Aprendizaje automático relacional aplicado a un conjunto de datos

Nicolás Sánchez Mendoza
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
nicsanmen@alum.us.es, nicsanmen

Mariano Martín Avecilla
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
marmarave, marmarave@alum.us.es

***Resumen*—El aprendizaje trata sobre cualquier cambio en un sistema que le permita realizar la misma tarea de manera más eficiente la próxima vez. Si nos centramos en el aprendizaje automático en concreto, se trata en construir programas que mejoran automáticamente con la experiencia. Por último, podemos decir que el aprendizaje automático relacional combina lo anterior además de tener en cuanta propiedades relacionales. A partir de un modelo de datos nos centramos en construir manualmente atributos relacionales para abordar un problema de clasificación relacional. Para ello contaremos con una red compuesta por nodos. El objetivo es que, dado el modelo, lo entrenamos y le apliquemos un tipo de validación con un método determinado y a partir de ahi, compararlo con otro modelo en el cual se han añadido propiedades relacionales y ver el comportamiento de uno sobre el otro a través de la exactitud obtenida en ambos casos.**

***Palabras clave*—Aprendizaje Automático, Aprendizaje Automático Relacional, Inteligencia Artificial, Atribuos relacionales, Modelos...**

I. Introducción

El Aprendizaje Automático (AA), en inglés Machine Learning, es uno de los campos claves de investigación en la Inteligencia Artificial. El AA se basa fundamentalmente en la Matemática y las Ciencias de la Computación; y se define como el área de estudio que se encarga de desarrollar algoritmos que permiten a las computadoras adquirir habilidades (a través de la experiencia) para las que no fueron explícitamente programadas. Normalmente, el Aprendizaje Automático aprende de datos estructurados de manera tabular, encontrando patrones similares entre los datos del conjunto afectado. Tratando de ir más allá del análisis convencional, en el cual se estudian los objetos aisladamente; se desea aprender a partir de las relaciones en las que participan los objetos. Gracias a esta necesidad nace el Aprendizaje Automático Relacional. El Aprendizaje Automático Relacional busca mejorar predicciones mediante propiedades relacionales, usando las conexiones entre las entidades para el descubrimiento de patrones. El trabajo que hemos desarrollado ha estado enfocado al aprendizaje automático relacional. Para su realización contamos con el entorno de desarrollo de jupyter a través de la herramienta de notebook. Las principales librerías que hemos usados han sido pandas, networx, scikit-learn aunque también hemos usado algunas pero con menor impacto como pueden ser keras y tensorflow. Contamos con un dataset obtenido de la página web https://icon.colorado.edu/ en el apartado metadata. Esta página fue la que nos recomendó nuestro profesor y tutor del trabajo. Para la elección de nuestros dataset tuvimos en cuenta que tuviese formato dataset y que además tuviera los suficientes atributos, nodos y aristas para realizar un trabajo más competente. Finalmente, elegimos un dataset relacionado con quakers. Los quakers eran una sociedad religiosa de amigos de los años 1600-1700 aproximadamente. A partir de este dataset, leímos su contenido a través de la librería pandas y luego procedimos a la representación del grafo mediante la librería matplot. En la figura 1 podemos observar una imagen de la tabla de datos obtenida a partir de la librería pandas y en la figura 2 podemos ver el grafo generado gracias a las librerías matplot y networx.

En segundo lugar, nos documentamos y aplicamos distintos métodos a través de la librería de networx para obtener las siguientes propiedades relacionales: centralidad, clustering, valencias de cada nodo y centralidad de intermediación. Todos estos atributos se añadieron a un nuevo modelo y se le aplicaron los siguientes métodos de clasificación: k-nn, redes neuronales, random forest y árboles de decisión. También hay que destacar que hemos usado validación cruzada atendiendo a la estructura de nuestros datos para asegurarnos que la partición entre datos de pruebas y entrenamientos son independientes. Nuestro objetivo ha sido a partir de una serie de datos, determinar cómo atributos la fecha de muerte y de nacimiento de nuestros personajes históricos que pertenecían a la sociedad anteriormente nombrada y como objetivo el género de cada uno de ellos. A partir de ahí comprobar la exactitud aplicando diferentes métodos y luego añadirle los atributos relacionales también nombrados con anterioridad. Una vez entrenados ambos conjuntos, ver cual nos aporta una mayor exactitud y ver si los datos añadidos a nuestro conjunto nos aportan información relevante a nuestro modelo. Si es así la exactitud mejora.

