico como empresarial. Muestra de ello es la publicación del modelo de generación de texto GPT-3 de OpenAI a finales de mayo de 2020, el cual supera en 117 veces los modelos de mayor tamaño anteriormente publicados. Gracias a publicaciones como la mencionada, es posible mantener conversaciones hiper-realistas con chatbots, realizar tareas de traducción tanto entre idiomas como entre lenguajes de programación, e incluso realizar tareas sin necesidad de escribir líneas de código.

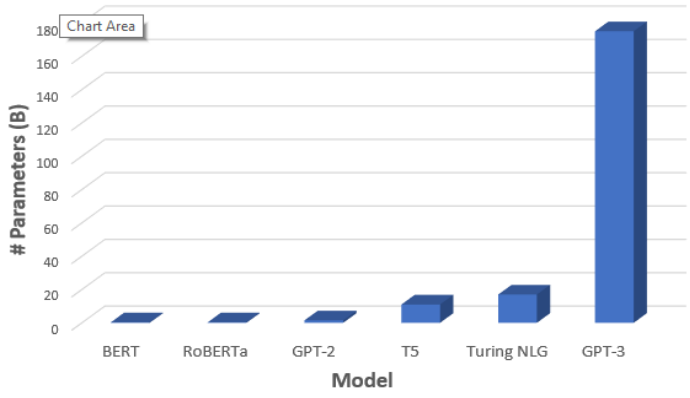


Gráfico 1. Comparativa de número de parámetros de los últimos modelos presentados.

Una de las evoluciones más disruptivas en el ámbito del Procesamiento Natural de Lenguaje ha sido el desarrollo de los Transformers por parte de Google en 2017, presentado en el paper "*Attention Is All You Need*" (Vaswani, y otros, 2017), los cuales han mejorado notablemente los resultados de avances previos en tareas como la traducción, mediante la sustitución de capas recurrentes por las denominadas *capas de atención*. Éstas, permiten al modelo conocer la posición relativa de cada palabra en el corpus. Comúnmente, se distinguen dos fases en la utilización de los Transformers: una primera en la que el modelo aprende cómo se estructura el lenguaje de forma genérica y, una segunda, en la que se busca adaptar el modelo a tareas concretas.

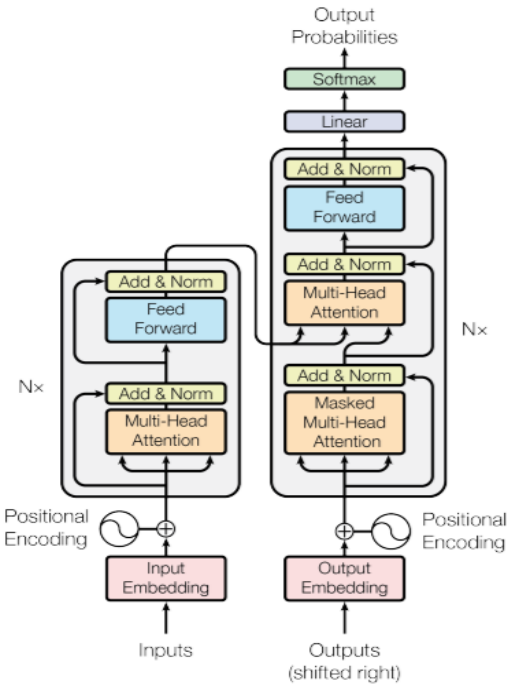


Gráfico 2. Arquitectura de un Transformer

El contexto actual de la sociedad sigue una tendencia claramente favorable hacia la digitalización y el aprendizaje automático, en la cual estas técnicas son de gran relevancia haciendo que se llegue a transformar la sociedad, tanto en el ámbito laboral como en el personal, cambiando el paradigma tal y como lo conocemos hoy en día. El desarrollo acelerado de estos campos se debe

principalmente al interés que suscita, tanto dentro de la comunidad científica, como en el mundo empresarial. Las aplicaciones más comunes que se le dan hoy en día al Procesamiento Natural del Lenguaje son, entre otras, la generación de texto, generación de recomendaciones, identificación de entidades, respuesta de preguntas, y la clasificación de texto.

3.- INTRODUCCIÓN

La motivación del presente trabajo es la aplicación de algunas de las técnicas previamente mencionadas con el objetivo de realizar predicciones en el comportamiento de las acciones de Tesla en base al lenguaje utilizado en la red social Twitter. Por ello, el punto de partida del proyecto es la formulación de nuestra hipótesis, la cual vamos a tratar de validar: la relación entre el lenguaje utilizado en las publicaciones de la red social Twitter acerca de la empresa Tesla, y el comportamiento intradía de las acciones de esta en Bolsa.

En concreto, las técnicas de Procesamiento de Lenguaje Natural utilizadas en la presente investigación son:

I. Topic Modeling
II. Clasificación de Texto según las emociones detectadas
III. Text Summarization.

Adicionalmente, se han utilizado técnicas tanto de Machine Learning como de Deep Learning durante el proceso de modelización, las cuales serán detalladas más adelante.

El flujo de trabajo seguido a lo largo del proyecto ha sido el siguiente, el cual iremos desarrollando en esta memoria:

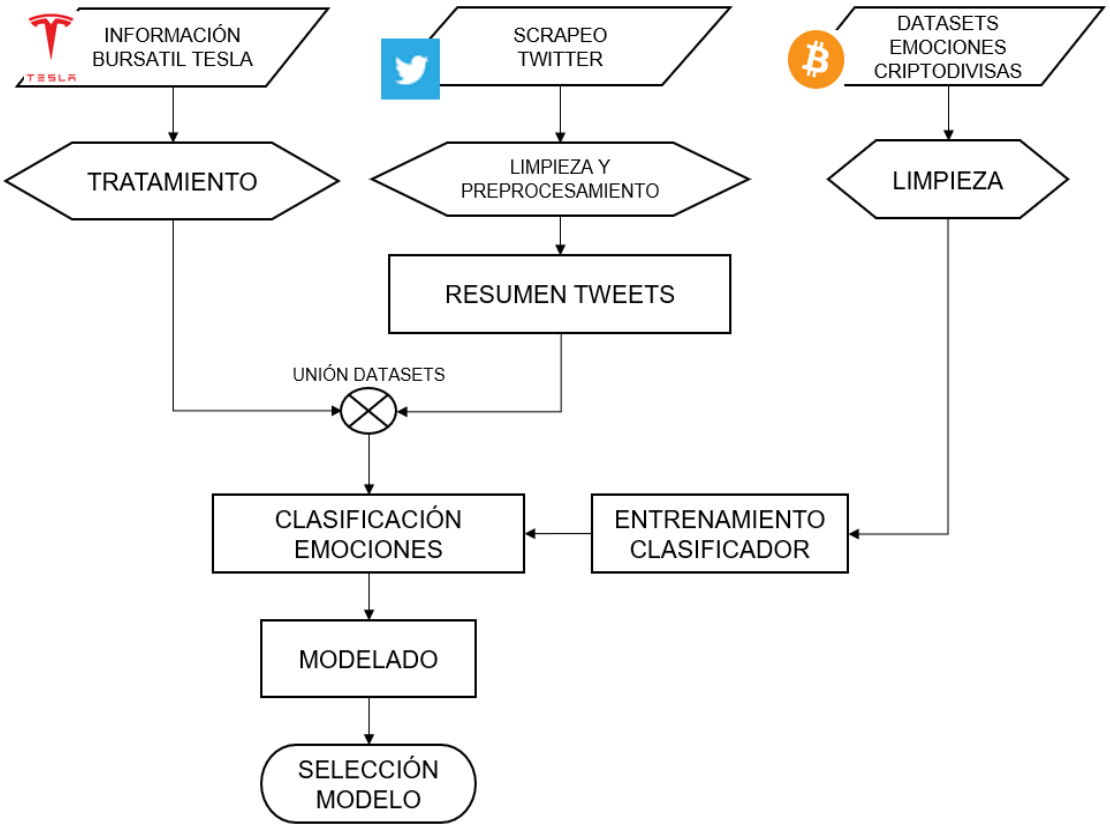


Gráfico 3. Workflow del proyecto

Concretamente, los diferentes desarrollos realizados en este proyecto se localizan en los siguientes puntos de la memoria:

1. Los datasets de partida están detallados en el punto "**conjuntos de datos utilizados**" de la memoria.
2. Los procesos de tratamiento, limpieza y preprocesamiento están detallados en el punto "**tratamiento de la información**" de la memoria.
3. Los procesos de clasificación de emociones están detallados en el punto "**modelado**" de la memoria.

