Aprendizaje Automático Relacional

Inteligencia Artificial – Ingeniería del Software
Curso 2022/2023

Propuesta de trabajo del profesor Pedro Almagro Blanco

1. Introducción y objetivos

El aprendizaje automático relacional es una subdisciplina del aprendizaje automático que se ocupa del desarrollo de modelos orientados al **aprendizaje a partir de datos relacionales** [1].

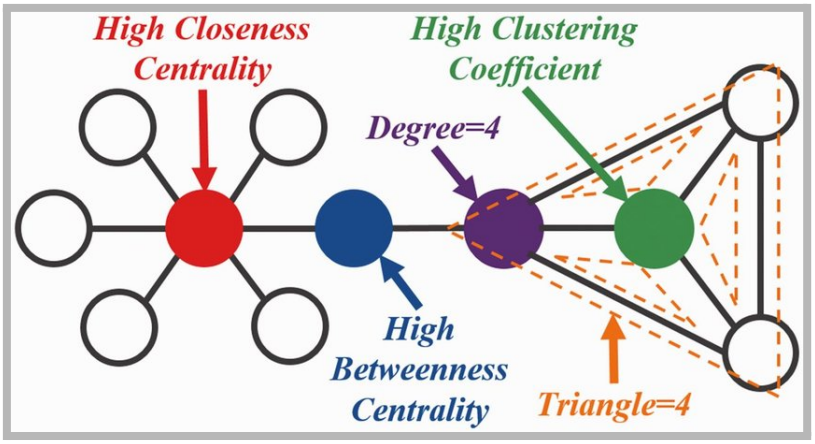
Es habitual utilizar la estructura matemática del **grafo** (grafo con propiedades en caso de que los datos posean propiedades además de participar en relaciones) para representar datos relacionales. A su vez, en los últimos años se han desarrollado bases de datos orientadas al almacenamiento y consulta de datos con una alta carga relacional. Estas bases de datos (que son más ágiles a la hora de consultar en base a relaciones) son denominadas bases de datos en grafo [6].

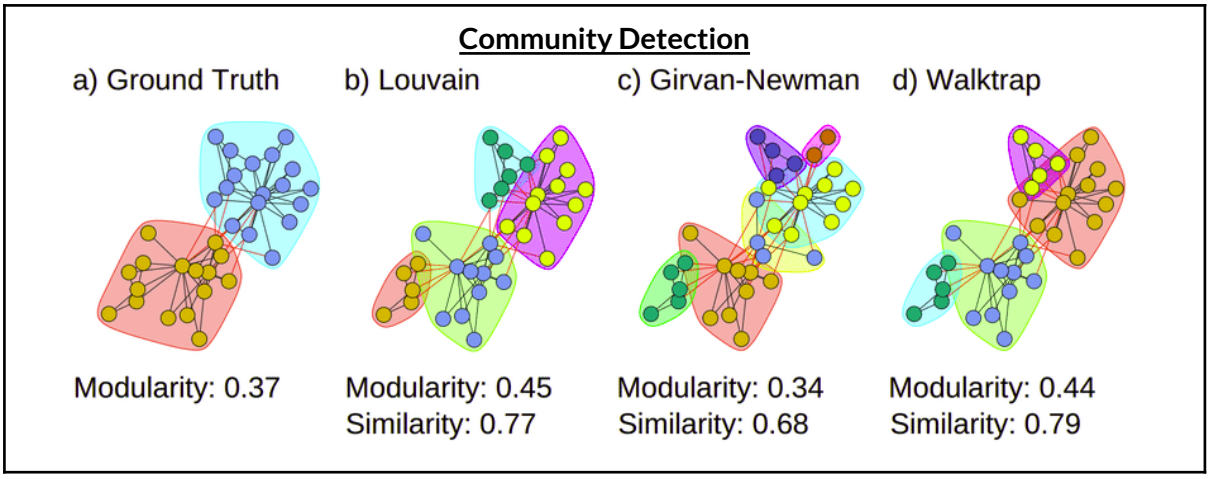
En los casos habituales de aprendizaje automático, los ejemplos están caracterizados normalmente por una serie de propiedades y no es necesario a priori capturar las relaciones entre los ejemplos a la hora de aprender. En el caso del aprendizaje automático relacional, los ejemplos poseen **relaciones** entre ellos (y éstas son tomadas en cuenta a la hora de generalizar) y opcionalmente, una serie de propiedades.