II. Preliminares

En este trabajo hemos hecho uso de los modelos knn (k-nearest neighbors), redes neuronales, árboles de decisión y bosque aleatorio (random forest). Todos estos modelos se han utilizado tanto con las propiedades del propio grafo como con las propiedades relacionales que hemos añadido. Nos

	name	historical_significance	gender	birthdate	deathdate	id
0	Joseph Wyeth	religious writer	male	1663	1731	10013191
1	Alexander Skene of Newtyle	local politician and author	male	1621	1694	10011149
2	James Logan	colonial official and scholar	male	1674	1751	10007567
3	Dorcas Erbery	Quaker preacher	female	1656	1659	10003983
4	Lilias Skene	Quaker preacher and poet	male	1626	1697	10011152
...	...	...	...	...	...	...
114	Thomas Ellwood	religious controversialist	male	1639	1713	10003945
115	William Simpson	Quaker preacher	male	1627	1671	10011114
116	Samuel Bownas	Quaker minister and writer	male	1677	1753	10001390
117	John Perrot	Quaker schismatic	male	1555	1665	10009584
118	Hannah Stranger	Quaker missionary	female	1656	1671	10011632

119 rows × 6 columns

Fig. 1. Datos obtenidos del fichero csv con la libreria pandas

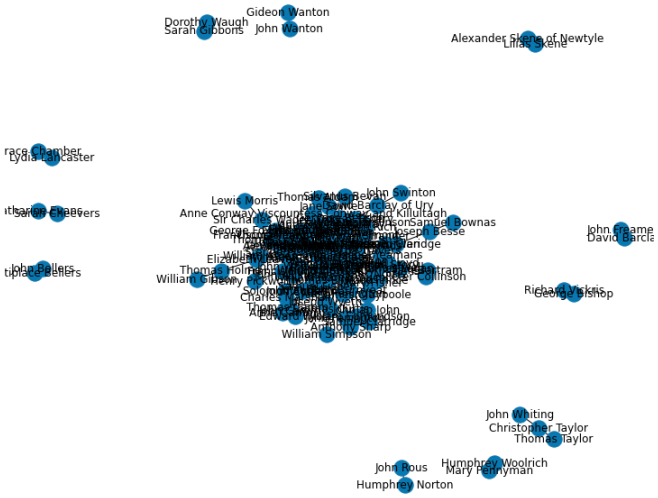

Fig. 2. Grafo obtenido del fichero csv con las librerias matplotlib y networx

	name	historical_significance	gender	birthdate	deathdate	id	degree	centrality	clustering	bcentrality
0	Joseph Wyeth	religious writer	male	1663	1731	10013191	1	0.008475	0.000000	0.000000
1	Alexander Skene of Newtyle	local politician and author	male	1621	1694	10011149	1	0.008475	0.000000	0.000000
2	James Logan	colonial official and scholar	male	1674	1751	10007567	4	0.033898	0.333333	0.026945
3	Dorcas Erbery	Quaker preacher	female	1656	1659	10003983	1	0.008475	0.000000	0.000000
4	Lilias Skene	Quaker preacher and poet	male	1626	1697	10011152	1	0.008475	0.000000	0.000000
...	...	...	...	...	...	...	...	...	...	...
114	Thomas Ellwood	religious controversialist	male	1639	1713	10003945	8	0.067797	0.178571	0.046191
115	William Simpson	Quaker preacher	male	1627	1671	10011114	1	0.008475	0.000000	0.000000
116	Samuel Bownas	Quaker minister and writer	male	1677	1753	10001390	1	0.008475	0.000000	0.000000
117	John Perrot	Quaker schismatic	male	1555	1665	10009584	7	0.059322	0.333333	0.028296
118	Hannah Stranger	Quaker missionary	female	1656	1671	10011632	2	0.016949	1.000000	0.000000

