Facultad de Ciencias Experimentales
Grado en Biología

UNIVERSIDAD DE JAÉN
Facultad de Ciencias Experimentales

Trabajo Fin de Grado

Estudio del funcionamiento de técnicas de minería de datos sobre Conjuntos de Datos relacionados con la Biología

Alumno/a: **Martín Moret Sánchez**

Junio, 2021

Universidad de Jaén

Facultad de Ciencias Experimentales

Trabajo Fin de Grado

Estudio del funcionamiento de técnicas de minería de datos sobre Conjuntos de Datos relacionados con la Biología

Alumno: Martín Moret Sánchez

Jaén, junio, 2021

ÍNDICE

RESUMEN

En la actualidad se generan volúmenes masivos de datos en todos los ámbitos, desde datos con propósitos económicos, como los recogidos por empresas para predecir tendencias de compra de los consumidores, hasta datos de origen clínico como historiales médicos de pacientes. Para que estos datos sean útiles hacen falta herramientas de minería de datos. Una de las más potentes y más populares hoy día es el aprendizaje automático o "machine learning", disciplina que emplea una amplia variedad de algoritmos que generan a su vez otros algoritmos para descubrir patrones en los datos. En este proyecto se pretendió analizar el rendimiento y comportamiento de un total de 8 algoritmos basados en reglas sobre 30 conjuntos de datos relacionados con el ámbito de la biología mediante un experimento en KEEL, observando cómo afectan diferentes factores al rendimiento del proceso de aprendizaje y determinando cuales de los algoritmos eran más eficaces. De manera adicional, estudiamos más detenidamente un caso concreto de estudio para la mejora de la comprensión del funcionamiento de los algoritmos basados en reglas.

Palabras claves: aprendizaje automático, aprendizaje supervisado, clasificación, aprendizaje basado en reglas, minería de datos

ABSTRACT

Massive volumes of data are currently being generated in all spheres, from data for economic purposes, such as those collected by companies to predict consumer purchasing trends, to data of clinical origin such as patient medical records. To make this data useful, we need data mining tools. One of the most powerful and most popular today is machine learning, a discipline that uses a wide variety of algorithms that in turn generate other algorithms to discover patterns in the data. This project aimed to analyse the performance and behaviour of a total of 8 rule-based algorithms on 30 data sets related to the sphere of biology through an experiment in KEEL, observing how different factors affect the performance of the learning process and determining which of the algorithms are more efficient. Additionally, we study a specific case to improve the understanding of the operation of rule-based algorithms.

Keywords: machine learning, supervised learning, classification, rule-based learning, data mining

1. INTRODUCCIÓN

En la actualidad vivimos en lo que se conoce como la “Era del Big Data”, un término que hace referencia a la cantidad masiva de datos que se generan y que están al servicio de investigadores de diferentes ramas de conocimiento o al servicio de las entidades y empresas encargadas de su recopilación (Benke y Benke, 2018). Hoy día se generan volúmenes cada vez mayores y de mayor complejidad de datos procedentes de todos los ámbitos de la vida de las personas, desde aquellos que utilizan las empresas para la mejora de sus productos y servicios para sugerir mejores recomendaciones (Batmaz *et al.*, 2018), como pueden ser los datos de navegación que recogen empresas como Google o Amazon, hasta los datos médicos y biológicos que se recopilan tanto en investigaciones científicas, como en clínicas y hospitales con el fin de mejorar la calidad y esperanza de vida de las personas (Marx, 2013).

En la última década ha habido una explosión en la generación precisamente de esos datos médicos y biológicos debido en gran parte a la digitalización de registros médicos de pacientes y a la reducción del coste de técnicas moleculares, lo que permite entre otros muchos factores recoger más datos de pacientes, generando conjuntos de datos en los que cada paciente supondría una instancia o ejemplo, y las diferentes características clínicas los atributos. Esto ha llevado además a que los datos que se generen tengan, por naturaleza, una gran variedad y dimensionalidad (gran número de atributos por cada instancia) obteniéndose datos de todo tipo, desde datos basados en texto hasta datos basados en video, pasando por aquellos basados en imágenes o sonidos, lo que conlleva a su vez una mayor complejidad a la hora de tratar con ellos (Benke y Benke, 2018). Sin embargo, ese gran volumen de datos por sí mismo no tiene ningún valor, es necesario un procesamiento y análisis para transformar esos datos en información útil que pueda resultar en un beneficio humano, razón por la cual muchas compañías ya han empezado a aplicar diversas herramientas de análisis y procesamiento de esos datos de origen clínico para poder extraer aplicaciones útiles en clínica (Deo, 2013). Es ahí donde entra lo que se conoce como minería de datos.

1.1. Minería de datos

El termino minería de datos o *"*data mining*"* hace referencia al proceso computacional que se aplica con frecuencia para analizar grandes conjuntos de datos con el objetivo de descubrir patrones subyacentes a los mismos, extraer conocimiento que pueda ser procesable y predecir futuros resultados de eventos próximos. Para ello la minería de datos se nutre de métodos basados en una combinación muy variada de disciplinas incluyendo Matemáticas, Estadística, Inteligencia Artificial y Aprendizaje automático. (Bellinger *et al.*, 2017).

De este modo, mediante la utilización de algoritmos computaciones se pueden extraer resultados de análisis de los conjuntos de datos de interés, existiendo además algoritmos no solo para el análisis en sí, sino también para el preprocesamiento y el postprocesamiento de dichos datos con el fin de visualizar más fácilmente los resultados obtenidos del análisis, siendo especialmente útiles estos algoritmos en situaciones en las que los datos poseen un gran número de instancias o ejemplos, cada uno de ellos con un gran número de variables (Bellinger *et al.*, 2017). Entre las diferentes herramientas que se utilizan en la minería de datos, sin duda una de las que ha dado mejores resultados es la ya mencionada Inteligencia Artificial, consiguiendo resolver un alto porcentaje de problemas y paradigmas frente a los cuales la estadística clásica no conseguía obtener unos resultados satisfactorios (Bellinger *et al.*, 2017).

1.2. Inteligencia Artificial

Inteligencia Artificial o IA es el nombre que se le da al proceso mediante el cual una máquina o software responde a la entrada de nuevos datos por parte del entorno con la mínima intervención humana posible (Hamet y Tremblay, 2017). El sistema, dependiendo de la información que reciba el algoritmo de ese entorno, decidirá actuar de una forma diferente con el objetivo de mejorar su respuesta, por lo que el objetivo último de la Inteligencia Artificial, y por lo que recibe su nombre, es por la cualidad de las máquinas basadas en estos sistemas de imitar el comportamiento humano y tomar decisiones en función de la situación, es decir, relacionarse con el entorno (Benke y Benke, 2018).

Aunque es una disciplina de la computación que surgió hace relativamente poco tiempo, concretamente en la conferencia de Darmouth de 1956 cuando el informático John McCarthy acuñó por primera vez el termino, su desarrollo ha sido increíblemente acelerado sobre todo en lo que respecta a la automatización de procesos en la industria y los servicios, donde se pueden definir unas respuestas o salidas claras en función de la información de entrada que se le dé al algoritmo (Russel, 2021). Ese desarrollo acelerado derivo en lo que se denomina Aprendizaje Automático o *"*Machine Learning*"*, provocando una auténtica revolución dentro del análisis de datos (Leiner *et al.,* 2019).

1.3. Aprendizaje automático

El Aprendizaje Automático es una disciplina de las ramas de conocimiento de la computación que centra sus esfuerzos en comprender como aprenden las máquinas de los datos de entrada que se le proporcionan, con la peculiaridad de que se pretende que las máquinas no tengan que ser explícitamente programadas por un ser humano, sino que ella misma sea la que genere el algoritmo para buscar una solución al problema que se pretende resolver (Deo, 2013), de modo que se utiliza en la ya mencionada minería de datos como herramienta para encontrar patrones ocultos en grandes conjuntos de datos en los que el ser humano por sí mismo y por muy bien que estuviese formado no sería capaz de identificarlos (Goecks *et al.,* 2020).

La forma habitual que se tiene de presentar los datos a los algoritmos de aprendizaje automático son los datasets, matrices de datos en las que las filas de la matriz representan las instancias, es decir, los objetos de estudio que representan los ejemplos a partir de los cuales va a aprender el algoritmo, y las columnas representan cada una de las variables o cualidades que definen a cada una de las instancias, de modo que para cada instancia habrá una serie de datos correspondientes asociados a cada uno de los atributos, por tanto el aprendizaje se realiza mediante un modelo tipo inductivo de modo que, a partir de razonamiento hipotético de casos particulares de estudio, en este caso las instancias, se pretende generalizar para poder predecir y comprender futuros comportamientos del sistema que se está estudiando.

Durante el proceso de aprendizaje con los datasets se pueden distinguir dos etapas. La primera es la etapa de aprendizaje, en la cual el algoritmo aprende de los datos y genera los modelos correspondientes para predecir su comportamiento. La segunda

es la etapa de test o prueba, en la cual la máquina comprueba los modelos obtenidos durante la fase de aprendizaje y calcula la estimación del error. El problema es que el error tiende a ser muy variable dependiendo de qué datos de nuestro conjunto de datos hayamos seleccionado para el entrenamiento y el test. Por ello se emplea comúnmente una estrategia llamada validación cruzada con 10 particiones (Escoto *et al.,* 2020), una estrategia usada cuando se tiene un número pequeño de ejemplos, permitiéndonos no perder demasiados ejemplos en la fase de entrenamiento y obtener un buen estimador para el error. De esta forma tomaremos el conjunto de datos original y crearemos K-Folds (k=10) completamente aleatorios (Figura 1.1), con lo que dividiremos los ejemplos (número de ejemplos=n) en k conjuntos disjuntos (folds), de modo que con 9 de esas particiones se entrenará al algoritmo y con 1 se calculará la estimación del error, devolviéndonos el software la media de los errores de los K conjuntos o "folds". Además, es habitual emplear k=10 pero repetir el proceso varias veces para obtener una estimación aún más precisa del error.

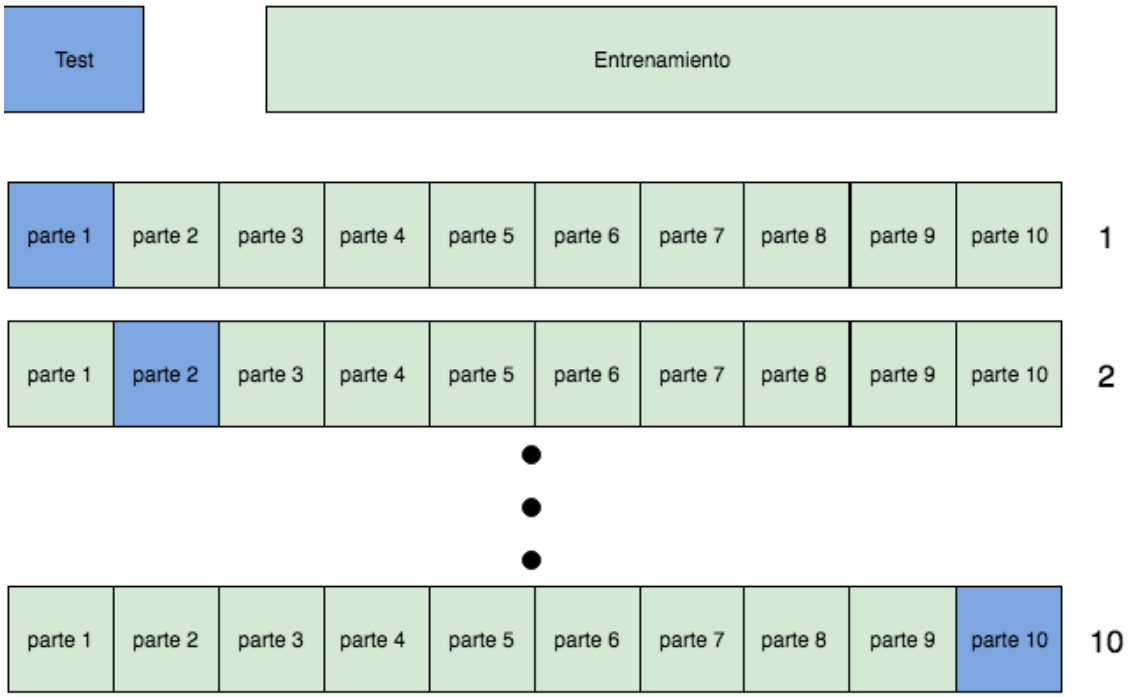

Figura 1.1. Representación del método de validación cruzada en K-Folds. En este caso el modelo se divide en 10 particiones, 9 de entrenamiento y 1 de test.

1.3.1. Aprendizaje automático supervisado y no supervisado

Dentro del Aprendizaje Automático se pueden diferenciar dos subcategorías o paradigmas clásicos en función de la metodología que se emplee para su análisis, aprendizaje automático no supervisado y aprendizaje automático supervisado (Deo, 2013).

En el aprendizaje automático no supervisado el análisis de los datasets y la generación de modelos está condicionada por la ausencia de etiquetas de salida en los datos (Schmidt *et al.*, 2019). El algoritmo carga el dataset en el sistema y lee los datos, de modo que cada una de las instancias tiene varios atributos, pero no tienen en ningún caso una etiqueta de salida, es decir, las instancias en la fase de entrenamiento no están asociadas a una etiqueta que las englobe dentro de una clase o categoría concreta, por lo que tendrá que ser el propio algoritmo el que, mediante metodologías y enfoques diferentes a los del aprendizaje supervisado como pueden ser las basadas en "clustering" o reglas de asociación, deberá generar grupos o clases hipotéticas hallando patrones o relaciones ocultas entre las diferentes instancias a partir de la información que pueda extraer de sus atributos (Chen *et al.,* 2020).

