Clasificación automática

Cálamo Educación S.L.
María Teresa Tijeras Pascual
Cálamo Educación S.L.
María Teresa Tijeras Pascual

Indice

Clasificación automática

El objetivo de esta unidad es profundizar en otra de las tareas populares del procesamiento del lenguaje natural orientada a la extracción de información, como es la clasificación (o categorización) automática de textos.

0. Objetivos de la unidad

Esta unidad se centra en otra de las tareas populares del procesamiento del lenguaje natural orientada a la extracción de información, como es la clasificación (o categorización) automática de textos, que consiste en asignar automáticamente a un texto de entrada una o varias categorías de un conjunto de categorías predefinidas según un modelo. En esta unidad se estudian los fundamentos y conceptos de la clasificación automática, sus aplicaciones, los diferentes escenarios existentes, se describen las métricas de evaluación que habitualmente se emplean en este tipo de sistemas y analizan en detalle las principales técnicas de clasificación automática.

El **apartado 1** realiza una introducción de los principales fundamentos y conceptos de este tema, describiendo los tipos de clasificación automática y sus aplicaciones más importantes.

El **apartado 2** presenta las métricas de evaluación que se utilizan habitualmente en cada uno de los escenarios de clasificación.

En el **apartado 3** se analizan y describen en detalle las diferentes técnicas de clasificación automática: modelos basados en reglas, sistemas basados en aprendizaje automático y, por último, modelos híbridos que combinan ambos enfoques.

Finalmente, el **apartado 4** se dedica a describir unos tipos específicos de clasificación automática que hoy en día se han vuelto muy populares y constituyen escenarios de aplicación en sí mismos.

1. Introducción

1.1. Fundamentos y conceptos básicos

La clasificación (o categorización) automática de textos (en inglés, *Text Categorization* o *Classification*) consiste en asignar automáticamente a un texto dado una o varias categorías (o clases) de un conjunto de categorías predefinidas en un modelo.

La clasificación de textos es una de las tareas fundamentales del procesamiento del lenguaje natural, con amplias aplicaciones, ya que puede utilizarse para organizar, estructurar y categorizar prácticamente cualquier tipo de texto, desde documentos, estudios médicos y archivos, hasta toda la web.

Por ejemplo, los artículos de un medio de comunicación pueden organizarse por temas; los tickets de soporte pueden organizarse por urgencia; las conversaciones de un chat pueden organizarse por idioma; las menciones de marcas pueden organizarse por sentimiento, etc.

Dado el texto (o *verbatim*) siguiente sobre una aplicación de banca móvil:

La interfaz de usuario es bastante sencilla y fácil de usar.

Un sistema de clasificación automática de texto puede tomar este texto como entrada, analizar su contenido y, a continuación, asignar automáticamente las etiquetas pertinentes entre las categorías de un modelo de clasificación, por ejemplo: “InterfazDeUsuario” y “FácilDeUsar”.

Un modelo de clasificación se compone de una serie de categorías, bien organizadas en forma de lista, es decir, sin ninguna jerarquía, o pueden estar organizadas en una taxonomía multinivel.

Por ejemplo, un departamento de Recursos Humanos que haga entrevistas a los empleados que dejan de trabajar en la empresa para conocer los motivos de su marcha (las llamadas entrevistas de salida o *exit interviews*), podría automatizar su proceso con un modelo de clasificación automática de las respuestas de los empleados. Para estar bien definido, este modelo debe contener todos los posibles motivos de marcha de los empleados (o al menos cubrirlos en gran parte), por ejemplo:

Categoría	Descripción
Remuneración	remuneración insuficiente
CargaDeTrabajo	carga de trabajo excesiva
AmbienteLaboral	ambiente laboral inadecuado
PolíticasOPrácticas	desacuerdo con las políticas o prácticas de la empresa
OfertaDeTrabajo	recepción de una oferta de trabajo más atractiva
NuevasExperiencias	búsqueda de nuevas experiencias o retos
MotivosPersonales	motivos personales

En este modelo, si la respuesta del empleado no coincide con ninguna de las categorías definidas, el texto no debería recibir ninguna categorización (es decir, el sistema no debería devolver etiqueta alguna). Como esto a veces puede confundirse con una falta de respuesta del sistema, es decir, un error, muchos modelos de clasificación incluyen una categoría de “Otros” para ser exhaustivos y cubrir todas las posibilidades. Así, si la respuesta del empleado no coincide con ninguna de las categorías, el sistema devolvería “Otros”.

En este caso se trata de una taxonomía en un solo nivel, sin jerarquía, plana, en modo de lista de categorías, pero muchos modelos habitualmente definen una taxonomía en varios niveles para especificar de forma más precisa las categorías. Por ejemplo, la categoría “AmbienteLaboral” podría tener las siguientes subcategorías:

Categoría	Descripción
AmbienteLaboral>Responsable	mala relación con su responsable (jefe)
AmbienteLaboral>Compañero	mala relación con algún compañero
AmbienteLaboral>Equipo	mala relación con todo el equipo
AmbienteLaboral>OtrosEquipos	mala relación con otros equipos

En función de su implementación, algunos sistemas solo devuelven las categorías más específicas, es decir, los últimos niveles de la taxonomía, y otros sistemas devuelven cualquiera de las categorías de forma independiente, es decir, por ejemplo, indicarían "AmbienteLaboral>Responsable" para indicar desavenencias con su jefe y también "AmbienteLaboral" para señalar otros motivos relacionados con el ambiente laboral no cubiertos por las subcategorías definidas. Para evitar esto, los modelos pueden incluir una subcategoría "Otros" en todos los niveles de la taxonomía (en este caso, "AmbienteLaboral>Otros").

Además, será importante tener en cuenta la jerarquía (cuando la haya) en los criterios de evaluación. Por ejemplo, ¿cómo de estricto se desea ser en la penalización si se clasifica el texto en "AmbienteLaboral" en vez de en "AmbienteLaboral>Equipo"?

La salida del sistema es una lista de ninguna, una o varias categorías, junto con su relevancia *(relevance, score)* respecto al texto, ordenadas de mayor a menor relevancia.

Por ejemplo, al analizar el texto de una respuesta de un empleado, el sistema podría devolver los siguientes resultados, indicando que el principal motivo de la marcha del empleado es por desavenencias con el jefe, seguido de desacuerdos con las políticas o prácticas de la empresa y finalmente, de forma más colateral, por motivos personales no especificados:

Categoría	Relevancia
AmbienteLaboral>Responsable	4.0
PolíticasOPrácticas	2.0
MotivosPersonales	0.5

El valor mostrado en el ejemplo anterior es un número real. En realidad existen dos tipos de relevancia:

Relevancia absoluta

La relevancia absoluta es un número real con la puntuación asignada por el sistema a ese texto para esa categoría.

Esta relevancia va desde 0 (el texto no tiene que ver en absoluto con esa categoría) hasta lo que considere el propio sistema, dependiendo de su implementación interna.

El hecho de no existir un límite superior en la relevancia absoluta hace más difícil comparar entre los resultados de clasificación de dos textos diferentes. Por ejemplo, si para un texto el sistema devuelve una relevancia de 1.4 para "MotivosPersonales" y para otro texto devuelve 6.9 para esa misma categoría, podría pensarse que el segundo texto se refiere más a motivos personales, pero esto no es cierto, ya que el segundo texto, por ejemplo, puede ser más largo y tener, por tanto, más referencias a palabras o expresiones relacionadas con esa categoría, lo que no significa que el texto está más relacionado con esa temática.

Relevancia relativa

La relevancia relativa es un porcentaje de importancia de cada categoría en el texto y se define desde 0% a 100% (o de 0.0 a 1.0 en tanto por uno).

Con la relevancia relativa es más fácil comparar diferentes textos. Por ejemplo, si un texto tiene una relevancia relativa para "MotivosPersonales" del 75% y otro texto tiene un 50%, esto significa que el primer texto está más relacionado con esa temática que el segundo.

Para calcular la relevancia relativa hay diferentes formas, pero, en general, los sistemas dividen el valor de relevancia absoluta de cada categoría por el valor de la relevancia absoluta de la categoría con mayor relevancia. Así, la categoría más relevante recibe un 100% y el resto de categorías un valor relativo decreciente según su importancia.

En el ejemplo planteado arriba, la relevancia relativa sería la siguiente:

Categoría	Relevancia	%
AmbienteLaboral>Responsable	4.0	100%
PolíticasOPrácticas	2.0	50%
MotivosPersonales	0.5	12.5%

Otra forma de calcular la relevancia relativa es dividiendo por la longitud del texto (en número de palabras o de frases).

Los beneficios de la clasificación automática son los siguientes:

Criterios coherentes

Un anotador humano comete errores al clasificar los datos de texto debido a las distracciones, el cansancio y el aburrimiento. Además, la subjetividad humana a veces produce criterios incoherentes. Los sistemas de clasificación automática, una vez que han sido entrenados correctamente, funcionan con la precisión necesaria y, además, aplican los mismos criterios homogéneos a todos los datos y resultados.

Escalabilidad

Un sistema de clasificación automática puede analizar automáticamente un alto volumen de textos a muy bajo coste y a gran velocidad, en la mayoría de casos, en tiempo real, lo que sería inviable de realizar de forma manual.

1.2. Tipos de clasificación

En este apartado se van a describir técnicamente los diferentes tipos de clasificación automática que existen, en función de las categorías que intervienen.

1.2.1. Clasificación binaria *(binary)*

Se denomina **clasificación binaria** *(binary)* cuando el modelo tiene solo clases o categorías, es decir, la tarea del sistema es escoger entre una u otra. Por extensión, se denomina clasificador binario aquel sistema que realiza una clasificación en dos clases.

Hay muchos ejemplos de aplicación de la clasificación binaria, como:

- Detección de *spam* en correo electrónico.
- Detección de mensajes abusivos (acoso, insultos, etc.) en correo electrónico, en redes sociales o blogs.
- Clasificación en información confidencial o no.
- Análisis de informes médicos para determinar si un paciente tiene una cierta patología o se le ha realizado un determinado análisis.
- Análisis de un texto técnico para comprobar si se cumple una especificación.

Los modelos de clasificación binaria son en general más sencillos, ya que solo tienen que centrarse en detectar las características discriminativas entre una categoría y otra (es decir, las que las diferencian).

La complejidad de la clasificación binaria viene cuando las dos clases no son simétricas, es decir, cuando una es mucho más frecuente que la otra. Por ejemplo, en el caso del clasificador de mensajes abusivos, la mayor parte de los mensajes serán lícitos y solo una pequeña parte serán mensajes abusivos.