Existen varias metodologías que permiten aprender a partir de la información relacional en un conjunto de ejemplos, algunas de ellas son: la construcción manual de features relacionales (1), el aprendizaje de representaciones latentes en baja dimensión (2) y el uso de formalismos lógicos que permiten obtener patrones relacionales que caracterizan a los ejemplos (3).

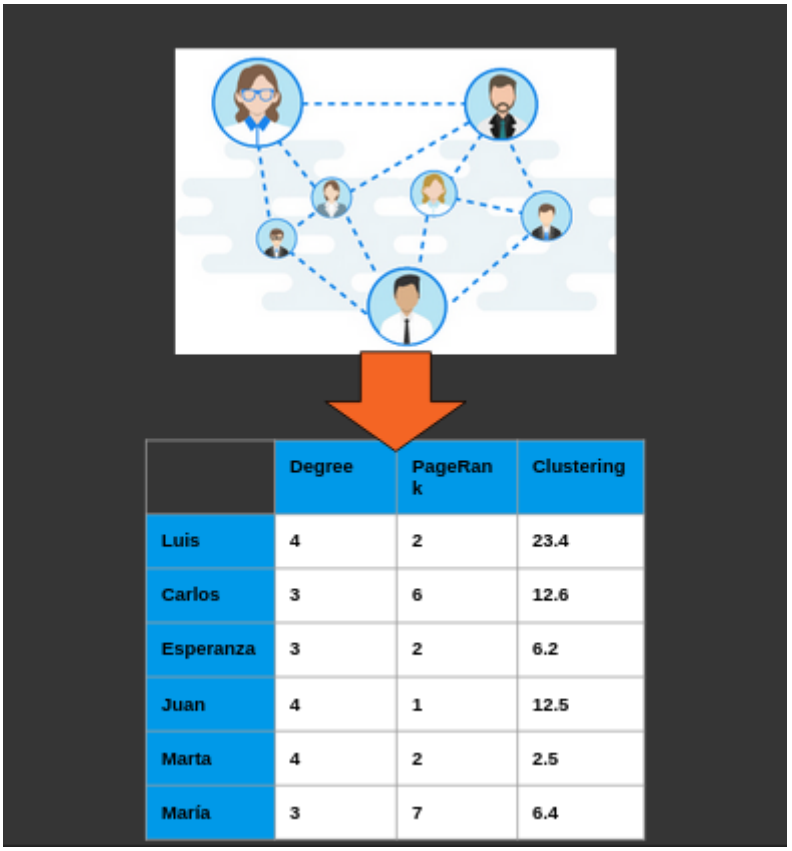
En este trabajo, nos centraremos en la **construcción manual de features relacionales** (1) para abordar un problema de clasificación relacional. En la configuración básica para aprender una tarea de clasificación sobre los nodos de una red, el objetivo es aprender una función que dadas las propiedades de un nodo en una red sea capaz de clasificar correctamente dicho nodo en una de las diferentes clases. Para ello, se espera que el sistema aprenda dicha función de clasificación a partir de una serie de ejemplos con nodos previamente clasificados, de los cuales se conocen sus propiedades.

En este trabajo, **diseñaremos y evaluaremos varios modelos de aprendizaje automático relacional para aprender a clasificar los nodos de un grafo a partir de sus propiedades y/o a partir de la información relacional**. Para ello, se utilizarán métricas propias de la teoría de grafos como propiedades predictivas de los nodos. Algunas de estas métricas están relacionadas con la *centralidad* [2], el *coeficiente de clustering* [4] o la *detección de comunidades* [3]. En los siguientes diagramas se da una representación gráfica de algunas de estas métricas:





El **objetivo principal** de esta propuesta es implementar modelos de clasificación de nodos en grafos. Para ello, caracterizaremos los nodos de un grafo a partir de información relacional (a través de métricas relacionadas con centralidad, detección de comunidades y clustering) y utilizaremos dicha caracterización para entrenar modelos ML estándar (árbol de decisión, redes neuronales, kNN…).



	Degree	PageRank	Clustering
Luis	4	2	23.4
Carlos	3	6	12.6
Esperanza	3	2	6.2
Juan	4	1	12.5
Marta	4	2	2.5
María	3	7	6.4

Una vez entrenados, los modelos ML deben ser capaces de clasificar nodos en base a su información relacional. Opcionalmente, se podrá enriquecer la caracterización de los nodos con propiedades específicas de éstos presentes en el grafo (en el caso de los grafos con propiedades).

Para ello, será necesario alcanzar los siguientes **objetivos específicos**:

1. Familiarizarse con los **conceptos básicos** de la teoría de grafos (al menos aquellos relacionados con la *centralidad*, la *modularidad* y el *coeficiente de clustering*).