4.- CONJUNTOS DE DATOS UTILIZADOS

A lo largo del análisis realizado, se ha hecho uso de tres conjuntos de datos extraídos de diferentes fuentes, los cuales se detallan a continuación:

I. Conjunto de tweets en inglés que incluyen los términos *Tesla* y/o *tsla* desde el 15 de mayo de 2019 hasta el 28 de diciembre del mismo año. Estos datos se han obtenido implementando técnicas de scraping haciendo llamadas a la API de Twitter Developer. Cada registro corresponde a un tweet o retweet, incluyendo además la lengua en la que se ha publicado, la fecha, hora y minuto de publicación, además del tipo de dispositivo desde el que se ha compartido.

II. Dataset con la información bursátil de las acciones de Tesla en slots de diez minutos, incluyendo los precios más altos y bajos del slot, el volumen, y el precio de inicio y fin.

III. Conjunto de seis datasets en formato json, clasificados por emociones con contenido relativo a criptodivisas, el cual ha sido obtenido de la comunidad online Kaggle. Este conjunto de datos se utilizará como información auxiliar para poder entrenar una red neuronal que clasifique los tweets de nuestro primer dataset por emociones.

Los datasets utilizados en el presente trabajo no tratan información personal, por lo que no se ha considerado oportuna la eliminación y/o anonimización de la misma.

5.- ANÁLISIS EXPLORATORIO

En primer lugar, se ha considerado oportuno realizar un análisis exploratorio del lenguaje utilizado en los tweets de nuestro dataset, los cuales vamos a tratar y modelos posteriormente.

5.1.- ANÁLISIS DE DISTRIBUCIONES

Empezaremos por la distribución de los tweets de nuestro dataset, en función del número de caracteres utilizados, el número de palabras utilizadas por tweet, y la longitud media de las palabras.

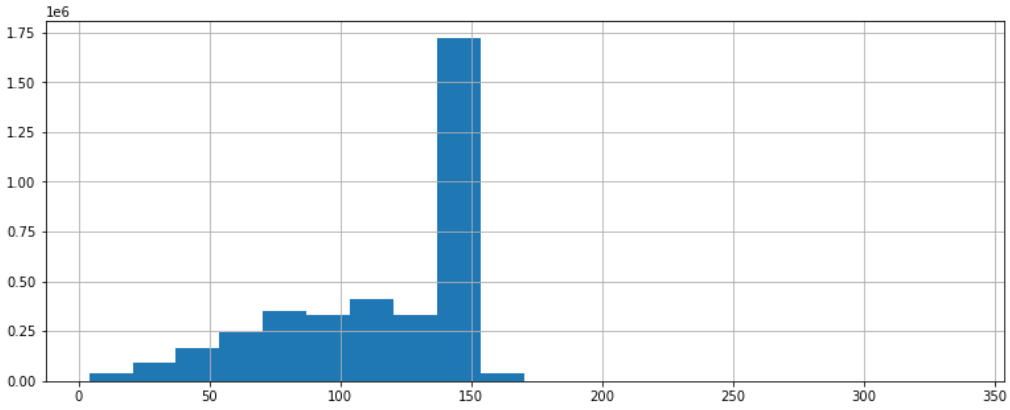


Gráfico 4. Distribución de tweets según número de caracteres.

Tal y como podemos observar en el Gráfico 4, la mayoría de los tweets tiene una longitud de 140 caracteres, longitud que coincide con el límite impuesto por la propia red social en el momento de recogida de tweets. A pesar de que no se observe con claridad gráficamente debido a su reducido volumen, existen registros con más de 140 caracteres. A priori puede parecer un error en el proceso de recogida, sin embargo, dichos tweets corresponden a fechas a partir de las cuales la red social señaló el nuevo límite de longitud en 280 caracteres por publicación.

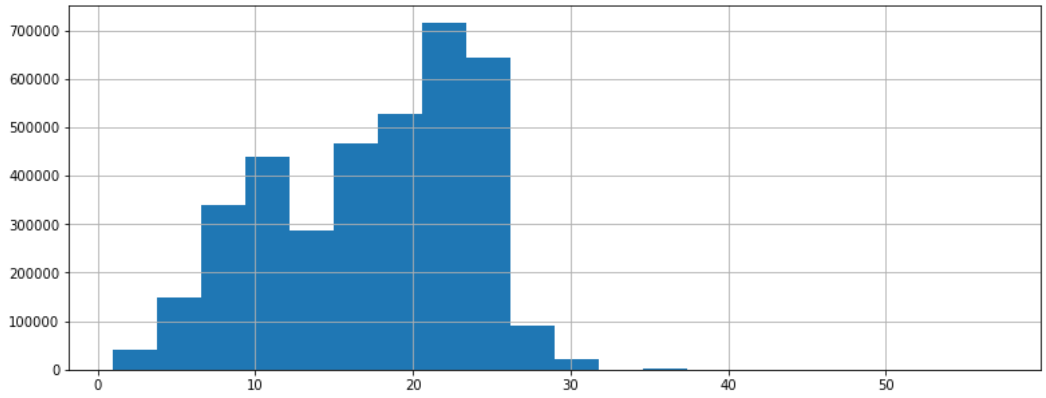


Gráfico 5. Distribución de tweets según número de palabras.

En cuanto al número de palabras utilizado en cada tweet, podemos observar que la estructura más frecuente está entre 22 y 24 palabras por publicación, tal y como se muestra en el Gráfico 5. Por último, se ha observado que la longitud de las palabras utilizadas en los tweets es relativamente pequeña, lo cual se debe al lenguaje utilizado en la red social, y a que no se ha realizado una limpieza de *stopwords*[1] previa en el corpus analizado.

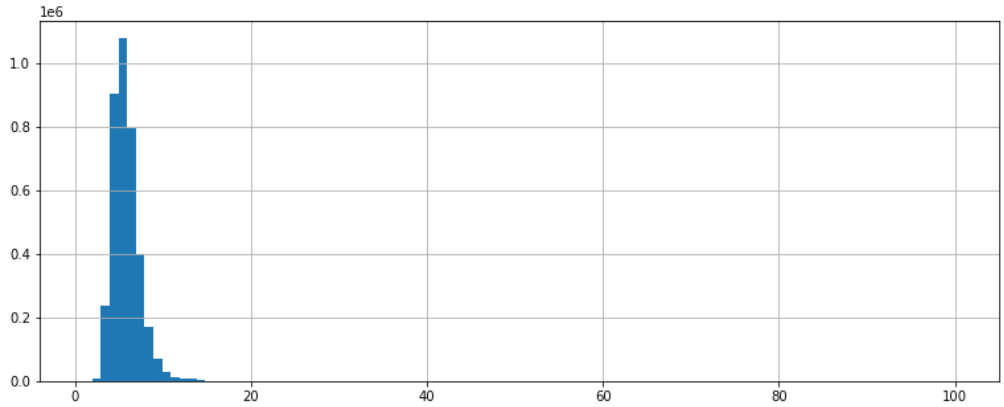


Gráfico 6. Distribución de longitud media de palabras utilizadas.

[1] Stopwords: palabras que no tienen significado por si solas. Suelen ser artículos, pronombres o preposiciones.