Gracias a su versatilidad y tasa de acierto se ha utilizado en multitud de campos y con un amplio abanico de aplicaciones como puede ser la recomendación de productos a consumidores (Batmaz *et al.*, 2018), vehículos de conducción autónoma (Stilgoe, 2017), o en el campo de la biomedicina tenemos ejemplos en el screening de cáncer de mama (McKinney et al., 2020) y cáncer de próstata (Cuocolo *et al.*, 2019), o el descubrimiento de nuevos fármacos (Stokes *et al.*, 2020).

En el aprendizaje automático supervisado en cambio los datos con los cuales se entrena a los algoritmos sí que tienen asociadas unas etiquetas de salida que pueden ser tanto numéricas como categóricas, por lo que el objetivo del algoritmo será predecir la clase a la que pertenecen las instancias del dataset en la fase de entrenamiento (Deo, 2013). Es por esto mismo por lo que el aprendizaje automático supervisado se emplea principalmente en labores de regresión y clasificación en las que el algoritmo genera un modelo que podrá ser usado para clasificar las futuras entradas en una clase o subgrupo concreto sin necesidad de intervención humana.

En este proyecto nos centraremos en aplicar algoritmos de clasificación (englobados dentro del aprendizaje supervisado) que buscan estimar una variable de salida

categórica. Dentro de los algoritmos de clasificación nos encontramos además con diferentes tipologías de algoritmos (más adecuados para uno u otro problema) como son las redes neuronales, los árboles de decisión, los sistemas basados en reglas, el método del vecino más cercano o las máquinas de soporte vectorial (SVM), entre otras muchas. Además, el aprendizaje automático supervisado tiene una característica que lo diferencia esencialmente del aprendizaje automático no supervisado, su similitud con el comportamiento humano y la interpretabilidad de los resultados generados.

Uno de los problemas del aprendizaje automático es que, al tratar con conjuntos de datos de una complejidad muy elevada, genera una serie de modelos y resultados de una muy baja interpretabilidad y diferentes a la forma de razonamiento humana (Abbasi y Goldenholz, 2019), dándose lo que se conoce comúnmente como modelos de cajas negras o “black boxes”. En estas situaciones tenemos unos sistemas con una entrada y una salida de datos, pero no sabemos realmente en que consiste el modelo que se ha generado ni cómo el algoritmo ha interpretado los datos para aprender de ellos (Azodi y Shiu, 2020), razón por la cual haya aplicaciones, como puedan ser la ayuda en la toma de decisiones para elegir un tratamiento con el que debe ser tratado un paciente, en las cuales no es aceptable elegir un algoritmo o sistema que deriva en una caja negra (Duch *et al.*, 2000).

A día de hoy, la mayoría de los algoritmos de aprendizaje supervisado no solo se centran en obtener resultados precisos sino también en resultados interpretables (buscando evitar el problema de la caja negra). Eso ha motivado que el aprendizaje automático se haya aplicado a una infinidad de labores de clasificación en biología como pueden ser la clasificación de enfermedades (Johnson *et al.,* 2018), en estudios sobre daño cardiaco (Awan *et al.*, 2018) o para predecir la enfermedad que sufre un paciente sin intervención de un médico especializado (Baştanlar y Özuysal, 2013).

1.4. Métodos de clasificación basados en reglas

En la actualidad existe un gran número de algoritmos de aprendizaje automático supervisado empleados en clasificación, englobados a su vez dentro de otros subgrupos dependiendo del funcionamiento, parámetros y reglas en las que se sustentas, desde los algoritmos más convencionales, como pueden ser los árboles de decisión clásicos, hasta algoritmos con planteamiento mucho más interdisciplinares como son los algoritmos evolutivos. En este proyecto se optó por la utilización de

algoritmos basados en reglas, algoritmos muy útiles y conocidos en el ámbito del aprendizaje automático debido a que son capaces de crear modelos fácilmente interpretables (algo imprescindible a día de hoy), debido fundamentalmente a que para clasificar los ejemplos estos algoritmos generan reglas que, en caso de cumplirse los condicionantes, le indican al algoritmo que debe clasificar el ejemplo en la clase que indique la regla, lo que se asemeja enormemente al razonamiento humano para la clasificación de objetos.

Hoy en día, con el auge del Internet de las Cosas o IoT (Evans, 2011) hay una incipiente necesidad de aumentar la explicabilidad de los resultados obtenidos. Por ejemplo, los coches automáticos o la medicina personalizada se basan precisamente en esta tecnología de *"*machine learning*"*, por lo que este tipo de técnicas son de gran utilidad para, entre otras cosas, resolver el problema de la caja negra (Castelvecchi, 2016; Rudin, 2019) gracias a que se obtienen modelos interpretables mediante los cuales, haciendo referencia al ejemplo anterior, podemos valorar si las reglas y premisas de las que parte el modelo son satisfactorias y compatibles con la ética y conocimiento humano.

A continuación, se incluirá una explicación sobre los fundamentos y funcionamiento de los distintos algoritmos que se utilizaran en la parte experimental de este proyecto.

1.4.1. ALGORITMO: C4.5

El árbol de decisión C4.5 es el sucesor del conocido ID3 (Quinlan, 1993), uno de los algoritmos más populares en *"*machine learning*"* y que ha sido citados en multitud de artículos desde que Quinlan lo diese a conocer por primera vez en 1986. Los árboles de decisión son un modelo de predicción muy conocido en el ámbito de la inteligencia artificial principalmente por su fácil comprensión (Chern *et al.*, 2019), ya que el concepto de árbol de decisión está íntimamente ligado al razonamiento humano.

Los árboles que genera el algoritmo tienen dos elementos principales, las hojas o nodos que serían cada uno de los puntos en los cuales el algoritmo tiene que tomar una decisión para seguir avanzando, y las flechas o direcciones en las cuales se bifurca el árbol dependiendo de las características de los datos de entrada, de tal modo que el algoritmo se encuentra en un primer nodo y dependiendo de los valores de entrada tomará la decisión de seguir hacia uno de los siguientes nodos hasta que acaba en el nodo final, que sería la clase a la que pertenece el objeto perteneciente a

la instancia de entrada. Tras la generación del árbol completo suele haber una fase de poda (Salzberg, 1993) en la que se eliminan nodos y ramas del árbol no útiles.

Una de las mayores ventajas de los árboles de decisión, además de su fácil interpretación y la posibilidad de implementar todo tipo de variables, es la posibilidad de visualizar fácilmente la estructura del árbol construido (Figura 1.2). Sin embargo, a veces no será posible recurrir a este tipo de algoritmos debido a problemas intrínsecos como su tendencia al sobreajuste (overfitting) o la posibilidad de crear árboles sesgados si una de las clases es mucho más numerosa que las otras.

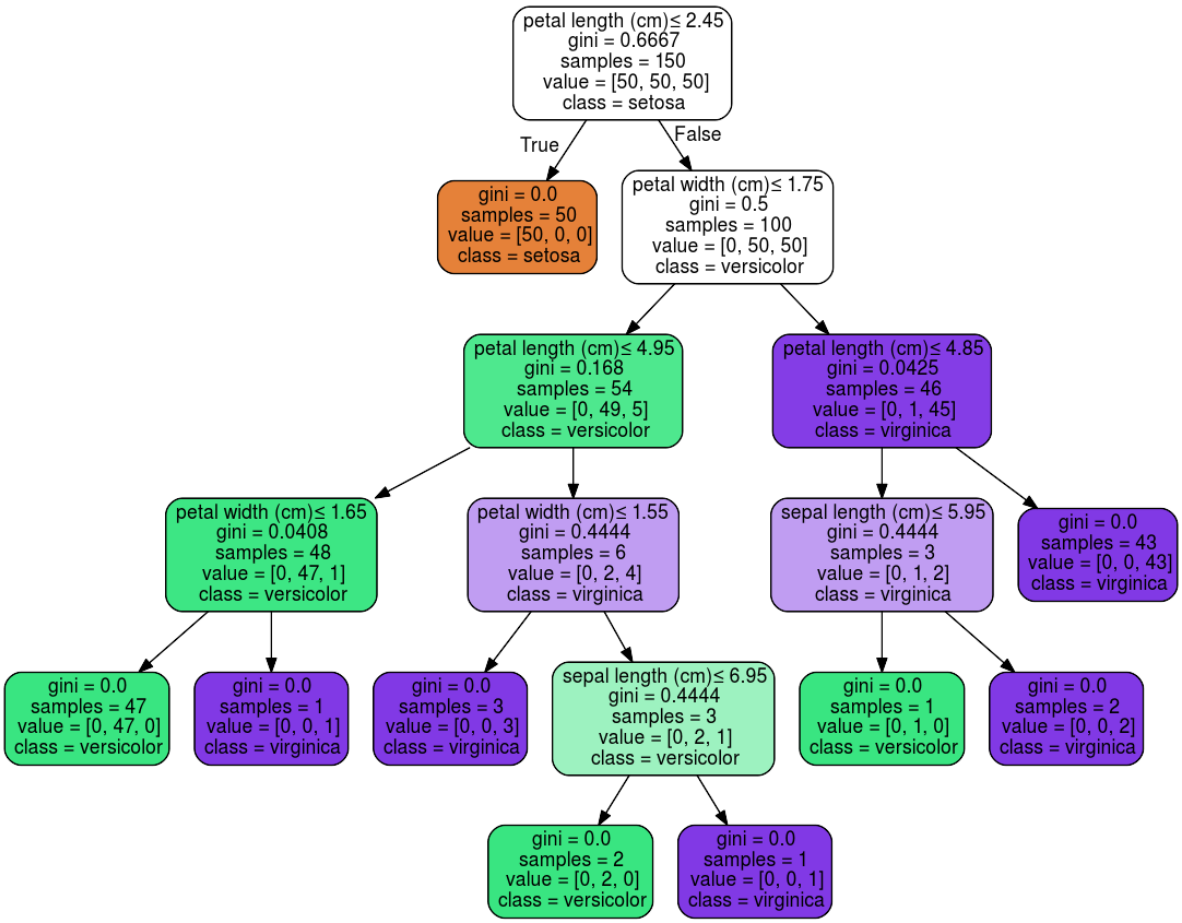

Figura 1.2. Árbol de decisión generado a partir del dataset "iris", empleando la librería SciKitLearn

Es por esto que a partir de los árboles de decisión surgieron nuevos tipos de algoritmos, como son las reglas de aprendizaje Crisp y Fuzzy. La principal diferencia estos dos tipos de algoritmos basados en reglas es que las reglas Crisp o convencionales se basan en lógica booleana, es decir en sentencias que son ciertas o falsas, de modo que las clases de salida son opciones binarias como por ejemplo la posibilidad de que un tumor fuese benigno o maligno, mientras que las reglas de

aprendizaje Fuzzy se basa en lógica de valores continuos o lógica difusa, por lo que no se mide si una afirmación es verdadera o falsa sino el grado en que esa misma afirmación es verdadera, lo que en lógica de datos se representaría como el valor de pertenencia de una instancia a una clase, un concepto que erróneamente se asocia a una función estadística y que es sustancialmente diferente ya que una función estadística nos indicaría las probabilidades de que un objeto pertenezca a una clase distinta, mientras que una función de pertenencia a una clase nos indica en que grado esa instancia que estamos estudiando pertenece a una clase concreta, pudiendo adoptar todos los valores entre el 0 y el 1, ambos inclusive (Hühn y Hullermeier, 2009).

Los algoritmos de tipo Fuzzy tienen una serie de ventajas frente a los algoritmos Crisp. Estos últimos, por la propia lógica en la que se basan, generan tomas de decisiones mucho más individualizadas y bruscas, dando lugar a clases mucho más definidas y separadas unas de otras, lo cual puede en muchas ocasiones ser poco intuitivo y no acercarse realmente a la realidad que están reflejando los datos. Este problema no está presente en los algoritmos de tipo Fuzzy, que generan reglas mucho más flexibles y que permiten una transición entre clases mucho más realista e intuitiva.

1.4.2. ALGORITMO: RIPPER

Los algoritmos del tipo Crisp más comúnmente empleados son RIPPER (Cohen, 1995) y PART (Frank y Witten, 1998). RIPPER es el que primero surgió como evolución a base de sucesivas modificaciones de un anterior algoritmo basado en reglas llamado IREP (Fürnkranz y Widmer, 1994). RIPPER surgió precisamente como respuesta al ya mencionado árbol de decisión C4.5 ya que este último, pese a tener unos tiempos de procesamiento significativamente mayores, superaba enormemente en las ratios de error a su antecesor IREP, que además era muy poco escalable y se veía seriamente afectado por los datos ruidosos o innecesarios, de modo que RIPPER se diseñó para obtener ratios de error iguales o inferiores a C4.5, intentando posicionarse como una alternativa competitiva (Cohen, 1993).

Además, algo que hay que tener en cuenta es que, con los parámetros predefinidos, para los datasets empleados en los primeros experimentos realizados RIPPER era mucho más eficiente y escalable frente a datos ruidosos que su competidor C4.5, lo que en aquella época lo hizo bastante popular.

