U3 :: Procesamiento del Lenguaje Natural

0. Introducción a la PLN

Piensa en las tecnologías que nos rodean:

- Corrección ortográfica y autocorrección
- Subtítulos de video generados automáticamente
- Asistentes virtuales como Alexa de Amazon
- Autocompletar
- Artículos sugeridos de su sitio de noticias

¿Qué tienen en común?

¡Todas estas tecnologías existen gracias al ***procesamiento del lenguaje natural!*** También conocido como ***PLN***, es un campo que se encuentra en la intersección entre la lingüística, la inteligencia artificial y la informática. ¿Cuál es su objetivo? Permitir que las computadoras interpreten, analicen y se aproximen a la generación de lenguajes humanos (como el inglés o el español).

La PLN comenzó alrededor de 1950 con la prueba de inteligencia artificial de Alan Turing que evaluaba si una computadora podía usar el lenguaje para engañar a los humanos haciéndoles creer que era humana.

La aproximación al habla humana es sólo una de entre las muchas aplicaciones que el PLN permite. Ejemplos como la detección de correos electrónicos no deseados o los sesgos que existen en los tweets hasta la mejora de la accesibilidad para las personas con discapacidades dependen en gran medida de técnicas de procesamiento del lenguaje natural.

El PLN se puede realizar en varios lenguajes de programación. Sin embargo, Python tiene algunas de las bibliotecas de PLN de código abierto más completas, incluido Natural Language Toolkit o ***NLTK***. Por esta razón, usaremos Python para tener un primer contacto con el PLN.

> Pre-procesamiento de texto

Limpiar y preparar el texto son cruciales para muchas tareas, y el PLN no es una excepción. ***El preprocesamiento de texto*** suele ser el primer paso en el PLN. Sin preprocesamiento, su computadora interpreta `"el"`, `"El"`y `"<p> El"` como palabras completamente diferentes.

NLTK simplifica esta tarea, permitiendo:

- **Eliminación de ruido ("Noise removal"):** eliminar el formato del texto (por ejemplo, etiquetas HTML).
- **Tokenización** : división del texto en palabras individuales.
- **Normalización** : limpieza de datos de texto de cualquier otra manera:
    - **Derivación ("Stemming"):** actúa identificando prefijos y sufijos de palabras. "Gritando" y "gritar" se convierten en "grito", pero "computadora" puede convertirse en "computa" y "dora", lo cual tiene menos sentido ...
    - **Lematización ("Lemmatization"):** permite simplificar términos a su forma raíz. Por ejemplo, el lematizador de NLTK sabe que "soy" y "son" están relacionados con "ser".
    - Otras tareas comunes incluyen minúsculas, eliminación de palabras vacías ("**Stopwords**"), corrección ortográfica, etc.

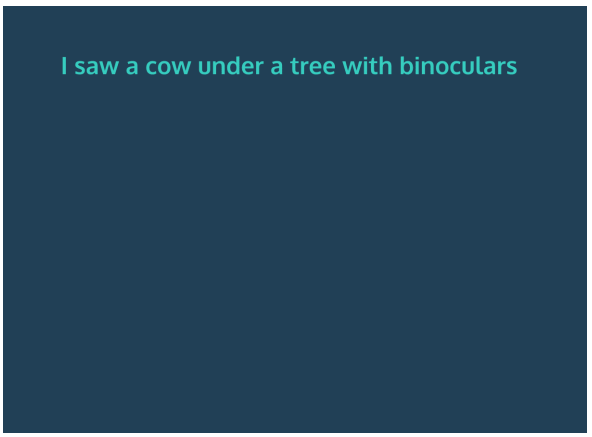
> Análisis ("parsing") de texto

Una vez se tiene una lista de palabras limpia y preprocesada. ¿Ahora que? Puede resultar útil saber cómo se relacionan las palabras entre sí y la sintaxis subyacente (gramática). ***El análisis*** es un proceso de PLN relacionado con la segmentación de texto según su sintaxis.

Probablemente no desee realizar ningún análisis a mano y NLTK tiene algunos trucos bajo la manga para ayudarlo:

- ***Etiquetado de parte del discurso ("Part-of-speech tagging")*** identifica partes del discurso (verbos, sustantivos, adjetivos, etc.).
- ***Reconocimiento de entidades nombradas ("Named entity recognition")*** ayuda a identificar los nombres propios (por ejemplo, "Natalia" o "Berlín") en un texto. Esto puede ser una pista sobre el tema del texto y NLTK los identifica.

- ***Árboles de dependencia gramatical:*** ayudan a comprender la relación entre las palabras de una oración. En cualquier idioma se maneja mucha ambigüedad, por lo que la sintaxis puede ser difícil, incluso para un programa de computadora. Echa un vistazo a la siguiente oración:



¿Tengo los prismáticos? ¿Tiene la vaca binoculares? ¿El árbol tiene binoculares?

