Universidad Internacional de Valencia

Análisis de ontologías basado en modelos del lenguaje: ¿Se dice lo mismo o falta algo?

Titulación:
Master en Inteligencia Artificial

Curso académico
2022-2023

Alumna: Oliveros Félez, Karen
D.N.I: 73107621M

Director de TFM: Bobed Lisbona, Carlos
Ponente: Colomer Granero, Adrián

Convocatoria:

Tercera

23 Octubre 2023

De:
Planeta Formación y Universidades

En dos ocasiones me han preguntado:
“Si pone datos incorrectos en la máquina,
¿Saldrán las respuestas correctas?".
Soy absolutamente incapaz de hacerme una idea
del tipo de confusión de ideas que pueden
provocar que alguien haga una pregunta así.

Charles Babbage

Agradecimientos

Deseo expresar mi sincero agradecimiento a Carlos, quien además de ser mi tutor es un amigo y compañero de trabajo, por su inagotable comprensión y paciencia. También quiero extender mi gratitud a Enrique y Jorge, quienes me introdujeron en el ámbito de las ontologías y la representación del conocimiento. Gracias a ellos, ahora formo parte de este mundo y lo he convertido en mi forma de vida.

Finalmente, deseo subrayar su apoyo inquebrantable y afecto incondicional de Javi, mi pareja, mi familia y mis amigos de L'Badina. Ellos han sido elementos fundamentales que han hecho posible alcanzar este hito en mi carrera académica sin perder la cordura.

Índice general

Índice de figuras . . . III

Resumen . . . 1

1. Introducción . . . 3

2. Marco Teórico . . . 5
   2.1. Ontologías. . . . 5
   2.2. Inteligencia Artificial. . . . 7
      2.2.1. Sistemas de regresión. . . . 8
      2.2.2. Sistemas de clasificación. . . . 9
      2.2.3. Representaciones distribuidas del lenguaje. . . . 10

3. Estado del arte . . . 13

4. Objetivos . . . 17

5. Metodología e implementación . . . 19
   5.1. Extracción y transformación de datos. . . . 19
   5.2. Procesos de predicción utilizando las distancias semánticas calculadas. . . . 22
      5.2.1. Problema de regresión. . . . 22
      5.2.2. Problemas de clasificación. . . . 24

6. Resultados y discusión . . . 29
   6.1. Resultados del problema de regresión . . . 30
   6.2. Resultados de los problemas de clasificación . . . 33
      6.2.1. Problema de distancia cualitativa . . . 33
      6.2.2. Clasificación entre parejas de conceptos con subsunción directa o sin ella. 35
      6.2.3. Clasificación entre parejas de conceptos con subsunción o sin ella. . . . 37

7. Conclusiones . . . 41
   7.1. Problema de regresión. . . . 41

7.2. Problema de clasificación de la distancia cualitativa. 42
7.3. Problema de clasificación de parejas con subsunción directa o sin ella. 42
7.4. Problema de clasificación de parejas con o sin subsunción. 42

8. Limitaciones y
Perspectivas de Futuro 43

Lista de Acrónimos 45

Bibliografía 46

Índice de figuras

2.1. Tripletas. Ejemplo KG 7
2.2. Esquema sobre la Inteligencia Artificial 8

5.1. Diagrama de proceso de obtención de datos para entrenar 19
5.2. Esquema UML de la base de datos 20
5.3. Proceso para obtener modelos de regresión que predigan la distancia ontológica 24
5.4. Proceso para obtener modelos de clasificación multi-clase 24

6.1. Diagrama de caja de la distancia jerárquica de los datos test 29
6.2. Diagrama de caja de las profundidades de las ontologías estudiadas en los datos test 29
6.3. Diagramas de caja de las variables predictivas 30
6.4. Métricas de los residuos del modelo RFR del problema de regresión de las distancias 31
6.5. Métricas de los residuos del modelo KNN del problema de regresión de las distancias 32
6.6. Diagrama del valor predicho/valor real y matriz de confusión del problema de regresión 32
6.7. Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de la distancia cualitativa. 33
6.8. Matriz de confusión del modelo RFC del problema de clasificación de distancias 34
6.9. Matriz de confusión del modelo KNN del problema de clasificación de distancias 34
6.10. Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de subsunción directa. 36
6.11. Matriz de confusión del modelo DTC para el problema de subsunción directa . . 36
6.12. Matriz de confusión del modelo RFC para el problema de subsunción directa . . 36
6.13. Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de subsunción. 38
6.14. Matriz de confusión del modelo DTC para el problema de subsunción 39
6.15. Matriz de confusión del modelo RFC para el problema de subsunción 39

Resumen

Actualmente las ontologías tienen un papel esencial en la representación del conocimiento de un dominio de interés. Junto a los grafos de conocimiento, se utilizan para los motores de búsqueda y procesamiento del lenguaje natural, para asistentes virtuales, sistemas de recomendación, análisis de redes sociales y finanzas entre otros. Este proyecto de investigación comienza el análisis de la adecuación de un corpus de un texto a una ontología. En particular, analiza la construcción de sus jerarquías utilizando la información semántica que contienen los conceptos. Los objetivos principales son conseguir encontrar la posible existencia de "huecos" en las jerarquías de conceptos de estas ontologías, pudiendo señalar la necesidad de incorporar conceptos intermedios, y observar si un concepto podría pertenecer a una jerarquía distinta a la originalmente concebida. Mejorando, de esta manera, la representación y comprensión de la ontología dado un corpus.

Se realiza una revisión exhaustiva del estado actual del campo explorando los modelos de lenguaje utilizados en esta área. Específicamente, se utiliza el modelo Glove-Poincaré, que ha despertado un considerable interés debido a su potencial para captar, entre parejas de palabras, señales de hiperonimia o hiponimia.

En este contexto, se recopilan datos de las ontologías curadas de *Linked Open Vocabularies* (LOV) como punto de partida para utilizarlos en los experimentos de entrenamiento. Estos experimentos se enfocan en abordar cuatro problemas diferentes, cuya resolución representa un logro significativo hacia el objetivo final.

Aunque los resultados obtenidos de los experimentos no alcanzan niveles sobresalientes, arrojan luz sobre el potencial de futuras investigaciones en este dominio. Estos hallazgos sugieren que aún queda un amplio espacio para continuar la línea de investigación utilizando otras técnicas que puedan mejorar los resultados obtenidos en este trabajo.

Este proyecto sienta las bases para futuras investigaciones en el campo del análisis de ontologías basado en modelos de lenguaje, proporcionando una perspectiva valiosa que invita a explorar nuevas vías y enfoques en busca de un mayor entendimiento y optimización de las estructuras ontológicas.

Introducción 1

La representación y organización efectiva del conocimiento es un pilar fundamental en el campo de la Inteligencia Artificial y la Informática. **Hace más de 50 años se tomó el concepto de ontología en la filosofía, que se preocupa del estudio del ser y la representación de conceptos e ideas, se tradujo al ámbito de las ciencias de la computación para abordar estos desafíos**. En este contexto, las ontologías adquieren un papel esencial al representar el conocimiento de un dominio de interés en un modelo, permitiendo su comprensión por parte de una máquina. Este modelo **consiste en describir el mundo a través de un conjunto de conceptos, etiquetas, propiedades y relaciones entre los conceptos.**

Los Grafos de conocimiento o *Knowledge graphs* (KG) extienden el conocimiento factual haciendo uso de las ontologías como fundamento conceptual. Las ontologías proporcionan una estructura y un marco formal y establecen la base semántica y organizativa, mientras que los KG amplían esta estructura incorporando datos reales y específicos al mundo utilizando las relaciones semánticas definidas en las ontologías.

Esta forma de representar la información, aunque ya había sido empleada en algunos ámbitos, ganó popularidad con la aparición del KG de Google en 2012 (Singhal, 2012). Durante la última década, ha desempeñado un papel crucial en el avance de la Inteligencia Artificial y en la representación de conceptos en diversos dominios, incluyendo la salud (Choi et al., 2017), la industria (Yahya et al., 2021) y la ingeniería (Siddharth et al., 2022), entre otros. Estos grafos se utilizan para el análisis de riesgos, de fraude, para la búsqueda de información o para la visualización de todo lo que rodea a un solo elemento del grafo, proveyendo al usuario información confiable, relevante y agregada de esa única entidad.

La popularidad de estas bases de conocimiento radica en que estructuran la información de una manera lógica, similar a la manera en la que estructura una persona. Al ser bases de datos NoSQL, ofrecen una gran flexibilidad y permiten un alto grado de conexiones entre entidades dentro del propio grafo semántico. Existe la posibilidad de crear ecosistemas de datos conectados y enriquecidos utilizando KG, y es muy común reutilizar sus esquemas y adecuarlos a las necesidades del usuario.

En el contexto del Procesado del Lenguaje Natural (PLN), los KG y las ontologías están siendo clave para mejorar estos sistemas de Inteligencia Artificial (IA) (Fellbaum, 2010). Éstos

garantizan el acceso, la integración y el uso de grandes cantidades de datos (Chaudhri et al., 2022). Actualmente, en IA, incluso se generan automáticamente utilizando como entrada un conjunto de textos o corpus (Dessì et al., 2020).

Sin embargo, existen muy pocas herramientas que ayuden al técnico que valida o genera esas ontologías a comprobar que la granularidad de su jerarquía de conceptos es correcta en un corpus específico o si debería añadir algún concepto más que la complete.

Es por ello, que **este trabajo se centra en utilizar diferentes distancias semánticas asociadas a las etiquetas de los conceptos para comprobar si se puede conseguir, o no, aprender las distancias jerárquicas a través de sólo la información semántica de sus etiquetas.** Y así, poder detectar distancias entre conceptos demasiado grandes que sugieran que falta alguna entidad intermedia para completar el grafo de conocimiento y detectar conceptos que puedan pertenecer a otras ramas jerárquicas.

Marco Teórico

2

En esta sección se va a describir las tecnologías y herramientas que se utilizarán a lo largo de la tesis. Se hará especial hincapié en las ontologías y en los sistemas de aprendizaje supervisado que se han utilizado para el desarrollo del mismo.

2.1. Ontologías.

Una **ontología es una especificación explícita de una conceptualización**, una visión abstracta y simplificada de una realidad que se desea representar para algún propósito (Gruber, 1995).

El universo de discurso es el conjunto de objetos que pueden ser representados en un formalismo declarativo de un dominio. Este conjunto de objetos y las relaciones descriptibles entre ellos se reflejan en el vocabulario de representación con el que un programa basado en el conocimiento lo representa. Por lo tanto, en el contexto de la IA, podemos describir la ontología de un dominio definiendo un conjunto de términos de representación. En las ontologías, las definiciones asocian los nombres de conceptos en el universo de discurso (por ejemplo, clases, relaciones, funciones u otros objetos) con un texto que describe lo que significan los nombres, y axiomas formales que limitan la interpretación y el uso bien formado de estos términos. Formalmente, una ontología es la declaración de una teoría lógica.

Las ontologías **deben ser claras y las definiciones de sus conceptos deben ser objetivas**, independientes del contexto social y computacional. Además, deben ser **coherentes**, permitiendo inferencias que sean consistentes con las definiciones, diseñarse anticipando los usos del vocabulario compartido y deben poderse extender de manera monótona.

La conceptualización debe especificarse a nivel del conocimiento sin depender de una codificación de nivel de símbolos particular y debe respaldar las actividades previstas de intercambio de conocimiento (Bobed Lisbona, 2013).

Los elementos principales de las ontologías son:

- **Conceptos** o clases: Son representaciones abstractas de una idea, categoría o elemento en un contexto. Los conceptos en una ontología organizan y representan el conocimiento de manera estructurada, permitiendo establecer relaciones y jerarquías entre diferentes tipos de conceptos. Por conveniencia, en muchos formalismos se considera que existe

un concepto que engloba todas las clases existentes, el concepto *Thing*. Así mismo, se define un concepto que es subclase de todas las clases y no puede ser superclase de ninguna, el concepto *Nothing*.

- **Relaciones** o propiedades: Son representaciones de la interacción entre dos conceptos. Cada concepto tendrá un rol específico en la relación, uno de ellos será el dominio y otro el rango.
- **Instancias**: representan individuos pertenecientes a un concepto.

El lenguaje estándar propuesto por *World Wide Web Consortium* (W3C) para formalizar una ontología es *Ontology Web Language* (OWL). Está basado en Lógicas Descriptivas (DL), un lenguaje de representación de conocimiento basado en lógica de predicados diseñado para capturar relaciones y conceptos complejos en ontologías de manera precisa y estructurada. Es capaz de representar relaciones entre clases complejas como la disyunción, cardinalidad o igualdad, entre otros.

Los KG son bases de conocimiento que tienen el énfasis en el conocimiento factual. Las ontologías son fundamentales en la semántica formal de un grafo de conocimiento, desempeñando un papel similar al esquema de datos, estableciendo claramente el significado de los datos presentes en él. Según Ehrlinger y Wöß (2016), el lenguaje más extendido para representar los KG en un lenguaje computacional es *Resource Description Framework* (RDF), diseñado por W3C. Esta familia de especificaciones hace declaraciones sobre las entidades en forma de conjuntos ordenados llamados tripletas. Todo elemento en el KG y en la ontología necesita un identificador único, en la implementación en RDF el identificador tiene forma de *Uniform Resource Identifier* (URI), característica crítica para un KG para evitar ambigüedades. Si U es el conjunto de URI y L el conjunto de literales, una tripleta es un conjunto (s, p, o) donde $s \in U$, $p \in U$ y $o \in U \cup L$.

