1. **MÉTODOS DEL ANÁLISIS DE SENTIMIENTO**

Existen actualmente tres métodos disponibles para realizar un análisis de sentimiento de forma automática:

1) Métodos basados en reglas

2) Métodos vasados en *features*

3) Métodos basados en *embedings*

Los métodos basados en reglas son los que implican un conjunto de reglas elaboradas manualmente para identificar la subjetividad, la polaridad o el tema/topic de una opinión. Para ello, se emplean técnicas como el stemming, la tokenización, el etiquetado de partes del discurso, el análisis sintáctico y los léxicos (es decir, listas de palabras y expresiones).

Por lo general, se preparan dos listas de palabras polarizadas (por ejemplo, palabras negativas como malo, peor, feo, etc. y palabras positivas como bueno, mejor, hermoso, etc.). Una vez preparado, se cuenta el número de palabras positivas y negativas que aparecen en un texto determinado. Si el número de palabras positivas es mayor que el número de palabras negativas, el sistema devuelve un sentimiento positivo. Si los números son iguales, el sistema devolverá un sentimiento neutro.

Son ejemplos de estos métodos: AFINN, Bing Liu's lexicon, MPQA subjectivity lexicon, SentiWordNet, TextBlob & VADER. Estos dos últimos serán incluidos en nuestro modelo y explicados con detalle más adelante.

Dado que los métodos basados en reglas suelen ser algo ingenuos (naïve) por naturaleza, ya que no tienen en cuenta cómo se combinan las palabras en una secuencia. Suelen utilizarse otros métodos basados en *features* que se apoyan en técnicas de ML en los que el análisis de sentimientos se modela como un problema de clasificación.

El primer paso en un clasificador de texto de aprendizaje automático es transformar la extracción del texto o la vectorización del mismo, y el enfoque clásico ha sido el de bolsa de palabras o bolsa de ngramas con su frecuencia. Más recientemente, se han aplicado nuevas técnicas de extracción de características basadas en embedings de palabras (también conocidas como vectores de palabras). Este tipo de representaciones permiten que palabras con un significado similar tengan una representación similar, lo que puede mejorar el rendimiento de los clasificadores. IBM Watson, también incluida en nuestro modelo, es un gran ejemplo de estos dos paradigmas. Emplea redes neuronales para la extracción de sentimiento, aunque nos reservamos su análisis para apartados posteriores.

* 1. **VADER**

VADER (Valence Aware Dictionary and Sentiment Reasoner). Es una herramienta de análisis de sentimientos basada en reglas y en un lexicón. Está específicamente orientada y adecuada a los sentimientos expresados en las redes sociales. Este lexicon de sentimientos es una lista de características léxicas (en este caso palabras), etiquetadas según su orientación semántica como positiva o negativa. VADER no sólo nos informa de la puntuación de positividad y negatividad, sino también de lo positivo o negativo que es un sentimiento. Está disponible como Código abierto bajo la licencia MIT.

El algoritmo hace uso de una serie de elementos, que detallamos a continuación: a) Token, b) Puntuaciones medias del sentimiento, c) Desviación estándar, y d) Puntuaciones en bruto del sentimiento realizadas por humanos.

De forma inmediata, se hace uso de los dos primeros elementos (Token y Valencia media). Los dos últimos elementos (Desviación estándar y Valoraciones brutas humanas) se proporcionan para el rigor. Básicamente, para permitir la posibilidad de añadir nuevos tokens al lexicon y, a la vez, mantener la consistencia del modelo.

Vader funciona en base a las calificaciones de sentimiento de diez calificadores humanos independientes. Todos ellos preseleccionados, formados y con un control de calidad posterior, para garantizar una óptima fiabilidad entre calificadores y calificaciones. Más de 9.000 características léxicas fueron calificadas en una escala del "[-4] Extremadamente negativo" a "[4] Extremadamente positivo", con la posibilidad de "[0] Neutral (o Ninguno, N/A)". Se conservaron todos los rasgos léxicos que tenían una calificación media distinta de cero y cuya desviación estándar fuera inferior a 2,5, según la suma de esos diez calificadores independientes. Esto dejó con algo más de 7.500 rasgos léxicos con puntuaciones de valencia validadas que indicaban tanto la polaridad del sentimiento (positivo/negativo), como la intensidad del sentimiento en una escala de -4 a +4. Por ejemplo, la palabra "okay" tiene una valencia positiva de 0,9, "good" es de 1,9 y "great" es de 3,1, mientras que "horrible" es de -2,5, el emoticono del ceño fruncido :( es de -2,2 y "sucks" y su derivado del argot "sux" son ambos de -1,5.

