ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA
Universidad de Córdoba

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Algoritmo CNN de selección de Instancias para Aprendizaje con Múltiples Instancias

Manual Técnico

Autor: Enrique Álvarez Gallego

Directoras: Dª. Amelia Zafra Gómez

Dª. Eva Lucrecia Gibaja Galindo

UNIVERSIDAD D CÓRDOBA

Dra. Amelia Zafra Gómez y **Dra. Eva Lucrecia Gibaja Galindo**, ambas profesoras del departamento de Informática y Análisis Numérico de la Universidad de Córdoba

Certifican:

Que el presente Trabajo Fin de Grado, titulado **"Algoritmo CNN de Selección de Instancias para Aprendizaje con Mútiples Instancias"** ha sido desarrollado por **D. Enrique Álvarez Gallego**, para aspirar al grado de Ingeniería Informática, bajo su dirección en la Escuela Politécnica Superior de la Universidad de Córdoba, reuniendo, a su juicio, las condiciones necesarias exigidas en este tipo de trabajos.

Y para que conste, se expida y firma el presente informe.

Córdoba, julio 2018

Fdo.:

Prof. Dª. Amelia Zafra Gómez Prof. Dª. Eva Lucrecia Gibaja Galindo

Contenido

Lista de figuras

Lista de tablas

Capítulo 1. Introducción

El aprendizaje automático (machine learning [1]) es el campo de las ciencias de computación, cuyo objetivo es dotar con la habilidad de “aprender” a una computadora mediante diferentes técnicas. Se trata de un proceso de inducción de conocimientos, ya que consiste en crear algoritmos que aprendan a hacer predicciones generalizadas a partir de una información suministrada en forma de ejemplos.

Dentro de los diferentes tipos de aprendizaje automático, el aprendizaje supervisado [2] deduce una función a partir de datos de entrenamiento donde se conoce la salida que se desea obtener (datos de entrada y datos de salida, es decir, los resultados deseados). Más concretamente, la tarea en la que nos centraremos en el aprendizaje supervisado sería la clasificación [3], donde el sistema de aprendizaje trata de etiquetar (clasificar) los ejemplos de entrada utilizando una entre varias categorías (clases).

El aprendizaje con multi-instancia [4] es una variante del aprendizaje automático tradicional basado en instancias simples, donde el conjunto de entrenamiento está compuesto por bolsas donde cada una contiene un conjunto de instancias. En este aprendizaje, se conoce la salida a la que pertenecen las bolsas, pero no la salida particular de cada una de las instancias.

La selección de instancias [5] es una técnica que persigue la reducción del tamaño de los conjuntos de datos. El objetivo es la selección de un subconjunto representativo de los ejemplos que representan el conjunto de entrenamiento inicial. Aquellas instancias que se desean obviar del subconjunto que queremos formar, bien porque son superfluas, es decir, no son necesarias para clasificar, o bien porque generan ruido, es decir, aquellas que solapan las diferentes clases y pueden llevar al clasificador a error.

Uno de los primeros algoritmos de selección de instancias que se diseñaron para el aprendizaje con instancias simples es el algoritmo CNN [6] (Condensed Nearest Neighbor Rule) (Hart. 1968). Este TFG pretende realizar

una adaptación de este algoritmo al aprendizaje con múltiples instancias y comprobar su funcionamiento.

La idea principal es reducir el conjunto de datos reduciendo a solo los puntos importantes. Los puntos se podrían dividir en 3 grupos [7]:

- Outliers: puntos que no serían reconocidos como correctos si se agregan a la base de datos más tarde.

- Prototipos: el set mínimo de puntos requeridos para que todos los puntos no-outliers sean bien reconocidos.

- Puntos absorbidos: puntos que no son outliers, y que serian bien reconocidos por el conjunto de puntos prototipo.

Los nuevos puntos solo necesitarán ser comparados con los puntos prototipo, en vez de con el conjunto de datos completo.

Capítulo 2. Definición del problema

En este punto se tratará de mostrar detalladamente el problema al que se pretende dar solución con la realización del proyecto. Para definir el problema, se distingue entre el problema real, el cuál trata sobre la propia visión ofrecida por el usuario, y el problema técnico, que define el problema desde el punto de vista de la ingeniería.

2.1 Identificación del problema real

El trabajo fin de grado que se va a realizar pretende ofrecer una librería que permita el estudio de la repercusión de la selección de instancias para conjuntos de datos multi-instancias.

Por tanto, en este trabajo se pretende diseñar dicha librería, donde se diseñen y desarrollen las clases necesarias que mantengan una estructura para poder llevar a cabo un estudio experimental con diferentes algoritmos.

Una vez diseñada la librería con toda su estructura de clases generales, será incluido el algoritmo CNN adaptado para trabajar en el aprendizaje con múltiples instancias, y se realizará un estudio experimental para comprobar su rendimiento.

2.2 Identificación del problema técnico

Para identificar el problema técnico se utilizará una técnica de ingeniería denominada PDS (Product Design Specification), una metodología que permite realizar un análisis de los principales condicionantes técnicos del problema,

dando respuesta a la formulación de una serie de preguntas que se detallan a continuación.

2.2.1 Funcionamiento

En este apartado se van a analizar las principales prestaciones que debe reunir el sistema informático que se pretende implementar desde un punto de vista técnico:

- La librería debe permitir cargar un experimento mediante un archivo de extensión xml [8].

- La librería será compatible con Weka [9], para que pueda hacerse uso de sus algoritmos de clasificación y de su formato del conjunto de datos.

- La librería debe permitir realizar un estudio experimental donde se indique el método de selección de instancias y el algoritmo de clasificación que hará uso de dicho conjunto reducido.

- La librería debe diseñarse de forma general para permitir la inclusión de nuevos algoritmos de selección de forma sencilla y eficiente.

- Una vez diseñada la librería con la funcionalidad anterior, se incluirá el algoritmo de selección de instancias CNN, que será adaptado para trabajar en este paradigma de aprendizaje.

2.2.2 Entorno

Esta sección se encuentra dedicada por completo a analizar el entorno relativo al componente software resultante del desarrollo de este trabajo. Dicho análisis será realizado desde cuatro puntos de vista diferentes: el entorno de Programación, de Software, de Hardware y de Usuario:

- Entorno de Programación: el sistema será desarrollado en el lenguaje de programación orientado a objetos Java [10], haciendo uso de la plataforma Eclipse [11], con el IDE [12] (Entorno de Desarrollo Integrado) de Java.

- Entorno Software: para el correcto funcionamiento de la aplicación será necesario disponer de la plataforma Eclipse, lo cual nos

permitirá utilizar el proyecto tanto en sistemas GNU/Linux [13] como Windows [14].

- Entorno Hardware: hace referencia a las características del sistema informático en en el que se ejecutará la aplicación, así como el ambiente que lo rodea. El software a desarrollar se instalará en un ordenador personal, existiendo en principio unos requisitos moderados con respecto a los recursos de memoria, almacenamiento y velocidad de los que deberá disponer; será necesario que el rendimiento del mismo resulte adecuado para poder hacer uso de la aplicación de manera correcta. En cuanto al entorno en el que se encontrará dicho ordenador personal, se considerará que está en condiciones propicias, cuando haya una iluminación adecuada para la detección del puntero láser.

- Entorno de Usuario: las clases creadas serán sencillas e intuitivas para el fácil uso del usuario.

2.2.3 Vida esperada

La vida esperada de un producto software puede definirse como el tiempo estimado durante el cual puede realizarse una aplicación útil del mismo.

2.2.4 Ciclo de mantenimiento

El ciclo de mantenimiento trata sobre el conjunto de modificaciones que una aplicación puede soportar frente a las diversas circunstancias que puedan surgir, o nuevas exigencias procedentes por parte del sistema o del usuario final.

La plataforma utilizada para desarrollar el proyecto posee un diseño modular que permitirá encontrar cualquier fallo en alguna de sus partes.

2.2.5 Competencia

Actualmente, no hay ningún sistema que permita el estudio de la repercusión de la selección de instancias, para conjuntos de datos multi-instancia.

2.2.6 Aspecto externo

En este apartado se abordan el conjunto de características que definirán la presentación de los componentes software generados en la realización del trabajo fin de grado:

- Interfaz de usuario: la interfaz del sistema será la misma que tenga la plataforma que se va a utilizar durante el desarrollo de la misma.

- Documentación: documentación que ha sido necesaria para el desarrollo del proyecto:

  - Manual Técnico: reúne todas las fases de búsqueda de requisitos, el diseño, el análisis y las pruebas del sistema.

  - Manual de Código: incluye información detallada de la implementación del software final.

  - Manual de Usuario: se detalla de una forma sencilla el acceso a la aplicación y el uso de todos los componentes de la misma.

- Formato de almacenamiento: la documentación realizada se entregará de forma digital en un CD-ROM, debido a la popularidad de este formato, su fácil portabilidad y el bajo coste que supone el mismo.

2.2.7 Estandarización

Se ha utilizado para el desarrollo del sistema el lenguaje de programación Java, las librerías de weka, multiInstanceLearning [15], citationKNN [16], commons-configuration [17].

El código fuente sigue un estándar de codificación, nombrando variables adecuadamente, tabulando el código a diferentes niveles para su fácil lectura, documentando el código para su fácil comprensión, separando el código fuente en varios ficheros y respetando el uso del espaciado aunque sea ignorado.

2.2.8 Calidad y fiabilidad

La calidad y la fiabilidad son factores muy importantes para que el usuario tenga garantías del correcto funcionamiento del software que está usando.

Destacar que, a priori, los únicos errores que debería cometer el sistema serán errores producidos por el uso incorrecto por parte del usuario.

2.2.9 Fases de desarrollo del proyecto

Conjunto de etapas que se seguirán para desarrollar la aplicación. Las etapas son las siguientes:

- Fase de Investigación: se definirá el dominio del problema, sus restricciones, y la naturaleza del mismo.

- Fase de Estudio: se estudiarán y comprenderán las tecnologías y herramientas que van a ser usadas en la realización del proyecto.

- Fase de Análisis: se han de identificar las funciones que deberá realizar el software y el rendimiento esperado.

- Fase de Diseño: se traducirán las características formales generales de la aplicación, tales como la estructura de los datos, la arquitectura del software, la caracterización del interfaz y los procedimientos, recogidos en la etapa anterior.

- Fase de Desarrollo: se implementará el diseño de la etapa anterior, utilizando los lenguajes y el conocimiento aprendidos en la fase de estudio.

- Fase de Pruebas: se realizarán todas las pruebas para comprobar el correcto funcionamiento del sistema, obteniendo las salidas deseadas.

- Fase de Experimentación: se realizará un pequeño estudio experimental para comprobar el correcto funcionamiento del algoritmo de selección CNN-MI.

- Fase de Documentación: de cada una de las fases anteriores se realizará una documentación en donde quedará constancia de todo el proceso realizado y los pasos seguidos. Esencialmente son tres documentos a realizar:

2.2.10 Pruebas

En este apartado se verificará que se cumplan las especificaciones que se planteen en el apartado de requisitos para así eliminar los posibles errores que se hayan cometido durante el desarrollo de la aplicación.

Todas las pruebas realizadas serán debidamente documentadas en el apartado correspondiente de pruebas.

2.2.11 Seguridad

La seguridad del sistema que se pretende desarrollar consistirá en garantizar que en la ejecución no se realice ninguna actividad incorrecta ajena a su propia funcionalidad.

No hará falta medidas de seguridad referentes a la confidencialidad de los datos, ya que el usuario no deberá proporcionar ningún tipo de información personal, careciendo estos de privacidad.

2.2.12 Planificación temporal

La normativa para un Trabajo Fin de Grado en Ingeniería Informática exige que tenga una duración de 300 horas que equivale a 12 créditos que se deben ser cubiertos. Estas horas se han repartido entre las diferentes fases del desarrollo del proyecto como se puede ver en la Tabla 2.1.

Tareas	Tiempo(h)
Investigación	25
+ Aprendizaje con múltiples instancias	5
+ Selección de instancias	10
+ Algoritmo CNN	10
Aprendizaje tecnologías	40
+ Eclipse	10
+ Java	15
+ Librerías (weka, CitationKNN, multiInstanceLearning)	15
Análisis	10
+ Funciones a realizar por el software	10
Diseño	60
+ Diseño de las estructuras de datos	40
+ Diseño de los modelos computacionales	20
Desarrollo	60
+ Programación de las estructuras de datos	35
+ Programación de los modelos computacionales	25
Pruebas	35
+ Diseño de las pruebas	10
+ Implementación de las pruebas	10
+ Análisis de los resultados	15
Experimentación	20
Documentación	50
+ Manual técnico	35
+ Manual de usuario	10
+ Manual de código	5
Total	300

Tabla 2.1. Planificación temporal

Capítulo 3. Objetivos

El objetivo principal de este trabajo fin de grado es el diseño e implementación de una librería que permita el estudio de la repercusión de la selección de instancias para conjuntos de datos multi-instancia, y que incluirá el algoritmo CNN adaptado para trabajar en el aprendizaje con múltiples instancias. Además se realizará un estudio experimental para comprobar su rendimiento.