Un vocabulario RDF es una colección de URIs que se van a usar en un KG. Por ejemplo ‹*http://www.w3.org/2000/01/rdf-schema#*› es la URI que la W3C utiliza para denotar el vocabulario del esquema. Se pueden organizar en espacios de nombres o *namespaces*, abreviándose a través de un prefijo. En este caso su forma corta es **rdfs:**. De esta manera la URI ‹*http://www.w3.org/2000/01/rdf-schema#subClassOf*› tendrá la forma *rdfs:subClassOf*.

El espacio de nombres **rdfs:** pertenece al lenguaje de descripción de vocabularios en RDF, RDF Schema (RDFS). A través de este lenguaje se puede representar de una manera más sencilla que en OWL una ontología y puede ser empleada por un KG.

Este trabajo se centra en los recursos tipo *rdfs:Class*, que representan conceptos como por ejemplo “Perro” o “Edificio”, y en las propiedades *rdfs:Property*, que definen una relación entre recursos que están como sujeto y recursos que están como objeto.

Las propiedades (*rdfs:Property*) que se van a emplear en este trabajo son *rdfs:label* y *rdfs:subClassOf*. La primera proporciona una etiqueta clara del nombre de una instancia o con-

cepto, aportando así información semántica. La segunda indica **la relación de subsunción** entre conceptos, definida por la W3C; esta propiedad indica que **todas las instancias de la primera clase, la que está como sujeto, son instancias también de la otra**.

En la Figura 2.1 se muestra un conjunto de tres tripletas obtenidas de Wikidata[1], que se corresponden a cada fila de la tabla. El sujeto y el predicado en la primera tripleta son recursos de tipo *rdfs:Class* y representan a las clases *:Perro* y *:Animal* respectivamente. Esta tripleta indica que existe relación de subsunción entre los dos conceptos puesto que su predicado es *rdfs:subClassOf*, recurso de tipo *rdfs:Property*. La segunda tripleta indica que el recurso con la URI *wd:Q186486*, instancia de Wikidata[2], pertenece a la clase *:Perro*. La tercera tripleta, indica que la etiqueta de ese recurso es *Hachikō*.

sujeto	predicado	objeto
:Perro	rdfs:subClassOf	:Animal
wd:Q186486	rdf:type	:Perro
wd:Q186486	rdfs:label	Hachikō

Figura 2.1: ***Ejemplo de tripletas.***

Para capturar las distancias entre conceptos, se va a utilizar **distancia jerárquica**, d^H, que se define como el número mínimo de conceptos que deben ser visitados, utilizando la propiedad *rdfs:subClassOf*, para llegar del primero al segundo. Esta distancia solo es posible hallarla en el caso de que dos entidades pertenezcan a la misma jerarquía.

Se define la **distancia ontológica** d entre dos conceptos a y b de manera que

$$d(a,b) = d^H(a, LCA) + d^H(b, LCA)$$

En la que LCA es el ancestro común más cercano a ambos.

2.2. Inteligencia Artificial.

"La Inteligencia Artificial (IA) es el estudio de cómo hacer que los ordenadores realicen tareas en las que, hasta el momento, las personas son mejores" (Rich, 1985). Según la Real Academia de la Lengua, la Inteligencia Artificial es aquella disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o razonamiento lógico.

En Chowdhary (2020) **se define la IA como una rama de las Ciencias de Computación enfocada en la automatización de comportamientos inteligentes.** Estos sistemas de computación avanzada se han estado desarrollando durante varias décadas, pero en las últimas dos décadas se han conseguido grandes avances debido a la mejora y desarrollo del hardware de computación. Se han implantado en muchos ámbitos, como en la ciencia médica,

[1] https://www.wikidata.org/
[2] https://www.wikidata.org/wiki/Q186486

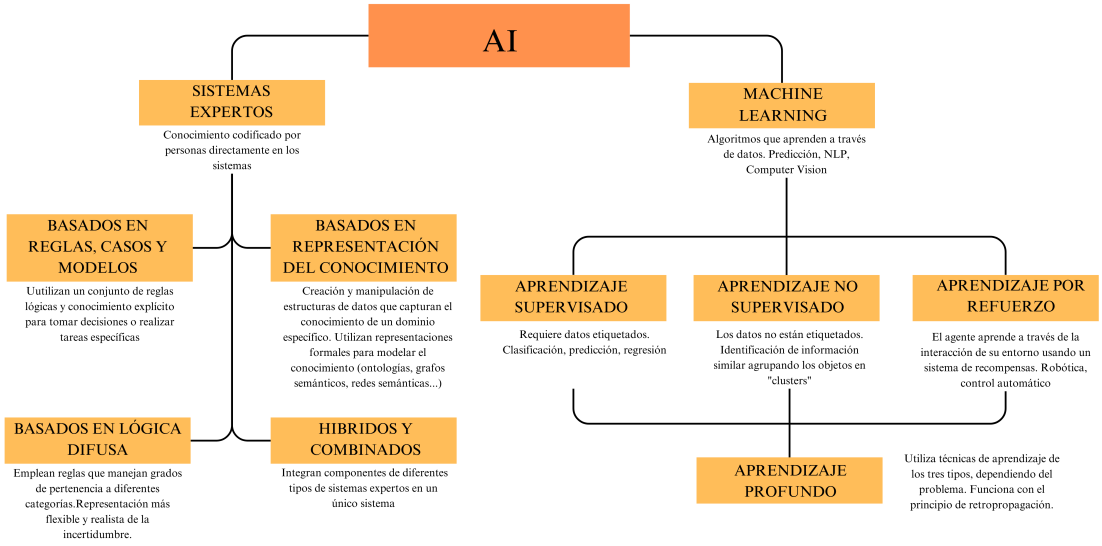

Figura 2.2: ***Esquema sobre la Inteligencia Artificial*** *(Date, 2019)*

en procesos de manufacturación, publicidad, vehículos, domótica... La IA se divide en muchos sub-campos en los que se utiliza gran variedad de técnicas. Uno de esos campos es el Aprendizaje Automático o *Machine Learning* (ML), en este campo se utilizan datos del entorno para que el sistema desarrollado aprenda.

Tal y como se muestra en la Figura 2.2 el ML se divide en tres categorías. En este trabajo se van a utilizar aquellos sistemas que requieren datos etiquetados, específicamente los de aprendizaje supervisado. También se van a utilizar técnicas de PLN, que consisten en la recuperación de información en texto, traducción, creación de resúmenes y preguntas y respuestas automáticas en un corpus. En este caso, se va a hacer uso de las representaciones distribuidas del lenguaje generadas a través de algoritmos de aprendizaje no supervisado.

2.2.1. Sistemas de regresión.

Los sistemas de regresión abordan la tarea de predecir valores numéricos continuos basados en patrones y relaciones encontradas en los datos. Estos sistemas permiten modelar y comprender cómo diferentes variables influyen en una salida numérica, siendo fundamentales en aplicaciones que involucran pronósticos, estimaciones y análisis de tendencias. La regresión es una herramienta poderosa en campos como el análisis de datos, la economía, la ciencia e ingeniería, brindando conocimiento valioso basado en los datos para la toma de decisiones informadas y la comprensión de fenómenos complejos.

En este trabajo se utilizarán algoritmos de regresión supervisados, en este caso, buscarán minimizar el error entre las predicciones del modelo y los valores reales en el conjunto de entrenamiento. Se utilizaran distintos modelos para realizar regresión como la Regresión Lineal, la Regresión Polinómica, regresión basada en Árboles de Decisión (*Decision Tree Regressor*), regresión *Random Forest*, rRegresión (*Gradient Boosting Regressor* (GBC) y regresión *K*-

Nearest Neighbors (KNN).

La **Regresión Lineal** tiene como objetivo predecir un valor numérico continuo basado en una o mas variables independientes, en el caso de la **Regresión Polinómica** se intenta modelar relaciones no lineales al introducir términos polinómicos. Los ***Decision Tree Regressor*** predicen un valor numérico continuo dividiendo el espacio de características en regiones con valores de salida homogéneos y consigue capturar relaciones no lineales y complejas. La **Regresión *Random Forest*** utiliza varios árboles de decisión para realizar predicciones y luego promedia o toma la mediana en las predicciones de todos los árboles. La técnica ***Gradient Boosting Regressor*** construye un modelo aditivo por etapas permitiendo la optimización de funciones de pérdida diferenciables arbitrarias; en cada etapa se ajusta un árbol de regresión al gradiente negativo de la función de pérdida dada. Finalmente, la **Regresión K-Nearest Neighbors** consiste en predecir el valor basado en las K instancias más cercanas.

Para evaluar la eficacia de un modelo de regresión, se utilizan métricas como el error cuadrático medio (MSE), el coeficiente de determinación (R^2) y los gráficos de residuos:

- **El error cuadrático** es $MSE = \frac{\sum(y_i - \widehat{y}_i)^2}{n}$ donde y_i representan los valores reales , $\widehat{y}_i$ representan los valores predichos y n el número total de observaciones. Un valor bajo indica un mayor ajuste del modelo a los datos.
- **El coeficiente de determinación** R^2 evalúa qué proporción de la variabilidad del elemento a predecir puede ser explicada por las variables independientes en el modelo. Un modelo con R^2 cercano a uno, indicaría una buena explicación de la variabilidad, y por tanto, de la salida.
- **Los residuos de la predicción** deben tener una distribución normal, ser aleatorios e independientes.

2.2.2. Sistemas de clasificación.

Los sistemas de clasificación en Inteligencia Artificial son algoritmos y técnicas fundamentales que se utilizan para categorizar o etiquetar datos en clases o categorías predefinidas.

En este trabajo, se empleará el enfoque de aprendizaje supervisado para utilizar los sistemas de clasificación, donde se entrena un modelo utilizando un conjunto de datos etiquetado, es decir, datos en los que ya se conoce la clase a la que pertenecen. Algunos de los algoritmos de clasificación más utilizados en el aprendizaje supervisado incluyen Árboles de decisión, **Random Forests**, **Gradient Boosting Classifier** y *K-Nearest Neighbors* análogos a los explicados en la Subsección 2.2.1 pero con un problema de clasificación.

La evaluación adecuada de los sistemas de clasificación es esencial para garantizar su rendimiento óptimo. Para ello, se aplican técnicas de validación cruzada para medir el rendimiento de los modelos de manera robusta. Se utilizan métricas de evaluación como precisión,

recuperación o *recall*, *F1-score* y la matriz de confusión

- **La exactitud o *accuracy*** mide la proporción de instancias correctamente clasificadas de todas las clases entre el número total de casos examinados.

$$Acc = \frac{\text{Verdaderos Positivos} + \text{Verdaderos Negativos}}{\text{Verdaderos Positivos} + \text{Falsos Positivos} + \text{Verdaderos Negativos} + \text{Falsos Negativos}}$$

- **La precisión** mide la proporción de instancias correctamente clasificadas de una clase específica en relación con todas las instancias clasificadas como esa clase.

$$P = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Positivos}}$$

- **La exhaustividad o *recall*** representa la proporción de instancias correctamente clasificadas de una clase específica en relación con todas las instancias clasificadas como esa clase.

$$R = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Negativos}}$$

- **El F1-score** es la media armónica de precisión y recuperación. Proporciona un equilibro entre ambas métricas y es útil cuando las clases están desequilibradas.

$$F1 = \frac{2 \cdot P \cdot R}{P + R}$$

- **La matriz de confusión** es una tabla que muestra la cantidad de predicciones correctas e incorrectas para cada clase en un problema de clasificación.

2.2.3. Representaciones distribuidas del lenguaje.

En la Sección 2.2 se da una breve definición de PLN. En este apartado se va a hablar de las diferentes técnicas y modelos que se utilizan para la representación del lenguaje.

Las técnicas de representación del lenguaje o técnicas de *word embedding* que vamos a usar toman cada palabra como unidades atómicas indivisibles que no tienen relación entre sí. Inicialmente, se emplea *One-Hot Enconding*, en la que una palabra es representada por un vector del tamaño de nuestro vocabulario, que tiene en todas las posiciones ceros excepto en la i-ésima posición, posición que corresponde a la palabra en cuestión.

Para representar documentos o frases, las técnicas *bag-of-words* (BOW) no tienen en cuenta el contexto en el que están las palabras, simplemente se representan las mismas en forma vectorial. En el caso de que se necesite el contexto, se utilizan técnicas que utilizan cadenas de Markov, como *N-gram*. En estos últimos se tienen en cuenta n unidades adyacentes, ya sea la unidad una letra, una palabra o sílaba. Este tipo de modelos necesitan una gran cantidad de textos para estimar las probabilidades de que una unidad sea adyacente a las n siguientes y

así encontrar su distribución de probabilidad. Éstos últimos son muy útiles para la generación de lenguaje natural o introducción de texto predictivo.

En el artículo Mikolov et al. (2013) se describe el modelo Word2Vec, en el que se emplea la técnica *Skip-gram* o *C-Bow* y una red neuronal que aprende representaciones vectoriales de palabras basadas en sus contextos, permitiendo calcular una representación que captura su semántica siguiendo la hipótesis de la semántica distribucional de Harris. Este modelo asigna vectores a palabras individuales en un espacio semántico y, al utilizar su información contextual, palabras similares en términos del contexto tienen representaciones vectoriales cercanas entre sí. Este modelo es especialmente bueno en analogías de palabras o relaciones sintácticas. GloVe (Pennington et al., 2014), *Global Vectors for Word Representation* es un algoritmo de aprendizaje no supervisado que obtiene representaciones vectoriales de las palabras utilizando las estadísticas de co-ocurrencia de las palabras en un corpus. Con esto, se genera una matriz simétrica en la que cada elemento representa con qué frecuencia dos palabras co-ocurren en una ventana de contexto específica. Posteriormente, esta matriz se utiliza para capturar las relaciones semánticas y contextuales entre las palabras, lo que permite que las representaciones aprendidas reflejen de manera efectiva el significado y las asociaciones de las palabras en el lenguaje natural.

