INSTITUTO TECNOLÓGICO DE CIUDAD MADERO

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

DOCTORADO EN CIENCIAS DE LA INGENIERÍA

"POR MI PATRIA Y POR MI BIEN"

TESIS

MODELO MULTIAGENTE PARA AJUSTE DE PARÁMETROS DE APLICACIONES PREDICTIVAS EN QUÍMICA

Que para obtener el grado de:

Doctor en Ciencias de la Ingeniería

Presenta:

MCC. José Ángel Villarreal Hernández

D10071371

CVU 861658

Directora:

Dra. María Lucila Morales Rodríguez

CVU 211781

Co-Director:

Dr. Nelson Rangel Valdez

Ciudad Madero, Tamaulipas. Mayo 2025.

Educación
Secretaría de Educación Pública

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Ciudad Madero
Subdirección Académica
División de Estudios de Posgrado e Investigación

Ciudad Madero, Tamaulipas, 14/Marzo/2025

Oficio No. U10.014/2025
ASUNTO: Autorización de Impresión

C. JOSÉ ÁNGEL VILLARREAL HERNÁNDEZ
No. DE CONTROL D10071371
P R E S E N T E

Me es grato comunicarle que después de la revisión realizada por el Jurado designado para su Examen de Grado del Doctorado en Ciencias de la Ingeniería, el cual está integrado por los siguientes catedráticos:

PRESIDENTE :	DRA.	MARÍA LUCILA MORALES RODRÍGUEZ
SECRETARIO:	DR.	NELSON RANGEL VALDEZ
VOCAL 1:	DRA.	NOHRA VIOLETA GALLARDO RIVAS
VOCAL 2:	DRA.	LAURA CRUZ REYES
VOCAL 3:	DRA.	ANA MARÍA MENDOZA MARTÍNEZ
SUPLENTE:	DR.	ULISES PÁRAMO GARCÍA

Se acordó autorizar la impresión de su tesis titulada:

"MODELO MULTIAGENTE PARA AJUSTE DE PARÁMETROS DE APLICACIONES PREDICTIVAS EN QUÍMICA"

Es muy satisfactorio para esta División compartir con Usted el logro de esta meta. Espero que continúe con éxito su desarrollo profesional y dedique su experiencia e inteligencia en beneficio de México.

A T E N T A M E N T E
Excelencia en Educación Tecnológica.
"Por Mi Patria y Por Mi Bien" ®

RAFAEL CASTILLO GUTIÉRREZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

EDUCACIÓN
INSTITUTO TECNOLÓGICO DE CIUDAD MADERO
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

ccp. Archivo
RCG/NRV/jar

Índice general

Índice de Tablas

Índice de Figuras

Índice de Algoritmos

Modelo Multiagente para Ajuste de Parámetros de Aplicaciones Predictivas en Química

Resumen

En la actualidad los expertos en química utilizan técnicas de predicción computacional para orientarse hacia experimentos de laboratorio exitosos. Sin embargo, dado que no están entrenados en ciencias computacionales el proceso de elegir una técnica de pronóstico y configurar sus opciones adecuadamente resulta complicado. Estas dificultades abonan a los costes de materiales, horas humanas y tiempo de ocupación de instalaciones especializadas. Un escenario de solución común es aquel en el que un experto en química consulta a un grupo de expertos en computación para elegir sobre los diferentes puntos de decisión involucrados en la configuración de una tarea de pronóstico. Se analizó el razonamiento de humanos expertos en computación en la configuración de parámetros para técnicas de minería de datos y aprendizaje automático, y las técnicas para lograr estas configuraciones. A partir del análisis del escenario de solución, se propone una arquitectura que hace analogía del escenario y refleja los aspectos clave: el conocimiento y experiencia de los expertos en computación, los compromisos asumidos por estos expertos para abordar la tarea de pronóstico, y la discusión que estos expertos tendrían que realizar para llegar a un consenso. La implementación de la arquitectura para la configuración de parámetros proporcionará al experto en química una herramienta que produce propuestas de solución que cubren las diversas etapas de una tarea de pronóstico utilizando decisiones consensuadas por parte de agentes inteligentes y un enfoque de aprendizaje automático.

(**Palabras clave:** Pronóstico químico, Ajuste de parámetros, Negociación multiagente, Arquitectura DBI)

Multi-Agent Model for Parameter Tuning in Predictive Applications in Chemistry

Abstract

Nowadays, experts in chemistry use computational prediction techniques to guide successful laboratory experiments. However, since they are not trained in computer science, the process of selecting a forecasting technique and configuring its parameters properly can be challenging. These difficulties contribute to the costs of materials, human labor, and the occupation time of specialized facilities. A common solution scenario involves a chemistry expert consulting with a group of computer science experts to make decisions regarding the various configuration options involved in a forecasting task. This work analyzes the reasoning processes of computer science experts when configuring parameters for data mining and machine learning techniques, as well as the methods they use to perform such configurations. Based on the analysis of the solution scenario, an architecture is proposed that mirrors this situation and captures its key aspects: the knowledge and experience of the computing experts, the commitments they make when engaging with the forecasting task, and the deliberations they would need to carry out to reach a consensus. The implementation of this architecture for parameter tuning provides the chemistry expert with a tool that generates solution proposals covering the various stages of a forecasting task, through consensus-based decisions made by intelligent agents and a machine learning approach.

(**Keywords:** Chemical forecasting, Parameter tuning, Multi-agent negotiation, DBI architecture)

CAPÍTULO 1

Introducción

La química computacional es una disciplina que utiliza métodos matemáticos y computacionales para estudiar la estructura, propiedades y reactividad de sistemas químicos, permitiendo a los investigadores obtener información valiosa sin recurrir exclusivamente a experimentos de laboratorio [E. San Fabián, 2020]. Entre sus enfoques tradicionales se encuentran métodos ab initio, mecánica molecular y mecánica cuántica, los cuales han sido implementados en diversas aplicaciones de software [Frisch et al., 2016, Young, 2002, Gordon and Schmidt, 2005, Barca et al., 2020].

En años recientes, técnicas provenientes de otras ramas de la computación, como la minería de datos y el aprendizaje máquina, han comenzado a integrarse a la química computacional, con aplicaciones en el diseño de materiales, predicción de propiedades fisicoquímicas y análisis de interacciones moleculares [Hao Li et al., 2019, Goh et al., 2017]. Estas técnicas

permiten desarrollar modelos predictivos a partir de datos experimentales o simulados, como por ejemplo, el uso de *K Nearest Neighbors* (KNN) para anticipar comportamientos químicos como la solubilidad o la formación de geles en mezclas binarias.

Sin embargo, el uso efectivo de estos modelos requiere conocimientos especializados en ciencias computacionales, lo que representa una barrera para muchos expertos en química. Aunque existen herramientas como WEKA [Frank et al., 2005] y Orange [Demšar et al., 2013] que facilitan la aplicación de algoritmos sin necesidad de programar, estas plataformas no asisten al usuario en tareas clave como la selección de parámetros o la preparación adecuada de los conjuntos de datos.

Este trabajo propone un sistema basado en múltiples agentes que automatiza la configuración de técnicas de aprendizaje máquina para tareas de predicción en el contexto de la química computacional. El sistema actúa como un asistente inteligente que recomienda configuraciones específicas (por ejemplo, el valor de K en KNN o la transformación previa del conjunto de datos) de acuerdo con la situación química a estudiar. A través de deliberaciones entre agentes de software que representan diferentes criterios (como precisión, interpretabilidad o robustez), se busca facilitar el uso de modelos predictivos a investigadores sin formación en informática, promoviendo una integración más efectiva del aprendizaje automático en la práctica química.

1.1. Planteamiento del problema

Las técnicas de minería de datos y de aprendizaje máquina se han empleado en una variedad de problemas en ingeniería, tanto de las ciencias químicas como ciencias de materiales [Goh et al., 2017]. Estas técnicas computacionales requieren la configuración de distintas opciones

de trabajo llamadas parámetros. Por ejemplo, el algoritmo *K Nearest Neighbors*(KNN) debe tener un número *K* de vecinos cercanos a considerar, para las técnicas de árboles de decisiones debe elegir opciones de división del árbol y poda de las ramas, o en las redes neuronales artificiales elegir el número de las capas ocultas a usar.

El ajustar los parámetros de un algoritmo es un problema de optimización que estudia los efectos de los parámetros de un algoritmo sobre los resultados que arroja y como alterarlos para encontrar los resultados que se requieren. La dificultad de atender el ajuste de parámetros radica en que las relaciones entre las variables implicadas no siempre es clara. Para el algoritmo KNN sería la relación entre el parámetro *K* y características del dataset de entrenamiento como la magnitud de registros, cantidad de columnas, o el número de miembros de las clases. Si bien existen casos en que la relación de variables implicadas puede ser compartida entre algoritmos similares, como en la familia de algoritmos KNN, algoritmos con menor parentesco tendrán menos elementos en común.

La dificultad de encontrar configuraciones adecuadas puede crecer con la cantidad de procedimientos, parámetros de trabajo, y cálculos derivados de analizar el dataset de entrenamiento. A las consideraciones antes mencionadas se agrega la situación de adaptar las configuraciones a problemas específicos de un área como los investigados en química. En el área de química es común que los cuerpos de datos producidos estén ligados al problema específico del investigador. De modo que la descripción de sustancias puede contener datos no procesables por un algoritmo, o que resulten poco determinantes para una tarea de pronóstico. De igual manera puede ocurrir que los registros no sean representativos de la realidad en un ámbito mas general, lo que afecta el funcionamiento de algoritmos basados en ejemplos.

En este trabajo de tesis se plantea llegar a configuraciones de algoritmo adecuadas simulando la toma de decisiones de expertos en computación. Para automatizar las deliberaciones de

expertos en computación acerca de una tarea de pronóstico es necesario identificar y plasmar los distintos enfoques que entran a discusión. También se debe modelar el proceso de deliberación, y generar el conocimiento que permita el razonamiento automático para este proceso. El paradigma de agentes provee de los elementos adecuados para realizar esta tarea, particularmente la teoría de negociación multiagente. Un enfoque de negociación entre agentes sobre el problema de ajuste de parámetros puede beneficiarse del análisis de una misma situación desde perspectivas distintas y la acumulación de experiencias.

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar un sistema multiagente para el ajuste automático de parámetros de técnicas de minería de datos y aprendizaje máquina con aplicación dentro del área química.

1.2.2. Objetivos específicos

- Analizar y seleccionar técnicas de minería de datos y aprendizaje máquina con aplicación en problemas dentro del área de química.
- Diseñar una metodología de evaluación del desempeño de técnicas minería de datos y aprendizaje máquina para predicción.
- Desarrollar un sistema multiagente para el proceso de recomendación de parámetros en algoritmos predictivos.

1.3. Preguntas de investigación

La creación de un sistema de ajuste de parámetros que cumpla el objetivo planteado requiere de la construcción de conocimiento sobre como están constituidos los algoritmos y como son afectados por los parámetros. Además, se requiere un modelo de ajuste de parámetros que permita a un grupo de agentes adquirir el conocimiento necesario para discutir y configurar los algoritmos. En concordancia con lo anterior se formulan las siguientes preguntas:

¿Que características tienen las técnicas de minería de datos y aprendizaje máquina empleadas para resolver problemas mediante predicciones o pronósticos en el área química?

¿Que relaciones existen entre las distintas configuraciones paramétricas y las técnicas de minería de datos y aprendizaje máquina seleccionadas?

¿Cómo pueden ajustarse los parámetros de las técnicas de minería de datos y aprendizaje máquina seleccionadas para adecuarlas a problemas específicos?

¿Se puede automatizar y alcanzar una mejora significativa en el desempeño para el caso de estudio aplicando ajuste de parámetros producido por un enfoque de sistema multiagente?

1.4. Hipótesis

En correspondencia a los objetivos y preguntas planteadas se formulan las hipótesis centrales para el proyecto de investigación. Primero se relaciona el dominio de las aplicaciones de las técnicas de minería de datos y aprendizaje máquina con un problema de optimización, después se traslada esa relación al paradigma multiagente.

Hipótesis 1 Dado que el ajuste de parámetros es un problema de optimización es posible encontrar configuraciones paramétricas adecuadas para un problema específico.

Hipótesis 2 Es posible automatizar el proceso de adecuación de la H1 como un sistema multiagente y provocar una mejora significativa en el desempeño para el caso de estudio.

1.5. Justificación y beneficios

Existen diferentes áreas de oportunidad para la química computacional, entre ellas esta el desempeño y comportamiento de las técnicas de minería de datos y aprendizaje máquina. Se busca automatizar el ajuste de parámetros que utilizan estas técnicas, de modo que puedan ser usadas en problemas específicos efectivamente. Podemos observar que muchas de las aplicaciones actuales en química computacional se desenvuelven utilizando múltiples parámetros de trabajo. Estos parámetros definen la forma en que se ejecutan los algoritmos y técnicas computacionales, de modo que guían su comportamiento en el espacio de soluciones. Por lo general el investigador estima cual configuración de parámetros tiene mejores oportunidades

de producir los efectos deseados, pero sus estimaciones pueden no ser óptimas o tener un grado de fiabilidad poco claro. En el caso de decidir la configuración paramétrica mediante diseño de experimentos la disponibilidad de tiempo o potencia de cómputo puede dirigir al investigador a acotar las experimentaciones computacionales.

Es conveniente conocer con qué parámetros un algoritmo o técnica puede producir los mejores resultados para determinados problemas o circunstancias de este. Como señalan Molina y col. en [Molina et al., 2012], la tarea de ajustar los parámetros con que opera una técnica de minería de datos puede significar altos costos computacionales y existe la posibilidad de trabajar sobre supuestos que sesguen los resultados. El ajuste de parámetros automático es un problema tratado en la literatura con distintos enfoques y técnicas. En nuestra revisión de la literatura no se encontraron trabajos en que se empleara un enfoque de Sistemas Multiagente o la Inteligencia Artificial Distribuida para abordar este problema de optimización en el contexto de química computacional.

La complejidad de resolver estas especificaciones pueden ser propias del problema y estar relacionadas con la forma en que se aborda, requiriendo elegir entre algoritmos, sus parámetros y el cuerpo de datos de trabajo. Estas decisiones requieren de un entrenamiento no esencial para los expertos en química, y es posible que no cuenten con un asesor disponible para estudiar la cuestión. Un sistema que resuelva automáticamente estos puntos de decisión podría proporcionar pronósticos acertados y ahorrar tiempo a los investigadores.

1.6. Alcances y limitaciones

El foco de esta investigación se encuentra en la automatización de la toma de decisiones necesaria para la configuración de técnicas de pronóstico. Su realización requiere estudiar los algoritmos, identificar el efecto de sus parámetros, y plasmarlo de modo computable. Esto requiere de la siguiente infraestructura que permita a los agentes inteligentes deliberar acerca de las configuraciones paramétricas:

- Una colección de técnicas de minería de datos y aprendizaje máquina. Estas técnicas deben pertenecer a una variedad distinta a fin de dar robustez a la colección. La colección se ha acotado a la familia de algoritmos KNN ya que comparten la característica de basarse en ejemplos pero tienen mecanismos de generalización distintos.
- Experiencia acerca de las técnicas seleccionadas y su comportamiento. Se ha optado por producir experimentaciones propias mediante un programa generador.
- Capacidad de razonar acerca de las configuraciones paramétricas. Se propone una estrategia de negociación basada en ejemplos y aprendizaje perezoso.
- Un medio de interacción y comunicación. Se ha identificado al protocolo de ofertas múltiples alternadas como un protocolo de negociaciones que permite igualdad de condiciones a los participantes.
- Una arquitectura integradora que permita a un experto en química realizar una consulta al sistema multiagente, de modo que este ejecute una discusión entre los agentes y estos puedan responder con una configuración paramétrica adecuada.

1.7. Organización de la tesis

La organización del documento es la siguiente. En el capítulo 2 proporciona una base teórica para la investigación. Se describen conceptos esenciales como el ajuste de parámetros, los sistemas multiagentes y la negociación, así como sus aplicaciones en la inteligencia artificial distribuida. Además, se profundiza en el aprendizaje máquina y la minería de datos, con un enfoque en los frameworks específicos para implementar variantes del algoritmo KNN. En el capítulo 3 se realiza una revisión de los trabajos previos relacionados con la investigación. Se enfoca particularmente en la predicción en química computacional, incluyendo el uso de bases de datos químicas, la minería de datos y el aprendizaje máquina en este campo. También se analizan técnicas existentes para el ajuste de parámetros y las herramientas disponibles en el aprendizaje máquina. El capítulo concluye identificando vacíos en la literatura que justifican la relevancia del enfoque propuesto. En el capítulo 4 se presenta el diseño e implementación del modelo multiagente enfocado en la configuración y ajuste de parámetros. Se describen los módulos y su constitución, así como las fases clave del modelo: la fase de entrenamiento, que permite a los agentes aprender de los datos, y la fase de pronóstico, donde los agentes aplican lo aprendido para resolver problemas de predicción. El capítulo 5 documenta los experimentos realizados para validar el modelo propuesto. Se describe el diseño experimental y el entrenamiento del modelo, destacando las configuraciones específicas de las variantes del KNN. Posteriormente, se evalúa el desempeño del modelo en dos aspectos: por caso de estudio y por tasa de diferenciación. El análisis de resultados valida la efectividad del enfoque de negociación multiagente. El capítulo 6 presenta las conclusiones, resumiendo los hallazgos clave y destacando cómo el enfoque basado en agentes inteligentes mejora la configuración de parámetros en escenarios complejos. También se proponen líneas futuras de investigación, como la incorporación de nuevas técnicas de minería de datos y mejoras en la estrategia de negociación. Finalmente, se enumeran las principales contribuciones del trabajo.

CAPÍTULO 2

Marco conceptual

En este capítulo se presentan los contenidos teóricos relacionados a este proyecto, los cuales son el ajuste de parámetros, la negociación multiagente, aprendizaje máquina y la minería de datos. Para darnos una perspectiva acerca de los enfoques que se han usado en ajuste de parámetros se dedica la primera parte de este capítulo a tratar las distintas clases de ajustes de parámetros, momento en que se aplica y las técnicas empleadas. La segunda parte de este capítulo se dedica a comprender los conceptos relacionados con las comunidades de agentes y la interacción negociación. La tercer sección trata los fundamentos de las disciplinas de aprendizaje máquina y la minería de datos, su relación con la ciencia de datos, y algunos elementos valiosos para el desarrollo del proyecto de tesis. La cuarta sección del capítulo profundiza en los conocimientos relacionados a la configuración del algoritmo KNN y algoritmos derivados de este.

2.1. Ajuste de parámetros

El problema de ajuste de parámetros se refiere a la selección adecuada de valores para los parámetros de los algoritmos, esto con el propósito de manipular su comportamiento [Birattari, 2005]. Es una problemática de optimización computacional y es conocido que la eficiencia de los algoritmos se ve afectada significativamente por el valor de sus parámetros. A la combinación de valores particulares para los parámetros de un algoritmo se le llama configuración paramétrica.

Michalewicz y Fogel, en su libro [Michalewicz and Fogel, 2004], proponen una clasificación de los distintos ajustes de parámetros considerando niveles de adaptación, esta se representa en la Figura 2.1. Podemos observar que en un inicio el ajuste de parámetros puede clasificarse por el momento en el cual se realiza el ajuste. Si este ocurre previo a la ejecución del algoritmo se le llama sintonía de parámetros, que se caracteriza por ofrecer una configuración inicial para el algoritmo. Esta configuración puede obtenerse a través de prueba y error manual, a través del diseño de experimentos (*Design Of Experiments*) o a través de algoritmos metaheurísticos para optimizarlas progresivamente (metaevolución).

Por otro lado, está el ajuste de parámetros en tiempo de ejecución, es decir el control de parámetros. Esta clase de ajuste monitoriza cambios desde el estado actual del algoritmo. Este ajuste se pude realizar por un control determinístico en el que la configuración paramétrica se es dada por reglas que no consideran retroalimentación. Caso opuesto al control adaptativo de parámetros donde si existe retroalimentación y esta se usa para realizar un cambio a los parámetros ya sea de sentido o magnitud. En el control autoadaptativo se pueden codificar los parámetros en un algoritmo de búsqueda que les aplica variaciones (mutaciones, recombinaciones), asociando soluciones con su correspondiente configuración paramétrica (por ejemplo,

contenidas en el cromosoma de un algoritmo genético) y evolucionar esos parámetros. Si bien la sintonía y el control se han usado de forma separada e independiente una de la otra, autores como Eiben recomiendan obtener una configuraciones paramétricas iniciales mediante técnicas de sintonía y mantener un control de parámetros [Eiben et al., 1999].