Otro problema es cuando el coste de la decisión es muy diferente para una u otra clase. En el mismo ejemplo anterior, es menos grave detectar un mensaje como abusivo sin serlo (un falso positivo) que pasar por alto un mensaje abusivo (un falso negativo), ya que finalmente llega a su destinatario.

1.2.2. Clasificación multiclase *(multiclass)*

La **clasificación multiclase** *(multiclass)*, a veces denominada clasificación multinomial, es la extensión del caso binario y consiste en clasificar ejemplos escogiendo entre tres o más clases (categorías).

El ejemplo anterior de clasificación de las entrevistas de salida es un caso de clasificación multiclase.

Es un tipo de sistema más complejo porque es necesario discriminar entre más categorías. Como se verá más adelante, en el apartado de técnicas de clasificación, hay modelos que permiten de forma natural (intrínseca) el uso de más de dos clases. En aquellos casos con modelos binarios por naturaleza, es posible convertirlos en clasificadores multiclase mediante diversas estrategias, por ejemplo, la de uno-contra-todos *(one-vs-all o one-vs-rest)*, que construye un clasificador binario para cada clase frente a todas las demás, de tal forma que determina si un texto es de dicha clase o, por el contrario, de las demás.

1.2.3. Clasificación *single-label* vs. *multi-label*

Es importante no confundir la clasificación binaria y/o multiclase con la clasificación *single-label* (etiqueta única) y/o *multi-labels* (multietiqueta).

La clasificación *single-label* consiste en que el sistema asigna como máximo una única etiqueta a cada texto. Por extensión, la clasificación *multi-label* consiste en que el sistema puede asignar más de una etiqueta a cada texto.

Ambos casos son generalizaciones de la clasificación multiclase, ya que es el único escenario donde se puede elegir entre varias etiquetas (el caso binario solo consiste en decidir entre una y otra, y solo una de ellas, de forma exclusiva).

Formalmente, la salida de la clasificación multietiqueta se modela como un vector donde cada dimensión representa una categoría del modelo y el valor asignado a cada dimensión es la relevancia de dicha categoría respecto al texto. Habitualmente, muchos modelos calculan la relevancia (peso) de cada categoría de forma relativa, es decir, asignan un número real a cada una de ellas, de tal forma que la suma de todos los valores sea 1.0.

Por ejemplo, la salida [0.25, 0.0, 0.75, 0.0] indica que el sistema selecciona la categoría 3 como la más relevante (relevancia 0.75), la categoría 1 con relevancia 0.25 y las otras dos categorías no aparecen.

1.3. Aplicaciones

Como se describió en la unidad 1, algunas estimaciones calculan que alrededor del 80% de la información de las organizaciones no está estructurada y el texto es uno de los tipos más comunes de datos no estructurados. Debido a la naturaleza desordenada del texto, el análisis, la comprensión, la organización y la clasificación de los datos es difícil y requiere mucho tiempo, por lo que la mayoría de las empresas no aprovechan todo su potencial.

Mediante el uso de clasificadores de texto, las organizaciones pueden estructurar automáticamente todo tipo de texto relevante, desde correos electrónicos, documentos legales, medios sociales, chatbots, encuestas y otros, de una manera rápida y rentable. Esto permite ahorrar tiempo en el análisis de los datos de texto, automatizar los procesos empresariales y tomar decisiones de negocio basadas en datos.

La clasificación de textos se utiliza tanto para textos breves (por ejemplo, mensajes de redes sociales, consultas en chatbots, etc.) como para textos más extensos (por ejemplo, reseñas de clientes, artículos de noticias, contratos legales, encuestas a clientes con preguntas abiertas, etc.).

La clasificación de textos tiene miles de casos de uso y se aplica a una amplia gama de tareas. En algunos casos, las herramientas de clasificación de datos trabajan entre bastidores para mejorar las funciones de las aplicaciones con las que interactuamos a diario (como el filtrado de *spam* del correo electrónico). En otros casos, los clasificadores son utilizados por vendedores, gestores de productos, ingenieros y vendedores para automatizar los procesos empresariales y ahorrar cientos de horas de procesamiento manual de datos.

A continuación, se presentan algunos ejemplos de aplicaciones y casos de uso de la clasificación de textos.

Etiquetado de contenido o productos
Etiquetar contenidos o productos mediante categorías es una forma de mejorar la navegación o identificar contenidos relacionados en el sitio web de una organización. Por ejemplo, las plataformas de comercio electrónico, las agencias de noticias, los medios de comunicación o los blogs pueden utilizar estas tecnologías para clasificar y etiquetar contenidos y productos. La clasificación del contenido en el sitio web mediante etiquetas ayuda a Google a rastrear un sitio web con facilidad, lo que en última instancia ayuda al SEO. Además, la automatización de las etiquetas de contenido en el sitio web y la aplicación pueden mejorar la experiencia del usuario y ayudar a estandarizarlas. Otro caso de uso para los profesionales del *marketing* sería investigar y analizar las etiquetas y las palabras clave utilizadas por los competidores.

Análisis de sentimientos

Con la ayuda de la clasificación de textos, las empresas pueden dar sentido a grandes cantidades de datos utilizando técnicas como el análisis de sentimientos, basado en aspectos para entender de qué habla la gente y cómo habla de cada aspecto. Por ejemplo, una posible crisis de relaciones públicas, un cliente que está a punto de cambiar de opinión, quejas sobre un problema de errores o un tiempo de inactividad que afecta a más de un puñado de clientes.

Automatización de los procesos de atención al cliente

Construir una buena experiencia de cliente es una de las bases de una empresa sostenible y en crecimiento. Se estima que las personas tienen un 93% más de probabilidades de repetir como clientes en empresas con un excelente servicio de atención al cliente.

La clasificación de textos puede ayudar a los equipos de asistencia a proporcionar una buena experiencia. Al automatizar tareas, ahorra un tiempo valioso a los humanos, que pueden dedicarse a cosas más importantes o delicadas.

Por ejemplo, la clasificación de textos se utiliza a menudo para automatizar el análisis y enrutamiento de los tickets de soporte. La clasificación de textos permite dirigir automáticamente las solicitudes de asistencia a un compañero de equipo con conocimientos específicos sobre el producto en cuestión. O si un cliente escribe preguntando sobre reembolsos, puede asignar automáticamente el ticket al compañero de equipo con el permiso para realizar reembolsos. Esto garantizará que el cliente reciba una respuesta de calidad más rápidamente.

Los equipos de soporte también pueden utilizar la clasificación de sentimientos para detectar automáticamente la urgencia de un *ticket* de soporte y priorizar aquellos que contengan sentimientos negativos. Esto puede ayudar a reducir la pérdida de clientes e incluso a dar la vuelta a una mala situación.

Análisis de la voz del cliente

Las empresas aprovechan las encuestas para escuchar la voz de sus clientes (VoC, *Voice of the Customer*) en cada etapa del viaje. La información recopilada es tanto cualitativa como cuantitativa y, mientras que las preguntas cerradas (donde el cliente escoge una puntuación) son fáciles de analizar, las respuestas abiertas requieren un análisis más profundo utilizando técnicas de clasificación de texto.

En lugar de depender de los humanos para analizar los datos de la voz del cliente, se pueden procesar rápidamente los comentarios abiertos de los clientes con el aprendizaje automático. Los modelos de clasificación pueden ayudar a analizar los resultados de las encuestas para descubrir patrones y perspectivas como:

- ¿Qué le gusta a la gente de nuestro producto o servicio?
- ¿Qué deberíamos mejorar?
- ¿Qué debemos cambiar?

Al combinar los resultados cuantitativos y los análisis cualitativos, los equipos pueden tomar decisiones mejor fundamentadas sin tener que pasar horas analizando manualmente cada una de las respuestas abiertas.

Análisis de la voz del empleado

Cada vez son más las empresas que se comienzan a preocupar por el bienestar de los empleados. Hoy en día resulta indispensable conocer las necesidades de los empleados para poder mejorar su experiencia.

Típicamente, se recogen encuestas de opiniones de los empleados (vía correo electrónico, web…) con cierta frecuencia y, gracias a su análisis, se pueden impulsar medidas para aumentar su satisfacción e impulsar así su productividad, aumentar la retención y optimizar a la vez la experiencia del cliente.

2. Métricas de evaluación

2.1. Escenarios *single-label*

Las métricas utilizadas para evaluar el rendimiento de un clasificador de textos son las métricas estándar de las tareas de procesamiento del lenguaje natural presentadas en la unidad 1:

Precisión (precision)

El porcentaje de textos que se han clasificado con la etiqueta correcta.

Cobertura (recall)

El porcentaje de textos que el clasificador predijo para una etiqueta determinada respecto al número total de textos de evaluación que debería haber predicho para dicha etiqueta.

En otras palabras, el porcentaje de textos de una determinada etiqueta que el sistema es capaz de encontrar.

Medida-F

La media geométrica entre precisión y cobertura, con la fórmula estándar.

Matriz de Confusión

Presenta los resultados desagregados de cada categoría. En las columnas se muestran el valor predicho de cada una de las categorías de la taxonomía y en las filas su valor real.

		Valor Predicho		
		Gato	Perro	Conejo
Valor Real	Gato	5	3	0
	Perro	2	3	1
	Conejo	0	2	11

Ejemplo de matriz de confusión
Fuente de la imagen: Wikipedia.org

Sin embargo, por la redacción de la descripción de las métricas anteriores, se puede intuir que estas métricas solo valen para un caso *single-label*, es decir, cuando el clasificador devuelve una única etiqueta para cada texto.

Para evaluar el resultado de la clasificación, estas métricas se calcularán comparando la salida del sistema frente al etiquetado manual del conjunto de test realizado por los lingüistas.

2.2. Escenarios *multi-label*

El problema de los escenarios *multi-label* es que esas métricas no permiten evaluar bien los errores al devolver menos etiquetas de las que se deberían devolver (falsos negativos), o bien devolver más etiquetas de las previstas (falsos positivos).

La raíz del problema es que, en escenarios *multi-label*, las predicciones para un ejemplo son un conjunto de etiquetas y, por tanto, se puede considerar el concepto de solución totalmente correcta frente a la parcialmente correcta. Además, aparte de evaluar la calidad de la categorización en clases, también podríamos evaluar si las clases están correctamente clasificadas por relevancia.

Dos métricas muy básicas son la **cardinalidad** de las etiquetas, es decir, el número medio de etiquetas devueltas por el clasificador, y la **densidad** de etiquetas, es decir, el número de etiquetas por cada texto de ejemplo dividido por el número total de etiquetas de la ontología, promediado para todo el conjunto de test.