Más concretamente, se pueden definir los siguientes sub-objetivos:

1. Introducción al aprendizaje con múltiples instancias y a la selección de instancias tanto en el aprendizaje con instancias simples como múltiples.
2. Diseño y creación de las clases principales de una librería para la selección de instancias para aprendizaje con múltiples instancias.
3. Dicha librería deberá permitir realizar un estudio experimental, configurando los experimentos mediante ficheros xml, y donde se pueda especificar el conjunto de datos, el método de selección de instancias, y el algoritmo de clasificación.
4. Diseño e implementación de un generador de informes de salida, de los experimentos realizados.
5. Diseño e implementación de la adaptación del algoritmo CNN al aprendizaje con múltiples instancias, CNN-MI.
6. Diseño y evaluación de un pequeño estudio experimental para comprobar el funcionamiento del algoritmo CNN-MI en el entorno de aprendizaje con múltiples instancias.

Capítulo 4. Antecedentes

En este apartado vamos a ver con más detalle la técnica de selección de instancias y el algoritmo CNN.

4.1 Selección de instancias

La selección de instancias es una técnica cuyo objetivo es la reducción del número de instancias en el conjunto de datos, intentando no perder la calidad en la clasificación. Los beneficios de su aplicación son disminuir los requerimientos de almacenamiento y de tiempo en la etapa de clasificación.

Con la disminución del tamaño de los datos también reducimos los costes de procesamiento de esos datos: cualquier algoritmo de los empleados en minería de datos tendrán el cardinal del conjunto de datos con los que trabaja, en la descripción de su complejidad.

Esto especialmente sensible en los algoritmos de clasificación “lazy”, como el de los k-vecinos más cercanos: cuando se intenta clasificar un nuevo elemento este se debe comparar con todas las instancias del conjunto de entrenamiento para encontrar el vecino o los vecinos más cercanos, por lo que el coste computacional va a depender del tamaño de ese conjunto.

A la hora de reducir el tamaño de los conjuntos de entrenamiento, tendremos que seleccionar aquellos que sean más relevantes y que representen mejor al conjunto completo, de manera que consigamos aproximadamente los mismos resultados que si trabajáramos con el conjunto de datos completo.

Además de que el tiempo de cómputo de los algoritmos se vea reducido, veremos que en muchas ocasiones tendremos un rendimiento mejor del algoritmo de clasificación usando el subconjunto seleccionado que usando el conjunto completo. Esto se debe a que en ocasiones, el algoritmo de selección consigue eliminar outliers, ruido y elementos redundantes.

En el caso de multi-instancias, varía que el conjunto de datos son bolsas de instancias, por lo que no haremos una reducción de instancias en sí, sino una reducción del número de bolsas. Al eliminar una bolsa del conjunto de datos, estamos eliminando todas las instancias que se encontraban en ella.

Se podría decir que de un conjunto de datos T con un número de bolsas n, elegiremos un subconjunto S con número de bolsas m, donde: $n>m$.

4.2 Algoritmo CNN

El algoritmo CCN (Condensed Nearest Neighbor Rule) fue motivado por la regla del vecino más cercano.

La regla del vecino más cercano asigna a una instancia sin clasificar la misma clase que la instancias más cercana a ella bien clasificada.

El algoritmo CNN funciona de la siguiente forma:

Lo primero es crear dos contenedores llamados "*Store*" y "*Garbage*".

La primera instancia es alojada en el contenedor *Store*.

La segunda instancia es clasificada por la regla NN, usando como referencia el contenedor Store (la cual solo contiene la primera instancia en el momento). Si la segunda instancia es bien clasificada, es guardada en el contenedor *Garbage*, en caso contrario en *Store*.

El procedimiento anterior es realizado con cada una de las instancias, con las mismas consecuencias, si son bien clasificadas son alojadas en *Garbage*, y en caso incorrecto en *Store*.

Cuando ya se hayan clasificados todas las instancias del set original, volverá a repetirse el mismo procedimiento, pero clasificando las instancias que esté guardadas en *Garbage*.

El algoritmo parará bien cuando el contenedor *Garbage* esté vacío, es decir, el subconjunto resultante del algoritmo es el conjunto de datos original, o bien cuando se complete una "vuelta" completa de *Garbage*, sin que ninguna instancia de esta haya sido transferida al contenedor *Store*.

El contenido final de *Store* es el subconjunto resultante que buscamos.

Capítulo 5. Restricciones

En este apartado se van a exponer todas aquellas restricciones o limitaciones existentes en el diseño y que van a condicionar la elección de una u otra alternativa. Estos factores se encuentran divididos en dos tipos.

5.1 Factores dato

Son los factores que están impuestos por la propia naturaleza del proyecto, sin dar lugar a otra posibles alternativas.

- El proyecto será desarrollado en el lenguaje de programación Java, dado por las librerías que se desean utilizar.

- Debido a que se va a codificar en Java, será necesario abordar el proceso de especificación de requisitos y el diseño de la aplicación desde la perspectiva de la orientación a objetos.

- Limitaciones de hardware y software, utilizando recursos personales.

5.2 Factores Estratégicos

Son los factores cuyo valor ha sido establecido por los técnicos del proyecto, dependiendo de las posibles alternativas que se hayan considerado.

- Para la codificación del proyecto se va a utilizar la plataforma Eclipse. Entre las razones por las que se ha elegido esta plataforma están ser software libre, y ya tener incluido tanto el IDE de Java como el compilador.

- Para la elaboración de la documentación se utilizará el editor de textos LibreOffice Writer [18], así como el editor de diagramas online draw.io [19] y el editor de gráficas online generadordegraficos.com [20]. No se han tenido en cuenta otros editores de texto, como Microsoft Office [21], debido a que se desea realizar la mayor parte de este proyecto con software libre.

Capítulo 6. Recursos

6.1 Recursos humanos

El autor de este proyecto encargado del análisis, diseño, programación y de la elaboración de la documentación es el alumno del Grado en Ingeniería Informática Enrique Álvarez Gallego.

Las directoras encargados de la coordinación del proyecto son:

- Dª. Amelia Zafra Gómez, profesora del Departamento de Informática y Análisis Numérico, como directora del proyecto, encargada de la supervisión del proyecto, ayudando, aconsejando y corrigiendo el proyecto documento del alumno.

- Dª. Eva Lucrecia Gibaja Galindo, profesora del Departamento de Informática y Análisis Numérico, como directora del proyecto, encargada de la supervisión del proyecto, ayudando, aconsejando y corrigiendo el proyecto documento del alumno.

6.2 Recursos hardware

En este apartado se hace referencia a todas las herramientas físicas disponibles para llevar a cabo el trabajo para el desarrollo del proyecto. Estas herramientas son:

- Ordenador de sobremesa:

  - Procesador Intel® Core™ i5-6500 3,20GHz

  - 8,0 GB de memoria RAM

- 120 GB de disco duro en estado sólido (SSD)
- Tarjeta gráfica NVIDIA EVGA GeForce GTX 1060 SC Gaming – 3 GB GDDR5.

- Ordenador portátil:
  - Procesador Intel® Core™ i7-4500 1,80GHz
  - 8,0 GB de memoria RAM
  - Tarjeta gráfica NVIDIA GeForce GTM 740 M – 2 GB DDR3

6.3 Recursos software

Se detallan a continuación los programas y paquetes que se han empleado en la elaboración del proyecto:

- Sistema Operativo GNU/Linux Ubuntu 16.04.4 de 64 bits instalado en el ordenador portátil.
- Sistema Operativo Windows 10 de 64 bits instalado en el ordenador de sobremesa.
- En la elaboración de la documentación, se utilizará como editor de texto LibreOffice Writer versión 5.2.5.1.
- Para las presentaciones se utilizará LibreOffice Impress [22] versión 5.2.5.1.
- Lenguaje de programación interpretado y orientado a objetos Java, en su implementación OpenJDK versión 1.8.
- Plataforma Eclipse, versión 4.7 Oxygen, con el IDE de Java. Todo el código del proyecto se realizará bajo esta plataforma.
- Librería de Weka.
- Librería de CitationKNN.
- Librería de multiInstanceLearning.

Capítulo 7. Especificación de requisitos

En este apartado se detallan todos los requisitos que especifiquen lo que se espera que realice el sistema, lo que debe hacer para cumplirlos, la información con la que va a tratar, lo que deben satisfacer los componentes externos que interactúan con el usuario y las características que éste deberá presentar, pero que no añadan ninguna funcionalidad.

7.1 Requisitos de información

En este apartado se recoge la información que la aplicación deberá gestionar y almacenar. Dichos requisitos van a ser codificados con las siglas RI seguidas de un número de identificación.

Los requisitos de información son:

- RI-01: Algoritmo de clasificación.
  - Descripción: el sistema almacenará la información relativa al algoritmo de clasificación que se utilizará en las pruebas
  - Datos específicos: los datos que se manejarán son:
    - Clasificador: nombre del clasificador.
    - Opciones de ejecución del clasificador, las cuáles dependerán del clasificador que estemos utilizando.

- RI-02: Algoritmo de selección.

  - Descripción: el sistema almacenará la información relativa al algoritmo de selección que se pretende utilizar.

  - Datos específicos: los datos que se manejarán son:

    - Algoritmo: el algoritmo en cuestión.

    - Opciones del algoritmo de selección, las cuáles dependerán del algoritmo que estemos utilizando.

- RI-03: Archivo de conjunto de datos.

  - Descripción: el sistema almacenará la información relativa al conjunto de datos que se desea estudiar.

  - Datos específicos:

    - Dirección del fichero.

- RI-04: Archivo de salida

  - Descripción: el sistema almacenará información relativa al fichero donde se escribirá la salida obtenida.

  - Datos específicos:

    - Dirección del fichero.

7.2 Requisitos funcionales

A continuación se van a detallar los requisitos funcionales que deberá satisfacer el sistema software. Dichos requisitos van a ser codificados con las siglas RF seguidas de un número de identificación.

Los requisitos funcionales son:

- RF-01: La librería deberá permitir al usuario configurar el algoritmo de selección.

- RF-02: La librería deberá permitir al usuario configurar el algoritmo de clasificación.

- RF-03: La librería deberá permitir al usuario ejecutar un algoritmo de selección.

- RF-04: La librería deberá permitir al usuario ejecutar el algoritmo de clasificación que utilizará un método de selección de instancias.

- RF-05: La librería deberá permitir al usuario obtener un informe de salida.

7.3 Requisitos no funcionales

A continuación se van a detallar los requisitos no funcionales que deberá satisfacer el sistema software. Dichos requisitos van a ser codificados con las siglas RNF seguidas de un número de identificación.

Los requisitos no funcionales son:

- RNF-01: El sistema deberá ser eficaz, robusto y fiable.

- RNF-02: El sistema se deberá poder mantener de forma fácil.

- RNF-03: La jerarquía de clases debe ser intuitiva para el fácil uso del usuario.

- RNF-04: El sistema deberá responder contra los posibles errores, ya sean por parte del usuario o del propio sistema.

- RNF-05: Todos los mensajes de error deberán ser significativos, de tal forma que incluyan un texto descriptivo e indicaciones que el usuario deberá efectuar ante dicho error.

- RNF-06: El sistema deberá estar siempre disponible.

Capítulo 8. Análisis

8.1 Análisis de los casos de uso

En este apartado se van a describir, mediante los diagramas de casos de uso de UML [21], el comportamiento y la comunicación que deberá tener el sistema mediante la interacción con el usuario, pero sin llegar a especificar cómo se va a implementar dicho comportamiento.

Los casos de uso explican el comportamiento deseado de la librería, si especificar su implementación. Un caso de uso describirá toda la secuencia de posibles eventos y acciones que se puedan producir entre un actor y el sistema, los cuales interactúan para conseguir un objetivo. Dicho esto, detallaremos los casos de uso que se dan en nuestra librería, viendo en ellos todas las funcionalidades de las que dispondrá el sistema a desarrollar.

Para especificar los casos de uso se seguirá la plantilla que se puede observar en la Tabla 8.1.

Casos de uso ID	**Nombre del caso de uso.**
Actores	Actores que intervienen en el caso de uso.
Descripción	Descripción informar de los objetivos del caso de uso.
Precondiciones	Conjunto de precondiciones que deben cumplirse para que el caso de uso se realice de forma correcta.
Pasos a seguir	Secuencia de pasos necesarios a seguir para la correcta realización del caso de uso.

Tabla 8.1: Plantilla de casos de uso

A la vista de los objetivos y requisitos que se han ido exponiendo en el sistema a desarrollar, sólo habrá un tipo de actor, que representará al usuario de la librería.

En los siguientes apartados se van a especificar los casos de uso de la aplicación que se va a desarrollar, para lo que se harán uso de diagramas y la tabla anterior.