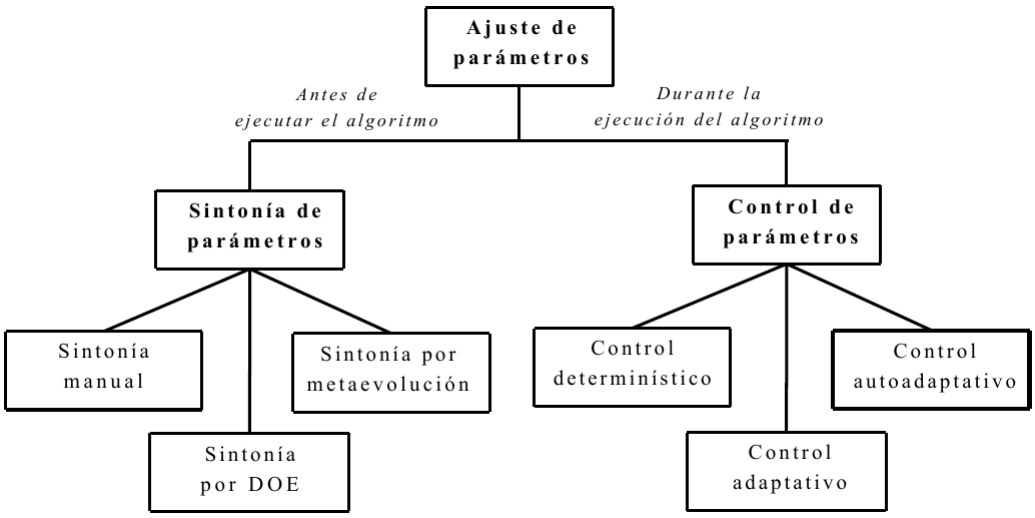

Figura 2.1: Clasificación del ajuste de parámetros de Michalewicz y Fogel [Michalewicz and Fogel, 2004] (versión de Rivera [Rivera Zárate, 2009]).

El ajuste de parámetros estudia los efectos de los parámetros de un algoritmo sobre los resultados que arroja, como alterarlos para encontrar los resultados que se requieren y hacerlos evitando sobreajustes. Los sobreajustes ocurren cuando los algoritmos o sus parámetros se adaptan en sobremanera al conjunto de datos.

Eiben [Eiben et al., 1999] nos presenta las siguientes pautas para comprender las relaciones entre parámetros y sus efectos. (1) Los parámetros tienen distinto peso en los algoritmos por lo que existe una relación entre el parámetro y el impacto que tiene, esta describe la influencia del parámetro en el desempeño del algoritmo. (2) Paralelamente, el problema que se busca resolver tiene características de distinto impacto, es necesario encontrar cuales hacen inadecuados los valores de parámetro y que precisen ser cambiados. (3) En cuanto a los algoritmos, debe estudiarse su comportamiento, sus estados internos, y el rol de cada

parámetro para provocar los efectos específicos deseados. (4) También debe considerarse qué información puede obtenerse ya sea acerca del problema, de la mecánica de aprendizaje del algoritmo, estadísticas, estado de la ejecución del algoritmo. Esta información puede entrar en juego en el proceso de generar nuevas configuraciones paramétricas. Este proyecto de tesis busca implementar una visón de estas pautas y automatizarlas en la medida de lo posible simulando a un grupo de expertos en computación, empleando el paradigma de agentes.

2.2. Sistemas multiagentes y negociación

Para el propósito de simular a un grupo de expertos en computación que discutan la manera de solucionar una tarea de pronóstico se eligió el paradigma de agentes. Bajo este paradigma el proceso de debatir opciones y tomar una decisión por consenso es conocido como negociación.

Existe una variedad de diseños de programas de agentes según el tipo de información utilizada en el proceso de decisión que les fue asignado. Los diseños varían en eficiencia, compacidad y flexibilidad. El diseño apropiado del programa de agente depende de las percepciones, acciones, objetivos y entorno. El proceso de tomar decisiones razonando con conocimiento es fundamental para el diseño exitoso de agentes. Esto significa que la forma en que se representa el conocimiento es importante.

En esta sección se tratan las partes que hacen a un sistema multiagente, el qué y cómo son los propios agentes, su rol en la inteligencia artificial distribuida, su concepción y desarrollo. Estas temáticas están dirigidas a adentrar en la teoría de los Sistemas Multiagentes e introducirnos en los conceptos necesarios para contextualizar la teoría relacionada con la negociación multilateral entre agentes de software.

2.2.1. Sistemas multiagentes en la inteligencia artificial distribuida

Según Iglesias [Iglesias Fernández, 1998], la inteligencia artificial distribuida (IAD) se ha definido como un subcampo de la Inteligencia Artificial que se centra en los comportamientos inteligentes colectivos que son producto de la cooperación de diversas entidades denominadas agentes. Iglesias [Iglesias Fernández, 1998] nos muestra como esta disciplina puede dividirse en dos partes, ver Figura 2.2:

- Solución de problemas distribuidos (SPD): sistemas diseñados centralmente, con la cooperación preconstruida y un problema global a resolver.
- Sistemas Multiagentes (SMA): grupo de agentes heterogéneos que maximizan sus utilidades individuales en un mismo ambiente, incluso si este es competitivo.

Si bien los SPD y SMA tienen componentes en común como la formulación y descomposición de problemas, en la SPD hay un plan prefijado, controlado y coordinado por un miembro centralizador de datos y resultados. Como disciplina, la SPD estudia el cómo dividir problemas en subproblemas para resolverlos de forma separada e independiente. Caso distinto a los SMA, ya que Ferber [Jacques Ferber, 1995] nos dice que en ellos los miembros son autónomos y deciden sobre sus interacciones, tareas, recursos y presentan cooperación incluso cuando tienen objetivos diferentes. Así que, los SMA explotan el potencial de resolución de problemas mediante coordinación de inteligencias individuales y su estudio está enfocado a los comportamientos e interacciones sociales. La propuesta para configurar una aplicación de pronóstico es la interacción de negociación entre múltiples agentes. Estos agentes son autónomos y distribuidos, y pueden actuar por interés propio o ser cooperativos. Para permitirles negociar se debe contar con una infraestructura que especifica protocolos de comunicación e interacción [Weiss, 1999].

Los agentes sociales tienen un ciclo de vida en el que se desenvuelven con autonomía y se relacionan con otros agentes. Para ello el ambiente del SMA debe permitir interacción entre los agentes y proveer herramientas de comunicación. Teniendo en cuenta que los agentes del sistema tienen objetivos propios, se pueden presentar situaciones de conflicto que ellos mismos deberán resolver. Esto se logra mediante comunicación y mecanismos de coordinación que permitan sobrellevar estas situaciones, lograr acuerdos y compartir información necesaria para lograr metas individuales.

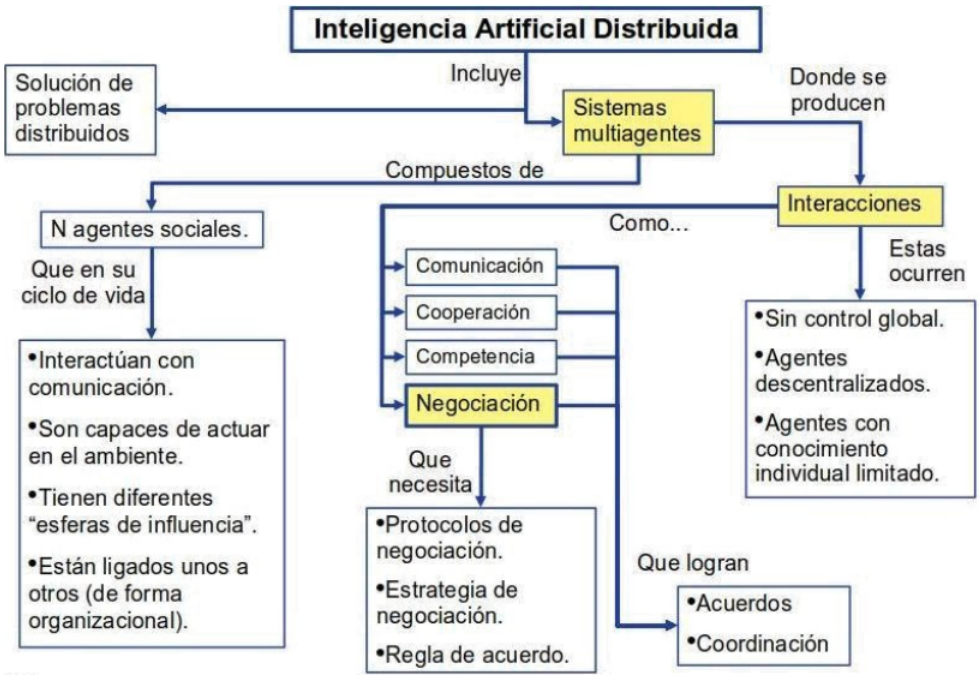

Figura 2.2: Mapa conceptual Sistemas Multiagentes y Negociación (de [Villarreal Hernández, 2019]).

2.2.2. La interacciones de negociación multiagentes

La negociación es una actividad central en la sociedad humana, se utiliza para lograr acuerdos de compra, acuerdos laborales o acuerdos en política. Todas las partes pueden proponer y argumentar sobre la decisión final, y se implican tendencias individuales. Este comportamiento

es el que se espera de un grupo de expertos que deciden como resolver un problema, en nuestro caso la configuración de una aplicación de pronóstico. La negociación permite a un grupo de individuos con distintos intereses encontrar soluciones realistas y satisfactorias. Se entiende como un proceso que trata de lograr un acuerdo, siendo este acuerdo la solución a situaciones en que es necesario alcanzar tratos comerciales, resolver conflictos o para formar alianzas [Baarslag, 2014]. Para la temática que nos ocupa, los negociadores son agentes inteligentes y para dotarles de esta capacidad debemos dirigirnos a la negociación automática. El área de estudio de la negociación automática se dedica a la investigación de las mecánicas y estrategias de negociación, resultando en procedimientos computables usados para la obtención de acuerdos. La negociación crea acuerdos a través del intercambio estructurado de información relevante, para ello se requieren tres elementos [Iglesias Fernández, 1998]: un lenguaje para la comunicación, modelo de decisiones que refleje estrategias y preferencias, y el desarrollo del proceso que muestre la conducta de los participantes.

En una negociación las entidades involucradas mantienen una comunicación, pero ¿cómo se han de comunicar? ¿qué se va a comunicar? El *cómo* es resuelto por el protocolo de negociaciones y el *qué* por el modelo de toma de decisiones del agente.

Protocolo de negociación

El protocolo determina el orden general de las acciones que ocurren durante una negociación, es el conjunto de reglas que rigen la forma en que se lleva a cabo la negociación [Baarslag, 2014]. Los protocolos describen si la negociación ha finalizado, cuál es el acuerdo, qué acciones se pueden hacer en la próxima ronda. Se tiene en cuenta el número de participantes y las acciones válidas de los participantes en cada estado de negociación particular, por ejemplo: qué mensajes pueden ser enviados por cual agente, a cuales agentes puede enviarlos y en qué

etapa. Hindriks menciona el protocolo de ofertas múltiples alternadas [Hindriks et al., 2009], en el cual los agentes siguen una dinámica de mesa redonda que garantiza la participación equitativa (plasmado en la Figura 2.3). Para iniciar el protocolo se definen tanto la situación a negociar como el número de rondas máximas para la negociación. Cada ronda de negociación consiste en dos turnos para cada agente, estos turnos son consecutivos primero todos atienden el turno de ofertas y luego todos atienden el turno de votaciones. Durante el turno de ofertas cada uno de los agentes enterado de la situación de negociación propone una solución a esta situación entonces a través del protocolo los tres agentes comparten con los demás su propuesta original. Al final del turno de oferta todos los agentes han visto una propuesta de los otros agentes, entonces la ronda de negociación avanza al turno de votaciones. En la ronda de votaciones cada agente decide si alguna de las ofertas propuestas por los otros agentes le resulta aceptable, si es así el protocolo cuenta un voto a favor de esa propuesta de lo contrario es un voto en contra. Entonces el protocolo cuenta todos los votos y localiza aquellos que fueron aceptados por toda la mesa de negociación en este caso tres agentes. Si existen ofertas con votos unánimes el protocolo declara un acuerdo y entonces termina retornando los acuerdos.En caso de que después del conteo de votos no se encuentran ofertas que hayan sido aceptadas por todos el protocolo declara un no acuerdo para la ronda, luego verifica si ya se han ejecutado todas las rondas permitidas y decide: si efectivamente se terminaron las rondas permitidas entonces la negociación termina en un no acuerdo; si por otro lado aún no se ha llegado al límite de rondas entonces el protocolo declara una nueva ronda e inicia un nuevo turno de ofertas. Este proceso se debe de repetir hasta encontrar el acuerdo o agotar todas las rondas permitidas Incluso si el estado es el de no acuerdo. Ahora bien, cómo los agentes generan las mencionadas ofertas y votos, depende de su modelo de toma de decisiones.

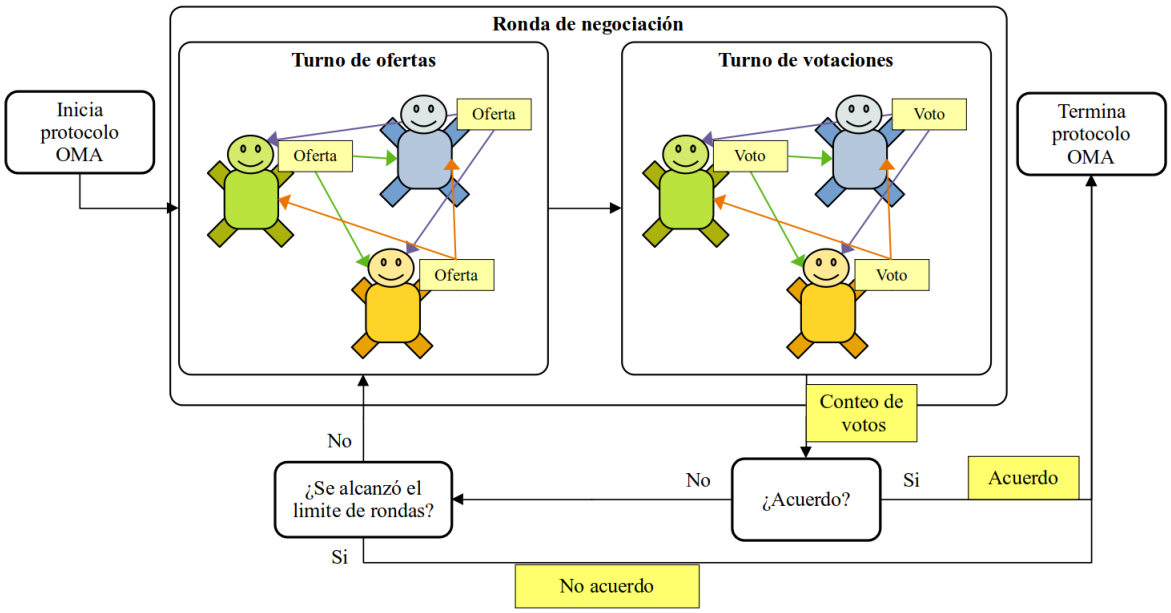

Figura 2.3: Protocolo de ofertas múltiples alternadas.

Modelo de toma de decisiones

Cuando el protocolo es tal que deja espacio para el razonamiento estratégico, el éxito de un agente se determina por la efectividad de su modelo de toma de decisiones [Baarslag, 2014]. El enfoque principal en los modelos de toma de decisiones está en los módulos de razonamiento y las estrategias de negociación que los agentes usan para alcanzar sus objetivos individuales. Durante la negociación, el agente intercambia propuestas con los otros participantes para alcanzar un acuerdo aceptable, que es un contrato que todas las partes negociadoras dan por aceptable. El rango de contratos sobre los que se está negociando (es decir, el conjunto de todos los posibles resultados de negociación) se denomina dominio de negociación. Por supuesto, las propuestas deben enviarse de acuerdo con ciertas reglas y ser válidas de acuerdo con las restricciones establecidas por el protocolo de negociación. Cada agente tiene preferencias sobre el dominio de negociación, que definen el escenario de negociación particular, ver Figura 2.4. Dentro del escenario de negociación existen contratos con diferente valor para

los negociadores. Existe una región en que la que los mejores contratos se balancean entre negociadores, y alejarse de este punto de equilibrio implica que algún negociador perderá mucho más de lo que otro ganaría.

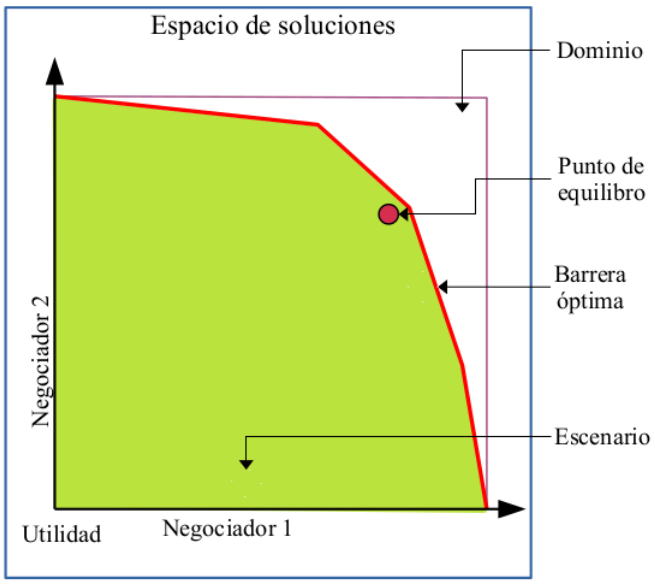

Figura 2.4: Relación del espacio de soluciones en negociación bilateral (de [Villarreal Hernández, 2019]).

2.3. Aprendizaje máquina y minería de datos

Las técnicas de aprendizaje máquina y minería de datos tienen la capacidad de analizar los conjuntos de datos para construir conocimiento, empleando métodos de la inteligencia artificial y la estadística. Estas características son útiles para diseñar esquemas de descubrimiento de conocimiento y predicción, estos esquemas son comúnmente agrupados en el concepto de ciencias de datos. En la Tabla 2.1 podemos comparar las características de estas tres áreas de conocimiento. Esta matriz muestra cómo la ciencia de datos funciona como un campo integrador, con base teórica y herramientas provenientes tanto de la minería de datos como del aprendizaje máquina. Las diferencias metodológicas y de objetivo sugieren que cada

campo puede contribuir a distintas etapas de un proyecto de análisis de datos o modelado predictivo. Así, las técnicas predictivas de las ciencias de datos nacen de la minería de datos o del aprendizaje máquina, y son un subconjunto de estas disciplinas. Se deben estudiar las técnicas predictivas de modo que pueda crearse el conocimiento requerido para operar con ellas, ajustarlas, y diseñar el razonamiento que se implantará en los agentes para desempeñar la tarea de ajustarlas.

Tabla 2.1: Matriz de diferencias y similitudes entre la minería de datos, el aprendizaje máquina y ciencias de datos.

	Minería de datos (MD)	Aprendizaje máquina (AM)	Ciencias de datos
Base teórica	Estrategias estadísticas	Algoritmos de inteligencia artificial	Minería de datos y aprendizaje máquina
Método	Análisis exploratorio	Acumulación de experiencia	Análisis automático
Entradas	Colecciones de datos	Colecciones de datos	Colecciones de datos
Objetivo	Descubrir relaciones	Mejorar por experiencia	Dar significado a los datos
Ejemplo de técnica predictiva	Induction decision trees	K Nearest Neighbors	Máquinas de soporte vectorial

Las siguientes secciones proporcionan el contexto teórico necesario para comprender el propósito y funcionamiento de las técnicas de minería de datos y aprendizaje máquina, así como su relevancia en la construcción del repertorio para los agentes deliberativos en el modelo propuesto.

2.3.1. Minería de datos

La minería de datos es una estrategia para descubrir relaciones intrínsecas y realizar predicciones adecuadas basadas en estadísticas de datos recopilados [Hao Li et al., 2019]. Esto la vuelve un campo de estudio relacionado con las estadísticas computacionales, centrado en el análisis exploratorio [Fehri et al., 2019] con grandes colecciones de datos que pueden contener valiosas tendencias implícitas a descubrir automáticamente. El objetivo de aplicar la minería de datos sobre estas colecciones es predecir o generar las variables que son difíciles de adquirir a partir de otros métodos como la experimentación. Esto se logra a partir de variables fáciles de obtener y usándolas como entrada del proceso de extracción de patrones que una vez definidos son sencillos de procesar [Mitchell, 1997]. Si bien la minería de datos se puede aplicar a cualquier tipo de datos, estos deben ser significativos para la aplicación de destino. Así, la minería de datos es el proceso de descubrir patrones de interés a partir de grandes cantidades de datos y estos patrones interesantes representan el conocimiento [Han et al., 2012].

En general las tareas que se logran mediante la minería de datos se clasifican como descriptivas y predictivas [Han et al., 2012]. Las tareas descriptivas caracterizan las propiedades de los datos en un conjunto de datos objetivo, mientras que las tareas predictivas realizan inducción sobre los datos actuales para lograr predicciones. En estas tareas pueden estar implicadas algunas de las distintas funcionalidades que ofrece la minería de datos:

- Caracterización.
- Discriminación.
- Extracción de patrones.

- Asociaciones y correlaciones frecuentes.
- Clasificación y regresión.
- Análisis de agrupamiento.
- Análisis de valores atípicos.