- ***El análisis de expresiones regulares***, otorga flexibilidad y potencia en el reconocimiento de patrones (por ejemplo, utilizando la biblioteca de Python re). Cuando se combina con el etiquetado POS, permite identificar fragmentos de frases específicos. Resulta útil encontrar direcciones, correos electrónicos y muchos otros patrones comunes dentro de grandes trozos de texto.

Modelos de lenguaje

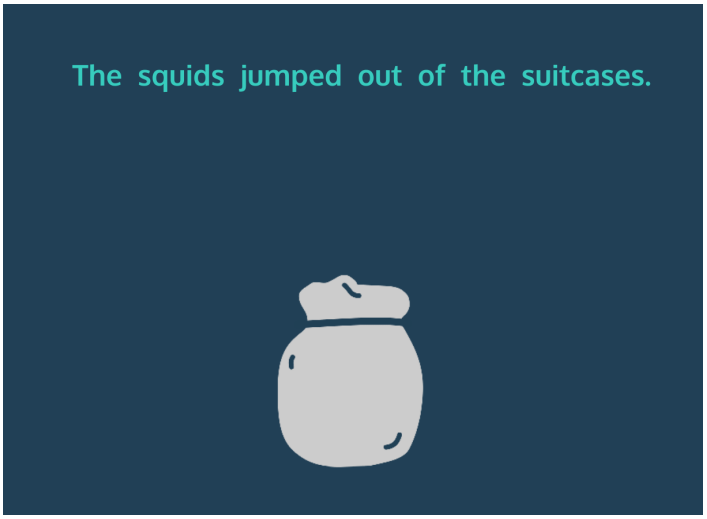
Bolsa de palabras ("Bag-of-words")

¿Cómo podemos ayudar a una máquina a dar sentido a un montón de símbolos de palabras? Podemos ayudar a las computadoras a hacer predicciones sobre el lenguaje entrenando un modelo de lenguaje en un *corpus* (un montón de textos de ejemplo).

Los modelos lingüísticos son modelos probabilísticos del lenguaje. Construimos y usamos estos modelos para calcular la probabilidad de que se use un sonido, letra, palabra o frase determinados. Una vez que se ha entrenado un modelo, se puede probar en nuevos textos.

Uno de los modelos de lenguaje más comunes es el modelo monograma ("unigram"), un modelo de lenguaje estadístico comúnmente conocido como ***"bolsa de palabras"***. Como sugiere su nombre, una bolsa de palabras no contiene mucho orden en su interior. Lo que sí tiene es un recuento de cada instancia para cada palabra.

Considera el siguiente ejemplo de texto:



Si se proporciona un procesamiento inicial, la bolsa de palabras daría como resultado una asignación como:

```
{"the": 2, "squid": 1, "jump": 1, "out": 1, "of ": 1, " maleta ": 1}
```

Ahora mira esta oración mapeada:" ¿Por qué tus maletas están llenas de calamares saltarines? "

```
{"why": 1, "be": 1, "your": 1, "suitcase": 1, "full": 1, "of": 1, "jump": 1, "squid": 1}
```

Puedes ver cómo incluso con diferentes órdenes de palabras y estructuras de oraciones, "saltar", "calamar" y "maleta" son temas compartidos entre los dos ejemplos. Las "bolsas de palabras" pueden ser una excelente manera de ver el lenguaje cuando se desea hacer predicciones sobre el tema o el sentimiento de un texto. Cuando la gramática y el orden de las palabras son irrelevantes, probablemente usar este modelo sea una buena elección.

N-Gramas y NLM

Para analizar frases enteras o realizar predicciones es mejor utilizar un modelo que preste atención a los términos que acompañan a cada palabra. A diferencia de la "bolsa de palabras", el modelo ***n-grama*** considera una secuencia de *n* unidades y calcula la probabilidad de cada unidad en un *corpus* dada la secuencia anterior de longitud *n*. Debido a esto, las *n* probabilidades de gramas con valores *n* más grandes pueden ser útiles en tareas de predicción.

Echa un vistazo a nuestro ejemplo revisado de calamar: "Los calamares saltaron de las maletas. Los calamares estaban furiosos ".