En ambos casos son modelos estáticos ya que por cada palabra tenemos una única representación densa de la misma. Sin embargo, existen otros modelos dinámicos como BERT, los cuales obtienen un vector de la palabra dependiendo del contexto en el que aparezcan.

Este trabajo se va a centrar en los modelos estáticos ya que son mas sencillos de conseguir dado un corpus de documentos. En concreto se va a usar distintas implementaciones de GloVe entrenadas en distintos conjuntos de datos, provistos por Gensim[3] y Spacy[4].

En la literatura (Budanitsky y Hirst, 2006) se utiliza como **medida de similitud semántica la distancia coseno o similitud coseno** S_c. Se calcula a través de las representaciones vectoriales de las etiquetas (ambos vectores distintos del vector nulo) definidos en un espacio vectorial con producto interno y deriva del producto vectorial euclídeo $\mathbb{R}$. Dados $\bar{a}, \bar{b} \in \mathbb{R}^k$,

$$S_c(\bar{a}, \bar{b}) = \cos(\theta) = \frac{\bar{a} \cdot \bar{b}}{||\bar{a}|| \cdot ||\bar{b}||}$$

Otras medidas de similitud de vectores definidos en un espacio vectorial con producto interno son la distancia euclídea, derivada del teorema de Pitágoras, o la distancia de Minkowski. Dados $\bar{a} = (a_1, \ldots, a_k), \bar{b} = (b_1, \ldots, b_k) \in \mathbb{R}^k$,

La **distancia euclídea** se define como

$$S_e(\bar{a}, \bar{b}) = ||\bar{a} - \bar{b}|| = \sqrt{\sum_i (a_i - b_i)^2}$$

[3] https://pypi.org/project/gensim/
[4] https://spacy.io/

La **distancia de Minkowski** de orden p, donde p es un entero, es

$$S_{mp}(\bar{a}, \bar{b}) = \left(\sum_{i=1}^{k} |a_i - b_i| \right)^{\frac{1}{p}}$$

Estado del arte

3

Desde que en 2012 se extendió la popularidad de los grafos de conocimiento, el número de ontologías disponibles se ha visto engrosado año a año. Cada ontología tiene su propio diseñador, su propio área de conocimiento y nivel de detalle, y la interoperabilidad entre estas ontologías puede ser muy complicada. Por ello, el alineamiento de ontologías es crucial para llevar a cabo la homogeneización de la Web Semántica (Tounsi Dhouib et al., 2019). Lo que se quiere conseguir con el alineamiento de ontologías es averiguar la relación que tiene cada ontología con el resto y unirlas a través de relaciones jerárquicas o de equivalencia. Hay dos técnicas principales de alineamiento de ontologías:

- **A nivel elemento**: en la que se intenta descubrir la similitud entre elementos a través de su información léxica (normalmente sus etiquetas).
- **A nivel estructural**: en la que se intenta determinar su similitud basando su análisis en el vecindario de dos elementos.

Para definir la naturaleza de la relación entre dos clases, en Tounsi Dhouib et al. (2019) se define el concepto de clúster en este contexto. **Un clúster es un conjunto de vectores de representación de etiquetas que están directamente o indirectamente subsumidos por la misma clase raíz o concepto**. Las clases representadas por los miembros del clúster están más estrechamente relacionadas entre sí que cualquier otra clase, porque comparten un ancestro común. Para decidir la naturaleza de las relaciones entre clases (jerárquica o de equivalencia) se compara el radio de su respectivo clúster de los vectores del *word embedding* o de la representación de palabras en un espacio multidimensional.

Utilizan *word embeddings* para tener en cuenta la similitud semántica y sintáctica entre palabras. En este artículo extraen la información léxica de las clases y buscan relaciones de equivalencia entre ellas basándose en ellos. Es decir, utilizan el corpus para extraer una ontología de él y **utiliza el radio y la distancia al centroide del clúster y un umbral determinado empíricamente para encontrar relaciones jerárquicas y de equivalencia entre conceptos**. También consiguen una precisión en la similitud de 0.83 y una exhaustividad de 0.69 pero no es eficiente a la hora de determinar relaciones jerárquicas entre ontologías, ya que ese no era su cometido original.

Siguiendo la investigación del artículo anterior, en el artículo Tounsi Dhouib et al. (2021) también se extrae la información léxica y estructural y se generan las representaciones en

word embeddings utilizando fastText[1]. Para aquellas clases que constan de más de una palabra en su etiqueta, se hace la media entre los vectores que representan cada palabra para obtener el vector de representación de esa clase. La métrica que utilizan para observar la correspondencia entre dos clases es la similitud coseno, y se define que existe correspondencia entre dos clases si esta medida es menor que un umbral determinado empíricamente. En este artículo, para la relación de subsunción, se consigue una precisión entre el 0.04 y el 1 y una exhaustividad de entre el 0.12 y el 0.84.

En Nickel y Kiela (2017) dan una alternativa a los ***word embeddings* utilizados en la literatura anterior para aprender representaciones jerárquicas de datos simbólicos creando un *embedding* en el espacio hiperbólico de Poincaré.** La idea surge debido a que los métodos de *embedding* habituales no pueden capturar la complejidad de las relaciones y patrones sin aumentar exponencialmente la dimensionalidad del modelo. Sin embargo, los *embeddings* no euclídeos podrían mitigar este problema. En este artículo se centra en representar grandes conjuntos de datos cuyos objetos se pueden organizar en una jerarquía latente entre los mismos, es decir, capturan el *embedding* utilizando la información ontológica. Ya esta demostrado que cualquier árbol finito puede ser incrustado en un espacio hiperbólico de manera que las distancias se preservan aproximadamente. Los resultados para los experimentos de reconstrucción de jerarquías en el espacio de Poincaré con un *embedding* de dimensonalidad 200 con una precisión media de 0.87 supera a otros con la misma dimensionalidad para grandes taxonomías.

Inspirados en el artículo anterior, en Tifrea et al. (2018) se propone un *word embedding* basado en un producto cartesiano de espacios hiperbólicos con los que se conectan con los *word embeddings* Gausianos y su geometría de Fisher. De este modo, enriquecen el espacio euclídeo de la representación de Glove utilizando los espacios hiperbólicos de Poincaré.

Dado que se encuentran en un espacio hiperbólico de dimensión k, se denota como $\mathbb{H}^k$. Es un espacio homogéneo con curvatura constante negativa, y es difeomorfo (homeomorfismo diferenciable con inversa diferenciable) a $\mathbb{R}^k$. En concreto se utiliza el modelo de la bola de Poincaré, donde se establece una biyección con la bola unidad

$$\mathbb{D}^k = \left\{ (x_1, \ldots, x_k) \,|\, \sqrt{x_1^2 + \ldots + x_k^2} = 1 \right\}$$

La similitud Poincaré en este espacio se define como

$$S_{\mathbb{D}^k}(\bar{a}, \bar{b}) = 1 + \lambda_a \lambda_b \frac{||\bar{a} - \bar{b}||_2^2}{2}$$

Donde $\lambda_a \equiv \frac{2}{1-||a||}$

En particular, se utiliza la distancia definida en el espacio del producto de p bolas, $(\mathbb{D}^k)^p$. Extendiendo la distancia al producto vectorial:

[1] https://fasttext.cc/

$$S_{(\mathbb{D}^k)^p}(\bar{a}, \bar{b})^2 = \sum_{i=1}^{p} S_{\mathbb{D}^k}(\bar{a_i}, \bar{b_i})^2$$

También se utiliza una distancia análoga a la distancia coseno, la distancia girocoseno pero utilizando las propiedades del espacio vectorial hiperbólico.

Utilizando estas últimas distancias en sus experimentos, consiguen precisiones del 0.79, que superan ampliamente el estado del arte en ese momento. Se destaca el modelo entrenado que se utilizará en los experimentos de este trabajo, "poincare_glove_100D_cosh-dist-sq_init_trick.txt" y al que se hará referencia como modelo Glove-Poincaré.

No consta ningún trabajo que se centre única y exclusivamente en medir la adecuación o calidad de una ontología a partir de un corpus. En el siguiente capítulo se describe el objetivo principal del proyecto completo y en específico de este trabajo.

Objetivos

4

El objetivo último de este proyecto es analizar la adecuación de un corpus de un texto a una ontología. El primer paso es ver si con estas herramientas se pueden detectar huecos en la jerarquía de una ontología dada y asegurar la pertenencia de una clase a su jerarquía. Para ello se debe conseguir averiguar si existe algún tipo de representación semántica de las etiquetas de los conceptos en una ontología que pueda servir para, a través de técnicas que se sirven de IA, encontrar información sobre la jerarquía entre dos conceptos. Este primer hito se puede dividir en:

1. **Extracción de datos.** La extracción de datos es uno de los objetivos principales del trabajo. Es necesario extraer la suficiente cantidad de etiquetas y distancias jerárquicas entre conceptos como para poder abarcar los siguientes objetivos del proyecto. En esta fase se filtrarán aquellas parejas en las que una de las etiquetas sea el concepto “Thing” y aquellas parejas en las que se repita etiqueta. Se generarán, a través de consultas. dos conjuntos de datos:
    - Conjunto de parejas de conceptos que pertenecen a la misma rama ontológica, es decir, tienen relación de subsunción y por tanto uno de los dos es el Ancestro Común más Bajo o *Lowest Common Ancestor* (LCA).
    - Conjunto de parejas de conceptos que no pertenecen a la misma rama ontológica, es decir, no tienen relación de subsunción y por tanto ninguno de los dos es el LCA.

2. **Transformación de etiquetas y cálculo de distancias semánticas.** Este objetivo consiste en transformar las etiquetas de las ontologías a distintos espacios vectoriales semánticos de manera que se puedan calcular distancias entre tales vectores.

3. **Entrenamiento de regresión con las distancias ontológicas calculadas, tomándolas como variable continua.** Se observará si es posible conseguir una predicción de la distancia jerárquica a través de ellas tomando como la distancia jerárquica una variable continua.

4. **Entrenamiento de clasificación con las distancias ontológicas.** En esta sección se abordarán varios problemas utilizando las distancias ontológicas calculadas entre dos conceptos.

- Determinar a qué distancia están dos conceptos concretos de manera cualitativa agrupando los registros en tres clases.
- Determinar si dos conceptos tienen una relación de subsunción directa o no, es decir, que estén a distancia uno.
- Determinar si dos conceptos tienen relación de subsunción o no. Es decir, ver si están en la misma rama jerárquica. Esto es, ver si el LCA es alguno de la pareja.

Metodología e implementación

5

En este capítulo se va a detallar la metodología empleada para conseguir los objetivos anteriormente descritos y la implementación que se ha hecho para conseguirlos.

5.1. Extracción y transformación de datos.

En este apartado se va a describir el proceso que se ha seguido para la obtención de los datos de entrenamiento y test.

La primera etapa del proceso, tal y como se muestra en el primer punto de la Figura 5.1 es hacer una búsqueda y descarga de distintas ontologías publicadas en la Web. Los resultados de este trabajo se han entrenado con datos extraídos de *Linked Open Vocabularies* (LOV)[1], un catálogo de vocabularios disponibles para reutilizar con el objetivo de describir la información en la Web. Recoge definiciones, clases y propiedades en distintos dominios y agrupados en distintos vocabularios o contextos que conforman ontologías. Las ontologías publicadas en LOV son aquellas publicadas por entidades como W3C, bibliotecas nacionales como BnF y DNB o grandes entidades corporativas como la BBC, INSEE.

Tal y como muestran la segunda y tercera fase del diagrama 5.1, el fichero descargado en

[1] https://lov.linkeddata.es/dataset/lov/

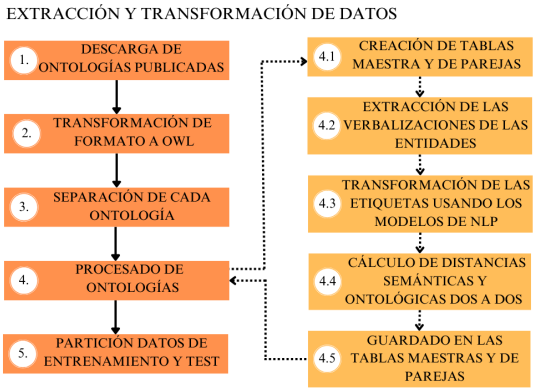

Figura 5.1: ***Diagrama de proceso de obtención de datos para entrenar***

N-Quads[2] se ha transformado a OWL y después se ha dividido el conjunto de datos de LOV en sus respectivos vocabularios usando OwlReady2[3] (Jean-Baptiste, 2021) en Python.

Una vez separados y organizados los datos, se procesa cada uno de los vocabularios recogiendo información clave para el estudio (Fase 4 del Diagrama 5.1). Para ello, se ha organizado una base de datos en SQLite en la que un vocabulario constará de dos tablas (Fase 4.1 del Diagrama 5.1), cuyo esquema UML se muestra en la Figura 5.2. A continuación se van a describir las dos tablas que tendrá cada vocabulario:

- **La tabla principal o maestra** tendrá el mismo nombre que el vocabulario, sustituyendo caracteres no alfanuméricos por "_". Esta tabla albergará una columna con su identificador numérico, dos columnas tipo *varchar* con la etiqueta del concepto y su URI y varias columnas tipo BLOB con las transformaciones de las etiquetas en distintos espacios. Los espacios vectoriales elegidos han sido los de Spacy, Glove y Glove-Poincaré.