Al momento de realizar una búsqueda de patrones en colecciones de datos cabe la posibilidad de encontrar los que son muy relevantes y los que son menos, poco o nada importantes. En lo general solo una parte de estos patrones encontrados serían de interés para un usuario específico, pero es posible que aún guarden cierta utilidad [Han et al., 2012].

La aplicación conjunta de la minería de datos y el aprendizaje máquina tiene el potencial de acelerar la optimización de los procesos de ingeniería, ya que presentan características que se complementan para ayudar a llegar a conclusiones de forma abreviada.

2.3.2. Aprendizaje máquina

El aprendizaje máquina es una rama de la inteligencia artificial que puede definirse como el área de estudio dedicada a algoritmos que mejoran automáticamente a través de la experiencia y uso de colecciones de datos [Mitchell, 1997]. En ciencias computacionales cuando un sistema mejora su desempeño después de hacer observaciones sobre el ambiente en el que se encuentra se dice que ha aprendido [Stuart J. Russell et al., 2010].

Así, el propósito y estudio del aprendizaje máquina es el como construir programas de cómputo que automáticamente forman su experiencia y mejoran con ella [Stuart J. Russell et al., 2010].

Se considera que el aprendizaje máquina debe tener la capacidad de adaptarse a nuevas circunstancias a la vez de detectar y extrapolar patrones reconocidos. Es decir que dispone de un mecanismo para analizar el ambiente detectando sus distintas situaciones y otro mecanismo dedicado a la construcción sistemática del conocimiento, formando experiencia. Las mejoras en el aprendizaje del sistema y las técnicas utilizadas para realizarlas dependen de cuatro factores: (1) Qué conocimientos se han logrado previamente, (2) Qué retroalimentación está disponible para analizar y aprender de ella, (3) Qué componente de lo ya aprendido se va a cambiar, (4) Qué representación se utiliza para los datos y el componente. Dado lo anterior, se puede decir que un sistema de aprendizaje máquina es exitoso si puede hacer predicciones o decisiones adecuadas sin estar programados explícitamente para hacerlo.

Existen variedades de problemas que se abordan mediante aprendizaje máquina a los que les acompañan cuerpos de datos de los cuales el sistema de aprendizaje puede alimentarse. Debido al origen, es común que los conjuntos de datos sean grandes, multidimensionales y desordenados [Bishop, 2006]. Por lo que se debe decidir qué datos son necesarios e identificar el método de aprendizaje máquina apropiado para procesar los datos. Algunos autores [Stuart J. Russell et al., 2010] señalan que para algunos problemas hay suficientes datos y modelos lo suficientemente potentes para poder encontrar patrones de aprendizaje óptimos, pero el cálculo para encontrar esos patrones es demasiado complejo y se opta por una aproximación subóptima. También señalan que con la creciente recopilación de datos se van empequeñeciendo las diferencias entre los distintos algoritmos.

El aprendizaje máquina forma conocimiento de manera sistemática y es común que la representación de su experiencia sea en forma de reglas lógicas (por ejemplo, políticas). Suele emplear mecanismos de detección de conocimiento nuevo y de incorporación de este conocimiento que hacen eficiente el proceso de aprendizaje. La detección de conocimiento nuevo requiere entendimiento del conocimiento ganado, mientras que la incorporación de nuevo conocimiento

consiste en ajustar como se relacionan las variables del conocimiento ganado. Este ajuste ocurre sin conocer la forma empírica de las relaciones entre los datos [Hao Li et al., 2019], y podemos considerarlas originales. Así, se genera un mecanismo que podría predecir con precisión las tendencias de los datos. Aunque esto no significa que el aprendizaje logrado por el algoritmo sea el conocimiento exacto de la realidad. Este es una aproximación general que podría abordar problemas de una manera más fácil, sobretodo en dominios poco entendidos o difíciles en que puede no existir el conocimiento necesario para desarrollar algoritmos directos efectivos.

En el campo de estudio del aprendizaje máquina los algoritmos toman distintos enfoques de aprendizaje y pueden clasificarse en varias categorías a la vez. Podemos usar como ejemplo al algoritmo de los K vecinos cercanos [Xindong Wu and Vipin Kumar, 2009] (K-nearest neighbors – KNN).

2.4. Framework para configuración de KNN

El KNN es un método de clasificación basado en ejemplos, y es ampliamente usado por su sencilla implementación y buenos resultados.

Para usar KNN, existen algunos requisitos operativos. Se requieren dos conjuntos de datos; el primero es un conjunto de elementos C que ya han sido clasificados por un experto. El segundo es un conjunto S de elementos sin clasificación conocida. En ambos conjuntos, los elementos se describen con las mismas variables, que se consideran dimensiones espaciales en el algoritmo KNN. También se debe definir una métrica de distancia M a usar, por ejemplo la euclidiana. Para KNN la distancia es la forma de establecer qué tan similar o diferente son

dos elementos ya que indica la diferencia en los valores que les describen, correspondiendo que es más similar si es más cercano. El algoritmo analiza los elementos más similares al que intentamos clasificar, de modo que debe ser definido el número K de elementos a tomar en cuenta. Por consistencia y correcta operación del algoritmo los elementos deben respetar ciertas relaciones: los conjuntos C y S no pueden ser vacíos, las clases presentes en C deben ser como mínimo 2, K puede tomar valores desde 1 y superiores pero no exceder el número de miembros en el conjunto C.

Con las anteriores especificaciones el proceso del algoritmo KNN es el siguiente: se toma el primer elemento sin clasificación S_0 y se opera con él. Con la métrica de distancia M se calcula la distancia entre S_0 y cada uno de los elementos C_i en C. Estas distancias se almacenan en la lista D, asociando la distancia D_{CS} a los elementos que la producen. Una vez calculadas las distancias se procede a formar la vecindad de referencia *neighbors*. El procedimiento *selectNearby* recibe como entrada el tamaño de vecindad K y las distancias D_{CS_0} asociadas al elemento S_0 que buscamos clasificar; entonces *selectNearby* busca las K menores distancias en D_{CS_0}, crea una lista con los elementos C_i asociados a esas menores distancias y los devuelve al procedimiento principal. Con la vecindad *neighbors* definida, se procede a identificar la clase mayoritaria con el procedimiento *majorityClass*: este procedimiento recibe como entrada la lista *neighbors* que consiste en miembros de C los cuales tienen un atributo $C.class$, entonces procede creando contadores asociados a las clases presentes en *neighbors* y registrando sus apariciones; se comparan los contadores, y se retorna al procedimiento principal la clase asociada al contador mayor. Así definimos la clase $S_0.class$ de S_0. El proceso completo se repite para los siguientes miembros S_j de S, cada vez se construye una nueva lista de distancias D_{CS} y una vecindad de referencia *neighbors* diferente para cada elemento S_j sin clasificación, definiendo cada vez la clasificación correspondiente al elemento S_j. Todo el proceso del algoritmo KNN se encuentra plasmado en el Algoritmo 1.

Algorithm 1 Standard KNN.

Require: $C \neq \emptyset$ and $|C.class| \geq 2$ and $S \neq \emptyset$ and $K \geq 1$ and $|C| \geq K$
  for S_j in S **do**
    for C_i in C **do**
      $D_{C_i S_j} \leftarrow distance(M, C_i, S_j)$
    end for
    $neighbors \leftarrow selectNearby(K, D_{CS_j})$
    $S_j.class \leftarrow majorityClass(neighbors)$
  end for

El algoritmo KNN pertenece a la clase de técnicas y algoritmos dedicados a la clasificación. El método usado por el algoritmo es el aprendizaje basado en instancias (o aprendizaje basado en memoria [Stuart J. Russell et al., 2010]) ya que requiere un conjunto de ejemplos conocidos para dar una respuesta [Xindong Wu and Vipin Kumar, 2009]. Al KNN se le identifica como un modelo no paramétrico ya que con cada consulta usa todos los ejemplos de entrenamiento para predecir la clase del nuevo elemento. El KNN también es considerado un algoritmo de aprendizaje perezoso, ya que no procesa el dataset de entrenamiento antes de que se le presente una consulta. Esto traslada la carga de trabajo a la fase de consulta y permite que el dataset de ejemplos se siga actualizando [Han et al., 2012]. En cuanto a complejidad se deben considerar algunas nociones. El KNN no tiene una etapa para crear un modelo de generalizaciones, debido a que requiere que se comparen todos los elementos conocidos con el que se le da a clasificar. Por lo que se le ubica con una complejidad de almacenamiento de $O(n)$ y una complejidad de tiempo $O(n)$ [Xindong Wu and Vipin Kumar, 2009] para la clasificación.

En la literatura se han reportado cambios y adiciones al algoritmo sobre su forma original con la finalidad de mejorar su precisión, lo que ha creado toda una familia de algoritmos basados en la idea original del KNN. Esta familia de algoritmos KNN es el caso de estudio abordado en esta tesis.

Las variantes de KNN introducen modificaciones al algoritmo KNN estándar, afectando una o más de sus partes. Aunque estas contribuciones fueron publicadas con una idea específica en mente, en general se presentan como algoritmos completos sin resaltar con claridad los cambios realizados respecto al KNN original. En esta sección se presenta un framework propio para el análisis y desarrollo de variantes de KNN, inspirado en la observación de que estas variantes comparten una estructura esencial con el algoritmo estándar [Villarreal Hernández et al., 2023]. El objetivo de este framework es automatizar la combinación de configuraciones posibles del algoritmo, facilitando así tanto la recreación de variantes existentes como la generación de nuevas configuraciones. Un framework se entiende como una estructura de software compuesta por componentes intercambiables para la descripción, desarrollo y documentación de una aplicación específica [Sarasty, 2015, Gutiérrez, 2005]. En este caso, el framework permite describir variantes de KNN como selecciones particulares en distintos puntos de decisión. Mediante etapas de configuración, este framework facilita la inclusión o exclusión de módulos de forma flexible, permitiendo implementar variantes conocidas o construir nuevas variantes adaptadas a diferentes contextos de aplicación. Las etapas del framework se definen de la siguiente manera:

- Estimación de variables del dataset: Esta etapa se dedica a evaluar qué variables (dimensiones) se utilizarán para calcular las distancias.
- Reducción de registros del dataset: En esta etapa, los registros que tienen poca contribución al proceso de clasificación KNN se descartan.
- Métricas de cercanía-similitud: En esta etapa se elige el método para calcular la distancia entre puntos.
- Estimación del valor de K: El parámetro K determina el tamaño del vecindario y debe elegirse en esta etapa.

- Políticas de asignación de clases: La política es una técnica heurística que decide una clase resultante para el punto desconocido, dada una determinada vecindad de referencia.

Este framework permite referenciar y reutilizar configuraciones del algoritmo KNN, abarcando técnicas de preprosesamiento del dataset de ejemplos, métodos para elegir le valor K y las políticas de asignación de clase. Para describir una variante de KNN en concreto se requiere elegir una opción específica en una o más de las etapas. En las siguientes secciones se revisan las opciones a utilizar en cada una de las etapas.

2.4.1. Reducción del dataset para el algoritmo KNN

Cuando se le pide al algoritmo KNN que clasifique un nuevo punto, este recorre el dataset de ejemplos para calcular las distancias hacia el nuevo punto. Debido a este comportamiento los cálculos realizados por el KNN crecen con el número de elementos en el dataset de ejemplos y con las dimensiones que los describen. Si bien se considera que un dataset de ejemplos con muchos elementos describe con mayor fidelidad a las clases, es posible modificar el dataset de ejemplos para reducir el numero de registros [Muhammad Ejazuddin Syed, 2014] o de variables-dimensiones [Stuart J. Russell et al., 2010] que lo componen y mantener el desempeño de las clasificaciones.

Estimación de variables del dataset

Clásicamente, el almacenamiento de cuerpos de datos para el procesamiento computacional se realiza en tablas. La convención es que dentro de la tabla, las columnas representan las variables que describen cada registro y cada registro tiene una sola entrada en la tabla. La

estimación de variables se refiere al ejercicio de ponderar la contribución de las variables de un dataset respecto al desempeño del algoritmo. La etapa de estimación de variables del dataset busca optimizar la influencia de estas variables, que en términos del KNN son dimensiones espaciales.

Existen diferentes técnicas para implementar la estimación de variables. Se identifican dos enfoques para este objetivo: ponderación de variables y selección de subconjuntos mínimos [Muhammad Ejazuddin Syed, 2014]. Las técnicas de ponderación de variables establecen la relevancia de las variables asignándoles una puntuación que puede utilizarse como modificador. Las técnicas de pesaje variable incluyen algunas conocidas generalmente como Correlación de Pearson, Chi-cuadrado y Análisis de componentes relevantes [Yeung and Chang, 2006]. Las técnicas de selección de subconjuntos mínimos seleccionan las variables que formarán parte de los conjuntos de datos, lo cual está relacionado con el cálculo de la distancia en el algoritmo KNN. Algunas técnicas de ejemplo para la selección de subconjuntos mínimos pueden ser Cobertura de conjuntos codiciosos, Análisis discriminante lineal y el algoritmo OBLIVION [Langley and Sage, 1994].

Reducción de registros del dataset

Una reducción adecuada de los registros en el dataset de elementos conocidos C puede ahorrar los cálculos requeridos sin sacrificar la precisión de pronóstico. En la etapa de Reducción de registros del dataset se descartan los registros de puntos de baja contribución. Este proceso de reducción debe ocurrir antes de que la implementación de KNN forme el vecindario de referencia.

Shi [Shi, 2012] identifica tres clases de registros-puntos que componen el dataset. La primera clase son los elementos 'prototípicos' que son un subconjunto de C que se reconoce como el

mínimo para que otros puntos se clasifiquen correctamente. La segunda clase son los elementos 'absorbidos' que son un subconjunto de C que se reconocen correctamente con elementos prototípicos. La tercera clase de elementos son los 'outliers', aquellos que no se clasificarían correctamente usando solo los elementos prototípicos. En la literatura encontramos diferentes técnicas relacionadas con el algoritmo KNN que utilizan estos conceptos y pueden ser utilizadas para implementar la etapa de Reducción de registros del dataset.

La condensación de Hart [Hart, 1968] y la edición de Wilson [Wilson, 1972] son técnicas cuyo procedimiento recorre todos los registros en C y una ejecución de KNN los clasifica usando un dataset de entrenamiento temporal T. Deciden la permanencia de cada punto comparando la clase real y la clase obtenida, manteniendo tanto los elementos prototípicos como los atípicos.

La reducción de Gates descrita en [Gates, 1972] es un procedimiento que se basa en la condensación de Hart para producir un dataset reducido de C. Comienza con un dataset condensado, extrae el primer elemento de él e intenta clasificar los puntos de C usando el campo condensado modificado. Si la clasificación es defectuosa, se reembolsa el punto extraído. Esto se repite para todos los puntos del dataset condensado original.

En su artículo [Angiulli, 2005], Angiulli presenta el algoritmo de condensación rápida inspirado en el trabajo de Hart. Su método tiene algunas ventajas sobre la condensación de Hart, por ejemplo, el método de Angiulli no es el orden de revisión de los datos.

En la variante KNN de centroide más cercano [Gou et al., 2012, Mehta et al., 2018], la representación de las clases se simplifica a un solo representante, que es una generación de puntos prototípica. Entonces, el número de posibles vecinos es igual al número de clases.

2.4.2. Métricas de cercanía-similitud

Uno de los elementos clave del KNN es la métrica con la que se obtiene la distancia entre los elementos conocidos y desconocidos. En términos del algoritmo KNN, la distancia representa qué tan similar es un elemento a otro. Esta similitud normalmente se calcula como un valor numérico, como las distancias en el espacio euclidiano.

Con el tiempo se han ido creando técnicas que proponen formas de calificar los casos, manteniendo el principio de que la cercanía debe indicar en qué medida dos puntos son similares y por tanto la probabilidad de pertenecer a la misma clase. Para las métricas de distancia, la cercanía se refiere a la medida en que la diferencia absoluta es cercana a cero. Entre las métricas de distancia más utilizadas para el algoritmo KNN podemos encontrar la euclidiana, la de Manhattan y la de Minkowsky (con las que puedes experimentar variando el parámetro p). Otras métricas utilizadas son Chebyshev (también llamada distancia de ajedrez), Mahalanobis (que mide la distancia entre un punto y una nube de puntos). Para medir la diferencia entre textos se puede utilizar la distancia de Hamming o de Levenshtein. También hay funciones de semejanza. Por lo general, estas funcionan en el rango de $[0,1]$, en el que la similitud de dos puntos x y y son iguales cuando $s(x,y) = 1$ [Metcalf and Casey, 2016]. Una opción común para una función de similitud es la similitud del núcleo gaussiano, que se calcula utilizando la distancia euclidiana. Si lo que necesitas es comparar dos documentos puedes usar la similitud Coseno.

2.4.3. Estimación del valor de K

El algoritmo KNN utiliza un valor de K para formar la vecindad, que son los vecinos de distancia mínima al punto de clase desconocida. La etapa de estimación del valor de K establece este valor para la operación general de la implementación de KNN. Las etapas cuya técnica implica ejecuciones del algoritmo KNN dependen de la elección de K. La etapa de política de asignación de clase requiere esta vecindad de tamaño K, sin embargo, la técnica elegida para esta etapa puede usar este valor indirectamente o no utilizarlo en absoluto.

Algunos problemas relacionados con la elección del número de vecinos de referencia ocurren cuando K es muy bajo, ya que el resultado puede ser sensible a los puntos de ruido. Si hay suficientes elementos conocidos disponibles, los valores más grandes de K se vuelven resistentes al ruido [Xindong Wu and Vipin Kumar, 2009]. Otra situación problemática es que cuando K es muy alto, el vecindario se puede formar con otras clases menos similares que diluyen a la que debería ser la clase elegida o directamente la reemplazan. Las recomendaciones en la literatura suelen manejar un valor de K entre 5 y 9, pero estas no suelen tener en cuenta el tamaño del dataset ni de las clases.

Una forma de simplificar los cálculos necesarios para el funcionamiento del KNN es elegir un valor de $K = 1$. El resultado de elegir este valor es que se eliminan los cálculos relacionados con la política de decisión de clase, cambiando la política de conteo de frecuencia para simplemente asignar la clase del vecino más cercano. Se han diseñado varios métodos de reducción de conjuntos de datos conocidos y algunas métricas de similitud con el caso $K = 1$ en mente. Según Ejazuddin [Muhammad Ejazuddin Syed, 2014] si no se tienen en cuenta las consideraciones anteriores, $K = 1$ da resultados inestables debido a su alta varianza y sensibilidad.

En línea con lo tradicional está la elección del valor de K como la raíz cuadrada del dataset de entrenamiento [Ahmad Basheer Hassanat et al., 2014]. Si bien se ha señalado que puede ser un punto de partida para experimentar con el algoritmo, en retrospectiva, a menudo es un valor alto de K para la vida real e incluso para aplicaciones académicas. Sin embargo, es una forma rápida de proponer un límite superior para los valores de K.

En su estudio [Enas and Choi, 1986], Enas et al. aborda los efectos del valor K para el caso en que uno decide entre dos clases posibles utilizando pesos uniformes. Los autores reportan lineamientos basados en el tamaño de las clases y sus matrices de covarianza, de acuerdo a estas reglas el valor de K puede ser $n^{\frac{2}{8}}$ o $n^{\frac{3}{8}}$.

Otra forma de elegir un valor de K es a través de la experimentación para guiarnos. La validación cruzada [Stone, 1974] es una técnica de evaluación que mide el rendimiento de un proceso dividiendo el dataset clasificados C en dos partes, en un conjunto de entrenamiento y un conjunto de prueba. De esta forma, luego de evaluar los experimentos, podemos tomar los mejores valores de K. Un aspecto a considerar en las experimentaciones es la métrica de distancia empleada, en la Tabla A.1.1 y Tabla A.1.2 exponemos distintas publicaciones que asocian el desempeño del algoritmo KNN, el valor K y la métrica de distancia. Estas publicaciones buscan guiar a la elección del valor K dada una métrica.

2.4.4. Políticas de asignación de clases

La etapa política de asignación de clases es aquella en que se recibe la vecindad de referencia, se le aplica una heurística y se calcula la clase para el punto de clase desconocido. La técnica heurística que implementa esta etapa tiene únicamente la función de extraer una clase del vecindario. Es muy común que las variantes de KNN se presenten como un algoritmo completo.

Debido a esto es posible encontrar que algunas políticas están vinculadas a una técnica o resultado de otra etapa. Por ejemplo, la regla del vecino más cercano [Cover and Hart, 1967] es una variante que, dado un implícito valor de $K = 1$, realiza una asignación directa con el único miembro de la vecindad; esa asignación directa es la política de asignación de clase.

La política de vecinos con pesos [Klaus Hechenbichler and Klaus Schliep, 2004, Tan, 2005], consiste en asignar un peso *w* a la cuenta de cada miembro, en vez de solo contarlo como 1. El concepto nuclear de esta política es dar más representatividad a los vecinos más cercanos sin anular al resto. El modificador *w* puede calcularse con cualquier función, por ejemplo, puede ordenar la vecindad y contar al vecino mas cercano como $1 + 0.09$, al segundo como $1 + 0.08$ y continuar de ese modo. Otra opción para el método de pesaje es elegir una función inversa de distancia, como las métricas de similitud.

