



**Escuela Superior
de Ingeniería y Tecnología**
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Clasificación automática de iniciativas parlamentarias

Automatic classification of parliamentary initiatives

Roberto Carlos García Cruz

La Laguna, 12 de julio de 2023

Dña. **Sánchez Nielsen, María Elena**, con N.I.F. 42.848.599-J profesora Titular de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

CERTIFICA (N)

Que la presente memoria titulada:

“Clasificación automática de iniciativas parlamentarias”

ha sido realizada bajo su dirección por D. **Roberto Carlos García Cruz**, con N.I.F. 79.096.303-Q.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 12 de julio de 2023

Agradecimientos

En primer lugar, agradecer a la profesora María Elena Sánchez Nielsen por guiarme, aconsejarme y mostrar toda la atención necesaria para lograr la realización del actual proyecto, me ha parecido una tutora ejemplar.

A mis amigos y pareja, que han sido un pilar muy importante por siempre motivarme y alegrarme el día a día.

A mi familia, por siempre tener en cuenta que podría alcanzar dicha meta y siempre confiar en que podría superar todos estos obstáculos.

Finalmente, el mayor de mis agradecimientos a mi madre y antiguos amigos míos que sin ellos no estaría donde estoy ahora mismo, me han dado la confianza suficiente como para terminar lo que decidí abandonar un día.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.

Resumen

La inteligencia artificial está tomando un papel cada vez más importante en nuestra vida diaria y probablemente en el futuro sea aún más fundamental para nosotros. Una de las ramas del campo de la inteligencia artificial es el aprendizaje automático, dicha rama trata de dotar de la capacidad de aprendizaje a las máquinas para identificar y aprender sobre los patrones, siendo así posible clasificar elementos, realizar predicciones, identificar distintos sujetos en imágenes y un largo etcétera.

El principal objetivo de este trabajo es la clasificación de las iniciativas parlamentarias del Parlamento de Canarias usando aprendizaje automático supervisado. Se utilizará el tesoro multilingüe y multidisciplinario de EuroVoc para la etiquetación de cada una de las diferentes iniciativas parlamentarias en sus correspondientes temáticas.

Se implementarán y evaluarán diferentes algoritmos con el fin de obtener la máxima precisión posible y las condiciones más favorables para cada algoritmo.

Palabras clave: Aprendizaje automático supervisado, clasificación, iniciativa parlamentaria, temática, EuroVoc.

Abstract

Artificial intelligence (AI) is tacking more and more importance in daily lives and it's going to take a fundamental role in a near future. One of the branch of artificial intelligence is the machine learning, which propose is to give the ability to learn to the machines to identify and learn about patrons, giving it the ability to classify elements, do predictions, identify different subjects in pictures, etc.

The main purpose of this project is to classify the parliamentary initiatives of Parliament of Canary Islands using the supervised machine learning. We will use the multilingual and multidisciplinary EuroVoc thesaurus to label each of the different parliamentary initiatives with their corresponding themes.

Different algorithms will be implemented with the objective to obtain the highest possible accuracy and the most favorable conditions for each algorithm.

Keywords: Supervised machine learning, classification, parliamentary initiative, theme, EuroVoc.

Índice general

Capítulo 1	Introducción	1
1.1	Contexto.....	1
1.2	Objetivos.....	2
1.3	Organización.....	2
Capítulo 2	Estado del arte	3
2.1	Situación actual.....	3
2.2	Tipos de aprendizaje	4
2.3	Plataformas que utilizan aprendizaje automático	5
Capítulo 3	Análisis y requisitos	6
3.1	Análisis de requisitos	6
3.1.1	Lenguaje de programación	6
3.1.2	Formato.....	6
3.1.3	Procesamiento del lenguaje natural (NLP).....	7
3.1.4	Aprendizaje automático.....	7
3.2	Herramientas y soluciones.....	8
3.2.1	Lenguaje de programación	8
3.2.2	Formato.....	8
3.2.3	Procesamiento del lenguaje natural (NLP).....	8
3.2.4	Aprendizaje automático.....	8
Capítulo 4	Desarrollo de software	10
4.1	Estructura y diseño del proyecto	10
4.1.1	Modelo.....	10
4.1.2	Vista.....	12
4.1.3	Controlador	13
4.2	Implementación del procesamiento del lenguaje natural.....	14
4.2.1	Procedimiento	14
4.2.2	Lematización o stemming.....	15
4.2.3	Bolsas de palabras	16
4.3	Implementación del aprendizaje automático.....	17
4.3.1	Entrenamiento.....	17
4.3.2	Clasificación	17
Capítulo 5	Guía de usuario	19

5.1	Gestión de conjuntos y selección de algoritmo.....	19
5.2	Entrenamiento.....	21
5.3	Clasificación.....	22
5.4	Cambio de variables y ayuda.....	23
Capítulo 6	Desarrollo de herramienta para el aprendizaje.....	25
6.1	EuroVoc.....	25
6.2	Elaboración.....	26
6.2.1	Criterios para clasificar.....	26
6.3	Resultados.....	27
Capítulo 7	Evaluación de herramientas de inteligencia artificial.....	28
7.1	Desarrollo con ChatGPT.....	28
7.1.1	Sobre aprendizaje automático.....	28
7.1.2	Sobre software y fallos.....	34
7.1.3	Sobre EuroVoc e iniciativas parlamentarias.....	37
7.2	Conclusiones acerca de la herramienta.....	42
Capítulo 8	Evaluación de algoritmos.....	43
8.1	Conceptos.....	43
8.1.1	Matriz de confusión.....	43
8.1.2	Precision.....	44
8.1.3	Recall.....	44
8.1.4	F1.....	44
8.1.5	Accuracy.....	44
8.2	Algoritmos.....	45
8.3	Resultados y conclusiones.....	48
Capítulo 9	Conclusiones y líneas futuras.....	51
9.1	Conclusiones.....	51
9.2	Líneas futuras.....	51
Capítulo 10	Summary and Conclusions.....	52
10.1	Summary.....	52
10.2	Conclusions.....	52
Capítulo 11	Presupuesto.....	54

Índice de figuras

Figura 2.2.1: Ejemplo de aprendizajes	4
Figura 2.3.1: Logotipo de Google.....	5
Figura 2.3.2: Logotipo de Amazon	5
Figura 2.3.3: Logotipo de Facebook.....	5
Figura 3.2.1: Herramientas seleccionadas	9
Figura 4.1.1.1: Diagrama de clases de Modelo	11
Figura 4.1.2.1: Diagrama de clases de Vista.....	12
Figura 4.1.3.1: Diagrama de clases de Controlador	13
Figura 4.2.2.1: Ejemplo de lematización.....	15
Figura 4.2.2.2: Ejemplo de stemming	15
Figura 4.2.2.3: Ejemplo de iniciativas sin NLP	15
Figura 4.2.2.4: Ejemplo de iniciativas con NLP.....	15
Figura 4.2.3.1: Ejemplo de representación de bolsa de palabras.....	16
Figura 4.3.2.1: Ejemplo de clasificación del software (multi-etiqueta)	18
Figura 5.1.1: Ventana principal del programa.....	19
Figura 5.1.2: Opciones de abrir conjuntos	20
Figura 5.1.3: Excepción al abrir conjunto.....	20
Figura 5.1.4: Opciones de mostrar conjuntos	20
Figura 5.1.5: Ventana para mostrar conjunto.....	20
Figura 5.2.1: Opción de entrenamiento.....	21
Figura 5.2.2: Excepción de entrenamiento	21
Figura 5.2.3: Ejemplo de rendimiento del algoritmo	21
Figura 5.3.1: Opción de clasificación	22
Figura 5.3.2: Ejemplo de iniciativas clasificadas por el programa	22
Figura 5.4.1: Opción de configurar variables	23
Figura 5.4.2: Ventana para configurar variables	23
Figura 5.4.3: Ventana que confirma los cambios	24
Figura 5.4.4: Opción de ayuda	24
Figura 5.4.5: Texto de ayuda	24
Figura 6.1.1: Página principal de EuroVoc	25
Figura 6.3.1: Las temáticas de Eurovoc en el conjunto de datos	27
Figura 8.3.1: Resultados de los algoritmos	48

Índice de tablas

Tabla 8.1.1.1: Ejemplo de matriz de confusión.....	43
Tabla 8.1.1.2: Ejemplo de matriz de confusión con valores.....	43
Tabla 8.3.1: Rendimiento de algoritmos ordenados.....	49
Tabla 11.1: Presupuesto de análisis y diseño.....	54
Tabla 11.2: Presupuesto de implementación y desarrollo.....	54

Capítulo 1

Introducción

La tecnología ha avanzado mucho en las últimas décadas, tanto, que muchas de las creencias que se tenían antes de entrar al segundo milenio están llegando a ser una realidad. Información al alcance de nuestra mano, dispositivos inteligentes con capacidad para hacer cualquier tarea o la realidad virtual son elementos que forman parte de nuestro día a día y parece imposible hacerse la idea de vivir sin ellos actualmente.

Este avance ha llegado a tal punto que el mercado de la **inteligencia artificial** (IA) se encuentra en un gran auge, sacando a la luz herramientas con grandes virtudes para resolver ciertas tareas en cuestión de segundos que a un humano puede llevarle horas. Una de las ramas en la cual se está teniendo especial fijación es el **aprendizaje automático**, habilidad que dota a las máquinas para aprender por ellas mismas y en la que se hará especial hincapié a lo largo de este proyecto.

1.1 Contexto

En la actualidad, el aumento exponencial de volúmenes de información ofrece nuevos desafíos para las instituciones parlamentarias. En el caso de los Parlamentos autonómicos, dado el volumen masivo de iniciativas parlamentarias, presenta diferentes desafíos, tales como poder estructurar y clasificar dichas iniciativas. Una iniciativa parlamentaria es un documento por el cual modifica, regula o crea leyes jurídicas.

Por ello, el objetivo final de este Trabajo Final de Grado es el diseño y estudio de técnicas que permitan la clasificación automática de estas iniciativas mediante el concepto de aprendizaje automático previamente comentado, así entendiendo muchos de sus apartados y aprendiendo a usar una herramienta muy útil en los tiempos que corremos junto a otras herramientas de inteligencia artificial.

1.2 Objetivos

El presente trabajo pretende hacer frente a un caso real que principalmente se basa en la creación de un programa informatizado que automatiza la clasificación de las iniciativas parlamentarias del Parlamento de Canarias mediante el análisis de sus extractos. Estas iniciativas serán clasificadas según el tesoro multilingüe y multidisciplinario de **EuroVoc**, un tesoro compuesto por diversas temáticas establecido a los documentos europeos, según el tipo de vocabulario que utilice dicha iniciativa indicará que temática refleja.

Una vez finalizado el desarrollo del software, es de relevante interés hacer un estudio acerca del rendimiento de cada uno de los algoritmos que ofrece el software como punto de apoyo para seleccionar los algoritmos y base de estudio para el proyecto.

Adicionalmente, se propone obtener conclusiones acerca de **ChatGPT**, la herramienta que está a la orden del día por su gran habilidad para poder responder sobre cualquier tema y su capacidad para realizar distintas actividades.

Este análisis se establece con el propósito de determinar si es tan buena herramienta como se expone, abarcándose distintos temas y tratando en muchos casos a ChatGPT como una guía en momentos de dificultad para comprobar su grado de fiabilidad, obviamente con la verificación previa de que la información es correcta.

1.3 Organización

Para realizar un trabajo de manera adecuada es necesario identificar cada uno de los puntos a desarrollar siguiendo un orden:

- **Estado del arte:** Información y situación actual acerca del tema que se quiere hablar.
- **Análisis y requisitos:** Estudio sobre las necesidades más relevantes del proyecto y las posibles soluciones para satisfacerlas.
- **Desarrollo de software:** Diseño y codificación del software con ayuda de las librerías requeridas para satisfacer los objetivos.
- **Desarrollo de herramienta para el aprendizaje:** Creación de una herramienta adecuada según el tipo de aprendizaje utilizado en el proyecto.
- **Evaluación de herramientas de inteligencia artificial:** Estudio de las herramientas de inteligencia artificial, en este caso se evaluará ChatGPT.
- **Evaluación de algoritmos:** Estudio de los algoritmos para especificar la configuración más apropiada y determinar su desempeño.
- **Memoria del proyecto:** Realización de la memoria del proyecto.

Capítulo 2

Estado del arte

En el capítulo anterior se concretó el ámbito del proyecto y sus principales objetivos. Acto seguido, se procede a estudiar la situación actual acerca del aprendizaje automático, algunos de sus usos más recientes e información relevante acerca de esta rama.

2.1 Situación actual

La inteligencia artificial es uno de los temas más en boca de todos en el presente que estamos viviendo debido al gran escándalo generado por las herramientas que se han creado últimamente.

Muchos de estos softwares son capaces de hacer tareas que una persona no imaginaba que una máquina podría hacer, ya sea, crear arte, ediciones e inclusive generar un programa informático. Un ejemplo de ello recientemente ha sido el ganador de un prestigioso concurso de fotografía, el cual afirmó que la imagen ganadora ha sido generada por inteligencia artificial [1].

Este tipo de casos ha generado una gran controversia en todo el mundo, mientras que unos han generado un temor desmesurado por la posible sustitución de cualquier puesto de trabajo, otros entienden que la tecnología debe seguir creciendo y no hay de que temer, puesto que aún no son sujetos tan perfeccionados como para hacer el trabajo de una persona.

Como es de suponer, todas estas inteligencias artificiales incluyen el aprendizaje automático, lo cual les permite aprender, detectar y analizar patrones, mejorando mediante la retroalimentación de casos erróneos o no óptimos.

Sabiendo esto, podemos observar este tipo de herramientas como un arma de doble filo, puesto que al utilizarlas mejorarán hasta tal punto que desconocemos, aunque nos facilitan la vida de una manera muy descabellada.

El aprendizaje automático [2] es una rama muy usada para abarcar una gran diversidad de tareas que utilizamos sin percatarnos, como lo son por ejemplo:

- Búsquedas.
- Medicina.
- Seguridad.
- Clasificaciones.
- Recomendaciones.
- Aparatos inteligentes.

Por eso, cada día que pasa se está mostrando aún más interés por esta rama, ya que presenta una mejora indescriptible en lo que se refiere a la experiencia de usuario, aparte de que en muchos casos da la capacidad de evolucionar a muchos programas o inteligencias artificiales sin necesidad de añadir código, por lo tanto, presenta una gran ventaja y adelanto hacia el futuro.

2.2 Tipos de aprendizaje

El aprendizaje automático contiene múltiples tipos de aprendizaje, siendo los más destacados el **aprendizaje supervisado** y el **aprendizaje no supervisado**.

El **aprendizaje supervisado** requiere de un conjunto de datos etiquetado para predecir un resultado de salida gracias a dicho conjunto, al mismo tiempo, esto da a entender que es necesaria una gran supervisión humana para estos casos, ya que es un especialista quien crea este conjunto de datos etiquetado.

Habitualmente este tipo de aprendizaje es utilizado para realizar clasificaciones y predicciones, a su vez estas pueden devolver un valor real o una etiqueta de un conjunto de posibles clases.

Al contrario que el aprendizaje supervisado, el **aprendizaje no supervisado** no dispone de un conjunto de datos etiquetado, sino que se basa en la capacidad para identificar patrones y determinar aquellos con gran similitud, este aprendizaje es utilizado para simplificar conjuntos de datos.

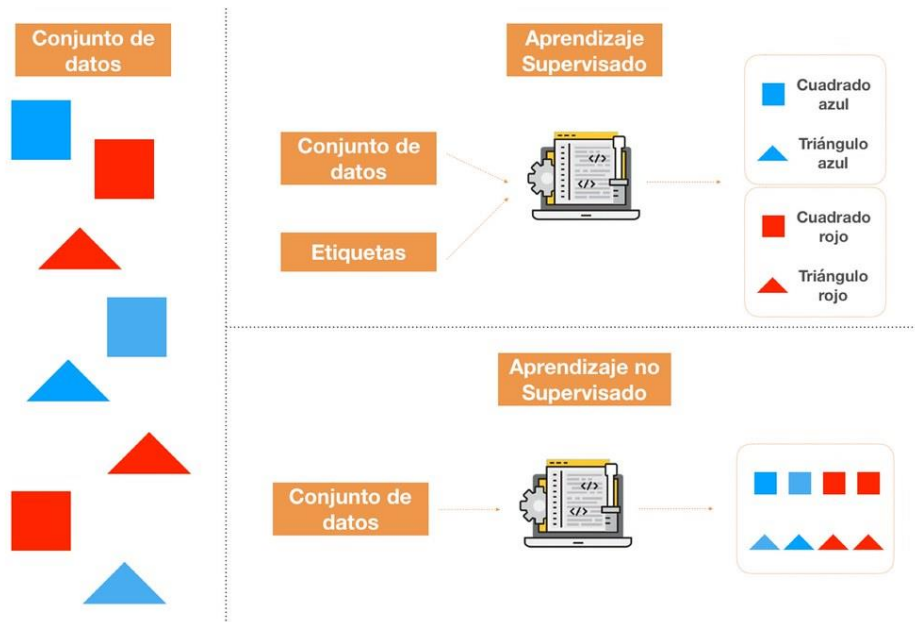


Figura 2.2.1: Ejemplo de aprendizajes

Existen otros dos tipos de aprendizaje aparte de los ya mencionados, el **aprendizaje semi-supervisado** y el **aprendizaje por refuerzo** [3].