Fig. 3. Datos con los atributos relacionales añadidos

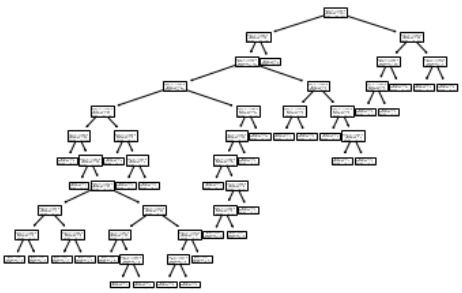

Fig. 4. Árbol de decisión

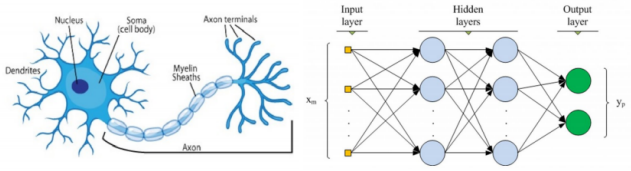

Fig. 5. Redes neuronales

hemos apoyado para ello en las prácticas de la asignatura, concretamente en la práctica 1 (aprendizaje automático) y en la práctica 2 (redes neuronales). Además de las prácticas hemos contado con la ayuda de nuestro tutor con varias tutorías grupales convocadas en las cuales hemos aprendido mucho y nos ha guiado como trabajar en todo momento.

A. Métodos empleados

La primera técnica usada ha sido el método kNN o también conocido como k-nearest-neighbor. Este método sirve para clasificar valores discretos y para predecir. Su principal uso es clasificar valores buscando los putos de datos más similares por cercanía aprendidos en la etapa de entrenamiento y haciendo conjeturas de nuevos puntos basados en esa clasificación.

La segunda técnica usada ha sido árboles de decisión. Es un método analítico que a través de una representación esquemática de las alternativas disponible facilita la toma de mejores decisiones. Los árboles de decisión son interesantes en el proceso de análisis de datos ya que son robustos frente a valores atípicos y valores perdidos, por lo cual se da una menor limpieza de datos.

La tercera técnica usada ha sido redes neuronales. Es uno de los algoritmos más populares y robustos. Consisten en neuronas artificiales interconectadas que relacionan las entradas con las salidas deseadas. El objetivo principal de este modelo es aprender modificándose automáticamente a si mismo de forma que puede llegar a realizar tareas complejas que no podrían ser realizadas mediante la clásica programación basada en reglas. De esta forma se pueden automatizar funciones que en un principio solo podrían ser realizadas por personas.

Finalmente, la cuarta técnica usada ha random forest(bosques aleatorios) es una combinación de árboles predictores tal que cada árbol depende de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de estos. Es uno de los métodos más sencillos de implementar.

B. Trabajo Relacionado

Como mencionamos anteriormente, para la realización de este trabajo nos han sido muy útiles las prácticas de la asignatura en la que trabajábamos con el aprendizaje automático y con las redes neuronales. También nos pudimos apoyar en la teoría que hemos visto durante el cursado de la asignatura y, sobre todo, nos ayudó mucho las tutorías y revisiones con nuestro profesor y tutor Pedro. Gracias a todo

ello conseguimos darle el enfoque que queríamos y realizar el proyecto.

III. Metodología

Esta sección se dedica a la descripción del método implementado en el trabajo. Esta parte es la correspondiente a lo realmente desarrollado en el trabajo, y se puede emplear pseudocódigo (nunca código), esquemas, tablas, etc.

Durante este trabajo hemos trabajado como mencionamos con un dataset sobre los Quakers, que no era más que un archivo .csv con de un grafo en formato edge-list. Para poder crearlo, hemos hecho uso de la librería pandas, que se encargaba de leer el csv y la librería networkx para así generar el grafo. Seleccionamos como atributos la fecha de nacimiento y de muerte y como objetivo añadimos el género, pues, tras muchas pruebas con todos los atributos entre ellos mismos, vimos que el único que mantenía algún tipo de relación para ser el objetivo era el género y que los únicos atributos relevantes para esto son los mencionados anteriormente, descartando así el resto que si bien eran obvios que no se podría sacar una relación clara (por tratarse de un id, un nombre y la relevancia histórica del personaje) aun así lo probamos para comprobarlo. También ayudó a la hora de elegir los atributos y el objetivo el hecho de que sean valores numéricos, siendo mucho más fácil para trabajar con ellos. A partir de ahí seleccionamos las propiedades relacionales con las que íbamos a trabajar, para ellos nos apoyamos en las tutorías de Pedro, así como en la documentación de la librería networkx. Finalmente elegimos utilizar la centralidad del grado, clustering, valencia y b-centrality. La centralidad de grado es la más simple, básicamente se calcula viendo el grado del nodo y dividiendo por el número total de nodos. La siguiente propiedad fue el clustering (agrupamiento) que en resumen nos dice los posibles triángulos que se pueden generan en un nodo 'u'*2 dividido entre el grado de 'u' * el grado de ('u'-1). La siguiente propiedad que añadimos fue simplemente el grado, la valencia de cada método decidimos que nos podría indicar de una forma sencilla lo "popular" que era ese nodo y poder ver de manera sencilla la cantidad de relaciones que había en él. La propiedad relacional que decidimos añadir en último lugar fue la centralidad de intermediación. La centralidad de intermediación (betweenness centrality) de un nodo es la suma de las fracciones de todos los pares de caminos más cortos que lo atraviesan. Cabe destacar que intentamos añadir una última propiedad relacional, que era la de comunidades. Pensamos que sería interesante e intuitivo ver la relación que tenían unos nodos con otros mediante el uso de esta propiedad, que básicamente nos divide el grafo en diferentes secciones. Pero por temas de tiempo y de desconocimiento no conseguimos finalmente añadirla, por tanto, optamos por simplemente añadir las cuatro mencionadas antes.