Como sucedía con la evaluación de la tarea de reconocimiento de entidades con nombre, si se ignoran las predicciones parcialmente correctas (y se consideran incorrectas), en escenarios de clasificación *multi-label* se puede utilizar una definición estricta con la métrica *exact match ratio* (o *subset accuracy*) que cuenta como acierto solo si se devuelven **todas y cada una de las categorías** asignadas manualmente, y ninguna otra. Es decir, es acierto si las etiquetas automáticas coinciden exactamente con las etiquetas manuales, luego es una métrica muy exigente, que mide el caso más difícil.

Ilustrando con un ejemplo, supongamos que el siguiente texto debe clasificarse en “Cubismo” y “Pintura”:

El cubismo es un movimiento artístico en pintura caracterizado por representar la realidad mediante elementos geométricos, fragmentando líneas y superficies.

Si el sistema devuelve “Cubismo” y “Pintura”, considerando *exact match ratio*, acierta completamente, por lo tanto, precisión del 100%.

Si un sistema devuelve únicamente “Cubismo”, le falta la etiqueta “Pintura” (un falso negativo), luego la precisión es de 1/2 (50%).

Si el sistema devuelve “Cubismo” y “Arte”, su precisión será también del 50%, pero en este caso por un falso negativo en la categoría “Pintura” y un falso positivo en la categoría “Arte”.

Y si el sistema devuelve “Cubismo”, “Pintura” y “Arte”, su precisión será de 2/3 (67%), porque le sobra la categoría “Arte” (falso positivo).

Como se trata de una métrica muy estricta, para capturar la noción de parcialmente correcto, necesitamos evaluar la diferencia entre el conjunto de etiquetas que se han devuelto frente a las que se debería haber devuelto realmente.

Shantanu Godbole y Sunita Sarawagi, en su artículo “Discriminative methods for multi-labeled classification”, publicado en Advances in Knowledge Discovery and Data Mining (Springer Berlin Heidelberg, 2004, págs. 22-30), propusieron la métrica **LBA** ***(Label Based Accuracy)***, que es hoy en día la medida de precisión multietiqueta más popular.

LBA mide el grado de coincidencia entre las categorías asignadas automáticamente por el sistema (S) y las categorías asignadas manualmente (T), calculando el cociente entre la intersección y la unión entre ambas listas de categorías. Es decir, LBA es el cociente entre las etiquetas automáticas que son además etiquetas manuales y la unión de etiquetas automáticas y etiquetas manuales.

$LBA = |T \cap S| / |T \cup S|$

Esto es, LBA devuelve “cuántas etiquetas son correctas entre todas las etiquetas devueltas”.

LBA es, de hecho, una métrica combinada de precisión y cobertura, ya que tiene en cuenta tanto FP (categorías en S que no deberían ser devueltas) pero también FN (categorías que faltan en S).

3. Técnicas de clasificación automática

Hay tres tipos de técnicas para realizar la clasificación automática de textos: sistemas basados en reglas, sistemas basados en técnicas de aprendizaje automático (incluido *deep learning*) y sistemas híbridos (que son una combinación de las otras dos).

3.1. Modelos basados en reglas

La primera forma de llevar a cabo la clasificación automática de texto es usar un enfoque clásico basado en conocimiento, común en los años ochenta, que consiste en la creación de un sistema experto con reglas de clasificación definidas de forma manual por lingüistas expertos.

Estas reglas lingüísticas, que incorporan o se basan en técnicas básicas de procesamiento del lenguaje natural, indican al sistema cómo debe utilizar la información significativa presente en el texto para identificar qué categorías del modelo de clasificación son relevantes respecto al contenido del texto.

Cada regla es de la forma *if-then* (si-entonces): su parte izquierda (antecedente o premisa) es una operación lógica que combina términos del texto, funciones de comprobación con los típicos operadores booleanos: AND, OR y NOT, y su parte derecha (consecuente o conclusión) indica que el texto se clasifica en dicha categoría.

if (expresión lógica) then (categoría)

si se cumple condición → el texto se clasifica en categoría

Adicionalmente, las reglas incluyen una puntuación, de tal forma que, si se cumple la condición, el texto se clasifica en la categoría agregando cierta puntuación (o indicador de relevancia) para esa categoría, es decir:

si se cumple condición → suma X a la relevancia de la categoría

Por ejemplo, se quiere clasificar temáticamente artículos de prensa, en dos grupos: Deportes y Política. Como primer paso, habrá que definir dos conjuntos de reglas que indiquen al sistema cuándo hay que clasificar el texto en cada una de esas categorías. Por ejemplo, se podría definir una lista de palabras (entidades y/o conceptos) relacionadas con ambos temas (por ejemplo, nombres de deportistas y de políticos, nombres comunes en cada temática) y definir una regla como:

para cada palabra de la lista X → suma 1 a la relevancia de la categoría X

En la práctica, el clasificador, dado un texto, asigna como relevancia de cada categoría el número de palabras que aparecen en la lista de palabras de dicha categoría. La categoría con más palabras en el texto sería la más relevante.

Por ejemplo, el siguiente texto se clasificará como “Deportes” porque aparecen varios nombres de equipos de fútbol:

El Real Madrid jugará la final de la Champion contra el Paris Saint Germain.

Si el sistema implementa un escenario *single-label*, devolverá únicamente una categoría, aquella cuya relevancia sea más alta. Si el escenario es *multi-label*, típicamente los sistemas emplean un umbral de relevancia, devolviendo todas las categorías cuyas relevancias están por encima de un valor dado.

La capacidad expresiva del lenguaje de reglas depende del motor de reglas del sistema. Algunos sistemas permiten utilizar información semántica de las entidades con nombre que aparecen en el texto (palabras y su tipo semántico), además de incluir macros y definiciones complejas, así como operadores complejos (como operadores de distancia, de coincidencia aproximada, etc.), lo que les confiere una gran capacidad expresiva y una máxima productividad en la definición de codificaciones y relaciones a extraer.

Por ejemplo:

S@Organization>Company>FinancialCompany>
BankingCompany AND NOT Banco_Santander → #Competidores

(Si se detecta la aparición de una entidad de tipo empresa-banco, bajo cualquiera de sus sinónimos, y esa entidad no es el Banco de Santander, entonces: clasificar el texto bajo la categoría Competidores).

S@LivingThing>Animal AND NOT {MASCOTA} → #NoMascotas

(Si se detecta la aparición de un animal, bajo cualquiera de sus sinónimos, con la excepción de los animales listados en la macro MASCOTA -que tendrá: “perro”, “gato”, “conejo”, “tortuga”…-, entonces: clasificar el texto bajo la categoría NoMascotas)

viajar|viaje|visitar|visita|turismo AND G@Europe>Italy → #Turismo>Italia

(Si aparecen los términos “viajar”, “viaje”, “visitar”, “visita” o “turismo”, bajo cualquiera de sus formas, y una entidad geográficamente perteneciente a Italia, como “Roma”, “Sicilia” o “Venecia”, entonces: clasificar el texto bajo la categoría Turismo>Italia)

El empleo de la información generada por el reconocimiento de entidades y almacenada en la ontología permite desarrollar reglas con gran capacidad expresiva.

Los sistemas basados en reglas tienen como principales ventajas que son fácilmente comprensibles para el ser humano y pueden mejorarse con el tiempo, editando el conjunto de reglas. Se asume comúnmente que se pueden producir reglas tan precisas como sea necesario, simplemente añadiendo más reglas o mejorando o ajustando las reglas existentes.

Sin embargo, un inconveniente es que requiere, por un lado, un conocimiento experto sobre el dominio de clasificación y, por otro, un conocimiento específico sobre el lenguaje de reglas, todo ello sin mencionar la dificultad intrínseca de modelar una categoría con un conjunto de operaciones lógicas sobre términos lingüísticos.

En cualquier caso, el principal problema es que la construcción del conjunto de reglas cuando se trata con muchos cientos o incluso miles de categorías es una ardua tarea que hace que esta aproximación sea inabordable en la mayoría de los escenarios del mundo real. Aparte del esfuerzo de desarrollo del elevado número de reglas que es necesario, en algunos casos, los sistemas pueden ser difíciles de mantener y adaptar, ya que añadir nuevas reglas puede afectar a los resultados de las reglas preexistentes.

3.1.1. Tarea del lingüista

La tarea del lingüista en este tipo de sistemas suele empezar con la generación de una primera versión del conjunto de reglas para cada categoría (es decir, la generación del modelo lingüístico) a partir de un conjunto de textos de ejemplo preclasificados (corpus de entrenamiento de cuyo etiquetado, muchas veces, también se encarga).

Para esta fase es imprescindible contar con un buen **corpus de entrenamiento extenso y estable** y, deseablemente, un documento de **definición y descripción de las categorías** (criterios de clasificación).

Estas reglas se evalúan y se van **refinando iterativamente a través de varias versiones del modelo hasta alcanzar la exactitud deseada.** Cada una de estas iteraciones se compone de las siguientes tareas:

Etiquetado automático del corpus
Etiquetado del corpus de test utilizando la versión actual del modelo de reglas.
Evaluación de resultados
Evaluación de los resultados del etiquetado, aplicando las métricas más adecuadas, típicamente: precisión, cobertura y la matriz de confusión (que nos da resultados desagregados de falsos positivos y negativos por categoría).
Análisis de error
Análisis de los resultados de la evaluación, identificando los gaps con los valores deseados de las métricas.
Definición de la estrategia de ajuste
Formulación de una estrategia para la siguiente iteración centrándose, por ejemplo, en las categorías más frecuentes (que tendrán más presencia en el resultado final) y/o donde el *gap* de error es más alto.

Ajuste de las reglas

Refinamiento de las reglas para dichas categorías en función de las mejoras en las métricas que se desee alcanzar.

- Si es falso negativo: añadir reglas para cubrir ese caso, a partir del texto, identificando patrones e intentando generalizar.
- Si es falso positivo: ajustar las reglas actuales o añadir reglas específicas que excluyan ese caso, o términos negativos que restrinjan las reglas existentes.

También puede ocurrir que la regla acierte y haya sido el anotador humano del corpus quien haya fallado a la hora de categorizar algún *verbatim*. En ese caso, lo que habría que ajustar sería el etiquetado del texto correspondiente.

Y vuelta al paso 1.

Los siguientes aspectos son bastante importantes para que el proyecto se desarrolle con éxito:

1

Disponer de una taxonomía completa, no ambigua y sin solapamientos es primordial. Es vital partir de un conjunto de categorías sólido y bien definido, con criterios claros y que evite las “zonas grises” entre unas y otras, y cubra todos los casos relevantes.