1.4.3. ALGORITMO: PART

Cuando RIPPER se estableció como uno de los algoritmos predominantes junto a su competidor apareció PART. Este nuevo algoritmo surgió como un intento de aunar las propiedades tanto de C4.5 como de RIPPER, basándose parcialmente en un árbol de decisión, pero funcionando con reglas de inducción Crisp. PART intentaría evitar así los respectivos problemas de sus predecesores, entre los cuales destacaba la necesidad de realizar una optimización global para obtener reglas que proporcionan resultados precisos (Frank y Witten, 1998).

La estrategia que realiza PART es la de separar y concretas normas, de manera que el algoritmo construye una regla, elimina las instancias a las que pertenece engloba dicha regla y continúa creando nuevas normas de manera recursiva hasta que ya no quedan instancias por eliminar. Esta metodología genera árboles de decisión parciales en los que se realiza una poda temprana de las ramas que dan lugar a subárboles muy precisos, evitando varios problemas de sus antecesores como son la generalización temprana ya que las instancias clasificadas, al ser eliminadas de la función, no intervienen en el proceso de aprendizaje.

Los resultados de este algoritmo en los primeros experimentos fueron ciertamente sorprendentes en aquella época. La principal ventaja que destacaron sus desarrolladores era la simplicidad de los árboles generados sin sacrificar precisión en la clasificación de los objetos estudiados, llegando a ser en sus inicios más preciso que RIPPER.

1.4.4. ALGORITMO: Chi-RW

De manera simultánea fueron surgiendo sus principales competidores del grupo de los algoritmos basados en reglas de aprendizaje Fuzzy. El primero de ellos en surgir fue el conocido Chi-RW (Chi *et al.*, 1996). La principal diferencia con los demás sistemas basados en reglas es que la utilización de reglas de peso como es el caso de Chi-RW demostró tener en su momento tener un impacto significativo en el rendimiento de los procesos de clasificación de estos sistemas (Ishibuchi y Yamamoto, 2005). Entre las múltiples aplicaciones que tiene Chi-RW destacan especialmente la segmentación de mapas de imágenes, el reconocimiento de letras mayúsculas en textos escritos en inglés y el reconocimiento de números escritos a mano (Chi *et al.*, 1996).

1.4.5. ALGORITMO: FURIA

Pese a sus bondades, Chi-RW fue rápidamente superado en muchos aspectos por su sucesor FURIA (Hühn y Hullermeier, 2009). Su nombre se deriva de las siglas en inglés de “Fuzzy Unordered Rule Induction Algorithm”. Algo curioso de este algoritmo es que es una extensión y remodelación del ya mencionado algoritmo RIPPER, siendo diseñado en un principio con el objetivo de conservar ciertas ventajas como son su simpleza y la interpretabilidad de los grupos de reglas generados, pero optando por reglas de aprendizaje Fuzzy en lugar de las reglas convencionales y crea reglas de forma desordenada en lugar de forma ordenada como en el caso de RIPPER, modificando además los sistemas de poda del árbol que utiliza su predecesor. Además, para el tratamiento de las instancias usadas durante la fase de entrenamiento el algoritmo utiliza un método eficiente para amoldar las reglas, haciéndolas más versátiles y adaptables a más conjuntos de datos (Hühn y Hullermeier, 2009).

Las reglas de aprendizaje convencionales se pueden interpretar como una lista de decisiones. Las primeras clases que se generan son las más pequeñas, es decir la que en términos de frecuencia recoge los ejemplos que se encuentras más raramente, de modo que los grupos siguientes van creciendo hasta que hay una clase que es la más abundante y la que, nuevamente en termino de frecuencia de ocurrencia, engloba a más ejemplos, por lo que a la hora de tomar decisiones esta última clase será la que tenga preferencia sobre las demás a ojos del algoritmo, que interpreta que como es la clase que más ejemplos engloba, cuando aparezca una nueva instancia será más probable que se englobe dentro de esta clase, la clase por defecto. Esto no debería ser necesariamente malo ya que, al fin y al cabo, en términos de aprendizaje es muy similar a la lógica que seguimos las personas para englobar objetos en grupos, sin embargo esto trae consigo una serie de perjuicios para el proceso de clasificación de futuras instancias ya que las reglas están ordenadas de manera intrínseca por prioridad y por lo tanto no resulta del todo objetivo, pudiendo conllevar a un sobreajuste (overfitting) de los datos, no generalizando bien para futuras instancias (Hühn y Hullermeier, 2009). Este es el principal motivo por el cual FURIA decide generar un conjunto de reglas no ordenadas en la que se da una situación en la que cada clase se enfrenta contra el resto de igual forma, sin orden o jerarquía interna, por lo que no hay ninguna clase que sea la clase por defecto.

Este planteamiento tiene principalmente dos problemas. Por un lado, podría darse una situación en la que una instancia pueda ser cubierta igualmente por dos clases diferentes, generando un conflicto para el algoritmo. Sin embargo, a nivel experimental se ha visto que este problema ocurre con una frecuencia muy baja y que es fácil de solventar. Por otro lado, al generar sets de normas sin un orden o jerarquía puede ocurrir que el modelo que se genera no esté necesariamente completo, pudiéndose dar situaciones en las que una nueva instancia no se puede clasificar en ninguna de las clases existentes.

Este problema se vio ya en sus inicios y por eso sus autores implementaron un método para el ajuste de reglas basados a su vez en el modelo propuesto en el año 2001 por Martin Eineborg y Henrik Boström. Este método se basa en la sustitución de todas las reglas por la generalización mínima dada por la instancia, un término que hace referencia al proceso por lo que se van eliminando todos los antecedentes de una regla en los cuales no puede encuadrarse la instancia problema. Una vez que tenemos las generalizaciones mínimas, se reevalúan y se clasifican en función de la precisión de Laplace en la etapa de entrenamiento (Eineborg y Boström, 2001). Este método proporcionaba buenos resultados, pero es de una gran complejidad computacional. Para solventar esto, FURIA realiza una serie de modificaciones de este método al aprovechar las instancias que ya han sido aprendidas y ordenarlos en forma de lista en vez de como matriz de datos, lo que a nivel computacional es significativamente más eficiente que la original.

1.4.6. ALGORITMO: GAssist-ADI

A pesar de ello, se siguieron buscando nuevas formas de optimizar el proceso de aprendizaje, ya que los resultados eran mejorables sobre todo cuando nos encontrábamos frente a espacios de búsqueda extensos y no lineales. De esta forma, paralela al desarrollo de nuevos algoritmos basados en lógica clásica y difusa, surgieron otro tipo de algoritmos con un enfoque distinto a lo que se había visto en computación clásica y que pretendía precisamente unir dos ramas de conocimiento como son la informática y la biología, los algoritmos evolutivos (Holland, 1975). Este tipo de algoritmos imitan los sistemas en los que se fundamenta la selección natural con el objetivo de optimizar los procesos de aprendizaje. Para ello los algoritmos evolutivos generan diferentes modelos y reglas y los tratan como si fuesen individuos

de una población, de tal modo que cada uno de los individuos representaría una posible solución frente al problema dado, en este caso la clasificación. Los individuos, al igual que en el medio natural, sufren cambios y son sometidos a un proceso de selección en el que los individuos (soluciones) que no sean aptos serán descartados, perdurando en el tiempo solo aquellos que den un porcentaje de clasificación óptimo. Este proceso se va repitiendo de forma cíclica y a cada ciclo se le llama generación, por lo que transcurridas una serie de generaciones se espera que solo queden los individuos más aptos, es decir, las mejores soluciones al problema de clasificación.

El más conocido de estos algoritmos es GAssit-ADI (Bacardi y Garrel, 2003), un algoritmo genético derivado de su predecesor GAssist-Intervalar, predecesor a su vez del original GABIL (De Jong *et al.*, 1993). Los algoritmos genéticos son un subgrupo de algoritmos evolutivos que transforman soluciones mediante operadores matemáticos, destacando la recombinación genética o cruce, es decir el intercambio parcial de características de unos modelos a otros.

GAssit-ADI se caracteriza porque, a pesar de basarse en la estructura de normas de GABIL, implementa una serie de reglas para la discretización adaptativa de intervalos o “Adaptive Discretization Intervals” (ADI) y se basa en el conocido como enfoque de Pittsburgh. En la representación de las normas de GABIL, cada situación problema o instancia se representa como un conjunto de valores que toman los atributos y es codificada como una cadena estática de valores binarios. La ventaja que presenta el sistema ADI es que la representación de los intervalos no es estática, produciendo sucesivos ciclos de escisión y fusión de unos operadores por otros a lo largo de las generaciones dando lugar a la ya mencionada recombinación genética (Bacardi y Garrel, 2003).

En el artículo original, los desarrolladores de este algoritmo ya sugirieron una serie de modificaciones para el sistema ADI. Los investigadores vieron que la probabilidad de recombinación necesitaba un ajuste específico para cada dominio de reglas, lo que no toma en cuenta el número de atributos de las instancias y que deriva en que un problema con el doble de atributos necesitará también duplicar la probabilidad de recombinación, para lo cual los investigadores propusieron definir la probabilidad de cada regla en función de cada atributo, cuyo valor se vio experimentalmente que era 0,05 (Bacardi y Garrel, 2003).

1.4.7. ALGORITMO: FARCHD

A medida que se fueron haciendo más populares los algoritmos basados en lógica difusa y que el Big Data iba cobrando forma en la sociedad, se vio que los sistemas de clasificación basados en lógica difusa para el aprendizaje inductivo sufrían un aumento exponencial del espacio de búsqueda cuando el número de atributos de las instancias era alto o cuando los patrones encontrados eran muchos, lo que supone no solo un perjuicio en el proceso de aprendizaje sino también en la escalabilidad y complejidad del modelo. Es este el motivo por el cual surgió otro tipo de algoritmo basado en lógica difusa, pero intentando aprovechar las ventajas y escalabilidad de los algoritmos evolutivos, el algoritmo FARCHD derivado del inglés “Fuzzy Association Rule-based Classification method for High-Dimensional problems” (Alcalá-Fdez et al., 2011).

Como su propio nombre indica, este es un algoritmo basado en reglas de asociación para problemas en los que los datos tengan una alta dimensionalidad, es decir las instancias tengan un gran número de atributos, obteniendo una buena precisión y buenos conjuntos de clasificadores basados en reglas con un coste computacional bajo. El método que emplea se puede dividir en 3 etapas:

- Extracción de reglas de asociación para la clasificación: Empleando un árbol de búsqueda para encontrar todos los conjuntos de elementos difusos posibles y generar las reglas de asociación.
- Selección de las reglas candidatas: El número de reglas generadas en la primera fase puede ser muy alto y el coste computacional puede ser muy elevado, por lo que se utilizan los subgrupos descubiertos en la fase anterior y una medida de precisión relativa ponderada (wWRAcc’) para preseleccionar las reglas más interesantes.
- Selección de reglas genéticas: Se emplea un algoritmo genético para seleccionar y adaptar un conjunto de reglas difusas de muy alta precisión en la clasificación para conseguir una sinergia positiva entre la lógica difusa y el algoritmo genético propiamente dicho, obteniendo así un modelo mucho más escalable y con un coste computacional mucho más bajo.

1.4.8. ALGORITMO: BioHEL

Con el paso del tiempo la conjugación entre las ciencias biológicas y las computacionales fue siendo cada vez mayor, surgiendo algoritmos expresamente diseñados para el aprendizaje automático de datos biológicos aprovechando los enfoques ya existentes. Así es como surgió BioHEL (Bioinformatics-oriented hierarchical evolutionary learning), un algoritmo evolutivo diseñado específicamente para la minería de datos biológicos y con el objetivo de mejorar la escalabilidad de GAssist y manejar conjuntos de datos a gran escala (Bacardit et al., 2008).

Este algoritmo aplica reglas de aprendizaje iterativo (Venturini, 1993), lo que permitió a sus desarrolladores obtener unos resultados de clasificación muy competentes frente a conjuntos de datos muy extensos del ámbito de la Biología. Estos conjuntos de datos, por su propia naturaleza, suelen tener la peculiaridad de tener muchos datos ruidosos y un gran número de atributos que no son relevantes y que lo único que hacen es entorpecer el aprendizaje y aumentar el costo computacional. Para solucionar esto BioHEL trabaja con una lista de unos cuantos dominios de atributos, en lugar del conjunto completo, y dos operadores se van encargando de añadir o eliminar en función de una probabilidad dada, a lo que hay que añadir un proceso de recombinación entre las listas propio de los algoritmos evolutivos. Todo esto permite que los operadores de coincidencia solo evalúen los subconjuntos de atributos, evitando así los cálculos irrelevantes y el coste computacional que ello conlleva. Además, como solo se emplean listas de atributos que sí son influyentes los operadores siempre recombinarán datos que sí son relevantes, lo que puede conducir a un mejoramiento del proceso de aprendizaje y de la interpretabilidad de los datos (Bacardit et al., 2008).

En el artículo original los investigadores comprobaron que emplear únicamente los atributos clave para el proceso de aprendizaje aumenta sustancialmente la eficiencia del mismo, mostrando no solo que los resultados a la hora de eficacia de clasificación eran muy competentes con respecto a sus predecesores, sino también una reducción significativa del tiempo de ejecución, haciendo al sistema hasta 2 y 3 veces más rápido (Bacardit *et al.*, 2008).

2. OBJETIVOS

2.1. Objetivos generales

El principal objetivo de este proyecto es el ahondar en las técnicas de minería de datos en la que se emplea la inteligencia artificial para analizar y resolver problemas de clasificación con datos relacionados con la biología, ampliando el conocimiento sobre biología computacional y explotando los beneficios que presenta un campo de conocimiento de tanta relevancia en la actualidad como es el aprendizaje automático, para lo cual se emplearan algoritmos computacionales propios de este campo sobre datasets de datos relacionados con el mundo de la biología.