El aprendizaje semi-supervisado se trata de una combinación de los dos aprendizajes vistos anteriormente, se obtiene un conjunto de datos etiquetados manualmente en unas proporciones mucho más pequeñas y se entrena un modelo con ese conjunto, el resto de las clasificaciones deseadas serán realizadas por este modelo entrenado.

El aprendizaje por refuerzo basa su sistema de aprendizaje en el ensayo y error, recompensando los comportamientos adecuados y castigando los no adecuados. Suele utilizarse en situaciones con objetivos a largo plazo como los juegos, al principio la máquina cometerá muchos fallos, pero posteriormente será capaz de realizar acciones bastante óptimas gracias a la retroalimentación hecha desde el inicio de la actividad.

2.3 Plataformas que utilizan aprendizaje automático

En los últimos años, muchas de las compañías del sector de la informática reconocidos a nivel mundial han realizado descomunales inversiones en el campo inteligencia artificial para mejorar sus plataformas y productos con la intención de seguir siendo líderes. Evidentemente, una de las ramas de mayor inversión ha sido el aprendizaje automático, con el propósito de sobresalir en cuanto a innovación o mejorar la experiencia de usuario.

En este apartado se mostrarán algunas de esas empresas que utilizan el aprendizaje automático en sus servicios y para qué lo utilizan.



Figura 2.3.1: Logotipo de Google

En primer lugar, nos encontramos con **Google** [4], la famosa empresa de búsqueda hace gran uso de este método en mucho de sus servicios. Ejemplos de ello son: mejorar su motor de búsqueda, procesar el lenguaje natural mediante ciertas consultas de voz en Google Home, mejorar la búsqueda de Google Maps y Youtube mediante el historial del usuario, deducir correos de spam en Gmail y un largo etc. De manera resumida, la mayor parte de las interacciones que se tiene con Google son incitadas por el aprendizaje automático.

Amazon [5] no se queda atrás respecto al uso del aprendizaje automático. Una gran mejora en las búsquedas para dar con el artículo adecuado, recomendaciones personalizadas, procesamiento del lenguaje y reconocimiento de voz, son los métodos más valorados de Amazon, siendo capaz de demostrárnoslo mediante uno de sus artículos más famosos, **Alexa**.



Figura 2.3.2: Logotipo de Amazon

Para finalizar, tenemos a **Facebook** [6], la famosa red social enfoca muchas de sus acciones con aprendizaje automático. Publicidad individualizada, contenido específico para cada usuario, clasificación automática de imágenes o un analizador de texto como ninguno en comparación son algunos ejemplos que nos puede ofrecer su plataforma en dicha materia.



Figura 2.3.3: Logotipo de Facebook

Estos han sido algunos ejemplos, obviamente quedan un gran conjunto de empresas que no han sido mencionadas. Como se puede observar, todas las grandes empresas desde hace un tiempo están invirtiendo grandes cantidades de dinero en la inteligencia artificial para perfeccionar todos sus métodos y atraer el máximo número de usuarios posible. Es de esperar que de aquí en adelante el propósito de muchas empresas sea este.

Capítulo 3

Análisis y requisitos

Antes de iniciar cualquier actividad asociada a la implementación o diseño del proyecto, es imprescindible determinar cuales son las necesidades más relevantes de este y como van a satisfacerse estos requerimientos. Hagamos memoria, el proyecto propone obtener resultados mediante el análisis del extracto de cada iniciativa parlamentaria, por lo tanto, al menos es necesario tener en cuenta cuatro componentes indispensables: el lenguaje de programación, el formato en el que se trabajará, el procesamiento del lenguaje natural y herramientas de aprendizaje automático.

3.1 Análisis de requisitos

Para determinar que herramientas se ajustan mejor al proyecto es preciso determinar que acciones van a realizarse a lo largo del mismo con el propósito de ahorrarnos posibles problemas en un futuro.

3.1.1 Lenguaje de programación

Uno de los elementos más importantes en todo proyecto es la selección del lenguaje de programación, según el que se escoja puede ofrecernos diferentes facilidades y dificultades, por ello, se debe concretar la situación a la que se va a enfrentar para determinar las preferencias.

El lenguaje de programación debe ser uno que proporcione las siguientes características:

- Facilidades en el campo de la inteligencia artificial.
- Facilidad de uso y aprendizaje.
- Gran comunidad.
- Multitud de tipo de librerías.

3.1.2 Formato

Para obtener los extractos y temáticas de cada una de las iniciativas parlamentarias, se debe especificar el tipo de archivo con el que podrá trabajar el programa.

Se puede escoger entre un gran conjunto de tipos de archivo, siendo los más recomendables para estos casos los formatos **JSON** y **CSV**, ya que son los tipos más comunes a la hora de trabajar. Además, cualquier documento subido por un órgano legislativo suele ofrecer como mínimo estos formatos mencionados.

No hay que olvidarse de la complejidad que puede suponer trabajar con algún formato, por ejemplo, leer y escribir documentos **XML** es mucho más complicado frente a la simpleza que presenta hacerlo en CSV.

3.1.3 Procesamiento del lenguaje natural (NLP)

El **procesamiento del lenguaje natural** [7] es una tecnología de aprendizaje automático que permite al computador interpretar y entender el lenguaje humano. Para analizar los distintos extractos que nos encontraremos es muy relevante una buena herramienta para el procesamiento del lenguaje natural, probablemente este sea el proceso más importante del proyecto, a causa de que la salida de este procedimiento afectará en mayor medida al resultado final.

Es de extrema importancia realizar un buen procesamiento del lenguaje, para ello, hay que tener en cuenta los métodos indispensables que se realizarán:

- Tokenización.
- Eliminación de palabras vacías y repeticiones.
- Stemming/lematización.
- Bolsa de palabras.

3.1.4 Aprendizaje automático

Por la parte del aprendizaje automático, es necesario una herramienta lo suficientemente completa que nos permita trabajar con una gran cantidad de algoritmos y contenga los métodos más fundamentales de la misma materia, los cuales deberían ser:

- Entrenamiento de algoritmos.
- Separación de datos de prueba y datos entrenamiento dentro de un mismo conjunto de datos.
- Obtención de resultados del entrenamiento de algoritmos.
- Clasificaciones mediante el algoritmo previamente entrenado.

No obstante, para poder trabajar con cualquier herramienta de aprendizaje automático hay que tener en cuenta una pregunta muy importante, ¿de qué manera sabrá nuestro software como debe realizar las clasificaciones?

Como ya se ha comentado previamente en el estado del arte, existen varios tipos de aprendizaje que se pueden aplicar, entendiendo las diferencias de cada uno de ellos, se debe seleccionar el tipo de aprendizaje más adecuado para este caso.

3.2 Herramientas y soluciones

Tras haber estudiado los componentes más importantes que necesita nuestro proyecto, se debe elegir la opción más apropiada al alcance de nuestra mano. A continuación se procederá directamente a indicar la herramienta seleccionada junto a su justificación.

3.2.1 Lenguaje de programación

Como lenguaje de programación se ha optado por **Python** [8], lenguaje con una sintaxis muy sencilla de aprender que es apoyado por una gran comunidad, además de ofrecer una cantidad inimaginable de librerías que pueden suplir muchas de nuestras necesidades. Entre ellas, se encuentra una gran variedad de librerías para procesar el lenguaje natural y realizar tareas de aprendizaje automático, sin duda es el lenguaje de programación ideal para esta situación.

3.2.2 Formato

Debido a su facilidad para trabajar en Python mediante la librería **Pandas** [9] y su extenso uso en múltiples plataformas, el tipo de archivo con el que se trabajará será CSV.

3.2.3 Procesamiento del lenguaje natural (NLP)

En el procesamiento del lenguaje natural existe una gran variedad de librerías que pueden sernos de gran ayuda, la más conocida de ellas es **Nltk** [10], al ser una herramienta muy fácil de usar y ofrecer muchas posibilidades.

Pero Nltk no es adecuado para nosotros por una simple razón, esta herramienta no es capaz de trabajar en todas sus funciones en el idioma español, sin ir más lejos, su lematizador no tiene la capacidad de funcionar en dicho idioma.

Al estar ante iniciativas parlamentarias en español, se ha escogido una librería que permite trabajar en todas sus funciones en una gran multitud de idiomas gracias a sus diversos modelos, esa librería es **Spacy** [11]. Spacy satisface todo lo pedido y es bastante sencilla de utilizar, pudiendo ser fácilmente igual de potente que Nltk.

3.2.4 Aprendizaje automático

Para el caso del aprendizaje automático, se encuentra **Scikit-learn** [12], librería por excelencia para cualquier situación en relación con esta rama. Scikit-learn ofrece todos los métodos requeridos comentados en el análisis previo junto a una gran cantidad de algoritmos de aprendizaje automático para poder escoger libremente el que mejor convenga.

El tipo de aprendizaje que se empleará es el aprendizaje supervisado, por lo tanto, se debe construir un conjunto de entrenamiento con iniciativas parlamentarias del Parlamento de Canarias, clasificándolas manualmente según nuestro criterio.

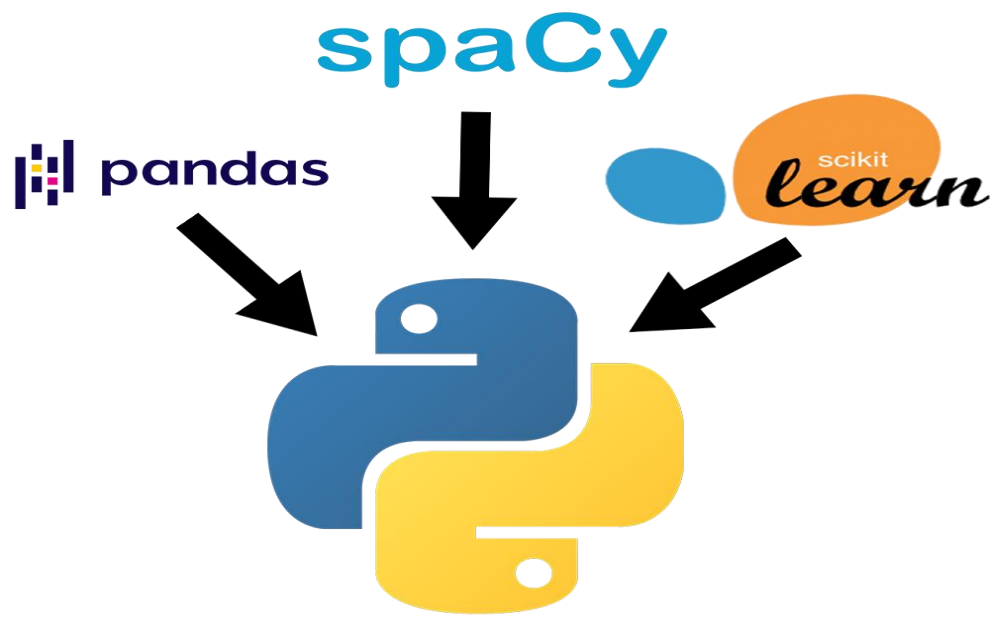


Figura 3.2.1: Herramientas seleccionadas

Capítulo 4

Desarrollo de software

Resueltas todas las necesidades del proyecto mediante el análisis, se da inicio al desarrollo del proyecto para poder clasificar automáticamente las iniciativas parlamentarias.

4.1 Estructura y diseño del proyecto

Para mostrar cierta parte del desarrollo de la aplicación sin entrar en muchos detalles, se procede a exponer y comentar la estructura que se ha seguido desde un inicio en el trabajo. Se harán abreviaciones sobre cada una de las clases y relaciones entre ellas.

De primera mano, se puede observar que la estructura principal del proyecto está basada en el famoso patrón estructural **MVC** [13], por lo tanto, está dividido en tres partes principales: **Modelo**, **Vista** y **Controlador**.

4.1.1 Modelo

El Modelo se encarga de todas aquellas clases que estructuran la información del software y el manejo de los datos.

El paquete del Modelo contiene 8 clases:

- **Modelo**: Clase que será utilizada para comunicarse con el Controlador cediéndole los métodos más relevantes para manejar y controlar la información.
- **Clasificador, ClasificadorML y ClasificadorAR**: Realizan los entrenamientos y clasificaciones pertinentes. Las clases “ClasificadorML” y “ClasificadorAR” están unidas mediante un patrón **método plantilla** [14] a la clase Clasificador.

Muchos de los algoritmos requieren como entrada unos datos estándar, estos algoritmos son iniciados por medio de la clase “ClasificadorML”, mientras que otros necesitan una transformación a array, este el caso de los algoritmos que son iniciados con la clase “ClasificadorAR”. Como se ve, es un diseño preparado para la extensión por si algún algoritmo requiere de otro tipo de dato de entrada.

En todas estas clases se ha hecho uso de la librería Scikit-learn.

- **Lector y LectorCSV**: Se encargan de leer el contenido de los archivos, la clase “Lector” es la clase padre de la clase “LectorCSV”, la cual obviamente solo lee archivos de tipo CSV. Se trata de una estructura pensada para extender a múltiples tipos de lectura.
- **Dataset**: Clase que contiene los métodos propios para el manejo de un conjunto de datos.
- **Procesador**: Clase utilizada para el procesamiento del lenguaje natural haciendo uso de la librería Spacy.

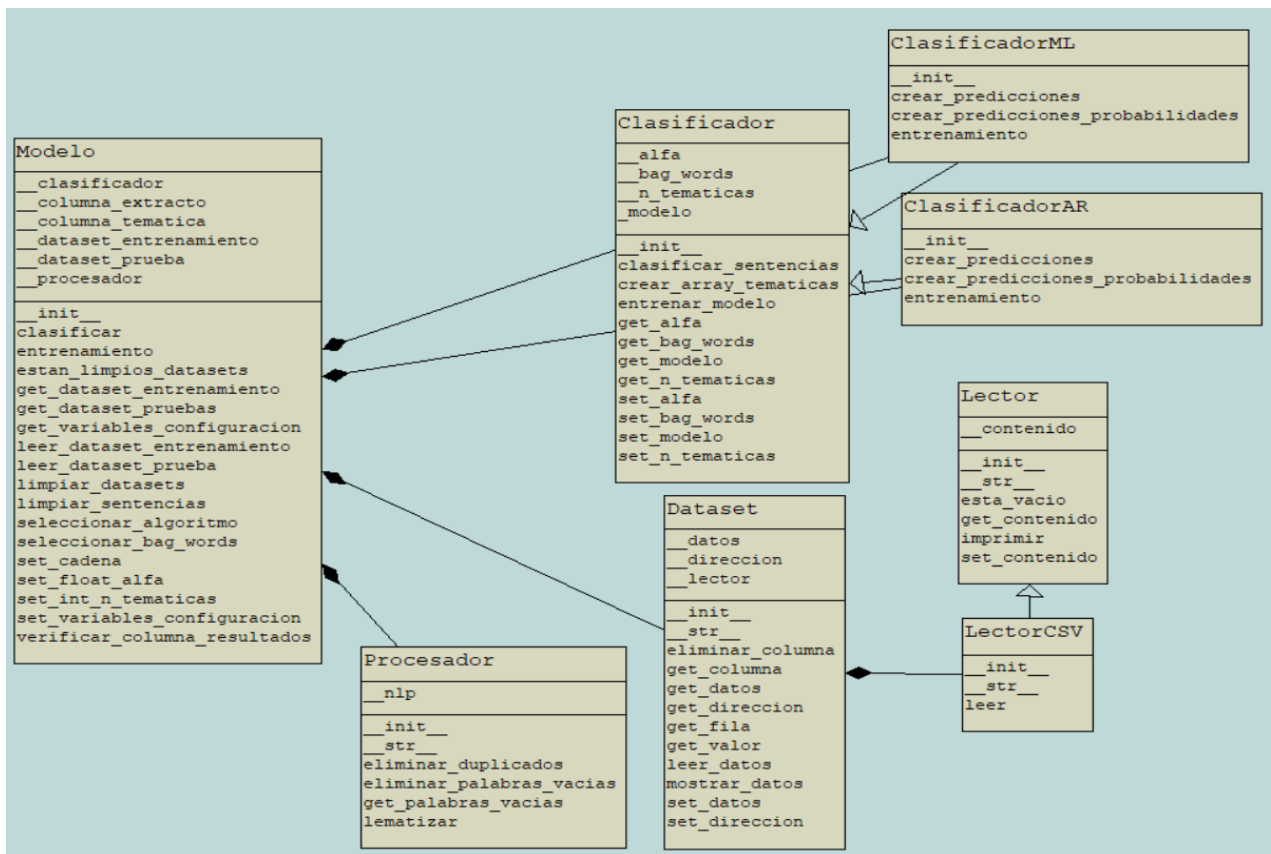


Figura 4.1.1.1: Diagrama de clases de Modelo

4.1.2 Vista

La Vista se encarga de ofrecer todo lo relacionado con el entorno visual de la aplicación, para que el programa cumpla con la función del patrón MVC correctamente, es necesario transmitir muchos de los elementos de la Vista al Controlador (botones, combos, entradas, etc.) con el propósito de que el funcionamiento de cada uno de estos elementos pueda ser manejado.