2

Además, es importante que la taxonomía esté orientada al objetivo del proyecto. Es conveniente aplicar una categorización que permita extraer el valor de negocio de los mensajes a analizar y tomar decisiones accionables, y no tanto identificar lo que dice el cliente.

3

Debido a que el proceso está muy influido por la frecuencia de las categorías, resulta de suma importancia tener un corpus equilibrado, representativo de las categorías esperadas en el sistema real.

4

El corpus de test debe estar etiquetado con las mismas etiquetas y siguiendo exactamente los mismos criterios que el corpus de entrenamiento y que esté formado por una muestra que represente al conjunto total de textos, de modo que los estadísticos, frecuencias de distribución de las categorías, etc. sean iguales en ambos. De no ser así, se estaría introduciendo un error sistemático en el proceso de evaluación.

5

Disponer de un corpus de entrenamiento estable resulta vital. Es importante que el conjunto de datos que se usa para configurar el sistema no sufra modificaciones durante el proceso, de lo contrario, será difícil que el entrenamiento vaya convergiendo y aproximándose a la calidad deseada. Es cierto que, como se ha mencionado en el paso 5 del proceso anterior, si el etiquetado es incorrecto puntualmente, corregirlo no debería suponer problemas para el sistema.

6

El preprocesado del texto es clave en muchos proyectos. Es imprescindible limpiar el contenido que se va a analizar de formularios, firmas, mensajes de respuesta, etc. para centrarse en el texto relevante.

7

Si intervienen varios idiomas, la identificación del idioma es un paso crucial. Esta fase tiene una gran influencia en los resultados finales. Es necesario asegurar que la identificación es la mejor posible, ya que los errores se acumulan.

3.2. Técnicas de aprendizaje automático

Por otro lado, el enfoque de aprendizaje automático se ha convertido desde los años noventa en el más popular para realizar la clasificación automática. En este caso, se proporciona al sistema un conjunto de textos preclasificados (etiquetados o anotados) para cada categoría, que se usa como conjunto de entrenamiento para construir un clasificador que emplea alguno de los algoritmos existentes de aprendizaje supervisado.

La ventaja es que solo se necesita un mínimo conocimiento del dominio para asignar una categoría a cada texto o *verbatim* del conjunto de entrenamiento, lo que implica una carga de trabajo mucho menor que la escritura de las reglas.

El primer paso para entrenar un clasificador basado en aprendizaje automático consiste en la extracción de características *(features)* del texto, aptas para su procesamiento por el algoritmo matemático en que se base el aprendizaje, para lo cual se utiliza algún método para transformar dicho texto en una representación numérica en forma de vector.

Uno de los enfoques de representación más utilizados es el método de la bolsa de palabras *(bag of words)*, que representa el texto mediante un vector donde cada dimensión es una palabra de un diccionario de palabras predefinido y el peso del vector en cada dimensión representa la frecuencia o peso de dicha palabra en el texto.

Por ejemplo, dado un diccionario definido con las palabras siguientes:

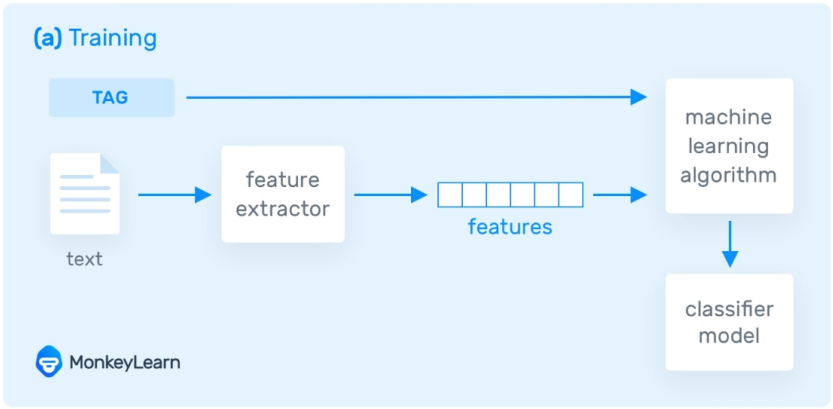
{esto, es, el, no, impresionante, malo, baloncesto}

Al vectorizar el texto "Esto es impresionante", se obtiene la siguiente representación vectorial:

(1, 1, 0, 0, 1, 0, 0)

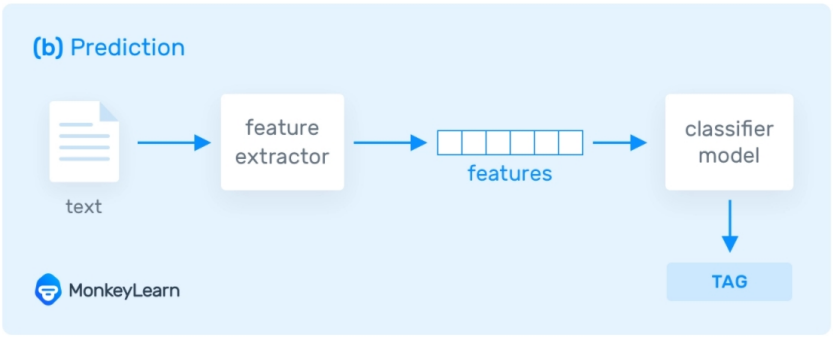
El siguiente paso es ejecutar el algoritmo de aprendizaje automático con los datos de entrenamiento que consisten en pares de conjuntos de características (el vector anterior para cada ejemplo de texto) y las etiquetas aplicables a cada ejemplo, para producir un modelo de clasificación.

Lo habitual es eliminar las palabras de parada *(stopwords)* y quedarse solo con las palabras que aportan significado semántico, típicamente nombres, adjetivos, verbos y adverbios.



Entrenamiento del modelo de clasificación automática
Fuente de la imagen: MonkeyLearn.com

Una vez entrenado el modelo, se puede utilizar para clasificar textos nuevos. El mismo extractor de características se utiliza para transformar el texto en conjuntos de características, que se introducen en el modelo de clasificación para obtener las predicciones sobre categorías.



Predicción empleando el modelo de clasificación automática
Fuente de la imagen: MonkeyLearn.com

Se han utilizado numerosos algoritmos y técnicas de aprendizaje supervisado para construir dichos clasificadores, como, por ejemplo, Naive Bayes, regresión lineal o polinómica, Nearest Neighbor, árboles de decisión, redes neuronales, máquinas de vectores de soporte, el algoritmo LSA, *boosting*, algoritmos genéticos, etc.

Los algoritmos estadísticos **Naive Bayes** han sido tradicionalmente muy populares para clasificación automática, porque funcionan bien incluso cuando el conjunto de datos de entrenamiento no es demasiado grande y los recursos computacionales son escasos.

Por el mismo motivo, las **máquinas de vectores soporte** (*support vector machines*, SVM) son también otro popular algoritmo de aprendizaje automático para la clasificación de textos. Aunque en este caso los SVM requieren más recursos informáticos que Naive Bayes, los resultados obtenidos suelen ser todavía más rápidos y precisos.

En los últimos dos o tres años, los enfoques basados en *deep learning*, como caso específico del aprendizaje automático, han supuesto una revolución en la clasificación automática (como en todas las tareas de procesamiento del lenguaje natural). Estos modelos se basan en arquitecturas neuronales profundas con redes neuronales recurrentes (RNN) y/o convolucionales (CNN) combinadas con capas de representación de *embeddings*.

Los algoritmos de *deep learning* suelen sustituir la representación del texto como *bag-of-words* por una representación basada en *embeddings*, bien con modelos independientes del contexto como Word2Vec o GloVe, o bien utilizando arquitectura de *transformers* con *embeddings* de tipo BERT o derivados. De esta forma, se logra una mejor representación vectorial del contenido semántico del texto y mejorar la precisión de los clasificadores.

El problema de los algoritmos de *deep learning* es que requieren (en general, sin usar aprendizaje por transferencia) de un ingente volumen de datos de entrenamiento, mucho mayor que el que necesitan los algoritmos de aprendizaje automático tradicionales (con millones de ejemplos etiquetados). Pero su ventaja es que sus resultados resultan incomparables, en términos de precisión y cobertura, respecto a los métodos tradicionales.

En definitiva, cada algoritmo utiliza una aproximación diferente (basada en estadística, probabilidad, distancia respecto a los ejemplos, lógica borrosa, redes neuronales profundas, etc.), pero todas ellas, de una forma u otra, se basan en el hecho de que cuantas más veces aparezca un término en el texto, más relevante es ese término para la categoría.

Aunque se ha demostrado que el enfoque de aprendizaje automático puede generar clasificadores igual de buenos que los sistemas basados en reglas, pero con un menor esfuerzo, también tienen sus inconvenientes, fundamentalmente relacionados con el hecho de que (en la mayoría de los algoritmos de aprendizaje empleados) el modelo no es comprensible por el ser humano, con lo que es difícil diagnosticar la razón de los falsos positivos/negativos para poder ajustar el sistema.

Así, en la práctica, la única manera de mejorar el clasificador es invertir un mayor esfuerzo en la construcción del conjunto de entrenamiento y evaluar distintas alternativas para construir los vectores de rasgos.

3.2.1. Tarea del lingüista

En este caso, la tarea del lingüista pasa principalmente por desarrollar, corregir o incrementar el volumen del conjunto de textos de entrenamiento y evaluación.

Adicionalmente, podría colaborar en realizar los ajustes apropiados a los parámetros de los algoritmos de aprendizaje y/o ayudar a definir la estrategia de representación semántica del texto como vector, por ejemplo, decidiendo sobre el valor de la longitud máxima de secuencia del algoritmo, el tamaño del vocabulario, si se normaliza o no la capitalización del texto, ajustes en la segmentación en tokens, etc.

El proceso típico de trabajo se ilustra a continuación:

Etiquetado del corpus de entrenamiento y test
Selección de textos y anotación del corpus de entrenamiento y test.
Entrenamiento del modelo y generación de resultados
Entrenamiento del modelo de aprendizaje automático utilizando el corpus de entrenamiento y ejecución del modelo sobre el corpus de test para generar los resultados de la clasificación automática.
Evaluación de resultados
Evaluación de los resultados del etiquetado, aplicando las métricas más adecuadas.
Análisis de error
Análisis de los resultados de la evaluación, identificando los fallos más importantes del sistema.
Definición de la estrategia de ajuste
Formulación de una estrategia para la siguiente iteración centrándose, por ejemplo, en las categorías con más error. Y vuelta al paso 1.

3.3. Modelos híbridos

Los sistemas híbridos consisten en combinar los dos enfoques anteriores.

En general, suelen combinar un primer clasificador entrenado mediante aprendizaje automático, que es “sencillo” y “barato” de construir a partir de datos de entrenamiento, con un segundo clasificador posterior, conectado en cascada a su salida, basado en reglas, que se utiliza para filtrar y mejorar los resultados del primer clasificador.