En la política de la clase de menor promedio [Zapata-Tapasco et al., 2014, Gou et al., 2012], se agrupan los miembros de la vecindad por su clase, se obtiene la distancia promedio para cada clase presente y se asigna la clase con menor distancia promedio. Es decir, se calcula la similitud con la clase en base a que los puntos seleccionados como vecinos son una referencia de su clase en esa región.

Algunas políticas consideran la opción de no otorgar calificación si no se cumplen ciertas condiciones. En la política de umbral de probabilidad [Dalitz, 2009], el conteo de la clase mayoritaria debe exceder un umbral mínimo U de acuerdo al número total de vecinos seleccionados. También se pueden usar mecánicas de mayoría absoluta, donde $conteoclasemayoritaria > (U + segundaclasemayoritaria)$.

2.4.5. Normalización del framework

En total, se consideran cinco etapas de configuración para el algoritmo KNN: estimación de variables del dataset, reducción de registros del dataset, métricas de cercanía-similitud, estimación del valor de K, políticas de asignación de clases. El orden listado refleja una organización preferente de su configuración. En la Figura 2.5 se aprecia que este orden coincide con el momento de ejecución.

En orden de ejecución tenemos dos momentos temporales: el pre-procesamiento que ocurre fuera del algoritmo, y el procesamiento dentro del algoritmo propiamente. El pre-procesamiento se conforma de las etapas de estimación de variables y reducción de registros, su propósito es mejorar el desempeño del algoritmo desde el dataset de entrenamiento. Las otras tres etapas de configuración se utilizan dentro del algoritmo KNN y están presentes desde el algoritmo KNN estándar. Esta distinción nos indica su rol en la composición de una implementación del algoritmo KNN. En la Figura 2.6 se señala esta relación de composición en la configuración del algoritmo KNN. Las clases abstractas cuya conexión es un rombo blanco tienen una relación de composición débil, es decir el elemento puede o no formar parte de la configuración. En cambio las clases abstractas donde su conexión es un rombo negro tienen una relación de composición fuerte, es decir el elemento debe estar definido y formar parte de la configuración. Las clases abstractas están definidas con un método publico *Constructor* donde captan la entidad requerida para operar, y un método publico *get* con el cual propagan el resultado de su proceso interno. En seguida se definen algunos ejemplos de clases reales que se derivan de la clase abstracta, estas clases reales serían las implementaciones funcionales que resuelven la etapa indicada por la clase abstracta.

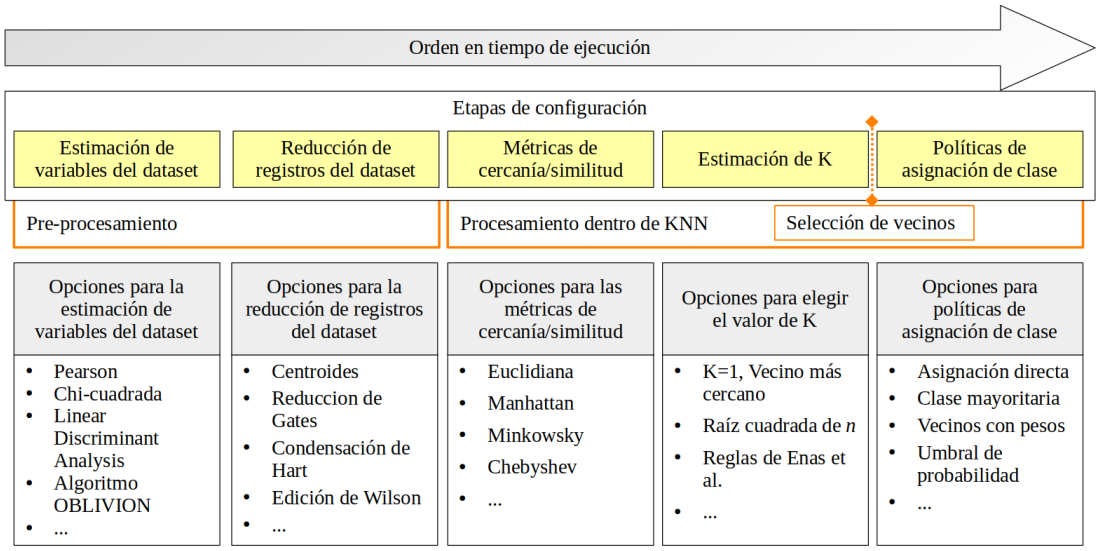

Figura 2.5: Esquema general de las etapas de configuración para implementaciones de KNN.

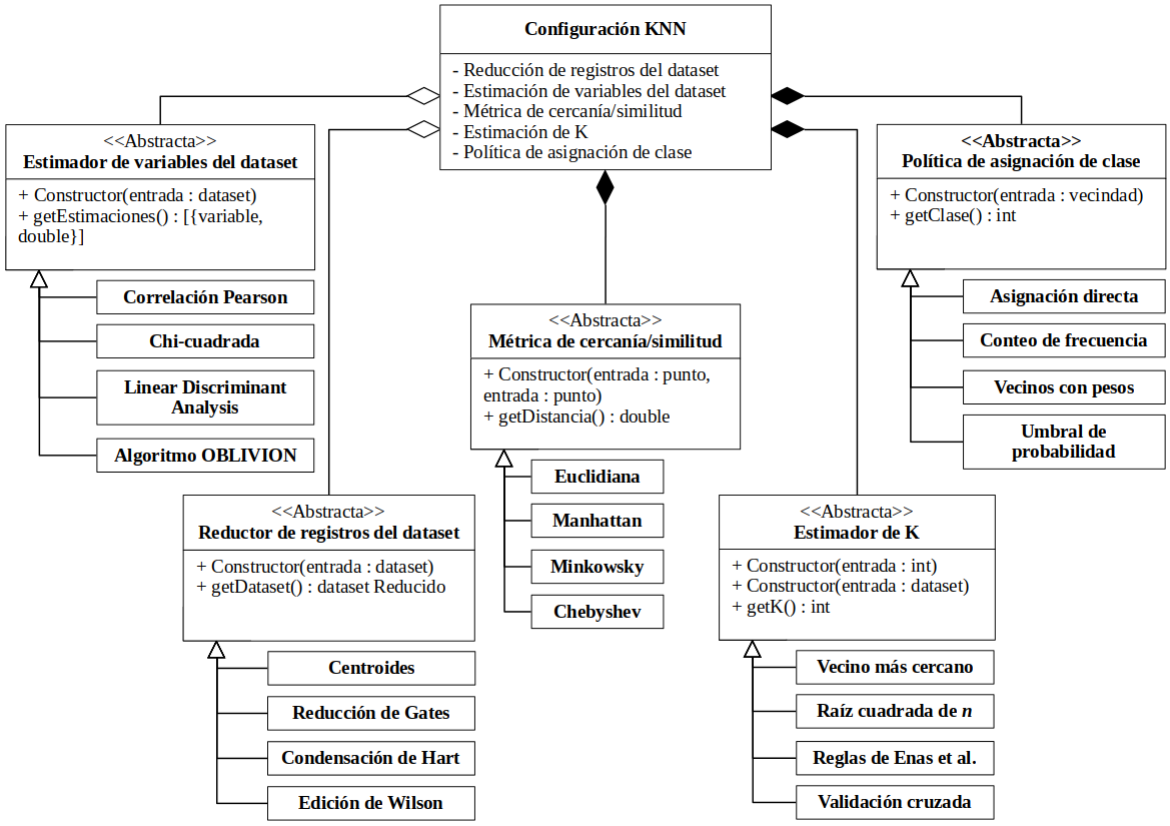

Figura 2.6: Relaciones de composición de las etapas de configuración para implementaciones de KNN.

También existen situaciones de trabajo en que la implementación de una o más etapas esta predeterminada. En ambos casos, una guía de afectaciones entre etapas de configuración resulta útil para operar adecuadamente el algoritmo KNN y obtener resultados significativos.

Las etapas de configuración pueden presentar comportamientos o requerimientos que restrinjan la operabilidad en conjunto con otras etapas. En la etapa de estimación de variables del dataset la opción elegida puede arrojar distinto resultado si antes se hizo una reducción de registros del dataset, ya que es posible que el número de miembros en el dataset no sean suficientes para los métodos estadísticos. Algunas técnicas de estimación de variables requieren comprobar predicciones para determinar la relevancia de las variables, esto se obtendría observando el resultado de corridas de KNN de modo que se ve afectada por la configuración de KNN usada. Para correcta configuración se sugiere: realizar la estimación de variables antes de una reducción de registros; si se realiza de manera póstuma a una reducción de registros, verificar la cardinalidad del dataset; si la opción requiere corridas de KNN, debe usarse la misma configuración.

Para la etapa de reducción de registros del dataset se hacen las siguientes observaciones. Algunas técnicas que resuelven esta etapa están diseñados para funcionar en un esquema de vecino mas cercano, de modo que requieren un tamaño especifico de vecindad (valor de $K = 1$). Otras técnicas de reducción agregan al dataset registros artificiales y no hay seguridad de que el dataset resultante incluya alguno de los registros originales. Derivado de esto es posible que el dataset solamente incluya un registro por clase o sólo unos pocos más, de modo que es fácil elegir una K que abarque la totalidad del dataset. Existen técnicas de estimación de reducción que usan el resultado de KNN para determinar si el registro pertenecerá al nuevo dataset, así que su proceso depende de la configuración de KNN usada. Se recomienda: observar si

la política de asignación de clase requiere una reducción de registros; verificar el número de registros presentes en el dataset a fin de que K cubra sólo una porción de ellos; si usa el algoritmos de predicción en su proceso, debe usarse la misma configuración.

En la etapa donde elige la métrica de cercanía/similitud existe una variedad de métodos para distintas objetos comparables, debe cuidarse que la métrica sea acorde al tipo de dato (numérico, texto, documentos). También debe observarse la relación entre objetos comparables y el resultado de la métrica, ya que existen métricas que operan con relaciones uno a uno, uno a muchos, y muchos a muchos. Otro aspecto a considerar es la clase de métrica, ya que una métrica de similitud nos da como resultado un indicador entre nula similitud de 0 y similitud total de 1, de forma inversa una métrica de cercanía indica los elementos más parecidos con valores más cercanos al 0. La selección de vecindad y algunas política de asignación de clase hacen uso del resultado de la métrica elegida, y la diferencia de orientación puede generar resultados incorrectos. Las recomendaciones son: identificar las necesidades de su objetivo de trabajo para elegir la métrica pertinente al tipo de dato; la relación entre objetos comparables está ligada a los registros del dataset, por lo que el número de registros puede impedir el uso de métricas con relación de uno a muchos o muchos a muchos; si la opción mide similitud (alta es cercano), la selección de vecindad y la opción de política de asignación de clase debe ir en el mismo sentido.

La etapa de estimación de K cuenta con distintas técnicas para elegir su valor que en general se calcula con respecto al número de registros en el dataset de entrenamiento. Por lo tanto, toda opción que impacte la cantidad de posibles vecinos se debe considerar al elegir K. Por ejemplo, la elección de política para asignación de clase o de técnica para reducción de registros del dataset puede requerir valor de $K = 1$, o en dado caso imponer un límite superior dado que siempre se da $K \geq 1$. El número de registros en el dataset de entrenamiento condiciona en cierta medida la elección de K, ya que K no debe exceder esta cantidad o de lo contrario la

vecindad de referencia está incompleta. Verifique el número de registros presentes en el dataset, la vecindad de referencia debe representar sólo una fracción del total. En algunas políticas para asignación de clase no se contemplan los casos de empate, por lo que se recomienda elegir valores impares para configurar K.

Las políticas para asignación de clase suelen estar en función de la vecindad de referencia, por lo que son afectadas directamente por el parámetro K. Estas políticas se encuentran entre las posibles restricciones para la estimación de K, ya que algunas puede requerir una reducción de registros del dataset y como se mencionó esto puede limitar las opciones para K. Del mismo modo las políticas pueden requerir una vecindad de referencia con más de un miembro debido a que realiza análisis de mayor complejidad. Algunas políticas requieren la configuración de un límite inferior para los contadores de clases presentes en la vecindad, este límite debe sintonizarse con el tamaño del dataset de entrenamiento y de la vecindad de referencia de modo que no resulte en una condición imposible de cumplir.

CAPÍTULO 3

Estado del arte

Este capítulo explora los avances y aportes relacionados con la creación de un sistema de configuración de técnicas computacionales que emplee negociación multiagente. La primera parte aborda los aportes relacionados a la predicción en química computacional, tanto las técnicas computacionales empleadas con fines predictivos como los cuerpos de datos existentes de descriptores químicos. La segunda parte se dedica a las contribuciones en técnicas heurísticas y evolutivas con las que se ha abordado el ajuste automático de parámetros. La tercera sección recopila las librerías de software que permitan el rápido uso de técnicas de aprendizaje máquina. La cuarta sección de este capitulo muestra las herramientas de software que dan soporte a la negociación de múltiples agentes inteligentes.

3.1. Predicción en química computacional

El problema de la predicción de estructuras y propiedades químicas es una cuestión tratada en la literatura de variadas formas. Existen herramientas de software que implementan métodos fisicoquímicos como la Teoría de Funcionales de la Densidad (*Density functional theory* - DFT) mediante cálculo computacional [Finocchi, 2011]. La dificultad de emplear métodos de cálculo radica en que suelen requerir una cantidad poco manejable de procedimientos, y con ellos se asocian necesidades de procesamiento, almacenamiento y gestión. Por esta razón se han desarrollado aplicaciones de minería de datos y aprendizaje máquina capaces de llegar a estas predicciones.

3.1.1. Bases de datos químicas

Esta sección se revisa un conjunto de bases de datos químicas con potencial relevancia para el proyecto. Se observan distintas bases de datos producidas para química computacional, su contenido y disponibilidad, esto con la finalidad de localizar bases de datos que puedan dar a los agentes una perspectiva de la variedad de estas. Son una selección de bases de datos químicos encontradas en la literatura que contienen descripciones de moléculas, materiales, propiedades y datos cristalográficos. En cada caso se indica el enfoque con el que se construyeron los conjuntos de datos, si su origen es experimental o hipotético, los registros totales que contienen y la licencia de acceso que ofrecen. Para consultar o indagar acerca de otras bases de datos químicas puede dirigirse al sitio de la *International Union of Crystallography* o al sitio de *Information Resources on Inorganic Chemistry*, en los que está disponible un directorio de proyectos de bases de datos químicas e información sobre esos proyectos.

La *Cambridge Structural Database* (CSD) [Groom et al., 2016] es una base de datos de estructuras experimentales orgánicas y metal-orgánicas. *The Cambridge Crystallographic Data Centre* (CCDC) compila y distribuye la CSD y la ha tratado para ser utilizada por investigadores de todo el mundo. Contiene 1,123,238 estructuras obtenidas experimentalmente.

La *Crystallography Open Database* (COD) [Gražulis et al., 2012] es un esfuerzo para el libre acceso a datos cristalográficos. La base de datos es una colección de estructuras cristalinas de minerales y compuestos orgánicos, inorgánicos, metal-orgánicos, pero excluyen datos sobre biopolímeros. En conjunto denominan el contenido de la base de datos como "moléculas pequeñas / células unitarias de tamaño pequeño a mediano". Según su pagina web oficial cuenta con 475,130 entradas obtenidas experimentalmente. El acceso es libre tanto para ingresar nuevas entradas (asociadas a una publicación) como para descargar la base de datos completa o partes de esta.

El *Protein Data Bank* [Berman et al., 2007] es una colección de datos de la estructura tridimensional (3D) de macromoléculas biológicas que se ha construido en cooperación internacional. Contiene los datos de estructura tridimensional y metadatos experimentales de proteínas, ADN y ARN. Según su pagina web oficial tienen un total de 188,403 modelos y entradas principalmente obtenidos experimentalmente. El acceso a la base de datos es libre y en el formato Crystallographic Information File (CIF).

La *Predicted Crystallography Open Database* (PCOD) [Le Bail, 2005] es una colección de archivos CIF hipotéticos, los autores toman como entrada modelos predichos por un algoritmo y producen las características de estos compuestos con el software GRINSP. Los resultados obtenidos incluyen CIF hipotéticos de óxido de boro aluminio y fluoruros de elementos 3d.

Según su pagina web cuenta con 1,062,771 entradas obtenidas computacionalmente. El acceso es libre tanto para ingresar nuevas entradas como para descargar la base de datos completa o partes de esta.

La *Inorganic Crystal Structure Database* (ICSD) [Hellenbrandt, 2004] es una base de datos producida por el *National Institute of Standards and Technology* (E.U.A). Como su nombre lo indica, consiste en datos de la estructura cristalina de compuestos inorgánicos que contiene información bibliográfica, química, celda unitaria, grupo espacial, entornos experimentales, grupo de mineral, etc. Según su pagina web cuentan con una recopilación de 210,000 entradas obtenidas de la literatura. Su modelo de acceso es mediante compra de suscripción a través del *National Institute of Standars and Technology* (E.U.A.).

The Open Quantum Materials Database (OQMD) [Kirklin et al., 2015] es una colección de de propiedades termodinámicas y estructurales calculadas mediante DFT. No distribuyen las estructuras originales, sólo el código de colección en ICSD. Según el sitio web del proyecto, la base de datos contiene los resultados calculados para 815,654 materiales. La colección de cálculos se encuentran con acceso libre.

La *Virtual 2D Materials Database* (V2DB) [Sorkun et al., 2020] es una base de datos conformada por materiales 2D y sus propiedades, ambas generados computacionalmente. Utiliza el formato de valores separados por comas (CSV) y contiene información sobre la fórmula química, el prototipo de cristal y las propiedades fisicoquímicas calculadas. El conjunto de datos esta formado por 72,522,240 materiales 2D candidatos que aun no han sido verificados experimentalmente.

Tabla 3.1: Sumario de la sección 3.1.1 Bases de datos químicas.

Base de Datos	Tipo	Tipo de Acceso	Registros Totales
Cambridge Structural Database (CSD)	Experimental	Restringido	1,123,238
Crystallography Open Database (COD)	Experimental	Abierto	475,130
Protein Data Bank (PDB)	Experimental	Abierto	188,403
Predicted Crystallography Open Database (PCOD)	Hipotética	Abierto	1,062,771
Inorganic Crystal Structure Database (ICSD)	Experimental	Restringido	210,000
The Open Quantum Materials Database (OQMD)	Experimental	Abierto	815,654
Virtual 2D Materials Database (V2DB)	Hipotética	Abierto	72,522,240

3.1.2. Minería de datos y aprendizaje máquina en química computacional

En esta sección se exploran aportes que dan una perspectiva de técnicas de minería de datos y aprendizaje máquina. Según nos muestra Li [Hao Li et al., 2019] estas técnicas son exitosas

en predecir la tendencia de los datos y se han usado ampliamente en resolver problemas de procesos químicos, y procesos de ingeniería. Por ejemplo, una aplicación de la minería de datos es la de Günay [M. Erdem Günay et al., 2018] que emplea análisis de datos y árboles de decisiones para localizar mejores condiciones de electrorreducción de CO2. Usaron árboles de decisiones para determinar la significancia de las variables para encontrar las de impacto mas relevante.

Un ejemplo de las aplicaciones del aprendizaje máquina para problemas en química se encuentra en el trabajo de Gupta [Gupta et al., 2016] se enfoca en predecir gelificación empleando distintos métodos estadísticos y de aprendizaje automático: *Support Vector Machines* (aprendizaje supervisado), *Random Forest* (aprendizaje conjunto), *Neuronal Networks* (aprendizaje supervisado) y *K Nearest Neighborns* (reconocimiento de patrones). También en el área de aprendizaje máquina se encuentra el trabajo Loredo [Loredo Pong, 2022], enfocado en la predicción de gelificación de organogeladores benzoato. Mediante el diseño de experimentos selecciona variables relevantes y valores de parámetro para usar el algoritmo *K Nearest Neighborns* (KNN). De modo que se puede entregar como clasificación el estado de agregación (gel, soluble, insoluble, precipitado) para el contexto del caso de estudio.

Como ejemplo de la aplicabilidad conjunta de la minería de datos y el aprendizaje máquina, se encuentra el trabajo de Cerecedo [Cerecedo Cordoba, 2020], en el cual propone una arquitectura para estimar la temperatura de fusión de líquidos iónicos. Su arquitectura contempla una amplia fase de entrenamiento. En ella emplea técnicas de clustering para formar grupos desde el análisis del cuerpo de datos, y también se generan redes neuronales artificiales (RNA) usando un algoritmo genético. Los resultados de las RNA se ponderan con técnicas de validación cruzada y son sometidas a un proceso de selección.

En el caso del aprendizaje profundo, Goh argumenta [Goh et al., 2017] que las redes neuronales profundas poseen alta aplicabilidad y eficacia en problemas dentro de la química computacional como *Quantitative Structure-Activity Relationship*, predicción de estructura de proteínas, diseño de materiales y predicción de propiedades.