- **La tabla distancias** tendrá el mismo nombre que el vocabulario, sustituyendo caracteres no alfanuméricos por "_" y añadiendo al final del nombre "_dist". Esta tabla albergará las distintas distancias calculadas entre los pares de conceptos de esa ontología.

  Las dos primeras columnas son de tipo entero y son los identificadores de los dos conceptos a calcular la distancia entre ellos. La tercera columna es también de tipo entero, y consiste en el identificador del LCA de los dos conceptos. El resto de columnas son de tipo *Float64* y contienen las distancias calculadas de los vectores de las etiquetas transformadas. Las distancias elegidas calculadas son, con el modelo de Spacy y de Glove: la distancia euclídea modificada, coseno, Minkowski modificada con $n = 3$, y con el modelo de Glove-Poincaré: las distancias Girocoseno, coseno y Poincaré.

  Las distancias euclídea y Minkowski calculadas en este trabajo tienen una pequeña mo-

[2]Formato de texto plano para codificar un conjunto de datos RDF.
[3]https://owlready2.readthedocs.io/en/v0.42/

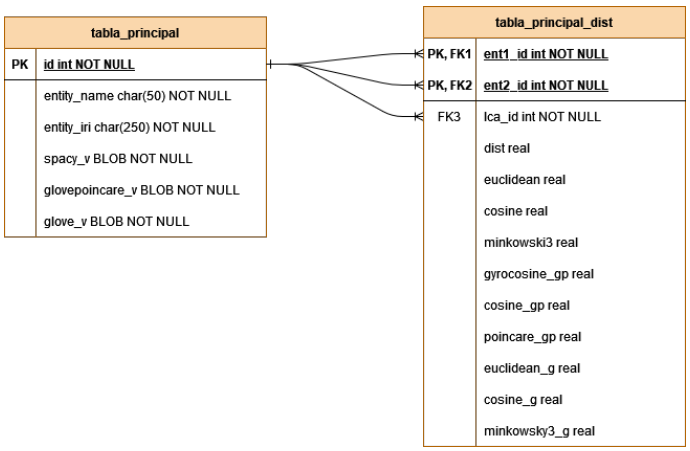

Figura 5.2: ***Esquema UML de la base de datos.***

dificación con respecto a las originales. Esta transformación se aplica debido a que, al tener los vectores de reperesentación una dimensionalidad tan alta, las distancias son muy grandes. Se definen, siendo d_m la distancia modificada y d la distancia euclídea o Minkowski original:

$$d_m = \frac{1}{1+d}$$

En este punto del proceso, es necesario extraer las verbalizaciones de los conceptos (Fase 4.2 del Diagrama 5.1). Se ha utilizado el último fragmento de su URIs[4] y se ha limpiado de caracteres[5]. En los casos en los que la URI no contenía la etiqueta del concepto, se ha utilizado el atributo "label" definido en la ontología para cada concepto[6]. Como algunos vocabularios estaban en distintas lenguas, también se ha procedido a hacer una traducción haciendo uso del módulo "deep_translator" de Google Translator.
Para obtener la transformación de las etiquetas en los distintos espacios vectoriales, se ha utilizado el módulo de Spacy con el modelo entrenado llamado "en_core_web_lg". Para generar los vectores de Glove-Poincaré se ha utilizado el modelo entrenado "poincare_glove_100D_cosh-dist-sq_init_trick.txt". Y finalmente, para generar los vectores con el módulo de Glove se ha cargado el modelo "vanilla_glove_100D_init_trick.txt".

En el caso de que la etiqueta conste de más de una palabra, se hace la transformación de cada una de ellas a sus vectores en sus respectivos espacios vectoriales y se calcula el centroide. Así, en el caso de que un concepto conste de más de una palabra su transformación será:

$$entityWordEmbedding(ent) = \frac{1}{n}\sum_{i=0}^{n} w_i$$

Donde w_i es el vector que representa cada palabra i que compone la etiqueta del concepto ent.

A continuación se debe calcular las distancias semánticas y ontológicas dos a dos (Fase 4.4 del Diagrama 5.1). Para la primera se han utilizado las representaciones vectoriales de las etiquetas de los conceptos y se han utilizado diferentes distancias conocidas en la literatura nombradas en la especificación de la creación de la tabla de distancias. Para la distancia ontológica, se ha utilizado la información estructural de cada ontología utilizando la relación "rdfs:subClassOf".

Una vez generada y guardada en la base de datos toda la información que se necesita para realizar los distintos experimentos de predicción (Fase 4.5 del diagrama 5.1) es necesario

[4]Entendiendo como último fragmento al fragmento final de la URI haciendo separaciones por "/" o "#".

[5]Todos aquellos que no pertenezcan al alfabeto latino se eliminan. El carácter "_" se sustituye por un espacio en blanco, ya que es una práctica común para separar palabras. En el caso de que se existiera una letra mayúscula seguida de minúsculas en una posición intermedia en el literal (camelCase), se añade un espacio delante de la mayúscula por la misma razón.

[6]Se analizó que en el conjunto de datos elegido, la mayor parte de los datos no contenían el atributo "label". En una situación normal se hubiera comenzado tomando el texto del atributo y en el caso de que no existiera se hubiera utilizado el último fragmento de la URI.

filtrarlos y organizarlos a la vez que se hace la separación de los datos de entrenamiento y test (Fase 5 del diagrama 5.1), es decir, porque en estos experimentos se ha hecho la separación a nivel ontología. Esto es, todas las parejas de conceptos de una misma ontología formarán parte sólo de los datos test o sólo de los datos de entrenamiento, nunca se van a separar. Como las diferentes ontologías poseen un número diferente de entidades se han hecho distintas particiones de manera que se ha procurado que el número de los datos de entrenamiento y test fueran un 80 %, 20 % respectivamente. Se deja como trabajo futuro la exploración con otras posibles particiones.

Los distintos conjuntos de datos que se van a utilizar en cada problema se generarán a través de consultas a la base de datos, filtrando siempre de manera que alguna de las dos etiquetas no sea "Thing", la distancia jerárquica entre ambas sea 0 o en el registro falte algún valor de distancias debido a que el modelo de lenguaje no contenía alguna de las dos etiquetas.

Finalmente se han extraído y calculado las distancias de un subconjunto de las ontologías creadas por *Ontology Engineering Group* (OEG)[7] que no tiene nada que ver con el conjunto de ontologías de LOV utilizado para entrenar y hacer el test inicial. Con este conjunto de datos se probará el modelo finalmente elegido y se evaluará.

5.2. Procesos de predicción utilizando las distancias semánticas calculadas.

En esta sección se va a describir los procesos y modelos de predicción que se han utilizado para resolver los distintos problemas escritos en la Sección 4.

5.2.1. Problema de regresión.

En el problema de regresión, **el objetivo es tratar de predecir las distancias jerárquicas mediante las distancias semánticas obtenidas a través de la representación de las etiquetas en distintos espacios vectoriales**. Es decir, utilizando sólo las distancias semánticas calculadas, obtener la distancia jerárquica d^H entre dos conceptos.

Se comenzará haciendo la selección de variables predictivas realizando un análisis exploratorio de los datos, como el cálculo de la media, mediana, desviación estándar y los valores mínimos y máximos. En los modelos basados en árboles y en el modelo *K Nearest Neighbor*s no se hará selección de características y se entrenará con todas las variables. Para la selección de variables en algunos problemas se utilizará *boruta*, del paquete de *BorutaPy*[8]. Se empleará una o una combinación de distancias como variables predictivas del modelo y como variable a predecir se tomará la distancia jerárquica.

[7] https://oeg.fi.upm.es/index.html
[8] https://github.com/scikit-learn-contrib/boruta_py

En este experimento en concreto no se han balanceado los datos pero si que se ha probado a entrenar los modelos utilizando pesos a partir de $d^H = 4$. El proceso que se ha seguido para cada uno de los experimentos está representado en la Figura 5.3.

Primero se hace una carga de los datos en el sistema, en este problema se hace uso de los datos de subsunción. Se normalizan utilizando *StandardScaler*[9] del paquete *preprocessing* de *sklearn*, se hacen diferentes selecciones de variables y tipo de regresión para luego hacer un ajuste del modelo y una posterior evaluación inicial del modelo. En esta fase se utilizarán técnicas de validación cruzada para encontrar los modelos que mejor estadísticas tengan, para ello se empleará *cross_validate* del módulo de *model_selection* de *Scikit-Learn*[10]. Esto corresponde a las fases tres, cuatro y cinco del diagrama de flujo de la Figura 5.3. Los modelos con los que se probará serán los siguientes:

- **Regresión Lineal y Lineal Polinómica.** Se utilizará *LinearRegression* del paquete *linear_model* de *Scikit-Learn*[11]. Para las regresiones lineales polinómicas se utilizarán *PolynomialFeatures* del paquete *preprocessing* de *Scikit-Learn*. Se probarán polinomios hasta de grado cuatro. También se harán regresiones lineales multivariantes de grado uno con estas mismas herramientas.

- **Regresión de Árbol de Decisión (DTR).** Utilizando *DecisionTreeRegressor* del paquete *tree* de *Scikit-Learn*[12]. Se harán experimentos cambiando los hiperparámetros del modelo. En especial “criterion”, que es la función para medir la calidad de una partición y “max_features”, que es el número de características consideradas para realizar la mejor partición en cada rama.

- **Regresión Random Forest (RFR).** Utilizando *RandomForestRegressor* del paquete *ensemble* de *Scikit-Learn*[13]. Se harán experimentos cambiando los hiperparámetros como “n_estimators”, “bootstrap” y “max_samples” que son el número de árboles en el bosque, si las muestras boostrap se utilizan para crear árboles y si lo son, el número de muestras a extraer para entrenar cada estimador respectivamente.

- **Gradient Boosting Regressor (GBR).** Utilizando *GradientBoostingRegressor* del paquete *ensemble*[14]. En algunos experimentos se cambiarán algunos hiperparámetros como “n_estimators” y “learning_rate” que son el número estimadores y la tasa de aprendizaje respectivamente. También se han modificado parametros como “max_depth” que es la profundidad máxima del árbol. Como *Gradient Boosting* es bastante robusto al sobreajuste, normalmente un mayor numero de estimadores resulta en mejores resultados, así que se aumentará considerablemente este parámetro.

[9] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
[10] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_validate.html
[11] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
[12] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
[13] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[14] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

Una vez entrenados, se procede a evaluar de manera inicial los modelos observando su coeficiente de determinación R^2 haciendo validación cruzada. Posteriormente, se hará la predicción de los datos test con la salida discretizada[15] sólo con los modelos seleccionados, que serán aquellos que expliquen la salida. Finalmente, se observarán la distribución de los residuos y el ajuste del modelo a los datos test y se escogerá un único modelo determinado. Se procederá a hacer las predicciones de las distancias de los conceptos de las ontologías de OEG como grupo de control y se podrá determinar si se ha conseguido un modelo que prediga la distancia jerárquica d^H o no.

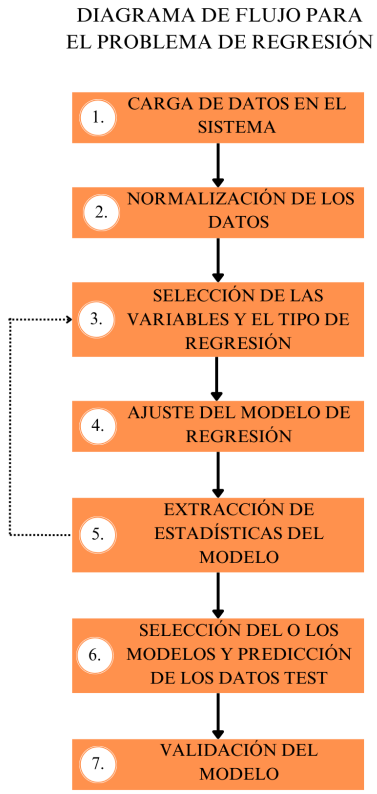

Figura 5.3: ***Proceso para obtener modelos de regresión***

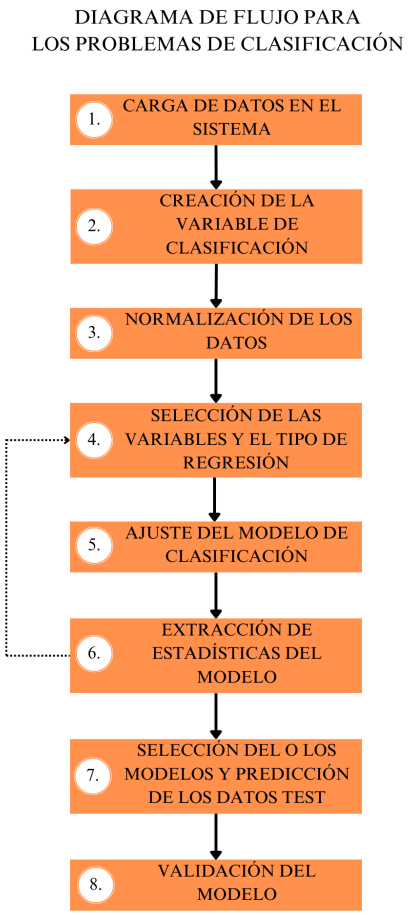

Figura 5.4: ***Proceso para obtener modelos de clasificación multi-clase***

5.2.2. Problemas de clasificación.

En esta subsección se describirá primero la metodología común utilizada para seleccionar un modelo de clasificación sea cual fuere su objetivo. Al final del apartado se especificarán las particularidades de cada problema de clasificación nombrado en el Capítulo 4.