8.1.1 Casos de uso

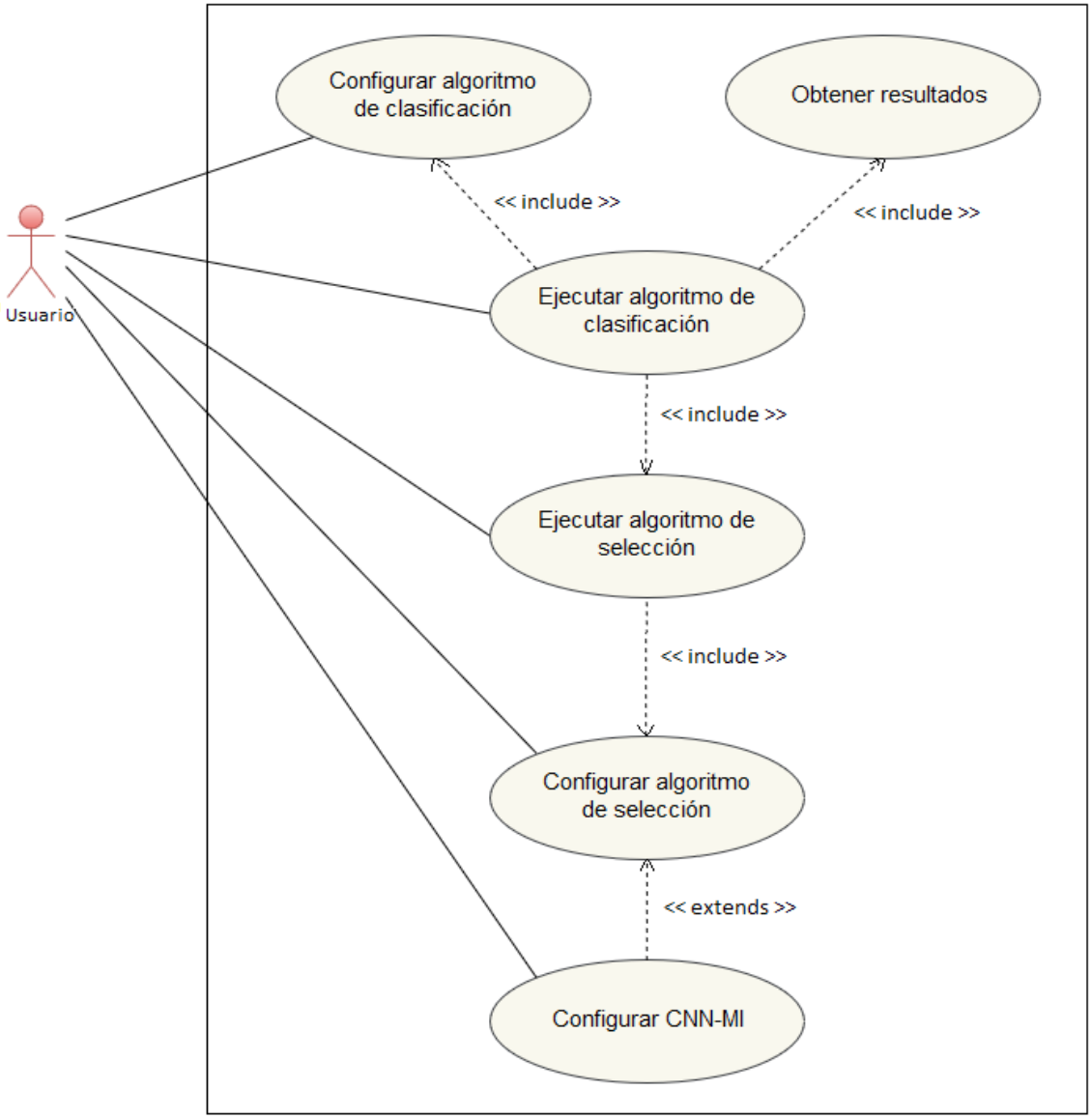

Figura 1: Diagrama CU Algoritmo ad

Casos de uso 1	Ejecutar algoritmo de clasificación
Actores	Usuario
Descripción	El usuario ejecuta el algoritmo de clasificación sobre un conjunto de datos reducido.
Precondiciones	-
Pasos a seguir	1. El usuario selecciona ejecutar el algoritmo. 2. El sistema prepara la configuración del algoritmo y el conjunto de datos que se va a utilizar en la ejecución. 3. El sistema ejecuta el algoritmo. 4. El sistema obtiene los resultados.

Tabla 8.2: CU Ejecutar algoritmo de clasificación

Casos de uso 2	Configurar algoritmo de clasificación
Actores	Usuario
Descripción	El usuario introduce los parámetros para el algoritmo de selección.
Precondiciones	Existe un fichero xml para la introducción de parámetros del algoritmo.
Pasos a seguir	1. El usuario configura el algoritmo de clasificación. 2. El usuario configura el directorio para el fichero de salida.

Tabla 8.3: CU Configurar algoritmo de clasificación

Casos de uso 3	Ejecutar algoritmo de selección
Actores	Usuario
Descripción	El usuario ejecuta el algoritmo de selección sobre un conjunto de datos
Precondiciones	-
Pasos a seguir	1. El usuario selecciona ejecutar el algoritmo. 2. El sistema prepara la configuración del algoritmo y el conjunto de datos que se va a utilizar en la ejecución. 3. El sistema ejecuta el algoritmo. 4. El sistema obtiene el conjunto de datos reducido.

Tabla 8.4: CU Ejecutar algoritmo de selección

Casos de uso 4	Configurar algoritmo de selección
Actores	Usuario
Descripción	El usuario introduce los parámetros para el algoritmo de selección.
Precondiciones	-
Pasos a seguir	1. El usuario configura el algoritmo de selección. 2. El usuario configura conjunto de datos a utilizar. 3. El usuario configura el directorio para el fichero de salida.

Tabla 8.5: CU Configurar algoritmo de selección

Casos de uso 5	Configurar CNN-MI
Actores	Usuario
Descripción	Configurar los parámetros del algoritmo de selección CNN-MI.
Precondiciones	-
Pasos a seguir	1. El usuario define la semilla a utilizar. 2. El usuario define si desea utilizar el límite de porcentaje. 3. El usuario define el porcentaje límite de instancias a mantener.

Tabla 8.6: CU Configurar CNN-MI

Casos de uso 7	Obtener resultados
Actores	-
Descripción	Los resultados obtenidos de un experimento son guardados en un fichero de salida.
Precondiciones	Se ha especificado correctamente un fichero de salida en la configuración del algoritmo.
Pasos a seguir	1. El sistema ejecuta el algoritmo. 2. El sistema obtiene el conjunto de datos reducido y las métricas. 3. El sistema guarda el conjunto de datos reducido y las métricas en el fichero de salida.

Tabla 8.7: CU Obtener resultados

8.2 Validación de casos de uso

Para finalizar este capítulo, se genera la siguiente tabla Tabla 8.9, en la que se puede observar la realización de una matriz que interrelaciona cada

caso de uso con los requisitos funcionales establecidos. Esto nos permite comprobar que la consecuencia de cada uno de los casos de uso descritos, definiría de forma completa la funcionalidad esperada.

RF/CU	CU1	CU2	CU3	CU4	CU5	CU6
RF1				X	X	
RF2		X				
RF3			X			
RF4	X					
RF5						X

Tabla 8.8: Validación casos de uso

Capítulo 9. Diseño

En este apartado se va a definir la arquitectura general del proyecto, especificando las partes que componen este y la relación entre ellos.

9.1 Diseño arquitectónico

Se describirá la arquitectura del sistema utilizando el diagrama de despliegue de UML, figura 3. Se identifican dos piezas claves en nuestro sistema, por un lado la parte de selección de instancias, y por otra los clasificadores multi-instancia.

- Selección de instancias: se encargará de realizar la selección de instancias del dataSet que se va a utilizar en el estudio.

- Algoritmos de clasificación: en este paquete se encuentran los algoritmos de clasificación multi-instancia disponibles para usar en el estudio experimental a realizar.

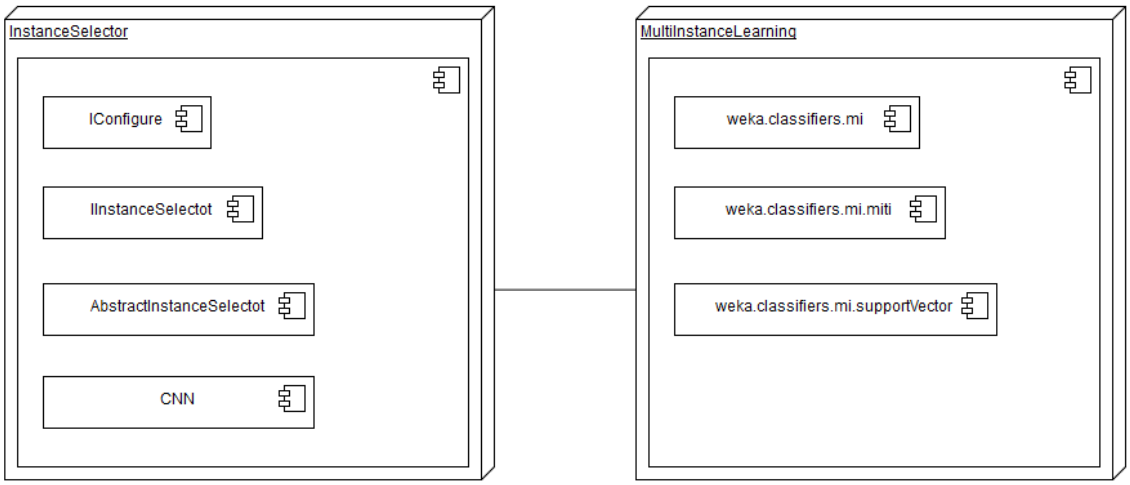

Figura 2: Diagrama despliegue

9.2 Diseño procedimental

En este apartado se van a detallar las clases que se tendrán que utilizar e implementar. Estas se van a extraer de la especificación de requisitos.

En cada una de las clases, que se van a describir a continuación, se distinguirán principalmente los siguientes apartados:

- Nombre de la clase: nombre que se le va a asignar a la clase para su identificación. Este nombre, deberá ser descriptivo, de tal manera que proporcione una idea sobre el funcionamiento general del elemento al cual pertenece.

- Descripción general de la clase: se explicará el objetivo u objetivos de una forma breve y concisa, además de cualquier dato e información de interés sobre esta.

- Variables miembro de la clase: se analizarán todos los atributos que definan la clase y necesarios para almacenar la información necesaria para el correcto desarrollo de la clase y la funcionalidad que se le ha asignado.

- Métodos de la clase: se analizarán todos aquellos métodos pertenecientes a la clase, es decir, las funciones en las que se estructurará la funcionalidad desarrollada por ella.

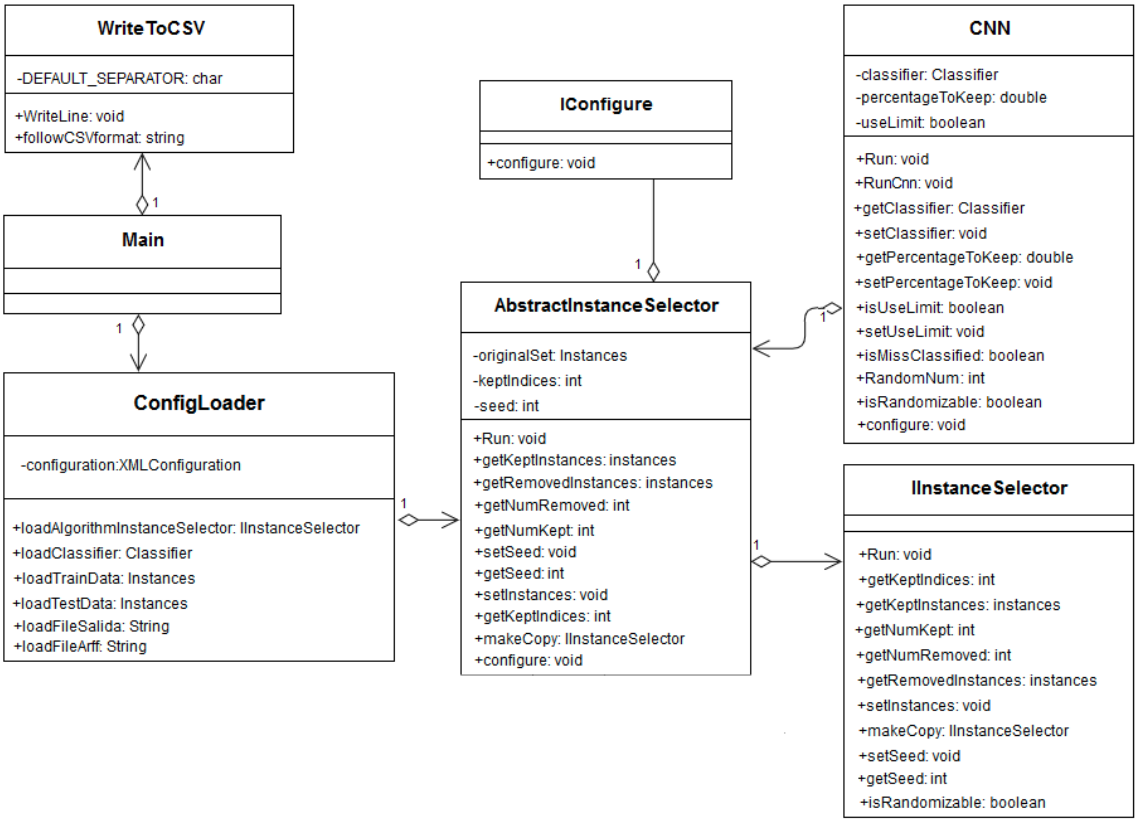

Figura 3: Diagrama de clases

9.2.1 Paquete Configuration

En esta sección se van a detallar todas las clases que se encuentran dentro del paquete Configuration.

9.2.1.1 Clase ConfigLoader

- Nombre: ConfigLoader
- Descripción: esta clase es la encargada de leer el archivo xml donde están todos los datos para el experimento.
- Variables miembro:
  - configuration: objeto de tipo XMLConfiguration, donde esta toda la información para el experimento.
- Métodos de la clase:
  - configLoader: constructor de la clase, crea el objeto configuration con el fichero XML.