El paquete Vista contiene está formado por 3 clases:

- **Vista:** Clase que será utilizada para comunicarse con el Controlador mediante los métodos necesarios, esencialmente aquellos que permiten obtener los elementos visuales para que el Controlador haga su trabajo.
- **App:** Representa el conjunto de todas las ventanas y menús que aparecen a lo largo del programa.
- **MainFrame:** Su función es ser el depósito del contenido de la ventana principal.

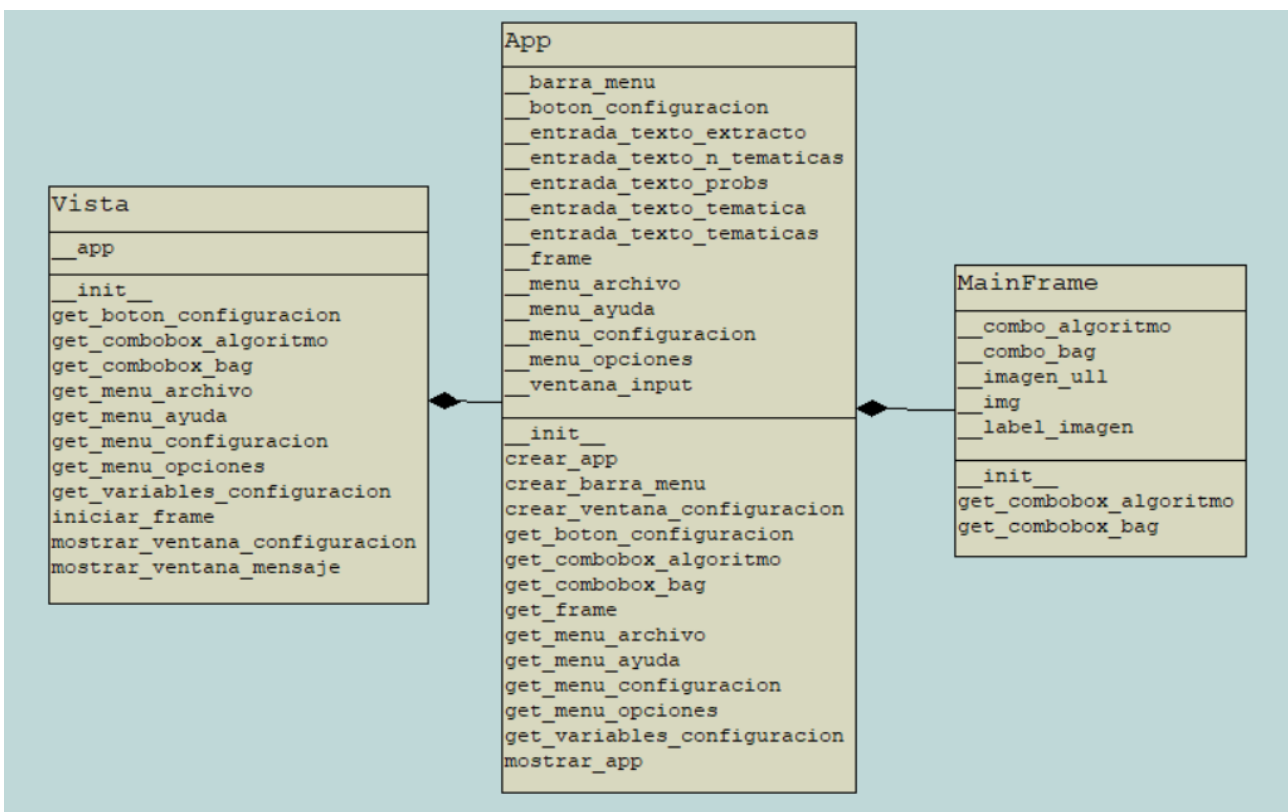


Figura 4.1.2.1: Diagrama de clases de Vista

4.1.3 Controlador

El Controlador es la parte más esencial de dicho patrón, se encarga de ser el intermediario entre el usuario y el programa, al mismo tiempo que une el Modelo y la Vista. Básicamente, es el que se encarga del manejo de la aplicación en todos los sentidos.

Únicamente se encuentra una clase en el paquete Controlador, la misma clase “Controlador”, mediante la unión de las clases “Modelo” y “Vista”, utiliza los métodos más relevantes de ambas clases para cohesionar el programa.



Figura 4.1.3.1: Diagrama de clases de Controlador

4.2 Implementación del procesamiento del lenguaje natural

A partir del apartado actual, se explicará la implementación de las partes más interesantes del proyecto, empezando por el procesamiento del lenguaje natural.

4.2.1 Procedimiento

Para poder llevar a cabo clasificaciones sobre un texto, es de extrema importancia realizar una buena examinación y limpieza del mismo, con el fin de sacar ciertas conclusiones acerca del tipo al que más se asemeja.

Es esencial reducir el número de palabras de un texto que se quiera estudiar para poder clasificarlo y facilitar el trabajo a los algoritmos. El resultado que se desea obtener es un conjunto de palabras clave que nos ayudará a determinar de que tipo se trata.

Para que la salida del procesamiento del lenguaje sea satisfactoria, deben llevarse a cabo ciertas etapas fundamentales en el siguiente orden [15]:

- **Tokenización:** Trata de transformar frases, palabras o secuencias de caracteres (dependiendo mucho del tipo de problema) en una unidad de procesamiento para el computador, a esta unidad se le denomina **token**, en este caso, lo más recomendable es que cada palabra sea un token.
- **Eliminación de palabras vacías y puntuaciones:** En todos los lenguajes hay palabras que no aportan una gran relevancia para determinar el contexto de una frase, hablamos de las **palabras vacías**. En el castellano son elementos como: artículos, pronombres, preposiciones, etc. Estas palabras aun quitándolas podemos entender el contexto de la frase, esto junto a los signos de puntuación es preciso que sea eliminado de los textos para determinar cuales son las palabras que más describen su tipo.
- **Eliminación de duplicaciones:** Al realizar las etapas anteriores, como resultado se obtendrán un conjunto tokens de supuesta relevancia en el contexto del texto, algunos de estos tokens es posible que puedan ser repetidos. Para mejorar un poco el rendimiento y quitar cierta carga de trabajo a los algoritmos de aprendizaje, se eliminan dichos tokens repetidos.
- **Lematización:** Lematizar es la transformación de una palabra a su forma base, es decir, las palabras pueden tener distintos tiempos conjugados, plurales, singulares, etc. Para llegar a un estándar, es necesario poner todas esas palabras a su lema (forma base).
- **Bolsa de palabras (BOW):** La bolsa de palabras es una técnica que crea un vector de las distintas palabras existentes en un texto en formato de diccionario. Este vector almacena cada una las palabras y su frecuencia sin importar el orden o el contexto de la frase con la idea de obtener ciertas conclusiones basándose en estas frecuencias.

4.2.2 Lematización o stemming

Normalmente, en estos casos suele tomarse la decisión de realizar la lematización o bien el **stemming**, esta última pretende recortar cada palabra y quedarse solo con aquella parte que no cambia (stemm).

Palabra	Lema
<i>Habló</i>	<i>Hablar</i>
<i>Hablarán</i>	<i>Hablar</i>
<i>Manuel</i>	<i>Manuel</i>
<i>Libros</i>	<i>Libro</i>
<i>Libro</i>	<i>Libro</i>

Figura 4.2.2.1: Ejemplo de lematización.

Palabra	stemm
<i>Libertad</i>	<i>Lib</i>
<i>Libre</i>	<i>Lib</i>
<i>Libro</i>	<i>Lib</i>
<i>Hablarán</i>	<i>Habl</i>
<i>Habló</i>	<i>Habl</i>

Figura 4.2.2.2: Ejemplo de stemming.

Ambas presentan ciertos pros y contras, mientras que para textos grandes es utilizado el stemming, debido a que suele ser más rápido que los algoritmos de lematización y podemos ver relaciones con ciertas palabras de manera más sencilla, presenta otras características negativas como puede ser la pérdida del contexto, ya que solo tenemos las raíces de las palabras y eso a veces puede llevar a resultados no del todo correctos.

La lematización, en cambio, es utilizada en textos más pequeños debido a que suele requerir más, pero no pierde tanto contexto de las sentencias como el stemming.

En este proyecto se ha optado por utilizar la lematización, al tratarse de extractos de iniciativas parlamentarias podemos permitirnos utilizar esta técnica para perder el mínimo contexto posible [16].

De la presidencia y del gobierno de canarias.
De políticas de juventud de canarias.
De presupuestos generales de la comunidad autónoma de canarias para 2023.

Figura 4.2.2.3: Ejemplo de iniciativas sin NLP.

['presidencia', 'gobierno', 'canarias']
['política', 'juventud', 'canarias']
['presupuesto', 'general', 'comunidad', 'autónomo', 'canarias', '2023']

Figura 4.2.2.4: Ejemplo de iniciativas con NLP.

Como se puede apreciar en las imágenes, podemos entender de que tratan las iniciativas parlamentarias simplemente con sus palabras clave.

4.2.3 Bolsas de palabras

Hay que tener en cuenta que no se está ante un problema donde se obtengan resultados de datos numéricos, hay que obtener resultados de un conjunto de palabras, si los datos fueran numéricos, con unas simples fórmulas se puede tener un resultado que el ordenador puede reconocer, pero este caso es diferente, ya que debemos sacar resultados de un conjunto de palabras clave. Para que el ordenador pueda obtener un resultado que reconozca a través de palabras es necesario añadir todos estos términos clave en una bolsa de palabras.

La bolsa de palabras definida anteriormente es la solución para este problema, debido a que se encarga de simplificar el contenido tratando de enfocarse en las palabras y el número de veces que se repiten sin importar su orden.

Existe una gran variedad de modelos de bolsas de palabras, pero en este caso, solamente nos vamos a enfocar en los dos que se han incluido en el software, que son los que más relevancia presentan:

- **CountVectorizer**: Es el modelo genérico de la bolsa de palabras, este tipo de modelo simplemente realiza las acciones ya comentadas previamente. Es decir, crear un vector de palabras y contar su frecuencia.
- **TFIDFVectorizer** [17]: Es una extensión mejorada del countvectorizer que tiene como adición entender como de importante es una palabra dentro de un conjunto de textos. Esto se hace observando si una palabra se repite mucho en un texto en comparación al resto, si una palabra se repite mucho en un texto concreto, pero poco en el **corpus**, ese texto que tenga dicha palabra tendrá cierta relevancia a la hora de buscar por ese término, en cambio, si una palabra se repite mucho en el corpus, no es una palabra que nos facilite la búsqueda y no es nada relevante tenerla en cuenta.

	Gato	Gusta	Mordio	Perro	Queso	Raton	
El perro mordió al gato	0	1	0	1	1	0	0
El gato mordió al ratón	1	1	0	1	0	0	1
Al ratón le gusta el queso del gato	2	1	1	0	0	1	1

Figura 4.2.3.1: Ejemplo de representación de bolsa de palabras

4.3 Implementación del aprendizaje automático

Llevado a cabo todo lo relacionado con el procesamiento del lenguaje natural, es el momento de ajustar todos los métodos necesarios para las clasificaciones de las iniciativas parlamentarias mediante la librería Scikit-learn.

4.3.1 Entrenamiento

En el apartado anterior, explicamos todo el procedimiento hasta obtener la bolsa de palabras. Este resultado obtenido de todo el procesamiento del lenguaje es el elemento que los algoritmos necesitan para poder hacer sus respectivas operaciones y devolvernos el desempeño que muestra el algoritmo.

El programa con el que trabajamos contiene dos tipos de conjuntos, el conjunto de entrenamiento, utilizado para entrenar los algoritmos, y el conjunto de pruebas, documentos que contienen iniciativas parlamentarias para que sean clasificadas tras haber realizado el entrenamiento del algoritmo.

Obviamente, el algoritmo no obtendrá un criterio propio una vez consuma la información que se le proporciona, sino que tendrá un criterio similar a la persona que haya clasificado esos datos.

En el proceso de entrenamiento, el conjunto de entrenamiento que se le suministra al programa se divide en los siguientes tipos de dato:

- **Datos de entrenamiento**, los cuales estudiará el algoritmo para intentar establecer los patrones para clasificar cada elemento, este tipo de dato suele rondar el 70% - 80% del conjunto.
- **Datos de prueba** [18], los datos restantes que no han sido seleccionados como datos de entrenamiento son utilizados para que sean clasificados por el algoritmo entrenado y así poder verificar su resolución, de estas clasificaciones es de donde se obtienen los resultados acerca del rendimiento del algoritmo.

Tras este proceso se informa al usuario sobre ciertos atributos que determinan el desempeño del algoritmo en unidad de proporción (porcentaje), estos atributos serán comentados y definidos en apartados posteriores.

4.3.2 Clasificación

El método de clasificación desarrollado no solo es capaz de hallar una única temática para cada iniciativa, sino que puede devolver un conjunto de temáticas según el número de temáticas que se desee mostrar y la cercanía de la temática más probable respecto al resto, a este tipo de clasificación se le denomina clasificación multi-etiqueta [19].

Para poder hacer tal procedimiento es necesario que todos los algoritmos tengan activado un parámetro que devuelve la probabilidad de cada temática, para así poder trabajar con todas esas probabilidades.

Si se quiere que más de una temática pueda ser mostrada, no basta solo con que modifiquemos el número de temáticas que se puedan ver, sino que debe de haber una cercanía inferior a un umbral concreto respecto a la temática con más probabilidades.

La aplicación dispone de este modelo de clasificación debido a que habitualmente las iniciativas parlamentarias suelen contener en su vocabulario más de una temática, siendo posible que no solo pertenezca a una. El usuario puede modificar tanto el número de temáticas a mostrar como el umbral de diferencia de probabilidades entre las distintas temáticas.

Umbral: 0.15

Nº temáticas: 3

PROBABILIDADES

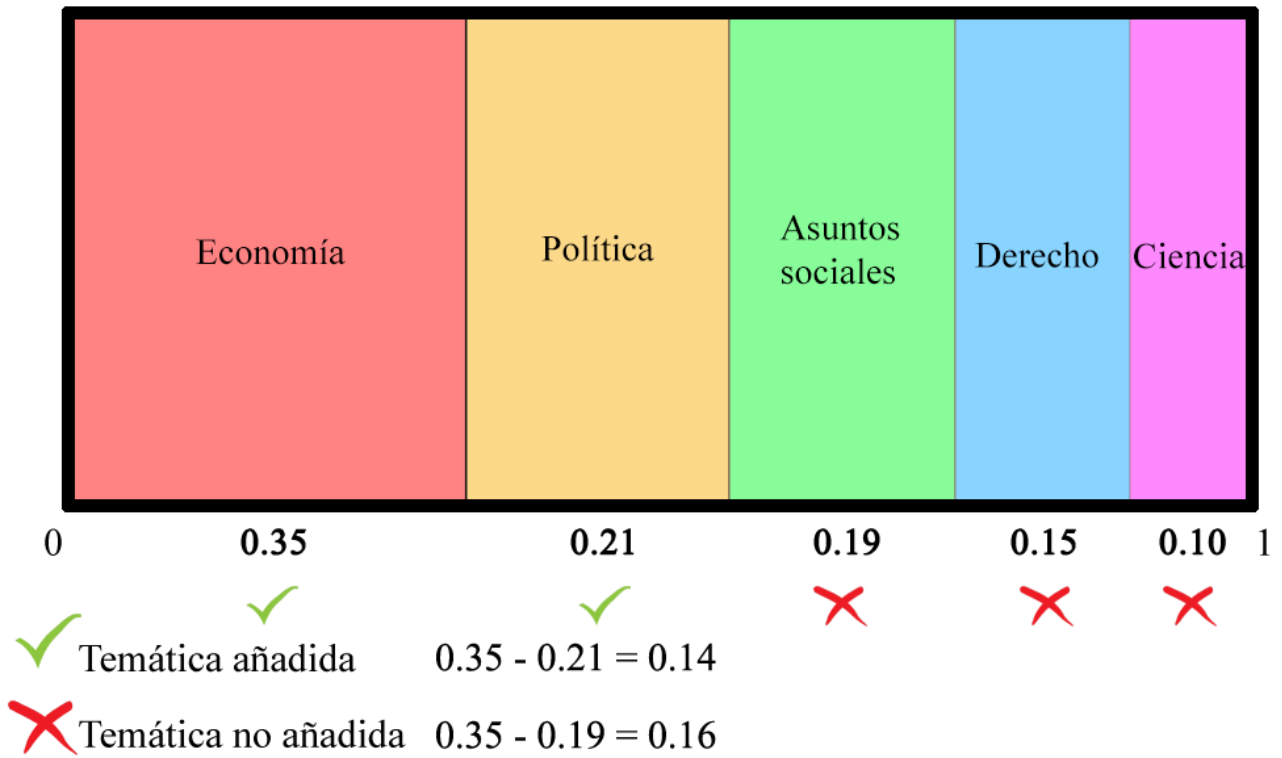


Figura 4.3.2.1: Ejemplo de clasificación del software (multi-etiqueta)

Se procede a dejar el GitHub del proyecto: <https://github.com/robersharkY/TFG-023-Machine-Learning>

Capítulo 5

Guía de usuario

En el siguiente apartado, a modo de guía, se mostrará el funcionamiento del programa y los pasos necesarios para realizar cualquier acción que se desee.