[15]Debido a las características únicas de los datos, se discretiza la salida de la predicción redondeandola para obtener como resultado enteros, pues la distancia jerárquica siempre se dará como entero.

El proceso, tal y como se muestra en la Figura 5.4, comenzará cargando los datos en el sistema. A continuación, se debe calcular la variable discreta a predecir tal y como se ha descrito anteriormente, utilizando la distancia ontológica o jerárquica para ese fin. Como en los modelos de regresión los datos se normalizan utilizando *StandardScaler* del paquete *preprocessing* de *Scikit-Learn*.

En el caso de los sistemas de clasificación se optará por hacer pruebas con o sin balancear los datos. Se utilizarán técnicas de submuestreo (*undersampling*) aleatorio o sobremuestreo (*oversampling*) aleatorio y se complementará con la aplicación de diferentes pesos a las clases para dar una mayor penalización de error a la clase menos abundante.

Una vez adecuados los datos, se comenzará haciendo la selección de variables predictivas. Para ello se comenzará haciendo una análisis exploratorio de los datos, como el cálculo de la media, mediana, desviación estándar y los valores mínimos y máximos. Se utilizarán gráficos de caja y diagramas de correlación para identificar patrones y se calculará la matriz de correlación entre variables para evitar utilizar variables muy correladas.

Una vez elegidas las variables, se procederá a ajustar los diferentes modelos de clasificación utilizando distintas elecciones en los valores de sus hiperparámetros. En esta fase se utilizarán técnicas de validación cruzada para encontrar los modelos que mejor estadísticas tengan, para ello se empleará tambíen *cross_validate* del módulo de *model_selection* de *Scikit-Learn*. Los modelos sobre los que se han probado son los siguientes:

- **Árboles de decisión (DTC).** Utilizando *RandomTreeClassifier*[16] del paquete *tree* de *Scikit-Learn* y variando su hiperparámetro “criterion”, que es la función para medir la calidad de una partición; “splitter”, la estrategia escogida para dividir cada nodo y “max_depth” que es la profundidad máxima del árbol.
- **Random Forests (RFC).** Utilizando *RandomForestClassifier*[17] del paquete *ensemble* de *Scikit-Learn* y variando su hiperparámetro “criterion”, que es la función para medir la calidad de una partición; “n_estimators” es el número de árboles y “max_depth” que es la profundidad máxima en cada árbol.
- **Gradient Boosting Classifier (GBC).** Importando *GradientBoostingRegressor*[18] del paquete *ensemble* de *Scikit-Learn* y variando sus hiperparámetros “max_depth”, parámetro que indica la profundidad máxima de los árboles; “min_samples_split” que es el parámetro que indica el mínimo numero de muestras necesarias para relaizar un split y finalmente “n_estimators”, parámetro que indica las etapas a realizar.
- **K-Nearest Neighbors (KNN).** Utilizando *KNeighborsClassifier*[19] del paquete *neighbors*

[16] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[17] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[18] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html
[19] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

de *Scikit-Learn* y variando sus hiperparámetros “weights”, que es el parámetro que indica la función peso empleada en la predicción, “algorithm”, que modifica el algoritmo empleado y el número de vecinos empleados, “n_neighbors” .

5.2.2.1. Particularidades del proceso de clasificación para obtener la distancia cualitativa entre dos conceptos.

En este apartado se describirán las particularidades de la metodología del proceso de clasificación para seleccionar un modelo adecuado con el fin de medir distancias jerárquicas cualitativamente. **La distancia jerárquica cualitativa se refiere a la categorización de las distancias entre conceptos en diferentes niveles de cercanía o lejanía, como “cercano”, “lejano” o “muy lejano”, basándose en las distancias jerárquicas calculadas entre ellas**. Por lo tanto, se debe definir a qué clases pertenece cada distancia.

- **Clase 0:** Par de conceptos muy cercanos. Son aquellos que están a distancia jerárquica 1 o 2.
- **Clase 1:** Par de conceptos lejanos. Son aquellos que están a una distancia jerárquica entre 3 y 4.
- **Clase 2:** Par de conceptos muy lejanos. Son aquellos que están a una distancia jerárquica mayor que 4.

Como el objetivo final es detectar huecos en la jerarquía de una ontología, se ha tomado como datos aquellos registros que contienen aquellas parejas de conceptos (y sus diferentes distancias) que pertenecen a la misma jerarquía. Con estos datos y con la definición de las clases anterior, se define la variable a predecir. El resto de proceso para obtener un modelo de clasificación es análogo al visto en la Subsección 5.2.2.

5.2.2.2. Particularidades del proceso de clasificación para determinar si dos conceptos tienen relación de subsunción directa.

En este apartado se describirán las particularidades de la metodología del proceso de clasificación para seleccionar un modelo adecuado con el fin de determinar si dos conceptos tienen relación de subsunción directa. Es decir, **se debe determinar si alguno de los dos conceptos es subclase del otro a distancia uno. Se definen dos clases**:

- **Clase 0:** Parejas de conceptos que tienen relación de subsunción directa. Es decir, la distancia jerárquica es 1.
- **Clase 1:** Parejas de conceptos que no tienen relación de subsunción directa. Es decir, la distancia jerárquica es mayor a 1.

Como el objetivo final es detectar huecos en la jerarquía de una ontología, se ha tomado como datos aquellas parejas que contienen pares de conceptos (y sus diferentes distancias)

que pertenecen a la misma jerarquía, es decir, utiliza el mismo conjunto de datos que la Subsección 5.2.2.1. Con estos datos y con la definición de las clases anterior, se define la variable a predecir y el resto de proceso para obtener un modelo de clasificación es análogo al visto en la Subsección 5.2.2.

5.2.2.3. Particularidades del proceso de clasificación para determinar si dos conceptos están en la misma rama jerárquica.

En este apartado se describirán las particularidades de la metodología del proceso de clasificación para seleccionar un modelo adecuado con el fin de determinar si dos conceptos están en la misma rama jerárquica. Es decir, se debe determinar si el ancestro común de la pareja de conceptos es alguno de ellos o no. Se determinan dos clases obtenidas de conjuntos de datos distintos:

- **Clase 0:** Parejas de conceptos que no pertenecen a la misma rama jerárquica. Son aquellas pares de parejas de conceptos cuyo LCA no es ninguno de los dos.
- **Clase 1:** Parejas de conceptos que sí pertenecen a la misma rama jerárquica. Son aquellas pares de parejas de conceptos cuyo LCA es alguno de los dos.

Siguiendo la definición de las clases, se asignará una etiqueta a cada conjunto y se unirán y mezclarán los datos para obtener los conjuntos de datos con los que se entrenará y evaluará el modelo. El resto de proceso para obtener un modelo de clasificación es análogo al visto en la Subsección 5.2.2.

Resultados y discusión

6

En este capítulo se van a describir los resultados para cada uno de los objetivos de predicción o clasificación.

En primer lugar se hace una primera exploración de los datos y así es posible hacerse una idea de la distribución y complejidad de los mismos. Se observa en el diagrama de caja de la Figura 6.1 que casi la totalidad de las distancias tienen valores entre uno y tres, dejando a aquellos datos por encima de ese valor como datos atípicos. Esto no es nada extraño, puesto que los elementos con distancias jerárquicas más altas deben estar a mucha más profundidad de la ontología, es decir, ser una "hoja" y por tanto ser menos comunes.

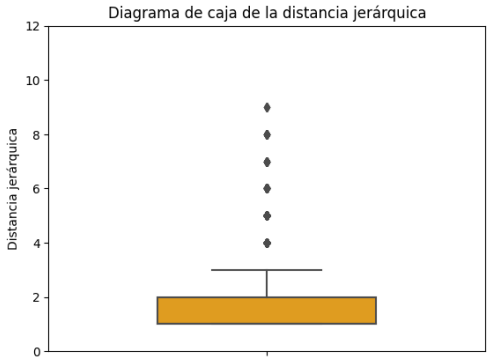

Figura 6.1: ***Diagrama de caja de la distancia jerárquica.***

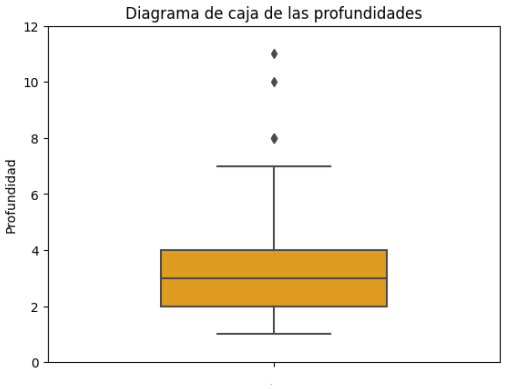

Figura 6.2: ***Diagrama de caja de la profundidad.***

De hecho, en el diagrama de caja de la Figura 6.2 donde se representan las profundidades de las ontologías de entrenamiento, se observa que en el 50 % de ellas tienen una profundidad máxima de 3. Y que el 75 % de las ontologías tienen una profundidad máxima de 4, aunque los bigotes se extienden hasta cubrir un rango de profundidad máxima entre 1 y 7.

A continuación, en la Figura 6.3, se muestran las distribuciones normalizadas de cada una de las variables que se van a utilizar para predecir la distancia jerárquica. Se observa que en cada uno de los diagramas de caja el valor mínimo es -2 y la media y los rangos intercuartílicos están entre -1 y 1 como cabría esperar. Sin embargo, se observa una cantidad inusual de valores atípicos principalmente en la característica de la distancia euclídea modificada y en la

característica de la distancia Minkowski modificada.

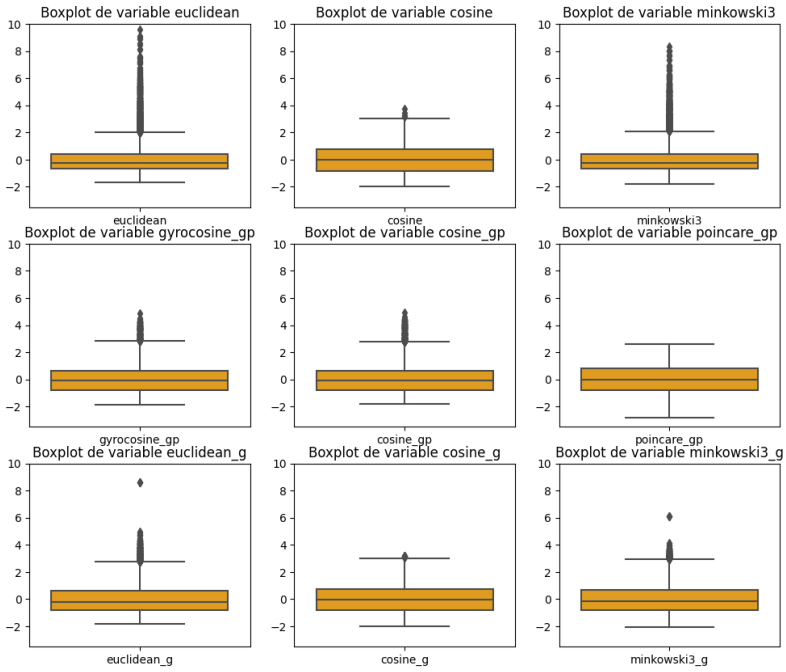

Figura 6.3: ***Diagramas de caja de las variables predictivas***

En las siguientes secciones se dispondrán los resultados de cada uno de los problemas en una tabla. En ellas, la primera columna denotará el modelo en el que se está entrenando[1]. La segunda columna contendrá las especificaciones o parámetros utilizados para entrenar el modelo. El resto de columnas dependerá del tipo de problema que se vaya a abarcar y se definirán en cada sección.

6.1. Resultados del problema de regresión

Para la evaluación de los problemas de regresión se utilizarán las métricas habituales y la notación descrita en la Sección 6. En la tercera columna de la tabla de resultados se tendrá la raíz del error cuadrático medio (RMSE), en la cuarta el R^2 de los datos de entrenamiento y en la última columna el p-valor. En la Tabla 6.1 sólo se han representado las métricas de aquellos modelos con mejores resultados con la salida discretizada.

Utilizando Boruta se observa que las características que aportan más información a los modelos son coseno, gyrocosine_gp y poincare_gp. Tiene sentido, ya que tal y como se ha visto en la literatura, el modelo Glove-Poincaré captura mucho mejor la hiperonimia o hiponimia que el resto de modelos.

[1]Como los modelos *Decision Tree Regresor* (DTR), *Decision Tree Classifier* (DTC), *Random Forest Classifier* (RFC), *Random Forest Regressor* (RFR), *Gradient Boosting Classifier* (GBC) o *K-Nearest Neighbors* (KNN).

Modelo	Especificaciones	RMSE	R2	p-value
DTR	max_features=log2	1.28	0.99	$8 \cdot 10^{-38}$
RFR	max_features=None, n_jobs=-1	0.98	0.91	$1 \cdot 10^{-111}$
RFR	Weighted	0.98	0.90	$1 \cdot 10^{-121}$
GBR	n_estimators=500, max_depth=50	1.28	0.99	$1 \cdot 10^{-111}$
KNN	weights=distance, n_jobs=-1, n_neighbors=15	1.02	0.99	$1 \cdot 10^{-31}$

Tabla 6.1: ***Tabla con resultados para el problema de regresión de la distancia jerárquica***

Después de hacer validación cruzada, los modelos de Regresión Lineal, Regresión Lineal Multivariante y Regresión Polinómica con las características más relevantes, no obtienen ningún resultado destacable, ya que su R^2 en los datos de entrenamiento está muy cercano a cero. Sin embargo, los modelos más prometedores son los modelos *K-Nearest Neighbors* y *Random Forest* entrenando con todas las características y además sin limite de características. Probándolos en los datos test (redondeando a enteros la salida para discretizarla) se pueden dibujar distintas gráficas de los residuos que pueden ser útiles para el análisis de los modelos. En las Figuras 6.4 y 6.5 se puede observar que los errores son negativos en las predicciones más altas y positivos en las predicciones más bajas, los errores se distribuyen aleatoriamente a lo largo de todo el conjunto de datos y que la distribución de los residuos es normal.