Así, estos sistemas híbridos pueden ajustarse fácilmente añadiendo reglas específicas para aquellas etiquetas conflictivas que no han sido modeladas correctamente por el clasificador base.

Lo bueno de los modelos híbridos es que presentan las ventajas de ambas aproximaciones y resuelven los problemas:

- No son excesivamente costosos de construir, ya que no hay que desarrollar miles de reglas, sino solo aquellas que sean necesarias para corregir los errores del clasificador basado en aprendizaje automático.
- Pueden mejorarse de forma sencilla, añadiendo reglas para detectar categorías que faltan (falsos negativos) o eliminar categorías que sobran (falsos positivos).

4. Tipos específicos de clasificación

4.1. Análisis de sentimientos

El análisis de sentimientos (también conocido como análisis de opinión) es probablemente el ejemplo más popular de clasificación de textos y consiste en el proceso automatizado de lectura de un texto en busca de la polaridad de sentimiento expresada en él.

Típicamente, la polaridad del sentimiento se representa mediante una escala de etiquetas para categorizar desde sentimientos muy positivos *(strong positive)*, positivos *(positive)*, neutros *(neutral)*, negativos *(negative)* hasta muy negativos *(strong negative)*. Hay sistemas que diferencian entre sentimiento neutro (ni positivo ni negativo), por ejemplo: “el libro ni me gusta ni me disgusta”, y ausencia de sentimiento: “el libro tiene 200 páginas”.

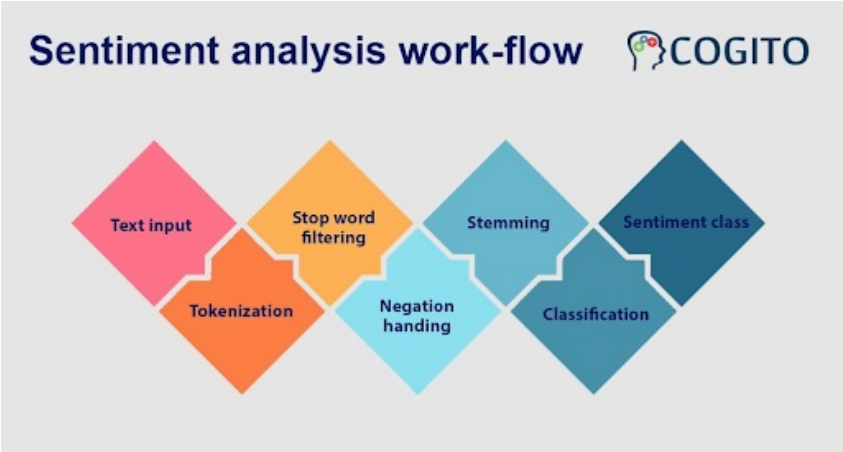
Algunos sistemas, sin embargo, representan la polaridad de sentimiento como un número real entre -1 (muy negativo) y 1 (muy positivo), pasando por el 0 (neutral).

Es posible convertir etiquetas a números reales asociando un valor numérico o discretizando el valor en intervalos (por ejemplo, si la polaridad es mayor o igual a 0.5, asignar “muy positivo”, etc.).

Esta tarea es crucial para empresas de productos, servicios, medios de comunicación, etc. Este tipo de empresas utilizan los clasificadores de sentimiento con muchos objetivos como el análisis de productos, la supervisión de marcas, los estudios de mercado, la atención al cliente, el análisis de la mano de obra, etc. A menudo monitorizan el sentimiento en las redes sociales para rastrear y analizar los comentarios sobre una determinada marca en general, un producto concreto, una característica particular o compararse con la competencia.

Desde el punto de vista de la clasificación, se trata de un escenario *single-label* donde el sistema asigna una única etiqueta. Pero esta etiqueta puede asignar un valor de “sentimiento global”, promediando la polaridad para el conjunto del contenido del texto, o analizar el “sentimiento asociado a aspectos”, es decir, asignar una etiqueta de polaridad a cada frase, cada entidad, cada temática detectada en el texto, etc. (lo que resulta especialmente interesante en textos largos que contienen información y sujetos/entidades de diversa índole).

El análisis de sentimiento presenta muchas dificultades desde el punto de vista de la pragmática lingüística. No se trata de una clasificación de carácter temático, donde las palabras se usan más o menos unívocamente para clasificar el texto en una categoría, sino que se trabaja con palabras con marcas de polaridad clara (por ejemplo, “bueno” indica polaridad positiva, “malo” indica polaridad negativa, “pésimo” indica polaridad muy negativa) que pueden verse afectadas por intensificadores (“muy bueno” representa polaridad muy positiva) o negadores que invierten su polaridad (“no es bueno” manifiesta polaridad negativa). Por esta razón, para analizar el sentimiento es muy importante disponer de un motor sintáctico capaz de reconocer la presencia de intensificadores y negadores y establecer a qué palabras afectan en el árbol sintáctico para reforzar o invertir su polaridad.



Proceso de análisis de sentimiento
Fuente de la imagen: Cogito

Por otro lado, tener en cuenta las palabras en su contexto resulta especialmente importante. Mientras que con "morir" podría asociarse una polaridad negativa, "morirse de risa" es altamente positivo. Detectar sutilezas como el sarcasmo y la ironía en ocasiones es computacionalmente factible (sobre patrones o convenciones lingüísticas), pero en otras es realmente complicado transferir este conocimiento a la máquina o que ella misma lo infiera de manera automática.

Por último, estos sistemas a veces incluyen un indicador de *agreement* o *disagreement* que señala si el texto contiene polaridades positivas y negativas (aunque predominen unas frente a otras).

La gran dificultad del entrenamiento de este tipo de sistemas reside en que para el propio humano (que etiqueta y valida) es complicado y subjetivo interpretar la intención comunicativa de otros, más aún expresada por escrito. En muchas ocasiones es difícil ponerse de acuerdo en tareas de evaluación, lo que exige definir muy bien los criterios en situaciones ambiguas a la hora de etiquetar los corpus y, aun así, es habitual utilizar técnicas de votación para terminar de aprobar las conclusiones (se etiquetan los corpus por más de un evaluador y se selecciona la etiqueta más elegida en caso de desacuerdo).

Los emoticonos surgieron y han ido evolucionando para imitar las expresiones faciales con el fin de utilizarlos como recurso comunicativo para asegurar la correcta interpretación de los sentimientos y emociones trasladados por escrito.

Ejemplo de emoticono
Fuente de la imagen: Wikipedia.org

4.2. Análisis de emoción

El análisis de emoción está relacionado con el de sentimiento, pero el objetivo del algoritmo no es solo saber si se trata de una comunicación positiva, negativa o neutra, sino detectar si el sentimiento se relaciona con una emoción y de qué emoción se trata (felicidad, tristeza, sorpresa, enfado...).

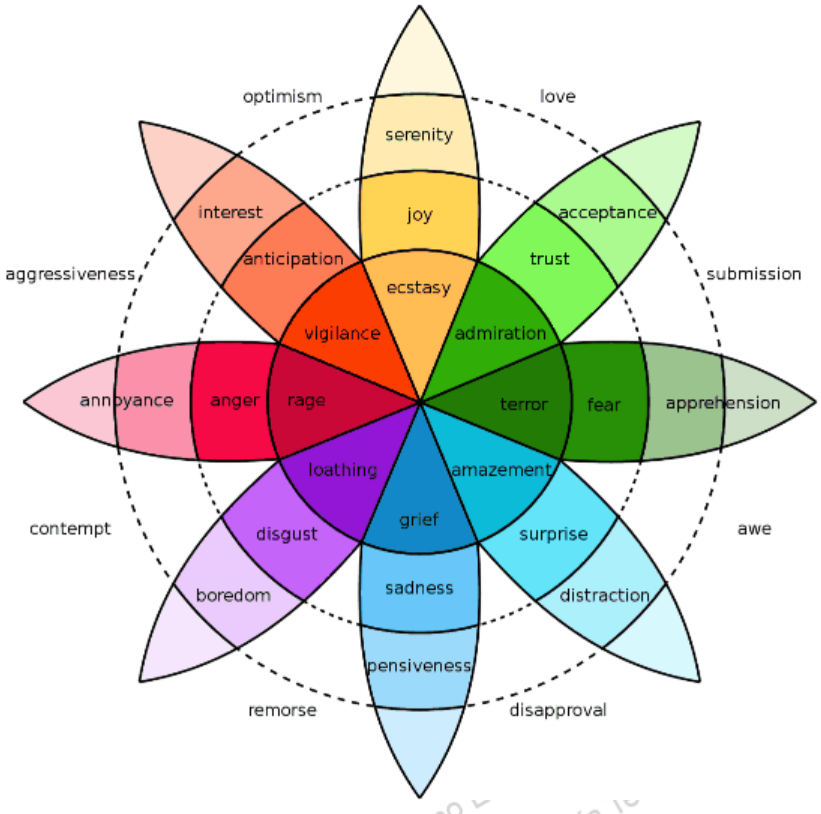
Aunque el análisis de sentimiento puede ser extremadamente útil, no profundiza en las razones subyacentes del resultado. Por ejemplo, “Este producto no es atractivo” y “Odio este producto” indican ambos sentimientos negativos, pero emocionalmente son muy diferentes. El análisis de emoción asignaría una etiqueta distinta a cada uno de los ejemplos.

Así, los vendedores y/o los equipos de atención al cliente actuarían de forma diferente con cada cliente. Por ejemplo, para tratar a los clientes enfadados, además de atenderlos con prioridad por tratarse de usuarios descontentos (con sentimiento negativo), las empresas suelen actuar con procedimientos que implican el uso de un lenguaje suave para calmarlos y la propuesta de compensaciones. En cambio, si observan que el cliente se muestra inseguro (de nuevo, un sentimiento negativo), la manera de atenderle será a través de un lenguaje asertivo y firme para mitigar su desconfianza.

Igualmente, las empresas utilizan la monitorización de su marca en redes sociales para rastrear, por ejemplo, posibles reacciones de clientes y localizar entre ellas las quejas o malas experiencias que poder responder de inmediato.

Existen muchos modelos de análisis de la emoción. Paul Ekman fue pionero en el estudio de las emociones y su relación con las expresiones faciales, y definió su famoso conjunto de emociones básicas: alegría *(happiness)*, tristeza *(sadness)*, miedo *(fear)*, asco *(disgust)*, ira *(anger)*, y sorpresa *(surprise)*.