2.2. Objetivos específicos

Los objetivos específicos marcados para este proyecto son:

- Recopilación y adaptación de un conjunto de datasets relacionados con la biología para poder ser utilizados.
- Aplicación de diferentes algoritmos computacionales de aprendizaje automático para extraer información de los datasets utilizando el programa KEEL (Triguero *et al..,* 2017).
- Extracción y análisis de los resultados obtenidos tras el experimento, ayudándonos de diferentes herramientas y test estadísticos para evaluar el comportamiento de los diferentes algoritmos frente a diferentes situaciones, pudiendo servir de ayuda para futuras investigaciones en las que se deba elegir emplear un algoritmo concreto.

3. MATERIAL Y MÉTODOS

3.1. Conjuntos de datos

El primer paso en la realización de este proyecto fue la selección de los datasets que se iban a emplear durante el experimento. Los conjuntos de datos utilizados para este proyecto fueron obtenidos de los repositorios de “Knowledge Extraction based on Evolutionary Learning” (KEEL) (Alcalá-Fdez *et al.*, 2011), “Irvine machine learning Repository” (UCI) (Dua y Graff, 2019), Kaggle y Weka. Todas ellas son plataformas

encargadas de almacenar datasets de muy diversa índole específicamente para experimentos de aprendizaje supervisado.

Sus repositorios son comúnmente usados en numerosos proyectos de investigación y publicaciones científicas debido a su calidad, además de estar dispuestos en multitud de formatos, pudiendo seleccionar específicamente aquellos datasets destinados a experimentos de clasificación o regresión, lo que ha llevado a estas plataformas a ser avaladas y respaldadas por una gran parte de la comunidad de investigadores que se dedica al estudio del aprendizaje supervisado, por lo que el haber elegido estas plataformas nos permite obtener unos resultados mucho más extrapolables y comparables con los de otras investigaciones similares, pues muchos de los datasets empleados en nuestro proyecto ya han sido utilizados previamente.

Para poder ser empleados en el experimento, los datasets fueron seleccionados ateniendo a una serie de características relacionadas principalmente con las limitaciones de hardware:

- Número de ejemplos o instancias: Se busco un número de instancias razonable, con un tamaño mínimo de 90 ejemplos (post-operative) para que el aprendizaje fuese posible y un tamaño máximo de 5300 (banana) para que el tiempo de procesamiento fuese razonable.
- Número de clases: Se han utilizado datasets con un tamaño mínimo de 2 clases hasta un tamaño máximo de 28 clases para comprobar cómo afecta el número de clases a la eficacia de los algoritmos.
- Número de atributos: El número de atributos de los datasets va desde 2 (banana) hasta 34 (dermatology).
- Naturaleza de los datos: Se busco que tuviesen la mayor heterogeneidad posible y así estudiar cómo afectan las diferentes variables al comportamiento y eficacia de los algoritmos. Se seleccionaron algunos datasets artificiales para ver su comportamiento con respecto a los reales y datasets con valores sin especificar (missing values) para ver como los procesaba el software.

Teniendo en cuenta todas estas características, se emplearon un total de 30 datasets cuyos nombres y características están plasmadas en la Tabla 3.1.

Tabla 3.1. Datasets seleccionados para el experimento junto a las diferentes características de los mismos. La columna MV hace referencia a los datos no especificados, representándose con * la ausencia de los mismos en el dataset.

Dataset	Instancias	Atributos	Clases	MV	Origen
abalone	4174	8	28	*	Abulones
amino_acid_composition	698	20	27	*	Proteínas
appendicitis	106	7	2	*	Apendicitis
banana	5300	2	2	*	Bananas
breast	286	9	2	9	Cáncer de mama
bupa	345	6	2	*	Hígado
cleveland	303	13	5	6	Corazón
contraceptive	1473	9	3	*	Anticonceptivos
dermatology	366	34	6	8	Dermatosis
ecoli	336	7	8	*	Bacterias
flare	1066	11	6	*	Sol
hayes-roth	160	4	3	*	Artificial
heart	270	13	2	*	Corazón
hepatitis	155	19	2	75	Hepatitis
hidrophobicity	698	21	27	*	Proteínas
ionosphere	351	33	2	*	Ionosfera
iris	150	4	3	*	Plantas (iris)
lymphography	148	18	4	*	Linfografía
mammographic	961	5	2	131	Mamografía
monk-2	432	6	2	*	Artificial
pima	768	8	2	*	Diabetes
polarity	698	21	27	*	Proteínas
post-operative	90	8	3	3	Postoperatorio
saheart	462	9	2	*	Corazón
van_der_waals	698	21	27	*	Proteínas
wdbc	569	30	2	*	Cáncer de mama
wine	178	13	3	*	Vino
wisconsin	699	9	2	16	Cáncer de mama
yeast	1484	8	10	*	Levaduras
zoo	101	16	7	*	Animales

3.1.1. Formato de los datasets

A la hora de introducir los conjuntos de datos en KEEL deberemos hacerlo a través de la pestaña “Data Management” de la interfaz del programa. Debido a la naturaleza tan heterogénea de los datos, estos deben de ser editados y adaptados para poder ser interpretados por el programa.

El formato de KEEL (Figura 3.1) está compuesto por la primera línea en la cual se detallará el nombre del dataset y a continuación los atributos con sus respectivos nombres, en cada uno de los cuales se detalla el valor máximo y mínimo que toman las instancias dentro del conjunto de datos. El último de los atributos hará referencia a las clases en las cuales pueden ser clasificadas las instancias, ya que al encontrarnos en un experimento de aprendizaje automático supervisado todas las instancias tiene que llevar asociadas una etiqueta de salida. Después tendremos la línea de inputs que nos indica qué atributos son las cualidades propiamente dichas de las instancias y la línea de outputs que indica al programa qué atributo hace referencia a las clases. Por último, nos encontraremos las instancias en las que cada línea corresponde a un ejemplo con los valores de los atributos colocados en el orden del propio documento.

```
@relation yeast
@attribute Mcg real [0.11, 1.0]
@attribute Gvh real [0.13, 1.0]
@attribute Alm real [0.21, 1.0]
@attribute Mit real [0.0, 1.0]
@attribute Erl real [0.5, 1.0]
@attribute Pox real [0.0, 0.83]
@attribute Vac real [0.0, 0.73]
@attribute Nuc real [0.0, 1.0]
@attribute Class {MIT, NUC, CYT, ME1, ME2, ME3, EXC, VAC, POX, ERL}
@inputs Mcg, Gvh, Alm, Mit, Erl, Pox, Vac, Nuc
@outputs Class
@data
0.58, 0.61, 0.47, 0.13, 0.5, 0.0, 0.48, 0.22, MIT
0.43, 0.67, 0.48, 0.27, 0.5, 0.0, 0.53, 0.22, MIT
0.64, 0.62, 0.49, 0.15, 0.5, 0.0, 0.53, 0.22, MIT
0.58, 0.44, 0.57, 0.13, 0.5, 0.0, 0.54, 0.22, NUC
0.42, 0.44, 0.48, 0.54, 0.5, 0.0, 0.48, 0.22, MIT
0.51, 0.4, 0.56, 0.17, 0.5, 0.5, 0.49, 0.22, CYT
0.5, 0.54, 0.48, 0.65, 0.5, 0.0, 0.53, 0.22, MIT
```

Figura 3.1. Ejemplo de dataset “yeast”. Se puede observar cómo los ejemplos tienen un total de 8 atributos y se pueden clasificar dentro de un total de 10 clases diferentes.

3.2. Diseño del experimento con KEEL

El software KEEL dispone de una interfaz intuitiva en la pestaña “Experimentos” que nos permite organizar nuestro proyecto de forma sencilla atendiendo a un criterio basado en un esquema de flujo de trabajo. El experimento realizado en este proyecto (Figura 3.2) empieza con los datos, todos ellos introducidos por el software como un único bloque de trabajo, pues todos ellos serán tratados de forma conjunta de igual forma a lo largo del experimento.

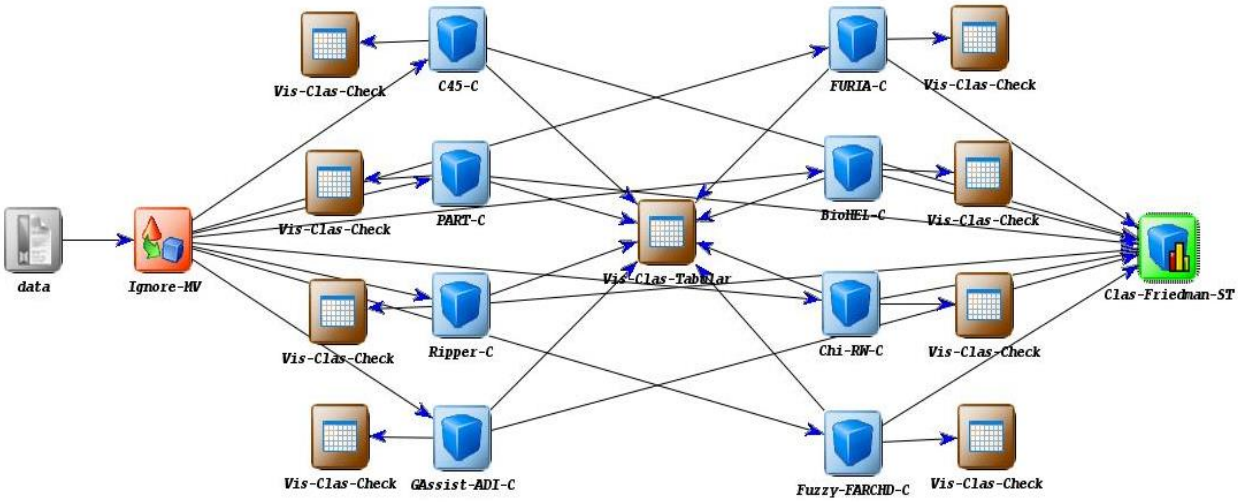

Figura 3.2. Flujo de trabajo del experimento realizado en KEEL.

El primer proceso que hará el software será pasar todos los conjuntos de datos por un algoritmo llamado Ignore-MV (Gourraud *et al.*, 2004) que filtrará los datos. Este algoritmo lo que hará será eliminar, dentro de cada dataset, las instancias que presenten “missing values” o valores perdidos para que no influyan negativamente dentro del proceso de aprendizaje durante el resto del experimento.

Una vez los datasets hayan sido filtrados y editados, todos los datos serán empleados en el proceso de aprendizaje automático por los diferentes algoritmos: C4.5 (Quinlan, 1993), RIPPER (Cohen, 1995), PART (Frank y Witten, 1998), Chi-RW (Chi *et al.*, 1996), FURIA (Hühn y Hullermeier, 2009), GAssit-ADI (Bacardi y Garrel, 2003), FARCHD (Alcalá-Fdez *et al.*, 2011) y BioHEL (Bacardit *et al.*, 2008). Estos algoritmos son los que realmente se encargarán de realizar el proceso de aprendizaje automático y construir los algoritmos basados en reglas que se encargarán de clasificar los diferentes ejemplos en las diferentes clases.

Una vez que se hayan construido los modelos de aprendizaje el propio software de KEEL se encargará de testear la eficacia de clasificación y en general la calidad del algoritmo construido. Tras esta fase de testeo, todos los resultados obtenidos por cada algoritmo en particular podrán ser visualizados a través al método Vis-Class-Check. Para obtener una visión más general de los resultados emplearemos el método Vis-Class-Tabular, que nos permitirá comparar de forma más rápida el rendimiento de unos algoritmos con otros. De manera adicional se ha incorporado a este experimento un método llamado Vis-Friedman-ST que nos permitirá visualizar un ranking sencillo en el que se ordenarán nuestros algoritmos de mejor a peor rendimiento.

3.3. Parámetros de los algoritmos

En el experimento todos los algoritmos han sido utilizados empleando los parámetros por defectos que venían implementados en KEEL y que, por lo tanto, especificaron sus investigadores en sus respectivos artículos.

Debido a la gran heterogeneidad de los datos, modificar los parámetros para que estos se adaptasen a todos los datasets de la forma más optima posible sería inabarcable para este proyecto, por lo que se mantienen los parámetros estándar que determinaron sus investigadores que eran los más óptimos con el objetivo de que los resultados de este experimento sean lo más extrapolables para otros investigadores que quieran utilizarlos.

3.4. Metodología de análisis

El principal problema a la hora de analizar los resultados de nuestro experimento era el sesgo producido al segmentar los datos implementados en los subconjuntos de entrenamiento y test. Para evitar el sobreajuste de los modelos a los datos de entrenamiento y obtener resultados poco generalizables se utilizó la validación cruzada de K-Folds (Yang y Huang, 2014). Mediante este método, a la hora de particionar los datasets, cada conjunto de datos se divide en K particiones de forma aleatorizada, siendo en nuestro experimento K=10, 9 de las particiones se utilizan en la fase entrenamiento para construir el modelo y 1 se emplea para testear el rendimiento del mismo, de modo que todas las particiones se emplean 9 veces para la fase de entrenamiento y 1 para test. De este modo, los resultados obtenidos no están influenciados por la forma en la que se particionan los datos.

Tras esta fase de testeo, todos los resultados obtenidos por cada algoritmo en particular podrán ser visualizados gracias al algoritmo StatCheckCL (Martínez y Rodríguez, 1993). Gracias a este método nos será posible realizar un análisis estadístico del desempeño que ha tenido cada uno de los algoritmos de clasificación a través de diferentes medidas estadísticas para calcular el error de clasificación, pudiendo analizar más detenidamente el rendimiento de casos particulares de datasets de forma más precisa.