El programa desarrollado presenta las siguientes funcionalidades:

- Lectura de conjunto de entrenamiento y pruebas.
- Mostrar contenido de los conjuntos.
- Selección de algoritmos y bolsa de palabras.
- Entrenamiento de algoritmos mediante el conjunto de entrenamiento previamente leído.
- Clasificación del conjunto de pruebas mediante el algoritmo entrenado anteriormente.
- Modificación de ciertas variables a la hora de entrenar, clasificar y leer los conjuntos.
- Apartado de ayuda que explica el proceso habitual a seguir.

5.1 Gestión de conjuntos y selección de algoritmo

Al iniciar el programa, lo primero que nos aparece es la ventana principal, esta ventana está compuesta por un menú superior que contiene múltiples opciones y unos combobox que permite la selección de la bolsa de palabras y el algoritmo de aprendizaje automático.

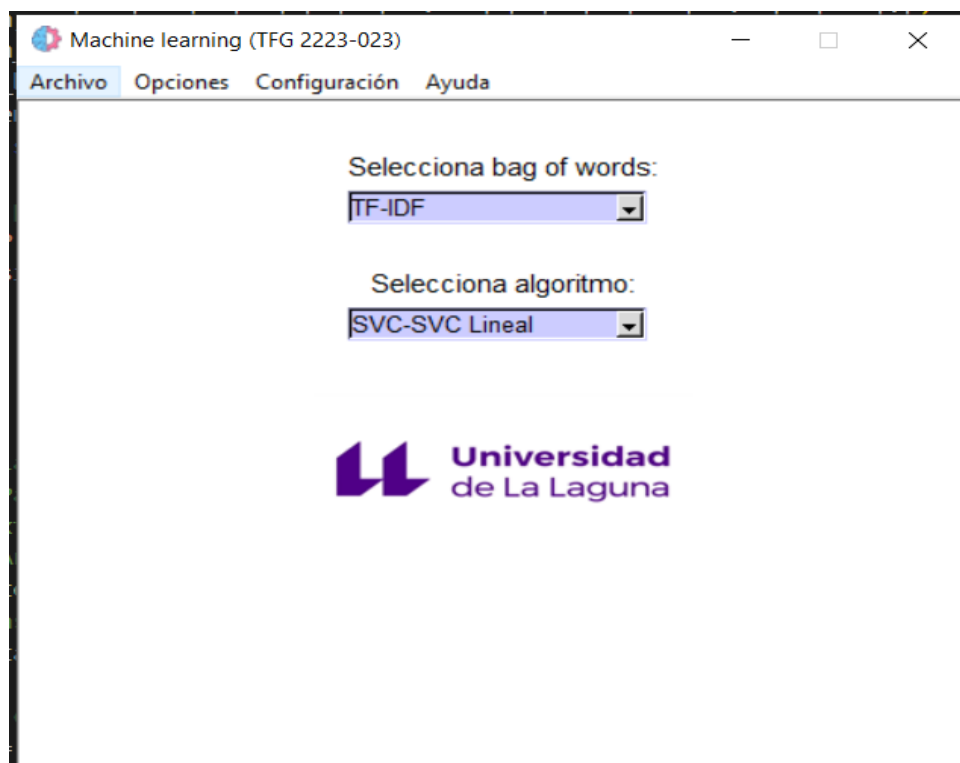


Figura 5.1.1: Ventana principal del programa

Tanto el algoritmo como la bolsa de palabras pueden cambiarse en cualquier momento durante la ejecución del programa.

Para empezar a dar uso al programa, es necesario cederle un conjunto de pruebas y un conjunto de entrenamiento, para añadir cualquiera de los dos debe dirigirse hacia **Archivo > Abrir conjunto de pruebas/entrenamiento**. Tras ejecutar dicha opción aparecerá una ventana para seleccionar un archivo, se debe seleccionar solo un archivo, ya que si no saltará una excepción.

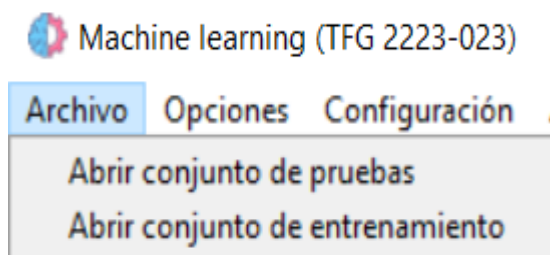


Figura 5.1.2: Opciones de abrir conjuntos

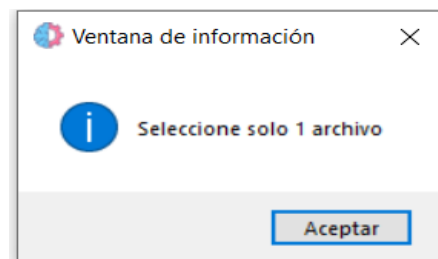


Figura 5.1.3: Excepción al abrir conjunto

Es posible mostrar el contenido de estos conjuntos en **Opciones > Mostrar conjunto de pruebas/entrenamiento**, al pulsar, se abrirá una nueva ventana en formato de tabla mostrando los datos que contiene dicho conjunto.

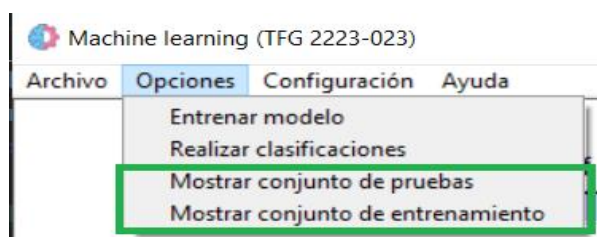


Figura 5.1.4: Opciones de mostrar conjuntos

	ID_INICIATIVA	LEGISLA	EXTRACTO
1	10L/PL-0019	10	De la renta de ciudadanía de canarias.
2	10L/PL-0020	10	Por el que se modifica la ley 4/2017, de 13 de julio, del suelo y de los espacios naturales protegidos de canarias,
3	10L/PL-0022	10	De la presidencia y del gobierno de canarias.
4	10L/PL-0021	10	De políticas de juventud de canarias.
5	10L/PL-0023	10	De presupuestos generales de la comunidad autónoma de canarias para 2023.

Figura 5.1.5: Ventana para mostrar conjunto

5.2 Entrenamiento

Leídos ambos archivos, se puede proceder a entrenar el algoritmo que se ha escogido seleccionando **Opciones > Entrenar modelo**, en caso de faltar algún archivo, es decir, el conjunto de entrenamiento o el conjunto de pruebas, una excepción saltará pidiendo que añadamos el archivo que falta.

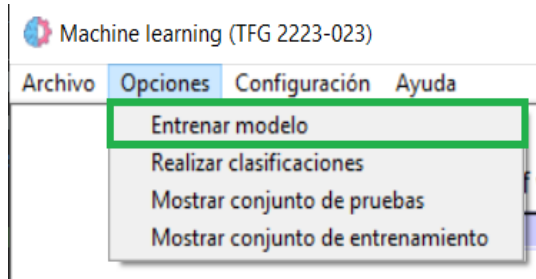


Figura 5.2.1: Opción de entrenamiento

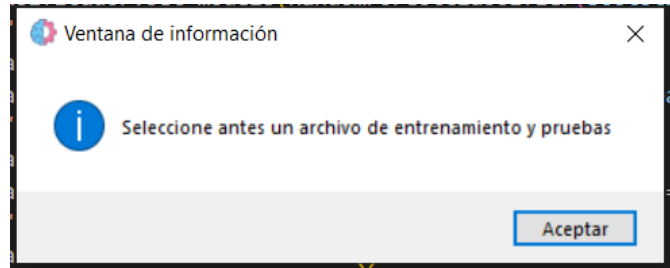


Figura 5.2.2: Excepción de entrenamiento

Este proceso puede tardar un poco dependiendo de la longitud del conjunto de entrenamiento y el algoritmo seleccionado, puesto que en esta funcionalidad se realiza el procesamiento del lenguaje natural y posteriormente el entrenamiento del algoritmo, las acciones que más recursos requieren del programa.

Al acabar el entrenamiento se nos mostrará una pequeña ventana que nos indica el rendimiento del algoritmo mediante ciertas variables como: **accuracy**, **precision**, **recall** y **F1**.

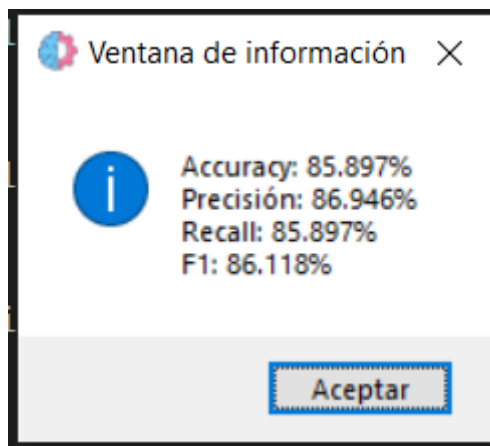


Figura 5.2.3: Ejemplo de rendimiento del algoritmo

5.3 Clasificación

Una vez realizado el entrenamiento, es posible realizar las clasificaciones deseadas (iniciativas del conjunto de pruebas), para que esto ocurra se debe seleccionar **Opciones > Realizar clasificaciones**. Hay que tener en cuenta que siempre que se quiera efectuar las clasificaciones es preciso llevar a cabo un entrenamiento previo del algoritmo, de no ser así saltará una excepción indicándose.

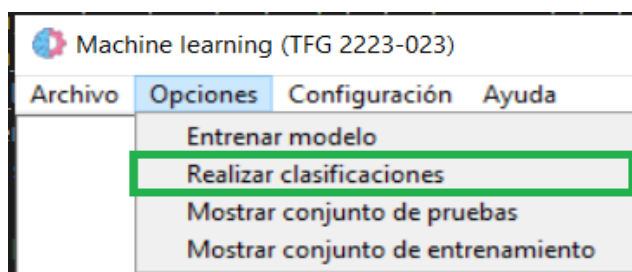
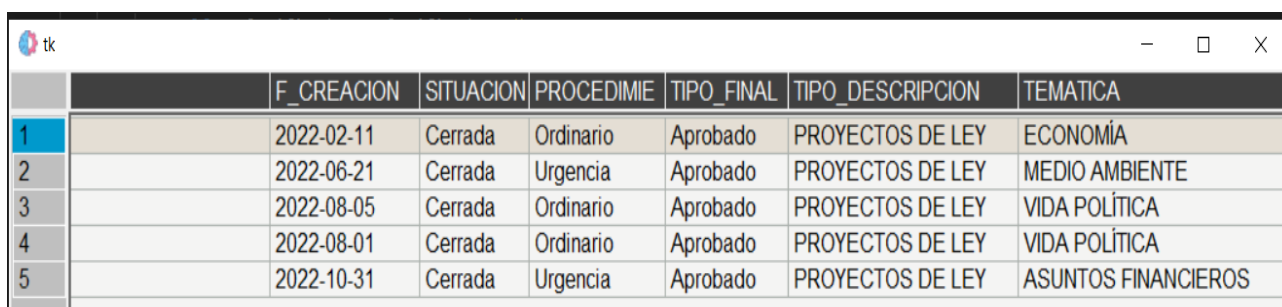


Figura 5.3.1: Opción de clasificación

Al pulsar dicha opción aparecerá una ventana que muestra el conjunto de pruebas con una nueva columna añadida denominada "TEMATICA", esta columna indica las posibles temáticas de la iniciativa.



	F_CREACION	SITUACION	PROCEDIMIE	TIPO_FINAL	TIPO_DESCRIPCION	TEMATICA
1	2022-02-11	Cerrada	Ordinario	Aprobado	PROYECTOS DE LEY	ECONOMÍA
2	2022-06-21	Cerrada	Urgencia	Aprobado	PROYECTOS DE LEY	MEDIO AMBIENTE
3	2022-08-05	Cerrada	Ordinario	Aprobado	PROYECTOS DE LEY	VIDA POLÍTICA
4	2022-08-01	Cerrada	Ordinario	Aprobado	PROYECTOS DE LEY	VIDA POLÍTICA
5	2022-10-31	Cerrada	Urgencia	Aprobado	PROYECTOS DE LEY	ASUNTOS FINANCIEROS

Figura 5.3.2: Ejemplo de iniciativas clasificadas por el programa

5.4 Cambio de variables y ayuda

El programa puede recibir ciertas modificaciones a la hora de clasificar si se cambian ciertas variables por parte del usuario, este puede modificar: el número de temáticas a mostrar, el umbral de diferencia de probabilidades entre las distintas temáticas, el nombre de las columnas donde se leerán los extractos y el nombre de las columnas donde se leerán y crearán las temáticas.

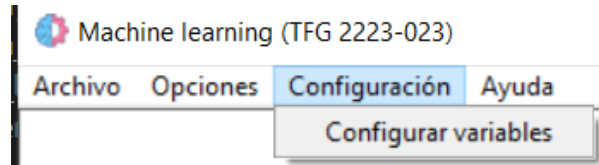


Figura 5.4.1: Opción de configurar variables

Para modificar cualquiera de las variables se debe cumplir ciertos requisitos, ya que saltará su respectiva excepción en caso de que se haya puesto un tipo de valor equivocado en algún campo, los criterios son los siguientes:

- En el campo del número de temáticas a mostrar, debe introducirse un entero del 1 al 10 y no debe dejarse vacío.
- En el campo del umbral de diferencia, debe introducirse un valor float entre el 0 y 1 sin incluir estos mismos valores, este campo no debe dejarse vacío tampoco.
- En los campos del nombre de la columna a extraer y de temática, no deben dejarse vacíos.

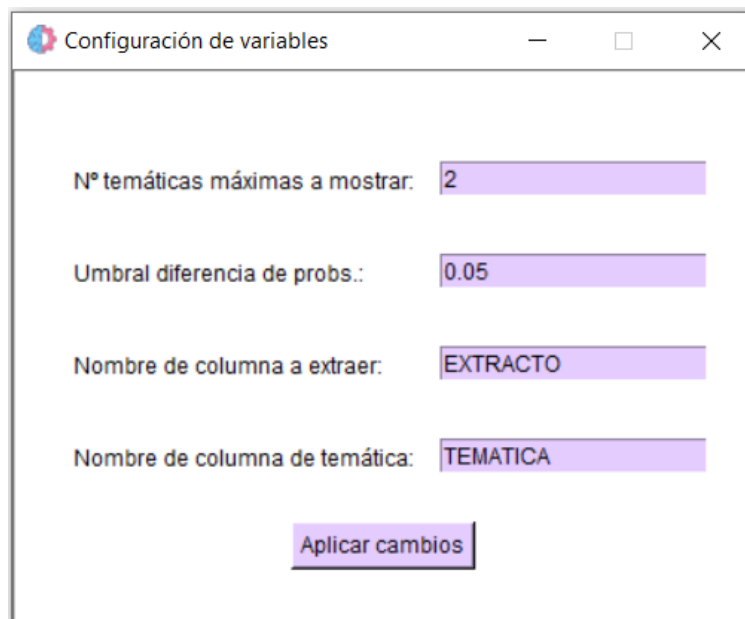


Figura 5.4.2: Ventana para configurar variables

Si hemos puesto las variables de manera correcta aparecerá una ventana indicándonos que se han actualizado los datos perfectamente.

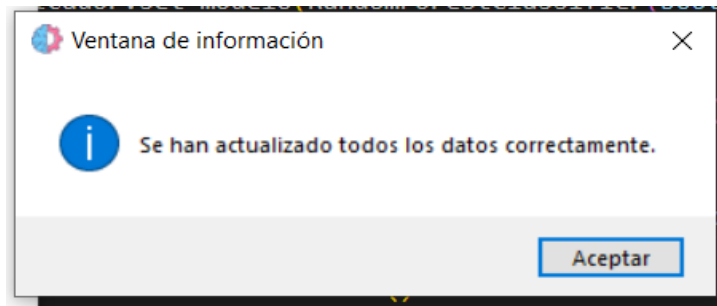


Figura 5.4.3: Ventana que confirma los cambios

Finalmente, como funcionalidad de apoyo existe un apartado de ayuda el cual se puede escoger accediendo en **Ayuda > Información**, dicha opción nos muestra una información bastante resumida acerca de como se utiliza el programa.