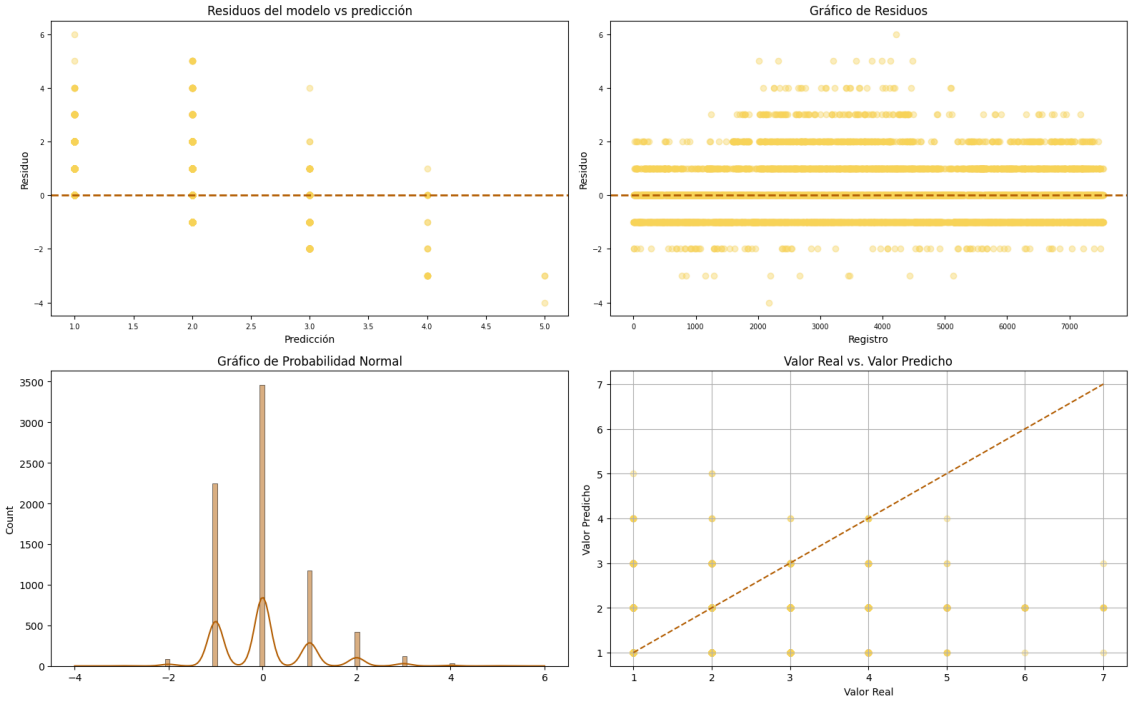

Figura 6.4: ***Métricas de los residuos del modelo RFR con max_features=None, n_jobs=-1***

Se puede concluir que ambos modelos capturan bastante bien las distancias jerárquicas, pero es posible que las características que se utilizan para intentar predecirla no sean lo su-

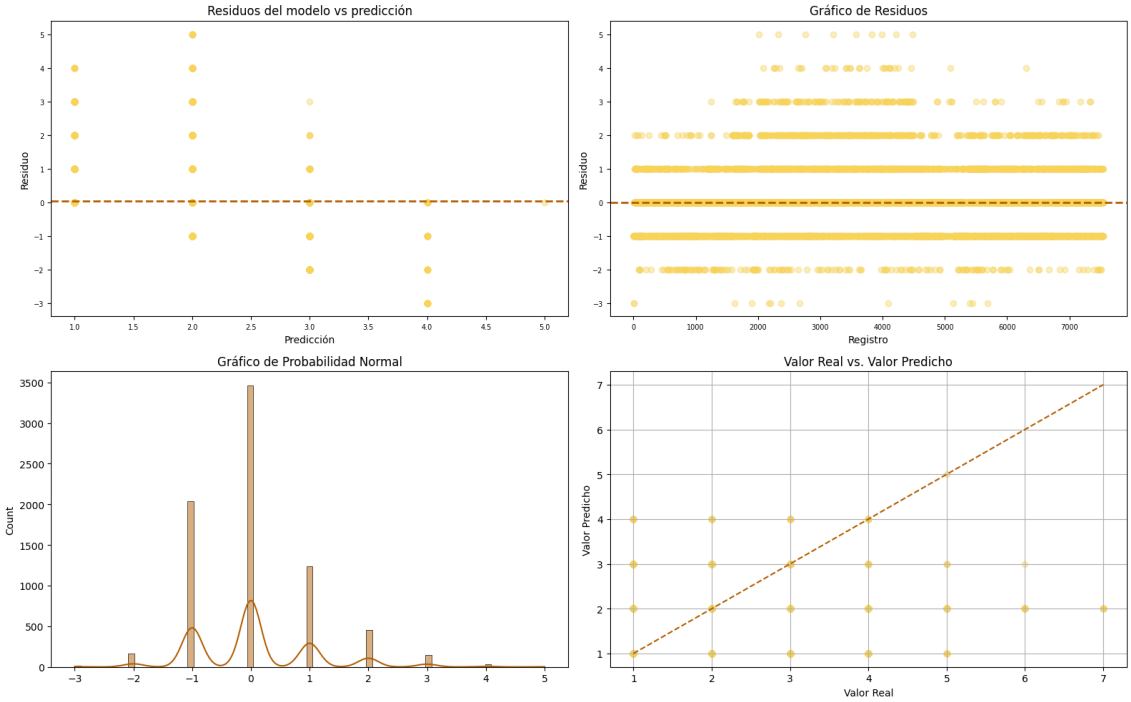

Figura 6.5: ***Métricas de los residuos del modelo KNN con n_jobs=-1, weights=distance***

ficientemente descriptivas como para poder obtener un modelo fiable. Esto puede ser debido también a que los modelos no sean lo suficientemente flexibles como para capturar las complejas interacciones entre las características para realizar una predicción certera.

En este caso concreto, se va a evaluar el modelo KNN, ya que tiene unas estadísticas ligeramente mejores que el modelo RFR, con unos datos test que no pertenecen al conjunto de LOV. Se obtiene una métrica de RMSE, calculándolo con las salidas discretizadas, de 1.06.

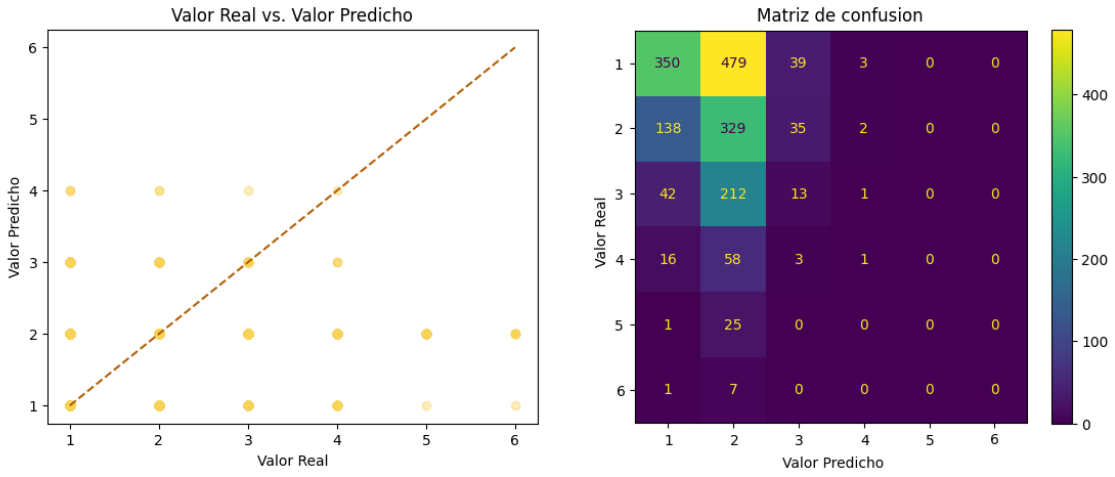

Figura 6.6: ***Diagrama del valor predicho/valor real y matriz de confusión del modelo KNN***

Al haber discretizado la predicción de la regresión se puede ver, como análisis complementario, el tipo de error utilizando una matriz de confusión. En la Figura 6.6 se han dibujado dos gráficas, una de las cuales es una matriz de confusión en la que se observa que los valores predichos están entorno a una unidad de diferencia de los valores reales, siendo menos preciso a la hora de predecir distancias por encima de 4. Esto es explicable debido a la distribución de distancias que tenemos en nuestro conjunto de datos.

6.2. Resultados de los problemas de clasificación

En este apartado se verán los resultados de los distintos problemas de clasificación. Para ello se dibujarán tablas que utilizarán la notación descrita en la Sección 6 y además se añadirán métricas típicamente utilizadas en los problemas de clasificación. La tercera columna será la exactitud (acc), la cuarta la precisión (prec), la quinta columna será la exhaustividad (rec) y la última será la medida F1 (F1).

6.2.1. Problema de distancia cualitativa

En este problema se divide el conjunto de distancias en tres grupos. La “clase 0”, será la clase que componen los datos relativos a distancia cercana (distancias 1 y 2), la “clase 1” será la clase que componen los datos relativos a la distancia lejana (distancias 3 y 4) y la “clase 2” será la clase que componen los datos relativos a la distancia muy lejana (distancias mayores a 4).

Se hace un diagrama de cajas por clases y por variable para ver qué variables podrían ser clave para la clasificación.

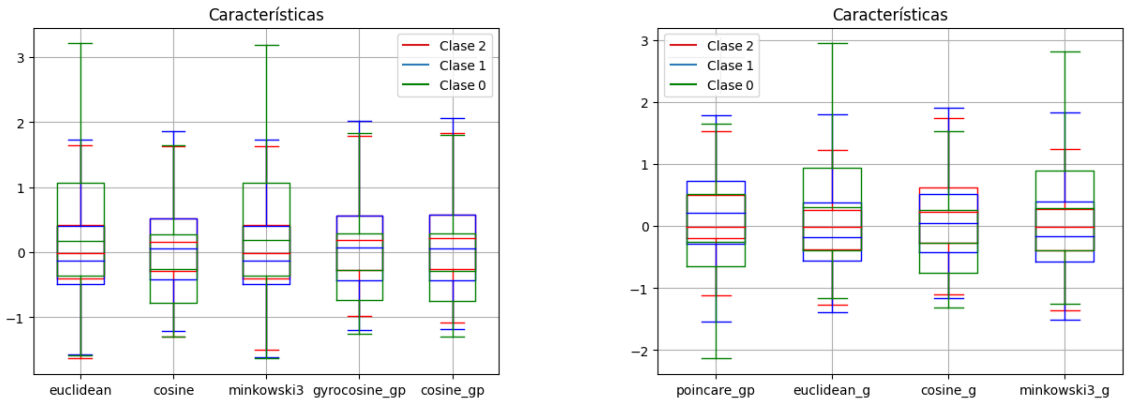

Figura 6.7: ***Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de la distancia cualitativa***

Se observa en la Figura 6.7 que hay una pequeña diferencia entre las clases en los rangos intercuartílicos y medianas. Esa diferencia es más notoria en la variable euclidea, minkowski3, euclidea_g y minkowski_g, a la hora de diferenciar entre clase 3 y el resto. También se observa

que la mediana en las variables gyrocosine_gp, cosine_gp y poincare_gp estan desplazadas entre una clase y otra, siendo el elemento diferenciador de entre la clase 1 y la clase 2. Al hacer la exploración de datos, también se ha visto una gran descompensación entre las tres clases. Para balancear los datos se utilizará la técnica de *oversampling* aleatorio.

Los modelos más prometedores en las pruebas de validación cruzada se han utilizado para construir la Tabla 6.2. Al ser un problema con tres clases desbalanceadas se ha utilizado la media macro o *macro average*, una media aritmética de los resultados de las clases, de manera que las tres clases aportan por igual.

Modelo	**Especificaciones**	**acc**	**prec**	**rec**	**f1**
DTC	max_features = 4	0.53	0.34	0.31	0.31
RFC	class_weight=balanced, max_depth=10, max_features=0.5	0.62	0.31	0.32	0.31
GBC	max_depth=100, n_estimators=500	0.57	0.31	0.31	0.29
KNN	n_jobs=-1, weights=distance	0.83	0.31	0.33	0.31

Tabla 6.2: ***Tabla con resultados para el problema de clasificación cualitativa***

Los modelos más interesantes son *K-Nearest Neighbors* y *Random Forest Classifier*, ya que aproximadamente tienen los mismos resultados en precisión, exhaustividad y F1.

Estos modelos se han puesto a prueba con los datos test, utilizando su matriz de confusión 6.8 y 6.9, es posible tomar una decisión de qué modelo conviene mas para dar solución al problema.