Pasamos ahora a hablar sobre los modelos empleados. Para este trabajo hemos decidido utilizar los siguientes modelos: k-nn (k-vecinos más próximos), redes neuronales, bosque aleatorio, y árboles de decisión. Destacar que intentamos añadir un último modelo que fue el de clasificación por Naive Bayes pero no fue posible debido a que los atributos requeridos de entrada debían ser categóricos, entonces el profesor no nos lo recomendó. Por tanto, empezamos utilizando el modelo de k-nn, el método k-neighbors permite encontrar los (índices de los) k-vecinos más cercanos de los ejemplos proporcionados, así como las distancias a las que se encuentran. Este método lo pudimos utilizar fácilmente apoyándonos en la primera práctica y la teoría de clase. El siguiente modelo que usamos en el proyecto es el de árbol de decisión como se puede ver en la Fig.4, este algoritmo es uno de los más usados dentro del Aprendizaje Automático Supervisado ya que puede construir límites de decisión complejos al dividir el espacio de características en rectángulos, de tal manera que se obtiene mayor precisión y estabilidad en la predicción. El objetivo de este algoritmo consiste en dividir de manera recursiva el conjunto de entrenamiento en subconjuntos cada vez más homogéneos utilizando todas las variables disponibles. El criterio de división se realiza de acuerdo a algoritmos que permiten decidir las variables más significativas, en nuestro caso, hemos elegido el criterio de validación cruzada atendiendo a la naturaleza de nuestro dataset. El siguiente modelo que utilizamos fue el de bosque aleatorio. Los bosques aleatorios son un conglomerado de varios árboles de decisión que en ocasiones mejora el desempeño obtenido por un solo árbol. Logran la construcción de un modelo robusto a la hora de tener una generalización más acertada y evitar el sobreajuste. La idea es subdividir el espacio de características de entrada en regiones más pequeñas que se vuelven más manejables. Un bosque aleatorio (BA) tiene un mejor rendimiento de generalización que un árbol de decisión individual debido a la aleatoriedad, que ayuda a disminuir la varianza del modelo. Además, son menos sensibles a los valores atípicos y no requieren mucho ajuste de parámetros en vista de que el único parámetro que usualmente se necesita experimentar es el número de árboles con el que se va a entrenar y los parámetros habituales en los árboles de decisión. Al entrenar un BA los árboles individuales que lo componen toman un subconjunto aleatorio de las características que componen el fenómeno a predecir. Sin embargo, al tratar de interpretarlo se comportan como una caja negra ya que debido a la combinación de árboles es difícil comprender la relación entre las variables. Este algoritmo se usa cuando se tiene como prioridad el desempeño predictivo del modelo en lugar de la interpretación de las características. Aun así, hemos visto interesante el añadir un modelo de este estilo para ver cómo se comportaba tanto nuestro dataset como nuestras propiedades ante un modelo de estas características. Por último, elegimos como modelo la utilización de redes neuronales (un ejemplo gráfico de una red neuronal humana y una aplicada a la inteligencia artificial se puede apreciar en la fig5), uno de los algoritmos más populares y robustos dentro del Aprendizaje Automático, las cuales, inspiradas en el cerebro humano, y consisten en neuronas artificiales interconectadas que relacionan las entradas con las salidas deseadas. Las redes neuronales artificiales están normalmente organizadas en capas, donde cada una de estas consta de varias

neuronas que tienen una función de activación, de manera que se interconectan y asignan pesos correspondientes a cada conexión en relación con la salida deseada. En este caso tuvimos unos pequeños problemas debido a que no sabíamos aplicar del todo bien el algoritmo y no podíamos importar correctamente las librerías con las que se utilizan (que son tensorflow y keras). Apoyándonos en la práctica resuelta y en los conocimientos adquiridos en la asignatura durante este curso, conseguimos hacer funcionar el modelo.