Otro modelo muy conocido es el de Robert Plutchik, que propuso un enfoque de clasificación representado en la famosa **"rueda de las emociones"** para explicar su propuesta de forma gráfica. Su esquema consiste en ocho emociones bipolares básicas: alegría-tristeza *(joy-sadness)*, confianza-asco *(trust-disgust)*, ira-miedo *(anger-fear)* y sorpresa-anticipación *(surprise-anticipation)*.



Rueda de las emociones de Plutchik

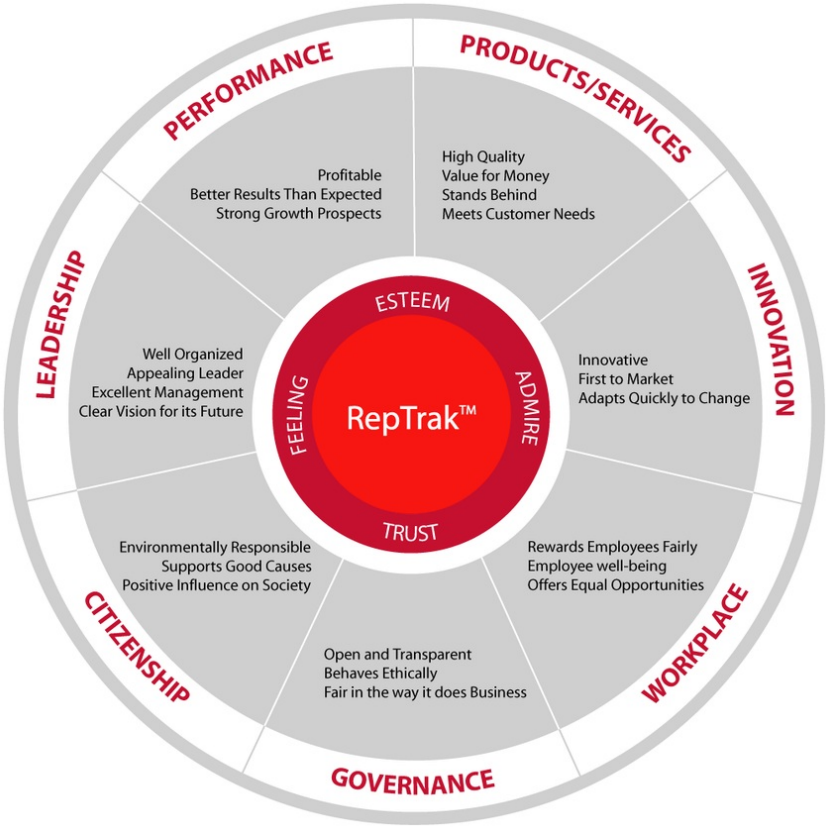
Igual que en el análisis de sentimientos, es esencial considerar la negación para poder identificar correctamente la emoción (por ejemplo, “no soy feliz” indica tristeza, lo contrario de felicidad).

4.3. Análisis de reputación

La reputación es la percepción que tiene el público sobre una empresa u organización basándose en su experiencia personal (directa o indirecta) con ella. Abarca desde la calidad de sus productos o servicios hasta la forma en que tratan a sus clientes o a sus empleados.

En definitiva, se trata de un escenario de clasificación de textos en una serie de dimensiones o categorías específicamente definidas para analizar este tipo de juicios personales.

Un modelo muy popular de análisis de reputación es el modelo RepTrak, propuesto por el Reputation Institute, con siete dimensiones de análisis: Productos/Servicios, Innovación, Lugar de trabajo, Gobernanza, Ciudadanía, Liderazgo y Rendimiento.



Modelo RepTrak de análisis de reputación
Fuente de la imagen: RepTrak.com

Como tarea de procesamiento del lenguaje natural, el análisis de reputación es un escenario de clasificación *multi-label*, ya que el modelo puede devolver ninguna, una o varias etiquetas relacionadas con las distintas dimensiones reputacionales.

Por ejemplo, el sistema devolvería “Liderazgo” y “Lugar de trabajo” para el texto:

Me gusta mucho la organización de la empresa y el horario flexible me permite compatibilizar mi vida personal

4.4. Detección de idioma

La detección automática del idioma (o idiomas) de un texto es otro escenario de la clasificación automática de textos. El objetivo es asignar al texto de entrada una o varias etiquetas que corresponden a un código de idioma.

Estos clasificadores de texto se utilizan a menudo con fines de enrutamiento, por ejemplo, para asignar los *tickets* de soporte al equipo apropiado según su idioma, o como primer paso cuando se desconoce el idioma del texto para luego aplicar otros modelos de clasificación u otros servicios de extracción donde sea necesario especificar el idioma del texto.

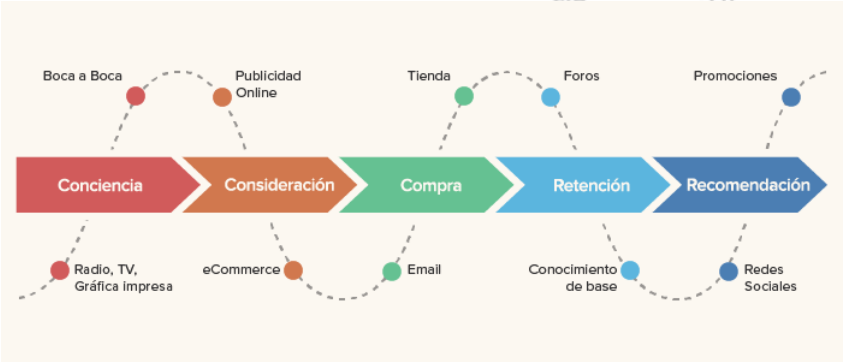
4.5. Detección de intenciones

La detección de intenciones (o clasificación de intenciones) es otro escenario popular de la clasificación automática que analiza el texto para comprender el motivo último que hay detrás de ciertos comentarios de los clientes.

El objetivo es detectar en las interacciones con el centro de atención al cliente, encuestas o conversaciones en redes sociales, si lo que hace el cliente es pedir información sobre productos o servicios, pretende comprarlos, solicita asistencia, hace una reclamación, quiere cancelar una suscripción, etc., de tal manera que se le pueda ofrecer un servicio personalizado que optimice su experiencia.

Por ejemplo, un modelo de análisis de intenciones básicas que resume el llamado "viaje del cliente" *(customer's journey)* en el proceso comercial podría incluir las siguientes etiquetas:

- Información (el cliente solicita información).
- Asesoramiento (el cliente pide asesoramiento para decidirse).
- Compra (el cliente quiere comprar).
- Soporte (el cliente pide soporte).
- Recomendación (el cliente recomienda nuestro producto o servicio).
- Reclamación (el cliente hace una reclamación).
- Cancelación (el cliente cancela el producto o servicio).



El viaje del cliente

De nuevo, se trata de un escenario de clasificación *multi-label*, con un modelo basado en categorías que representan las intenciones de los clientes. Y, una vez más, será necesario contemplar el tratamiento de la negación (para distinguir "comprar" de "no comprar"), así como la información de tiempo e intención implícitas en las formas y perífrasis verbales clave que aparezcan en el texto ("querer comprar" vs. "compré").

RESUMEN

En esta unidad se han presentado los conceptos y fundamentos principales de la clasificación automática, una de las tareas del procesamiento del lenguaje natural cuyo objetivo es la **asignación a un texto de entrada de una o varias etiquetas** de entre las definidas en un modelo.

Existen diferentes escenarios de clasificación. Se denomina **clasificación binaria** *(binary)* cuando el modelo tiene solo clases o categorías, es decir, la tarea del sistema es escoger entre una u otra. La **clasificación multiclase** *(multiclass)*, a veces denominada clasificación multinomial, es la extensión del caso binario y consiste en clasificar ejemplos escogiendo entre tres o más clases. Adicionalmente, la **clasificación *single-label*** consiste en que el sistema asigna como máximo una única etiqueta a cada texto. Por extensión, la **clasificación *multi-label*** consiste en que el sistema puede asignar más de una etiqueta a cada texto. La complejidad del problema que se pretenda abordar es diferente según el escenario, como se ha descrito.

En referencia a las métricas de evaluación, hay que distinguir los escenarios de *single-label* y *multi-label*. En el primer caso, las métricas de evaluación empleadas habitualmente son las típicas **métricas de precisión, cobertura y medida-F**. Sin embargo, para el segundo caso, como el sistema devuelve un conjunto de etiquetas, el uso de la misma métrica requeriría un acierto completo del sistema en todas y cada una de ellas, la llamada *exact match ratio* (lo que puede resultar injusto). Por este motivo, normalmente se utiliza la **métrica LBA** *(label based accuracy)*, que también considera los aciertos parciales.

La primera técnica para llevar a cabo la clasificación automática de texto es usar un **enfoque clásico basado en conocimiento**, común en los años ochenta, que consiste en la creación de un sistema experto con reglas de clasificación definidas de forma manual por lingüistas expertos. Estas reglas lingüísticas, que incorporan o se basan en técnicas básicas de procesamiento del lenguaje natural, indican al sistema cómo debe utilizar la información significativa presente en el texto para identificar qué categorías del modelo de clasificación son relevantes respecto del contenido del texto.

Por otro lado, el enfoque de aprendizaje automático se ha convertido desde los años noventa en el más popular para realizar la clasificación automática. En este caso, se proporciona al sistema un **conjunto de textos preclasificados** (etiquetados o anotados) para cada categoría, que se usa para entrenar y construir un clasificador empleando alguno de los algoritmos existentes de aprendizaje supervisado.

Los **sistemas híbridos** consisten en combinar los dos enfoques anteriores: un primer clasificador entrenado mediante aprendizaje automático, que es "sencillo" y "barato" de construir a partir de datos de entrenamiento, y un segundo clasificador posterior, conectado en cascada a su salida, basado en reglas (propuestas por humanos), que se utiliza para filtrar y mejorar los resultados del primer clasificador.

Ejercicios

Ejercicio 3: Clasificación de textos con un sistema real

Duración estimada del ejercicio
40 minutos

1. Introducción

El objetivo de este ejercicio es familiarizarse con la tarea de clasificación de textos empleando un sistema real.