La Tabla 3.2 recoge las aplicaciones de química computacional tratadas en esta sección. Se les agrupa en el contexto de aplicación correspondiente, planteados en la sección 1 Introducción: 1) representación de la entrada, 2) procesamiento de los datos y 3) presentación de los datos.

Tabla 3.2: Sumario de la sección 3.1.2 Minería de datos y aprendizaje máquina.

Contexto	Ejemplo de aplicación
1.- Representación de la entrada	SMILES [Weininger, 1988], CIF [Hall et al., 1991]
2.- Procesamiento de los datos	Cálculo computacional: •Gaussian [Frisch et al., 2016], •GAMESS [Gordon and Schmidt, 2005] Clasificación y predicción: •Aplicaciones de minería de datos [M. Erdem Günay et al., 2018, Loredo Pong, 2022] •Aplicaciones de aprendizaje máquina [Goh et al., 2017, Gupta et al., 2016, Cerecedo Cordoba, 2020]
3.- Presentación de los datos	Jmol [Hanson, 2016], MGLTools [Sanner, 1999]

3.2. Técnicas para el ajuste de parámetros

En la literatura se ha reportado se el uso de una variedad de técnicas para atender para el problema de ajuste de parámetros, abarcando métodos exactos, heurísticos, y de computación evolutiva.Como ejemplo del uso de técnicas heurísticas se encuentra el trabajo de Martínez [Martínez González, 2001] en el que empleó el método simplex para ajustar los parámetros de distintos modelos. Realizan ejercicios simulación de una columna de desorción empleando el software Ecosim. El autor destaca la efectividad del método simplex Nelder-Mead al trabajar con muchos parámetros.

En cuanto a técnicas metaheurísticas (estrategias generales para construir algoritmos) encontramos algunos trabajos, como el de Rivera [Rivera Zárate, 2009] donde trata el ajuste de parámetros en condiciones dinámicas para el problema de enrutamiento de consultas semánticas (*Semantic Query Routing Problem* – SQRP). El método propuesto para el SQRP es un algoritmo del tipo colonia de hormigas. Rivera emplea competencias de Hoeffding para la configuración del algoritmo, lo que lo hace alcanzar un desempeño destacado para SQRP en la literatura.

Como ejemplo de la aplicación de computación evolutiva se tiene el trabajo de Fernández [Eduardo Fernandez et al., 2019] en donde tratan la elicitación de parámetros para el problema de decisión multicriterio, empleando un algoritmo evolutivo (AE). Se parte de un modelo de preferencias y un conjunto de ejemplos dados por el tomador de decisiones, después los ejemplos son procesados por el *Preference Disaggregation Analysis* (PDA). La técnica PDA es un método ad-hoc de regresión y es usada para estimar los valores de parámetro del modelo

de preferencias. Fernández usa una variante múltiobjetivo del algoritmo genético (AG) para optimizar el resultado del PDA. El autor indica que la elicitación indirecta permite la inferencia de parámetros a partir de un conjunto de ejemplos.

Cerecedo en [Cerecedo Cordoba, 2020] propone una arquitectura para estimar la temperatura de fusión de líquidos iónicos. En su fase de entrenamiento emplea *Exhaustive grid search* para configurar *Support Vector Regression* (SVR) y *Regression Trees* (RT) de modo que formen clusters a partir de los datos. En un momento posterior genera distintas RNA y optimiza su configuración mediante algoritmo genético. Su objetivo es que las RNA aprendan de la selección de modelos estimadores SVR y RT.

Aunque existe variedad en cuanto a técnicas y enfoques de solución para el problema de ajuste de parámetros, la revisión del estado del arte no encontró trabajos en que el paradigma de agentes inteligentes fuera aplicado para este problema en el contexto de química computacional. De modo que consideramos novedoso el uso de enfoques basados en agentes inteligentes en este campo en particular. Los trabajos mencionados en esta sección se agrupan por el tipo de técnica que emplean para tratar el ajuste de parámetros en la Tabla 3.3.

3.3. Librerías de aprendizaje máquina

El sistema de recomendaciones propuesto requiere una colección de técnicas de aprendizaje máquina. La revisión de la literatura condujo a ubicar nueve librerías más desarrolladas: NumPy, SciPy, Keras, MlPack, Orange, Scikit-learn, Theano, PyTorch y TensorFlow.

Tabla 3.3: Sumario de la sección 3.2 Técnicas para el ajuste de parámetros.

Tipo de técnica	Técnica aplicada	Contex
Exacto	Simplex [Martínez González, 2001]	Model colum
Heurístico	*Preference Disaggregation Disaggregation Analysis* [Eduardo Fernandez et al., 2019]	*Portfo Multi- Optim*
	Exhaustive grid search [Cerecedo Cordoba, 2020]	Config *Suppo* y *Regr*
Metaheurístico	Colonia de hormigas [Rivera Zárate, 2009]	Enruta redes
Computación evolutiva	Algoritmo Genético [Cerecedo Cordoba, 2020, Eduardo Fernandez et al., 2019]	[Cere neuron [Edua *Multi- Optim*

El grupo conformado por Keras, Theano, PyTorch y TensorFlow está enfocado en el cálculo numérico mediante tensores. Los tensores son una generalización de vectores y matrices que se conciben como una matriz multidimensional. Las principales aplicaciones de los tensores son redes neuronales artificiales profundas.

Las otras librerías revisadas tienen funciones de uso más general. Orange si bien ofrece herramientas que permiten a quienes no tienen experiencia en programación desplegar algoritmos de aprendizaje máquina, su entorno de programación no es lo suficientemente flexible para acoplarse a las otras tecnologías implicadas en este proyecto de tesis.

MlPack está dedicado al aprendizaje máquina y esta implementado en lenguaje C++, y ofrece interfaces de software para acoplarlo con aplicaciones escritas en otros lenguajes de programación.

SciPy es una librería constituida de algoritmos matemáticos, cuenta con herramientas como algoritmos de optimización, manejo de ecuaciones algebraicas, y ecuaciones diferenciales.

NumPy ofrece facilidades para operaciones matemáticas, manejo de datos, como lectura de formatos y manipulación de matrices. Es común que otras librerías con un mayor abstracción internamente empleen Numpy para sus operaciones.

La librería seleccionada para el proyecto es Scikit-learn [Pedregosa et al., 2011] la cual está orientada al análisis predictivo de datos. La librería está escrita en lenguaje Python e implementa una colección de técnicas de minería de datos, aprendizaje máquina y estadística fácilmente configurables. También provee herramientas para la manipulación de datos y una documentación detallada nutrida de una selección de aportes publicados en la literatura [Buitinck et al., 2013].

3.4. Conclusiones

En los trabajos mencionados en este capítulo se observan tres aspectos que les describen, estos pueden visualizarse como tareas independientes que unidas forman un proceso mayor, en la Figura 3.1 se muestra la estructura de composición. En primer lugar cuentan con una capa de problema real, es decir, donde se indica la cuestión de predicción que interesa al investigador. Sobre de esa se encuentra una capa dedicada a como será tratado el problema, aquí se define el algoritmo a emplear y la forma del cuerpo de datos. La capa superior es la de elección de parámetros, en esta se indican las técnicas usadas para definir los valores de configuración del algoritmo y lograr su correcto desempeño.

Con el esquema propuesto podemos abstraer los distintos casos de aplicación, por ejemplo el trabajo de Loredo [Loredo Pong, 2022] encaja como el caso A, en su capa de problema real ubicamos la predicción de gelificación. El problema real lo aborda mediante el algoritmo KNN, indicado en su capa de tratamiento del problema. El modo en que obtuvo los valores de parámetro para hacer trabajar este algoritmo de clasificación fue el diseño de experimentos que debe indicarse en la capa de selección de parámetros. En la Figura 3.1 se muestra la estructura de capas describiendo las aplicaciones de Loredo y Cerecedo. El trabajo de Cerecedo [Cerecedo Cordoba, 2020] es presentado como el caso B, en este el problema real es estimar la temperatura de fusión de líquidos iónicos. La manera en que se da tratamiento al problema es la técnica de neuroevolución. Finalmente, el autor emplea Exhaustive grid search para la selección de parámetros.

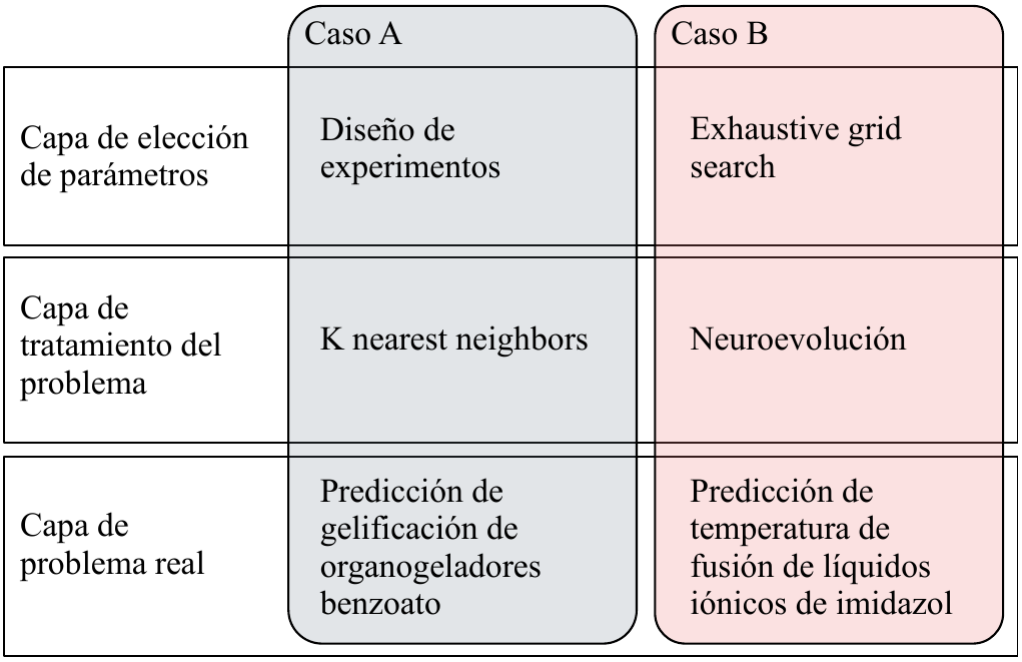

Figura 3.1: Diagrama abstracción de los casos de aplicación descritos pertenecientes al campo de la química computacional.

Los aportes descritos en la sección 3.2 corresponden a la capa de elección de parámetros. La premisa observada en estos trabajos es que dado un algoritmo, el proceso se dedica a buscar o aproximar los valores de parámetro al óptimo. Esto presenta algunas situaciones interesantes a considerar.

La primera de estas situaciones es que algunos de los métodos aplicados a ajustar los parámetros los tratan como un todo, es decir, los recombinan sin asignarles un significado a cada uno y sin distinciones entre ellos. Tomando en cuenta que los parámetros tienen un rol especifico en la operación de los algoritmos, su ajuste implica compromisos entre sus roles que deben balancearse para una operación adecuada del algoritmo. Si bien pueden presentarse múltiples resultados y seleccionar entre ellos los más convenientes, esto puede requerir cierta experiencia que el usuario puede no poseer y requiere tiempo obtenerla. Por otro lado, los

métodos de ajuste enfocados en un sólo parámetro del algoritmo probablemente implicarán una función que relaciona otros parámetros. De modo que la cuestión del balance se traslada a los parámetros que forman parte del cálculo del parámetro individual.

En otra situación se ubica la falta de experiencia. Como anteriormente se menciona, seleccionar una configuración paramétrica de entre un grupo puede hacerse mediante la experiencia de un experto. En un ejercicio de selección de este tipo el experto debería tomar en cuenta tanto la operación del algoritmo, los objetivos de la investigación y evitar sesgar los resultados del algoritmo. Puede observarse que los algoritmos del estado del arte no incorporan mecanismos para formar experiencia con el tiempo. En cambio son ejercicios individuales de ajuste enfocados en la situación de investigación especifica y para trasladarse a otras situaciones se requiere estudiar los mecanismos implicados. Como se muestra en el trabajo de Cerecedo [Cerecedo Cordoba, 2020], se puede emplear una estrategia en la que se pueden generar alternativas en el modo de emplear los algoritmos, seleccionar los mas importantes y descartar el resto. La experiencia podría acelerar la generación de configuraciones con mayor relevancia o reducir las configuraciones descartadas de intentos insatisfactorios.

CAPÍTULO 4

Modelo multiagente para el ajuste de parámetros

En este capitulo se describe la propuesta de solución para lograr la configuración de tareas predictivas enfocadas al área de química. Esta configuración considera aspectos como la preparación del cuerpo de datos de trabajo, selección del algoritmo predictivo, y definición de los valores paramétricos que requiera el algoritmo. El enfoque de solución elegido para la configuración de parámetros es un software asistente que provea estas opciones de configuración. Este software estaría formado por un sistema multiagente (SMA) donde cada agente colabora con el resultado final. Se considera que el paradigma de agentes ofrece características útiles para la resolución del problema ya que pueden trabajar independientemente aplicando distintos enfoques sobre la problemática y con atributos como la experiencia pueden evolucionar su forma de proceder a fin de optimizar los resultados que proveen.

Se propone un modelo de ajuste de parámetros requeridos por un conjunto de algoritmos preseleccionados. El modelo buscará recomendar los parámetros adecuados para la problemática presentada por el experto en química. Esto se lograría empleando un SMA donde los agentes discuten y deciden en conjunto la configuración adecuada. Para razonar y lograr el consenso los agentes pueden emplear técnicas de aprendizaje máquina que les permitan explotar su experiencia, logrando formular configuraciones paramétricas exitosas y considerar las de los otros agentes. La abstracción de esta propuesta se ilustra en la Figura 4.1. La capa de problema real la define el experto en química, este presentará al modelo la situación a resolver: si la cuestión es predecir gelificación, temperatura de fusión, etc. El modelo solicitará al grupo de agentes que deliberen sobre la situación basándose en la información provista. Cada agente analizará sus experiencias previas y mediante un proceso de negociación le presentará opciones configuración a los otros agentes. La resolución de la negociación abarca la elección de algoritmo predictivo como la configuración paramétrica pertinente, de modo que el modelo de ajuste de parámetros interviene tanto en la capa de elección de parámetros como en la capa de tratamiento del problema.

Un modelo de ajuste de parámetros como se propone en este capítulo requiere resolver una serie de problemáticas. Se debe definir qué proceso elige la técnica predictiva a usar o cómo se ajustan las opciones con las que cuenta esa técnica. Este modelo de ajuste de parámetros requiere el estudio de las técnicas predictivas que conformarán su repertorio, además de representaciones de estas técnicas de modo que se pueda automatizar el razonamiento acerca de ellas. Dado que el modelo de ajuste plantea que se resuelva mediante negociación multiagente también se debe estudiar que protocolo de negociación es apropiado emplear, definir como los agentes dialogan sobre la solución del problema y la estrategia de negociación que deben emplear.

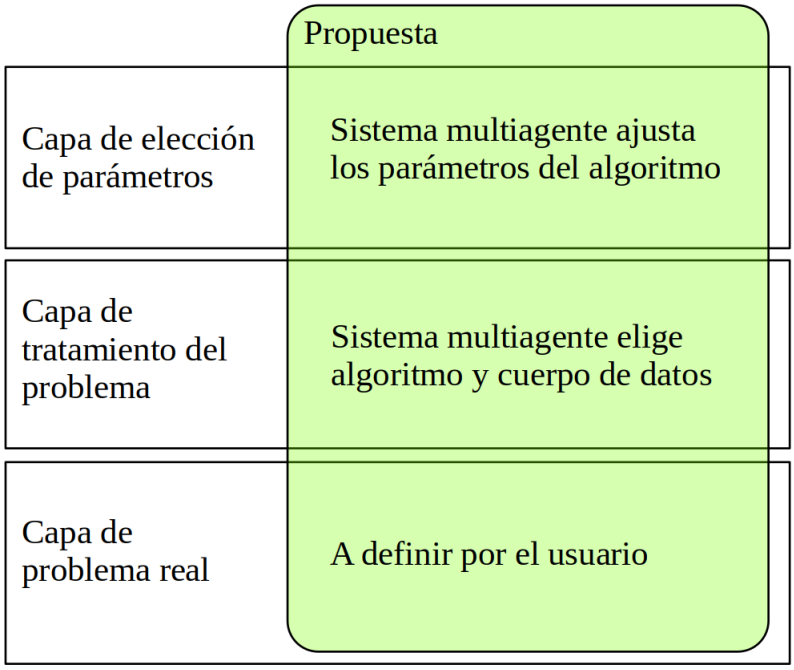

Figura 4.1: Diagrama abstracción del caso de aplicación de la propuesta.

La arquitectura para configuración paramétrica simula el razonamiento de un grupo de expertos en computación con distintos enfoques que discuten la configuración adecuada de una tarea de pronóstico dada. Es probable que diferentes expertos en computación propongan diferentes opciones. Sus propuestas podrían variar en el tipo de técnica computacional a utilizar, qué algoritmo específico utilizar, qué parámetros son óptimos o el tratamiento que se debe aplicar al conjunto de datos antes de aplicar el algoritmo.

4.1. Análisis de escenario

El químico solicita asesoría al grupo de expertos en informática, a quienes llamaremos configuradores. El usuario realiza una consulta que proporciona tanto una descripción del problema de pronóstico como el corpus de datos asociado con el problema. Los expertos en informática compararán esta consulta con su experiencia profesional y, como resultado

de la comparación, pueden formular una configuración desde su perspectiva. El proceso de comparación y formulación de la configuración está influenciado por el compromiso de configuración que apoyan. En este punto, se han trabajado las mejores configuraciones para compromisos individuales. Es necesario que los configuradores evalúen las soluciones de otros configuradores y decidan si formular nuevas configuraciones para lograr un equilibrio entre los compromisos de configuración que nos acerque a mejores configuraciones. Cuando los configuradores estén satisfechos con las configuraciones, se presentarán al usuario. En resumen, tenemos siete elementos, que son la consulta del usuario, los configuradores, la experiencia del configurador, los compromisos de configuración, la formulación de configuraciones, la discusión de las configuraciones y una adaptación de las configuraciones.

La consulta del usuario es la presentación del problema a los configuradores. Las características del problema permiten al configurador relacionarlo con situaciones anteriores en su experiencia. Por ejemplo, abordará la situación de manera diferente si se ha realizado una selección de variables en el corpus de datos que si no.

La experiencia del configurador es la acumulación de hechos observados sobre el comportamiento de los algoritmos en diferentes circunstancias. En la vida real, la experiencia de los expertos en informática depende de múltiples factores relacionados con su formación profesional. Existen beneficios de tener una experiencia amplia en comparación con una experiencia limitada; sin embargo, no es deseable que haya un sesgo en una discusión debido a diferencias en la experiencia. La experiencia de los configuradores debe ser objetiva, razonablemente amplia y uniforme entre los configuradores para lograr la estandarización.

La formulación de configuraciones es un proceso interno del configurador de búsqueda en la experiencia previa. Consiste en localizar el ejemplo en la experiencia que sea más similar a la consulta del usuario. Esto implica un análisis y ordenamiento de la totalidad de la experiencia.

La discusión de las configuraciones es el proceso mediante el cual los configuradores pueden llegar a un consenso. Los configuradores comparten entre ellos las configuraciones que consideran adecuadas para la tarea de pronóstico, y los otros configuradores deciden si están de acuerdo en proceder con ella. Esto significa que, internamente, los configuradores pueden evaluar una configuración e identificar si es aceptable o no.

La adaptación de las configuraciones es un mecanismo interno del configurador que modifica la formulación de configuraciones. En el transcurso de la discusión, existe la oportunidad de cambiar de opinión o hacer que otros cambien de opinión. Cuánto cambian de opinión los configuradores depende de cuán dispuestos estén a alejarse de su configuración ideal.

Los compromisos de configuración son las tendencias de enfoque que los expertos en informática tienen al abordar tareas de pronóstico. Las configuraciones propuestas por los expertos en informática pueden variar en el tipo de técnica computacional a utilizar, qué algoritmo específico usar, qué parámetros son óptimos o el tratamiento que se debe aplicar al conjunto de datos antes de aplicar la técnica de pronóstico.

Nuestra propuesta involucra automatizar la toma de decisiones que estos compromisos significan, por lo que debemos delimitar su comportamiento. Los compromisos tienen un efecto en cómo los configuradores perciben su experiencia previa y la consulta del usuario. Los compromisos amplifican la importancia de algunas características involucradas en la tarea de pronóstico y la reducen en otras. Esto significa que si se compara un elemento de la experiencia con la consulta del usuario, las características de mayor importancia tendrán más influencia para decidir cuán similares son estos dos elementos. Si el elemento de la experiencia es más similar a la consulta del usuario, también estará mejor adaptado para resolverlo. Para cada compromiso, extraemos las características importantes e identificamos las variables que se relacionan con ellas.

La discusión, adaptación y combinación de las diferentes configuraciones que los configuradores pueden proponer es un problema en sí mismo. Las configuraciones finales deben ser aceptables para los objetivos de las partes y deben sumar las cualidades de cada propuesta. El enfoque de solución para esta situación es concebir a los configuradores como un SMA en el que negocian para contribuir a ubicar las mejores configuraciones.