Una vez explicado todo esto, hay que añadir dos cosas: primero de todo aclarar como dijimos antes, que todos nuestros modelos utilizan validación cruzada (recomendada por nuestro tutor) a la hora de entrenar y probar el conjunto. La validación cruzada (cross-validation) es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Consiste en repetir y calcular la media aritmética obtenida de las medidas de evaluación sobre diferentes particiones. Y en segundo lugar, añadir para finalizar que una vez realizados todos los modelos para todos los atributos del grafo, se realizaron añadiendo las propiedades relacionales anteriormente nombradas, para así poder comparar los resultados obtenidos en las distintas mediciones. Esto lo mencionaremos con más profundidad en el próximo punto.

IV. Resultados

En esta sección detallaremos tanto los experimentos realizados como los resultados conseguidos:

En primer lugar, realizamos una lectura del fichero gracias a la librería pandas y el resultado fue exitoso ya que pudimos contrastarlo imprimiendo los datos por pantalla y comprobando que las filas y columnas coincidan con el fichero csv del que disponíamos. El dataset que elegimos fue acorde al formato que tenía que era de tipo edge-list y no teníamos que aplicarle ningún método a la hora de tratarlo como podía ocurrir en los tipos matriz de adyacencia.

En segundo lugar, aplicamos la obtención de los atributos relacionales, el único problema que tuvimos fue con el atributo community que el resultado era una lista de string y no conseguimos tratarlos para que los modelos trabajasen con él. Con el resto de los atributos no tuvimos problemas y la solución que le dimos al problema anterior fue cambiar dicho atributo por el atributo centralidad de intermediación.

Ahora pasamos con los métodos usados: Método kNN: No tuvimos ningún problema a la hora de realizar este método ya que estábamos bastante familiarizado con él. El resultado fue que al aplicar el algoritmo a los datos sin los atributos relacionales la exactitud generada tenía un valor aproximado a 0.83 mientras que cuando le añadíamos los atributos relacionales la exactitud ascendía 1. Por tanto, concluimos que el modelo añadiendo las propiedades relacionales mejora el rendimiento posiblemente porque estos atributos guarden algún tipo de relación directa.

Árboles de decisión: El único problema que tuvimos no fue un tema algoritmo o del propio modelo sino un problema más bien en cuanto al entorno jupyter, ya que disponíamos de una versión demasiado reciente lo que impida el uso de ciertos métodos de la librería sklearn. El resultado que generó el algoritmo para determinar la exactitud fue sin los atributos relacionales de 0.98 aproximadamente y de 1 con los atributos relacionales. Por tanto, se puede concluir que añadiendo los atributos relacionales mejora la exactitud al igual que pasaba con el método anterior.

Redes neuronales: En este caso los problemas que nos surgieron fueron de una índole más teórica que práctica. Tuvimos que reforzar conocimientos y hacer distintas pruebas con el algoritmo hasta obtener los resultados que considerábamos más óptimos. El resultado sigue en la misma línea que los anteriores métodos, aunque hemos obtenido una exactitud menor con ambos modelos que con los otros métodos. 0.48 con los atributos originales frente a 0.55 con los atributos relacionales. Por tanto, podemos concluir que la exactitud mejora con los atributos relacionales.

Bosques aleatorios. Este fue el método más sencillo de implementar ya que no tuvimos ningún problema, sólo tuvimos que seguir las instrucciones dadas en la documentación. Los resultados fueron similares a los anteriores casos y el entrenamiento mejora con los atributos relacionales. Concretamente 0.92 frente a 1 de exactitud.