2. Familiarizarse con las **librerías** “NetworkX” y “Scikit-Learn” de Python.

3. **Generar distintos modelos** - combinación de conjuntos de propiedades (relacionales y/o no relacionales) y algoritmo de aprendizaje automático - **y evaluar su rendimiento** sobre un conjunto de evaluación.

4. Conocer, comprender y aplicar distintas técnicas para el **análisis de la utilidad de las métricas** relacionales, la estimación de hiper-parámetros, test estadísticos de autocorrelación, etc., que ayuden al modelado y eviten el sobreajuste de éste a los datos de entrenamiento.

5. **Seleccionar**, de entre todos los modelos construidos, **aquel que realice una mejor clasificación** y que se considere que generaliza mejor.

6. **Documentar el trabajo** realizado usando un formato de artículo científico.

7. **Realizar una presentación** (PDF, PowerPoint o similar) de los resultados obtenidos.

2. Descripción del trabajo

Metodología

A continuación, se describe con más detalle cómo debe llevarse a cabo el trabajo. Para la consecución de los objetivos específicos, existen multitud de referencias y material en la web con ejemplos y videos de apoyo. Una referencia útil a la hora de conocer diferentes métricas para los nodos de un grafo es “Statistical analysis of network data with R” de Eric D. Kolaczyk, donde podéis encontrar los fundamentos del análisis de redes y una lista exhaustiva de métricas que pueden ser utilizadas como propiedades predictivas de los nodos.

Cada grupo de trabajo se encargará de una tarea de clasificación relacional solicitando al profesor el conjunto de datos con el que trabajar. Para ello, el grupo deberá escribir un correo a palmagro@us.es con el asunto “Trabajo IAIS Relacional (Grupo X)” donde el valor de X corresponderá al número del grupo de trabajo correspondiente. El profesor le hará llegar al grupo mediante correo eléctronico las instrucciones para descargar el dataset asignado. Ese conjunto de datos deberá ser tratado y adaptado cuando se requiera y se dividirá en un subconjunto de entrenamiento (train) y un subconjunto de validación (val). Se debe proceder, entonces, a analizar de manera exploratoria el conjunto de datos, obtener conjuntos de propiedades relacionales y/o no relacionales para los nodos, realizar análisis de residuos y autocorrelaciones, realizar ajuste de hiper-parámetros, entrenar los modelos a partir del subconjunto de entrenamiento y estimar su rendimiento sobre el subconjunto de validación.

Las medidas a usar para ello serán las correspondientes para medición de errores en tareas de clasificación. Se valorará la realización de visualizaciones para el análisis visual en todas las fases. Se recomienda utilizar entornos de programación y presentación para Python como Matplotlib y Jupyter.

IMPORTANTE: En caso de no superar la evaluación práctica en alguna de las convocatorias, **será obligatorio el cambio de conjunto de datos** (solicitándolo nuevamente al profesor).

Modelos

Se deben construir al menos **tres modelos de clasificación** relacionales, aunque pueden ser más, para poder realizar comparativas de rendimiento. La finalidad de construir varios modelos estriba en disponer de diferentes opciones entre las que elegir aquella que tenga una mejor combinación de métricas relacionales y/o propiedades y algoritmo de aprendizaje. Es decir, aquella combinación que generaliza mejor. Esta capacidad de generalización es lo que se estima mediante el subconjunto de validación a la hora de entrenar el modelo. El resultado obtenido influirá en la evaluación del trabajo.

Documentación y entrega

El trabajo deberá documentarse siguiendo un formato de artículo científico, con una extensión mínima de 6 páginas. En la página web de la asignatura se pueden encontrar plantillas donde se sugiere una estructura general. Estas plantillas siguen el formato de los *IEEE conference proceedings*, en cuyo sitio web se puede encontrar la guía para autores que ofrece información más detallada. El documento entregado deberá estar en formato PDF. Se valorará el uso del sistema LATEX. En el caso concreto de este trabajo, la memoria deberá al menos incluir: introducción; descripción del conjunto de datos y la tarea de predicción a realizar; descripción de las métricas relacionales utilizadas; descripción de los algoritmos de aprendizaje automático utilizados, explicando las dificultades encontradas y las decisiones de diseño adoptadas para abordarlas; descripción de los resultados alcanzados; conclusiones; bibliografía. **En ningún caso debe incluirse código en la memoria.**

La entrega del trabajo consistirá en un único fichero *zip* conteniendo la memoria del trabajo, el código implementado (ficheros .py o cuadernos de Jupyter) y el conjunto de datos utilizado.