Antes de realizar dichas calificaciones, se comenzó construyendo una lista inspirada en los de bancos de palabras de sentimientos más empleados de forma general y que gozan de mayor confianza por la comunidad investigadora (LIWC, ANEW y GI). A continuación, se incorporaron numerosos rasgos léxicos comunes a la expresión de sentimientos en los microblogs, incluyendo una serie de características que hacen de Vader un lexicon único. Entre ellas destacamos:

- Una lista completa de emoticonos de estilo occidental, por ejemplo, :-) que denota una cara sonriente y generalmente indica un sentimiento positivo. Así como el uso de emojis (codificados en utf-8).

tweet1 = " What a fine day I am having today"

tweet2 = " What a fine day I am having today :-)"

tweet3 = " What a fine day I am having today :-) :-)"

tweet4 = "I love the team and how they played last night"

tweet5 = "I love the team and how they played last night 💘"

tweet6 = "I love the team and how they played last night 😁"

- El uso de alfabetos/palabras en mayúsculas que normalmente indican un aumento de la magnitud del sentimiento. Por ejemplo:

tweet1 = "I like the fact that monsoon is over"

tweet2 = "I LIKE the fact that monsoon is over"

- Acrónimos relacionados con los sentimientos (por ejemplo, LOL y WTF):

tweet1 = "I am laughing like crazy"

tweet2 = "I am laughing like crazy lmao"

tweet3 = "I am laughing like crazy lol"

- Jerga de uso común con valor sentimental (por ejemplo, nah, meh y giggly):

Tweet1 = "He is kinda bored"

tweet2 = "He is friggin bored"

- El uso de signos de puntuación como !, ?, etc. que añaden intensidad al texto.

- Las conjunciones que provocan un cambio de polaridad. La última parte (después de la conjunción) actúa como parte dominante en la definición de la magnitud de la polaridad:

tweet1 = "He is good but his mother is irritating"

tweet2 = "The thai curry was bad, however pasta was delicious"

tweet3 = "The thai curry was ok and pasta was delicious"

- Modificadores de grado. Como su nombre indica, intensifican el grado de forma positiva o negativa según el uso:

tweet1 = "Real Madrid's game play was good last night."

tweet2 = "Real Madrid's game play was extremely good last night."

tweet3 = "Real Madrid's game play was somewhat good last night."

tweet4 = "Real Madrid's game play was terrible last night."

tweet5 = "Real Madrid's game play was awfully terrible last night."

Uso de negaciones:

tweet1 = "He wasn't very good at the play"

tweet2 = "He was not very good at the play"

Finalmente, se confirma empíricamente la aplicabilidad general de cada característica léxica candidata a las expresiones de sentimiento, utilizando un enfoque de sabiduría de la multitud (WotC) según (Surowiecki, 2004) para adquirir una estimación puntual válida de la valencia del sentimiento (polaridad e intensidad) de cada característica candidata libre de contexto.

Por otro lado, el código Python para el motor de análisis de sentimientos basado en reglas implementa las reglas gramaticales y sintácticas descritas, incorporando cuantificaciones derivadas empíricamente para el impacto de cada regla en la intensidad percibida del sentimiento en el texto a nivel de frase. Es importante destacar que esta heurística va más allá de lo que normalmente se capta en un modelo típico de bolsa de palabras.

En cuanto a las puntuaciones. La puntuación compuesta se calcula sumando las puntuaciones de valencia de cada palabra del léxico, ajustadas según las reglas, y luego normalizadas para que estén entre -1 (la más extrema negativa) y +1 (la más extrema positiva). Esta es la métrica más útil si se desea una única medida unidimensional del sentimiento para una frase determinada. De esta forma, el nombre completo de dicha métrica debería ser: "puntuación compuesta normalizada y ponderada".