- loadAlgorithmInstancesSelector: método que crea el algoritmo de selección leyendo el fichero XML, y lo devuelve.

- LoadClassifier: método que crea el algoritmo de clasificación y lo devuelve, leyendo el fichero XML.

- loadTrainData: método que crea y devuelve el objeto Instances de la base de datos con la que vamos a trabajar leyendo el fichero XML.

- LoadTestData: método que crea y devuelve el objeto Instances de la base de datos con la que vamos a trabajar leyendo el fichero XML.

- loadFileSalida: método que obtiene y devuelve el nombre del fichero de salida para las métricas.

- loadFileArff: método que obtiene y devuelve el nombre del fichero de salida para el conjunto de datos reducido obtenido.

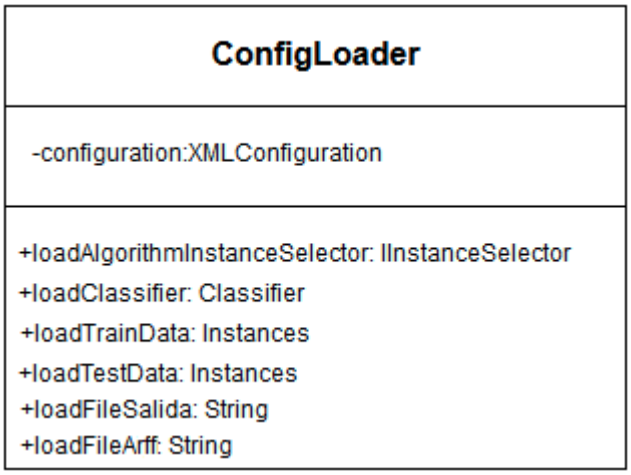

Figura 4: Clase ConfigLoader

9.2.2 Paquete InstanceSelector

En esta sección se van a detallar todas las clases que se encuentran dentro del paquete Configuration.

9.2.2.1 Clase IConfigure

- Nombre: IConfigure

- Descripción: esta clase funciona como interfaz para la clase genérica de los selectores de instancia, para las funciones que leen el archivo XML.

- Variables miembro: no hay.

- Métodos de la clase:

  - configure: constructor de la clase.

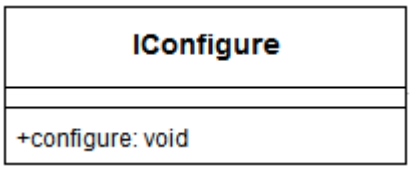

Figura 5: Clase IConfigure

9.2.2.2 Clase IInstanceSelector

- Nombre: IInstanceSelector

- Descripción: esta clase funciona como interfaz para la clase genérica de los selectores de instancia.

- Variables miembro: no hay.

- Métodos de la clase:

  - Run: realiza la selección de instancias.

  - getKeptIndices: obtendrá el vector de indices seleccionados.

  - getKeptInstances: obtendrá las instancias seleccionadas.

  - getNumKept: obtendrá el número de instancias seleccionadas.

  - getNumRemoved: obtendrá el número de instancias no seleccionadas.

  - getRemovedInstances: obtendrá las instancias no seleccionadas.

  - setInstances: establecerá el conjunto original de instancias.

- makeCopy: realiza una copia.
- setSeed: establecerá la semilla.
- getSeed: obtendrá la semilla.
- isRandomizable: obtendrá si es aleatorio o no.

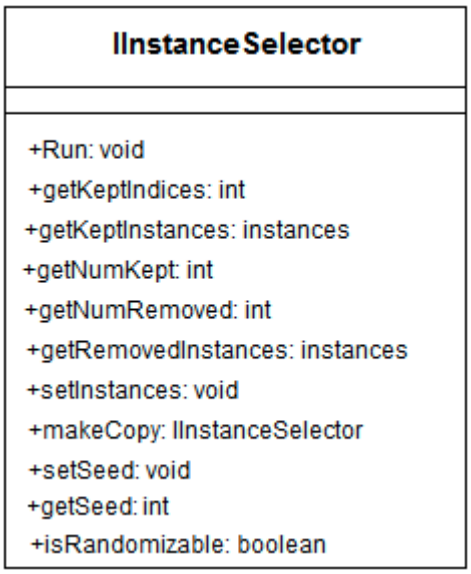

Figura 6: Clase IInstanceSelector

9.2.2.3 Clase AbstractInstanceSelector

- Nombre: AbstractInstanceSelector
- Descripción: esta clase funciona como clase genérica para los algoritmos de selección de instancias.
- Variables miembro:
  - originalSet: objeto de tipo Instances, conjunto de datos.
  - keptIndices: objeto de tipo int, vector que indica que instancias son seleccionadas.
  - seed: objeto de tipo int, semilla para la aleatoriedad.
- Métodos de la clase:

- AbstractInstanceSelector: constructor de la clase.
- Run: realiza la selección de instancias.
- setSeed: establecerá la semilla.
- getSeed: obtendrá la semilla.
- setInstances: establecerá el conjunto original de instancias.
- getKeptIndices: obtendrá el vector de indices seleccionados.
- getNumKept: obtendrá el número de instancias seleccionadas.
- getNumRemoved: obtendrá el número de instancias no seleccionadas.
- getKeptInstances: obtendrá las instancias seleccionadas.
- getRemovedInstances: obtendrá las instancias no seleccionadas.
- makeCopy: realiza una copia.
- configure: establece el valor de la semilla según el archivo xml.

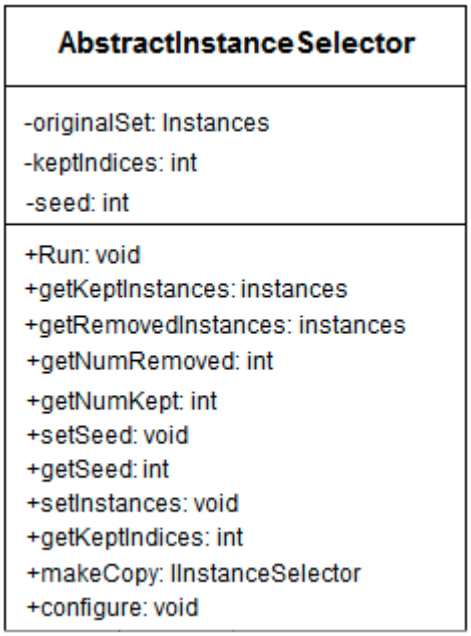

Figura 7: Clase AbstractInstanceSelector

9.2.2.4 Clase CNN

- Nombre: CNN
- Descripción: clase para el selector de instancias CNN. Esta clase hereda de la clase AbstractInstanceSelector.
- Variables miembro:
  - classifier: objeto de tipo classifier.
  - percentajeToKeep: objeto de tipo double, define el porcentaje de instancias a seleccionar.
  - useLimit: objeto de tipo boolean.
- Métodos de la clase:
  - CNN: constructor de la clase.
  - Run: pasos previos para la realización de la selección de instancias.
  - RunCNN: realiza la selección de instancias.

- GetClassifier: obtendrá el clasificador utilizado en la selección de instancias.

- SetClassifier: establecerá el clasificador a utilizar en la selección de instancias.

- isUseLimit: obtendrá si utilizar el limite de tamaño.

- SetUseLimit: establecerá si utilizar el limite de tamaño.

- getPercentageToKeep: obtendrá el porcentaje de instancias a mantener.

- setPercentageToKeep: establecerá el porcentaje de instancias a mantener.

- IsMissClassified: obtendrá si un instancia está bien o mal clasificada.

- RandomNum: obtendrá un número aleatorio diferente a otro.

- configure: establece el valor de la porcentaje según el archivo xml.

- IsRandomizable: obtendrá si es aleatorio o no.

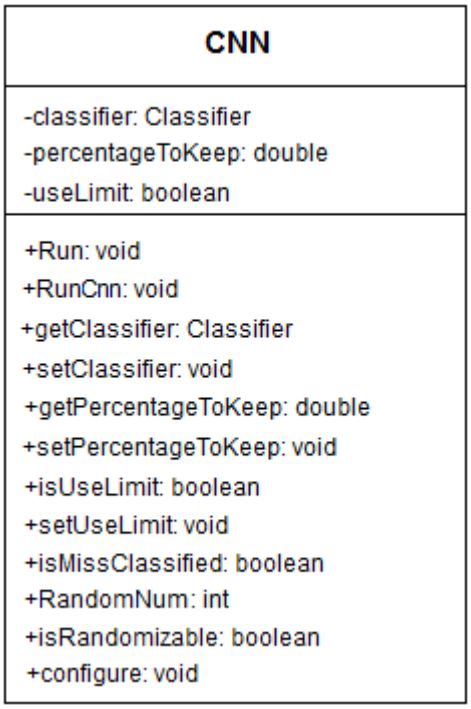

Figura 8: Clase CNN

9.2.3 Paquete Utils

En esta sección se van a detallar todas las clases que se encuentran dentro del paquete Utils.

9.2.3.1 Clase WriteToCSV

- Nombre: WriteToCSV
- Descripción: esta clase permite escribir líneas en un archivo CSV. Se utilizará para escribir los datos de los experimentos realizados en el archivo de salida.
- Variables miembro:
  - DEFAULT_SEPARATOR = objeto de tipo char para separar las diferentes celdas/columnas.
- Métodos de la clase:
  - WriteLine: escribirá una linea en el archivo indicado.
  - followCSVformat: establece el formato de escritura.

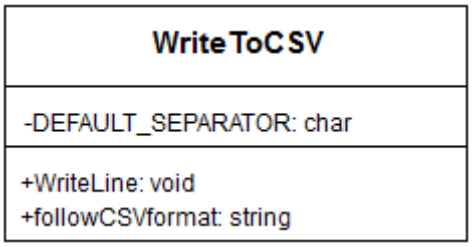

Figura 9: Clase WriteToCSV

Capítulo 10. Pruebas

En este apartado se van a realizar las pruebas de software, cuya función es poder verificar que se cumplen los objetivos que se fijaron en la fase de análisis. Deberán asegurar que la aplicación funciona adecuadamente, siendo por tanto una parte muy importante a la hora de desarrollar un software, derivando en los siguientes objetivos:

- Detectar los posibles fallos en el sistema, para subsanarlos antes de su entrega.

- Comprobar y validar la satisfacción de los requerimientos condicionados a través de la sección 7.2 (Requisitos funcionales).

Se llevarán a cabo una serie de pruebas clasificadas como:

- Pruebas de caja blanca o estructurales: se encargan de la secuencia interna de acciones que se ejecutan para cada unidad de cada componente software.

- Pruebas de caja negra o funcionales: su finalidad es confirmar el correcto funcionamiento de los componentes software.

10.1 Enfoque estructural

En paralelo a la fase de codificación, se realizaron pruebas a cada elemento que se iba incorporando.

Desde que la aplicación solo era capaz de mostrar de aplicar el algoritmo de selección de instancias CNN, hasta la funcionalidad de obtener los resultados del experimento en un fichero de salida externo. También se han

ido resolviendo errores que se iban encontrando a lo largo de la fase de codificación.

10.2 Enfoque funcional

Las pruebas realizadas se exponen a continuación, y se corresponden con los requisitos funcionales que se establecieron.

10.2.1 Casos de prueba

Las tablas 10.1-10.6 recogen los diferentes casos de prueba que se han realizando.

Nombre	**Ejecutar algoritmo de clasificación**
Condiciones	Debe haber un conjunto de datos indicado para ejecutar el algoritmo de clasificación.
Descripción	El algoritmo de clasificación clasifica el conjunto de datos indicado.
Esperado	Se obtienen unos resultados de la clasificación.
Resultado	Correcto.

Tabla 10.1: CU Ejecutar algoritmo de clasificación

Nombre	**Configurar algoritmo de clasificación**
Condiciones	Existe un fichero xml para la introducción de parámetros del algoritmo.
Descripción	El usuario introduce los parámetros para el algoritmo de clasificación.
Esperado	Se espera que los parámetros del algoritmo cambien según los nuevos establecidos.
Resultado	Correcto.

Tabla 10.2: CU Configurar algoritmo de clasificación

Nombre	Ejecutar algoritmo de selección
Condiciones	Debe haber un conjunto de datos indicado para ejecutar el algoritmo de selección.
Descripción	El algoritmo de selección obtiene un subconjunto de datos del conjunto de datos original.
Esperado	Se obtiene un subconjunto del conjunto de datos original.
Resultado	Correcto.

Tabla 10.3: CU Ejecutar algoritmo de selección

Nombre	Configurar algoritmo de selección
Condiciones	Existe un fichero xml para la introducción de parámetros del algoritmo.
Descripción	El usuario introduce los parámetros para el algoritmo de selección.
Esperado	Se espera que los parámetros del algoritmo cambien según los nuevos establecidos.
Resultado	Correcto.

Tabla 10.4: CU Configurar algoritmo de selección

Nombre	Configurar CNN-MI
Condiciones	Existe un fichero xml para la introducción de parámetros del algoritmo.
Descripción	El usuario introduce los parámetros para el algoritmo de selección CNN.
Esperado	Se espera que los parámetros del algoritmo CNN cambien según los nuevos establecidos.
Resultado	Correcto.