4.2. Módulos-agentes necesarios

El modelo de recomendaciones deberá contar con una base de conocimiento y técnicas computacionales que puedan dotarle de capacidad predictiva. El modelo procesará una descripción del problema y del conjunto de datos que contiene lo que se sabe de la cuestión de predicción. Este proceso estaría compuesto de tres tipos de decisión que relacionan los conocimientos del modelo con la situación que se le pide resolver:

1. Relación del tipo de problema con la eficacia de las técnicas computacionales disponibles.
2. Relación de configuraciones de parámetros y los desempeños de las técnicas computacionales.
3. Relación de configuraciones paramétricas y los tipos de dataset.

Estos tres tipos de decisión son compromisos que el experto en computación asume en el proceso de elección de la configuración. Un experto en química que busque asesoramiento para la realización de una tarea de previsión podrá recibir asesoramiento de un experto en

informática que sea partidario de uno de estos compromisos. Sin embargo, al hacer esto los demás compromisos deben ajustarse al que se fijó. Sin embargo, esto no quiere decir que ese compromiso en particular sea el mas adecuado.

Resultado de razonar acerca de estas relaciones, el modelo de recomendaciones proporcionará al usuario una lista de opciones describiendo técnica y parámetros que considera adecuado para la situación de pronóstico presentada.

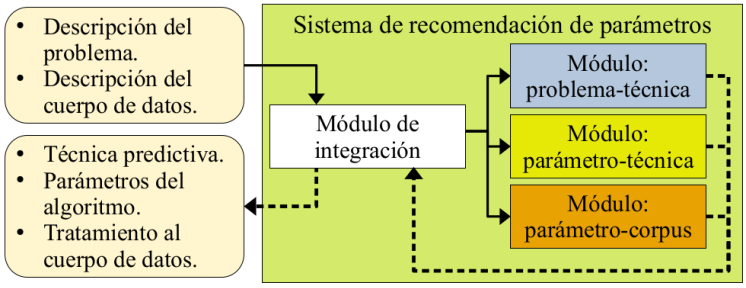

Figura 4.2: Arquitectura elemental del SMA de recomendaciones.

La arquitectura para configuración paramétrica se puede visualizar como una serie de módulos que están encargados de desempeñarse en los tres tipos de decisión descritos. Esta descripción general del funcionamiento del modelo de recomendaciones se encuentra plasmada en la Figura 4.2. En este esquema cada aspecto de decisión se transforma en un modulo con propósito especifico descrito a continuación:

- Módulo de relación problema-técnica: Este módulo se dedica a valorar y proponer algoritmos basándose en el tipo de problema a resolver. Dada la amplitud de trabajos en esta área, se emplearan los resultados de efectividad del estado del arte para construirlo.
- Módulo de relación parámetro-técnica: El rol de este módulo esta enfocado en sugerir los parámetros que serán usados en los procesos de las técnicas computacionales (ciclos,

podas, etc.). Este módulo se aproxima a lo que comúnmente se le llama ajuste de parámetros. Para construirlo se emplearán resultados de efectividad del estado del arte y ajuste automático.

- Módulo de relación parámetro-corpus: Este módulo estará dirigido a los parámetros que tienen que ver con el conjunto de datos, como las variables relevantes o descarte de registros redundantes. Se usarán resultados de efectividad del estado del arte y ajuste automático para su construcción. Los datos producidos permitirían sugerir configuraciones para usar según los tipos de conjuntos de datos.

- Módulo de integración: Este módulo se dedica a identificar los rasgos característicos del problema de pronóstico que busca resolver el usuario. De este modo se le permitirá a los otros módulos razonar acerca de la situación que buscarán resolver. De forma inversa, tomará la solución propuesta por los otros módulos y la presentará al usuario junto con casos de éxito.

El módulo de integración tiene tareas de una clase distinta a los primeros tres que están directamente ligados a los compromisos de configuración. Se puede observar que existe una interdependencia entre el módulo de integración y los otros módulos, pero no existe una interdependencia entre módulos relacionales. El rol del módulo de integración esta enfocado en proveer a los otros módulos información suficiente y en compaginar los resultados de los otros módulos. Compaginar las respuestas de los módulos relacionales es un problema en si mismo ya que la configuración paramétrica resultante debería ser aceptable para los objetivos de las partes y en lo posible sumar las cualidades de cada una. Durante el proceso de compaginación pueden producirse distintas situaciones de conflicto. Por ejemplo, el módulo problema-técnica podría elegir una técnica basado en otros problemas similares que se han resuelto con esa técnica, mientras que el módulo parámetro-corpus podría proponer otra

respecto a las características del dataset. Por su parte, el módulo parámetro-técnica podría elegir una opción distinta de técnica que los otros módulos buscando los parámetros con el mejor desempeño independientemente del problema o el dataset de trabajo.

El enfoque de solución para esta situación es concebir los módulos como un SMA, de modo que se puedan explotar las cualidades de la interacción de negociación (vea Figura 4.3). Las negociaciones permiten a los agentes proponer y opinar sobre las configuraciones paramétricas, de modo que todos contribuyen y aceptan al resultado final. De modo que se visualiza a cada módulo como un agente, tres agentes negociadores y un arbitro. El agente correspondiente al módulo de integración tendría la función arbitro y regular las interacciones de los otros tres agentes, los cuales configurarán la tarea de predicción. Será mediante las interacciones de negociación en que los agentes podrán intercambiar datos y acordar configuraciones paramétricas. Esta perspectiva de la arquitectura para configuración paramétrica se plasma en la Figura 4.3, en la que ubicamos dos fases: una de entrenamiento donde se prepara a los agentes y una de pronóstico donde los agentes configuran la tarea de pronóstico. En las siguientes secciones 4.3 y 4.4 se aborda el detalle de los componentes de estas fases.

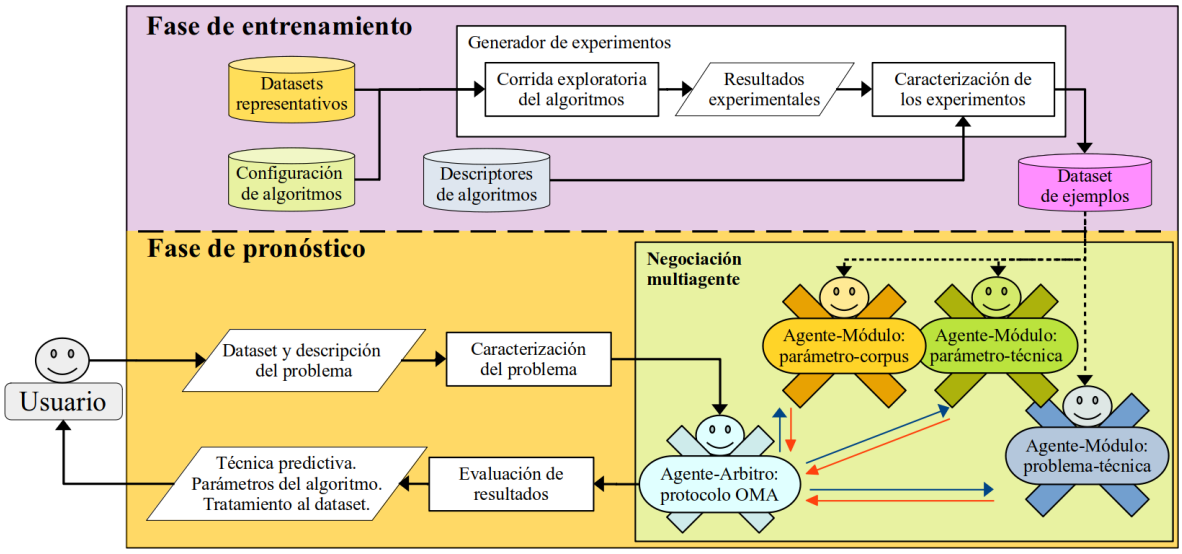

Figura 4.3: Arquitectura para la configuración de parámetros.

4.3. Fase de entrenamiento

El modelo de recomendación de parámetros propuesto requiere de una capacidad de razonamiento sobre las técnicas de pronóstico que usará como solución al problema planteado por el usuario. La forma de proponer configuraciones de algoritmos al usuario final es mediante del consenso de agentes inteligentes. Para que estos agentes puedan valorar y proponer configuraciones deben de contar con alguna experiencia o perspectiva acerca del comportamiento de las técnicas de pronóstico en distintas situaciones de trabajo. Así, el modelo de configuración paramétrica consta de dos fases, la primera es una fase de entrenamiento donde se genera la experiencia que los agentes inteligentes tomarán como base de su razonamiento, la segunda fase es donde los agentes inteligentes examinan bajo su propia perspectiva el significado de esa experiencia y deciden que opciones ofrecer como solución final.

La fase de entrenamiento produce la experiencia que usarán los agentes, formando el dataset de experiencia que contiene resultados experimentales caracterizados. La fase de entrenamiento comienza con una selección de datasets representativos y configuraciones de algoritmos (vea Figura 4.3). Un dataset representativo es uno cuyas características lo vuelven un ejemplo de un caso insignia entre los distintos tipos de datasets. Los datasets representativos tendrán variaciones entre ellos en aspectos como el número de registros, variables usadas, cantidad de clases diferentes y diferencia de tamaño entre clases. Paralelamente debe existir un almacén de conocimiento respecto a los algoritmos y a la configuración de estos. El dataset de configuraciones de algoritmos se compone de las partes en las que se puede dividir la configuración de un algoritmo y las formas en que se puede resolver cada sección resultante. Para formar este cuerpo de conocimiento pueden realizarse estudios como el de Villarreal [Villarreal Hernández et al., 2023] enfocado en la familia de algoritmos KNN. Este algoritmo puede requerir tomar distintas decisiones durante su configuración. Las etapas de decisión para la configuración del

algoritmo KNN cubren tareas como preparar el conjunto de datos de entrenamiento, elegir la heurística que decide la clase final, o elegir un valor de K [Villarreal Hernández et al., 2023]. Usando estas etapas de decisión podemos manipular la configuración del algoritmo y generar las distintas variantes de este mediante un proceso combinatorio.

La entidad descriptores de algoritmo contiene información de la literatura acerca del algoritmo como el tipo de aprendizaje que emplea, si produce un modelo de generalización, o si requiere modificar el dataset de trabajo.

Con los elementos antes tratados podemos generar ejecuciones exploratorias del algoritmo a fin de obtener datos sobre el desempeño del algoritmo en diferentes condiciones, esto está representado en el generador de experimentos. El generador de experimentos produce un solo cuerpo de experiencia que sería compartido por los agentes inteligentes y les permitirá asociar el comportamiento del algoritmo con diferentes circunstancias. Así, se produce una colección de resultados experimentales caracterizados, que nombraremos dataset de experiencia.

Para abordar la operación del generador de experimentos definimos lo siguiente:

DR es la lista de datasets representativos de tamaño r.

CA es una matriz de configuraciones de algoritmos con la forma $[Resolutor, Etapa]$, donde $Etapa$ es la etapa de configuración y $Resolutor$ el método de resolver la etapa.

PP es una colección de las p columnas en CA que se asocian al pre-procesamiento de datasets, cada una con tamaño y.

DA es la matriz de nombres de *CA* con la misma forma $[Resolutor, Etapa]$, donde se obtienen los nombres de las etapas y resolutores.

DSM es la lista de tamaño s donde se almacenan los datasets modificados.

PA es una colección de las q columnas en *CA* que se asocian al procesamiento de datasets dentro del algoritmo, cada una con tamaño z.

DSExperience es el archivo donde se almacenan los experimentos caracterizados.

Algorithm 2 Experiment generator.

Require: $DR, CA, DA \neq \emptyset$
$PP \leftarrow pre\_processing(CA)$
$PA \leftarrow complement(CA, PP)$
$DSM \leftarrow combinatorDRPP(DR, PP)$
$DSExperience \leftarrow combinatorDSMPA(DSM, PA, CA, DA)$
Ensure: $DSExperience$

El generador de experimentos (Algoritmo 2) inicia obteniendo las entradas *DR* y *PP*. Usa el procedimiento *combinatorDRPP* para generar los datasets modificados por el pre-procesamiento. Este procedimiento genera las combinaciones entre los datasets y las opciones de pre-procesamiento, y ejecuta el método de pre-procesamiento con ese dataset. Para el caso en que sólo hay una columna con opciones de pre-procesamiento se obtienen las combinaciones señaladas en la Tabla 4.1, incluyendo los datasets originales (pre-procesamiento nulo). Para el caso $p \geq 2$ se realizan las combinaciones de la Tabla 4.1 por las p de columnas de *PP*. Esto significa que el *combinatorDRPP* contempla el caso de pre-procesamiento acumulado, por ejemplo aplicar una selección de variables por Pearson y al dataset resultante aplicarle una condensación rápida. De modo que *DSM* tiene datasets modificados con las combinaciones $[DR_0 + \emptyset, ..., DR_r + \emptyset, DR_0 + PP_{0,0}, ..., DR_0 + PP_{y,0}, DR_1 + PP_{0,0}, ..., DR_1 + PP_{y,0}, ..., ..., DR_r + PP_{0,p}, ..., DR_r + PP_{y,p}]$.

Tabla 4.1: Matriz combinatoria de datasets modificados para una columna de pre-procesamiento.

DR	$PP_{\emptyset}$	PP_0	PP_1	$PP_{...}$	PP_y
0	0+∅	0+0	0+1	0+...	0+y
1	1+∅	1+0	1+1	1+...	1+y
...	...+∅	...+0	...+1	...+...	...+y
r	r+∅	r+0	r+1	r+...	r+y

Una vez que ha generado los datasets modificados *DSM*, el siguiente paso es realizar experimentos con las configuraciones del algoritmo que son del proceso interno. El procedimiento *combinatorDSMPA* toma las opciones de configuración en *PA*, genera la combinación de opciones de configuración con los datasets de *DSM*, y la ejecuta con ese dataset. Es decir, genera configuraciones de forma $(DSM_i, PA_{j,0}, PA_{l,1}, ..., PA_{z,q})$ ya que se considera que las partes del proceso interno se eligen una sola vez por ejecución. La Tabla 4.2 ilustra el orden combinatorio de este procedimiento. Estas combinaciones son configuraciones de algoritmo que se ejecutan en un esquema de validación cruzada, de modo que se obtienen mediciones de su desempeño. Así, el procedimiento *combinatorDSMPA* puede entregar una lista de experimentos y mediciones de estos. Esta lista se elabora revisando la asociación entre los procedimientos en *CA* y los nombres en *DA*, así como descriptores de los datasets en *DR*. De modo que puede transformarse cada experimento y su medición en una cadena de texto simple.

La lista completa de características que forman al dataset de experiencia se encuentra en la Tabla 4.3, cada característica forma parte de un grupo de descriptores. Se están considerando cinco tipos de descriptores: del problema de pronóstico, del dataset utilizado, del algoritmo utilizado, de la configuración paramétrica utilizada, y de la evaluación de desempeño del experimento. En la sección 4.4 se detalla como los agentes usan el dataset de experiencia para proponer configuraciones para la tarea de pronóstico.

Tabla 4.2: Matriz combinatoria de experimentos del generador de experimentos.

DSM	$PA_{j,0}$	$PA_{l,1}$	$PP_{m,...}$	$PA_{x,q}$
0	+0	+0	+0	+0
0	+1	+0	+0	+0
0	+...	+0	+0	+0
0	+z	+0	+0	+0
0	+0	+1	+0	+0
0	+1	+2	+0	+0
0	+...	+...	+0	+0
0	+z	+z	+0	+0
0	+0	+1	+1	+0
0	+1	+2	+2	+0
0	+...	+...	+...	+0
0	+z	+z	+z	+0
0	+0	+1	+1	+1
0	+1	+2	+2	+2
0	+...	+...	+...	+...
0	+z	+z	+z	+z
1	+0	+0	+0	+0
...	...	...	...	...
s	+z	+z	+z	+z

4.4. Fase de pronóstico

En la fase de pronóstico el objetivo es producir una configuración solución para la tarea de pronóstico planteada por el usuario final. Por ello la fase de pronóstico comienza con la interacción del usuario introduciendo la situación de pronóstico que desea resolver. El usuario ingresará una descripción del problema de pronóstico y el dataset de trabajo. Esta descripción debe hacerse en términos de los descriptores del problema de pronóstico y del dataset utilizado de la Tabla 4.3. Una vez que se han preparado los datos del problema el siguiente paso es solicitar al agente coordinador que empiece el proceso de deliberación. La primer tarea del agente coordinador es invocar a los agentes-módulo asignándoles un compromiso de

Tabla 4.3: Características de los experimentos agrupadas por grupo descriptor.

Descriptor	Característica del experimento
Descriptor del problema de pronóstico	- Clases a considerar - Clase objetivo - Preselección de variables - ¿Contiene campos categóricos?
Descriptor del dataset utilizado	- Número de registros - Clases presentes - Diferencia de tamaño entre clases - Número de variables
Descriptor del algoritmo utilizado	- Tipo de aprendizaje - Tipo de asociación - Requiere modidicar el dataset - Produce modelo de generalización
Descriptor de la configuración paramétrica utilizada	- Métrica de cercanía - Valor de parámetro K - Política de asignación de clase - Reducción de registros del dataset
Descriptor de la evaluación de desempeño del experimento	- Precisión media - Desviación estándar de la precisión

configuración y transmitiéndoles los datos relevantes del problema de pronóstico. En ese momento los agentes-módulo toman acción de inicialización, preparándose para la sesión de negociaciones.

4.4.1. Inicialización de los agentes

El proceso de inicialización, plasmado en la Figura 4.4 y el Algoritmo 3, inicia con la configuración del perfil de preferencias. El comportamiento sesgado que representan los compromisos identificados puede representarse mediante un esquema de preferencias. Para definir las preferencias de los agentes nos remontamos al conjunto de datos de experiencia producido en la

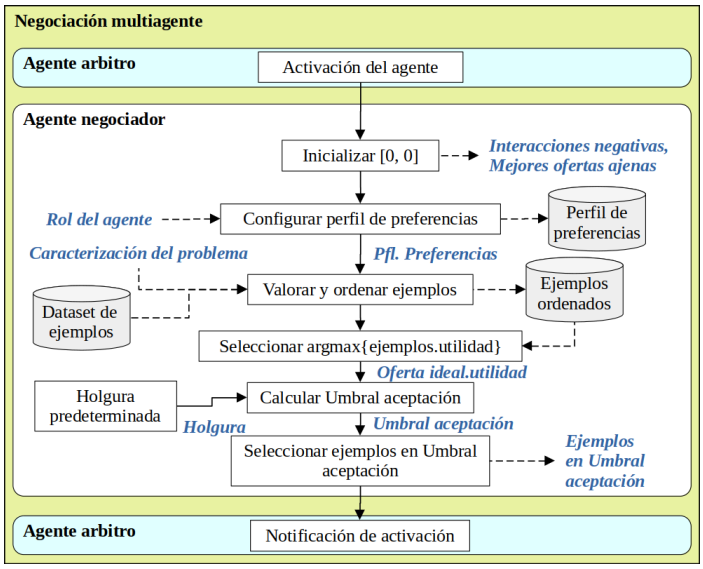

Figura 4.4: Inicialización del agente negociador.

fase de entrenamiento. Consideremos que cada uno de los descriptores incluidos en el conjunto de datos de experiencia representa un conjunto de variables específicas y también que los experimentos son registros descritos en esas variables. Estos registros son una colección de objetos únicos que se evaluarán según el perfil de preferencias. Una vez asociadas las variables y los descriptores, se debe establecer como se asocian los descriptores y los compromisos de configuración. De la descripción de los compromisos de configuración identificamos que cada uno relaciona dos aspectos: problemas y técnicas predictivas, parámetros y cuerpos de datos, parámetros y técnicas predictivas. Dadas estas relaciones la asociación corresponde al par de descriptores que representan esos aspectos. Para los agentes esta asociación indica el interés que tiene cada uno sobre los diferentes aspectos, por lo que les llamaremos relación de interés. Tenemos entonces que al agente que atiende el compromiso problema-técnica tiene una relación de interés sobre los descriptores del problema de pronóstico y los descriptores del algoritmo, al compromiso parámetro-corpus la relación de interés se encuentra entre los descriptores del dataset y de configuración del algoritmo, y para el compromiso parámetro-técnica la relación de interés va hacia los descriptores del problema del dataset y de configuración

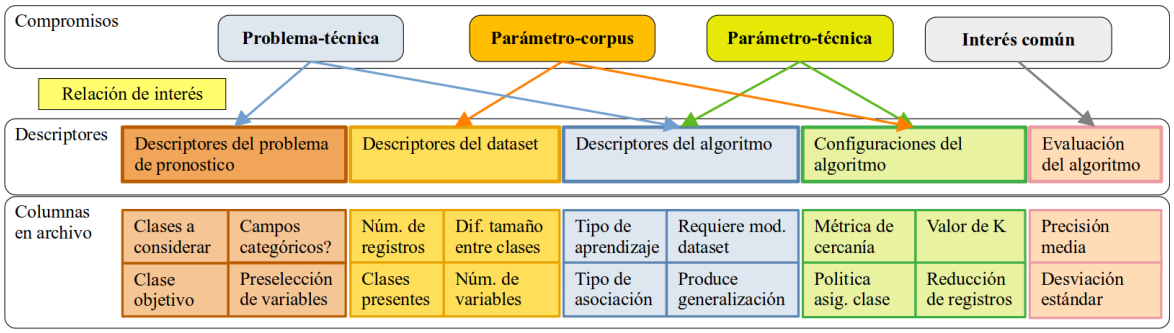

Figura 4.5: Relaciones de interés entre compromisos y descriptores.

del algoritmo; se entiende que siendo el objetivo de los agentes encontrar configuraciones exitosas, el descriptor de la evaluación del algoritmo es de interés de todos los agentes. La Figura 4.5 presenta un esquema con las relaciones antes descritas.