La puntuación compuesta es la más utilizada para el análisis del sentimiento por la mayoría de los investigadores. También es útil para los estudiosos que quieran establecer umbrales normalizados para clasificar las frases como positivas, neutras o negativas. Los valores de umbral típicos (utilizados en la literatura citada en esta página) son:

1. sentimiento positivo: puntuación compuesta >= 0,05

2. sentimiento neutro: (puntuación compuesta > -0,05) y (puntuación compuesta < 0,05)

3. sentimiento negativo: puntuación compuesta <= -0,05

* 1. **TEXTBLOB**

TEXTBLOB funciona de forma muy similar a VADER, y la mayoría de explicaciones realizadas para esta librería son aplicables a TEXTBLOB. Cuando calculamos el sentimiento de un texto a través de TEXTBLOB, éste nos proporciona valores numéricos de polaridad y subjetividad. El valor numérico de la polaridad describe en qué medida un texto es negativo o positivo. Del mismo modo, la subjetividad describe en qué medida un texto es objetivo o subjetivo.

Existe como en VADER un archivo de léxicos, en el cual no se incluye ningún tipo de stopwords, que no aporte ningún tipo de sentimiento. En el que cada palabra está definida en el archivo con su parte de la oración (POS), su polaridad, subjetividad, intensidad y confianza.

Al calcular el sentimiento, TEXTBLOB utiliza la técnica de "promediación" que se aplica para calcular una puntuación de polaridad para una sola palabra y, por lo tanto, realizará una operación similar para cada palabra y obtendrá una polaridad combinada para el texto objeto de análisis.

Como veíamos en VADER, esta librería también maneja las negaciones y los modificadores, también conocido como intensificadores.

* 1. **WATSON IBM**

Muchos sistemas de lenguaje natural, como VADER o TEXTBLOB, han intentado hacer hincapié en la precisión dentro de los límites de reglas específicas bien formadas. Por ejemplo, el análisis de sentimientos suele buscar un conjunto de palabras específicas y sus sinónimos en un sitio de medios sociales. A continuación, estos sistemas, sin evaluar el contexto en el que se utilizan esas palabras, contabilizan el número de veces que esas palabras se encuentran junto a alguna marca en la misma frase. Por ejemplo, se toma la frase "... pasé por la tienda de donuts DonutsManolo a tomar un café esta mañana, fue genial..." y se afirma que la colocación del nombre de la marca y el término "genial" son una indicación de un sentimiento positivo. Sin embargo, esto no suele ser cierto si el resto de la frase es: "..., ha sido genial saber que pronto se abrirá una nueva cafetería al lado, así no tendré la tentación de comer donuts todas las mañanas". Llamamos a estos métodos basados en reglas como procesamiento superficial (shallow) del lenguaje natural (PLN) porque, aunque puede ser bastante preciso dentro de su enfoque más estrecho, no es muy exacto.

Sin embargo, también es importante darse cuenta de que el NLP superficial tiene un papel importante en muchos sistemas. Si su intención es crear una evaluación estadísticamente relevante de las tendencias de sentimiento sobre enormes cantidades de información, la falta de precisión para cada ejemplo individual probablemente no sea un problema. Asumiendo que hay aproximadamente tantos falsos positivos como falsos negativos en un conjunto de muestras suficientemente grande, se anulan mutuamente. Y si el conjunto de cuentas canceladas se mantiene relativamente constante entre los conjuntos de muestras a lo largo del tiempo, los datos restantes no cancelados arrojan información de tendencias estadísticamente relevantes. Por lo tanto, los costes de procesamiento adicionales que se requieren para la precisión adicional de cualquier instancia podrían no estar justificados.

Sin embargo, cuando las instancias individuales importan, los sistemas que se diseñan para ser precisos sin centrarse en altos niveles de exactitud tienden a ser frágiles.

El símil fácil es la comparación de estos sistemas con el uso de técnicas de construcción con ladrillos. Los ladrillos son fuertes y bastante fáciles de construir. Durante décadas y siglos, hemos perfeccionado la técnica de construcción con ladrillos para que sea bastante precisa. Pudimos construir estructuras relativamente grandes, ornamentadas y duraderas. Sin embargo, aunque los edificios de ladrillo tienen una gran resistencia a la carga, tienen poca resistencia a la tracción. Se caen fácilmente en caso de terremoto. Y a partir de cierto punto, su resistencia a la carga también falla.