Tabla 10.5: CU Configurar algoritmo de selección

Casos de uso	Obtener resultados
Condiciones	Se ha especificado el fichero de salida.
Descripción	Los resultados obtenidos de un experimento son guardados en un fichero de salida.
Esperado	Se guardan los resultados en el fichero de salida.
Resultado	Correcto.

Tabla 10.6: CU Obtener resultados

Capítulo 11. Experimentación

En este apartado se van a definir los experimentos a realizar y los resultados obtenidos por estos.

Los experimentos consistirán en comparar los resultados de aplicar selección de instancias en multi-instancia utilizando un conjunto de datasets y algoritmos de clasificación para resolverlos que emplean tanto todas las instancias, como el conjunto reducido.

11.1 Conjunto de Datos

Para los experimentos disponemos de 5 conjuntos de datos diferentes, que son los siguientes:

- musk1.arff: representa la predicción de actividad de las drogas.
- musk2.arff: representa la predicción de actividad de las drogas.
- mutagenesis3_atoms.arff: contiene todos los átomos de una molécula compuesta.
- mutagenesis3_bonds.arff: contiene todas las tuplas de enlace atómico de una molécula compuesta.
- mutagenesis3_chains.arff: contiene todos los pares adyacentes en enlaces de una molécula compuesta.

Las características de los conjuntos de datos se pueden observar en la Tabla 11.1.

Nombre	Bolsas Positivas	Bolsas Negativas	Bolsas Totales	Atributos	Instancias	Tamaño de Bolsa Medio
Musk1	47	45	92	166	476	5,17
Musk2	39	63	102	166	6598	64,69
Mutagenesis3-Atoms	125	63	188	10	1618	8,61
Mutagenesis3-Bonds	125	63	188	16	3995	21,25
Mutagenesis3-Chains	125	63	188	24	5349	28,45

Tabla 11.1. Conjuntos de Datos

11.2 Configuraciones experimentos

Para cada uno de los conjuntos de datos probaremos un total de 10 clasificadores, utilizando las configuraciones que proporcionan mejores resultados dadas por el trabajo de Zafra y Ventura [23] y que son detalladas en la Tabla 11.2. Se realizará una validación cruzada 10-fold, de donde se sacarán las métricas a estudiar.

Algoritmo	Configuración	
	Clasificador	Método de testeo
SimpleMI	AdaBoost.M1 with PART learning	Arithmetic per-bag mean of each attribute
SimpleMI	PART learning	Arithmetic per-bag mean of each attribute
	Tipo de filtro	
MIOptimalBall	Implements no normalization/standardization	
MDD	Standardize training data	
MIDD	Standardize training data	
MIEMDD	Standardize training data	
	Kernel	MinMax espacio funciones
MISMO	RBF Kernel	Do not make use of MIMinimax feature space
	Clasificador	Método de testeo
MIWrapper	PART learning	Geometric Average of instance-leve class probabilities
MIWrapper	Bagging with PART learning	Geometric Average of instance-leve class

	Kernel	MinMax espacio funciones
		probabilities
MIWrapper	AdaBoost with PART learning	Arithmetic Average of instance-leve class probabilities
MIWrapper	SMO learning with PolyKernel kernel	Arithmetic Average of instance-leve class probabilities
MIWrapper	NaiveBayes learning	Geometric Average of instance-leve class probabilities
	Clasificador	Número máximo de iteraciones
MIBoost	RepTree learning	10
	HDRank	numCiters
CitationKNN	3	3
	MI Suposiciones	Tipo medio para posteriores
MILR	Collective Assumption	Arithmetic Mean

Tabla 11.2. Configuraciones

11.3 Resultados experimentales

En este apartado se van a mostrar los resultados de cada una de las configuraciones para cada uno de los conjuntos de datos.

Las métricas que se van a utilizar para determinar la eficiencia del algoritmo, y que serán representadas en la tabla, son:

- Método reducción: hace referencia al tipo de prueba realizado, bien sea haber aplicado el algoritmo de selección CNN, o haber hecho el estudio con el dataSet completo.

- Exactitud: porcentaje de instancias bien clasificadas. Como todas las instancias serán clasificadas, el resto serán mal clasificadas.

- Kappa: mide el acuerdo o relación entre los valores reales y los predichos. Puede tomar valores en el rango [-1,1]

  - 1 = acuerdo perfecto.

  - 0 = no existe relación.

- -1 = completo desacuerdo.

- F-measure/Valor-F: es la medida de precisión de un test.
  - El valor está en el intervalo [0 1], siendo 0 el peor y 1 el mejor.

- Sensibilidad: representa la capacidad de nuestro clasificador para dar como casos positivos aquellos que realmente lo sean, es decir, proporción de positivos bien clasificados.
  - El valor está en el intervalo [0 1], siendo 0 el peor y 1 el mejor.

- Especificidad: representa la capacidad de nuestro clasificador para dar como casos negativos aquellos que realmente lo sean, es decir, proporción de negativos bien clasificados.
  - El valor está en el intervalo [0 1], siendo 0 el peor y 1 el mejor.

- Tiempo Clasificación: mide en segundos el tiempo que tardó en realizar la clasificación dado un conjunto de datos.

- Tiempo Selección: mide en segundos el tiempo que tardó en realizarse la selección de instancias.

11.3.1 Conjunto de datos Musk1

En este apartado vamos a mostrar los resultados de los experimentos realizados a la base de datos musk1.

De la tabla 11.3 a 11.17, se muestran los resultados experimentales que se han obtenido con cada uno de los clasificadores.

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	40	-0,215	0,478	0,523	0,263	1,054	2,687
DataSet	92	86,956	0,739	0,872	0,872	0,127	3,518	0

Tabla 11.3. Resultados experimentales CitationKNN (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	47,5	0	0/Error	0	1	1,096	2,678
DataSet	92	75,0	0,508	0,716	0,617	0,382	80,227	0

Tabla 11.4. Resultados experimentales MDD (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	42,5	-0,150	0,439	0,428	0,421	2,098	2,650
DataSet	92	81,521	0,629	0,828	0,872	0,127	3,565	0

Tabla 11.5. Resultados experimentales MIBoost (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	47,5	0	0/Error	0	1	1,072	2,696
DataSet	92	85,869	0,717	0,860	0,851	0,148	84,071	0

Tabla 11.6. Resultados experimentales MIDD (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	65	0,287	0,708	0,809	0,473	32,119	2,690
DataSet	92	88,043	0,760	0,886	0,914	0,085	112,091	0

Tabla 11.7. Resultados experimentales MIEMDD (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	52,5	0,050	0,536	0,523	0,526	3,677	2,674
DataSet	92	88,043	0,760	0,884	0,893	0,106	12,344	0

Tabla 11.8. Resultados experimentales MILR (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	72,5	0,452	0,717	0,667	0,789	0,369	2,676
DataSet	92	79,347	0,58	0,786	0,744	0,255	0,502	0

Tabla 11.9. Resultados experimentales MIOptimalBall (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	60	0,181	0,68	0,809	0,368	1,021	2,696
DataSet	92	86,956	0,738	0,877	0,914	0,085	0,755	0

Tabla 11.10. Resultados experimentales MISMO (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	65	0,301	0,65	0,619	0,684	0,519	2,685
DataSet	92	76,086	0,520	0,775	0,808	0,191	1,078	0

Tabla 11.11. Resultados experimentales SimpleMI_AdaBoost (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	65	0,301	0,65	0,619	0,684	0,209	2,678
DataSet	92	79,347	0,586	0,8	0,808	0,191	0,336	0

Tabla 11.12. Resultados experimentales SimpleMI_PART (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	50	0,002	0,5	0,476	0,526	9,269	2,659
DataSet	92	89,130	0,782	0,888	0,851	0,148	21,136	0

Tabla 11.13. Resultados experimentales MIWrapper_AdaBoost (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	52,5	0,045	0,558	0,571	0,473	4,319	2,659
DataSet	92	88,043	0,760	0,886	0,914	0,085	9,131	0

Tabla 11.14. Resultados experimentales MIWrapper_Bagging (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	70	0,392	0,739	0,809	0,578	0,367	2,688
DataSet	92	81,521	0,629	0,824	0,851	0,148	0,350	0

Tabla 11.15. Resultados experimentales MIWrapper_NaiveBayes (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	55	0,106	0,526	0,476	0,631	1,529	2,661
DataSet	92	81,521	0,630	0,813	0,787	0,212	2,562	0

Tabla 11.16. Resultados experimentales MIWrapper_PART (Musk1)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	40	47,5	-0,055	0,511	0,523	0,421	1,016	2,662
DataSet	92	85,869	0,717	0,860	0,851	0,866	0,863	0

Tabla 11.17. Resultados experimentales MIWrapper_SMO (Musk1)

11.3.2 Conjunto de datos Musk2

En este apartado vamos a mostrar los resultados de los experimentos realizados a la base de datos musk2.

De la tabla 11.18 a 11.32, se muestran los resultados experimentales que se han obtenido con cada uno de los clasificadores.

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	45,098	-0,151	0,3	0,3	0,548	152,664	295,555
DataSet	102	80,392	0,588	0,75	0,769	0,23	625,164	0

Tabla 11.18. Resultados experimentales CitationKNN (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	52	57,692	0,089	0,389	0,318	0,767	795,120	590,847
DataSet	102	73,529	0,407	0,597	0,512	0,487	20712	0

Tabla 11.19. Resultados experimentales MDD (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	52,941	0,046	0,454	0,5	0,548	14,417	296,239
DataSet	102	74,509	0,480	0,697	0,769	0,230	24,399	0

Tabla 11.20. Resultados experimentales MIBoost (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	60,784	0	0/E	0	1	32,689	295,116
DataSet	102	82,352	0,611	0,742	0,667	0,334	8825,685	0

Tabla 11.21. Resultados experimentales MIDD (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	70,588	0,428	0,693	0,85	0,612	673,022	295,464
DataSet	102	84,313	0,677	0,809	0,871	0,128	1030,405	0

Tabla 11.22. Resultados experimentales MIEMDD (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	72,549	0,424	0,65	0,65	0,774	429,568	295,521
DataSet	102	80,392	0,592	0,756	0,794	0,205	1122,310	0

Tabla 11.23. Resultados experimentales MILR (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	70,588	0,354	0,571	0,5	0,838	5,355	295,522
DataSet	102	79,411	0,557	0,72	0,692	0,307	14,589	0

Tabla 11.24. Resultados experimentales MIOptimalBall (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	70,588	0,409	0,667	0,75	0,677	15,069	295,348
DataSet	102	84,313	0,674	0,804	0,846	0,153	44,770	0

Tabla 11.25. Resultados experimentales MISMO (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	58,823	0,143	0,487	0,5	0,645	1,033	295,152
DataSet	102	74,509	0,460	0,667	0,667	0,334	1,440	0

Tabla 11.26. Resultados experimentales SimpleMI_AdaBoost (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	60,784	0,162	0,473	0,45	0,709	0,343	295,723
DataSet	102	74,509	0,465	0,675	0,692	0,307	0,363	0

Tabla 11.27. Resultados experimentales SimpleMI_PART (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	54,901	0,078	0,465	0,5	0,580	410,808	295,415
DataSet	102	82,352	0,629	0,775	0,794	0,205	954,626	0

Tabla 11.28. Resultados experimentales MIWrapper_AdaBoost (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	52,941	0,062	0,478	0,55	0,516	276,116	295,158
DataSet	102	81,372	0,607	0,759	0,769	0,230	334,743	0

Tabla 11.29. Resultados experimentales MIWrapper_Bagging (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	56,862	0,155	0,478	0,523	0,263	2,223	295,108
DataSet	102	76,470	0,525	0,727	0,820	0,179	3,794	0

Tabla 11.30. Resultados experimentales MIWrapper_NaiveBayes (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	52,941	0,046	0,454	0,5	0,548	61,933	295,242
DataSet	102	80,392	0,584	0,743	0,743	0,256	72,502	0

Tabla 11.31. Resultados experimentales MIWrapper_PART (Musk2)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	51	64,705	0,272	0,571	0,6	0,677	34,502	295,287
DataSet	102	82,352	0,618	0,756	0,7117	0,282	103,620	0

Tabla 11.32. Resultados experimentales MIWrapper_SMO (Musk2)

11.3.3 Conjunto de datos Mutagenesis3_Atoms

En este apartado vamos a mostrar los resultados de los experimentos realizados a la base de datos mutagenesis3_atoms.

De la tabla 11.33 a 11.47, se muestran los resultados experimentales que se han obtenido con cada uno de los clasificadores.