El siguiente paso realizado es el análisis de *stopwords* utilizadas en función de su frecuencia de aparición en el corpus analizado (Gráfico 7), así como el análisis de la frecuencia de uso de aquellas palabras que no son stopwords (Gráfico 8).

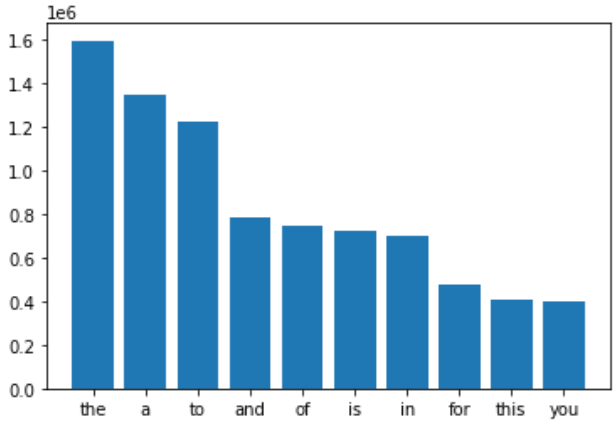


Gráfico 7. Frecuencia de stopwords en nuestro corpus.

Como conclusión del análisis realizado, podemos observar que la palabra más utilizada es “RT”, debido a que cada *retweet* es incluido en el dataset con el término “RT” al comienzo del propio registro. Esta palabra puede ser interesante de cara a realizar *feature engineering*. Sin embargo, posiblemente no aporte mucha informaión a la hora de analizar el lenguaje como es el caso. Por ello, tras el análisis realizado, se ha procedido a la eliminación de dicho término.

Adicionalmente, encontramos lógico que palabras como “*Tesla*”, *“@Tesla*” y “*$TSLA*” se encuentren dentro de los términos más frecuentes. Dado que se trata de palabras cuyo significado es similar, procederemos a la unificación de los términos mediante expresiones regulares.

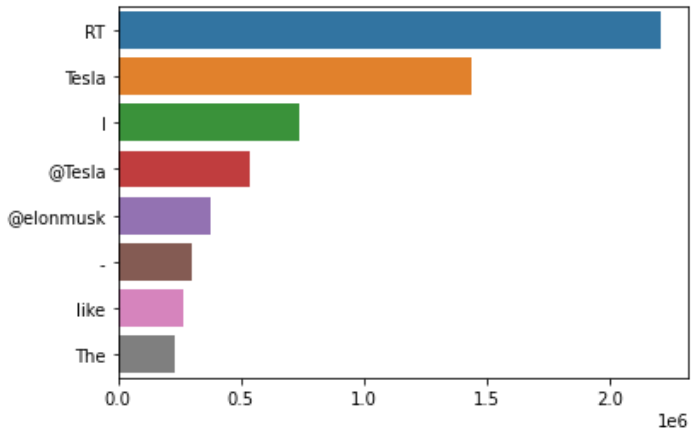


Gráfico 8. Frecuencia de palabras "no-stopword"

Además, podemos observar que se incluyen como palabras más frecuentes algunos términos que son *stopwords*. Esto se debe a que dichos términos incluyen su inicial en mayúscula, por lo que la librería *nltk* (Loper & Bird, 2002) utilizada no las ha reconocido como tal. Por ello, se ha procedido a la transformación de todos los términos del corpus a minúsculas, evitando este tipo de confusiones.

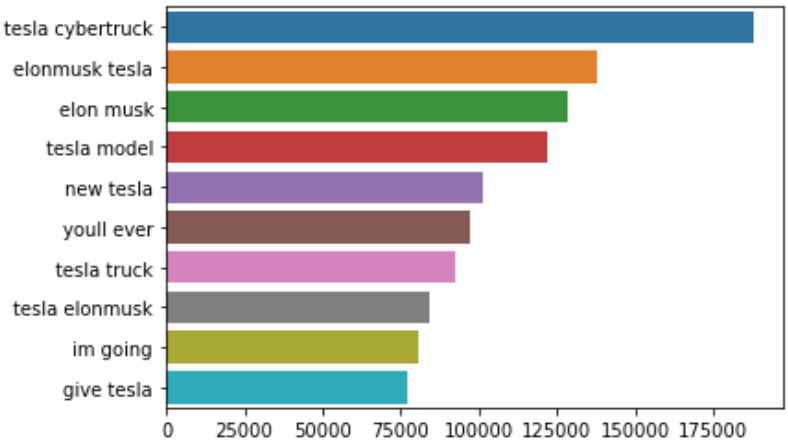


Gráfico 9. Top 10 bi-grams

Como resultado del análisis de conjuntos de palabras por n-grams (Gráfico 9) una vez realizadas las tareas de limpieza y preprocesamiento del texto, podemos confirmar que términos muy relacionados con la empresa Tesla, como pueden

ser *cybertruck*, *elon musk* o *tesla*, se encuentran presentes en los bigrams más frecuentes.

5.2.- TOPIC MODELING

Por último, se ha considerado relevante la implementación de *Topic Modeling* como parte del análisis exploratorio con el objetivo de analizar las diferentes temáticas de nuestro corpus. Para ello, se ha implementado *Latent Dirichlet Allocation*, LDA, (M. Blei, Y. Ng, & I. Jordan, 2003) como técnica de aprendizaje no supervisado el cual se basa en un modelo bayesiano jerárquico de tres niveles, cuya idea básica es que los documentos se representan como mezclas aleatorias sobre temas latentes, donde cada tema se caracteriza por una distribución sobre palabras.

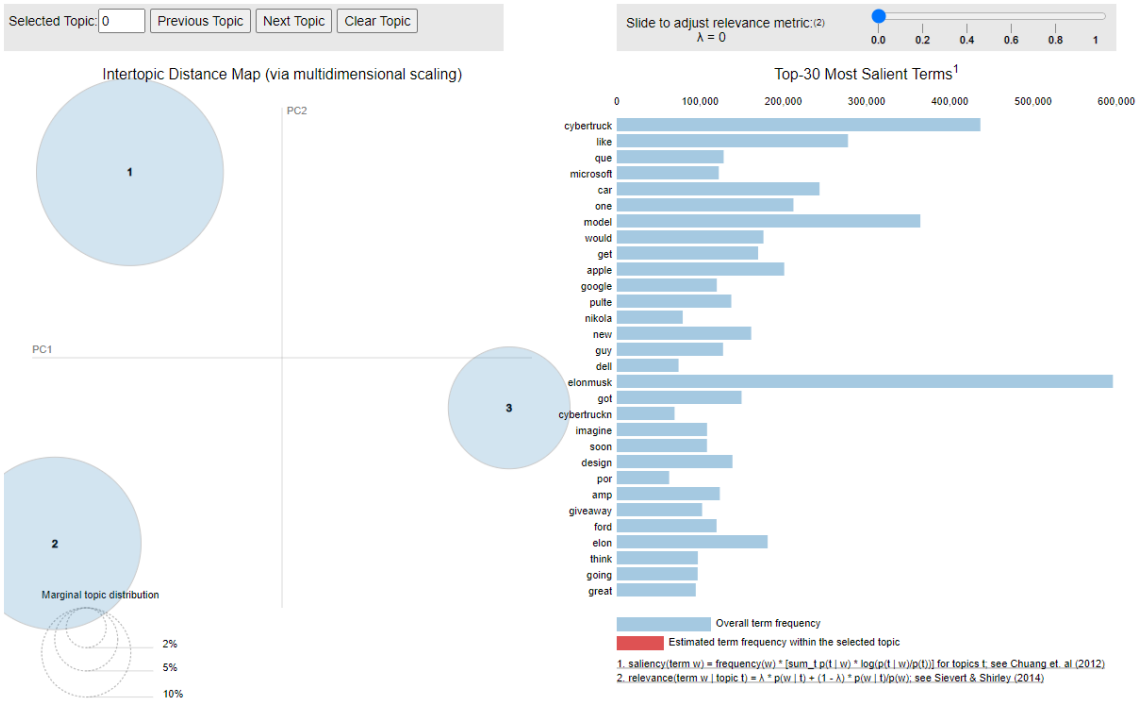


Imagen 1. Resultado general de la implementación de LDA sobre el corpus ya preprocesado.