Estas mismas limitaciones se pueden observar en algunos productos de consumo actuales. Por ejemplo, puedes utilizar tu asistente personal favorito activado por voz y decir: "Búscame una pizza". A cambio, obtienes un listado local de pizzerías, que es exactamente lo que querías. Ahora dices: "No me busques pizza". Sigue recibiendo un listado local de pizzerías, que no es exactamente lo que pidió. Del mismo modo, si dices "Encuéntrame pizza cerca" o "Encuéntrame pizza lejos", te devuelven los mismos listados locales. La cuestión es que estos sistemas están diseñados según un conjunto específico de reglas y buscan combinaciones específicas de palabras clave para determinar la respuesta a producir. Estos sistemas no saben distinguir entre cosas para las que no existe ninguna regla. Pueden ser precisos, pero no necesariamente muy exactos.

Para superar las limitaciones de la construcción de ladrillos, pasamos a utilizar acero y hormigón armado para los edificios más grandes. Siguiendo con esta comparativa, estamos asistiendo a un cambio en las técnicas de construcción del procesamiento del lenguaje natural cuando se necesita más exactitud que precisión. Estas técnicas incorporan mucho más contexto en la evaluación de la pregunta. Nos referimos a este concepto como procesamiento profundo del lenguaje natural, que a veces se denomina Deep Question-Answering (DeepQA).

IBM Watson es un sistema de NLP profundo. Alcanza la precisión tratando de evaluar el mayor contexto posible. Obtiene ese contexto tanto del pasaje de la pregunta como de su base de conocimientos (corpus) de la que dispone para encontrar respuestas.

Una manejo eficaz de la información no estructurada requiere unos nuevos métodos que llamamos sistemas cognitivos. IBM Watson es un ejemplo de estos sistemas cognitivos. Es capaz de desentrañar el lenguaje humano para identificar inferencias entre pasajes de texto con una precisión similar a la humana, y a una velocidad y escala mucho más rápidas y grandes de lo que cualquier persona puede hacer por sí misma. Puede lograr un alto nivel de precisión cuando se trata de entender la respuesta correcta a una pregunta.

Sin embargo, Watson no entiende realmente las palabras individuales del lenguaje. Más bien entiende las características del lenguaje que utilizan las personas. A partir de esas características, puede determinar si un pasaje de texto (que llamamos pregunta) infiere otro pasaje de texto (que llamamos respuesta), con un alto nivel de precisión en circunstancias cambiantes.

El contexto es importante. Las limitaciones temporales y espaciales son importantes. Todos estos conceptos contribuyen a que un sistema cognitivo se comporte con características similares a las del ser humano. Y, volviendo a un punto anterior, un enfoque basado en reglas podría necesitar un número casi infinito de reglas para captar todos los casos que podríamos encontrar en el lenguaje.

Watson separa la pregunta y las posibles respuestas en el corpus, y luego la examina junto con el contexto del enunciado de cientos de maneras. A continuación, Watson utiliza los resultados para obtener un grado de confianza en su interpretación de la pregunta y las posibles respuestas. Pero debemos retroceder un poco. ¿Cómo obtiene Watson sus respuestas a las preguntas? El proceso es el siguiente:

1. Cuando se le presenta una pregunta a Watson, la analiza para extraer las principales características de la misma.

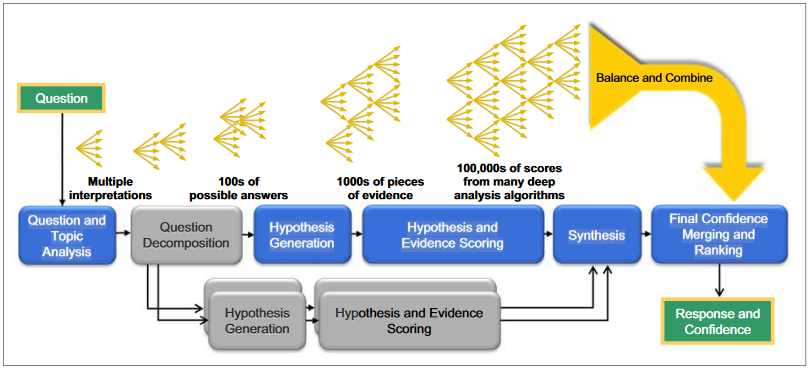
2. Genera un conjunto de hipótesis buscando en el corpus pasajes que tengan algún potencial para contener una respuesta valiosa.