Por otro lado, para obtener una visión más general de los resultados emplearemos el algoritmo StatTabularCL (Martínez y Rodríguez, 1993). Este está asociado a los resultados durante la fase de entrenamiento y testeo de todos los algoritmos empleados para el aprendizaje, de modo que se determina por un lado el porcentaje de ejemplos correctamente clasificados y por otro, mediante la implementación de la prueba W (Royston, 1982), se contrasta si las distribuciones de error son normales para las muestras a comparar: si las distribuciones son normales y tienen la misma varianza se ejecuta el test T, si las distribuciones son normales pero tienen diferentes varianzas se ejecuta el test TVAR, y si las distribuciones no son normales se ejecuta el test de Wilcoxon. Estas pruebas estadísticas se ejecutan con el fin de comparar las diferentes distribuciones de los métodos de clasificación y determinar cuál es mejor.

De manera adicional se ha incorporado a este experimento un algoritmo para la ejecución del test de Friedman (Zar, 1999), una prueba no paramétrica similar a la prueba de medias de ANOVA. Este test calcula la clasificación de los resultados observados por algoritmo para cada conjunto de datos, asignando al mejor de ellos la clasificación 1 y al resto una clasificación k, por lo que mediante esta prueba podremos extraer un ranking sencillo en el que se ordenarán nuestros algoritmos de mejor a peor rendimiento. La fiabilidad de este ranking se determinará mediante el test de Holm, en el cual si el valor supera el 0,05 se acepta la hipótesis de igualdad (Gacto *et al.,* 2019).

4. RESULTADOS

En este apartado se incluirán los resultados obtenidos en este trabajo. Una vez se haya realizado el experimento en KEEL, mediante el método Vis-Clas-Tabular podremos extraer datos como las tasas promedio de instancias correctamente clasificadas en la fase de test (Tabla 4.1), así como otros datos secundarios como los resultados en la fase de entrenamiento o las varianzas en ambas partes:

Tabla 4.1. Resultados de la tasa de ejemplos correctamente clasificados durante la fase de test para los diferentes algoritmos. Los resultados están comprendidos entre 0 y 1 (mejor rendimiento). Se resalta en negrita el mayor valor de rendimiento para cada dataset. También se incluye la media de funcionamiento del algoritmo en todos los datasets.

DATASET	C4.5	PART	RIPPER	GAssist	FARCHD	Chi-RW	BioHEL	FURIA
abalone	0,204	0,165	0,234	**0,248**	0,173	0,000	0,225	0,207
amino_acid	0,324	0,122	0,264	0,226	0,345	0,263	**0,363**	0,345
appendicitis	0,833	0,823	0,805	0,860	0,833	0,861	0,793	**0,879**
banana	**0,891**	0,591	0,622	0,858	0,855	0,602	0,888	0,880
breast	**0,769**	0,695	0,610	0,748	0,734	0,669	0,691	0,755
bupa	**0,670**	0,590	0,624	0,626	0,669	0,579	0,661	**0,670**
cleveland	0,525	0,539	0,405	0,552	0,552	0,380	0,526	**0,562**
contraceptive	0,527	0,429	0,525	**0,545**	0,530	0,399	**0,545**	0,544
dermatology	0,943	0,731	0,922	0,941	0,913	0,223	0,950	**0,955**
ecoli	0,795	0,447	0,727	0,768	0,798	0,720	0,747	**0,807**
flare	0,743	0,311	0,679	0,738	0,740	0,385	0,743	**0,748**
hayes-roth	0,800	0,419	**0,844**	0,725	0,744	0,588	0,806	0,806
heart	**0,781**	0,578	0,722	0,778	0,833	0,519	**0,781**	0,826
hepatitis	0,840	0,447	0,881	**0,904**	0,885	0,244	0,846	0,822
hidrophobicity	0,284	0,435	**0,929**	0,214	0,120	0,107	0,020	0,002
ionosphere	0,909	0,792	0,903	0,918	**0,923**	0,655	0,892	0,918
iris	**0,960**	0,333	0,953	**0,960**	0,953	0,927	0,940	0,947
lymphography	0,743	0,651	0,744	**0,799**	0,748	0,123	0,780	0,794
mammographic	0,830	0,789	0,773	0,828	**0,840**	0,808	0,798	0,834
monks	**1,000**	0,527	**1,000**	0,989	0,995	0,429	**1,000**	**1,000**
pima	0,742	0,651	0,707	0,729	**0,764**	0,731	0,691	0,755
polarity	**0,294**	0,116	0,233	0,203	0,114	0,110	0,243	0,241
post-operative	0,689	**0,690**	0,425	0,583	0,549	0,232	0,635	0,038
saheart	0,684	0,649	0,611	0,671	**0,688**	0,727	0,007	0,002
van_der_waals	**0,288**	0,145	0,206	0,213	0,140	0,090	0,026	0,002
wdbc	0,945	0,887	0,926	0,951	**0,953**	0,921	0,018	0,001
wine	0,949	0,640	0,927	0,926	0,942	0,938	0,909	**0,966**
wisconsin	0,956	0,698	0,956	0,962	**0,969**	0,913	0,946	0,962
yeast	0,555	0,322	0,504	0,533	0,588	0,292	**0,592**	0,581
zoo	0,928	0,861	0,931	0,927	**0,962**	0,651	0,958	0,945
MEDIA	**0,713**	0,536	0,686	0,697	0,695	0,503	0,634	0,626

Los datos de los resultados de la fase de entrenamiento los utilizaremos para enfrentarlos con los de la fase de test, comparando el promedio de tasa de acierto de clasificación de los algoritmos en ambas fases (Tabla 4.2) para poder hacernos una idea de la capacidad de generalización de nuestros modelos.

Tabla 4.2. Promedio de las tasas de ejemplos correctamente clasificados por los 8 algoritmos en la fase de entrenamiento y prueba para cada dataset.

DATASET	TOTAL	
	TRA	TST
abalone	0,284	0,182
amino_acid_composition	0,563	0,282
appendicitis	0,930	0,836
banana	0,784	0,773
breast	0,843	0,709
bupa	0,787	0,636
cleveland	0,759	0,505
contraceptive	0,593	0,505
dermatology	0,959	0,822
ecoli	0,829	0,726
flare	0,676	0,636
hayes-roth	0,806	0,716
heart	0,889	0,727
hepatitis	0,971	0,778
hidrophobicity	0,453	0,138
ionosphere	0,961	0,864
iris	0,898	0,872
lymphography	0,933	0,673
mammographic	0,843	0,813
monks	0,937	0,867
pima	0,810	0,721
polarity	0,447	0,194
post-operative	0,815	0,480
saheart	0,799	0,505
van_der_waals	0,445	0,139
wdbc	0,974	0,700
wine	0,952	0,900
wisconsin	0,950	0,920
yeast	0,588	0,496
zoo	0,973	0,895
MEDIA	0,782	0,634

Dado que uno de los objetivos principales del proyecto es poder contribuir a determinar qué algoritmo es mejor escoger dependiendo de las características de nuestros conjuntos de datos, será necesario evaluar el rendimiento de los algoritmos empleados dependiendo de las características en las que varían los datasets.

En primer lugar, lo que haremos será ordenar los datasets en función del número de instancias que estos posean y se generará una gráfica que nos permitirá ver cómo afecta este factor al rendimiento. De este modo en la primera gráfica que generaremos (Figura 4.1) lo que tendremos será en el eje Y la tasa de ejemplos correctamente clasificados, mientras que en el eje X los datasets ordenados de menor a mayor número de instancias, por lo que el número 1 correspondería al dataset "post-operative" que tiene un total de 90 ejemplos, mientras que el número 30 corresponderá al dataset "banana" con 5300 ejemplos.

Se decidió visualizar las gráficas en escala logarítmica para facilitar la visualización de las mismas y hacer más evidentes los cambios en los resultados de tasas de ejemplos correctamente clasificados.

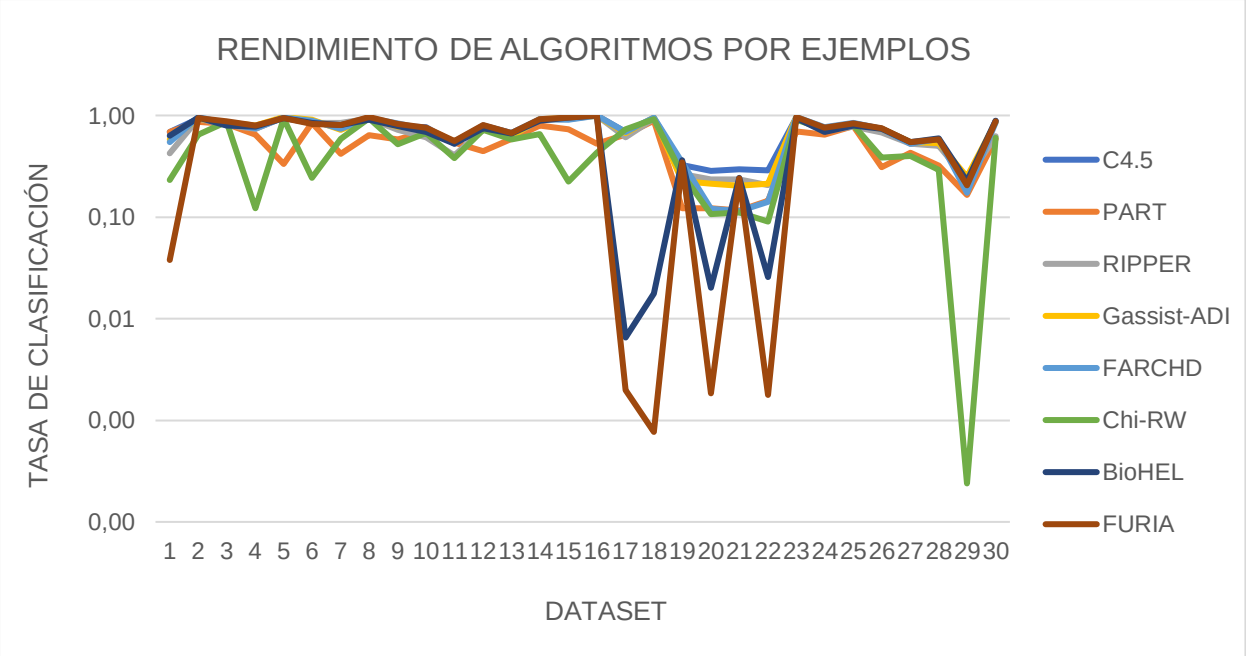

Figura 4.1. Tasa de ejemplos correctamente clasificados en la fase de test del experimento. Los datasets aparecen ordenados por número de ejemplos, siendo el primero "post_operative" y el último "banana".

Tras esto, el siguiente factor a analizar será el número de atributos, generando una gráfica (Figura 4.2) en la que se mostrará el rendimiento de los algoritmos en función del número de atributos.

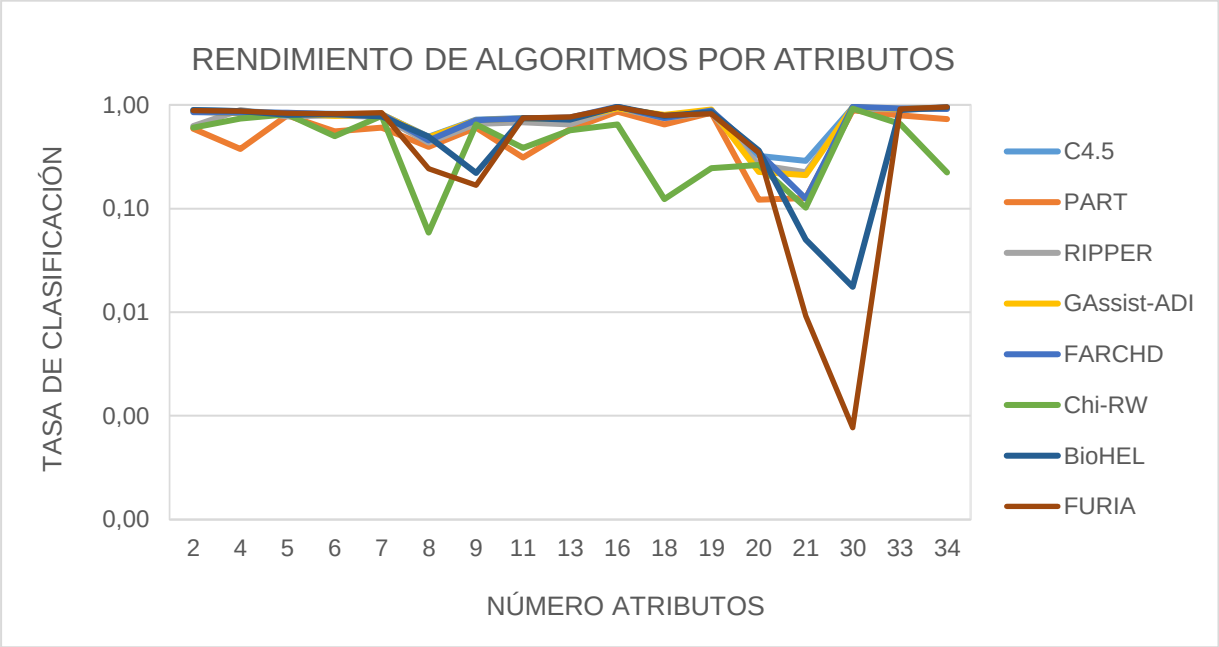

Figura 4.2. Tasa de ejemplos correctamente clasificados en la fase de test del experimento. Los datasets aparecen ordenados por número de atributos, siendo el menor 2 y el mayor 34 atributos.