Podemos observar que tres de los cuatro métodos que hemos utilizado han obtenido un alto valor en la exactitud en el modelo original y además la exactitud máxima cuando añadíamos los atributos relacionales. Pero si nos tenemos que quedar con uno sobre los otros nos quedaremos con el que ha tenido mayor rango de mejora. Por tanto, el algoritmo que mejor se ha comportado atendiendo a nuestro modelo ha sido el KNN ya que nos ha proporcionado una mayor diferencia de exactitud entre el modelo sin atributos relacionales y el modelo con atributos relacionales además de que la exactitud del modelo relacional ha sido de 1.

V. Conclusión

En conclusión hemos aprendido bastante en este proyecto. Sobretodo hemos mejorado el manejo del lenguaje de programación Pythhon, hemos aprendido a ser autodidactas y a encontrar y entender documentaciones oficiales de los métodos y librerías, hemos detectado los errores y hemos sabido solucionarlos, también nos hemos familiarizado con la consola de jupiter para actualizar o versionar paquetes que necesitábamos, hemos aprendido a redactar documentos con formato científico en la herramienta TeXworks con formato Latex.

En cuanto al proyecto, hemos adquirido bastantes conocimientos. En primer lugar, nos hemos familiarizado con las librerías pandas y numpy para lectura de datos. Para la formación del grafo hemos usado la librerias networkx y para su representación la librería matplotlib. Para la creación de atributos relacionales hemos usado métodos de las librerías networkx y para integrarlos con el dataset métodos de las librerías pandas.

Por último, para aplicar los distintos métodos de aprendizaje que se describen a continuación hemos usado: -Método KNN:

librería sklearn. -Árboles de decisión: librería sklearn. -Redes neuronales: librería keras. -Bosques aleatorios: librería sklearn.

En conclusión, hemos determinado que el aprendizaje mejora tras añadirle los atributos relaciones en todos los métodos menos en el método redes neuronales. Hemos llegado a esta conclusión porque creemos que los atributos relacionales añadidos guardan algún tipo de relación relevante entre los atributos para lograr el objetivo especificado. También hemos determinado que en el caso de las redes neuronales el resultado no tiene una exactitud tan elevada en ninguno de los dos casos porque el algoritmo necesita un mayor entrenamiento aumentando el número de capas y neuronas para obtener un resultado similar a los demás modelos aplicados. Aun así la exactitud obtenida al aplicarlo al modelo con atributos relacionales es mayor que al aplicarlo al modelo original. Por tanto, podemos concluir que los atributos relacionales siguen teniendo una relación directa con los atributos del modelo.

Finalmente, queríamos recalcar también que hemos echado en falta una herramienta de control de versiones para trabajar simultáneamente en un proyecto común. La manera en la que lo hemos hecho ha sido ir enviando versiones del proyecto por discord y asegurándonos que ninguno pisaba trabajo de otro.

Como conclusión final, decir que nos ha gustado mucho el trabajo, que nuestro tutor nos ha acompañado en todo momento y siempre se ha prestado a ayudar y a resolver cualquier tipo de dudas y que hemos trabajado muy cómodamente los dos ya que además de compañero somos amigos y siempre hemos intentado ayudarnos el uno al otro en cualquier tipo de problema.

REFERENCIAS

[1] https://www.aprendemachinelearning.com Teoría sobre aprendizaje automático
[2] https://living-sun.com/ Información para actualizar paquetes.
[3] https://www.askpython.com/ Informaciñon sovre la librería sklearn
[4] https://6dfb.tumblr.com/post/164795439991/six-degrees-of-pennsylvania-william-penn-and Documentación sobre nuestro conjunto de datos
[5] https://scikit-learn.org Obtener información y uso de librerías para la validación cruzada
[6] https://scikit-learn.org Información sobre el métdo KNN
[7] https://stackoverflow.com/ Hemos resuleto alguna que otra duda puntual gracias a esta página.
[8] https://www.cs.us.es/docencia/aulavirtual/ Apuntes y prácticas de las cuales nos hemos guiado.
[9] I Mitchell, T.M. Machine Learning (McGraw-Hill, 1997) I Caps. 3,6,8 y 10
[10] I Russell, S. y Norvig, P. Artificial Intelligence (A Modern Approach) (3rd edition) (Prentice Hall, 2010) I Seccs. 18.1, 18.2, 18.3, 20.1 y 20.2
[11] I Witten, I.H. y Frank, E. Data mining (Second edition) (MorganKaufmann Publishers, 2005) I Cap. 3, 4, 5 y 6.