Algorithm 3 Agent_Activation

Require: $DatasetEjemplos, Compromiso \neq \emptyset$

1: $InteraccionesNegativas, MejoresOfertasAjenas, EjemplosOrdenados,$
$OfertaIdeal, UmbralAceptacion, OfertasAlternativas \leftarrow \emptyset$
2: $Preferencias \leftarrow getPreferencias(Compromiso)$
3: $Pesos \leftarrow getPesos(Compromiso)$
4: $EjemplosOrdenados \leftarrow valorarYordenar(DatasetEjemplos, Preferencias)$
5: $OfertaIdeal \leftarrow \arg\max\{EjemplosOrdenados.utilidad\}$
6: $Holgura \leftarrow calcularHolgura(InteraccionesNegativas)$
7: $UmbralAceptacion \leftarrow Ofertaideal.utilidad - Holgura$
8: $OfertasAlternativas \leftarrow \arg\{EjemplosOrdenados \geq UmbralAceptacion\}$

Podemos señalar que cada agente tiene un interés diferente hacia cada descriptor según su compromiso. Estas relaciones de interés indican dos niveles de interés, interés alto e interés medio. Consideraremos que todos los descriptores son de interés medio excepto aquellos en los que su compromiso tiene afinidad, incluida la evaluación del algoritmo. Representaremos estas relaciones de interés con la forma $Interes_{Compromiso,Descriptor}$, donde $Interes$ es $Alto$ o $Medio$ según se relacionen el $Compromiso$ y el $Descriptor$. De esta manera tenemos que el compromiso problema-técnica sólo tiene interés alto para los descriptores del problema de pronóstico, del algoritmo utilizado y evaluación de desempeño del experimento. Estas relaciones de intereses tienen dos funcionalidades. La primer funcionalidad es definir el impacto de los descriptores en

el cálculo de la utilidad para la configuración de ese experimento. Al igual que como se calcula la utilidad utilizando perfiles de preferencia normalizados [Lin et al., 2014, Hindriks et al., 2009, Baarslag et al., 2019], podemos asignar un valor de preferencia a un tema interno a la situación de negociación, y en nuestro caso el tema son los descriptores. Los descriptores con interés alto contribuirán más a la cuenta de utilidad que aquellos con interés medio, y ambos contribuyen a la utilidad total. Definimos que el perfil de preferencias consta de una lista de valores asociados a cada columna (característica) en el dataset de experiencia, formando la lista de características con forma $[Caracteristica_0, Caracteristica_1, ..., Caracteristica_n]$. Así, definimos una función para calcular el valor de preferencia para cada característica según $Interes_{Compromiso,Descriptor}$ sea $alto$ o $medio$. Para producir el valor de preferencia para una característica en un descriptor de interés alto se diseñó la Ecuación 4.1. Para las características de un descriptor de interés medio está la Ecuación 4.2. Para ambas ecuaciones están dadas las siguientes definiciones en función del dataset de experiencia:

CC es la cuenta total de columnas que describen a los experimentos.

CA es la cuenta total de columnas con las que hay relación de interés alto.

CM es la cuenta total de columnas con las que hay relación de interés medio.

TD es la tasa de diferenciación, un porcentaje que indica la diferencia entre el interés alto y el interés medio.

VP_{Alto} es el valor de preferencia para una característica en un descriptor de interés alto según $Interes_{Compromiso,Descriptor}$.

VP_{Medio} es el valor de preferencia para una característica en un descriptor de interés medio según $Interes_{Compromiso,Descriptor}$.

$$VP_{Alto} = \frac{1+TD}{CC} \tag{4.1}$$

$$VP_{Medio} = \frac{CM-(CA \times TD)}{CC \times CM} \tag{4.2}$$

Se genera el perfil de preferencias como una lista de valores de preferencia $Perfil_{Compromiso} = [VP_0, VP_1, ..., VP_{CC}]$ en el cual VP_i contiene el valor de la función $VP_{Interes}$ (Ecuaciones 4.1 y 4.2) donde *Interes* esta dado por $Interes_{Compromiso,Descriptor}$. Para el caso mostrado en la Figura 4.5 se observa que esta compuesto por 18 columnas, de las cuales cada compromiso tiene interés alto en 10 columnas y por lo tanto en 8 hay interés medio. Configurando el valor de $Perilla = 0.15$ usamos la Ecuación 4.1 para obtener $VP_{Alto} = 0.063889$, y con la Ecuación 4.2 obtenemos $VP_{Medio} = 0.045139$. Entonces la lista de valores de preferencia para cada compromiso resulta como en la Tabla 4.4.

El segundo paso de la inicialización del agente es valorar y ordenar ejemplos del dataset de experiencia respecto a su valor de utilidad. Ya que se han establecido los valores de preferencias los usaremos para definir el procedimiento del cálculo de la utilidad. Al momento de la activación del agente se le comunicaron los datos relevantes del problema de pronóstico, que al igual que los registros del dataset de experiencia tiene la forma $[Caracteristica_0, Caracteristica_1, ..., Caracteristica_{CC}]$. Bajo la premisa de que los agentes buscan casos similares en su experiencia se propuso una técnica basada en cercanía relativa al caso del usuario para establecer la utilidad. Es importante precisar que toda utilidad está en función del perfil de preferencias y este se encuentra asociado a un compromiso de configuración. Para obtener el cálculo de la utilidad definimos los siguientes conceptos y ecuaciones:

Tabla 4.4: Perfiles de preferencias calculados para cada compromiso de configuración.

Característica del experimento	Perfil del compromiso problema-técnica	Perfil del compromiso parámetro-técnica	Perfil del compromiso parámetro-corpus
Clases a considerar	0.063889	0.045139	0.045139
Clase objetivo	0.063889	0.045139	0.045139
Preselección de variables	0.063889	0.045139	0.045139
¿Contiene campos categóricos?	0.063889	0.045139	0.045139
Número de registros	0.045139	0.063889	0.045139
Clases presentes	0.045139	0.063889	0.045139
Diferencia de tamaño entre clases	0.045139	0.063889	0.045139
Número de variables	0.045139	0.063889	0.045139
Tipo de aprendizaje	0.063889	0.045139	0.063889
Tipo de asociación	0.063889	0.045139	0.063889
Requiere modidicar el dataset	0.063889	0.045139	0.063889
Produce modelo de generalización	0.063889	0.045139	0.063889
Métrica de cercanía	0.045139	0.063889	0.063889
Valor de parámetro K	0.045139	0.063889	0.063889
Política de asignación de clase	0.045139	0.063889	0.063889
Reducción de registros del dataset	0.045139	0.063889	0.063889
Precisión media	0.063889	0.063889	0.063889
Desviación estándar de la precisión	0.063889	0.063889	0.063889

$Caso_{Caracteristica}$ es la característica del caso de pronóstico a resolver.

$Ejemplo_{i,Caracteristica}$ es la característica del i-ésimo ejemplo del dataset de experiencia.

Entonces obtenemos la distancia unidimensional máxima DUM para cada característica mediante la Ecuación 4.3:

$$\begin{aligned} DUM_{Caracteristica} = \arg\max\{ & |Caso_{Caracteristica} - Ejemplo_{0,Caracteristica}|, \\ & |Caso_{Caracteristica} - Ejemplo_{1,Caracteristica}|, \\ & |Caso_{Caracteristica} - Ejemplo_{...,Caracteristica}|, \\ & |Caso_{Caracteristica} - Ejemplo_{n,Caracteristica}|\} \end{aligned} \tag{4.3}$$

Y calculamos el aporte de utilidad por característica de cada ejemplo usando la Ecuación 4.4:

$$AU_{i,Caracteristica} = VP_{Caracteristica} \times 1 - \frac{|Caso_{Caracteristica} - Ejemplo_{i,Caracteristica}|}{DUM_{Caracteristica}} \tag{4.4}$$

De modo que se genera la lista de los aportes de utilidad para cada característica de un ejemplo $[AU_{i,0}, AU_{i,1}, ..., AU_{i,n}]$, y la sumatoria de esta resulta en la utilidad total del ejemplo (Ecuación 4.5) respecto al caso de pronostico. Así al evaluar un ejemplo que tenga la máxima valoración obtenemos la utilidad normalizada, ya que $(VP_{Alto} \times A) + (VP_{Medio} \times M) = 1$.

$$Utilidad_i = \sum_{Caracteristica=0}^{n} AU_{i,Caracteristica} \tag{4.5}$$

Ahora tenemos una asociación de un valor de utilidad y cada ejemplo del dataset de experiencia $Ejemplo_{Utilidad}$ que es privada para cada agente. Para seleccionar el ejemplo de mayor utilidad al que llamaremos $OfertaIdeal$, se prepara esta lista de asociaciones $Ejemplo_{i,Utilidad}$ corriendo un procedimiento de ordenamiento que los coloque de mayor a menor utilidad. Así asignamos a $OfertaIdeal$ el contenido de $Ejemplo_0$.

Para el paso de calcular el $UmbralAceptacion$, lo definimos como el valor que indica el límite inferior de utilidad que un agente esta dispuesto a considerar aceptable. Con el $UmbralAceptacion$ el agente puede comparar lo que otros agentes ofrecen como solución y decidir sobre ello. El $UmbralAceptacion$ se calcula mediante la Ecuación 4.6, que requiere la definición de un valor de holgura.

$$UmbralAceptacion = OfertaIdeal_{Utilidad} - Holgura \tag{4.6}$$

El umbral de holgura representa la capacidad del agente de aceptar ofertas que se alejan progresivamente de su $OfertaIdeal$. Para calcularla usaremos la Ecuación 4.7 y relacionaremos los siguientes conceptos:

A es la cuenta de agentes negociadores en la sesión.

R es el número de rondas posibles para la sesión de negociación, definidas por el agente arbitro.

TI es la cuenta total de los tipos de interacción posibles en la negociación. Ya que solo es posible ofertar y votar, esta variable contiene el valor 2.

EN es una estimación de las interacciones negativas posibles dada por la Ecuación 4.8.

CN es la cuenta total de las interacciones negativas que han ocurrido en la sesión de negociación. El agente que percibe la interacción la considerará como negativa cada vez que otro agente emita un voto en contra de su oferta de solución e incrementará el contador.

$$Holgura = \frac{\arg\max\{\frac{A}{TI}, CN\}}{EN} \tag{4.7}$$

$$EN = (A-1) \times R \times \frac{A}{TI} \tag{4.8}$$

Una vez calculado el valor de $UmbralAceptacion$, generamos una lista de ofertas alternativas $[OA_0, OA_1, ..., OA_n]$ a la $OfertaIdeal$ seleccionando los ejemplos del dataset de experiencia que cumplan la restricción $Ejemplo_{i,Utilidad} \geq UmbralAceptacion$.

El paso final del momento de activación de cada agente es crear los registros que permitirán monitorizar el estado del proceso de negociación. Definimos el mapa $MejorOferta_{Emisor}$ donde se registrarán las mejores ofertas de los otros agentes, el agente que recibe la oferta considerará como mejor oferta aquella que tiene mayor utilidad para su perfil de preferencias; toda vez que se encuentra una oferta mejor se reemplaza la anterior, manteniendo una única oferta por agente. Una vez que todos los agentes-módulo han concluido esta etapa de inicialización, el agente arbitro comienza la sesión de negociación.

4.4.2. Acciones de negociación

El proceso que lleva a los agentes a decidir en conjunto las configuraciones paramétricas es la negociación, la cual se desarrolla mediante la interacción de los agentes siguiendo un protocolo de negociaciones. Entre las distintas opciones para dinámica general de las negociaciones se eligió el protocolo de ofertas múltiples alternadas (vea Figura 4.6) el cual opera en rondas con dos turnos de acción para los agentes: un turno de oferta y un turno de votación sobre

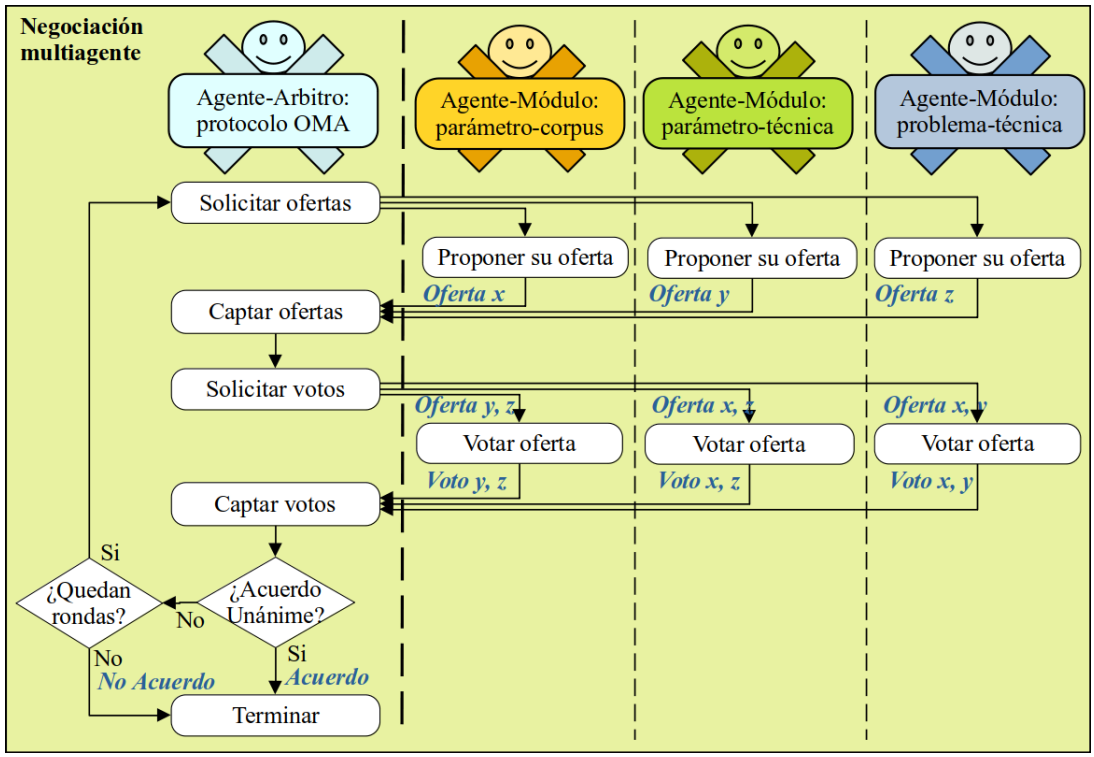

Figura 4.6: Protocolo de ofertas multiples alternadas mediante un agente arbitro.

esas ofertas. El agente arbitro es el encargado de gestionar la sesión de negociación llevando a cabo los pasos del protocolo de negociación. La Figura 4.6 ilustra el orden de acciones de los agentes, siendo el agente arbitro el que solicita acciones a los agentes negociadores y gestiona el paso de mensajes entre ellos. El agente arbitro inicia las acciones del protocolo pidiendo a los agentes negociadores una oferta, esta consiste en una propuesta de configuración. Los agentes negociadores usan un método que formula estas ofertas, una vez definida cada uno puede responder al agente arbitro. El arbitro recibe las ofertas de los agentes, las retransmite a los demás, y les indica que deben votarla. Cada agente debe decidir si las ofertas de otros agentes son aceptables o no, y en consecuencia transmitir el voto a favor o en contra al arbitro. Una vez que el arbitro captura y contabiliza los votos, detecta si alguna de las ofertas propuestas fue aceptada por todos los agentes negociadores. Si no fue así, repite la secuencia de pasos mientras se encuentre por debajo del límite de rondas permitidas.

Algorithm 4 Bidding_action

Require: $Activacin\_del\_agente, Solicitud\_de\_Oferta, VotosRondaAnterior$

1: $InteraccionesNegativas \leftarrow agregar(VotosRondaAnterior)$
2: **if** esPrimerRonda **then**
3: **return** $OfertaIdeal$
4: **else**
5: $Holgura \leftarrow calcularHolgura(InteraccionesNegativas)$
6: $UmbralAceptacion \leftarrow Ofertaideal.utilidad - Holgura$
7: $OfertasAlternativas \leftarrow \arg\{EjemplosOrdenados \geq UmbralAceptacion\}$
8: $OfertasAlternativas.dp \leftarrow distanciasPromedio(MejoresOfertasAjenas, OfertasAlternativas)$
9: **return** $\arg\min\{OfertasAlternativas.dp\}$

Algorithm 5 Voting_action

Require: $Activacin\_del\_agente, Oferta, Emisor$

1: $Oferta.utilidad \leftarrow calcularUtilidad(Oferta)$
2: **if** $MejoresOfertasAjenas[Emisor].utilidad \geq Oferta.utilidad$ **then**
3: $MejoresOfertasAjenas[Emisor] \leftarrow Oferta$
4: **end if**
5: **if** $Oferta.utilidad \geq UmbralAceptacion$ **then**
6: **return** $VotoFavor$
7: **else**
8: **return** $VotoContra$

El turno de ofertar (ver Figura 4.6 y Algoritmo 4) es la oportunidad de que un agente proponga una solución al *Caso* que es el problema de pronostico. En el primer turno se debe responder con la primer oferta de la negociación, en este punto ya se cuenta con los elementos definidos en el momento de inicialización. En este primer turno no se tienen indicadores acerca de los otros agentes, se opta por que los agentes propongan su $OfertaIdeal$ como oferta inicial. Esto tiene el beneficio de permitir que la mayor utilidad individual sea considerada para el acuerdo final. Para el resto de rondas de negociación iniciamos el procedimiento actualizando la $Holgura$ y el $UmbralAceptacion$ según las Ecuaciones 4.7 y 4.6. Con el nuevo $UmbralAceptacion$ procedemos a actualizar la lista de ofertas alternativas $[OA_0, OA_1, ..., OA_n]$ seleccionando aquellos ejemplos con $Utilidad \geq UmbralAceptacion$. De modo que la siguiente oferta que propone el agente será una seleccionada de las ofertas alternativas. Para esta selección se procede a generar una lista de distancias promedio entre todas las ofertas alternativas y cada una de las mejores ofertas de los otros agentes. Este método basado en distancia tiene una modificación que permite plasmar la percepción del agente en la toma de decisiones. La funcionalidad de las relaciones de interés es manipular el espacio en el que se encuentran las caracterizaciones de los experimentos, de modo que se minimice el impacto de las características de interés alto en el resultado de la función de distancia. Esta técnica se conoce como variables con peso, y podemos emplear los valores de preferencias para generarlos. Siguiendo las asociaciones de los valores de preferencia podemos definir una lista de escalares $Peso_{Caracteristica}$ que incremente o reduzca cada dimensión según la relación de interés. Así, el *Peso* para una característica en un descriptor de interés alto se calcula con la Ecuación 4.9 y para una característica en un descriptor de interés medio con la Ecuación 4.10. Al usar este escalar en la función de distancia euclidiana para *n* dimensiones (las características de los ejemplos) formamos la Ecuación 4.11.

$$Peso_{Alto} = 1 - VP_{Alto} \tag{4.9}$$

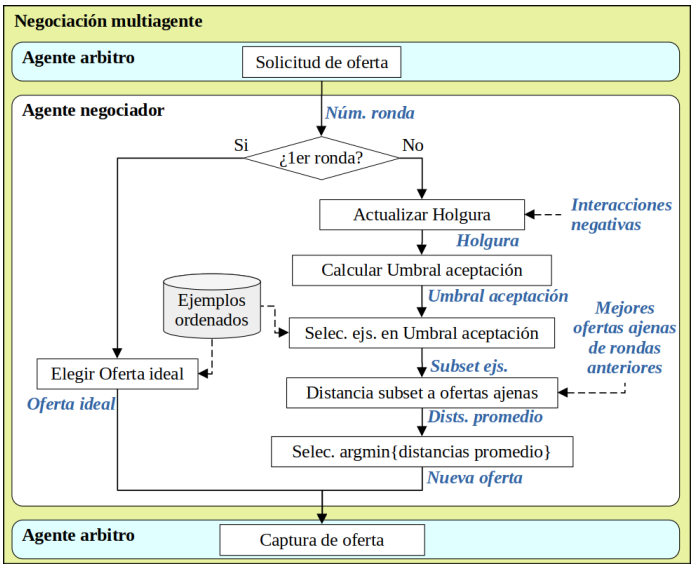

Figura 4.7: Procedimiento de generar una oferta.

$$Peso_{Medio} = 1 + VP_{Medio} \tag{4.10}$$

$$d(A,B) = \sqrt{\sum_{k=1}^{n} (Peso_k \times (A_k - B_k)^2)} \tag{4.11}$$

De esta forma al comparar el caso presentado por el usuario con la lista de ofertas alternativas cada agente percibirá como más cercanos y similares las alternativas con las dimensiones reducidas, que son las que tienen una relación de interés. En este sentido, las relaciones de interés tienen un comportamiento inverso según sea de utilidad o de distancia.

$$d(x,y) = \left(\sum_{i=1}^{i} (x_i - y_i)^p \right)^{1/p} \tag{4.12}$$

$$d(x,y) = \left(\sum_{i=1}^{i} diferencia(x_i, y_i)^p \right)^{1/p} \tag{4.13}$$

$$diferencia(a,b) = \begin{cases} 0 & Indefinido = tipoDato([a \vee b]) \\ |a-b| & Numerico = tipoDato([a \wedge b]) \\ levenshtein(a,b) & Texto = tipoDato([a \wedge b]) \end{cases} \tag{4.14}$$

$$d(x,y) = \left(\sum_{i=1}^{i} (Peso_i \times diferencia(x_i, y_i)^p) \right)^{1/p} \tag{4.15}$$

Con el ajuste de distancia con pesos, la formula de distancia queda con la forma de la Ecuación 4.15. Finalmente, el agente seleccionará la oferta alternativa que tiene menor distancia promedio (Ecuación 4.16) y la comunicará al agente arbitro.

$$NuevaOferta = \arg\min\{OA_{0,DistanciaPromedio}, OA_{1,DistanciaPromedio}, \cdots, OA_{n,DistanciaPromedio}\} \tag{4.16}$$

Una vez que el agente arbitro recopila las ofertas de cada agente negocaidor, la siguiente acción es solicitarles que voten esas las ofertas. Para decidir el voto (vea Figura 4.8 y Algoritmo 5) el agente primero valora la oferta que recibe utilizando las Ecuaciones 4.4 y 4.5. Una vez que se tiene la utilidad se compara con la mejor oferta del mismo agente y si es mayor, se actualiza $MejorOferta_{Emisor}$. Después compara la utilidad con el $UmbralAceptacion$ y si la utilidad de la oferta recibida es igual o mayor, se acepta votando 1; en caso contrario, se rechaza votando 0.