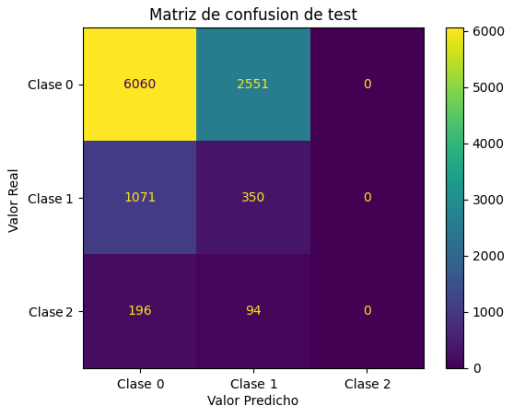

Figura 6.8: ***Matriz de confusión del modelo RFC con class_weight=balanced, max_depth=10, max_features=0.5***

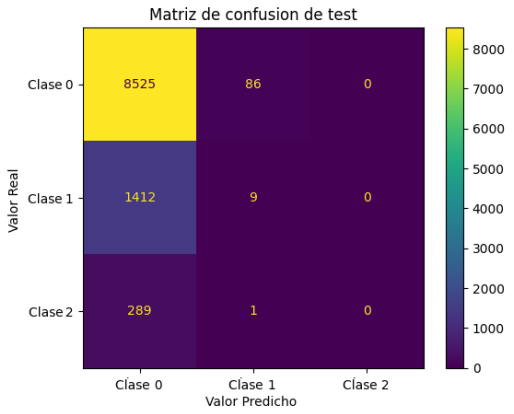

Figura 6.9: ***Matriz de confusión del modelo KNN con n_jobs=-1, weights=distance***

En ambos modelos se predice muy bien aquellos elementos que están muy cerca (Clase 0). Sin embargo, ambos tienen una tendencia a predecir Clase 0 erróneamente, en mayor medida el modelo KNN, y en ninguno de los dos casos se predice correctamente la categoría muy lejos

Clase	prec	rec	f1	soporte
0	0.84	0.62	0.71	1375
1	0.28	0.53	0.36	346
2	0.06	0.15	0.09	34
macro-avg	0.39	0.43	0.39	1755
weight-avg	0.71	0.59	0.63	1755

Tabla 6.3: ***Resultados del modelo elegido (RFC) con datos test finales***

(Clase 2). Esto puede ser debido a que se tienen muy pocos elementos de la última clase en test. Se va a escoger para la evaluación final el modelo RFC ya que comete menos error en la la Clase 1 y se observa menos tendencia a etiquetar erróneamente las muestras en Clase 0.

En la evaluación final con los datos extraídos de OEG de la Tabla 6.3 se observa que el modelo sí es capaz de predecir elementos de la clase dos. Las métricas con los datos test finales mejoran un poco pero aun así se puede decir que hay poca señal para realizar una clasificación de los elementos en las tres clases conveniente.

6.2.2. Clasificación entre parejas de conceptos con subsunción directa o sin ella.

Se comienza con una exploración de los datos, haciendo un diagrama de cajas por clases y por variable para ver qué variables podrían ser clave para la clasificación.

Se puede apreciar en la Figura 6.10 que hay pequeñas diferencias entre las dos clases en los rangos intercuartílicos y medianas. Esa diferencia es más notoria en la variable euclídea, minkowski3, euclidea_g y minkowski_g, aunque también se observa que la mediana en las variables gyrocosine_gp y cosine_gp y poincare_gp esta muy desplazada entre una clase y otra, casi haciendo coincidir la mediana de la clase 2 con el cuartil inferior de la primera clase.

Hay que tener en cuenta que clasificar dos clases que tienen relación directa erróneamente es un error mucho mas grave que a la inversa, puesto que el objetivo final es detectar huecos en la jerarquía. Se han ajustado hiperparámetros con validación cruzada y se ha construido la tabla 6.4 con los modelos más prometedores, en la que se puede observar que los modelos más equilibrado e interesantes son el Decision Tree con una precisión un poco mayor que el Random Forest, que sin embargo tiene una exhaustividad algo mayor.

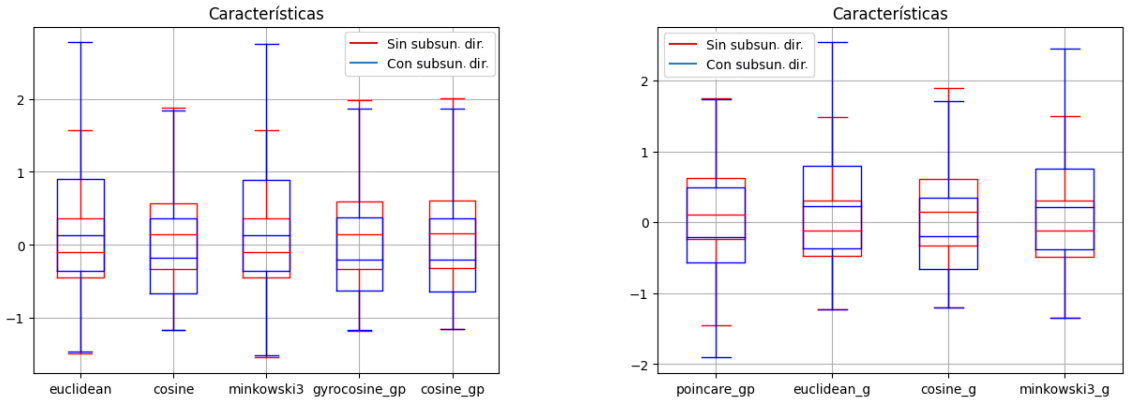

Figura 6.10: ***Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de subsunción directa.***

Modelo	Especificaciones	acc	prec	rec	f1
DTC	max_depth=10, max_features=0.5, min_samples_split=5	0.58	0.55	0.86	0.67
RFC	ccp_alpha=0.01, class_weight=balanced, max_depth=6, max_leaf_nodes=22, max_samples=0.4, min_impurity_decrease=0.01, min_weight_fraction_leaf=0.01	0.57	0.55	0.87	0.67
GBC	min_samples_split=10	0.47	0.47	0.61	0.53
KNN	n_jobs=-1, weights=distance, n_neighbors=15	0.48	0.47	0.62	0.54

Tabla 6.4: ***Tabla con resultados para el problema de clasificación des subsunción directa***

Dibujando la matriz de confusión de estos dos modelos, tal y como se muestra en las figuras 6.11 y 6.12 es posible tomar una decisión de qué modelo conviene mas para dar solución al problema. En estas matrices de confusión “Clase 0” se denota a aquellos elementos que sí tienen subsunción directa, y “Clase 1” a los que no tienen subsunción directa.

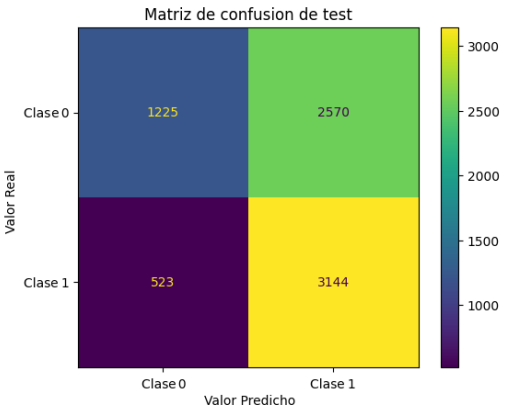

Figura 6.11: ***Matriz de confusión del modelo DTC***

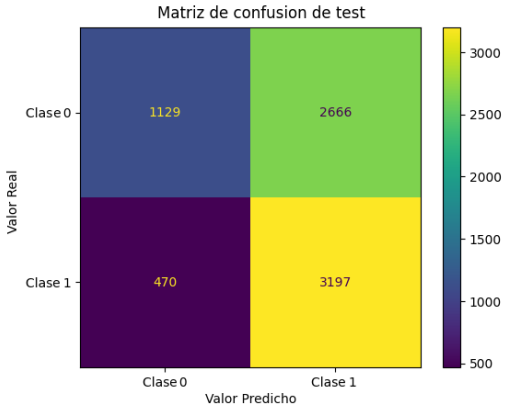

Figura 6.12: ***Matriz de confusión del modelo RFC***

acc	prec	rec	f1
0.63	0.59	0.88	0.71

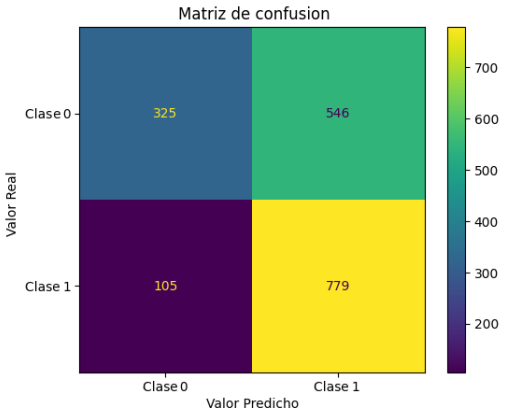

Tabla 6.5: ***Resultados del modelo elegido (RFC) con datos test finales***

En este caso los dos son muy parecidos, pero finalmente se escoge el modelo RFC debido a que tiene una exactitud mayor, no hace sobreajuste y al ser un algoritmo más complejo es probable que el modelo capture más información que el modelo DTC.

A continuación se va a probar el modelo entrenado sobre los datos finales test, creados por OEG. Se concluye haciendo uso de la Figura 6.5, que hace relativamente bien la clasificación de parejas que no tienen relación de subsunción directa. Aunque el modelo obtenga muchos falsos positivos, puede ser una herramienta que ayude a la detección de saltos muy grandes en la jerarquía de una ontología. Es posible que las distancias obtenidas a través de los vectores de representación no estén dando suficiente información para discriminar una categoría de otra y sugiere el uso de los vectores de representación de manera directa, aunque esto quedará como trabajo futuro.

6.2.3. Clasificación entre parejas de conceptos con subsunción o sin ella.

Como en las secciones anteriores, se comienza con una exploración de los datos haciendo un diagrama de cajas por clases y por variable para ver qué variables podrían ser clave para la clasificación. En la exploración de los datos de entrenamiento se observa que está desbalanceado, el número de datos con subsunción es de dos órdenes de magnitud menor que aquellos que no tienen subsunción. Por tanto, en este problema se hace sobremuestreo (*oversampling*) y se empleará, en algunos casos, la técnica de añadir distintos pesos a las clases.

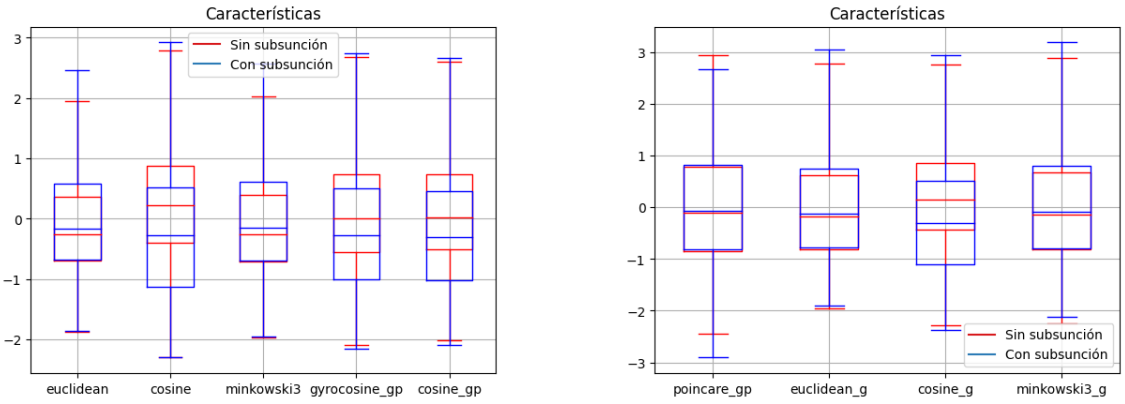

Figura 6.13: ***Diagrama de caja de las variables predictivas, por clases, para el problema de clasificación de subsunción.***

Se puede apreciar en la Figura 6.2.3 que en la mayor parte de las variables predictivas no hay diferencia entre clases. Solo se observa que hay pequeñas diferencias en las distribuciones de las dos clases en las variables coseno, girocoseno_gp, coseno_gp y coseno_g. Previsiblemente estas serán variables que necesitarán estar en el modelo para poder realizar alguna clase de clasificación.

Las clasificaciones con los distintos modelos más prometedores se ha construido la Tabla 6.6. En ella se puede observar que algunos de los resultados tienden a tener la medida de exactitud bastante alta, pero sin embargo el resto de medidas tienden a estar por debajo o alrededor de 0.5. Esto puede ser porque los datos test están muy desbalanceados y que el modelo entrenado tiende a predecir correctamente la clase más abundante.

Se han escogido como candidatos los el modelo de DTC sin ninguna especificación y el RFC de la Tabla 6.6. El primero se ha escogido debido a que tiene mas posibilidades de no haber hecho sobreajuste, y el segundo debido a que tiene unas métricas muy altas y podría ser mas explicativo que el resto de modelos.