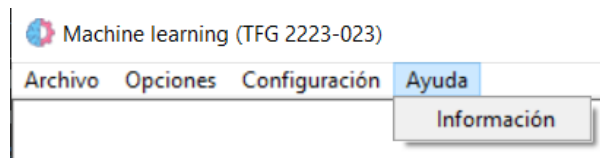


Figura 5.4.4: Opción de ayuda

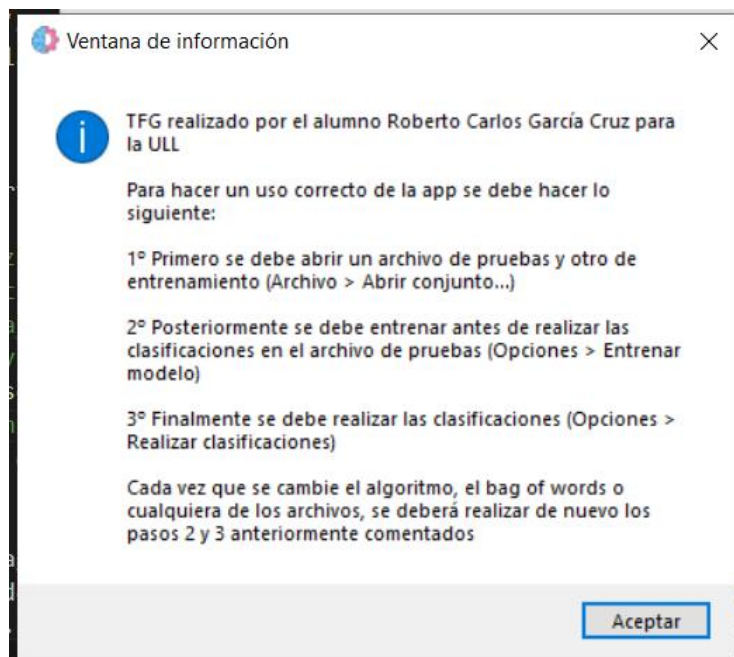


Figura 5.4.5: Texto de ayuda

Cada vez que se modifique un archivo, la bolsa de palabras, el algoritmo o las variables, es necesario realizar el entrenamiento del modelo para que se actualicen los cambios.

Capítulo 6

Desarrollo de herramienta para el aprendizaje

Todo sistema de aprendizaje automático necesita un método para poder adquirir conocimiento, como ya se ha visto antes, hemos seleccionado el aprendizaje supervisado para aplicarlo en nuestro proyecto. En este apartado se describirá como se realiza un conjunto de entrenamiento etiquetado, describiendo que clase de criterios hay que tener en cuenta a la hora de clasificar o de donde se extrae la información.

6.1 EuroVoc

EuroVoc [20] es un tesoro multilingüe y multidisciplinario que es utilizado para describir el tipo de contenido de los documentos de **EUR-Lex** [21], portal web que ofrece una multitud de textos legislativos de la Unión Europea.

Dicho tesoro está disponible en 24 idiomas distintos de la Unión Europea y es utilizado para disciplinar los contenidos que debe tener cada temática.

La clasificación se basará en este tesoro que contiene 21 campos temáticos, los cuales contienen a su vez 127 subcampos. Para determinar de que campo o subcampo se trata una iniciativa parlamentaria, debe realizarse una fijación sobre el vocabulario y las palabras claves que son utilizadas en dicha iniciativa.

Web oficial de la Unión Europea ¿Lo sabías? ▾

EUR-Lex
El acceso al Derecho de la Unión Europea

Español ES Mi EUR-Lex

Funciones experimentales

MENÚ

BÚSQUEDA RÁPIDA

Consejos de búsqueda

¿Necesita más opciones de búsqueda? Utilice la [Búsqueda avanzada](#)

EUROPA > EUR-Lex Inicio > EuroVoc

Explorar

Instituciones de la UE

EuroVoc

EuroVoc

EuroVoc es el tesoro multilingüe y multidisciplinario de la UE. Contiene palabras clave organizadas en 21 campos temáticos y 127 subcampos, que sirven para describir el contenido de los documentos en EUR-Lex.

Busque en la legislación de la UE y documentos afines (disposiciones, documentos preparatorios, acuerdos internacionales, jurisprudencia, preguntas parlamentarias, etc.) por campos y subcampos temáticos.

Más sobre [EuroVoc](#)

Filtro

Figura 6.1.1: Página principal de EuroVoc

6.2 Elaboración

Para elaborar un conjunto de datos, es necesario rellenar un archivo CSV que esté separado por comas cada una de sus columnas. Para dar conocimiento a los algoritmos de como se debe clasificar, es suficiente con desarrollar dos columnas, la columna para colocar los extractos y la columna para añadir su respectiva temática.

Los extractos de las iniciativas parlamentarias para el conjunto de entrenamiento serán extraídos de una API del Parlamento de Canarias [22] que nos muestra todas las iniciativas parlamentarias del Parlamento de Canarias desde el presente año hasta el año 1995. Las iniciativas parlamentarias de los conjuntos de prueba también serán sacadas del mismo punto, por lo tanto, es bastante aconsejable que se realice el entrenamiento de los algoritmos con las iniciativas parlamentarias de dicha web, ya que tendrán una estructura similar a las iniciativas que se desean clasificar.

Debido a la desmesurada cantidad de trabajo y tiempo que supone crear un conjunto de datos consistente, es inviable crear un conjunto con todas las temáticas o subcampos disponibles, por lo tanto, se ha seleccionado un grupo de temáticas que sean capaz de abarcar la mayor parte de los contenidos que se nos pueda ocurrir.

6.2.1 Criterios para clasificar

A pesar de parecer una tarea sencilla, clasificar manualmente un conjunto de datos es una tarea bastante compleja y laboriosa, ya que hay que tener en cuenta que el juicio que se tenga sobre las clasificaciones será el mismo que tendrá nuestro algoritmo de clasificación, es por eso que se tiene que tener en cuenta ciertos criterios a la hora de clasificar:

- Las iniciativas parlamentarias muchas veces mezclan temáticas, situación que puede llevar a confusiones si se clasifica mal. Deben seleccionarse iniciativas parlamentarias con temáticas puras para no dejar ningún tipo de incertidumbre a los algoritmos.
- Hay limitación de iniciativas parlamentarias con temáticas puras en el Parlamento, esto hace que aumente el tiempo de búsqueda a medida que crece el conjunto o inclusive que se agoten las iniciativas parlamentarias puras en algún momento.
- Se debe tener muy en cuenta que palabras claves se están utilizando para cada temática, si se clasifica un extracto con muchas palabras claves de otras temáticas probablemente el algoritmo disminuya su precisión.
- Debe estar balanceado el número de iniciativas parlamentarias para cada temática.
- A medida que se añadan nuevas temáticas, debe aumentarse el número de iniciativas parlamentarias de las demás temáticas anteriores, debido a que obtendremos una gran cantidad de temáticas con pocos extractos, lo que conlleva a una mala precisión.

Teniendo cada uno de estos factores en cuenta es posible crear un conjunto de datos de entrenamiento con una precisión bastante aceptable.

6.3 Resultados

Siguiendo los criterios previamente comentados, se ha obtenido un conjunto que contiene 13 temáticas en 780 iniciativas parlamentarias, 60 iniciativas por temática. Consiguiendo con los algoritmos más enfocados en este tipo de clasificaciones más de un **80%** de precisión, un resultado bastante aceptable respecto a lo que aprendizaje automático se refiere, puesto que un 70% [23] ya se considera apto en estos proyectos.

DISPONIBLES	NO DISPONIBLES
VIDA POLÍTICA	RELACIONES INTERNACIONALES
ECONOMÍA	UNION EUROPEA
ASUNTOS SOCIALES	INTERCAMBIOS ECONÓMICOS Y COMERCIALES
ASUNTOS FINANCIEROS	CIENCIA
PRODUCCIÓN, INNOVACIÓN Y TECNOLOGÍA	EMPRESA Y COMPETENCIA
DERECHO	SECTOR AGROALIMENTARIO
MEDIO AMBIENTE	GEOGRAFÍA
ENERGIA	ORGANIZACIONES INTERNACIONALES
TRANSPORTE	
AGRICULTURA, SILVICULTURA Y PESCA	
EDUCACION Y COMUNICACION	
TRABAJO Y EMPLEO	
INDUSTRIA	

Figura 6.3.1: Las temáticas de EuroVoc en el conjunto de datos

Se procede a dejar el conjunto de datos de entrenamiento creado para este proyecto: [https://drive.google.com/file/d/1h1gjW7ZCoIJCdiqQ_T2a4kLNApRfujF9/view?usp=drive link](https://drive.google.com/file/d/1h1gjW7ZCoIJCdiqQ_T2a4kLNApRfujF9/view?usp=drive_link)

Capítulo 7

Evaluación de herramientas de inteligencia artificial

De aquí en adelante, la memoria solo se centrará en estudios y evaluaciones, puesto que las fases de desarrollo ya han sido descritas.

Desde hace unos meses, una gran preocupación está siendo sembrada en la población a nivel mundial por las herramientas de inteligencia artificial, este temor no se debe solo a las rápidas y precisas respuestas que son capaces de realizar, pudiendo ser más eficientes en algunos casos que algunos profesionales que trabajan en cierto ámbito. Estos programas están llegando a límites insospechados, llegando a crear código de distintos lenguajes de programación o hasta hacer arte mediante la creación de ediciones, dibujos e imágenes.

Muchas de estas herramientas están circulando a día de hoy por las redes de nuestro círculo más cercano, pero entre ellas, la más destacada es ChatGPT [24], la herramienta que está comentando todo el mundo por su fácil accesibilidad y veloces respuestas que ofrece ante cualquier tarea que se le pida.

Tras semejante situación, se ha propuesto como método de estudio usar ChatGPT como herramienta guía para resolver dudas y problemas de este mismo proyecto, a medida que se ha ido avanzando han surgido distintas dificultades, las cuales han sido aprovechadas para posteriormente analizar las respuestas de ChatGPT y obtener ciertas conclusiones acerca de su nivel de fiabilidad.

7.1 Desarrollo con ChatGPT

Para comprobar el funcionamiento de ChatGPT se ha preguntado acerca de tres campos con ciertas diferencias para verificar su desempeño: aprendizaje automático, software y EuroVoc.

El modelo de evaluación será el siguiente, se irá exponiendo a lo largo de la memoria las conversaciones con ChatGPT comentando en algunos puntos ciertas observaciones, posteriormente se mostrarán las conclusiones.

7.1.1 Sobre aprendizaje automático

La conversación inicial con ChatGPT trató sobre aprendizaje automático y el procesamiento del lenguaje natural. Además de hablar ciertas generalidades del tema, también se le preguntó acerca de las librerías más recomendadas en cuanto a dichas ramas para presenciar como de fiables eran sus recomendaciones.

Yo: ¿Qué sabes sobre el machine learning?

ChatGPT:

Como modelo de lenguaje entrenado en GPT-3.5, tengo un conocimiento significativo sobre el tema de Machine Learning. Machine Learning es una rama de la inteligencia artificial que se enfoca en el desarrollo de algoritmos y técnicas que permiten a los sistemas informáticos aprender de los datos y mejorar su rendimiento en tareas específicas, sin necesidad de ser programados explícitamente para cada tarea.

En términos generales, el Machine Learning se divide en tres categorías principales: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo. En el aprendizaje supervisado, los algoritmos se entrenan con datos etiquetados previamente para realizar tareas específicas, como la clasificación o la predicción. En el aprendizaje no supervisado, los algoritmos se utilizan para encontrar patrones y estructuras en los datos no etiquetados. En el aprendizaje por refuerzo, el algoritmo aprende a tomar decisiones a través de la interacción con un ambiente.

El Machine Learning se aplica en muchos campos, como la visión por computadora, el procesamiento de lenguaje natural, la robótica, la biotecnología, (Texto parado por parte del usuario)

Yo: ¿Podrías decirme cuales son los mejores algoritmos para el aprendizaje automático con NLP?

ChatGPT:

Hay varios algoritmos de aprendizaje automático que se utilizan comúnmente en el procesamiento de lenguaje natural (NLP, por sus siglas en inglés), cada uno con sus propias fortalezas y debilidades dependiendo del problema específico que se esté abordando. Aquí te presento algunos de los algoritmos más utilizados:

1. Redes neuronales recurrentes (RNN): son muy populares en NLP, especialmente para tareas que implican secuencias, como la generación de lenguaje natural, la traducción automática y el análisis de sentimientos. Las RNNs se diseñan para modelar dependencias temporales a través de una secuencia, lo que las hace muy adecuadas para el procesamiento del lenguaje natural.
2. Redes neuronales convolucionales (CNN): las CNNs son otro tipo de red neuronal que se utilizan a menudo en tareas de procesamiento de texto, como la clasificación de documentos y el análisis de sentimientos. Las CNNs se utilizan comúnmente para extraer características de los textos, como n-gramas y patrones de palabras.
3. Máquinas de vectores de soporte (SVM): las SVM se utilizan a menudo en tareas de clasificación de texto, como la detección de spam y la clasificación de documentos. Las SVM buscan un hiperplano que separe las diferentes clases de datos, maximizando la distancia entre los puntos más cercanos.
4. Modelos de lenguaje basados en atención (BERT): BERT es un modelo de lenguaje preentrenado basado en atención que ha sido muy exitoso en una amplia variedad de tareas de procesamiento de texto, como la comprensión de lectura y la generación de texto. BERT utiliza una técnica de preentrenamiento llamada máscara de atención para aprender representaciones de palabras de alta calidad.

Cada uno de estos algoritmos tiene sus propias ventajas y desventajas, y la elección del algoritmo adecuado dependerá del problema específico que se esté abordando.

Yo: Se trata de un problema de clasificación y predicción

ChatGPT:

En ese caso, algunas opciones de algoritmos de aprendizaje automático que puedes considerar para un problema de clasificación y predicción en NLP son:

1. Naïve Bayes: es un algoritmo de clasificación probabilístico simple y rápido que se utiliza a menudo en problemas de clasificación de texto. Se basa en el teorema de Bayes para estimar la probabilidad de que un documento pertenezca a una categoría dada, dada su presencia o ausencia de ciertas características.
2. Árboles de decisión: son un tipo de algoritmo de aprendizaje supervisado que se utilizan a menudo en problemas de clasificación. Un árbol de decisión es un modelo de predicción que se construye a partir de una serie de decisiones en cascada, basadas en características específicas del texto.
3. Random Forest: es una técnica de ensamble que utiliza múltiples árboles de decisión para producir una predicción. Cada árbol de decisión se construye utilizando una muestra aleatoria del conjunto de entrenamiento y se utiliza para votar por la predicción final.
4. Redes neuronales: ya mencioné las redes neuronales recurrentes (RNN) y las convolucionales (CNN) como opciones para problemas de clasificación y predicción en NLP, pero también puedes considerar otros tipos de redes neuronales, como las redes neuronales de retroalimentación hacia adelante (FFNN), que se utilizan a menudo en la clasificación de texto.

La elección del algoritmo dependerá de los detalles específicos del problema que estés tratando de resolver, como la cantidad y calidad de los datos, la complejidad de la tarea y el tiempo disponible para entrenar el modelo.

Como se observa, en las preguntas generales no tiene ningún tipo de problema en responder, puesto que probablemente ya haya una gran cantidad de información dispersada a lo largo de la red que puede responder esa pregunta. Al tratar sobre asuntos concretos, es necesario ponerle en situación e indicarle concretamente lo que se quiere junto a las debidas restricciones.

Los algoritmos de aprendizaje automático que nos ha comentado no son los mejores algoritmos para esta situación, pero son bastante recomendables, por lo tanto, nos aporta información bastante aceptable.

Yo: ¿Qué librerías para Python de NLP me recomendarías?

ChatGPT:

Existen varias librerías populares para procesamiento de lenguaje natural (NLP, por sus siglas en inglés) en Python. Algunas de las más comunes y recomendadas son:

1. NLTK (Natural Language Toolkit): Es una de las librerías más antiguas y populares para el procesamiento de texto en Python. Proporciona una amplia gama de herramientas y recursos para el análisis de texto, como tokenización, análisis morfológico, etiquetado de partes del discurso, análisis de sentimientos y mucho más.
2. spaCy: Es una librería de procesamiento de lenguaje natural moderna y eficiente que se utiliza para tareas como tokenización, análisis morfológico, extracción de entidades, análisis de dependencias y más. Es conocida por ser muy rápida y escalable.
3. Gensim: Es una librería popular para el modelado de temas y la recuperación de información en grandes conjuntos de documentos de texto. También proporciona herramientas para el análisis de texto, como la tokenización y el análisis de similitud.
4. Scikit-learn: Es una librería de aprendizaje automático ampliamente utilizada que proporciona herramientas para la clasificación, la regresión y el clustering. También ofrece herramientas para el procesamiento de texto, como la vectorización de documentos y el análisis de sentimientos.
5. TensorFlow y PyTorch: Son dos de las librerías de aprendizaje profundo más populares que se utilizan para el procesamiento de lenguaje natural. Proporcionan herramientas para construir y entrenar redes neuronales, incluyendo modelos de lenguaje basados en atención como BERT.

La elección de la librería dependerá del problema específico que se esté abordando y del nivel de experiencia que se tenga en programación y procesamiento de texto en Python. En general, NLTK y spaCy son muy populares y fáciles de usar para los principiantes, mientras que Gensim, Scikit-learn, TensorFlow y PyTorch son más adecuados para usuarios avanzados que necesitan un mayor control sobre el procesamiento y la modelización de texto.

Yo: ¿Qué librería trabaja en español?