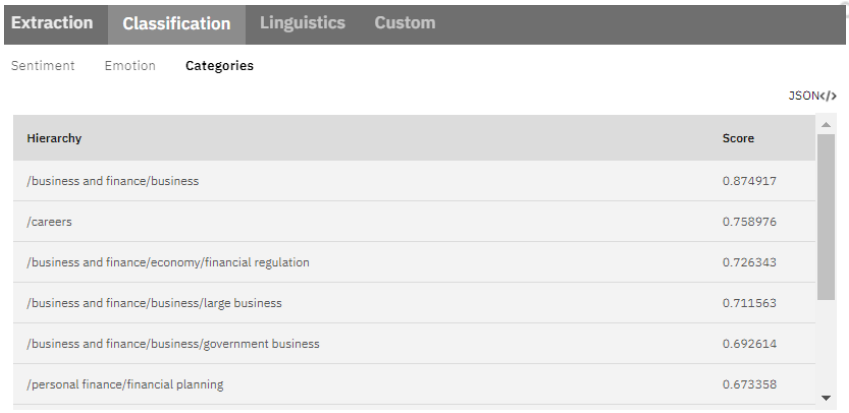
Como en la unidad anterior, en este ejercicio vamos a utilizar las páginas de demostración de IBM Watson Natural Language Understanding y de Google Cloud Natural Language.

2. IBM Watson Natural Language Understanding

Como en el primer ejercicio de la unidad anterior, accede al sitio web de la plataforma y pulsa sobre el botón “View demo” para ir al demostrador: https://www.ibm.com/cloud/watson-natural-language-understanding

Deja las opciones por defecto y pulsa en el botón “Analyze Text” para obtener los resultados para el texto de ejemplo del dominio legal.

Los resultados de clasificación se muestran en la pestaña **Classification**. La opción **Categories** presenta los resultados de la clasificación temática en categorías, como se muestra en la figura siguiente. El tema del texto es principalmente “/business and finance/business” con relevancia 0.88, y la segunda categoría es “/careers” con relevancia 0.76.



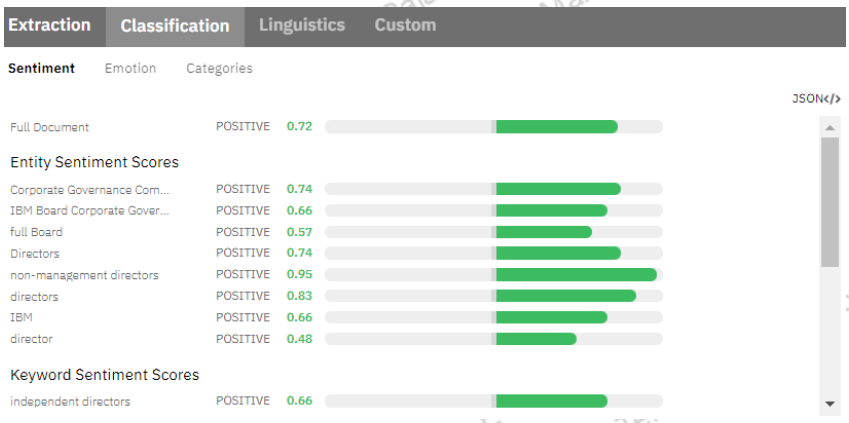
Clasificación temática

Observa que es una clasificación jerárquica en varios niveles: https://cloud.ibm.com/docs/natural-language-understanding?topic=natural-language-understanding-categories-hierarchy

Pulsando en el enlace “Learn More” del cuadro de ayuda sobre los resultados de **Categories**, se accede a la siguiente página donde se indica que se trata de una clasificación de 4 niveles según la taxonomía IAB 2.0.

Prueba también el texto de ejemplo del dominio financiero (“Financial”), para el que se obtiene que el tema principal es “/technology & computing/computing/computer software and applications/operating systems” seguido de “/technology & computing/computing/computer software and applications/graphics software”.

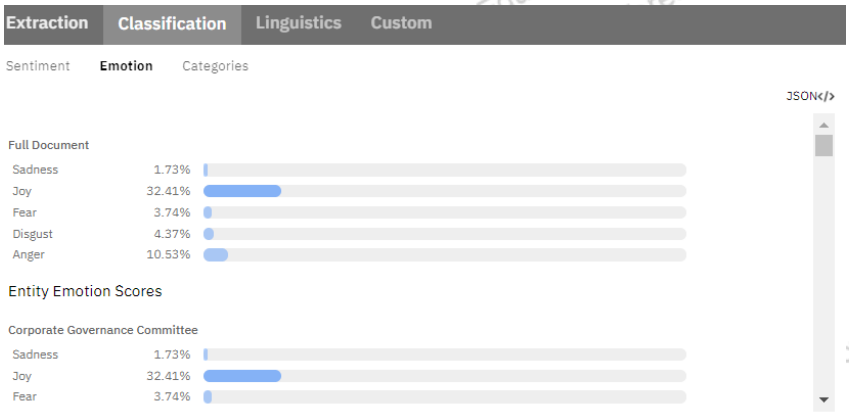
La pestaña **Sentiment** muestra los resultados del análisis de sentimientos (figura siguiente), tanto a nivel global (“Full Document”), en este caso POSITIVE, como a nivel de entidades y conceptos (*keywords* en la nomenclatura de ese sistema), también POSITIVE para todos ellos. La clasificación se muestra como un número entre -1.0 (negativo) y 1.0 (positivo), pasando por 0.0 (neutral).



Análisis de sentimientos

Prueba el texto “I didn't like his last movie” y comprueba que el resultado es una polaridad muy negativa.

Por último, la pestaña de **Emotion** presenta el análisis de emociones, también tanto a nivel global ("Full Document") como a nivel de cada entidad, como se muestra en la figura siguiente. El modelo empleado es el modelo de emociones de Paul Ekman, con las emociones básicas: alegría *(joy)*, tristeza *(sadness)*, miedo *(fear)*, asco *(disgust)* e ira *(anger)*, evaluadas de 0 a 100%.



Análisis de emoción

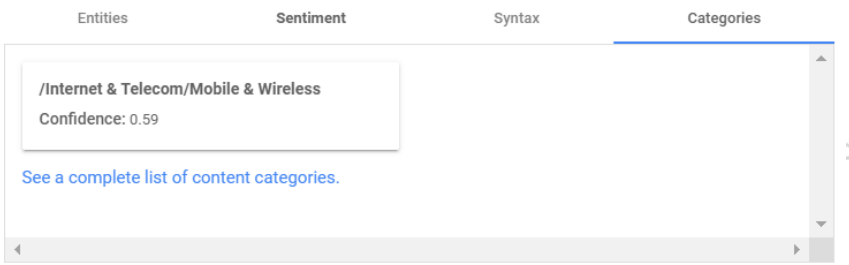
Prueba con otros textos y analiza los resultados obtenidos.

3. Google Cloud Natural Language

De nuevo, accede al sitio web de la plataforma y desplaza la ventana hacia abajo para llegar a la sección del demostrador: https://cloud.google.com/natural-language

Deja el texto de ejemplo y pulsa el botón "Analyze".

Los resultados de clasificación temática se presentan en la pestaña **Categories**, como se muestra en la figura siguiente. El tema del texto es principalmente "/Internet & Telecom/Mobile & Wireless" con relevancia 0.59.

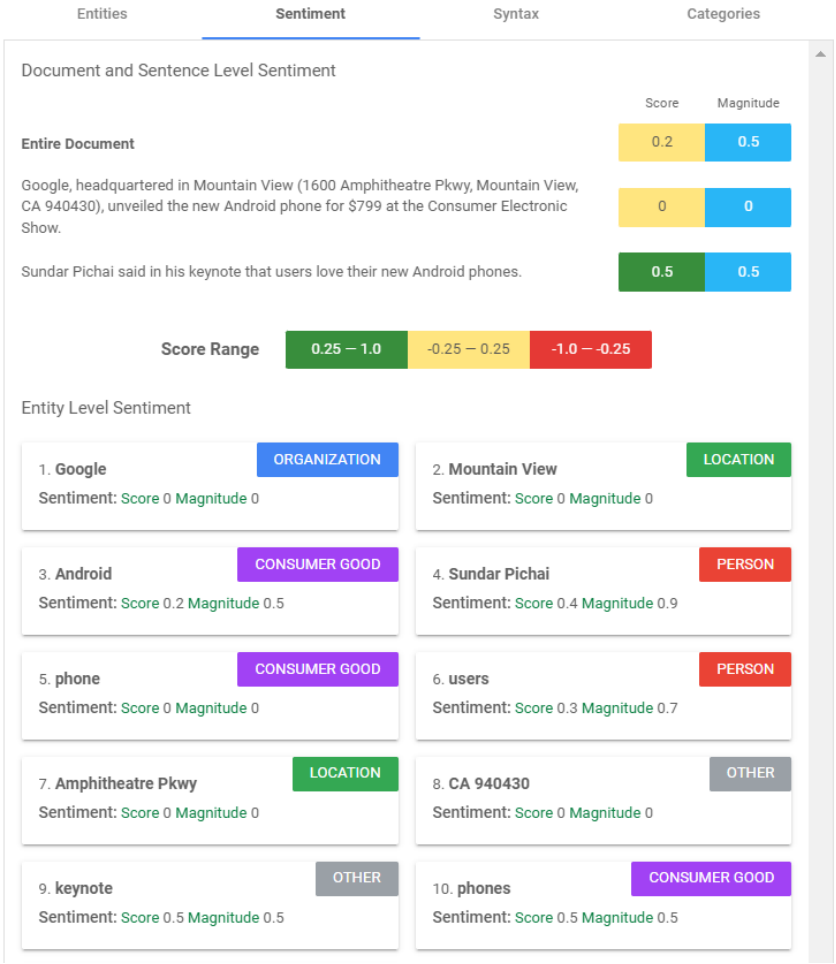


Clasificación temática

Como sucedía con IBM Watson, también se trata de una clasificación jerárquica en varios niveles. Pulsando en el enlace “See a complete list of content categories” se accede a la siguiente página donde se indica que se trata de una clasificación en tres niveles con más de 600 categorías: https://cloud.google.com/natural-language/docs/categories

La pestaña **Sentiment** muestra los resultados del análisis de sentimientos (figura siguiente), tanto a nivel global (“Entire Document”) como a nivel de frase y de entidad, marcados con polaridad entre -1.0 (negativo) y 1.0 (positivo), pasando por 0.0 (neutral).

El texto de ejemplo tiene polaridad global de 0.2 (ligeramente positivo). La polaridad de la primera frase es neutra (0.0), mientras que la segunda frase es positiva (0.5). La polaridad a nivel de entidad se muestra en la parte inferior.



Análisis de sentimientos

Prueba con otros textos y analiza los resultados obtenidos. Por ejemplo, puedes probar con el texto de Wikipedia sobre el pintor Pablo Ruiz Picasso de la unidad 1 del curso.

Ejercicio 4: Entrenamiento de un modelo de clasificación

Duración estimada del ejercicio

40 minutos

1. Introducción

El objetivo de este ejercicio es entrenar un modelo de clasificación basado en *deep learning*, programando en código Python en el entorno de Google Colaboratory.