3. Realiza una comparación profunda entre el lenguaje de la pregunta y el lenguaje de cada respuesta potencial utilizando varios algoritmos de razonamiento. Hay cientos de algoritmos de razonamiento, cada uno de los cuales realiza una comparación diferente. Por ejemplo, algunos se fijan en la coincidencia de términos y sinónimos, otros en las características temporales y espaciales, y otros en las fuentes relevantes de información contextual.

4. Cada algoritmo de razonamiento produce una o más puntuaciones, indicando el grado en que la respuesta potencial es inferida por la pregunta basada en el área específica de enfoque de ese algoritmo.

5. Cada puntuación resultante se pondera con respecto a un modelo estadístico que capta lo bien que lo hizo ese algoritmo a la hora de establecer las inferencias entre dos pasajes similares para ese dominio durante el "periodo de entrenamiento" de Watson. Ese modelo estadístico puede usarse entonces para resumir un nivel de confianza que Watson tiene sobre la evidencia de que la respuesta candidata es inferida por la pregunta.

6. Watson repite este proceso para cada una de las respuestas candidatas hasta que pueda encontrar respuestas que aparezcan como candidatos más fuertes que los otros.



De esta forma, en el funcionamiento de Watson, es de suma importancia un corpus de conocimiento. Este corpus se compone de todo tipo de conocimientos no estructurados, como libros de texto, directrices, manuales de instrucciones, preguntas frecuentes, planes de beneficios y noticias. Watson ingiere el corpus, revisando todo el cuerpo de contenido para ponerlo en una forma que sea más fácil de trabajar. El proceso de ingesta también limpia el contenido. Es decir, se centra en si el corpus contiene el contenido adecuado, eliminando los artículos o páginas que no están actualizados, que son irrelevantes o que proceden de fuentes potencialmente poco fiables. Cuando decimos "búscame una pizza", se da por sentado que nos referimos a algo cercano. Pero lo que está cerca es siempre relativo. En otros casos, las relaciones espaciales aparecen en relación con marcadores geográficos, por ejemplo, un barrio en una ciudad o un estado en un país. Asimismo, los rasgos temporales también están presentes en el contexto de gran parte de lo que escribimos. Cuando decimos: "Coge el queso de la tienda de camino a casa", se infiere un marco temporal. Es de suponer que el escritor y el destinatario tienen un entendimiento contextual compartido de cuándo estarán de camino a casa.

El contexto se deriva tanto de la información inmediata como del conocimiento disponible de forma más amplia. Watson puede obtener información inmediata del título de un documento, de otros pasajes del mismo o de la base de datos de la que procede.

El contexto también puede provenir de forma más amplia de una historia compartida. El sistema recuerda cuando ocurrieron ciertas cosas en relación con las demás. Sabemos que Facebook se creó hace bastante tiempo, pero sabemos que Abraham Lincoln vivió hace unos 150 años, mucho antes de que Facebook se hiciera popular. Por lo que si Lincoln se refiere a una “solicitud de amistad” la respuesta no se encontrará en mensajes de Facebook. El contexto y el razonamiento nos ayudan a crear una base cognitiva para procesar el lenguaje.

**1.4 COMPARATIVAS**

TEXTBLOB VS VADER <https://www.analyticsvidhya.com/blog/2021/01/sentiment-analysis-vader-or-textblob/>

Standford e IBM <https://www.zerone-consulting.com/natural-language-processing-with-ibm-watson-and-stanford-nlp/>

Google NL <https://fredrikstenbeck.com/google-natural-language-vs-watson-natural-language-understanding/>

Hay opciones de extraer “emotion” en ibm, también hay “emotion” en NRCLex

**1.5 BIBLIOGRAFÍA**

<https://github.com/cjhutto/vaderSentiment>

<https://pypi.org/project/vaderSentiment/>

<https://www.nltk.org/_modules/nltk/sentiment/vader.html>

<https://www.kaggle.com/datatattle/lexicon-sentiment-analysis-unsupervised>

<https://textblob.readthedocs.io/en/dev/>

<https://medium.com/red-buffer/sentiment-analysis-let-textblob-do-all-the-work-9927d803d137>

<https://www.ibm.com/cloud/watson-natural-language-understanding>

<http://www.redbooks.ibm.com/redpapers/pdfs/redp4955.pdf>