ChatGPT:

Todas las librerías que mencioné en mi respuesta anterior (NLTK, spaCy, Gensim, Scikit-learn, TensorFlow y PyTorch) tienen soporte para el procesamiento de texto en español, aunque algunas pueden tener más recursos y modelos pre-entrenados para el inglés debido a su mayor popularidad en el procesamiento de lenguaje natural.

Sin embargo, existen también algunas librerías especializadas en el procesamiento de texto en español que podrían ser de interés:

1. Pattern: Es una librería de procesamiento de lenguaje natural que proporciona herramientas para el análisis de texto en varios idiomas, incluyendo el español. Proporciona herramientas para la tokenización, el análisis morfológico, el análisis sintáctico y más.

2. Stanza: Es una librería de procesamiento de lenguaje natural que ofrece soporte para varios idiomas, incluyendo el español. Proporciona herramientas para la tokenización, el análisis morfológico, el análisis sintáctico y más.
3. Polyglot: Es una librería de procesamiento de lenguaje natural que proporciona herramientas para el análisis de texto en varios idiomas, incluyendo el español. Proporciona herramientas para la tokenización, el análisis morfológico, la detección de entidades, el análisis de sentimientos y más.

La elección de la librería dependerá del problema específico que se esté abordando y del nivel de experiencia que se tenga en programación y procesamiento de texto en Python. En general, NLTK y spaCy también tienen soporte para el español y son muy populares y fáciles de usar para los principiantes.

Yo: ¿Sabes lo que es la denominada "bag of words"?

ChatGPT:

Sí, la "bag of words" (BOG) es una técnica común utilizada en el procesamiento de lenguaje natural (NLP, por sus siglas en inglés) para representar y analizar el texto. Básicamente, se trata de un modelo de representación de texto que ignora la estructura gramatical y se enfoca en la frecuencia de las palabras en un documento.

En este modelo, cada palabra en el texto se convierte en una "bolsa" (bag) de palabras sin tener en cuenta su orden y relación con otras palabras. A continuación, se cuentan las ocurrencias de cada palabra en el documento y se crea un vector de características con la frecuencia de las palabras en el texto. Este modelo se utiliza a menudo para la clasificación de texto y la minería de texto en general.

Yo: ¿Qué tipos de bag of words existen?

ChatGPT:

Existen varios tipos de modelos "bag of words" (BOG) que se diferencian en la forma en que se procesa el texto y se construyen los vectores de características. Algunos de los modelos BOG más comunes son:

1. BOG con frecuencia de términos: Este modelo cuenta la frecuencia de aparición de cada término en el texto y lo utiliza para construir el vector de características.
2. BOG con frecuencia inversa de documentos (TF-IDF): Este modelo ajusta la frecuencia de los términos por la frecuencia con la que aparecen en el conjunto de documentos, lo que reduce el peso de los términos comunes y aumenta el peso de los términos raros.
3. BOG binario: Este modelo solo registra si una palabra aparece o no en un documento, en lugar de contar la frecuencia de cada palabra.
4. BOG con n-gramas: Este modelo considera conjuntos de n palabras consecutivas como características, en lugar de palabras individuales. Por ejemplo, un modelo BOG con n-gramas de 2 tendría características para "perro marrón", "marrón gato", etc.
5. BOG con palabras ponderadas: Este modelo asigna pesos a las palabras en función de su relevancia para el análisis. Por ejemplo, se pueden asignar pesos más altos a las palabras clave y pesos más bajos a las palabras comunes.

Yo: ¿Cuál es la mejor para la clasificación de textos?

ChatGPT:

La elección del mejor modelo BOG depende en gran medida del conjunto de datos y del objetivo de clasificación específico. En general, el modelo BOG con frecuencia inversa de documentos (TF-IDF) es uno de los modelos más populares y efectivos para la clasificación de textos.

TF-IDF reduce el peso de las palabras comunes que aparecen en muchos documentos, mientras que aumenta el peso de las palabras más raras que son más distintivas y pueden proporcionar una mejor información para la clasificación. Además, el modelo TF-IDF se puede combinar con diferentes algoritmos de aprendizaje automático, como Naive Bayes, SVM y redes neuronales, para mejorar la precisión de la clasificación.

Sin embargo, es importante probar varios modelos BOG y técnicas de preprocesamiento de texto para encontrar el que mejor se adapte a los datos y al problema de clasificación específico.

En este caso, ChatGPT acertó de lleno en la información proporcionada, puesto que estas herramientas de las que nos está hablando son las que se han seleccionado en el proyecto, tanto por la bolsa de palabras como por las librerías de procesamiento del lenguaje ofrecidas.

Por el momento, se puede ver como la información que ofrece suele ser bastante acertada, puede que no sea la mejor de todas, pero sí lo suficiente como para fiarnos de él en esta clase de temas si se explica más detalladamente la situación.

7.1.2 Sobre software y fallos

Lo siguiente que se va a mostrar es un caso real, la situación es la siguiente, se pretendía subir la aplicación una vez ya acabada a GitHub, pero el software contiene un conjunto de librerías que deben ser instaladas. Debido a mi inexperiencia con Python, tenía bastantes dudas sobre como exportar las librerías para poder compartirla con los usuarios de una manera cómoda y que no diera muchas complicaciones, este fue el resultado.

Yo: ¿Sabes de Github?

ChatGPT:

Sí, estoy familiarizado con GitHub. GitHub es una plataforma de desarrollo colaborativo basada en la web que utiliza el sistema de control de versiones Git. Proporciona un entorno para alojar, gestionar y colaborar en proyectos de software.

Algunas características destacadas de GitHub incluyen:

1. Repositorios: Los repositorios son espacios donde se almacena el código fuente y otros archivos relacionados con un proyecto. Los desarrolladores pueden clonar, contribuir y colaborar en los repositorios.
2. Control de versiones: GitHub utiliza Git como sistema de control de versiones, lo que permite a los desarrolladores realizar un seguimiento de los cambios en el código a lo largo del tiempo y revertirlos si es necesario.
3. Ramas (Branches): Las ramas permiten a los desarrolladores trabajar en diferentes versiones del código de forma aislada y luego fusionar los cambios cuando sea necesario.
4. Solicitudes de extracción (Pull Requests): Las solicitudes de extracción permiten a los desarrolladores proponer cambios en un repositorio y solicitar que se revisen y se fusionen en la rama principal.
5. Problemas (Issues): Los problemas se utilizan para realizar un seguimiento de tareas, errores o solicitudes de mejora en un proyecto. Los miembros del equipo pueden comentar, asignar y realizar un seguimiento del progreso de los problemas.

GitHub es ampliamente utilizado por desarrolladores y equipos de desarrollo de software en todo el mundo. Además de alojar repositorios de código, GitHub también proporciona herramientas para la colaboración, como wikis, proyectos y acciones (CI/CD).

Yo: ¿Como añado las dependencias a mi proyecto de Python y lo subo a Github?

ChatGPT:

Para añadir dependencias a tu proyecto de Python y subirlo a GitHub, puedes seguir estos pasos generales:

1. Inicializar un repositorio en GitHub: Primero, crea un repositorio vacío en GitHub. Puedes hacerlo directamente en la interfaz web de GitHub.
2. Configurar el repositorio local: En tu máquina local, ve al directorio de tu proyecto de Python usando la línea de comandos y ejecuta el comando **git init** para inicializar un repositorio Git local.

3. Crear un archivo de requisitos: Crea un archivo llamado **requirements.txt** en el directorio raíz de tu proyecto. En este archivo, puedes listar las dependencias de tu proyecto, una por línea, con el formato **nombre_dependencia==versión**. Por ejemplo:
makefileCopy code
numpy==1.19.2 pandas==1.1.3
4. Instalar las dependencias: Ejecuta el comando **pip install -r requirements.txt** en tu entorno de desarrollo para instalar todas las dependencias especificadas en el archivo **requirements.txt**.
5. Crear un archivo **.gitignore**: Crea un archivo llamado **.gitignore** en el directorio raíz de tu proyecto. En este archivo, puedes especificar los archivos y directorios que deseas excluir del control de versiones de Git. Por ejemplo, puedes excluir los archivos generados por el entorno virtual o los archivos de caché. Asegúrate de incluir el archivo **venv/** si estás utilizando un entorno virtual llamado **venv**.
6. Realizar los commits y push: Utiliza los comandos **git add .** para agregar todos los archivos al área de preparación y **git commit -m "Mensaje de commit"** para realizar un commit con un mensaje descriptivo. Luego, ejecuta **git remote add origin <URL del repositorio remoto>** para conectar tu repositorio local con el repositorio remoto en GitHub. Finalmente, utiliza **git push -u origin master** para subir tus cambios al repositorio remoto.

Una vez completados estos pasos, tu proyecto de Python y el archivo **requirements.txt** estarán disponibles en GitHub, lo que permitirá a otros clonar y ejecutar tu proyecto con las mismas dependencias especificadas. Recuerda actualizar el archivo **requirements.txt** siempre que agregues o modifiques las dependencias de tu proyecto.

Yo: ¿Puedes realizarme un archivo requirements?

ChatGPT:

Por supuesto, puedo ayudarte a generar un archivo **requirements.txt** basado en las dependencias de tu entorno de desarrollo actual. Para hacerlo, necesitaré acceso al entorno virtual de tu proyecto. A continuación se muestra un ejemplo de cómo generar un archivo **requirements.txt** utilizando la biblioteca **pipreqs**:

1. Instalar **pipreqs**:
Copy code
pip install pipreqs
2. Navega hasta el directorio raíz de tu proyecto en la línea de comandos.
3. Ejecuta el siguiente comando para generar el archivo **requirements.txt**:
cssCopy code
pipreqs . --force

Este comando buscará todas las dependencias utilizadas en tu código y generará un archivo **requirements.txt** en el directorio actual, que contendrá una lista de todas las dependencias y sus versiones. Asegúrate de ejecutar el comando dentro de tu entorno virtual activado para obtener las dependencias específicas de tu proyecto.

Una vez que tengas el archivo **requirements.txt**, puedes incluirlo en tu repositorio de GitHub siguiendo los pasos mencionados anteriormente.

Yo: ¿Como añado un modelo de la librería de spacy en el archivo “requirements.txt”, el modelo en específico es `es_core_news_sm`

ChatGPT:

Para agregar el modelo **es_core_news_sm** de SpaCy a tu archivo **requirements.txt**, puedes seguir estos pasos:

1. Asegúrate de tener SpaCy instalado en tu entorno virtual. Si no lo tienes instalado, ejecuta el siguiente comando:

Copy code

```
pip install spacy
```

2. Verifica si ya tienes el modelo **es_core_news_sm** instalado ejecutando el siguiente comando:

Copy code

```
python -m spacy info es_core_news_sm
```

Si el modelo ya está instalado, verás información sobre él. Si no está instalado, continúa con el siguiente paso.

3. Agrega la línea correspondiente al modelo **es_core_news_sm** al archivo **requirements.txt**:

makefileCopy code

```
es_core_news_sm==3.1.0
```

Asegúrate de incluir la versión específica que deseas utilizar. En este caso, se ha utilizado la versión 3.1.0 como ejemplo.

4. Guarda el archivo **requirements.txt** con la nueva línea agregada.

Una vez que hayas realizado estos pasos, cuando alguien instale las dependencias de tu proyecto utilizando el archivo **requirements.txt**, también se instalará el modelo **es_core_news_sm** de SpaCy en la versión especificada. Recuerda ejecutar **pip install -r requirements.txt** para instalar todas las dependencias.

Al realizar la búsqueda de la información, pude comprobar que esta era totalmente correcta y una solución bastante aceptable, esto ofrece una gran ventaja para el campo de la informática, mucho del tiempo que se pierde a la hora de trabajar es solucionando errores simples o dudas como esta, debido a que siempre se suele estar probando nuevas herramientas.

De esta manera, el usuario con poner una simple línea de código obtendrá todas las librerías necesarias.

7.1.3 Sobre EuroVoc e iniciativas parlamentarias

Ahora se hablará sobre algo que no tiene mucho que ver con la tecnología, esta vez será más bien sobre iniciativas parlamentarias, el tipo de alguna de estas y EuroVoc.

No es raro ver que se haya pasado a un tema totalmente distinto respecto a los anteriores, puesto que mucha de esa información que nos ha dicho antes probablemente está subida en distintas webs de la red y ChatGPT procesa la información de una manera específica para que sea más correcta.

Yo: ¿Sabrías clasificar leyes?

ChatGPT:

Sí, puedo ayudarte a clasificar leyes. Existen varias formas en las que las leyes pueden ser clasificadas, pero aquí te presento algunas de las clasificaciones más comunes:

1. Por su origen: Las leyes pueden ser clasificadas según su origen en leyes constitucionales, leyes estatutarias y leyes comunes. Las leyes constitucionales son aquellas que se derivan de la Constitución de un país y establecen los principios y valores fundamentales del estado. Las leyes estatutarias son aquellas aprobadas por el poder legislativo y establecen las reglas y normas para una determinada área o tema. Las leyes comunes son aquellas que se derivan de la costumbre y la jurisprudencia.
2. Por su alcance: Las leyes pueden ser clasificadas según su alcance en leyes nacionales, regionales y locales. Las leyes nacionales son aquellas que son aplicables en todo el territorio del país, mientras que las leyes regionales y locales son aquellas que son aplicables solo en una determinada región o localidad.
3. Por su ámbito de aplicación: Las leyes pueden ser clasificadas según su ámbito de aplicación en leyes civiles, penales, laborales, fiscales, administrativas, entre otras. Las leyes civiles establecen las normas para las relaciones civiles entre las personas. Las leyes penales establecen las normas para el castigo de los delitos. Las leyes laborales establecen las normas para las relaciones laborales entre los empleados y los empleadores. Las leyes fiscales establecen las normas para la recaudación de impuestos. Las leyes administrativas establecen las normas para la administración pública.
4. Por su temporalidad: Las leyes pueden ser clasificadas según su temporalidad en leyes permanentes y leyes temporales. Las leyes permanentes son aquellas que están diseñadas

Yo: ¿Sabes que es EuroVoc?

ChatGPT:

Sí, EuroVoc es un tesoro multilingüe utilizado por la Unión Europea para indexar y clasificar documentos y publicaciones relacionadas con las políticas y actividades de la UE. EuroVoc contiene más de 7.500 descriptores y 30.000 términos no descriptores en 23 idiomas oficiales de la UE y otros idiomas de trabajo.

El objetivo principal de EuroVoc es proporcionar un lenguaje común para la indexación, la búsqueda y la recuperación de documentos relacionados con las políticas de la UE. EuroVoc se utiliza en varias plataformas y sistemas de información de la UE, como el portal de la UE, el Registro de la UE y la Biblioteca Digital Europea.

EuroVoc también es utilizado por organizaciones y sistemas de información fuera de la UE como una herramienta de clasificación y búsqueda para documentos y publicaciones relacionados con las políticas de la UE.

Cada término en EuroVoc está asociado con un código numérico único y una jerarquía de relaciones semánticas que permite una indexación y recuperación más precisa de documentos. Además, EuroVoc se actualiza regularmente para reflejar los cambios en las políticas y las necesidades de los usuarios.

Yo: ¿Cuáles son las categorías de EuroVoc?

ChatGPT:

EuroVoc contiene más de 7.500 descriptores y 30.000 términos no descriptores, organizados en un sistema jerárquico de categorías temáticas. Estas categorías temáticas son:

1. GENERALIDADES
2. AMBIENTE
3. ASUNTOS ECONÓMICOS
4. EDUCACIÓN Y COMUNICACIÓN
5. EMPRESAS
6. ENERGÍA
7. FINANZAS
8. RELACIONES INTERNACIONALES
9. DERECHO
10. ORGANIZACIÓN DE ESTADOS Y TERRITORIOS
11. POLÍTICA
12. CIENCIA, TECNOLOGÍA E INVESTIGACIÓN
13. SOCIEDAD
14. TRANSPORTES
15. AGROALIMENTACIÓN
16. INDUSTRIAS DE LA PESCA Y ACUICULTURA
17. PRODUCCIÓN, TECNOLOGÍA Y RESEARCH
18. SALUD
19. SISTEMAS DE INFORMACIÓN, EDUCACIÓN Y CULTURA
20. CIENCIAS NATURALES Y CIENCIAS DE LA INGENIERÍA
21. ESPACIO, TIEMPO Y CIENCIAS DE LA TIERRA
22. ECONOMÍA Y GESTIÓN
23. PROTECCIÓN DE LAS PERSONAS Y DE LOS BIENES
24. CIENCIAS SOCIALES

Cada categoría temática se divide en subcategorías y sub-subcategorías más específicas. La estructura jerárquica de categorías y subcategorías permite la búsqueda y recuperación de documentos de manera más precisa y eficiente en función de su contenido y temática.

Yo: ¿Cuáles son los campos temáticos de EuroVoc?