El siguiente factor a analizar será el número de clases, generando una gráfica (Figura 4.3) que muestra los resultados desde 2 hasta 28 clases.

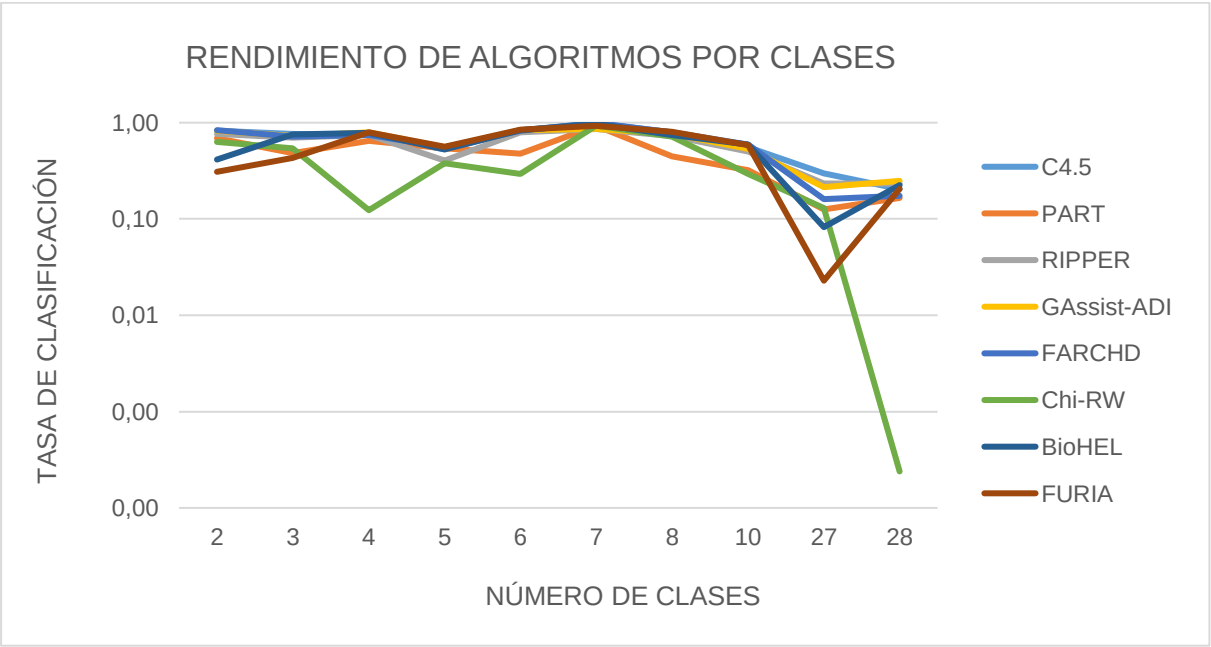

Figura 4.3. Tasa de ejemplos correctamente clasificados en la fase de test del experimento que permite analizar el efecto del número de clases.

Tras obtener las gráficas para la visualización de los datos, utilizaremos el ranking generado por el test de Friedman (Tabla 4.3).

Tabla 4.3. Ranking generado por el Test de Friedman, con los algoritmos ordenador de menor a mayor. El p-valor calculado por el test es de 8,46E-11, siendo mejor el que obtenga el ranking más bajo.

Friedman	
Algoritmo	**Ranking**
FARCHD	**3,23**
C4.5	**3,3**
FURIA	**3,37**
GAssist-ADI	**3,72**
BioHEL	**4,45**
RIPPER	**5,17**
PART	**6,3**
Chi-RW	**6,47**

Además, para corroborar los datos del ranking generados por el test de Friedman será necesario también tener en cuenta los datos obtenidos en el test de Holm para el estudio (Tabla 4.4).

Tabla 4.4. Resultados de la prueba de Holm tras el ajuste de los p-valor realizada sobre los resultados del experimento.

P-HOLM	
Algoritmo	**Ranking**
Chi-RW	**2,23*(10^-6)**
PART	**7,45*(10^-6)**
RIPPER	**0,01**
BioHEL	**0,22**
GAssist-ADI	**1,33**
FURIA	**1,67**
C4.5	**1,67**

Una vez visualizados los datos de forma general podemos disponernos a ver más detenidamente casos concretos de estudio para ver cómo se desarrollan los nuevos algoritmos basados en reglas y como procesa los resultados KEEL.

4.1. Caso de estudio: Dermatology

En este proyecto se escogió el dataset “dermatology” para su análisis en profundidad y visualización de las reglas generadas por el software. Este dataset fue escogido atendiendo a diferentes criterios, el principal de ellos el tener una tasa promedio de ejemplos correctamente clasificados bastante elevada. Además, este dataset está íntimamente ligado con el ámbito de la Biología, posee una serie de cualidades muy específicas y diversas (Tabla 4.5), es de una complejidad mayor que otros datasets con mayor tasa de acierto como pueden ser “wisconsin” y ha sido menos utilizado en el ámbito de la investigación, por lo que es más necesario su análisis en profundidad para poder contribuir a su uso en un futuro.

Tabla 4.5. Atributos numerados del dataset “dermatology”. Este dataset posee un total de 34 atributos útiles para la generación de reglas y 6 clases en las que clasificar los ejemplos.

ATRIBUTOS	
Clínicos	**Histopatológicos**
1: erythema	12: melanin incontinence
2: scaling	13: eosinophils in the infiltrate
3: definite borders	14: PNL infiltrate
4: itching	15: fibrosis (papillary dermis)
5: koebner phenomenon	16: exocytosis
6: polygonal papules	17: acanthosis
7: follicular papules	18: hyperkeratosis
8: oral mucosal involvement	19: parakeratosis
9: knee/elbow involvement	20: clubbing of the rete ridges
10: scalp involvement	21: elongation of the rete ridges
11: family history	22: thinning (suprapapillary epidermis)
34: Age	23: spongiform pustule
CLASES	24: munro microabcess
	25: focal hypergranulosis
C1: psoriasis	26: disappearance of the granular layer
C2: seboreic dermatitis	27: vacuolisation/damage (basal layer)
C3: lichen planus	28: spongiosis
C4: pityriasis rosea	29: saw-tooth appearance of retes
C5: cronic dermatitis	30: follicular horn plug
C6: pityriasis rubra pilaris	31: perifollicular parakeratosis
	32: inflammatory monoluclear inflitrate
	33: band-like infiltrate

Como se puede apreciar en la tabla (Tabla 4.5) el dataset “dermatology” posee un total de 34 atributos, 12 de las cuales corresponden a características clínicas del paciente que ya se tenían en la base de datos tras una primera evaluación, y 22 características histopatológicas tomadas a posteriori a través de muestras de piel. Los valores que toman estas muestras de piel a la hora de incluirlas en el dataset se determinaron a través de su observación al microscopio. Por otra parte, los ejemplos pueden clasificarse en un total de 6 posibles clases, las cuales hacen referencia a posibles enfermedades eritematoescamosas de la piel en el paciente. Las enfermedades de este grupo son la psoriasis, la dermatitis seborreica, el liquen plano, la pitiriasis rosada, la dermatitis crónica y la pitiriasis rubra pilaris.

Los atributos pueden tomar valores enteros comprendidos entre el 0 y el 3 en función de la gravedad y certeza que podemos tener de encontrar cada una de las características presentes en el paciente. Los únicos atributos que toman valores diferentes serían “Age”, que hace referencia a la edad del paciente en el momento de tomar la muestra en años y puede ir desde 0 hasta 75, y el atributo “family history”, que hace referencia a la presencia de alguna de las enfermedades relativas a las diferentes clases del dataset, pudiendo presentar valor 0 (no se ha encontrado presencia de ninguna de las enfermedades en familiares) y valor 1 (se ha encontrado presencia de al menos una de las enfermedades en familiares).

En este proyecto decidimos centrarnos en los resultados y reglas generadas por el algoritmo BioHEL por haber obtenido un resultado muy competente para el dataset de estudio y ser estadísticamente equivalente según el test de Holm con los dos algoritmos más potentes de los que hemos analizado (C4.5 y FARCHD), además de estar especialmente diseñado para la interpretación de datos biológicos y clínicos y por no haber sido tan empleado en investigación, por lo que existe una mayor necesidad de ampliar los conocimientos sobre su funcionamiento. De este modo, podemos observar los resultados que extraemos mediante el método Vis-Clas-Check (Figura 4.4).

```
TEST RESULTS
============
Classifier= dermatology
Fold 0 : CORRECT=0.9714285714285714 N/C=0.0
Fold 1 : CORRECT=0.8888888888888888 N/C=0.0
Fold 2 : CORRECT=0.9459459459459459 N/C=0.0
Fold 3 : CORRECT=0.9166666666666666 N/C=0.0
Fold 4 : CORRECT=1.0 N/C=0.0
Fold 5 : CORRECT=0.9166666666666666 N/C=0.0
Fold 6 : CORRECT=0.9428571428571428 N/C=0.0
Fold 7 : CORRECT=0.9714285714285714 N/C=0.0
Fold 8 : CORRECT=0.9722222222222222 N/C=0.0
Fold 9 : CORRECT=0.9714285714285714 N/C=0.0
Global Classification Error + N/C:
0.05024667524667524
stddev Global Classification Error + N/C:
0.03224130205190544
Correctly classified:
0.9497533247533247
```

Figura 4.4. Resultados en la fase de test del algoritmo BioHEL para el dataset "dermatology".

Se pueden apreciar los resultados de los diferentes “folds” expresados como tasa de ejemplos correctamente clasificados. En todos se obtiene un resultado bastante elevado, incluso en el “fold” 4 el algoritmo ha conseguido una tasa de clasificación perfecta. También es destacable que en ninguno de los test el algoritmo ha encontrado ejemplos que no es capaz de clasificar, algo que resulta especialmente útil visto el objetivo final de estos algoritmos, teniendo una tasa de error de clasificación global realmente baja. El resultado medio en la fase de test es pues de 0,9497 de tasa de ejemplos correctamente clasificados, un resultado realmente competente.

El diagnóstico diferencial de las enfermedades eritematoescamosas es un problema real en dermatología. Todas estas patologías comparten varias características clínicas como el eritema y la descamación, pero se diferencian en muy pocas. Por lo general, es necesaria una biopsia para el diagnóstico, pero desafortunadamente estas enfermedades también comparten muchas características histopatológicas. Otra dificultad para el diagnóstico diferencial es que una enfermedad puede mostrar los rasgos de otra enfermedad en la etapa inicial y puede tener los rasgos característicos en las etapas siguientes. Es por esto que el objetivo de los diferentes métodos con este conjunto de datos, entre ellos BioHEL, será desarrollar un algoritmo basados en un conjunto de reglas que permita predecir que enfermedad de la piel padece un paciente a partir de datos clínicos e histopatológicos de manera fiable y rápida.

De este modo, BioHEL generará una serie de reglas para cada “fold” basadas en las los patrones que observe el algoritmo. En el caso del “fold” 4, en el cual el algoritmo ha obtenido una clasificación perfecta de todos los ejemplos analizados durante la fase de test, las reglas generadas por el algoritmo son las siguientes:

- **Rule 0:** Att Fibrosis is [>0.0076363296] | Att Munro_microabcess is [<2.991524] | Att Vacuolisation is [<2.9987924] | **5**
- **Rule 1:** Att Inflammatory_monoluclear is [>0.0014304089] | Att Band-like_infiltrate is [>1.0062981] | **3**
- **Rule 2:** Att Elongation is [>0.00812252] | Att Spongiosis is [<0.9886784] | Att Perifollicular_parakeratosis is [<1.7703098] | **1**
- **Rule3:** Att Follicular_papules is [>2.4395781E-4] | Att Perifollicular_parakeratosis is [>0.0011993243] | **6**
- **Rule 4:** Att Koebner_phenomenon is [<0.999248] | Att Hyperkeratosis is [<1.9715142] | Att Granular_layer is [<0.99556065] | Att Age is [<61.795612] |**2**
- **Rule 5:** Att PNL_infiltrate is [<1.9996327] | Att Age is [<69.90731] | **4**
- **Rule 6:** Att Definite_borders is [0.73750657,2.761621] | **2**
- **Rule 7:** Default rule → **4**

Como podemos observar BioHEL ha sido capaz de generar 7 reglas mediante las cuales es capaz de clasificar todos los ejemplos de forma satisfactoria, un número bastante bajo en comparación a otros algoritmos como FARCHD, que para el mismo “fold” ha necesitado generar un total de 23 reglas. Las reglas se interpretan de forma que “Att” indica el atributo al que hace referencia, después aparece el nombre del atributo en cuestión junto a unos corchetes con los valores que toma dicho atributo para que la regla se cumpla, finalizando con una barra vertical que indica el final de la regla, de modo que esta estaría formada por varios condicionantes encadenados unos detrás de otros por barras verticales, acabando en un número que nos indica la clase a la que pertenecería el ejemplo que se pretende clasificar con dicha regla.

Existen algunas relaciones destacables. Por ejemplo, en el caso de la regla 0 si se cumplen los condicionantes que aparecen el ejemplo se clasificaría dentro de la clase 5, lo que nos estaría indicando que el paciente sufre dermatitis crónica. En caso de no poder clasificarse en ningún grupo, el algoritmo ha determinado que el ejemplo pertenece a la clase 4, por lo que el paciente sufriría pitiriasis rosada.