Un modelo de **bigrama** (donde *n* es 2) podría darnos las siguientes frecuencias de conteo:

```
{('', 'el'): 2, ('el', 'calamares'): 2, ('calamares', 'saltado') : 1, ('saltado', 'fuera'): 1, ('fuera', 'de'): 1, ('de', 'el'): 1, ('el', 'maletas'): 1, ('maletas', ''): 1, ('calamares', 'eran'): 1, ('estaban', 'furiosos'): 1, ('furiosos', ''): 1}
```

Hay un par de problemas con el modelo n-grama:

1. ¿Cómo puede su modelo de lenguaje darle sentido a la oración "El gato se durmió en el buzón" si nunca antes había visto la palabra "buzón"? Durante el entrenamiento, su modelo probablemente encontrará palabras que nunca antes había encontrado. Una táctica conocida como *suavizado del lenguaje* puede ayudar a ajustar las probabilidades de palabras desconocidas, pero no siempre es ideal.
2. Para un modelo que predice con mayor precisión los patrones del lenguaje humano, deseas que *n* (la longitud de su secuencia) sea lo más grande posible. De esa manera, tendrás un lenguaje que suena más natural, ¿verdad? Bueno, a medida que aumenta la longitud de la secuencia, se reduce el número de ejemplos de cada secuencia dentro de su corpus de entrenamiento. Con muy pocos ejemplos, no habrá suficientes datos para hacer muchas predicciones.

Gran parte del trabajo reciente dentro de la PLN ha involucrado el desarrollo y entrenamiento de redes neuronales para aproximar el enfoque que adoptan nuestros cerebros humanos hacia el lenguaje. Este enfoque de aprendizaje profundo ofrece a las computadoras una estrategia mucho más adaptativa para procesar el lenguaje humano. Los ***modelos de lenguaje neuronal (NLM*** comunes incluyen LSTM y modelos de transformador.

Modelos temáticos ("topic modeling")

Hemos abordado la idea de encontrar temas dentro de un cuerpo de lenguaje. Pero, ¿qué pasa si el texto es largo y los temas no son obvios?

El modelado temático (***"Topic modeling")*** es un área de la PLN dedicada a descubrir temas latentes u ocultos dentro de un cuerpo de lenguaje. Por ejemplo, se podría usar este enfoque para descubrir patrones dentro de las canciones de Taylor Swift relacionados con el amor y la angustia a lo largo del tiempo.

Una técnica común es despriorizar las palabras más comunes y priorizar los términos usados con menos frecuencia como temas en un proceso conocido como ***frecuencia de términos - frecuencia inversa en documentos (tf-idf)***. Cuando se trabaja con mucho texto tiene sentido descartar palabras como "el" y "es"

detrás de las cuales es díficil intuir temas. Las bibliotecas de Python `gensim` y `sklearn` tienen módulos para manejar tf-idf.

Al margen de que se use la "bolsa de palabras" (que da la frecuencia de los términos) o tf-idf, el siguiente paso en el modelado de temas suele ser ***la asignación de Dirichlet latente (LDA)***. LDA es un modelo estadístico que a partir de un corpus de documentos determina qué palabras siguen apareciendo juntas en los mismos contextos.

Al margen de esto, ***word2vec*** es otra gran técnica disponible. word2vec puede mapear los resultados de cierto "modelo temático" espacialmente (i.e. como vectores) para que las palabras usadas de manera similar estén más juntas. En el caso de una muestra de lenguaje consistente en "Los calamares saltaron de las maletas. Los calamares estaban furiosos. ¿Por qué tus maletas están llenas de calamares saltarines? ". Podríamos ver que " maleta "," saltar " y " calamar " eran palabras utilizadas en contextos similares. Este mapeo de palabra a vector se conoce como *incrustación de palabra*.

Similitud textual

La mayoría de nosotros sabemos que es la autocorrección. La aplicación de mensajería de nuestro teléfono cambia silenciosamente una letra por otra mientras escribimos y, de repente, el significado de nuestro mensaje ha cambiado (para nuestro horror o placer). Sin embargo, abordar la ***similitud textual***, incluida la corrección ortográfica, es un desafío importante dentro del procesamiento del lenguaje natural.

Abordar la similitud de palabras y la falta de ortografía para la corrección ortográfica o la autocorrección a menudo implica considerar la ***distancia de Levenshtein*** o distancia de edición mínima entre dos palabras. La distancia se calcula a través del número mínimo de inserciones, eliminaciones y sustituciones que tendrían que ocurrir para que una palabra se convierta en otra. Por ejemplo, convertir "casa" en "carpa" requeriría una sustitución ("s" por "p") y una inserción ("r"), por lo que la distancia de Levenshtein sería dos.

La similitud fonética también es un desafío importante dentro del reconocimiento de voz. Los humanos de habla inglesa pueden saber fácilmente por el contexto si alguien dijo "euthanasia" o "youth in Asia", ¡pero es una tarea mucho más desafiante para una máquina! La tecnología más avanzada de autocorrección y corrección ortográfica considera además la distancia de las teclas en un teclado y ***la similitud fonética*** (cuánto suenan de iguales dos palabras o frases).