ChatGPT:

Los campos temáticos de EuroVoc son una agrupación de categorías temáticas relacionadas entre sí. Estos campos temáticos se utilizan para clasificar documentos de manera más general y amplia, y son los siguientes:

1. Política, economía y gestión (PEM)
2. Sociedad (SOC)
3. Cultura y entretenimiento (CUL)
4. Ciencia y tecnología (CT)
5. Medio ambiente y cambio climático (ENV)
6. Agricultura, pesca y alimentación (AGR)
7. Transporte y movilidad (TRA)
8. Energía (ENE)
9. Ciencias médicas y de la salud (MED)
10. Derecho y justicia (DER)

Cada campo temático está compuesto por varias categorías temáticas relacionadas. Por ejemplo, el campo temático "Política, economía y gestión" incluye las categorías temáticas de "Política", "Asuntos económicos", "Finanzas", "Relaciones internacionales", "Derecho", "Organización de estados y territorios" y "Ciencias sociales".

Los campos temáticos son útiles para realizar búsquedas amplias y generar informes sobre áreas temáticas específicas. También ayudan a los usuarios a navegar por la gran cantidad de términos y categorías de EuroVoc de manera más eficiente.

Yo: Sabías que EuroVoc tiene 21 categorías temáticas

ChatGPT

Lo siento, mi respuesta anterior estaba desactualizada. Efectivamente, EuroVoc cuenta con 21 categorías temáticas, no con campos temáticos. Estas categorías temáticas son las siguientes:

1. Política, instituciones y administración
2. Derecho
3. Economía
4. Finanzas
5. Empresas y competencia
6. Empleo y formación
7. Asuntos sociales
8. Educación y comunicación
9. Cultura y religión
10. Medio ambiente
11. Agricultura, pesca y alimentación
12. Energía
13. Industria y tecnología
14. Transporte
15. Investigación y ciencia

16. Estadísticas
17. Geografía
18. Internacional
19. Ayuda humanitaria y protección civil
20. Defensa
21. Deporte, turismo y tiempo libre

Cada categoría temática de EuroVoc tiene una estructura jerárquica de descriptores y términos no descriptores que permiten una indexación y recuperación más precisa de documentos relacionados con las políticas y actividades de la UE.

Yo: ¿Cuántas categorías temáticas tiene EuroVoc?

ChatGPT:

EuroVoc cuenta con 21 categorías temáticas en total. Estas categorías temáticas abarcan una amplia gama de temas, desde política e instituciones hasta deportes, turismo y tiempo libre. Cada categoría temática se subdivide en una estructura jerárquica de descriptores y términos no descriptores, lo que permite una indexación y recuperación más precisa de documentos relacionados con las políticas y actividades de la UE.

A pesar de haberle puesto en contexto y haberle especificado de distintas maneras, no ha logrado responder correctamente a mi cuestión.

Tras la corrección, ChatGPT sabe que ha tenido un fallo y tiene que corregir dicha información, el número de temáticas ahora es correcta, pero las temáticas como tal no lo son, por ejemplo, no hay ninguna temática denominada "Defensa" o "Estadísticas" en EuroVoc.

Aun así, se ha procedido a preguntar sobre qué temática puede ser cierta iniciativa parlamentaria, los resultados en algunos casos han sido acertados, en otros no.

Yo: Iniciativa de creación del Colegio Profesional de Dietistas-Nutricionistas de Canarias, ¿de qué tipo es?

ChatGPT:

La iniciativa "De creación del Colegio Profesional de Dietistas-Nutricionistas de Canarias" puede clasificarse dentro de la categoría temática de "Salud" de EuroVoc, ya que se refiere a la creación de una entidad profesional dedicada a la nutrición y la dietética, que son disciplinas fundamentales para la promoción de la salud y la prevención de enfermedades. En este caso, la iniciativa se centra en la creación de un colegio profesional específico para dietistas-nutricionistas en la región española de Canarias.

Yo: Iniciativa de Memoria Histórica de Canarias y de Reconocimiento y Reparación moral de las víctimas canarias de la Guerra Civil y la dictadura franquista, ¿de qué tipo es?

ChatGPT:

La iniciativa "De Memoria Histórica de Canarias y de Reconocimiento y Reparación moral de las víctimas canarias de la Guerra Civil y la dictadura franquista" pertenece a la categoría temática de "Historia y Arqueología" de EuroVoc, ya que se refiere al estudio y análisis de los hechos históricos y a su interpretación y comprensión en el contexto actual. En este caso, la iniciativa se centra en la

memoria histórica y la reparación moral de las víctimas canarias de la Guerra Civil y la dictadura franquista, un tema relacionado con la historia y la política de la región.

Los casos que se han expuesto son las respuestas fallidas que se han detectado, ha acertado dos de cuatro, puesto que las temáticas “Salud” e “Historia y Arqueología” no existen ni en los campos ni en los subcampos, por lo tanto, es información bastante errónea que ni siquiera se sabe de donde la está sacando.

7.2 Conclusiones acerca de la herramienta

La conclusión que se puede sacar acerca de la herramienta es la siguiente, por el momento, según la temática de la que se hable, la herramienta puede resultar bastante fiable o no.

En cuestión a temas sobre la informática y ramas de la misma, es evidente que puede darnos información bastante fiable debido a la cantidad de gente de este campo que en los últimos tiempos la ha usado tanto para solucionar errores como para realizar sus programas, siendo también muy probablemente el creador de muchos trabajos desde el nivel escolar hasta el nivel universitario, retroalimentándose de toda esta información y siendo cada día más eficaz que el anterior.

Pero como se ha mencionado antes, temas que sean muy raros de comentar o que la gente no ha comentado con la herramienta, como pueden ser temas parlamentarios o políticos, en este caso, es muy probable que tienda a dar información ligeramente errónea, teniendo el usuario que ser muy específico y dando cierta retroalimentación a ChatGPT, suceso bastante entendible.

Eso no quiere decir que el día de mañana esta herramienta no esté preparada para abastecer dichos temas que no son muy hablados en la red, si se ha llegado hasta el punto de poder realizar cosas tan abstractas como crear arte o código fuente, es solo cuestión de tiempo que pueda dominar ciertas ideas, pero en el momento actual no es capaz de acertar en todo.

A pesar de todo, basta con decir que información que se ha tardado en encontrar minutos (inclusive horas), ChatGPT la ha dado en cuestión de pocos segundos y de manera bastante completa, así que en muchos ámbitos, sobre todo si es acerca de un campo del cual mucha gente busca bastante en internet, puede resultar bastante rentable utilizar esta herramienta.

Capítulo 8

Evaluación de algoritmos

La tarea pendiente por realizar es la evaluación de los algoritmos, en este apartado veremos el rendimiento que presenta cada algoritmo ante este problema a través de un conjunto de métricas básicas que se utilizan en el aprendizaje automático.

8.1 Conceptos

Para medir el desempeño de cada algoritmo se procederá a explicar las métricas [25] que debemos entender para comprender los resultados, nos centraremos esencialmente en las que se mostrarán en este trabajo.

8.1.1 Matriz de confusión

El primer elemento que se mostrará es la **matriz de confusión**, a través de este elemento es de donde se sacan el resto de métricas.

La matriz de confusión es una matriz de valores donde se hace una comparación de los valores predichos y los valores reales.

		Predicción	
		NO	SI
Realidad	NO	TN	FP
	SI	FN	TP

Tabla 8.1.1.1: Ejemplo de matriz de confusión

Este es un ejemplo muy simple de una matriz de confusión, puesto que se trata de una sobre predicción binaria, pero basta para entender los conceptos que se quieren comentar.

Para entenderlo inclusive mejor se propondrá el siguiente ejemplo, se pretende saber si un día va a llover o no, para ello tenemos una muestra de 100 días.

		Predicción	
		NO	SI
Realidad	NO	20	10
	SI	10	60

Tabla 8.1.1.2: Ejemplo de matriz de confusión con valores

A continuación, se describirán las métricas que nos ayudarán a entender como leer esta matriz.

8.1.2 Precision

La precisión determina la calidad del modelo que se tiene, es decir, calcula el porcentaje real de los casos afirmativos. Mediante la siguiente fórmula podemos hallar la precisión:

$$\text{PRECISION} = \text{TP} / (\text{TP} + \text{FP}) \rightarrow 60 / (60 + 10) = 0.85714$$

Esto quiere decir que cuando se predice un día lluvioso existe un 85% de probabilidades de que nuestro modelo acierte.

8.1.3 Recall

La métrica de recall, establece que cantidad de positivos reales se han identificado correctamente. Su cálculo es bastante similar a la fórmula anterior:

$$\text{RECALL} = \text{TP} / (\text{TP} + \text{FN}) \rightarrow 60 / (60 + 10) = 0.85714$$

Este modelo es capaz de identificar un 85% de los días que va a llover realmente.

8.1.4 F1

El valor F1, combina los valores de recall y precision en un solo valor. Su fórmula es la media armónica de ambas.

$$\text{F1} = 2 * (\text{PRECISION} * \text{RECALL}) / (\text{PRECISION} + \text{RECALL}) \rightarrow$$

$$2 * (0.85714 * 0.85714) / (0.85714 + 0.85714) = 0.85714$$

8.1.5 Accuracy

La métrica más usada y probablemente a la que más prestamos atención es la accuracy, puesto que se centra en calcular el porcentaje de los casos que ha acertado nuestro modelo. Se calcula de la siguiente manera:

$$\text{ACCURACY} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) = (60 + 20) / (60 + 20 + 10 + 10) = 0.80$$

Lo que nos pretende indicar es el porcentaje de acierto del modelo, que como se puede ver es exactamente un 80%.

Hay que tener en cuenta que para usar esta métrica es muy importante que las clases estén balanceadas.

8.2 Algoritmos

Para el presente proyecto se han elegido un total de 17 algoritmos, seleccionando los más relevantes de 8 familias distintas de modelos de clasificación para esta situación en concreto.

En el siguiente apartado se realizará una descripción breve de cada uno de ellos junto a su configuración ideal en la librería de Scikit-learn.

Las variables de cada configuración están definidas en la página de cada algoritmo en la web de Scikit-learn [26].

Algoritmo: Gaussian	Tipo: Naive Bayes (NB)
Descripción: Clasificador probabilístico influenciado por el Teorema de Bayes. Es usado cuando se espera que se siga una distribución gaussiana.	
Configuración: var_smoothing = 11	

Algoritmo: Multinomial	Tipo: Naive Bayes (NB)
Descripción: Clasificador probabilístico influenciado por el Teorema de Bayes. Es usado cuando se espera que se siga una distribución multinomial, normalmente es utilizado para resolver clasificaciones de texto.	
Configuración: alpha = 1.60, force_alpha = True	

Algoritmo: Bernoulli [27]	Tipo: Naive Bayes (NB)
Descripción: Clasificador probabilístico influenciado por el Teorema de Bayes. Es usado cuando se espera que se siga la distribución de Bernoulli, normalmente es utilizado en clasificaciones binarias.	
Configuración: alpha = 0.08, force_alpha = True	

Algoritmo: Decision Tree [28]	Tipo: Tree (TR)
Descripción: Clasificador el cual su toma de decisiones está basada en una estructura de árbol, a medida que toma esas decisiones está generando dicha estructura.	
Configuración: splitter = "random", class_weight = "balanced"	

Algoritmo: Extra Tree	Tipo: Tree (TR)
Descripción: Clasificador muy parecido a "Decision Tree", pero su toma de decisiones es aleatoria.	
Configuración: max_features = None	

Algoritmo: K – Neighbors	Tipo: Neighbors (KNN)
Descripción: Clasificador basado en la cercanía de las instancias respecto la que se quiere clasificar.	
Configuración: algorithm = "brute", n_neighbors = 13, weights = "distance"	

Algoritmo: Centroid [29]	Tipo: Neighbors (KNN)
Descripción: Clasificador de centroide más cercano, cada clase está representada por su centroide.	
Configuración: Predeterminada	

Algoritmo: Random Forest [30]	Tipo: Ensemble (ENS)
Descripción: Clasificador formado por un conjunto de árboles de decisión combinados con bagging .	
Configuración: bootstrap = False	

Algoritmo: Gradient Boost [31]	Tipo: Ensemble (ENS)
Descripción: Clasificador formado por un conjunto de árboles de decisión entrenados de manera secuencial, por lo que cada nuevo árbol será una mejora del anterior.	
Configuración: learning_rate = 0.5, max_depth = None, tol = 0.000000001	

Algoritmo: Ada Boost [32]	Tipo: Ensemble (ENS)
Descripción: Modelo que entrena un clasificador base y este es mejorado hasta tener un nuevo clasificador.	
Configuración: estimator = MultinomialNB(alpha = 1.60, force_alpha = True), algorithm = "SAMME.R", n_estimators = 200, learning_rate = 1.45	

Algoritmo: Lineal SVC [33]	Tipo: Support Vector Classification (SVC)
Descripción: Clasificador basado en el concepto de hiperplano. Utilizado para clasificación múltiple y regresión. Contiene un kernel lineal.	
Configuración: kernel = "linear", probability = True, decision_function_shape = "ovo"	

Algoritmo: SVR	Tipo: Support Vector Classification (SVC)
Descripción: Clasificador basado en el concepto de hiperplano. Utilizado para clasificación múltiple y regresión. Contiene un kernel rbf.	
Configuración: kernel = "rbf", probability = True	

Algoritmo: MLP	Tipo: Neural Network
Descripción: Clasificador basado en redes neuronales.	
Configuración: hidden_layer_sizes = (125,), alpha = 0.0000000001, max_iter = 300	

Algoritmo: Logistic Regression [34]	Tipo: Linear model
Descripción: Técnica de análisis de datos que encuentra relaciones entre dos factores mediante las matemáticas, gracias a esta relación puede predecir los valores.	
Configuración: class_weight = "balanced", C = 2.25, tol = 0.0000001	

Algoritmo: SGD	Tipo: Linear model
Descripción: Clasificador que no pertenece como tal a una familia de aprendizaje, es una técnica de optimización que está basada en máquina de vectores y regresión logística. Es utilizado en clasificación de textos y procesamiento del lenguaje natural.	
Configuración: loss = "huber"	

Algoritmo: Linear Discriminant	Tipo: Discriminant análisis (DIS)
Descripción: Clasificador que se centra en la reducción de dimensionalidad y la clasificación multiclase. Utiliza técnicas probabilísticas como la regla de Bayes y realiza una separación lineal de los datos.	
Configuración: solver = "lsqr", shrinkage = 0.75	

Algoritmo: Quadratic Discriminant [35]	Tipo: Discriminant análisis (DIS)
Descripción: Clasificador variante de "Linear Discriminant" que permite la separación no lineal de los datos.	
Configuración: tol = 0.0000001	

8.3 Resultados y conclusiones

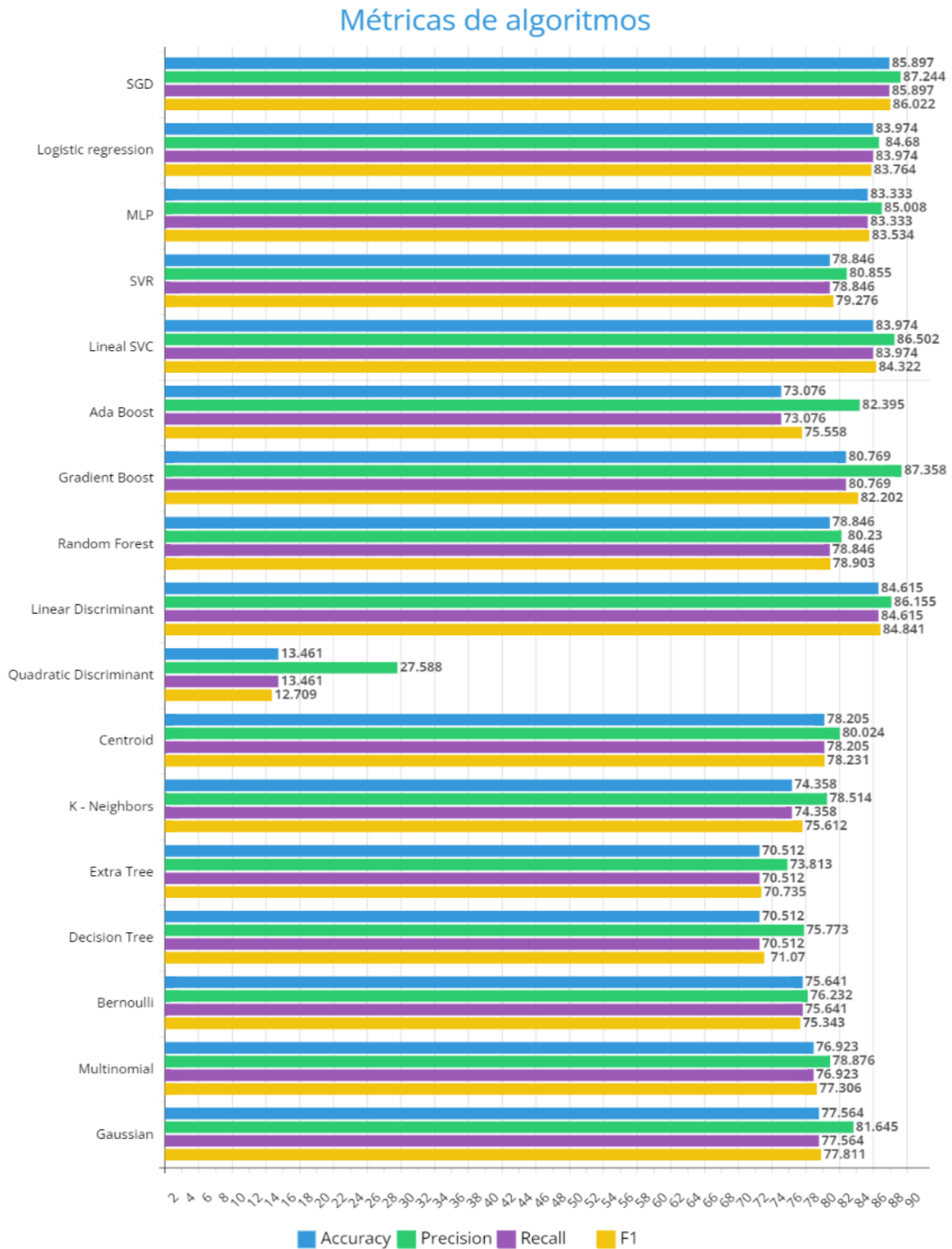


Figura 8.3.1: Resultados de los algoritmos

Estos son los resultados obtenidos tras realizar el estudio. Como se puede observar, la gran mayoría de algoritmos presentan unos resultados bastante aceptables, con la excepción del algoritmo “Quadratic Discriminant”.