Como resultado, se han obtenido tres agrupaciones claramente diferenciadas (Imagen 1), de las cuales, la tercera corresponde a palabras en otros idiomas como el español, francés y alemán, a pesar de haber filtrado por idioma inglés al inicio del proyecto. Esto se debe a que la API de Twitter utilizada para extraer las publicaciones de la red social, adjudica el idioma en función la configuración de la cuenta del usuario que ha publicado. De tal manera que, si una persona de habla latina tiene configurada la cuenta en idioma inglés, contabilizará como tweet en inglés. La primera agrupación corresponde al 44% de los tokens de nuestro corpus, seguido del 37,4% del segundo y, por último, un 18,7% del tercer tema.

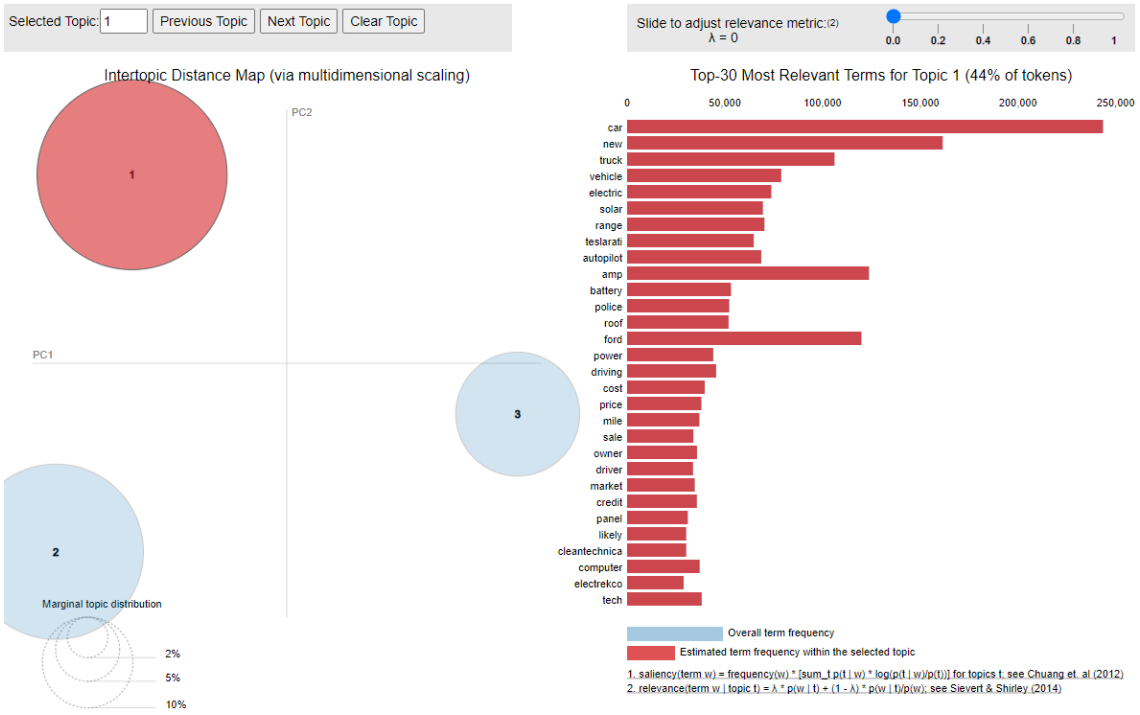


Imagen 2. Topic 1 resultante de la implementación de LDA.

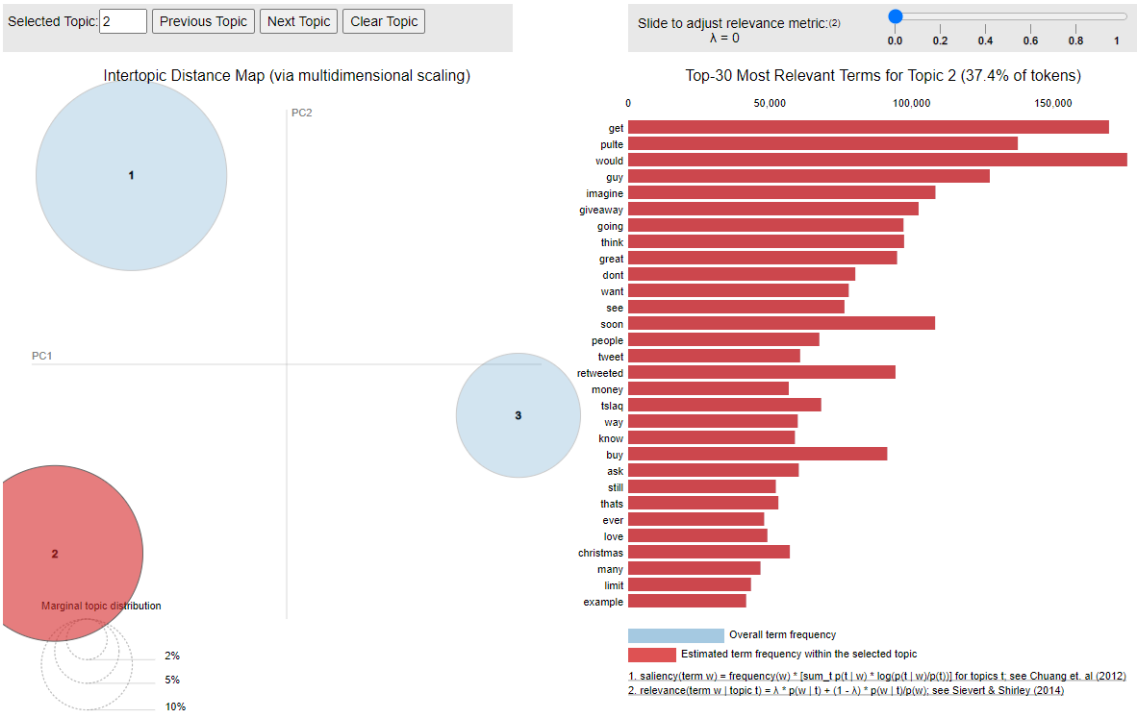


Imagen 3. Topic 2 resultante de la implementación de LDA.

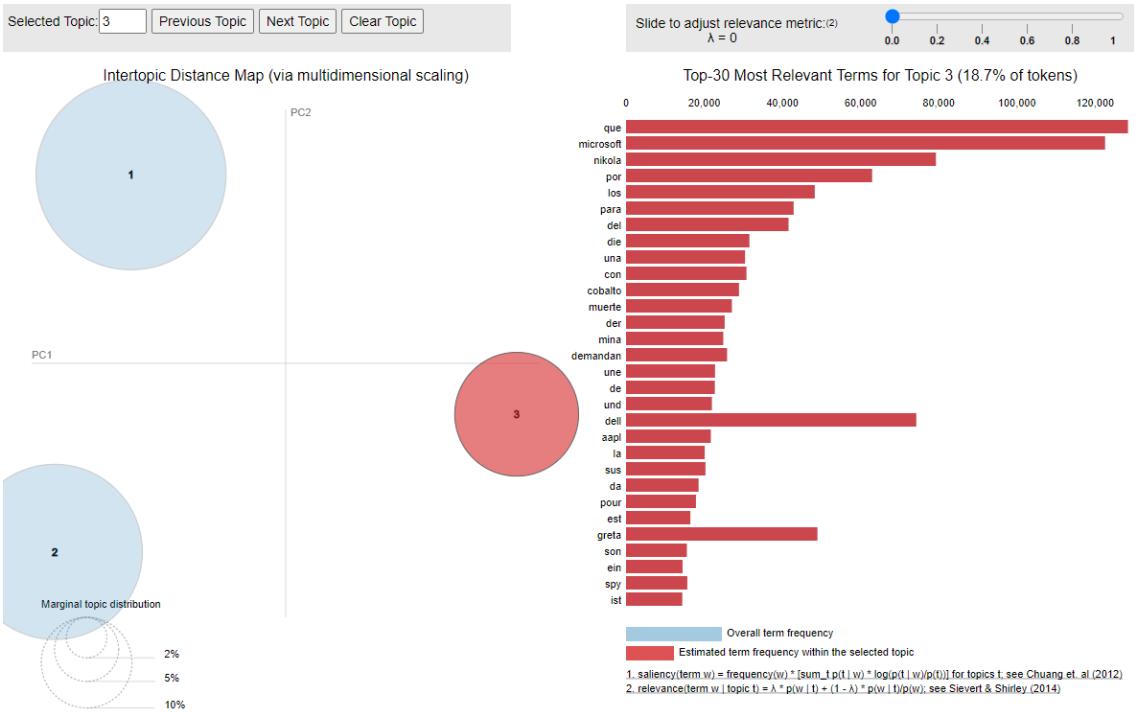


Imagen 4. Topic 3 resultante de la implementación de LDA.