El primer paso es crear un cuaderno de Colab nuevo, como se indicó en el ejercicio que hiciste en la unidad 2. Como vamos a emplear *deep learning*, asegúrate de que has activado el entorno de ejecución acelerado con GPU, como se indicó en ese mismo ejercicio.

Para resolver la tarea, vamos a emplear un paquete Python llamado Simple Transformers, que ofrece funcionalidad para utilizar modelos basados en la arquitectura de *transformers* de forma sencilla.

Para saber más sobre *Simple Transformers*, puedes pinchar en este enlace.

El primer paso es instalar el paquete de Python en el sistema. Copia el siguiente código en una nueva celda de código y ejecútalo pulsando en el icono de “Reproducir”. Si hay mensajes de aviso de compatibilidad, ignóralos, ya que no van a afectar en este ejercicio.

```
!pip install -q simpletransformers
```

2. Modelo de clasificación de intenciones

El objetivo del ejercicio es entrenar un modelo de clasificación de intenciones, con tres posibles categorías:

0 - solicitud de información

1 - queja o reclamación

2 - recomendación

Ahora es necesario proporcionar ejemplos etiquetados al sistema. Fara facilitar el código, vamos a proporcionar un corpus para entrenamiento y otro para evaluación:

```
corpus_train = [
    ['Quiero información sobre cómo contratar la línea', 0],
    ['Solicito ayuda de vuestros servicios', 0],
    ['Quiero poner una queja', 1],
    ['Voy a hacer una reclamación por el mal servicio', 1],
    ['Este producto es totalmente recomendable', 2],
    ['Me encanta el producto y lo recomiendo', 2],
]

corpus_eval = [
    ['Por favor, quiero ayuda de sobre el uso', 0],
    ['Me gustaría poner una reclamación', 1],
    ['Muy recomendable sin duda', 2],
]
```

Pincha aquí para acceder al código

corpus_train = [

['Quiero información sobre cómo contratar la línea', 0],

['Solicito ayuda de vuestros servicios', 0],

['Quiero poner una queja', 1],

['Voy a hacer una reclamación por el mal servicio', 1],

['Este producto es totalmente recomendable', 2],

['Me encanta el producto y lo recomiendo', 2],

]

corpus_eval = [

['Por favor, quiero ayuda de sobre el uso', 0],

['Me gustaría poner una reclamación', 1],

['Muy recomendable sin duda', 2],

]

Se observa que es un escenario *single-label multiclass*, ya que cada texto solo tiene una única categoría asignada.

	text	labels
0	Quiero información sobre cómo contratar la línea	0
1	Solicito ayuda de vuestros servicios	0
2	Quiero poner una queja	1
3	Voy a hacer una reclamación por el mal servicio	1
4	Este producto es totalmente recomendable	2
5	Me encanta el producto y lo recomiendo	2

Corpus de entrenamiento

3. Entrenamiento

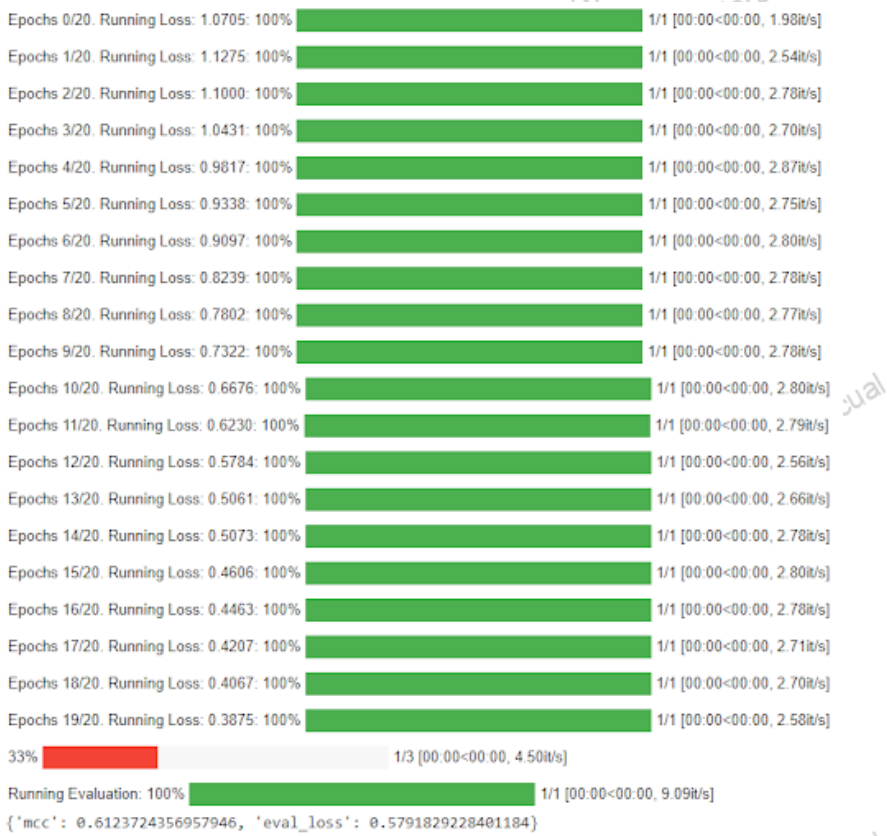
Como modelo base vamos a utilizar “distilbert-base-multilingual-cased”, un modelo de la variante de BERT llamado DistilBERT, que es multilingüe y ha sido entrenado con múltiples idiomas. El parámetro de “num_train_epochs” indica el número de iteraciones de entrenamiento sobre el conjunto de datos, en este caso, 20.

El código en Simple Transformers para entrenar un modelo de clasificación de este tipo es el siguiente. Copia y ejecútalo.

```
from simpletransformers.classification import ClassificationModel, ClassificationArgs

import pandas as pd

import logging


logging.basicConfig(level=logging.WARNING)

logging.getLogger('transformers').setLevel
(logging.WARNING)


# Preparar conjuntos

train_df = pd.DataFrame(corpus_train)

train_df.columns = ["text", "labels"]

eval_df = pd.DataFrame(corpus_eval)

eval_df.columns = ["text", "labels"]


# Preparar el modelo

model_args = ClassificationArgs()

model_args.num_train_epochs = 20

model_args.overwrite_output_dir = True

model = ClassificationModel('distilbert', 'distilbert-base-multilingual-cased', args=model_args,
num_labels=3)


# Entrenar el modelo

model.train_model(train_df)


# Evaluar el modelo

result, outputs, wrong = model.eval_model(eval_df)

print(result)
```

Tras un tiempo de entrenamiento que pueden ser de varios minutos (ignora los mensajes de aviso), la salida muestra la evaluación empleando la métrica MCC (coeficiente de correlación de Matthews) por defecto para la clasificación *multiclass* (en la figura, MCC=0.6123, pero puede que obtengas otro resultado).

IMPORTANTE: Si el sistema da un error por disco lleno, cambia el parámetro "num_train_epochs" a un valor más pequeño (por ejemplo, 10), reinicia el entorno de ejecución (en el menú "Entorno de ejecución" → "Reiniciar entorno de ejecución") y ejecuta de nuevo el código.



Proceso de entrenamiento del modelo

Para saber más sobre *Matthews correlation coefficient*, puedes pinchar en este enlace.

4. Ejecución

Para usar el modelo para clasificar textos nuevos, copia y ejecuta el siguiente código. Observa que al haber utilizado un modelo multilingüe, es posible utilizar textos en diferentes idiomas:

```
corpus_test = ['Lo único que se puede hacer es una reclamación',
               'Qué bueno es esto, lo recomiendo',
               'I want to make a complaint because the service is awful',
               'Je veux déposer une plainte concernant le service']

predictions, outputs = model.predict(corpus_test)
print(predictions)
print(outputs)
```

Pincha aquí para acceder al código

```
corpus_test = ['Lo único que se puede hacer es una reclamación',
               'Qué bueno es esto, lo recomiendo',
               'I want to make a complaint because the service is awful',
               'Je veux déposer une plainte concernant le service']

predictions, outputs = model.predict(corpus_test)
print(predictions)
print(outputs)
```


La siguiente figura muestra los resultados obtenidos:

```
100% ████████████████████████████████████████ 1/1 [00:00<00:00, 6.38it/s]
[1 2 1 1]
[[-0.44995117  0.66943359 -0.18884277]
 [-0.63085938 -0.19226074  0.86621094]
 [-0.33666992  0.68164062 -0.23681641]
 [-0.00386238  0.37133789 -0.05279541]]
```

Salida de los textos de prueba

La primera lista es la de etiquetas asignadas a cada una de las cuatro frases de test (queja, recomendación, queja y queja, respectivamente), que son correctas.

Las líneas siguientes muestran los valores de *score* asignados a cada categoría en cada frase de ejemplo. Así, para la primera frase, la categoría 0 tiene un *score* de -0.44, la categoría 2 de -0.18 y la categoría 1 de 0.66, que es el mayor valor y, por tanto, la categoría predicha.

Si el modelo funciona mal, la solución puede ser ampliar el número de ejemplos del corpus de entrenamiento, o bien aumentar el número de iteraciones de entrenamiento ("epochs"). Sin embargo, al haber escogido ese modelo determinado, la arquitectura y los parámetros vienen dados y no es posible ajustarlos.

Haz diferentes pruebas con diferentes textos y en diferentes idiomas.

Recursos

Enlaces de Interés

- **Matriz de confusión**
  https://es.wikipedia.org/wiki/Matriz_de_confusi%C3%B3n
- **Text Classification with Machine Learning**
  https://monkeylearn.com/text-classification/
- **Emoticon**
  https://en.wikipedia.org/wiki/Emoticon
- **RepTrak.com**
  https://www.reptrak.com/
- **IBM Watson Natural Language Understanding**
  https://www.ibm.com/cloud/watson-natural-language-understanding
- **Google Cloud Natural Language**
  https://cloud.google.com/natural-language
- **Google Colaboratory**
  https://colab.research.google.com/
- **Simple Transformers**
  https://simpletransformers.ai/
- **Matthews correlation coefficient**
  https://en.wikipedia.org/wiki/Matthews_correlation_coefficient
- **IBM Watson Natural Language Understanding**
  https://www.ibm.com/cloud/watson-natural-language-understanding
- **Clasificación jerárquica en varios niveles en IBM Watson Natural Language Understanding**
  https://cloud.ibm.com/docs/natural-language-understanding?topic=natural-language-understanding-categories-hierarchy
- **Google Cloud Natural Language**
  https://cloud.google.com/natural-language
- **Clasificación por niveles en Google Cloud Natural Language**
  https://cloud.google.com/natural-language/docs/categories