También es útil averiguar si los textos son iguales para protegernos contra el plagio, que podemos identificar a través de ***la similitud léxica*** (el grado en que los textos usan el mismo vocabulario y frases). Mientras tanto, ***la similitud semántica*** (el grado en que los documentos contienen temas o significados similares) es útil cuando desea encontrar (o recomendar) un artículo o libro similar a uno que terminó recientemente.

Predicción de idiomas y generación de texto

¿Cómo completa su motor de búsqueda favorito sus consultas de búsqueda? ¿Cómo sabe el teclado de su teléfono lo que desea escribir a continuación? ***La predicción del lenguaje*** es una aplicación de la PLN que se ocupa de predecir el texto dado el texto anterior. La autosugestión, la autocompletar y las respuestas sugeridas son formas comunes de predicción del lenguaje.

Su primer paso para la predicción del lenguaje es elegir un modelo de lenguaje. La bolsa de palabras por sí sola generalmente no es un gran modelo para la predicción del lenguaje; no importa cuál sea la palabra anterior, solo obtendrá una de las palabras más utilizadas de su corpus de entrenamiento.

Si sigues ruta de *n-gramas*, lo más probable es que uses ***las cadenas de Markov*** para predecir la probabilidad estadística de cada palabra (o carácter) siguiente según el corpus de entrenamiento. Las cadenas de Markov no tienen memoria y hacen predicciones estadísticas basadas completamente en el *n*-grama disponible.

Por ejemplo, tomemos una oración que comienza, "Comí muchos quesos a la plancha". Con un modelo de trigrama (donde *n* es 3), una cadena de Markov predeciría la siguiente palabra como "sándwiches" en función del número de veces que la secuencia "sándwiches de queso a la plancha" ha aparecido en los datos de entrenamiento de todas las veces "queso a la parrilla "Ha aparecido en los datos de entrenamiento.

Un enfoque más avanzado, que utiliza un modelo de lenguaje neuronal, es el modelo de ***memoria a corto plazo a largo plazo (LSTM)*** . LSTM utiliza el aprendizaje profundo con una red de "células" artificiales que administran la memoria, lo que las hace más adecuadas para la predicción de texto que las redes neuronales tradicionales.

Temas avanzados de PLN

Además de lo dicho, existe una gran cantidad de temas y aplicaciones avanzados de PLN, muchos de los cuales se basan en el aprendizaje profundo y las redes neuronales.

- ***Los clasificadores Naive Bayes*** son algoritmos de aprendizaje automático supervisados que aprovechan un teorema probabilístico para hacer predicciones y clasificaciones. Se utilizan ampliamente para el análisis de sentimientos (determinar si un determinado bloque de lenguaje expresa sentimientos negativos o positivos) y el filtrado de spam.
- Hemos logrado enormes avances en ***la traducción automática***, pero incluso el software de traducción más avanzado que utiliza redes neuronales y LSTM todavía tiene mucho que hacer para traducir con precisión entre idiomas.

- Algunas de las aplicaciones de PLN que más alteran la vida se centran en mejorar la ***accesibilidad del idioma*** para las personas con discapacidades. La funcionalidad de conversión de texto a voz y el reconocimiento de voz han mejorado rápidamente gracias a los modelos de lenguaje neuronal, lo que hace que los espacios digitales sean lugares mucho más accesibles.
- El PLN también se puede utilizar para detectar sesgos en la escritura y el habla. ¿Has notado que el discurso de un candidato político, un libro o una fuente de noticias está sesgado pero no sabes exactamente cómo? El procesamiento del lenguaje natural puede ayudar a identificar el idioma en cuestión.

Revisión

Revisemos lo expuesto en torno al procesamiento del lenguaje natural!

- El procesamiento del lenguaje natural combina la informática, la lingüística y la inteligencia artificial para permitir que las computadoras procesen los lenguajes humanos.
- NLTK es una biblioteca de Python utilizada para PLN.
- **El preprocesamiento de texto** es una etapa de la PLN centrada en limpiar y preparar el texto para otras tareas de PLN.
- **El análisis** es una técnica de PLN que se ocupa de dividir el texto según la sintaxis.
- **Los modelos de lenguaje** son modelos de máquina probabilísticos del uso del lenguaje para tareas de comprensión de PLN. Modelos comunes incluyen **"bolsa de palabras", modelos de n-grama** y los **modelos neuronales del lenguaje**.
- **El modelado de temas** es el proceso de PLN mediante el cual se identifican temas ocultos dado un cuerpo de texto.
- **La similitud de texto** es una faceta de la PLN relacionada con la semejanza entre instancias de lenguaje.
- **La predicción del lenguaje** es una aplicación de la PLN que se ocupa de predecir el lenguaje dado el lenguaje anterior.
- Hay muchas **consideraciones sociales y éticas** a tener en cuenta al diseñar herramientas de PLN.