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	48,780	-0,029	0,603	0,774	0,196	0,977	7,647
DataSet	188	75,0	0,388	0,826	0,896	0,104	2,297	0

Tabla 11.33. Resultados experimentales CitationKNN (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	66,667	0,334	0,664	0,667	0,668	12,382	7,426
DataSet	188	71,808	0,245	0,669	0,718	0,487	16,089	0

Tabla 11.34. Resultados experimentales MDD (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	79,674	0,593	0,796	0,790	0,803	0,936	7,386
DataSet	188	84,042	0,644	0,879	0,872	0,128	0,594	0

Tabla 11.35. Resultados experimentales MIBoost (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	70,731	0,414	0,723	0,758	0,655	36,486	7,495
DataSet	188	72,340	0,343	0,803	0,848	0,152	58,265	00

Tabla 11.36. Resultados experimentales MIDD (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	60,975	0,218	0,636	0,677	0,540	1,968	7,486
DataSet	188	69,680	0,196	0,802	0,928	0,072	2,397	0

Tabla 11.37. Resultados experimentales MIEMDD (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	66,667	0,331	0,705	0,790	0,540	0,373	7,125
DataSet	188	67,021	0,145	0,781	0,889	0,112	0,198	0

Tabla 11.38. Resultados experimentales MILR (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	69,918	0,397	0,713	0,741	0,655	0,709	7,438
DataSet	188	73,936	0,417	0,803	0,8	0,2	1,751	0

Tabla 11.39. Resultados experimentales MIOptimalBall (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	73,170	0,463	0,727	0,709	0,754	1,172	7,485
DataSet	188	67,553	0,206	0,774	0,84	0,16	2,029	0

Tabla 11.40. Resultados experimentales MISMO (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	78,861	0,577	0,779	0,741	0,836	0,335	7,369
DataSet	188	79,255	0,525	0,847	0,864	0,136	0,312	0

Tabla 11.41. Resultados experimentales SimpleMI_AdaBoost (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	74,796	0,496	0,743	0,725	0,770	0,143	7,419
DataSet	188	76,063	0,447	0,824	0,848	0,152	0,073	0

Tabla 11.42. Resultados experimentales SimpleMI_PART (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	69,918	0,398	0,699	0,693	0,704	1,500	7,351
DataSet	188	75,531	0,414	0,827	0,88	0,12	1,030	0

Tabla 11.43. Resultados experimentales MIWrapper_AdaBoost (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	73,983	0,749	0,746	0,758	0,721	1,912	7,503
DataSet	188	76,063	0,419	0,832	0,896	0,104	2,515	0

Tabla 11.44. Resultados experimentales MIWrapper_Bagging (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	47,967	-0,035	0,255	0,177	0,786	0,165	7,466
DataSet	188	66,489	0	0,798	1	0	0,084	0

Tabla 11.45. Resultados experimentales MIWrapper_NaiveBayes (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	69,918	0,398	0,704	0,709	0,688	0,624	7,545
DataSet	188	75	0,388	0,826	0,896	0,104	0,573	0

Tabla 11.46. Resultados experimentales MIWrapper_PART (Mutagenesis3_atoms)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	123	65,040	0,303	0,527	0,387	0,918	0,865	7,462
DataSet	188	66,489	0	0,798	1	0	0,604	0

Tabla 11.47. Resultados experimentales MIWrapper_SMO (Mutagenesis3_atoms)

11.3.4 Conjunto de datos Mutagenesis3_Bonds

En este apartado vamos a mostrar los resultados de los experimentos realizados a la base de datos mutagenesis3_bonds.

De la tabla 11.48 a 11.62, se muestran los resultados experimentales que se han obtenido con cada uno de los clasificadores.

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	41,052	-0,200	0,481	0,490	0,309	4,391	19,592
DataSet	188	76,595	0,444	0,834	0,88	0,12	19,840	0

Tabla 11.48. Resultados experimentales CitationKNN (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	64,210	0,270	0,685	0,698	0,571	25,651	19,462
DataSet	188	71,276	0,288	0,802	0,88	0,12	101,636	0

Tabla 11.49. Resultados experimentales MDD (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	54,736	0,098	0,565	0,528	0,571	1,609	19,477
DataSet	188	81,914	0,594	0,864	0,864	0,136	2,24	0

Tabla 11.50. Resultados experimentales MIBoost (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	69,473	0,367	0,747	0,811	0,547	55,49	19,585
DataSet	188	76,595	0,434	0,835	0,896	0,104	249,077	0

Tabla 11.51. Resultados experimentales MIDD (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	61,052	0,184	0,689	0,773	0,404	5,670	19,489
DataSet	188	73,404	0,329	0,820	0,912	0,088	11,736	0

Tabla 11.52. Resultados experimentales MIEMDD (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	57,894	0,129	0,649	0,698	0,428	1,013	19,438
DataSet	188	70,212	0,213	0,805	0,928	0,072	1,591	0

Tabla 11.53. Resultados experimentales MILR (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	69,473	0,385	0,718	0,698	0,690	1,265	19,542
DataSet	188	77,127	0,488	0827	0,824	0,176	5,802	0

Tabla 11.54. Resultados experimentales MIOptimalBall (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	71,578	0,430	0,732	0,698	0,738	2,847	19,583
DataSet	188	81,382	0,583	0,859	0,856	0,144	5,719	0

Tabla 11.55. Resultados experimentales MISMO (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	73,684	0,462	0,770	0,792	0,667	0,404	19,627
DataSet	188	85,638	0,683	0,889	0,872	0,128	0,292	0

Tabla 11.56. Resultados experimentales SimpleMI_AdaBoost (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	68,421	0,359	0,716	0,716	0,642	0,126	19,473
DataSet	188	85,106	0,668	0,887	0,88	0,12	0,115	0

Tabla 11.57. Resultados experimentales SimpleMI_PART (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	57,894	0,142	0,629	0,641	0,5	4,439	19,527
DataSet	188	81,382	0,573	0,862	0,88	0,12	11,455	0

Tabla 11.58. Resultados experimentales MIWrapper_AdaBoost (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	69,473	0,370	0,743	0,792	0,571	7,303	19,481
DataSet	188	84,042	0,639	0,880	0,889	0,112	16,938	0

Tabla 11.59. Resultados experimentales MIWrapper_Bagging (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	51,578	0,098	0,342	0,226	0,880	0,239	19,626
DataSet	188	67,021	0,121	0,786	0,912	0,088	0,239	0

Tabla 11.60. Resultados experimentales MIWrapper_NaiveBayes (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	65,263	0,287	0,702	0,735	0,547	1,211	19,545
DataSet	188	83,510	0,622	0,889	0,896	0,104	1,585	0

Tabla 11.61. Resultados experimentales MIWrapper_PART (Mutagenesis3_bonds)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	95	55,789	0,0	0,716	1	0	7,078	19,510
DataSet	188	66,489	0,0	0,798	1	0	94,168	0

Tabla 11.62. Resultados experimentales MIWrapper_SMO (Mutagenesis3_bonds)

11.3.5 Conjunto de datos Mutagenesis3_Chains

En este apartado vamos a mostrar los resultados de los experimentos realizados a la base de datos mutagenesis3_chains.

De la tabla 11.63 a 11.77, se muestran los resultados experimentales que se han obtenido con cada uno de los clasificadores.

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	49,579	-0,015	0,642	0,9	0,084	16,776	147,558
DataSet	188	75,0	0,427	0,812	0,832	0,168	52,035	0

Tabla 11.63. Resultados experimentales CitationKNN (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	72,268	0,445	0,717	0,7	0,745	262,916	147,555
DataSet	188	75,0	0,398	0,823	0,88	0,12	659,839	0

Tabla 11.64. Resultados experimentales MDD (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	73,109	0,462	0,719	0,683	0,779	3,163	147,450
DataSet	188	83,510	0,634	0,874	0,864	0,136	4,993	0

Tabla 11.65. Resultados experimentales MIBoost (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	199	72,268	0,444	0,740	0,783	0,661	376,126	147,353
DataSet	188	78,723	0,486	0,850	0,912	0,088	1012,513	0

Tabla 11.66. Resultados experimentales MIDD (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	68,067	0,361	0,667	0,634	0,728	19,221	147,539
DataSet	188	64,361	0,177	0,739	0,76	0,24	23,373	0

Tabla 11.67. Resultados experimentales MIEMDD (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	68,907	0,378	0,689	0,684	0,694	1,816	147,536
DataSet	188	72,640	0,314	0,810	0,888	0,112	3,607	0

Tabla 11.68. Resultados experimentales MILR (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	67,226	0,344	0,660	0,634	0,711	2,910	147,62
DataSet	188	73,936	0,367	0,817	0,88	0,12	9,935	0

Tabla 11.69. Resultados experimentales MIOptimalBall (Mutagenesis3_chains)

Método	Instancias	% Bien	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	77,310	0,546	0,765	0,734	0,813	5,564	147,560
DataSet	188	84,042	0,647	0,878	0,864	0,136	10,258	0

Tabla 11.70. Resultados experimentales MISMO (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	71,428	0,428	0,711	0,7	0,728	0,750	147,270
DataSet	188	81,382	0,577	0,861	0,872	0,128	0,406	0

Tabla 11.71. Resultados experimentales SimpleMI_AdaBoost (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	72,268	0,445	0,717	0,7	0,745	0,210	147,341
DataSet	188	76,595	0,482	0,821	0,808	0,192	0,120	0

Tabla 11.72. Resultados experimentales SimpleMI_PART (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	72,268	0,445	0,713	0,683	0,762	17,453	147,404
DataSet	188	83,510	0,628	0,876	0,88	0,12	38,106	0

Tabla 11.73. Resultados experimentales MIWrapper_AdaBoost (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	73,949	0,479	0,739	0,733	0,745	26,276	147,617
DataSet	188	85,106	0,668	0,887	0,88	0,12	49,624	0

Tabla 11.74. Resultados experimentales MIWrapper_Bagging (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	55,462	0,115	0,208	0,116	1	0,383	147,469
DataSet	188	58,510	0,283	0,551	0,384	0,616	0,272	0

Tabla 11.75. Resultados experimentales MIWrapper_NaiveBayes (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	75,630	0,512	0,752	0,734	0,779	3,613	147,224
DataSet	188	83,510	0,634	0,874	0,864	0,136	4,829	0

Tabla 11.76. Resultados experimentales MIWrapper_PART (Mutagenesis3_chains)

Método	Instancias	Exactitud	Kappa	Fmeasure	Sensibilidad	Especificidad	Tiempo	TiempoSelección
CNN	119	76,470	0,530	0,725	0,616	0,915	8,310	147,718
DataSet	188	67,021	0,020	0,801	1	0	252,070	0

Tabla 11.77. Resultados experimentales MIWrapper_SMO (Mutagenesis3_chains)

11.4 Discusión de resultados

En este apartado se van a desarrollar las conclusiones sacadas tras estudiar los resultados de los experimentos realizados.

En la siguiente gráfica se muestra el % de reducción de los conjuntos de datos obtenidos por el algoritmo de selección CNN para cada uno de los conjuntos de datos:

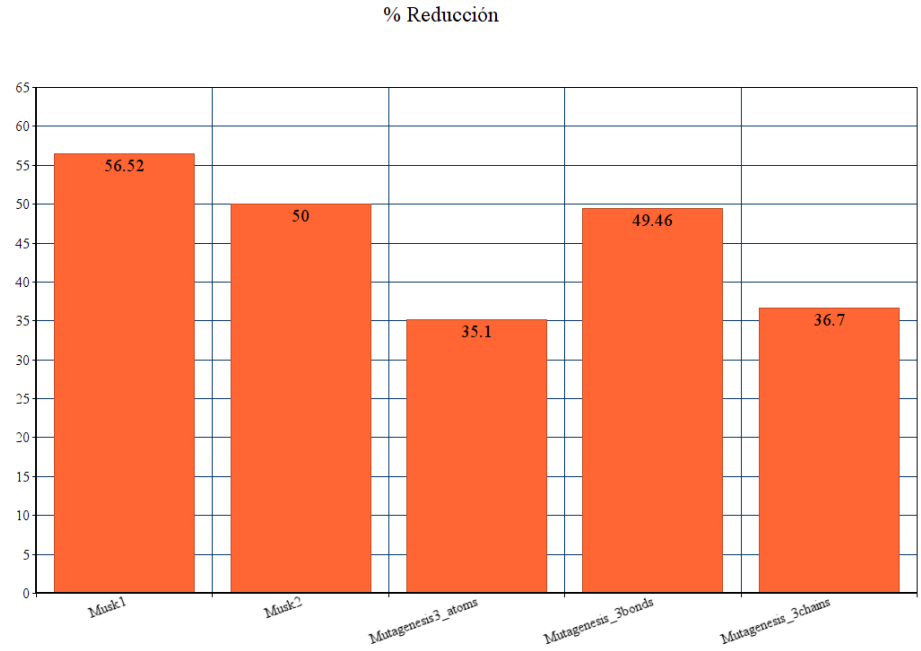

Figura 10: Conjuntos reducción

En las siguientes gráficas se muestra una comparación de los valores de exactitud para los 15 clasificadores según sea el conjunto de datos completo, o el conjunto de datos reducido, para cada uno de los conjuntos de datos.