Modelo	**Especificaciones**	**acc**	**prec**	**rec**	**f1**
DTC	-	0.68	0.96	0.68	0.79
RFC	max_depth=15, max_features=0.4, n_jobs=-1	0.98	0.96	0.96	0.95
GBC	learning_rate=0.3, max_features=0.5	0.98	0.96	0.98	0.97
KNN	n_jobs=-1, weights=distance, n_neighbors=10, p=1	0.98	0.96	0.98	0.97

Tabla 6.6: ***Tabla con resultados para el problema de clasificación de subsunción***

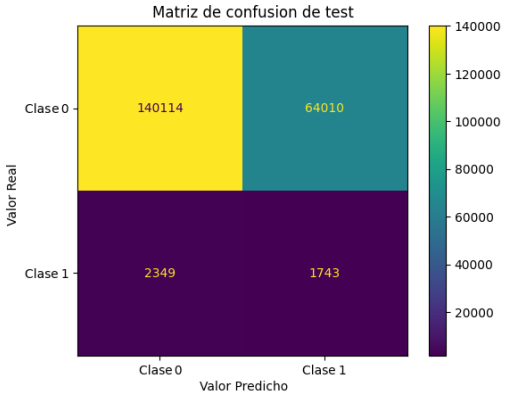

Figura 6.14: ***Matriz de confusión del modelo DTC***

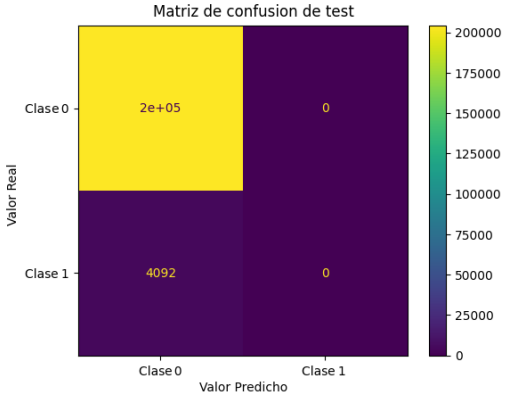

Figura 6.15: ***Matriz de confusión del modelo RFC con max_depth=15, max_features=0.4, n_jobs=-1***

En este caso, el se observa que el modelo RFC sufre de sobreajuste y se utilizará el modelo DTC para realizar las pruebas en las ontologías de OEG. En la Figura 6.7 se muestran los resultados obtenidos en la evaluación. A pesar de que las métricas son buenas, al observar la matriz de confusión se puede concluir que el modelo no es bueno al predecir parejas con subsunción, por lo que no es un modelo válido. Esto puede ser debido a que falta información contextual de la ontología, por ello en futuros desarrollos se deberían incluir métricas de la propia ontología a la que pertenecen o incluso información referente a su LCA y sus respectivas distancias al mismo.

acc	prec	rec	f1
0.83	0.80	0.84	0.82

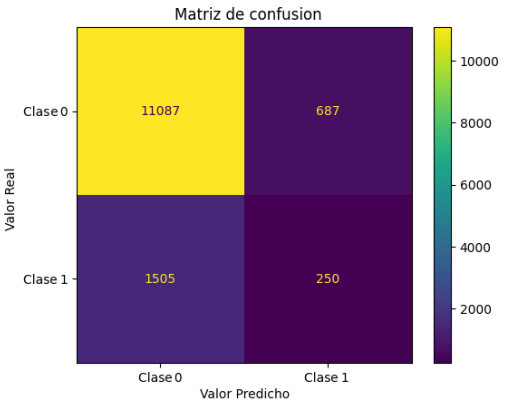

Tabla 6.7: ***Resultados del modelo elegido (DTC) con datos test finales***

Conclusiones

7

En esta sección, se presentan las conclusiones derivadas de los experimentos realizados para abordar los problemas descritos en la Sección 4. A lo largo de este trabajo, se han recolectado los datos y se han explorado y analizado minuciosamente el rendimiento de diversos enfoques y técnicas, tanto en tareas de regresión como de clasificación.

En cuanto a la recolección de datos, se concluye que a pesar que el origen de los datos es un conjunto de datos curado por un equipo y que cumple unos ciertos requisitos de calidad, se ha necesitado mucho trabajo manual para homogeneizar los metadatos de las ontologías procurando la limpieza de caracteres no alfabéticos.

Por otro lado, se ha observado en la exploración de los datos y la influencia de las características que las distancias derivadas del modelo Glove-Poincaré han sido clave para este estudio. Además, los resultados de los experimentos proporcionan información valiosa sobre la efectividad de los modelos y enfoques evaluados, así como posibles áreas de mejora y futuras direcciones de investigación. A continuación, se resumen los hallazgos más destacados y en la Sección 8 se plantean consideraciones para futuros trabajos.

7.1. Problema de regresión.

En la resolución problema de predicción de la distancia jerárquica entre dos conceptos utilizando regresión lineal se ha visto que al linealizar una variable que inicialmente es una variable discreta se obtienen gráficos descriptivos de la regresión un poco diferentes a lo usual. Las conclusiones a las que se han llegado son las siguientes:

1. Se predicen bien las distancias de parejas más comunes, y por lo tanto más bajas.

2. El modelo obtenido es explicativo, pero existe un sesgo a predecir distancias por debajo de dos, ya que son las distancias con mayor frecuencia. Es una consecuencia de no tener distancias a predecir homogéneas.

3. Hay indicios de que no existe demasiada señal en las distancias semánticas calculadas para poder hacer una buena predicción.

7.2. Problema de clasificación de la distancia cualitativa.

En la resolución problema de clasificación de la distancia cualitativa entre dos conceptos se ha visto que no existe mucha diferencia en la distribución de las características para cada clase. Esto esto es un indicio de que la clasificación puede no ser demasiado buena debido a que la señal para diferenciar cada clase puede ser escasa. Las conclusiones a las que se han llegado son las siguientes:

1. El modelo es relativamente bueno etiquetando muestras de clase 0 (muy cerca) que se corresponde a aquellos elementos que están a distancia 1 o 2. Son los elementos más comunes en el conjunto de datos.

2. El modelo no es bueno prediciendo el resto de clases. Como se ha hecho *oversampling* es muy probable que sea debido a que no hay mucha señal que pueda diferenciar una clase de otra.

7.3. Problema de clasificación de parejas con subsunción directa o sin ella.

En la resolución problema de clasificación de parejas de conceptos con subsunción directa o sin ella las conclusiones son las siguientes:

1. Evaluando en los datos test de LOV, el modelo seleccionado es muy bueno etiquetando muestras de la clase positiva. Es posible que haya una ligera tendencia a etiquetar erróneamente muestras de la clase negativa como si fueran de la clase positiva.

2. En la evaluación sobre los datos de las ontologías de OEG se observa esa misma tendencia, aunque no en tanta medida. Dejando un modelo que generaliza y predice bastante bien.

7.4. Problema de clasificación de parejas con o sin subsunción.

En la resolución del problema de clasificación parejas con o sin subsunción se ha visto que la diferencia de las distribuciones de las distancias semánticas calculadas (las características) no se diferencian entre una clase u otra. En los resultados de los modelos construidos se concluye:

1. Hay una tendencia muy a alta a predecir la hipótesis negativa: no hay subsunción.

2. No existe suficiente señal en las características como para poder generalizar un modelo que funcione bien en datos externos.

Limitaciones y Perspectivas de Futuro

8

Los resultados obtenidos en los experimentos de regresión y clasificación presentados en este trabajo han revelado ciertas limitaciones que merecen ser abordadas para mejorar la calidad y la eficacia de los modelos propuestos. Entre las limitaciones identificadas se encuentran:

1. **Representación limitada de los conceptos.** Al utilizar distancias ya calculadas de los vectores de representación, es posible que se esté simplificando demasiado la información semántica que proporcionan los conceptos.

2. **Representación limitada utilizando sólo técnicas de GloVe.** La representación de conceptos en los espacios vectoriales utilizando únicamente GloVe y técnicas relacionadas pueden no aportar información adicional que permitan encontrar buenas soluciones a los problemas propuestos. La selección de otras representaciones y enfoques de vectores podría enriquecer la información semántica, permitiendo capturar relaciones y matices más complejos entre los conceptos.

3. **Dependencia de representaciones genéricas.** Utilizar modelos de lenguaje pre-entrenados puede ser insuficiente para capturar las particularidades de un dominio específico.

4. **Complejidad Semántica no totalmente explorada.** La desambiguación de palabras polisémicas y la comprensión profunda de los significados de las clases en una ontología son aspectos que requieren más atención. La inclusión de elementos semánticos adicionales de la ontología podría ser vital para mejorar la precisión de los modelos y lograr una interpretación más acertada de los datos.

5. **Falta de consideración de la composición estructural de las ontologías a las que pertenecen las muestras.** Aunque se ha abordado el contenido semántico y las relaciones jerárquicas, no se ha hecho un análisis de cómo estas ontologías están construidas en términos de su arquitectura y jerarquía. Aspectos como la anchura o profundidad de una ontología podrían ser cruciales para una clasificación adecuada de la posible relación que tengan dos términos en esa propia ontología.

Para superar las limitaciones mencionadas y mejorar los resultados obtenidos en este trabajo, se proponen las siguientes perspectivas para futuras investigaciones:

1. **Uso de la representación vectorial de los conceptos.** Para capturar toda la información posible de la representación de los conceptos se propone utilizar directamente los vectores generados por los modelos de representación del lenguaje como entrada a los modelos de clasificación y regresión empleando técnicas con redes neuronales y aprendizaje profundo.

2. **Exploración de diversas representaciones de lenguaje.** Se debe investigar el uso de una amplia gama de técnicas de representación del lenguaje, incluyendo modelos basados en el contexto.

3. **Entrenamiento de Modelos de Lenguaje Específicos del Dominio.** La creación y entrenamiento de modelos de representación del lenguaje basados en el corpus de dominio podrían ser cruciales para una representación semántica más precisa y contextualizada. Esto facilitará la captura de relaciones y estructuras complejas dentro de un dominio particular.

4. **Incorporación de información semántica adicional.** La inclusión de información semántica adicional, como contexto de la ontología o definiciones de los conceptos, en el proceso de representación, podría mejorar la precisión y la interpretación de los resultados.

5. **Consideración de métricas estructurales de las ontologías a las que pertenecen las muestras.** Incluir métricas relativas a la ontología podría dar información valiosa al modelo para clasificar o predecir correctamente las distancias entre los conceptos definidos dentro de la propia ontología.

Lista de Acrónimos

BBC Corporación Británica de Radiodifusión o *British Broadcasting Corporation*.

BnF Biblioteca Nacional de Francia Catalogo General o *Bibliothèque nationale de France Catalogue Général*.

DL Lógicas Descriptivas.

DNB Biblioteca Nacional Alemana o *Deutsche Nationalbibliothek*.

IA Inteligencia Artificial.

INSEE Instituto Nacional de Estadística de Estudios Económicos o *L'Institut national de la statistique et des études économiques*.

KG Grafos de conocimiento o *Knowledge graphs*.

LCA Ancestro Común más Bajo o *Lowest Common Ancestor*.

LOV *Linked Open Vocabularies*.

ML Aprendizaje Automático o *Machine Learning*.

OEG *Ontology Engineering Group*.

OWL *Ontology Web Language*.

PLN Procesado del Lenguaje Natural.

RDF *Resource Description Framework*.

RDFS RDF Schema.

URI *Uniform Resource Identifier*.

W3C *World Wide Web Consortium*.

Bibliografía

Bobed Lisbona, C. (2013). *Semantic Keyword-based Search on Heterogeneous Information Systems*. PhD thesis, University Of Zaragoza.

Budanitsky, A. y Hirst, G. (2006). Evaluating wordnet-based measures of lexical semantic relatedness. *Computational linguistics*, 32(1):13–47.

Chaudhri, V., Baru, C., Chittar, N., Dong, X., Genesereth, M., Hendler, J., Kalyanpur, A., Lenat, D., Sequeda, J., Vrandečić, D., et al. (2022). Knowledge graphs: Introduction, history and, perspectives. *AI Magazine*, 43(1):17–29.

Choi, E., Bahadori, M. T., Song, L., Stewart, W. F., y Sun, J. (2017). Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 787–795.

Chowdhary, K. (2020). *Fundamentals of artificial intelligence*. Springer.

Date, P. (2019). *Combinatorial Neural Network Training Algorithm for Neuromorphic Computing*. PhD thesis.

Dessì, D., Osborne, F., Reforgiato Recupero, D., Buscaldi, D., Motta, E., y Sack, H. (2020). Ai-kg: an automatically generated knowledge graph of artificial intelligence. In *The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II 19*, pages 127–143. Springer.

Ehrlinger, L. y Wöß, W. (2016). Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2.

Fellbaum, C. (2010). Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5-6):907–928.

Jean-Baptiste, L. (2021). *Ontologies with Python: Programming OWL 2.0 Ontologies with Python and Owlready2*. Springer.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., y Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Nickel, M. y Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30.

Pennington, J., Socher, R., y Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Rich, E. (1985). Artificial intelligence and the humanities. *Computers and the Humanities*, 19(2):117–122.

Siddharth, L., Blessing, L. T., Wood, K. L., y Luo, J. (2022). Engineering knowledge graph from patent database. *Journal of Computing and Information Science in Engineering*, 22(2):021008.

Singhal, A. (2012). Introducing the knowledge graph: things, not strings, may 2012. *URL http://googleblog. blogspot. ie/2012/05/introducing-knowledgegraph-things-not. html*.

Tifrea, A., Bécigneul, G., y Ganea, O.-E. (2018). Poincaré glove: Hyperbolic word embeddings. *arXiv preprint arXiv:1810.06546*.

Tounsi Dhouib, M., Faron, C., y Tettamanzi, A. G. (2021). Measuring clusters of labels in an embedding space to refine relations in ontology alignment. *Journal on Data Semantics*, 10(3-4):399–408.

Tounsi Dhouib, M., Faron Zucker, C., y Tettamanzi, A. G. (2019). An ontology alignment approach combining word embedding and the radius measure. In *Semantic Systems. The Power of AI and Knowledge Graphs: 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9–12, 2019, Proceedings 15*, pages 191–197. Springer International Publishing.

Yahya, M., Breslin, J. G., y Ali, M. I. (2021). Semantic web and knowledge graphs for industry 4.0. *Applied Sciences*, 11(11):5110.