Adicionalmente, se han realizado pruebas para diferentes números de agrupaciones o temáticas. Como resultado, cuando se trató de agrupar el corpus en cuatro, el resultado obtenido fueron dos clústers muy cercanos que corresponden a la primera agrupación del análisis realizado previamente para tres *clústers*. Las otras dos agrupaciones resultantes, corresponden a las temáticas dos y tres del análisis anteriormente mencionado. Por ello, se ha considerado que la mejor aproximación posible para nuestro corpus se trata de una agrupación en tres clústers diferentes.

Los resultados obtenidos del *topic modeling* se consideran altamente interesantes de cara a poder realizar diferentes tratamientos y modelados para cada uno de los *topics* identificados. Sin embargo, dado que se escapa del alcance del proyecto actual, no se ha realizado tal distinción.

6.- TRATAMIENTO DE LA INFORMACIÓN

Una vez realizado el análisis exploratorio de nuestro dataset de texto, se ha procedido a la aplicación de técnicas de Machine Learning y Deep Learning tanto para la generación de nuevas variables que puedan aportar información a nuestro modelo, como para el proceso de modelización. El lenguaje de trabajo utilizado a lo largo de todo el trabajo ha sido *Python*, debido al volumen de librerías disponibles que facilitan el desarrollo e implementación de este. En cuanto a entorno de desarrollo de los propios *notebooks* se ha elegido *Jupyter*.

6.1.- DATA CLEANING Y PREPARACIÓN DE DATASETS

En primer lugar, previo a la aplicación de las técnicas mencionadas, se han implementado tareas de *data cleaning*, en las cuales se ha realizado:

I. Eliminación de puntuaciones y caracteres no alfanuméricos mediante expresiones regulares.
II. Transformación de todos los caracteres a minúscula.
III. Eliminación de emoticonos.
IV. Filtro del conjunto de datos por idioma inglés.

Adicionalmente, se han eliminado *stopwords* de nuestro corpus, y reducido la variabilidad de palabras quedándonos sólo con la raíz de estas mediante *stemming*, siendo necesario el uso del *word_tokenizer* para la implementación de esta. Las técnicas comentadas se han implementado haciendo uso de la librería *nltk*.

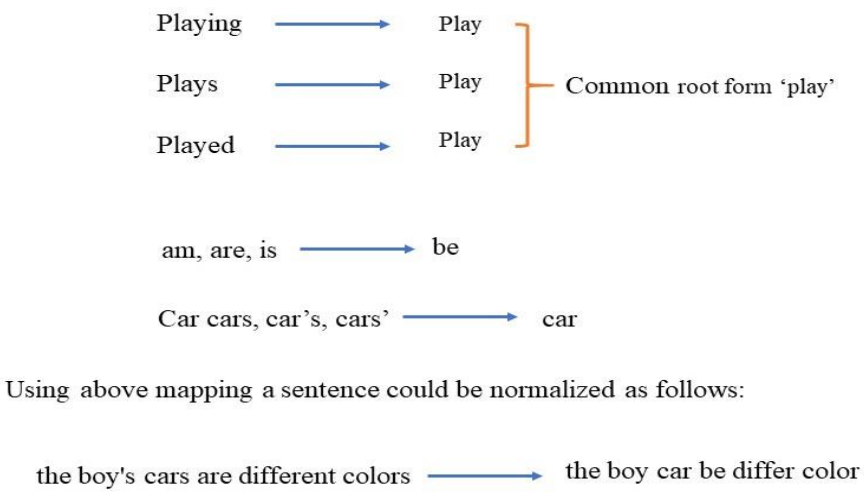


Imagen 5. Ejemplo de Stemming

Para el proceso de modelización es necesaria la unión de los conjuntos de datos con tweets de Tesla y el conjunto de datos de información bursátil de las acciones de Tesla. Para ello, se ha generado mediante los campos de fecha, hora y minuto, una variable que diferencia los registros en slots de diez minutos. Dado que el horario de apertura de la bolsa americana es limitado (de 3.30 p.m. CET a 10 p.m. de España), se ha optado por la eliminación de todos los tweets que no entren dentro de dicha franja horaria, ya que el análisis a realizar es intradía con slots temporales muy reducidos.

6.2.- TEXT SUMMARIZATION

Una vez unidos los dos datasets, y realizado el preprocesamiento del texto, se ha procedido a la implementación de *Text Summarization*, la cual se basa en conseguir resumir conjuntos de texto en base a la frecuencia de las palabras utilizadas. Para ello, se ha utilizado la librería *Tf-IDF*, la cual nos permite realizar esta tarea con relativa sencillez. El objetivo de este paso es reducir el volumen de texto a procesar, quedándonos con las partes más relevantes del texto. Con el modelo de resumen de texto implementado, se ha logrado pasar de una

longitud media de alrededor de 3000 palabras (sin stopwords) por slot de diez minutos, a alrededor de 200 como se puede observar en los gráficos comparativos adjuntos, Gráfico 11 y Gráfico 12, en los cuales, cada punto representa la longitud de cada uno de los registros de nuestro dataset, es decir, cada punto representa un slot de diez minutos de tweets.

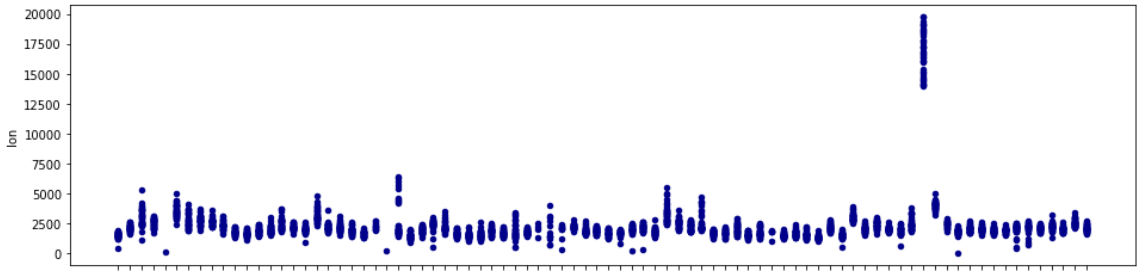


Gráfico 10. Distribución temporal de la longitud de los registros sin stopwords.

