**PROYECTO IDI 2024: Técnicas de NLP E IA aplicadas a escritos judiciales**

***Parte 1: TEXT MINING:***

Las empresas modernas disponen de mucha información sobre sus clientes o su sector de actividad. Las nuevas tecnologías como las redes sociales, el comercio electrónico, las aplicaciones móviles, entre otras, dan acceso a un **gran volumen de información**. Al analizar esos datos, es posible **obtener conocimiento** para después explotarlo de diferentes formas. Sin embargo, algunos **tipos de datos** son más difíciles de explotar que otros.

Como en el análisis de escritos judiciales se trabajar con textos la información que se obtiene de estos escritos no tiene formato alguno, es decir, que los datos obtenidos son **no estructurados.** Estos datos no pueden ser tratados correctamente por softwares o herramientas de análisis de datos tradicionales. Por lo que necesitaremos buscar otros métodos para poder tratar los textos y obtener el conocimiento que se necesita. Para esto vamos a recurrir al text mining.

***Leguaje natural:*** es el lenguaje en el que hablamos todos los días, es nuestra forma de comunicarnos por excelencia. Para nosotros es texto sin estructura, lo que puede llevar a ciertos problemas como:

* *Complejidad:* el lenguaje natural es complejo y flexible, permitiendo la expresión de una amplia gama de conceptos y significados.
* *Ambigüedad:* las palabras y frases pueden tener múltiples significados depende del contexto.

***Text Mining (análisis de datos):*** consiste en transformar un texto no estructurado en datos estructurados para proceder posteriormente al análisis.

Este trabajo se hace utilizando la tecnología NLP (Natural Language Processing) que consiste en modelos de inteligencia artificial que son capaces de comprender y tratar textos en leguaje natural (no estructurado) de manera automática.

Para empezar a trabajar con text mining y NLP necesitamos:

* *Python.*
* *Spacy:*es una biblioteca de procesamiento del lenguaje natural (NLP, por sus siglas en inglés) de código abierto desarrollada en Python. Se utiliza para realizar tareas como tokenización, lematización, reconocimiento de entidades nombradas, análisis de dependencias y más.

***POS tagging (Part-of-Speech tagging):***asigna a cada palabra de un texto una etiqueta gramatical que indica su función gramatical específica en una oración. Estas etiquetas gramaticales representan las categorías lingüísticas a las que pertenecen las palabras, ya sea sustantivos, verbos, adjetivos, adverbios, pronombres, preposiciones, conjunciones, entre otro.

Este proceso nos sirve para comprender la estructura gramatical de un texto, facilitando el análisis y el procesamiento automáticos del lenguaje natural.

A partir de ahora vamos a llamar a cada palabra **token**. Y a cada texto, ya sea una oración o un texto entero como **documento**.

Esto permite que a partir del uso de las etiquetas podamos obtener información. Por ejemplo, siguiendo el patrón especifico donde encuentre un DET + NOUN (Se puede extender a otro tipo de patrones y a otro tipo de “etiquetas”). Esto lo puede lograr a través del **Matcher** de spaCy.

**Matcher:**  es una herramienta que permite realizar coincidencias de patrones en documentos procesados. Se suele usar para encontrar secuencias específicas de tokens en un texto basándose en reglas definidas por patrones.

**Parser de dependencias:** se encarga de analizar la estructura gramatical de una oración y construir un **árbol de dependencias** que representa las relaciones sintácticas entre palabras.

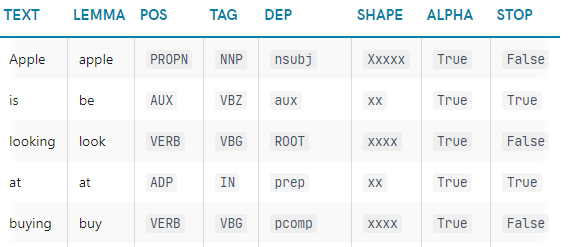
Asigna etiquetas a cada token en función de su función gramatical en la oración y establece las relaciones de dependencia entre ellos. Por ejemplo, algunas de las etiquetas comunes de dependencias incluyen nsubj (sujeto

nominal), ROOT (raíz), obj (objeto), amod (modificador adjetival).

En el contexto del Procesamiento del Lenguaje Natural (NLP), una **entidad** se refiere a un objeto o concepto que tiene una existencia real o abstracta y que puede ser claramente identificado en el texto. Las entidades pueden ser nombres de personas, organizaciones, ubicaciones, fechas, cantidades, productos, términos médicos y cualquier otro tipo de elemento concreto o abstracto que sea relevante para la tarea en cuestión.

Los operadores en los patrones del Matcher son modificadores que permiten especificar la cantidad de veces que un token debe aparecer en el texto. Estos operadores añaden flexibilidad a los patrones y permiten hacer coincidencias más generales o específicas. Alguno de ellos:

* \* (Asterisco): Indica que el token anterior puede aparecer 0 o más veces. Es decir, el token es opcional.
* + (Más): Indica que el token anterior debe aparecer 1 o más veces.
* ? (Interrogación): Indica que el token anterior puede aparecer 0 o 1 vez. Es decir,el token es opcional.
* ! (Exclamación): Niega el token anterior. Indica que el token no debe aparecer.