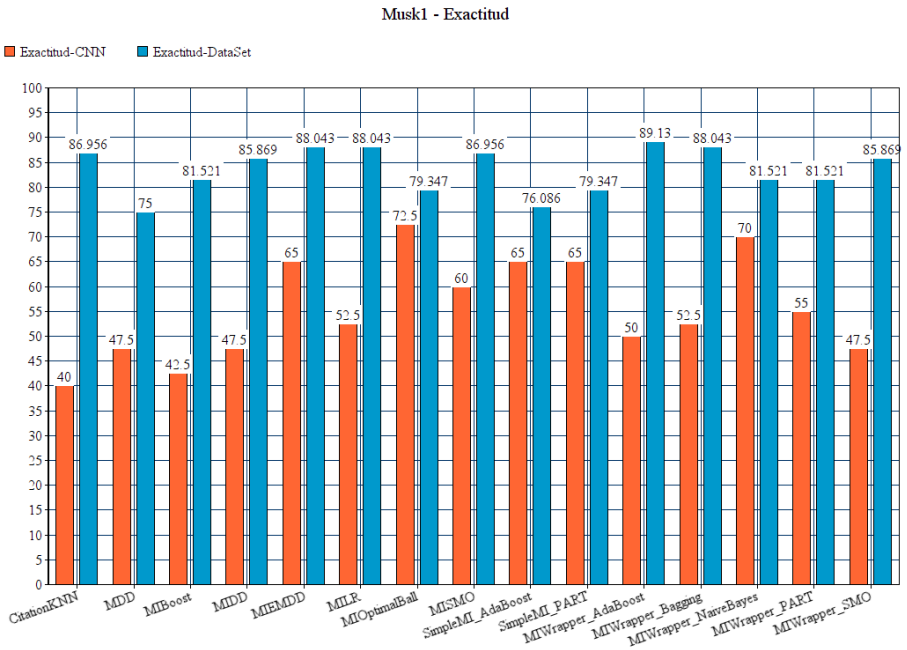

Figura 11: Musk1 exactitud

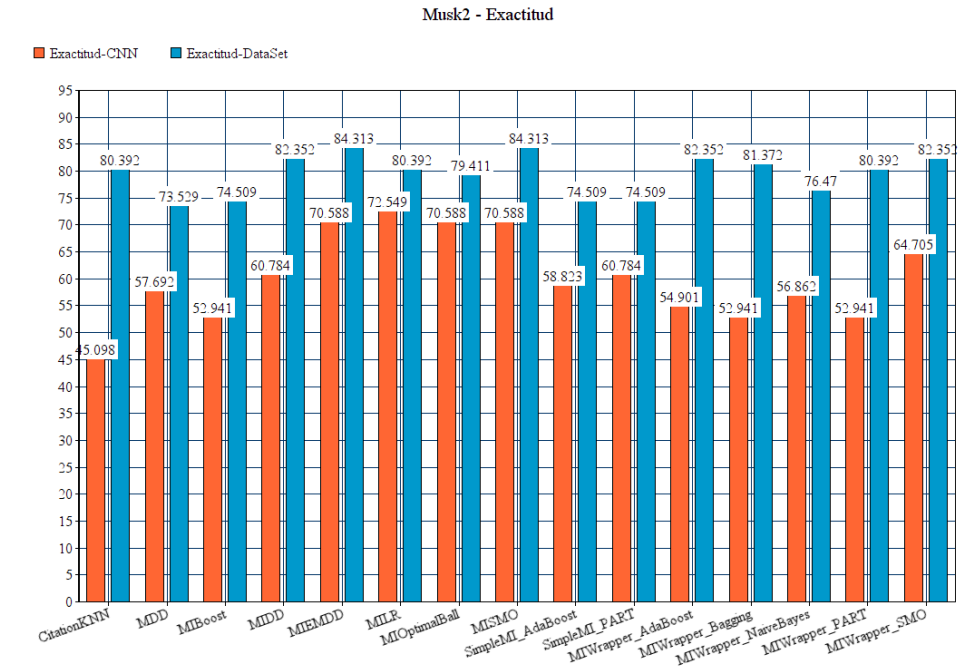

Figura 12: Musk2 exactitud

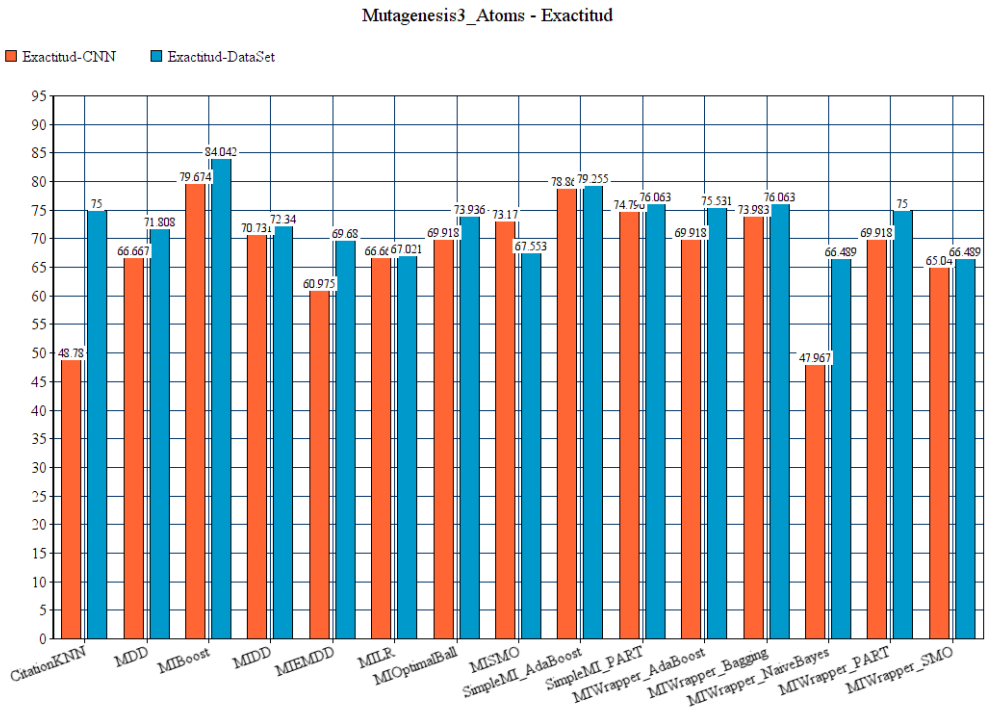

Figura 13: Mutagenesis3_atoms exactitud

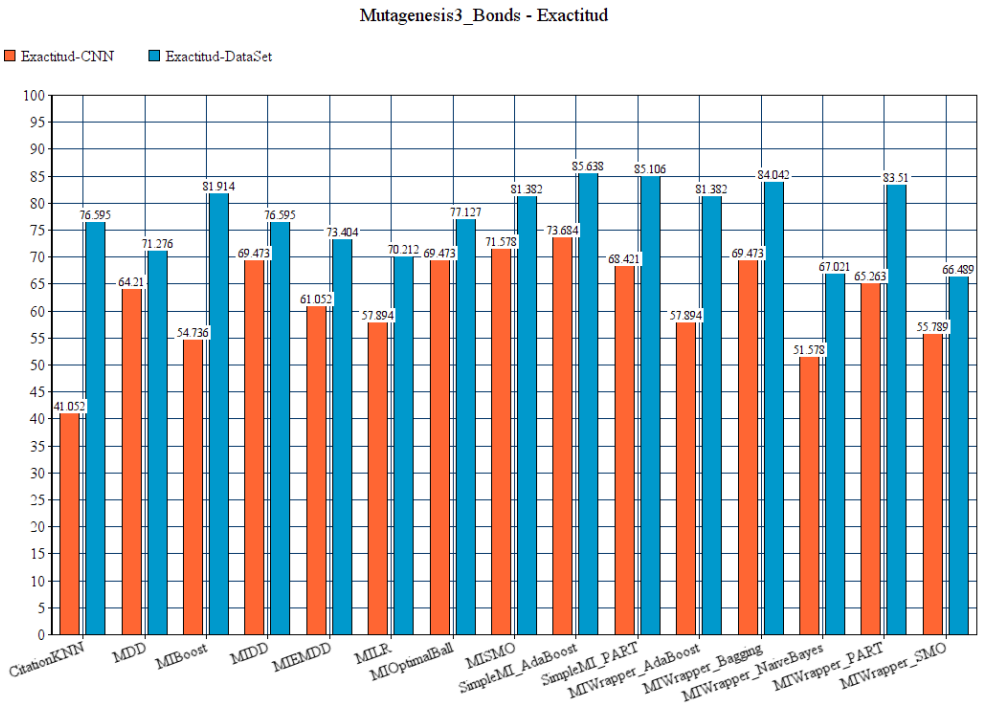

Figura 14: Mutagenesis3_bonds exactitud

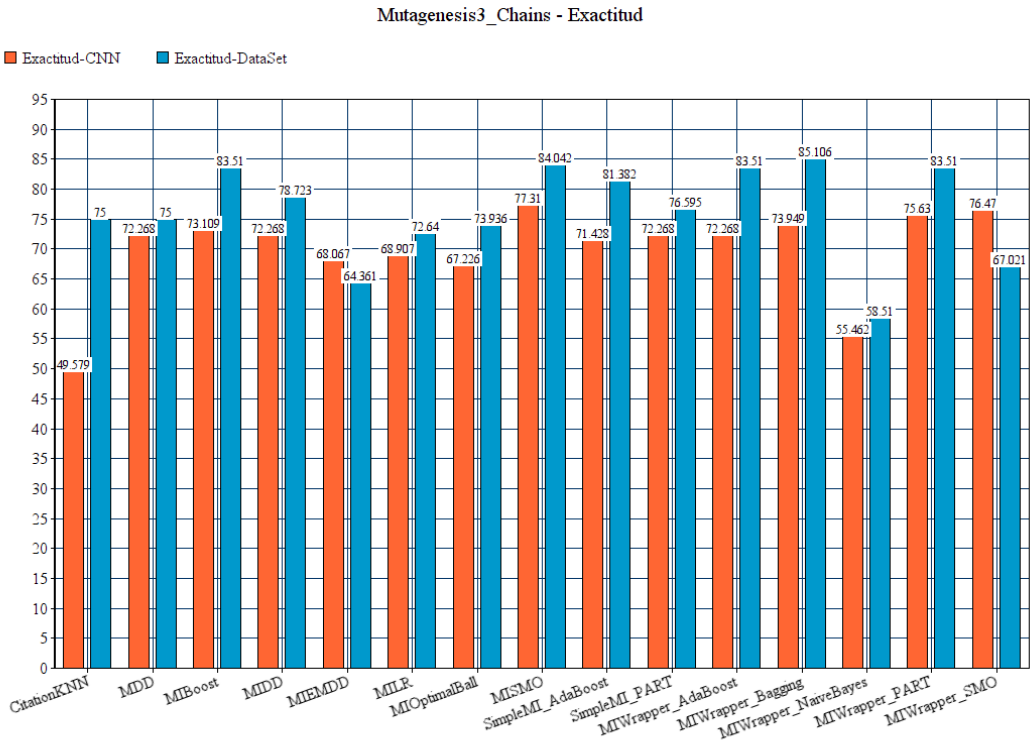

Figura 15: Mutagenesis3_chains exactitud

Se puede observar en las tablas anteriormente mostradas, que la peor relación de exactitud se obtiene al realizar la selección del conjunto de datos Musk1. Se puede ver que, aunque el algoritmo CNN no consigue un subconjunto que mejore los resultados del conjunto original, para los conjuntos de datos mutagenesis3 (atoms, bonds y chains), consiguen aproximarse más que los de musk1 y musk2.

Esto podría deberse por el mayor número de bolsas positivas de los conjuntos de datos mutagenesis3 (atoms, bonds y chains), ya que musk1 y musk2 llegan a presentar sensibilidades de 0, es decir, no han sido capaces de predecir como positivas ninguna de las bolsas, mientras que los conjuntos de datos mutagenesis no llega nunca a 0 (aunque si a valores bajos en algunos casos).

En las siguientes gráficas se muestra una comparación de los valores de f-measure para los 15 clasificadores según sea el conjunto de datos completo, o el conjunto de datos reducido, para cada uno de los conjuntos de datos.