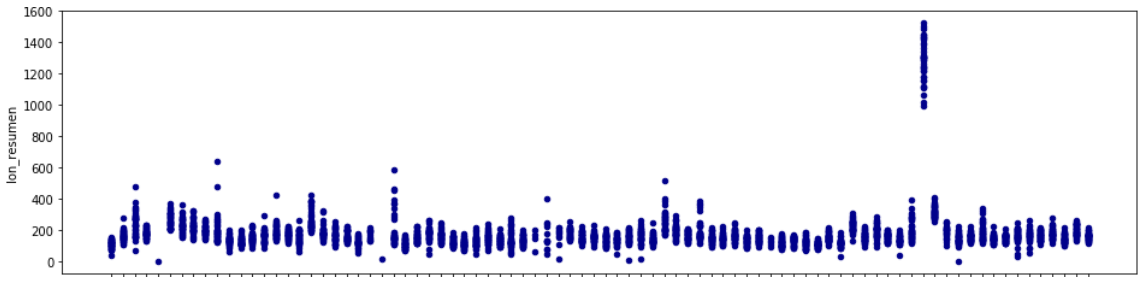


Gráfico 11. Distribución temporal de la longitud de los registros resumidos sin stopwords

Llama la atención la longitud significativamente superior de uno de los registros, el cual corresponde a la presentación del modelo Cybertruck por parte de la empresa Tesla, la cual fue muy polémica, dando lugar a un elevado volumen de contenido en las redes.

6.3.- FEATURE ENGINEERING

Una vez realizada la limpieza del texto, se ha procedido a la generación de variables que pueden ser de utilidad aportando información al modelo tales como:

- Volumen de tweets por cada slot: *tweet_count*
- Longitud media de los tweets de cada slot: *length*
- Número de menciones: *n_mentions*
- Número de menciones a la cuenta oficial de Elon Musk: *elon_mentions*
- Número de menciones a la cuenta oficial de Tesla: *tesla_mentions*
- Número de links incluidos en los tweets: *link*
- Número de tweets retweeteados en cada slot: *RT*

Con el objetivo de incrementar la información aportada al modelo mediante la inclusión de nuevas variables, se ha considerado relevante la inclusión de las emociones de los tweets a través de la implementación de un clasificador de texto mediante redes neuronales. Dado que no contamos con los tweets sobre Tesla de nuestro dataset clasificados por emociones, se ha utilizado como recurso adicional un conjunto de seis archivos JSON obtenidos de la web Kaggle, los cuales contienen tweets con información relacionada con criptodivisas, correspondiendo cada uno los archivos a una emoción. Las emociones incluidas son:

I. Anger.
II. Fear
III. Greed
IV. Hateful
V. Joy
VI. Sadness

Estos archivos JSON se han unificado en un mismo dataset, el cual se ha preprocesado realizando las mismas tareas de limpieza de texto que se han realizado sobre el dataset original en pasos anteriores. Adicionalmente, se han eliminado palabras que están altamente relacionadas con la temática criptodivisas, pero no lo están con nuestro tema de interés, como son *bitcoin*, *ethereum*, *crypto*, *btc* y *litecoin*, entre otras. Una vez realizada la limpieza de texto descrita, se ha utilizado *Synthetic Minority Over-Sampling Technique* (SMOTE) (V. Chawla, W. Bowyer, O. Hall, & Kegelmeyer, 2002) para solucionar los problemas de balanceo de emociones que existían en el dataset de criptodivisas.

En cuanto al clasificador de texto, se ha entrenado una red neuronal con una arquitectura en la que se utilizan Embedding Layer, capas LSTM, y capas densas. Como función de pérdida a minimizar se ha utilizado *sparse_categorical_crossentropy*, como optimizador *adam*, y como métrica de referencia *accuracy*, obteniendo la matriz de confusión que se muestra en Gráfico 13 sobre test. Existe margen de mejora en la clasificación de la clase 3, correspondiente a la emoción *hateful*, ya que nuestro clasificador lo confunde con la clase 0, *anger*. Dado que se trata de términos relativamente relacionados, se han considerado como satisfactorios los resultados obtenidos, y se utilizará el modelo entrenado para la clasificación del dataset original por emociones, obteniendo de esta manera una nueva variable para el entrenamiento de nuestro modelo.

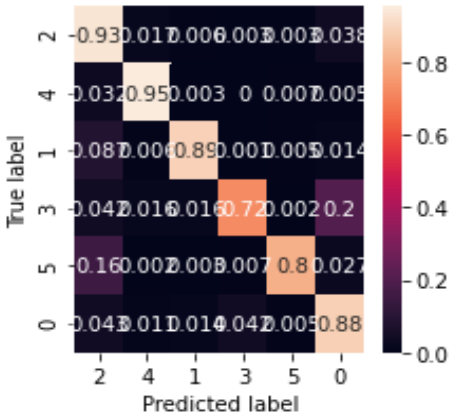


Gráfico 12. Matriz de confusión del clasificador de emociones sobre el dataset de Test.

7.- MODELIZACIÓN

En el apartado de modelización, se han considerado diversas alternativas de cara a poder valorar la opción que aporta la mejor solución. Para ello, se han considerado desde modelos de Machine Learning de forma individual como KNN, XGBoosting y Regresión Logística entre otros, hasta el entrenamiento de redes neuronales de diferentes arquitecturas, y modelos *stackeados*.

7.1.- DATASET PARA MODELADO

Las variables que se han considerado durante el modelado han sido las siguientes:

```
['open', 'low', 'high', 'volume', 'tweet_count', 'length', 'n_menti
ons', 'elon_mentions', 'tesla_mentions', 'link', 'RT', 'emotions']
```

- *Open*: precio de las acciones en la apertura del slot
- *Low*: precio más bajo de las acciones en cada slot
- *High*: precio más alto de las acciones en cada slot
- *Volumen*: volumen bursátil operado durante el slot
- *Tweet_count*: número de tweets publicados en el slot, incluyendo retweets
- *Length*: Lonigtud del resumen realizado para cada slot, el cual hemos comprobado que mantiene las proporciones del conjunto de datos original.
- *N_mentions*: Total de menciones a usuarios en cada uno de los slots.
- *Tesla_mentions*: Total de menciones a la cuenta oficial de Tesla en cada uno de los slots.
- *Link*: Total de links incluidos en los tweets de cada slot.
- *RT*: Total de retweets incluidos en cada slot.

- *Emotions*: emociones clasificadas para cada uno de los slots en pasos anteriores. Dado que se trata de una variable categórica, se ha optado por transformar esta columna a través de *one-hot encoding*.

	open	low	high	volume	tweet_count	length	n_mentions	elon_mentions	tesla_mentions	link	RT	0	1	2	3	4	5	CLASE
0	229.238	226.557	229.667	0.0997	155	113.402685	304	12	41	90	155	0	0	0	0	0	1	0
1	226.898	225.227	227.328	0.1783	207	122.841026	300	15	41	148	207	0	0	0	0	0	1	0
2	226.927	226.207	227.888	0.1280	198	121.530928	267	14	39	130	198	0	0	0	0	0	1	1
3	227.538	227.267	228.928	0.0927	197	117.746032	218	15	37	132	197	0	0	0	0	0	1	1
4	227.597	226.867	231.408	0.1421	177	119.710059	257	21	28	109	177	0	0	0	0	0	1	1

Imagen 6. Extracción de los cinco primeros registros del dataset de entrenamiento.

7.2.- PREPARACIÓN DATASET PARA MODELADO

Antes de proceder a la modelización, se ha procedido a la estandarización de los datos a través de *StandardScaler()*, el cual transforma las variables haciendo que la media de la distribución sea cero, y la desviación estándar sea 1.

Separamos nuestro conjunto de datos entre dataset de *train* y dataset de *test*. El primero de ellos lo utilizaremos para la fase de entrenamiento, y el segundo de ellos para poder evaluar la precisión del modelo entrenado previamente. Para ellos, utilizaremos la librería de python *sklearn* y, en concreto, *train_test_split*, reservando para entrenamiento el 60% del dataset, y para test el 40% restante. Con el objetivo de obtener resultados reproducibles, se ha fijado la semilla con *random_state = 42*.

A continuación, se presentarán los modelos entrenados, así como los resultados obtenidos para cada uno de ellos. Para los modelos de *machine learning* se ha utilizado la librería *sklearn*, y para las redes neuronales se han utilizado diferentes arquitecturas con capas de la librería *keras*.