5. DISCUSIÓN

En primer lugar, al observar los resultados generales del experimento durante la fase de test (Tabla 4.1) podemos observar a simple vista que en la mayoría de los casos el proceso de aprendizaje ha sido efectivo para desarrollar reglas que clasifiquen satisfactoriamente los ejemplos de la mayoría de datasets. Sin embargo, al observar los resultados comparativos entre la fase de entrenamiento y test (Tabla 4.2) vemos que, para algunos datasets, en la fase de entrenamiento se obtiene un resultado considerablemente mayor que en la fase de test, lo que nos estaría indicando que las reglas generadas durante en la fase de entrenamiento no son suficientemente extrapolables en algunos casos, este es el caso de los datasets "amino_acid_composition", "hidrophobicity", "polarity" y "van_der_waals", todos ellos algoritmos provenientes de repositorios de WEKA.

Dichos datasets poseen el mismo número tanto de ejemplos como de clases, principalmente debido a que fueron recopilados del mismo repositorio y hacen referencia a una serie de experimentos comunes. Hay varios factores que podrían hacer que estos datasets hayan tenido unos resultados insatisfactorios ya que, aunque su número de instancias es el adecuado, todos ellos poseen un número alto tanto de atributos como de clases, pero lo más probable es que el problema se deba a un fenómeno intrínseco de los datos, como podría ser un fallo en el formato de entrada o un error humano en la anotación de los mismos, provocando que algún atributo o clase se encuentre mal anotado, ya que otros datasets con características similares poseen datos muy diferentes a los de los datasets nombrados.

En lo que respecta a las diferentes características de los datasets, podemos observar que el número de ejemplos de los mismos parece afectar ligeramente al rendimiento de los algoritmos (Figura 4.1), pues se observa un descenso generalizado del rendimiento de los algoritmos a partir del dataset 24 (pima). Sin embargo, el dataset con mayor número de instancias (banana) ha obtenido un muy buen resultado, fenómeno que podría deberse a que este dataset posee tan solo 2 atributos y 2 clases, por lo que los algoritmos no necesitan generar modelos tan complejos y por lo tanto se ven menos afectados por este factor. Además, observamos una gran inestabilidad del rendimiento para los algoritmos BioHEL y FURIA entre los datasets 17 y 22, lo que corresponde a los datasets problema de WEKA más "seaheart" y "wdbc", mientras que

C4.5 y FARCHD son muy estables independientemente del número de ejemplos que tenga el dataset, en contraposición también con Chi-RW. Dado que para la mayoría de algoritmos los resultados son satisfactorios podemos determinar que el número de instancias no es un factor que repercuta al resultado del aprendizaje dentro de un número de instancias comprendido entre 90 y 5300, aunque serían necesarios más experimentos con volúmenes de datos mucho más grandes que por limitaciones de hardware no pudieron ser realizados en este proyecto.

En lo que a número de atributos de refiere, no se observa ningún patrón que nos indique que el número de atributos afecta positiva o negativamente a los resultados del aprendizaje (Figura 4.2). El descenso del valor de la tasa de clasificación en los algoritmos con mayor número de atributos coincide con los algoritmos problema de WEKA, por lo que no se podría afirmar que es un patrón de comportamiento, algo que se refuerza con el hecho de que en los datasets con mayor número de atributos sí que presentan buenos resultados. Lo que si podemos observar es que el algoritmo Chi-RW presenta nuevamente resultados muy variables.

Por otra parte, en lo que respecta al número de clases sí que observamos un patrón de comportamiento (Figura 4.3). Se puede observar claramente como, cuando aumenta el número de clases en los últimos datasets, se produce un descenso generalizado de la tasa de ejemplos correctamente clasificados para todos los algoritmos a partir de los datasets con 8 clases (yeast). Aunque bien es cierto que dentro de este tramo final se encuentran los datasets problema de WEKA, también están incluidos otros datasets como “abalone” que no presentan ninguna relación con los anteriores y que, de hecho, provienen del propio repositorio de KEEL, lo que reafirma la hipótesis de que el número de clases en los que pueden clasificarse los datasets, a partir de cierto punto, puede que afecta negativamente al proceso de aprendizaje de los resultados. Además, el rendimiento de los algoritmos empieza a descender en el momento en el hay un mayor número de clases que de atributos, un fenómeno que podría ser el responsable del descenso de la tasa de clasificación.

Para acabar con las características, en lo que respecta a las características de los datasets podemos afirmar que el método Ignore-MV resulta efectivo para el proceso de aprendizaje de los algoritmos empleados en este experimento, pues los datasets con “missing values” (por ejemplo “hepatitis”) por lo general tiene un buen rendimiento. No obstante, se debe tener en cuenta que el método se basa en la eliminación de los

ejemplos con missing values por lo que, si aplicamos el método a un dataset con un número de ejemplos con demasiados missing values, es posible que quede un número insuficiente de ejemplos para un aprendizaje satisfactorio.

Por otro lado, en lo que respecta al ranking de Friedman (Tabla 4.3) podemos observar que el algoritmo con mejor rendimiento ha sido FARCHD, algo que resulta lógico pues es el algoritmo más moderno y que se basa en un concepto que a nivel de aprendizaje resulta mucho más potente, aprovechando todas las ventajas de sus predecesores, lo que en nuestro experimento ha resultado ser el mejor algoritmo. Le sigue muy próximo C4.5, algo que resulta inaudito pues es el algoritmo más antiguo y que por tanto peores resultados se esperaría que devolviese en comparación a sus predecesores. Puede deberse a que este algoritmo fue considerado en 2007 uno de los mejores algoritmos de clasificación según el libro de "Top 10 algorithms in data mining". También puede deberse a que los conjuntos de datos empleados para el experimento eran relativamente pequeños, por lo que tal vez con otros datasets más extensos este algoritmo no fuese tan eficiente. Lo que si podemos afirmar a simple vista es que Chi-RW y PART son los que peor rendimiento han tenido.

Sin embargo, para corroborar la veracidad del ranking generado por el test de Friedman será necesario realizar otra prueba estadística, la prueba de Holm. En la tabla de esta prueba (Tabla 4.4) se puede apreciar como su valor está por encima de 0,05 y se acepta la hipótesis de igualdad para los algoritmos C4.5, BioHEL, GAssist-ADI y FURIA para el algoritmo FARCHD, mientras que para los algoritmos Chi-RW, RIPPER y PART se rechaza la hipótesis de igualdad con respecto a FARCHD. Podemos afirmar pues que se obtienen resultados equivalentes y que por lo tanto no podemos afirmar que FARCHD sea mejor C4.5, BioHEL, GAssist-ADI y FURIA, aunque si podemos afirmar que es mejor que RIPPER, Chi-RW y PART, algo que ya en las propias gráficas anteriores se puede apreciar.

Para acabar, en lo que respecta al caso concreto de estudio podemos observar cómo BioHEL ha sido capaz de desarrollar un bajo número de reglas (alta interpretabilidad) que genera una alta tasa de clasificación correcta, lo que nos indicaría la eficiencia en el proceso de aprendizaje de este algoritmo. En este tipo de casos se puede comprobar la utilidad de los algoritmos basados en reglas para el descubrimiento de patrones en los conjuntos de datos, permitiendo ampliar los conocimientos sobre los mismos a los investigadores y usuarios de estos métodos.

6. CONCLUSIONES

En este proyecto se ha podido comprobar la eficacia de los algoritmos destinados a la clasificación de ejemplos de conjuntos de datos relacionados con la biología. Se ha comprobado que la mayoría de algoritmos basados en reglas presentan buenas tasas de ejemplos correctamente clasificados en experimentos de aprendizaje supervisado y que algunos algoritmos son mejor que otros.

Para los algoritmos analizados en este documento el que mejor resultados ha obtenido podemos afirmar que es FARCHD, pero existen muchas otras alternativas viables que se pueden considerar para futuras investigaciones. Los únicos algoritmos que obtienen peores resultados en los experimentos serían RIPPER, Chi-RW y PART, siendo estos muy inestables dependiendo del dataset, por lo que desaconsejamos su uso para conjuntos de datos con cualidades similares a los de este experimento.

De las diferentes características de los conjuntos de datos empleados en este experimento, no podemos afirmar que el número de ejemplos y de atributos influye en el proceso de aprendizaje de los algoritmos, a lo que podemos añadir que el método Ignore-MV es una herramienta eficaz para el preprocesamiento de los conjuntos de datos. Sin embargo, se ha podido comprobar que, a partir de cierto punto, un número de clases creciente influye negativamente en el proceso de aprendizaje y de los resultados, por lo que se desaconseja emplear estos algoritmos para conjuntos de datos en los que exista un número elevado de clases, siendo interesante un futuro análisis más profundo de este fenómeno y de su relación con el número de ejemplos, ya que este último factor podría tener efectos solo cuando el número de clases sea elevado. En cualquier caso, es altamente recomendable una buena depuración de los datos para evitar que el algoritmo que utilicemos pierda eficacia por motivos fácilmente evitables.

Además, podemos afirmar que los algoritmos basados en reglas son una herramienta potente no solo en la automatización de procesos, sino también para profundizar en los patrones que subyacen a los conjuntos de datos de interés, permitiendo a los investigadores y especialistas ahondar en las cualidades y comportamientos no conocidos de los sistemas, lo que podría derivar en nuevas líneas de investigación y en una mayor comprensión de sistemas que se creían ya conocidos.

7. BIBLIOGRAFÍA

- ABBASI, B. y GOLDENHOLZ, D. M. (2019): "Machine learning applications in epilepsy", Epilepsia, 60(10), pp. 2037–2047.

- ALCALÁ-FDEZ, J., ALCALÁ, R. y HERRERA, F. (2011): "A Fuzzy Association Rule-Based Classification Model for High-Dimensional Problems with Genetic Rule Selection and Lateral Tuning", IEEE Transactions on Fuzzy Systems, 19:5. pp. 857-872.

- AWAN, S. E., SOHEL, F., SANFILIPPO, F. M., BENNAMOUN, M. y DWIVEDI, G. (2018): "Machine learning in heart failure", Current Opinion in Cardiology, 33(2), pp. 190–195.

- AZODI, C. B., TANG, J. y SHIU, S. H. (2020): "Opening the Black Box: Interpretable Machine Learning for Geneticists", Trends in Genetics, 36(6), pp. 442–455.

- BACARDIT, J. y GARRELL, J.M. (2004): "Analysis and improvements of the adaptive discretization intervals knowledge representation", Genetic and Evolutionary Computation Conference, 3103, pp. 726-738.

- BACARDIT, J., BURKE, E.K. y KRASNOGOR, N. (2009): "Improving the scalability of rule-based evolutionary learning", Memetic Comp. 1, pp. 55–67.

- BACARDIT, J. y GARRELL, J. M. (2003): "Evolving Multiple Discretizations with Adaptive Intervals for a Pittsburgh Rule-Based Learning Classifier System. Genetic and Evolutionary Computation", GECCO 2003, pp. 1818–1831.

- BASTANLAR, Y. y ÖZUYSAL, M. (2013): "Introduction to Machine Learning", MicroRNA Biology and Computational Analysis, pp. 105–128.

- BATMAZ, Z., YUREKLI, A., BILGE, A. y KALELI, C. (2019): "A review on deep learning for recommender systems: challenges and remedies", Artif Intell Rev 52, pp. 1–37.

- BELLINGER, C., MOHOMED JABBAR, M. S., ZAÏANE, O. y OSORNIO-VARGAS, A. (2017): "A systematic review of data mining and machine learning for air pollution epidemiology", BMC Public Health, 17(1), pp. 1.

- BENKE, K. y BENKE, G. (2018): "Artificial Intelligence and Big Data in Public Health", International Journal of Environmental Research and Public Health, 15(12), pp. 2796.

- CASTELVECCHI, D. (2016): "Can we open the black box of AI?", Nature, 538, pp. 20-23.

- CHEN, CC., JUAN, HH., TSAI, MY. y HORNG-SHING, H. (2018): "Unsupervised Learning and Pattern Recognition of Biological Data Structures with Density Functional Theory and Machine Learning", Scientific Reports, 8, pp. 557.

- CHERN, C. C., CHEN, Y. J. y HSIAO, B. (2019): "Decision tree–based classifier in providing telehealth service", BMC Medical Informatics and Decision Making, 19(1), pp. 104.

- CHI, Z., YAN, H. y PHAM, T. (1996): "Fuzzy algorithms: with applications to image processing and Pattern Recognition", World Scientific.

- CORDON, O., DEL JESUS, M. y HERRERA, F. (1999): "A proposal on reasoning methods in fuzzy rule-based classification systems", International Journal of Approximate Reasoning, 20, pp. 21-45.

- CUOCOLO, R., CIPULLO, M. B., STANZIONE, A., UGGA, L., ROMEO, V., RADICE, L., BRUNETTI, A. y IMBRIACO, M. (2019): "Machine learning applications in prostate cancer magnetic resonance imaging", European Radiology Experimental, 3(1), pp. 1.

- DE JONG, K. A., SPEARS, W. M. y GORDON, D. F. (1993): "Using genetic algorithms for concept learning", Machine Learning, 13, pp. 161-188.

- DEMSAR, J. (2006): "Statistical comparisons of classifiers over multiple data sets", Journal of Machine Learning Research, pp. 1-30.

- DEO, R. C. (2015): "Machine Learning in Medicine", Circulation, 132(20), pp. 1920–1930.

- DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K., ZAL, G. y HAYASHI, Y. (2000): "Fuzzy and Crisp Logical Rule Extraction Methods in Application to Medical Data", Fuzzy Systems in Medicine, pp. 593–616.