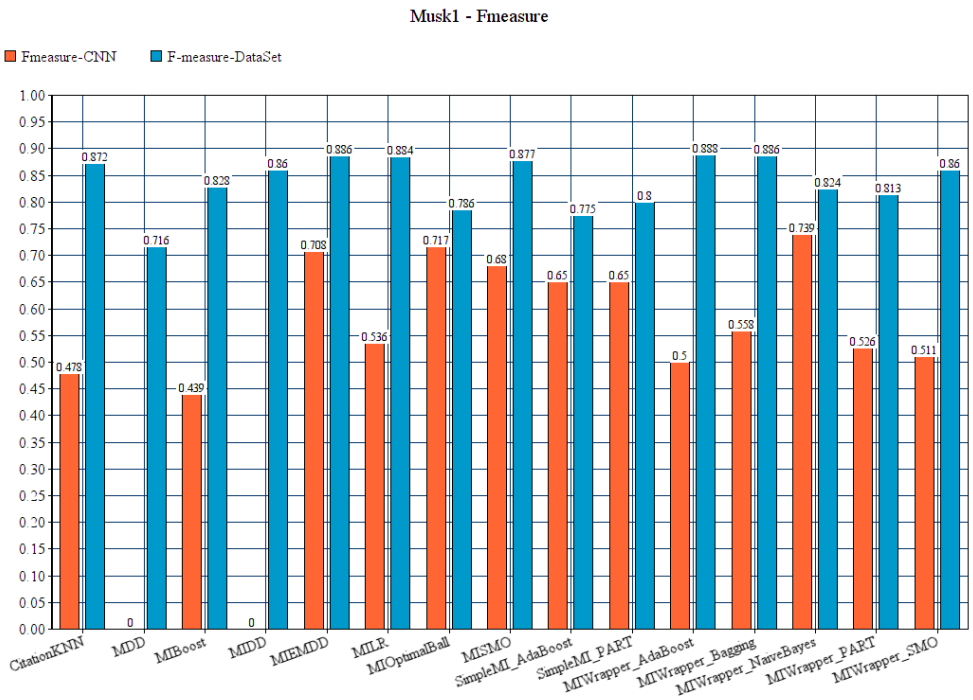

Figura 16: Musk1 fmeasure

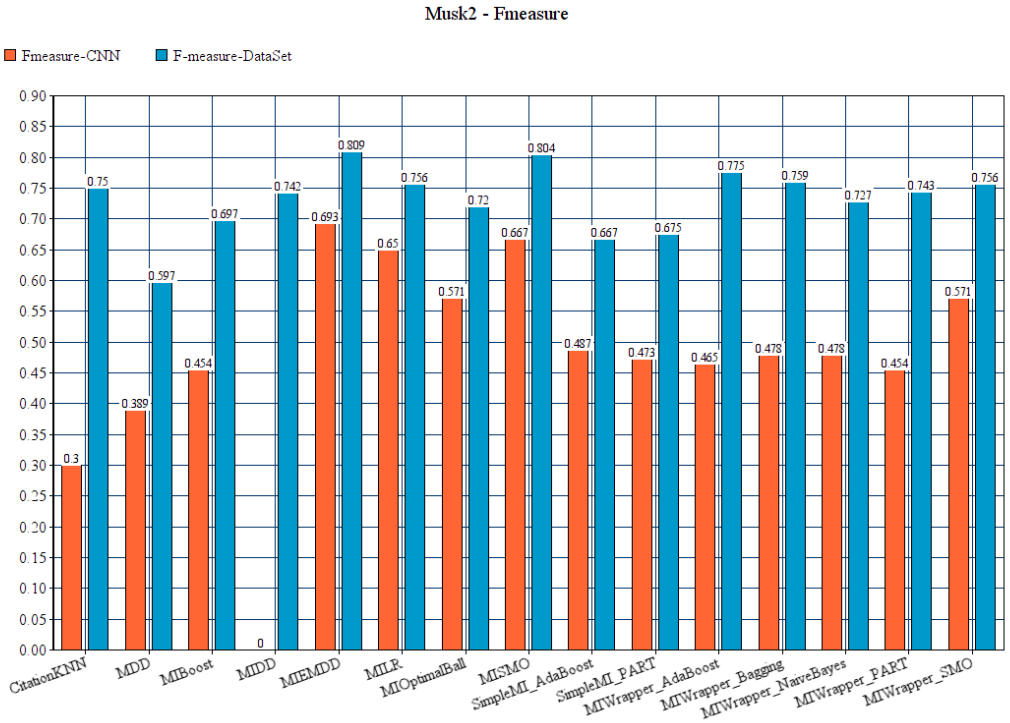

Figura 17: Musk2 fmeasure

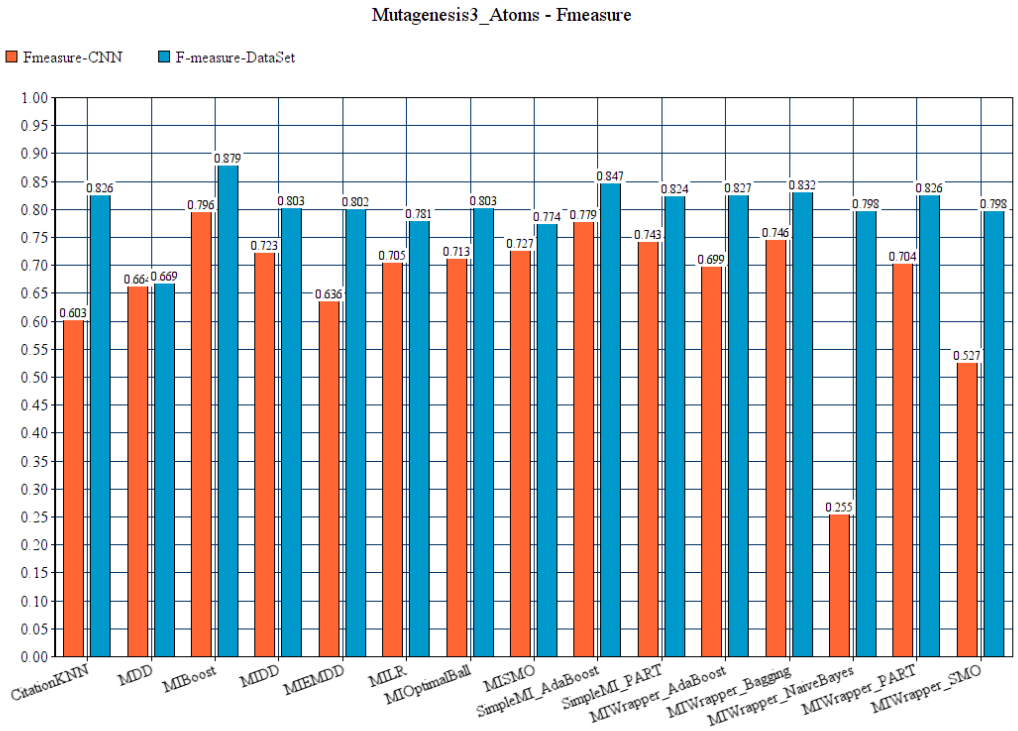

Figura 18: Mutagenesis3_atoms fmeasure

Figura 19: Mutagenesis3_bonds fmeasure

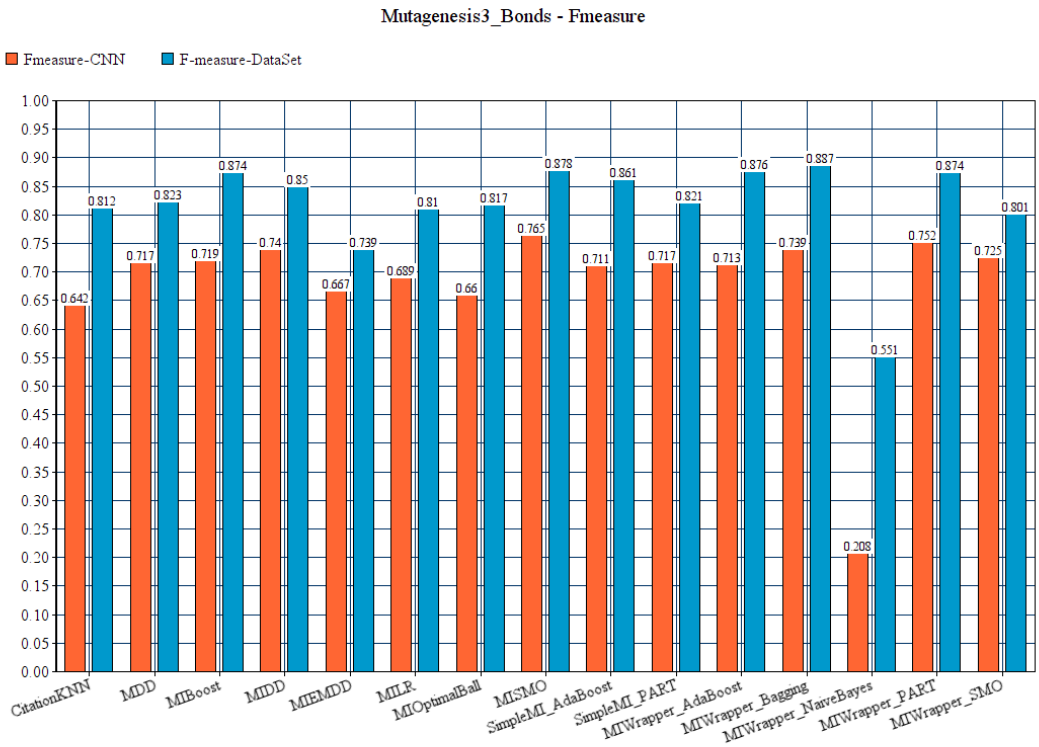

Figura 20: Mutagenesis3_chains fmeasure

Siguen el mismo patrón que los valores de exactitud, pero si podemos concluir que parece que el clasificador MIWrapper con NaiveBayes no es

capaz de obtener unos resultados con alta fiabilidad, ya que incluso para el conjunto de datos completo, presenta valores más bajos que los otros clasificadores para todos los conjuntos de datos

Como factor condicionante para obtener los resultados expuestos, se pueden nombrar dos características:

En el aprendizaje con múltiples instancias, cada patrón es una bolsa con múltiples instancias, donde unas instancias pueden representar el concepto de estudio, y otras no. Estas características, hacen de este aprendizaje, un aprendizaje ambiguo, en el que los algoritmos deben contemplar directamente sus características particulares.

Además, los conjuntos de datos con los que se ha trabajado no tienen muchas bolsas, y tras los resultados obtenidos, parece implicar que dentro de esas bolsas, no existen muchas innecesarias.

Capítulo 12. Conclusiones

En este capítulo se van a exponer las conclusiones a las que se ha llegado una vez finalizadas las diferentes fases de desarrollo de la aplicación. Dichas conclusiones estarán relacionadas con los objetivos que se han planteado inicialmente y con los objetivos después de realizar la fase de pruebas y experimentación. También se va a intentar llegar a una conclusión sobre las futuras mejoras que se podrían desarrollar en un futuro.

12.1 Objetivos alcanzados

Se puede decir, que los objetivos que se plantearon en el capítulo 3 de este manual, se han visto cumplidos de manera satisfactoria.

- El objetivo principal era desarrollar un sistema de clases que funcionase como una librería para selección de instancias en aprendizaje multi-instancia. Esto se ha conseguido con las diferentes clase genéricas que sirven como base para los algoritmos de selección, y que permiten añadir otros algoritmos de selección fácilmente.

- Se ha desarrollado un algoritmo capaz de hacer una selección de instancias aplicando el algoritmo CNN, para un conjunto de datos formado por bolsas de instancias.

- Se ha realizado un estudio experimental bastante amplio para comprobar el funcionamiento del algoritmo CNN-MI en el entorno de aprendizaje con múltiples instancias.

- En cuanto a un objetivo a nivel personal del alumno, se puede comentar el aprendizaje de dos términos nuevos para él, a pesar de haber hecho la especialidad de computación, que son el aprendizaje multi-instancia y la selección de instancias.

12.2 Futuras mejoras

La aplicación cumple con todos los objetivos que se han ido planteando, pero se podrían añadir funcionalidades nuevas y mejoras.

Dichas mejoras podrían ser las siguientes:

- Estudiar otras adaptaciones posibles que implementen la idea del algoritmo CNN para el aprendizaje con múltiples instancias y mejore los resultados obtenidos.

- Mejorar la salida de los datos de los experimentos al archivo csv.

Bibliografía

[1] Machine learning (Aprendizaje automático): C. Hervás-Martínez: Introducción al Aprendizaje Automático, Introducción al Aprendizaje Automático, Grado de Ingeniería Informática, Universidad de Córodoba.

[2] Aprendizaje supervisado: C. Hérvas, J.C Fernández, A.M. Durán: Introducción a Weka: Análisis Descriptivo de la Base de Datos, Introducción al Aprendizaje Automático, Grado de Ingeniería Informática, Universidad de Córdoba.

[3] Clasificación: N. Emilio Garcia Pedrajas: Introducción a la Minería de Datos.

[4] Aprendizaje con multi-instancias: F. Herrera, S. Ventura, R. Bello, C. Commelis, A. Zafra, D. Sánchez Tarragó, S. Vluymans: Múltiple Instance Learning – Foundations and Algorithms. Springer 2016, ISBN 978-3-319-47758-9, pp 1-233.

[5] Selección de instancias: R. Aler Mur: Aprendizaje Automático para el Análisis de Datos, Grado en Estadística y Empresa, Universidad Carlos III de Madrid.

[6] Algoritmo CNN: F. Angiulli: Fast Condensed Nearest Neighbor Rule. ICAR_CNR, Via Pietro Bucci 41C, 87036 Rende (CS), Italy.

[7] Introduction to kNN Clasification and CNN Data Reduction, Oliver Sutton, February 2012.

[8] Xml: https://es.wikipedia.org/wiki/Extensible_Markup_Language (visitado 05-2018).

[9] Librería Weka: http://weka.sourceforge.net/doc.dev/overview-summary.html (visitado 06-2018).

[10] Java: https://www.eclipse.org/downloads/packages/ (visitado 07-2018).

[11] Eclipse: https://www.eclipse.org/ (visitado 07-2018).

[12] IDE: https://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado (visitado 07-2018).

[13] GNU/Linux: https://www.gnu.org/gnu/linux-and-gnu.es.html (visitado 05-2018).

[14] Windows: https://www.microsoft.com/es-es/windows (visitado 05-2018).

[15] Librería MultiinstanceLearning: https://weka.wikispaces.com/Multi-instance+classification (visitado 07-2018)

[16] Librería CitationKNN: http://weka.sourceforge.net/doc.stable/weka/classifiers/mi/CitationKNN.html (visitado 07-2018).

[17] Librería Commons-configuration: https://commons.apache.org/proper/commons-configuration/ (visitado 07-2018).

[18] Libre Office: https://es.libreoffice.org/ (visitado 07-2018).

[19] Draw.io: https://www.draw.io/ (visitado 07-2018).

[20] https://www.generadordegraficos.com/ (visitado 07-2018).

[21] Microsoft Office: https://www.office.com/ (visitado 05-2018).

[22] UML: http://www.uml.org/ (visitado 07-2018).

[23] Zafra, A., & Ventura , S. (2010). GP3-MI: A generic programming algorithm for multiple instance learning. Information Sciences, 180(23), 4496-4513.