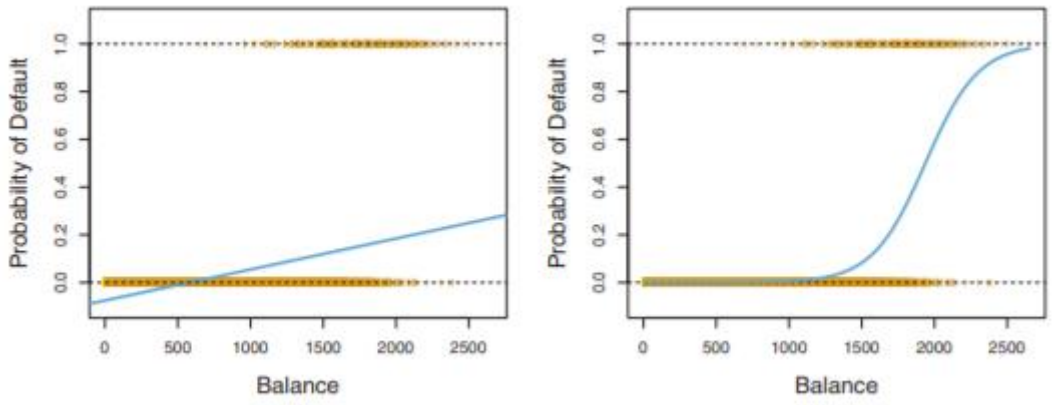
Además, para poder analizar los resultados obtenidos durante el proceso de modelado, se ha calculado la matriz de confusión para cada uno de ellos. En ella, se muestra la predicción realizada por cada uno de nuestros modelos para cada una de las clases, en base a la siguiente lógica:

		Valores Reales	
		0	1
Predicción	0	*True Negative*	*False Positive*
	1	*False Negative*	*True Positive*

En la tabla anterior, se incluirá el número de registros clasificados correcta, o incorrectamente según aplique.

7.3.- REGRESIÓN LOGÍSTICA

Se trata de un modelo utilizado tanto en problemas estadísticos tradicionales de clasificación, como en problemas más avanzados de Machine Learning, en el cual se predice si un suceso es Positivo (toma valor 1) o Negativo (toma valor 0), en lugar de predecir un target continuo como sucede en la regresión linear. En los siguientes gráficos se muestra la diferencia de ajuste entre una regresión lineal (izquierda), y una regresión logística (derecha), ante un problema de clasificación entre 0 y 1. En ambos casos, la línea azul representa la probabilidad de que nuestra variable objetivo tome valor 0 o 1. Como podemos comprobar, dado que nuestro problema es de clasificación binaria, tiene más sentido realizar el modelado según la regresión logística.



Por defecto, se tiende a clasificar que una observación pertenece a la clase 1 (clase superior de nuestro gráfico), si la probabilidad obtenida es mayor a 0.5, y viceversa. Sin embargo, puede definirse un *threshold* inferior o superior en función del interés del modelo en particular. En nuestro caso, no modificaremos dicho umbral, por lo que toda predicción en la que se obtenga una probabilidad igual o superior a 0.5, será considerada como clase 1, y toda predicción con una probabilidad resultante inferior a 0.5, será considerada como clase 0.

Para obtener dichas predicciones en un rango entre 0 y 1, no podemos utilizar la función de la regresión lineal, dado que se pueden obtener valores fuera de dicho rango. Por ello, se utilizará la siguiente expresión:

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

El objetivo de nuestro modelo será, a partir de los datos de entrenamiento proporcionados, estimar los valores de los coeficientes β_0 y β_1. Para ello, utilizaremos el método de máxima verosimilitud, a partir del cual se obtienen los estimadores de los parámetros desconocidos que hacen que se maximice la

función de verosimilitud, es decir, la probabilidad de haber observado precisamente los datos que se han obtenido.

Resultado: 56.71 %

Matriz de confusión obtenida:

```
[[232 338]
 [175 440]]
```

7.4.- LDA (Linear Discriminant Analysis)

Al igual que el Análisis de Componentes Principales (PCA), LDA se basa en reducir la dimensionalidad del dataset. Sin embargo, a diferencia de PCA, se centra en maximizar la separación entre las categorías que queremos clasificar de forma que puedan diferenciarse lo mejor posible. Para ello, LDA define nuevas dimensiones a través de las cuales busca maximizar la distancia entre las medias de cada clase (numerador), y minimizar la variabilidad de cada categoría (denominador, en la siguiente expresión:

$$\frac{(\mu_0 - \mu_1)^2}{{s_0}^2 + {s_1}^2}$$

Idealmente, la diferencia entre las medias de cada categoría debe ser lo más grande posible, y la suma de la variabilidad de cada una de las categorías debe ser lo más pequeña posible.

Dado que nuestro dataset es de reducidas dimensiones (2961 slots de 10 minutos cada uno durante el horario de apertura), merece la pena modelar con LDA a pesar de haber entrenado previamente regresión logística. Además, en

caso de que las clases estén separadas los parámetros estimados son altamente inestables en caso de la regresión logística, al contrario que LDA. A pesar de que este modelo se utiliza comúnmente más en problemas con más de dos clases a clasificar, no impide que modelemos nuestros datos con él.

Resultado: 84,05 %

Matriz de confusión obtenida:

```
[[460 110]
 [ 79 536]]
```

7.5.- K-NEIREST NEIGHBORS

Se trata de un modelo clasificador no paramétrico, el cual se basa en predecir de qué clase es una observación en base a los *K* vecinos más cercanos. Para ello, se han probado diversos valores de K, ya que valores muy bajos provocan predicciones con variabilidad muy alta y sesgo muy bajo y, valores muy altos, provocan predicciones poco flexibles, y que se ajustan poco a la realidad. En este caso los resultados de las predicciones realizadas en ambos casos han sido insatisfactorios, obteniendo el mejor resultado para un *k* = 50.

Resultado (n_neighbors = 50): 51.56 %

Matriz de confusión obtenida:

```
[[308 262]
 [312 303]]
```

7.6.- SUPPORT VECTOR MACHINE

Se basa en un método de clasificación en el que a través de la definición de un hiperplano que separe las dos clases del dataset de entrenamiento, sea capaz de predecir las clases del conjunto de datos de *test* en función del lado del hiperplano del que "*quede*". Como particularidad, cabe destacar que este modelo intenta que exista la máxima distancia posible entre el hiperplano definido y los puntos más cercanos al mismo.

Resultado: 53.58 %

Matriz de confusión obtenida:

```
[[139 431]
 [167 448]]
```

7.7.- RANDOM FOREST

Se trata de un tipo de modelo de aprendizaje Supervisado, es decir, tiene un target informado, que puede utilizarse tanto para problemas de Clasificación como en este caso, como para problemas de Regresión. Se basa en la construcción de árboles de decisión sobre datasets obtenidos con Bootstrap a partir del original, de forma que cada árbol entrene sobre un dataset diferente. Su resultado es la media de las predicciones realizadas en todos los árboles.

A diferencia del Bagging, a la hora de realizar un Split, se elige una muestra aleatoria de N predictores o variables candidatas para hacer el Split. Esto hace que el modelo no tenga en cuenta todas las variables disponibles en el modelo para entrenar en un árbol. A priori, puede parecer una desventaja, pero lo que

consigue es que el modelo en su totalidad (entre todos los árboles) sí que analice las diferentes variables, y no se quede ninguna sin ver. Por el contrario, Bagging se queda con los mejores predictores y pueden quedar variables sin observar/estudiar en el entrenamiento. Si lo comparamos con Boosting, su principal diferencia es que Random Forest construye los árboles de forma independiente entre ellos, a diferencia de Boosting que los construye secuencialmente.