Presentación y defensa

Como parte de la evaluación del trabajo se deberá realizar una defensa del mismo, para lo que se citará a los alumnos de manera conveniente. El día de la defensa se deberá realizar una pequeña presentación (PDF, PowerPoint o similar) de unos 5 minutos en la que deberán participar activamente todos los miembros del grupo que ha desarrollado el trabajo. Esta presentación deberá seguir a grandes rasgos la misma estructura que la memoria del trabajo, haciendo especial mención a los resultados obtenidos y al análisis crítico de los mismos. Se podrá usar un portátil (personal del alumno) y diapositivas. En los siguientes 5 minutos de la defensa, el profesor procederá a realizar preguntas sobre el trabajo, que podrán estar relacionadas tanto con la memoria como con el código fuente.

3. Evaluación del trabajo

Para que el trabajo pueda ser evaluado, se deben haber construido al menos tres modelos de clasificación relacionales. Además, es imprescindible asegurarse que todo el código proporcionado, librerías auxiliares, conjuntos de datos, etc., se pueden ejecutar en un notebook o fichero .py en una máquina independiente. Para la evaluación del trabajo se tendrán en cuenta los siguientes criterios, considerando una nota total máxima de 4 puntos:

1. **Memoria del trabajo** (aporta hasta 1,25 puntos): se valorará la claridad de las explicaciones, el razonamiento de las decisiones, el análisis y presentación de resultados y el correcto uso del lenguaje. La elaboración de la memoria debe ser original, por lo

que no se evaluará el trabajo si se detecta cualquier copia del contenido.

2. **Código fuente** (hasta 1,25 puntos): se valorará la claridad y buen estilo de programación, corrección y eficiencia de la implementación y calidad de los comentarios. El código debe ser original, por lo que no se evaluará el trabajo si se detecta código copiado o descargado de internet

3. **Modelo seleccionado** (hasta 1,5 puntos): se valorará, tanto de manera absoluta como comparativamente con el resto de los trabajos, el comportamiento de los modelos en la tarea de predicción, así como el análisis exploratorio inicial (métricas relacionales / modelos de aprendizaje automático), búsqueda de hiper-parámetros y análisis de residuos. El modelo seleccionado final debe estar claramente indicado y debe poder cargarse correctamente, en caso contrario este criterio se valorará con 0 puntos.

4. **Presentación y defensa** (factor multiplicativo): la nota final de cada estudiante del equipo estará condicionada por su participación en la defensa. De tal manera que la nota obtenida a partir de los tres apartados anteriores se multiplicará por un factor en el rango [0,1] correspondiente a la calidad de la participación del estudiante en la defensa del trabajo. Se valorará la claridad de la presentación y la buena explicación de los contenidos del trabajo, así como especialmente, las respuestas a las preguntas realizadas por el profesor.

IMPORTANTE: cualquier plagio, compartición de código o uso de material que no sea original y del que no se cite convenientemente la fuente, significará automáticamente la calificación de cero en la asignatura para todos los alumnos involucrados. Por tanto, a estos alumnos no se les conserva, ni para la actual ni para futuras convocatorias, ninguna nota que hubiesen obtenido hasta el momento. Todo ello sin perjuicio de las correspondientes medidas disciplinarias que se pudieran tomar.

Referencias

[1] Struyf J., Blockeel H. (2011) Relational Learning. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_719

[2] Koschützki, D.; Lehmann, K. A.; Peeters, L.; Richter, S.; Tenfelde-Podehl, D. and Zlotowski, O. (2005) Centrality Indices. In Brandes, U. and Erlebach, T. (Eds.) *Network Analysis: Methodological Foundations*, pp. 16–61, LNCS 3418, Springer-Verlag.

[3] S. Fortunato (2010). "Community detection in graphs". Phys. Rep. 486 (3–5): 75–174. arXiv:0906.0612. Bibcode:2010PhR...486...75F. doi:10.1016/j.physrep.2009.11.002. S2CID 10211629.

[4] P. W. Holland & S. Leinhardt (1971). "Transitivity in structural models of small groups". *Comparative Group Studies*. **2** (2): 107–124. doi:10.1177/104649647100200201. S2CID 145544488.

[5] Kolaczyk, E. D., & Csárdi, G. (2014). *Statistical analysis of network data with R* (Vol. 65). New York: Springer.

[6] *ArangoDB vs MySQL Performance Benchmarking*.