Un detalle que cabe resaltar es que los valores de accuracy y recall son exactamente iguales. Esto no es un error, existen dos medidas para obtener las métricas descritas en Scikit-learn para clasificaciones no binarias, el promedio (average) **macro**, realiza estas operaciones sin tener en cuenta el desbalance de clases, y el promedio **weighted** [36], la cual si tiene en cuenta el desbalance de clases.

Se ha aplicado la opción weighted, debido a que sí, esta estructura de datos está totalmente balanceada, pero puede que utilicemos el programa con otros datos. Según Scikit-learn, cuando se aplica esta opción automáticamente el atributo recall pasa a tener el mismo resultado que la accuracy.

Puede que analizar la imagen anterior sea un poco complejo, el punto final de este apartado es mostrar cuales son los mejores algoritmos, por eso, a continuación se creará una tabla ordenando los algoritmos de mayor a menor en cuestión a rendimiento mediante la media aritmética de las cuatro métricas ya vistas.

ALGORITMO	MEDIA DE MÉTRICAS
SGD	86.265
Linear Discriminant	85.057
Lineal SVC	84.693
Logistic Regression	84.098
MLP	83.802
Gradient Boost	82.775
SVR	79.456
Random Forest	79.206
Centroid	78.666
Gaussian	78.646
Multinomial	77.507
Ada Boost	76.026
Bernoulli	75.714
K-Neighbors	75.711
Decision Tree	71.967
Extra Tree	71.393
Quadratic Discriminant	16.805

Tabla 8.3.1: Rendimiento de algoritmos ordenados

Se procede a dejar el enlace sobre el estudio de los algoritmos:

https://docs.google.com/spreadsheets/d/1Kt41LYcwoTFipK3ebmHaH9QofUPk_GUJRggFw9wk03U/edit?usp=sharing

Capítulo 9

Conclusiones y líneas futuras

9.1 Conclusiones

Lo que empezó como un caso real a resolver se ha demostrado que puede llegar a tener una buena resolución. A pesar de no tener todas las temáticas disponibles y un conjunto de datos tan extenso como algunos desearían, puede verse como los resultados son bastante coherentes.

Elementos como un buen procesamiento del lenguaje y un conjunto de datos, se ha visto que son bastante determinantes a la hora de buscar la mejor precisión posible para cualquier proyecto similar.

Es evidente que el aprendizaje automático es una herramienta que permite perfeccionar todo tipo de programas, así como de realizar acciones de forma más precisa que una persona con un gran conjunto de datos en el caso de las predicciones o clasificaciones.

Cambiando de tema, respecto al estudio realizado sobre ChatGPT, se puede corroborar que este tipo de herramientas son bastante útiles para casi cualquier tipo de ayuda en cuestión a carecer de algún tipo de conocimiento. A día de hoy puede tener alguna que otra falla como se ha podido comprobar, pero eso no quiere decir que en un futuro no sean herramientas aptas para poder hacer lo que aún no imaginamos, como ya ha ocurrido en ocasiones anteriores.

Cualquiera puede llevarse una gran impresión sobre lo rápido que avanza la tecnología, en los últimos años la mayor parte de las inversiones que se han hecho en el campo de la informática se han centrado en la inteligencia artificial, haciendo que se creen herramientas tan potentes como la estudiada y las cuales van aprendiendo a medida que van siendo utilizadas.

Desde la salida de ChatGPT, muchos controladores, plugins y programas están saliendo a la luz, dándonos a entender que probablemente esta clase de programas sea el tipo de proyectos que más se requiera en los trabajos de un posible futuro.

9.2 Líneas futuras

El proyecto desarrollado tiene múltiples oportunidades para ser extendido, tanto por la parte del software como por el conjunto de datos creado.

Para empezar, sería muy sugerente que el programa permitiera cambiar muchas más variables, como puede ser: el ajuste de los algoritmos en tiempo de ejecución por parte del usuario o la elección de los atributos que se quiere mostrar en los resultados del algoritmo al efectuar el entrenamiento. Esto, más la adición de nuevos algoritmos, podría dar mucho juego al programa, permitiendo dar una gran libertad al usuario y evitar tocar el código lo menos posible.

Otra cosa muy importante que se puede pensar es añadir nuevos formatos con los que trabajar, como puede ser JSON. Muchas personas actualmente prefieren trabajar en este formato por la estructura, facilidad y limpieza que este ofrece.

Finalmente, respecto al conjunto de datos desarrollado, sería recomendable que se siguiera trabajando en él, añadiendo nuevas temáticas e iniciativas parlamentarias para tener un conjunto mucho más completo y eficaz.

Capítulo 10

Summary and Conclusions

10.1 Summary

This project was carried out with the purpose of finding a way to automatically classify the parliamentary initiatives of the Parliament of Canary Islands through one of the most demanded branches in recent years, machine learning.

The software to be developed must be able to classify the parliamentary initiatives according to the labels that the training set presents, in this case, the labels will be based on the themes of the EuroVoc thesaurus.

Before starting development, it is necessary to carry out a thorough analysis of all the necessary tools and establish the limits.

When developing, it can be seen that this work is made up of various tasks, from developing software to creating your own data set to satisfy the knowledge that algorithms need through supervised learning. In the development of the program, great importance is given to natural language processing, since it contains a set of techniques that are very necessary to understand human language and thus facilitate predictions.

Finally, a series of evaluations have also been carried out, one of them has been about the performance of the algorithms to establish which are the best for this case, the other evaluation deals with ChatGPT, with the objective of determining if the tool it is as amazing as it is described by everyone.

10.2 Conclusions

What began as a real case to solve has been shown to have a good resolution. Despite not having all the themes available and a data set as extensive as some would like, it can be seen that the results are quite consistent.

Elements such as good language processing and a data set have been seen to be quite decisive when looking for the best possible precision for any similar project.

It is clear that machine learning is a tool that allows all kinds of programs to be perfected, as well as to perform actions more accurately than a person with a large data set in the case of predictions or classifications.

Changing the subject, regarding the study carried out on ChatGPT, it can be confirmed that these types of tools are quite useful for almost any type of help in question, even if they lack some type of knowledge. Today it may have the occasional failure as has been verified, but that does not mean that in the future they are not suitable tools to be able to do what we have not yet imagined, as has happened on previous occasions.

Anyone can get a great impression about how fast technology advances, in recent years most of the investments that have been made in the field of computing have focused on artificial intelligence, creating tools as powerful as the one studied and which are learning as they are being used.

Since the release of ChatGPT, many drivers, plugins and programs are starting to appear, giving us to understand that this class of programs is probably the type of projects that is most required in

the works of a possible future.

Capítulo 11

Presupuesto

En este apartado se realizará una estimación del presupuesto sobre este proyecto.

Análisis y diseño			
Tarea	Nº horas	Precio por hora	Precio
Análisis de requisitos y estudio de herramientas	75	9 €	675 €
Estudio de estructura de conjunto de datos	12	10 €	120 €
Realización de diseño y clases	10	10 €	100 €
Total = 895 €			

Tabla 11.1: Presupuesto de análisis y diseño

Implementación y desarrollo			
Tarea	Nº horas	Precio por hora	Precio
Desarrollo de conjunto de datos	46	11 €	506 €
Desarrollo de software	64	13 €	832 €
Resolución de errores	33	13 €	429 €
Documentación	10	10 €	100 €
Configuración de algoritmos	23	13 €	299 €
Pruebas	30	13 €	390 €
Total = 2556 €			

Tabla 11.2: Presupuesto de implementación y desarrollo

Presupuesto total del proyecto = $895 + 2556 = 3451$ €

Bibliografía

1. Barrena, J. C. Una fotografía por inteligencia artificial gana un concurso y desata la polémica [en línea]. Recuperado de <https://www.heraldo.es/noticias/ocio-y-cultura/2023/04/22/fotografia-creada-inteligencia-artificial-gana-concurso-desata-polemica-1646683.html> (Último acceso 05/07/2023)
2. Iberdrola. Que es el 'Machine Learning' [en línea]. Recuperado de [https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatizado#:~:text=El%20Machine%20Learning%20es%20una,elaborar%20predicciones%20\(an%C3%A1lisis%20predictivo\)](https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatizado#:~:text=El%20Machine%20Learning%20es%20una,elaborar%20predicciones%20(an%C3%A1lisis%20predictivo)) (Último acceso 05/07/2023)
3. Gobierno de España. ¿Cómo aprenden las máquinas? Machine Learning y sus diferentes tipos [en línea]. Recuperado de <https://datos.gob.es/es/blog/como-aprenden-las-maquinas-machine-learning-y-sus-diferentes-tipos#:~:text=Estos%20son%3A%20aprendizaje%20supervisado%2C%20aprendizaje,supervisado%20y%20aprendizaje%20por%20refuerzo> (Último acceso 05/07/2023)
4. Cohete Digital. ¿Utiliza Google el Machine Learning? [en línea]. Recuperado de <https://cohete.digital/guia-carrera-profesional/ingeniero-machine-learning/utiliza-google-machine-learning/> (Último acceso 05/07/2023)
5. Pérez, E. Así está apostando Amazon por el machine learning para lograr vendernos antes sus productos [en línea]. <https://www.xataka.com/robotica-e-ia/asi-esta-apostando-amazon-machine-learning-para-lograr-vendernos-antes-sus-productos> (Último acceso 05/07/2023)
6. Arguello, A. Inteligencia artificial y redes sociales (Facebook) [en línea]. <https://www.linkedin.com/pulse/inteligencia-artificial-y-redes-sociales-facebook-adrian-arguello?originalSubdomain=es> (Último acceso 05/07/2023)
7. Amazon. ¿Qué es la NLP? [en línea]. Recuperado de <https://aws.amazon.com/es/what-is/nlp/> (Último acceso 06/07/2023)
8. Amazon. ¿Qué es Python? [en línea]. Recuperado de <https://aws.amazon.com/es/what-is/python/> (Último acceso 05/07/2023)
9. DataScientest. Pandas: La biblioteca de Python dedicada a la Data Science [en línea]. Recuperado de <https://datascientest.com/es/pandas-python#:~:text=La%20biblioteca%20de%20software%20de,manipular%20o%20incluso%20fusionar%20datos> (Último acceso 06/07/2023)
10. JesusLC. Procesamiento del lenguaje natural con Python (NLTK) [en línea]. Recuperado de <https://jesuslc.com/2012/11/28/procesamiento-del-lenguaje-natural-con-python-nltk/> (Último acceso 06/07/2023)

11. Gobierno de España. Probamos Spacy: mucho más que una librería para crear proyectos reales de procesamiento del lenguaje natural [en línea]. Recuperado de <https://datos.gob.es/es/blog/probamos-spacy-mucho-mas-que-una-libreria-para-crear-proyectos-reales-de-procesamiento-del> (Último acceso 06/07/2023)
12. Itop. Scikit-learn [en línea]. Recuperado de <https://www.itop.es/soluciones-tecnologicas/business-analytics-business-intelligence/scikit-learn.html> (Último acceso 05/07/2023)
13. Aguilar, J.M. ¿Qué es el patrón MVC en programación y por qué es útil? [en línea]. Recuperado de <https://www.campusmvp.es/recursos/post/que-es-el-patron-mvc-en-programacion-y-por-que-es-util.aspx> (Último acceso 07/07/2023)
14. Template Method [en línea]. Recuperado de <https://refactoring.guru/es/design-patterns/template-method> (Último acceso 07/07/2023)
15. Oracle. ¿Qué es el procesamiento del lenguaje natural (NLP)? [en línea]. Recuperado de <https://www.oracle.com/es/artificial-intelligence/what-is-natural-language-processing/> (Último acceso 07/07/2023)
16. Urdaneta Fernández, L. A. Reducir el número de palabras de un texto: lematización y radicalización (stemming) con Python [en línea]. Recuperado de <https://medium.com/qu4nt/reducir-el-n%C3%BAmero-de-palabras-de-un-texto-lematizaci%C3%B3n-y-radicalizaci%C3%B3n-stemming-con-python-965bfd0c69fa> (Último acceso 07/07/2023)
17. Kaplan, D. Machine Learning 101: CountVectorizer Vs TFIDFVectorizer [en línea]. Recuperado de <https://enjoymachinelearning.com/blog/countvectorizer-vs-tfidfvectorizer/#:~:text=CountVectorizer%20simply%20counts%20the%20number,is%20to%20the%20whole%20corpus.> (Último acceso 08/07/2023)
18. Recuero de los Santos, P. Datos de entrenamiento vs datos de test [en línea]. Recuperado de <https://telefonicatech.com/blog/datos-entrenamiento-vs-datos-de-test> (Último acceso 07/07/2023)
19. Nooney, K. Deep into multi-label classification..! (With detailed Case Study) [en línea]. Recuperado de <https://towardsdatascience.com/journey-to-the-center-of-multi-label-classification-384c40229bff> (Último acceso 07/07/2023)
20. Página oficial de EuroVoc. [en línea]. Recuperado de <https://eur-lex.europa.eu/browse/eurovoc.html?locale=es> (Último acceso 07/07/2023)
21. Gobierno de España. EUR-Lex [en línea]. Recuperado de <https://www.miteco.gob.es/es/ceneam/recursos/paq-web/portales-buscadores/eurlex.aspx> (Último acceso 07/07/2023)

22. E. Sánchez-Nielsen, A. Morales, O. Mendo and F. Chávez-Gutiérrez, "SuDaMa: Sustainable Open Government Data Management Framework for Long-Term Publishing and Consumption," in IEEE Access, vol. 9, pp. 151841-151863, 2021, doi: 10.1109/ACCESS.2021.3127472.
23. Allwright, S. *What is good accuracy score in machine learning?* [en línea]. Recuperado de <https://stephenallwright.com/good-accuracy-score/> (Último acceso 07/07/2023)
24. *Página principal de ChatGPT* [en línea]. Recuperado de <https://openai.com/chatgpt> (Último acceso 08/07/2023)
25. Martínez Heras, J. *Precision, Recall, F1 y Accuracy en clasificación* [en línea]. Recuperado de <https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/> (Último acceso 09/07/2023)
26. Scikit-learn. *Clases de Scikit-learn* [en línea]. Recuperado de <https://scikit-learn.org/stable/modules/classes.html> (Último acceso 11/07/2023)
27. Sharma, P. *Implementation of Gaussian Naive Bayes in Python Sklearn* [en línea]. <https://www.analyticsvidhya.com/blog/2021/11/implementation-of-gaussian-naive-bayes-in-python-sklearn/> (Último acceso 09/07/2023)
28. IBM. *¿Qué es un árbol de decisión?* [en línea]. <https://www.ibm.com/es-es/topics/decision-trees> (Último acceso 09/07/2023)
29. Scikit-learn. *NearestCentroid* [en línea]. <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestCentroid.html> (Último acceso 09/07/2023)
30. Heras Martínez, J. *Random Forest (Bosque Aleatorio): combinando árboles* [en línea]. <https://www.iartificial.net/random-forest-bosque-aleatorio/> (Último acceso 09/07/2023)
31. Amat Rodrigo, J. *Gradient Boosting con Python* [en línea]. https://cienciadedatos.net/documentos/py09_gradient_boosting_python (Último acceso 09/07/2023)
32. Universidad Nacional de Rosario. *Aplicación de algoritmo Boosting adaptativo (ADABOOST) a un problema de clasificación automática de textos* [en línea]. <https://rehip.unr.edu.ar/handle/2133/14282> (Último acceso 09/07/2023)
33. Amat Rodrigo, J. *Máquinas de Vector Soporte (SVM) con Python* [en línea]. <https://cienciadedatos.net/documentos/py24-svm-python> (Último acceso 09/07/2023)
34. Amazon. *¿Qué es la regresión logística?* [en línea]. <https://aws.amazon.com/es/what-is/logistic-regression/> (Último acceso 09/07/2023)

35. Döring, M. *Linear, Quadratic and Regularized Discriminant Analysis* [en línea]. <https://www.datascienceblog.net/post/machine-learning/linear-discriminant-analysis/> (Último acceso 09/07/2023)

36. Scikit-learn. *Recall_score* [en línea]. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html (Último acceso 09/07/2023)