Resultado: 52.32 %

Matriz de confusión obtenida:

```
[[263 307]
 [291 324]]
```

7.8.- XGBOOSTING

Modelo creado a partir de árboles de decisión, utilizado tanto en problemas de regresión como de clasificación. A diferencia de Bagging y RF, en Boosting los árboles no se crean de forma independiente, sino de forma secuencial, aprendiendo de los residuales de los anteriores árboles calculados. Tampoco implica Bootstrap sampling, en su lugar, cada árbol se entrena sobre una muestra diferente del dataset original. Puede definirse el número de nodos que limiten el crecimiento del árbol.

En este caso, se ha optado por realizar una búsqueda de hiperparámetros haciendo uso de Hyperopt, la cual, a diferencia de GridSearch, a medida que realiza la optimización de hiperparámetros va eligiendo/descartando las zonas

del espacio de hiperparámetros en las que cree que puede obtener mejores rendimientos, en base a los resultados que obtiene.

Resultado: 52.57%

Matriz de confusión obtenida:

```
[[315 255]
 [307 308]]
```

7.9.- MODELO STACKEADOS

Una vez implementados los modelos de forma individual, se ha implementado una arquitectura de modelos stackeados con los modelos que mejores resultados nos han aportado hasta ahora: Regresión Logística y Análisis Discriminante Lineal (LDA). Esta estrategia se basa en aprender a obtener la mejor combinación de predicciones de dos o más modelos.

En concreto la estructura implementada tiene dos niveles, un primer nivel en el que entrenaremos ambos modelos, y un segundo nivel en el que nuestro *"meta-modelo"*, LDA en este caso, combina las predicciones de los dos primeros. ¿Cómo funciona? A la hora de entrenar el modelo stackeado, hay datos de entrenamiento que no se utilizan con los modelos base. Se alimenta con dichos datos a los modelos base, se realizan predicciones, las cuales serán output del primer nivel de la estructura, e input del segundo nivel, el "*metamodelo"*. De esta manera, entrenamos al segundo nivel de nuestra estructura para aprender a elegir con qué predicción se tiene que quedar.

Como resultado logramos mejorar levemente la precisión obtenida con el modelo LDA descrito anteriormente. Adicionalmente, se han entrenado otras estructuras de *stacking* en las que se combinaban otros modelos analizados a lo largo del presente trabajo, obteniendo para todos ellos peores resultados.

Resultado: 85.23 %

7.10.- RED NEURONAL

Por último, se han probado diferentes arquitecturas de redes neuronales en las que se han combinado capas de la librería *keras*, probando combinaciones a través de búsqueda de hiperparámetros con *GridSearch* de:

- Número de *epoches*
- Número de capas *Dense* incluidas
- Número de capas *LSTM* incluidas
- Función de activación
- Tamaño de los lotes (batch_size)
- Inclusión / No inclusión de *Dropout*, con diferentes valores.

Dado que el tamaño de nuestro dataset es reducido, el entrenamiento con redes neuronales no es el método ideal para afrontar este problema de clasificación. Prueba de ello son los pobres resultados obtenidos, no mejorando los obtenidos con la regresión logística, anteriormente mencionados.

7.11.- RESULTADOS

	Regresión logística	*LDA*	*KNN*	*SVM*	*Random Forest*	*XGBoosting*	*Stacking*
Accuracy	56.71 %	84.05 %	51.56 %	53.58 %	52.32 %	52.57 %	85.23%

8.- CONCLUSIONES

Como hemos podido comprobar, hemos obtenido los mejores resultados con un modelo de stacking entre LDA y regresión logística, con un 85.23% de precisión, un resultado muy superior a los obtenidos con técnicas más avanzadas y que, a priori, en otros problemas suelen dar mejores resultados como pueden ser XGBoost o Redes Neuronales. Dado que nuestro conjunto de datos es de dimensiones reducidas, para muchos de estos modelos el número de registros ha sido insuficiente como para poder aprender de los datos de cara a realizar predicciones fiables.

En base a los resultados obtenidos, podemos afirmar que existe cierta relación entre la información modelizada y el comportamiento de las acciones de Tesla. Sin embargo, analizando la importancia en la modelización de cada una de las *features,* se observa que aquellas que han sido generadas con técnicas de NLP como son las emociones, no son las que más información aportan. Por tanto, no podemos afirmar que existe una relación entre el lenguaje utilizado en la red social Twitter sobre Tesla, y el comportamiento de las acciones de esta en Bolsa.

Adicionalmente, se desaconseja utilizar el modelo desarrollado con objetivo inversor, dado que el modelo únicamente realiza predicciones sobre el comportamiento del precio de las acciones, si sube o baja el precio, y no predice en qué cuantía lo hará. Adicionalmente, el modelo podría completarse con mayor cantidad de información exterior, como es el análisis en concreto de los principales titulares de prensa que mencionan a Tesla o la cotización de índices bursátiles globales como puede ser el S&P500.

Por último, para poder considerar el modelo como robusto, se considera insuficiente el ámbito temporal analizado. De esta manera, ampliando la ventana temporal del análisis a tres años, se podría realizar un análisis más exhaustivo mediante series temporales, en las que, entre otras cosas, se tuviera en cuenta la propia estacionalidad, y se contara con mayor volumen de registros de entrenamiento.

9.- LÍNEAS DE TRABAJO FUTURAS

A lo largo del desarrollo de este trabajo se han observado diversas maneras con las que completar el mismo, pero que no se han podido implementar debido a que se encontraban fuera del alcance fijado. Las líneas identificadas son:

- Tal y como se ha comentado anteriormente, una de las líneas de trabajo a seguir es no solo tener en cuenta la información recolectada en la red social Twitter, sino también la generada en otros medios digitales, como pueden ser artículos de prensa.
- Adicionalmente, se considera oportuno incorporar información externa a la entidad que refleje las tendencias del sector y mercado en los que opera. Por ejemplo, podría ser de gran utilidad incluir las cotizaciones de otras empresas del sector tecnológico, o incluso llegar a incluir las cotizaciones de índices bursátiles en los que no se encuentre incluido Tesla, dado que intrínsecamente el precio de Tesla estaría afectando al del índice, y estaríamos contaminando las variables de entrenamiento con el target.
- También se cree interesante la implementación del trabajo desarrollado de *topic modeling*, realizando un tratamiento independiente para cada uno de los topics obtenidos en este trabajo.
- Además, sería de gran interés incluir una mayor amplitud temporal en el análisis realizado, pudiendo realizar un análisis de estacionalidad, y obteniendo mayor volumen de registros de los que puedan aprender los modelos implementados.

- Realizar una búsqueda de hiperparámetros más intensiva combinando múltiples arquitecturas de redes neuronales, aplicando series temporales que permitan aprender de los registros de los slots pasados.

BIBLIOGRAFÍA

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python.* O'reilly.

Bollen, J., Mao, H., & Zeng, X.-J. (2010). *Twitter mood predicts the stock market.*

Brownlee, J. (6 de abril de 2016). *Machine Learning Mastery*. Obtenido de https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/

Goldberg, Y., & Levy, O. (2014). *Word2Vec Explained: Deriving Mikolov et al.'s Negative Sampling Word-Embedding Method.*

Harrell Jr, F. (2017). *Regression Modeling Strategies.* Department of Biostatistics, Venderbilt University School of Medicine.

Hastie, T., Tibshirani, R., & Friedman, J. (s.f.). *The Elements of Statistical Learning.* Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An Introduction to Statistical Learning.* Springer.

Li, Y., & Tao, Y. (2017). *Word Embedding for Understanding Natural Language: A Survey.* Springer.

Loper, E., & Bird, S. (2002). *NLTK: The Natural Language Toolkit.*

M. Blei, D., Y. Ng, A., & I. Jordan, M. (2003). *Latent Dirichlet Allocation.* Journal of Machine Learning Research 3.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality.*

Mittal, A., & Goel, A. (s.f.). *Stock Prediction Using Twitter Sentiment Analysis.*

V. Chawla, N., W. Bowyer, K., O. Hall, L., & Kegelmeyer, W. (2002). *SMOTE: Synthetic Minority Over-sampling Technique.* Journal of Artificial Intelligence Research 16.