* Text: The original word text.
* Lemma: The base form of the word.
* POS: The simple [UPOS](https://universaldependencies.org/u/pos/) part-of-speech tag.
* Tag: The detailed part-of-speech tag.
* Dep: Syntactic dependency, i.e. the relation between tokens.
* Shape: The word shape – capitalization, punctuation, digits.
* is alpha: Is the token an alpha character?
* is stop: Is the token part of a stop list, i.e. the most common words of the language?

***Preprocesamiento:*** los datos recopilados a través de las diferentes fuentes (encuestas, aplicaciones, dispositivos, etc.) pueden contener errores, inconsistencias y otros problemas. Particularmente, cuando tratamos con textos escritos por un humano podemos encontrarnos errores de ortografía, abreviaciones, emojis o palabras que no agregan información y que solo sirven de conectores, que pueden **generar ruido y sesgo** al momento de utilizar el texto para una aplicación particular. Esto puede **afectar el rendimiento** de modelos.

*Problemas de no preprocesar:*

* *Ruido:* datos con errores o fluctuaciones no deseadas que pueden deberse problemas de medición, entradas incorrectas o interferencia.
* *Datos faltantes:* falta de información en algunas observaciones o características, lo que puede conducir a una pérdida significativa de información.
* *Inconsistencias:* inconsistencias entre observaciones o características, lo que puede resultar en contradicciones en los datos.
* *Datos duplicados:* múltiples copias de la misma información, que pueden introducir sesgos y errores en el análisis.

Es decir, este proceso ayuda a **eliminar estos errores, corregir inconsistencias y mejorar la calidad** de los datos de entrada. Esto a su vez, mejora el rendimiento del sistema de IA al proporcionarle datos limpios y estructurados.

*A mayor cantidad de datos tendremos mejores resultados* (Específicamente entrenando modelos de IA).

El preprocesamiento **depende del uso que se les de a los datos,** este se debe hacer en base al trabajo que se quiere hacer con el texto en cuestión. Cualquier modificación que se realice a los datos tiene un gran impacto en el resultado final del procesamiento o tarea a la cual está dirigida, ya que al modificar surge el riesgo a la introducción de errores.

***Normalización:*** es una de las primeras tareas que se realizan al texto. Tareas que realiza:

* Se reescribe el documento por completo en minúscula.
* Se reemplazan las abreviaturas por sus palabras completas.
* Se eliminan caracteres especiales.
* Se revisa el documento completo por un spellcheker.

Estas tareas deben realizar de forma cuidadosa ya que pueden alterar fácilmente el documente.

Este proceso se puede saltear en diferentes ocasiones, por ejemplo si queremos entrenar una IA para detectar si un texto pertenece a una persona X y otro a una persona Y por la forma de escribir. Dependiendo de la aplicación del texto.

***StopWords:*** son palabras muy comunes que no tienen un significado o que tienen poco significado comparado con otras palabras (pueden tener o no semántica pero no aportan información). En general sirven para dar coherencia y naturalidad en el texto. Ej: y, como, cuando.

El sesgo de estas tiene un impacto importante al momento de comparar documentos, por lo que si necesitamos que un modelo identifique que un documento pertenece a una determinada clasificación podrían influir bastante e incluso hasta confundirlo.

***Stemming/Lemmatizing:*** una palabra puede tener varias formas (conjugaciones) en un documento (ej: jugando, jugar, jugó) lo que nos lleva a buscar alguna forma de representarla a una forma “estándar” (obtener la palabra raíz).

Técnicas con diferentes enfoques para obtener la palabra raíz:

* *Stemming:* se obtiene la raíz a partir de reglas o heurísticas que sacan prefijos y sufijos. Estos varían según el lenguaje empleado ya que las reglas son diferentes de un lenguaje a otro. El resultado que devuelve se denomina **stem**, y pueden tener varias desventajas como generar una representación univoca o generar raíces erróneas. Esta técnica es bastante sencilla e implica eliminar caracteres de una palabra siguiendo las reglas o heurísticas que tiene cada lenguaje.
* *Lemmatizing:* para obtener la raíz utiliza un diccionario y devuelve palabras válidas sin conjugar. El resultado que devuelve se denomina **lemma.** Esta técnica es muy compleja, ya que utiliza un montón de tablas y look-ups para poder resolver el lemma.

Uso de cada:

* Si tenemos un gran volumen de información a procesar, nos conviene usar la técnica mas liviana. Como Stemming lo que hace es sacar algunos caracteres y por ende el costo de procesamiento es mas bajo.
* Si necesitamos tener más precisión en los resultados, nos conviene usar más la técnica de Lemmatization.

En esta primera parte pudimos ver varias formas de procesar un texto, desde la vista semántica y gramatical hasta la sintáctica. Todas las tareas que vimos también se pueden combinar para obtener mejores resultados. Pero como siempre se dijo en esta parte, la aplicación de las diferentes técnicas **depende** de la aplicación a la que este destinado el documento. También vimos ventajas y desventajas de las técnicas en cuestión y las repercusiones que pueden tener en el resultado final.