- EINEBORG, M. y BOSTRÖM, H. (2001): "Classifying uncovered examples by rule stretching", Springer-Verlag, pp. 41–50.

- ESCOTO PONCE DE LEÓN, M. D. C., CERVANTES-LUNA, B. S. y CAMACHO RUIZ, E. J. (2020): "Cross-validation of the body appreciation scale-2: invariance

across sex, body mass index, and age in Mexican adolescents", Eating and Weight Disorders-Studies on Anorexia, Bulimia and Obesity, 26(4), pp. 1187–1194.

- FAGERLAND, M. W. y SANDVIK, L. (2009): "The Wilcoxon-Mann-Whitney test under scrutiny", Statistics in Medicine, 28(10), pp. 1487–1497.

- FRANCO, M. A., KRASNOGOR, N. y BACARDIT, J. (2013): "GAssist vs. BioHEL: critical assessment of two paradigms of genetics-based machine learning", Soft Computing, 17(6), pp. 953–981.

- FRANK, E. y WITTEN, I. (1998): "Generating Accurate Rule Sets Without Global Optimization", In Proceedings of the Fifteenth International Conference on Machine Learning, pp. 144-151.

- FÜRNKRANZ, J. y WIDMER, G. (1994): "Incremental reduced error pruning. In: Proceedings of the 11th international conference on machine learning", ICML, pp. 70–77.

- GACTO, M. J., SOTO-HIDALGO, J. M., ALCALA-FDEZ, J. y ALCALA, R. (2019): "Experimental Study on 164 Algorithms Available in Software Tools for Solving Standard Non-Linear Regression Problems", IEEE Access, 7, pp. 108916–108939.

- GOECKS, J., JALILI, V., HEISER, L. M. y GRAY, J. W. (2020): "How Machine Learning" Will Transform Biomedicine", Cell, 181(1), pp. 92–101.

- GOURRAUD, P. A., GÉNIN, E. y CAMBON-THOMSEN, A. (2004): "Handling missing values in population data: consequences for maximum likelihood estimation of haplotype frequencies", European Journal of Human Genetics, 12(10), pp. 805–812.

- GÜVENIR, H., DEMIRÖZ, G. y ILTER, N. (1998): "Learning differential diagnosis of erythemato-squamous diseases using voting feature intervals", Artificial Intelligence in Medicine, 13(3), pp. 147–165.

- HAMET, P. y TREMBLAY, J. (2017): "Artificial intelligence in medicine". Metabolism, 69, pp. 36-40.

- HOLLAND, J. H. (1975): *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, The University of Michigan Press.

- HÜHN, J. y HULLERMEIER, E. (2009): "FURIA: an algorithm for unordered fuzzy rule induction", data mining and Knowledge Discovery, 19 (3), pp. 293-319.

- ISHIBUCHI, H. y YAMAMOTO, T. (2005): *Rule weight specification in fuzzy rule-based classification systems*, IEEE Transactions on Fuzzy Systems, 13, pp. 428-435.

- JACKSON, D. A. (2000): "Biostatistical Analysis. Jerrold H. Zar", The Quarterly Review of Biology, 75(4), pp. 501–502.

- JOHNSON, K. W., TORRES SOTO, J., GLICKSBERG, B. S., SHAMEER, K., MIOTTO, R., ALI, M., ASHLEY, E. y DUDLEY, J. T. (2018): "Artificial Intelligence in Cardiology", Journal of the American College of Cardiology, 71(23), pp. 2668–2679.

- LONDON, A. J. (2019): "Artificial Intelligence and Black-Box Medical Decisions: Accuracy versus Explainability", Hastings Center Report, 49(1), pp. 15–21.

- MARTÍNEZ, A. y RODRÍGUEZ, C. (1993): *Inferencia estadística: un enfoque clásico.* Madrid: Ediciones Pirámide.

- MCKINNEY, S. M., SIENIEK, M., GODBOLE, V., GODWIN, J., ANTROPOVA, N., ASHRAFIAN, H., BACK, T., CHESUS, M., CORRADO, G. S., DARZI, A., ETEMADI, M., GARCIA-VICENTE, F., GILBERT, F. J., HALLING-BROWN, M., HASSABIS, D., JANSEN, S., KARTHIKESALINGAM, A., KELLY, C. J., KING, D. Y SHETTY, S. (2020): "International evaluation of an AI system for breast cancer screening", Nature, 577(7788), pp. 89–94.

- POCZETA, K., KUBUS, U. y YASTREBOV, A. (2019): "Analysis of an evolutionary algorithm for complex fuzzy cognitive map learning based on graph theory metrics and output concepts", Biosystems, 179, pp. 39–47.

- QUINLAN, J. R. (1986): "Induction of decision trees", Machine Learning, 1(1), pp. 81–106.

- ROOHI, A., FAUST, K., DJURIC, U. y DIAMANDIS, P. (2020): "Unsupervised Machine Learning in Pathology", Surgical Pathology Clinics, 13(2), pp. 349–358.

- ROYSTON, J. P. (1982): "Algorithm AS 181: The W Test for Normality", Applied Statistics, 31(2), pp. 176.

- RUDIN, C. (2019): "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead", Nature Machine Intelligence, 5, pp. 206-215.

- RUSSELL, S. J. (2021): *Inteligencia Artificial Un Enfoque Moderno* (2ª edición), PRENTICE HALL/PEARSON.

- SALZBERG, S.L. (1994): "C4.5: Programs for machine learning by J. Ross Quinlan", Mach Learn, 16, pp. 235–240.

- SCHMIDT, J., MARQUES, M.R.G., BOTTI, S. y MARQUES, M. (2019): "Recent advances and applications of machine learning in solid-state materials science", NPJ Computer Mater 5, pp. 83.

- SHESKIN, D. J. (2003): *Handbook of Parametric and Nonparametric Statistical Procedures (3rd edition)*, Chapman and Hall/CRC.

- STILGOE, J. (2017): "Machine learning, social learning and the governance of self-driving cars", Social Studies of Science, 48(1), pp. 25–56.

- STOKES, J. M., YANG, K., SWANSON, K., JIN, W., CUBILLOS-RUIZ, A., DONGHIA, N. M., MACNAIR, C. R., FRENCH, S., CARFRAE, L. A., BLOOM-ACKERMANN, Z., TRAN, V. M., CHIAPPINO-PEPE, A., BADRAN, A. H., ANDREWS, I. W., CHORY, E. J., CHURCH, G. M., BROWN, E. D., JAAKKOLA, T. S., BARZILAY, R. y COLLINS, J. J. (2020): "A Deep Learning Approach to Antibiotic Discovery", Cell, 180(4), pp. 688–702.

- TRIGUERO, I., GONZÁLEZ, S., MOYANO, J. M., GARCÍA, S., ALCALÁ-FDEZ, J., LUENGO, J., FERNÁNDEZ, A., DEL JESUS, M. J., SÁNCHEZ, L. y HERRERA, F. (2017): "KEEL 3.0: An Open-Source Software for Multi-Stage Analysis", data mining International Journal of Computational Intelligence Systems 10, pp. 1238-1249.

- VENTURINI, G. (1993): "SIA: A supervised inductive algorithm with genetic search for learning attributes-based concepts", Machine Learning, 93, pp. 280–296.

- YANG, Y. y HUANG, S. (2014): "Suitability of five cross validation methods for performance evaluation of nonlinear mixed-effects forest models- a case study", Forestry, 87, pp. 654-662.

- ZAR, J. (1999): *Biostatistical Analysis*, Prentice Hall.

8. ANEXO

En este anexo se incluye información adicional obtenida durante la realización del experimento.

Tabla 8.1. Resultados de la tasa de ejemplos correctamente clasificados durante la fase de entrenamiento para el experimento realizado en este proyecto.

DATASET	C4.5	PART	RIPPER	GAssist	FARCHD	Chi-RW	BioHEL	FURIA
abalone	0,752	0,166	0,463	0,252	0,173	0,001	0,241	0,224
amino_acid	0,839	0,131	0,793	0,256	0,538	0,496	0,779	0,673
appendicitis	0,910	0,890	0,958	0,940	0,940	0,890	1,000	0,915
banana	0,914	0,592	0,626	0,861	0,860	0,604	0,918	0,898
breast	0,776	0,718	0,865	0,864	0,914	0,858	0,968	0,778
bupa	0,859	0,616	0,863	0,794	0,785	0,599	0,983	0,797
cleveland	0,831	0,539	0,777	0,695	0,882	0,914	0,826	0,607
contraceptive	0,725	0,433	0,631	0,579	0,628	0,519	0,667	0,559
dermatology	0,983	0,756	0,996	0,960	0,999	1,000	0,992	0,987
ecoli	0,917	0,443	0,919	0,816	0,925	0,758	0,954	0,902
flare	0,786	0,311	0,752	0,765	0,800	0,451	0,786	0,759
hayes-roth	0,888	0,447	0,847	0,774	0,916	0,790	0,911	0,876
heart	0,916	0,574	0,914	0,909	0,938	0,972	1,000	0,885
hepatitis	0,949	0,905	0,958	0,996	1,000	0,989	1,000	0,968
hidrophobicity	0,815	0,141	0,799	0,230	0,239	0,190	0,713	0,495
ionosphere	0,987	0,812	0,982	0,974	0,987	0,977	0,993	0,976
iris	0,980	0,333	0,984	0,983	0,986	0,938	1,000	0,979
lymphography	0,923	0,718	0,935	0,955	1,000	1,000	1,000	0,936
mammographic	0,851	0,799	0,793	0,864	0,868	0,822	0,897	0,851
monks	1,000	0,528	1,000	0,995	0,999	0,972	1,000	1,000
pima	0,838	0,651	0,845	0,819	0,830	0,752	0,962	0,786
polarity	0,814	0,114	0,787	0,237	0,257	0,195	0,667	0,505
post-operative	0,718	0,716	0,822	0,843	0,906	0,865	0,932	0,719
saheart	0,786	0,657	0,796	0,822	0,823	0,785	0,979	0,746
van_der_waals	0,817	0,159	0,788	0,237	0,255	0,157	0,670	0,482
wdbc	0,991	0,891	0,985	0,983	0,986	0,961	0,999	0,995
wine	0,989	0,663	0,991	0,994	0,999	0,988	1,000	0,994
wisconsin	0,981	0,703	0,980	0,989	0,987	0,980	0,991	0,989
yeast	0,815	0,324	0,709	0,542	0,639	0,296	0,740	0,636
zoo	0,987	0,859	0,993	0,973	1,000	1,000	1,000	0,974
MEDIA	0,878	0,553	0,852	0,763	0,802	0,724	0,886	0,796

Tabla 8.2. Varianza media de los diferentes algoritmos obtenida para los diferentes datasets durante la fase de test.

DATASET	C4.5	PART	RIPPER	GAssist	FARCHD	Chi-RW	BioHEL	FURIA
abalone	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,001
amino_acid_composition	0,002	0,000	0,002	0,001	0,001	0,003	0,003	0,003
appendicitis	0,011	0,014	0,009	0,009	0,009	0,010	0,014	0,007
banana	0,000	0,002	0,004	0,000	0,000	0,000	0,000	0,000
breast	0,004	0,003	0,004	0,004	0,004	0,003	0,007	0,003
bupa	0,007	0,002	0,004	0,013	0,001	0,001	0,007	0,008
cleveland	0,004	0,001	0,008	0,003	0,009	0,006	0,006	0,004
contraceptive	0,003	0,000	0,003	0,002	0,002	0,002	0,001	0,001
dermatology	0,002	0,015	0,001	0,001	0,002	0,003	0,001	0,001
ecoli	0,002	0,004	0,003	0,004	0,001	0,003	0,005	0,001
flare	0,001	0,000	0,002	0,000	0,001	0,002	0,000	0,000
hayes-roth	0,012	0,002	0,006	0,020	0,008	0,008	0,004	0,011
heart	0,004	0,004	0,002	0,005	0,002	0,008	0,007	0,004
hepatitis	0,022	0,016	0,018	0,010	0,007	0,039	0,019	0,013
hidrophobicity	0,005	0,000	0,003	0,001	0,000	0,001	0,255	0,001
ionosphere	0,004	0,007	0,003	0,005	0,001	0,002	0,001	0,002
iris	0,002	0,000	0,002	0,001	0,002	0,002	0,001	0,002
lymphography	0,014	0,007	0,007	0,009	0,019	0,005	0,013	0,010
mammographic	0,003	0,007	0,007	0,002	0,003	0,002	0,002	0,003
monks	0,000	0,002	0,000	0,001	0,000	0,005	0,000	0,000
pima	0,001	0,000	0,002	0,001	0,002	0,002	0,003	0,001
polarity	0,004	0,000	0,001	0,000	0,001	0,001	0,001	0,001
post-operative	0,006	0,005	0,032	0,017	0,027	0,014	0,528	0,005
saheart	0,001	0,000	0,004	0,001	0,001	0,002	0,649	0,002
van_der_waals	0,002	0,000	0,002	0,001	0,001	0,001	0,261	0,001
wdbc	0,001	0,002	0,002	0,001	0,001	0,002	0,940	0,000
wine	0,003	0,003	0,002	0,004	0,005	0,003	0,005	0,001
wisconsin	0,001	0,010	0,001	0,001	0,001	0,001	0,001	0,000
yeast	0,002	0,001	0,002	0,001	0,001	0,001	0,000	0,001
zoo	0,004	0,042	0,006	0,004	0,004	0,013	0,003	0,007