El agente arbitro recolecta los votos y detecta si algunas de las ofertas de esta ronda obtuvo el voto a favor de todos los participantes. En caso negativo, repetirá los pasos del protocolo de ofertas multiples alternadas siempre que el contador de rondas no llegue al máximo. En

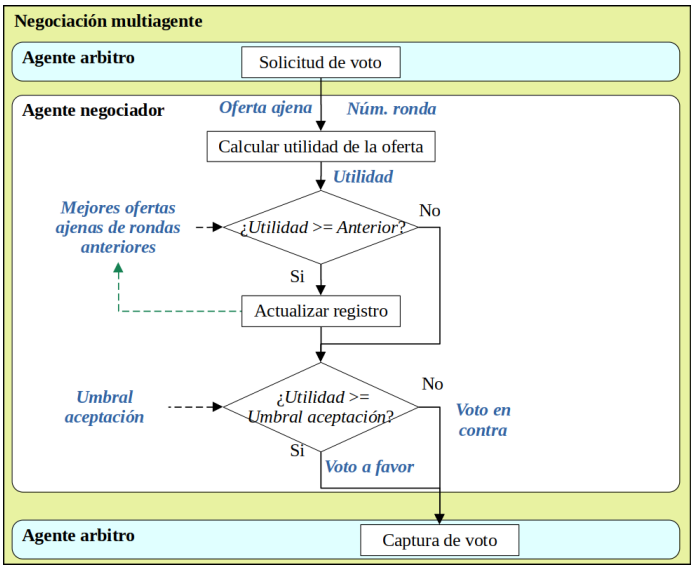

Figura 4.8: Procedimiento de decisión del voto.

caso positivo, el agente arbitro da por terminados los procedimientos del protocolo y con ello la negociación multiagente. También debe terminar el protocolo si al intentar iniciar otra ronda ya se ha alcanzado el límite de rondas. Siguiendo el esquema de la Figura 4.2, el agente arbitro debe propagar los resultados de la negociación que consisten en las ofertas y suma de votos correspondientes a la ultima ronda ejecutada. Dadas las condiciones de terminación del protocolo los resultados de la negociación pueden ser uno de dos posibles estados: acuerdo y no acuerdo. En ambos estados consideramos que los resultados de la negociación contienen las mejores configuraciones que pudieron generar los agentes, ya que serían los más próximos a la convergencia de los compromisos de los agentes. El modelo entonces ordena las configuraciones de mayor a menor cantidad de votos y procede a ejecutarlas. El modelo toma el cuerpo de datos proporcionados por el experto en química y ejecuta las configuraciones en un esquema de validación cruzada. De este modo se producen mediciones de precisión media y de desviación estándar de la precisión que se asocian a la configuración que las produce. Finalmente el modelo entrega al experto en química las configuraciones ordenadas de mayor a menor cantidad de votos junto a los resultados de las mediciones, de modo que pueda realizar la tarea de predicción con los nuevos datos.

CAPÍTULO 5

Experimentación y resultados

Este capitulo presenta el diseño experimental y su ejecución. Este diseño experimental busca, en primer momento, validar que las respuestas de los agentes son adecuadas para la tarea de pronóstico que buscan resolver. En seguida, comprobar si la calidad de las respuestas de los agentes son significativamente distintas al caso de estudio. El capitulo se compone de secciones que detallan este proceso de validación. La sección 5.1 define las pautas para la validación, las variables de trabajo y procedimientos usados en los casos de estudio que resolvieron los agentes. En la sección 5.2 se especifican los factores implicados en la generación del dataset de ejemplos que usan los agentes como experiencia previa. La sección 5.4 compara las respuestas dadas por los agentes, midiendo precisión y tiempo de las configuraciones. Se analizan los efectos obtenidos de la variación del hiperparámetro tasa de diferenciación, y se extraen recomendaciones de ellos.

5.1. Diseño experimental

El modelo de configuración paramétrica tiene dos fases, entrenamiento y pronóstico, en ambas fases se producen experimentos. Los experimentos de la fase de entrenamiento se enfocan en producir datos experimentales para alimentar la experiencia de los agentes. Esta experiencia permitirá a los agentes razonar acerca del caso de pronóstico y decidir la configuración que le parezca que resolverá mejor el caso.

En la fase de entrenamiento las experimentaciones son exploratorias, se busca producir una variedad de ejemplos experimentales que den perspectiva sobre el comportamiento de los algoritmos. Los datasets empleados son los clásicos para uso académico en clasificación, que son: vinos, iris y cáncer de mama. Estos datasets servirán como referencia para resolver los casos de estudio químicos en la fase de pronóstico. Para producir los ejemplos del corpus de experiencia se combinan las distintas opciones de configuración empezando por aquellas que modifican los datasets, de modo que se producen datasets modificados. En seguida, el generador de experimentos continua combinando opciones de configuración y por cada una realiza corridas de validación cruzada [Stone, 1974] con 5 folds. Las mediciones de desempeño de las validaciones son la precisión media y la desviación estándar de esta. Para los propósitos previstos no es necesario procesar estos resultados experimentales con pruebas que establezcan su calidad.

En la fase de pronóstico las experimentaciones van en propósito de resolver la tarea de pronóstico, de modo que se empieza por elegir los casos de estudio. Con los casos de estudio se procede a producir las consultas correspondientes y enviarlas a los agentes para comenzar el debate. Se recolectarán las respuestas que produzcan los agentes para los distintos contextos de

aplicación. Se realizarán corridas con un esquema de validación cruzada con la configuración del caso de estudio y la configuración recomendada por del modelo para producir mediciones de desempeño.

5.2. Entrenamiento del modelo

Para producir el corpus de experiencia se realizaron experimentaciones exploratorias, variando las configuraciones del algoritmo KNN y empleando distintos datasets representativos. Un análisis de los resultados de rendimiento recopilados debe producir la perspectiva necesaria para valorar las configuraciones según los aspectos de interés de cada investigador.

Para producir los experimentos se van combinando las métricas, políticas y valores de K, luego se las evalúa utilizando validación cruzada de 5 folds. Las métricas seleccionadas son la Euclidiana, Manhattan, Minkowski, Chebyshev y Mahalanobis. En cuanto a políticas para asignación de clase se seleccionaron las de Conteo de frecuencia (KNN estándar), Asignación directa (Vecino más cercano), Mayoría simple, Mayoría calificada, Vecinos con pesos. Para el valor de K se ubicó el valor más alto producido por las opciones: 5, 7, $^{2}\sqrt{n}$, además de los impares más próximos a $n^{2/8}$ y $n^{3/8}$. Este valor se empleó como limite superior, produciendo una lista desde 1 al valor más alto y generar una gama más amplia de experimentos. Los aspectos medidos son la media del porcentaje de precisión, y la desviación de la precisión. El objetivo de la implementación es crear un programa capaz de producir las experimentaciones definidas empleando distintas configuraciones del algoritmo KNN, obteniendo así los resultados experimentales exploratorios. En lo general, el generador de experimentos consiste en ciclos anidados que combinan las distintas opciones para el algoritmo KNN, haciendo que en su ciclo más interior se produzca una configuración única para el algoritmo, ejecute validación

cruzada con esa configuración y almacene los resultados. El generador de experimentos usa la librería Scikit-learn [Pedregosa et al., 2011] la cual está orientada facilitar las tareas de análisis predictivo de datos mediante implementación de algoritmos conocidos. El programa usa las librerías Numpy y Scikit-learn para manejo de datos y cargar los cuerpos de datos de iris, vino y cáncer de mama incluidos en ellas.

Las opciones de configuración y los descriptores del algoritmo giran en torno al algoritmo KNN, estas opciones se extraen de [Villarreal Hernández et al., 2023]. Las opciones de configuración implementadas se enumeran en la Tabla 5.1; para la estimación del valor de K, la columna n representa el número de registros en el dataset para el cual se está configurando K.

Tabla 5.1: Opciones de configuración implementadas para experimentación con KNN.

Estimación de variables en el dataset	Reducción de registros del dataset	Métricas de cercanía	Estimación de K	Política para asignación de clase
• "Ninguna" • Pearson • Chi-cuadrado • OneHotEncoder	• "Ninguna" • Centroides • Condensación rápida [Angiulli, 2005]	• Euclidiana • Manhattan • Minkowski $p = 1.5$ • Minkowski $p = 100$ • Chebyshev	• 7 (Sugerido) • 1 (1NN) • $\sqrt{n}$ • $n^{\frac{2}{8}}$ y $n^{\frac{3}{8}}$ [Enas, 1986]	• Conteo de frecuencia • Asignación directa • Mayoría simple • Mayoría calificada • Vecinos con pesos

A continuación se mapean las métricas de distancia y las política para asignación de clase, indicando con 0 las que admiten distintos valores de K y con 1 las opciones que solo admiten

el valor de $K = 1$. Definimos dos funciones auxiliares: $proximoImpar(n)$ toma el valor de n y retorna el número impar más próximo; la función $agregarSinRepetir(arr, n)$ añade un valor n a la lista arr si no se encuentra en arr.

Antes de realizar los experimentos, el generador de experimentos toma los datasets representativos y produce datasets modificados combinando la estimación de variables y las opciones de reducción de registros, produciendo un total de 36 datasets para la experimentación. Todas las configuraciones se probarán con estos 36 datasets modificados. Los datasets representativos empleados son los clásicos para uso académico en clasificación, que son: vinos, iris y cáncer de mama. El dataset iris cuenta con 150 registros, repartidos en 3 clases de 50 miembros, los puntos están representados en 4 dimensiones con valores de números reales positivos. El dataset vino se forma de 178 registros, tiene 3 clases con 59, 71 y 48 miembros, emplea 13 dimensiones con valores de números reales positivos. El dataset de cáncer de mama cuenta con 569 registros, repartidos en 2 clases de 212 y 357 miembros, los puntos están representados en 30 dimensiones con valores de números reales positivos.

Lo siguiente es construir los ciclos anidados que irán variando las opciones de política para asignación de clase, las métricas de distancia y los valores de K. En la parte más interna de los ciclos se agregan condiciones que ajustan las opciones del algoritmo. Lo siguiente es configurar la instancia de validación cruzada y ejecutarla. Al terminar la ejecución se genera una cadena de texto que describe la configuración del experimento y sus resultados en formato csv (Comma Separated Values) y se escribe el registro en el archivo de salida. El registro especifica el contenido de las características listadas en la Tabla 4.3, de las cuales definimos las siguientes:

- "Política" indica la política para asignación de clase que se esta usando en el experimento.

- "Métrica" indica la métrica de distancia que se esta usando en el experimento.
- "K" el valor de K empleado para el algoritmo KNN en ese experimento.
- "Precisión Media" la media del porcentaje de precisión de la validación cruzada.
- "Desviación estándar" de los resultados de la validación cruzada.
- "ID" este campo adicional es una cuenta creciente que identifica a cada experimento, con el fin de identificarlos fácilmente.

Antes de realizar cada experimento se verifica si los distintos elementos que intervienen son compatibles, de modo que solo se intentan configuraciones operantes. En caso de que la verificación lo detecte como inoperante se continúa con la siguiente iteración según los ciclos de combinaciones definidos. En general, se verifica si la política o la métrica pueden usarse con valores de K distintos a 1, también si el valor de K es menor que el número de registros del dataset de entrenamiento.

Cada experimento se agrega al dataset de ejemplos, para formar un archivo de experiencia. Con el ultimo experimento se cierran los registros del archivo de salida añadiendo una marca de tiempo al final y se escribe un archivo con los registros de todas las corridas producidas.

El generador procede a ejecutar validación cruzada con las configuraciones resultantes de combinar las opciones de las otras categorías. En las iteraciones del generador de experimentos, se descartan configuraciones inoperantes, como la coincidencia de $K = 1$ y el conteo de frecuencias, ya que esta situación funciona de la misma manera que la asignación directa. Con cada variación de configuración y dataset modificado, se produce un nuevo experimento; con las opciones mencionadas, se ejecutaron un total de 3539 experimentos con 18 dimensiones

cuyas características (vea Tabla 4.3) se almacenaron para formar el corpus de experiencia. Es decir, forman parte de la experiencia general de los agentes sobre el comportamiento del algoritmo.

5.3. Configuración de los experimentos

En la fase de pronóstico del sistema de configuración paramétrica es necesario configurar la negociación multiagente. Para el agente árbitro, definimos un máximo de 200 rondas de negociación. El árbitro también recibe los roles de los agentes configuradores, que son problema-técnica, parámetro-corpus y parámetro-técnica. El árbitro invoca a los agentes configuradores, quienes reciben su rol y sus relaciones de interés y comienzan el preámbulo de configuración.

Para desempeñar la estrategia de negociación los agentes generan sus preferencias y lista de pesos según las relaciones de interés de su compromiso. Ambos elementos dependen de la tasa de diferenciación, que se fija antes de las negociaciones. La tasa de diferenciación es un hiperparámetro que suaviza o endurece la percepción que los agentes tienen de los ejemplos de su experiencia y las ofertas que le envían los otros agentes. Para una primera perspectiva y análisis de las configuraciones encontradas se producen experimentaciones con una tasa de diferenciación fijada en 0.15. Para observar como la tasa de diferenciación afecta a la respuesta del grupo de agentes se realizaron consultas a los agentes con los distintos datasets de los casos de estudio, repitiendo las negociaciones usando distintos valores en la tasa de diferenciación.

El sistema de configuración paramétrica resuelve problemáticas de química usando la experiencia generada en la fase de entrenamiento, en la cual se experimentó con los datasets de

iris, vino y cáncer. Fue necesario analizar distintos casos de estudio químicos para seleccionar aquellas de comparación adecuada y datasets asociados disponibles. Los casos de estudio químicos seleccionados se describen a continuación. En el trabajo de Loredo [Loredo Pong, 2022] se emplea el algoritmo de *K* Vecinos Cercanos para pronosticar la gelificación de moléculas oxialquil benzoato. Este trabajo resulta adecuado para una comparación directa entre configuraciones de algoritmo ya que presenta un ejemplo de tarea de pronóstico química empleando una técnica de aprendizaje máquina. Este caso de estudio utiliza el algoritmo KNN para predecir el estado de agregación como solución (S), gel (G), precipitado (P) e insoluble (I). Este trabajo estudia una serie de oxialquil benzoatos y solventes caracterizados por los Parámetros de Solubilidad de Hansen y los carbonos en la cadena de alquilo. En el desarrollo de este trabajo, se generaron diferentes datasets de entrenamiento; sus características se enumeran en la Tabla 5.2. Para los datasets D y E, el problema se transformó en una situación de clasificación binaria, agrupando las clases como gel y no gel.

Tabla 5.2: Características de los datasets generados por Loredo.

Dataset	Variables	Población	Distribución por clase (G, I, P, S)	Tipo de dato de las columnas
A	38	240	28, 4, 7, 203	Int(1), Binary(30), Int(7)
B	24	270	47, 3, 3, 217	Binary(15), Decimal(1), Int(4), Decimal(3), Int(1)
C	11	270	47, 3, 3, 217	Int(1), Decimal(3), Int(3), Decimal(3), Int(1)
D	24	270	47, 223	Binary(15), Decimal(1), Int(4), Decimal(3), Int(1)
E	11	270	47, 223	Int(1), Decimal(3), Int(3), Decimal(3), Int(1)

También como caso de estudio tenemos el trabajo de Zamora [Zamora García Rojas, 2021] donde aborda el problema de transporte de crudos pesados, enfocándose en la mejora de sus propiedades de flujo. Una alternativa eficaz es la formulación de emulsiones W/O modificadas

con tensoactivos no iónicos (SAE10, SALE9, SALE3) y aminas como cotensoactivos (TBA, DEA, TEA) para reducir la viscosidad y optimizar su manejo. Este estudio evaluó las propiedades físico-químicas y reológicas de estas emulsiones mediante técnicas normadas ASTM, determinando su estabilidad térmica, comportamiento estructural y morfológico, así como la concentración micelar crítica. El crudo analizado mostró mejoras reológicas con la adición de tensoactivos, logrando emulsiones con viscosidad cercana a 1000 cP y comportamiento newtoniano en condiciones de transporte. Para este caso de estudio se propone encausar los datos obtenidos durante el estudio para generar un modelo de pronostico que indique la viscosidad. Los datos recopilados contienen 9512 registros con los siguientes campos descriptivos: Shear Rate (1/s), Shear Stress (Pa), Speed (1/min), Temperature (°C), Torque (µNm), Viscosity (cP). Todos los campos descriptivos son de tipo decimal. Se reemplazaron los valores continuos de viscosidad obtenida, en su lugar se usaron etiquetas que indican el rango del valor pronosticado, esta será la clase que asignará el algoritmo KNN. Los datos recopilados se etiquetaron siguiendo dos patrones. Un primer patrón centrado en 100 cP, con un margen de 30 unidades y rangos que se amplían en 300 y mil. Un segundo patrón con un primer intervalo de 0 a 200 seguido de ampliaciones en potencias de 10. Ambos patrones se detallan la Tabla 5.3. El dataset en el que se emplea el primer patrón de etiquetado le llamaremos dataset F, y al que usa el segundo patrón es el dataset G.

Tabla 5.3: Patrones para designación de etiquetas usadas en la clasificación de viscosidad.

Clase	Primer patrón	Población P.P.	Segundo patrón	Población S.P.
C1	[0, 70)	1065	[0, 200)	1322
C2	[70, 130)	155	[200, 2,000)	2172
C3	[130, 300)	198	[2,000, 20,000)	4539
C4	[300, 1,000)	673	[20,000, 200,000)	1427
C5	[1,000, ∞)	7421	[200,000, ∞)	52

Para las experimentaciones de la fase de pronóstico se evalúan tres aspectos distintos. El primer aspecto es cuanto tiempo suele requerir para dar sus resultados. El segundo aspecto es que tan frecuentemente entrega resultados correctos, indicado como la precisión media. El tercer aspecto es que tan general es la frecuencia de los resultados correctos, obtenida como la desviación estándar de los resultados de precisión. Las medidas se obtuvieron corriendo las configuraciones para clasificar en un esquema de validación cruzada con 5 folds, se midió precisión media, desviación estándar, y tiempo en segundos.

Con los datos de las mediciones es posible comparar los desempeños de las configuraciones con un método post hoc que nos indique si existen diferencias significativas y en cual sentido. Se compararon mediante la prueba de rangos de Friedman y la prueba de Holm con una significancia de 0.05. La prueba de rangos de Friedman es un método estadístico no paramétrico utilizado para analizar si existen diferencias significativas entre varios tratamientos o métodos, su objetivo es determinar si al menos uno de los tratamientos tiene un comportamiento significativamente diferente en comparación con los demás. La prueba de Holm es un método de comparación múltiple utilizado para identificar qué grupos o tratamientos son significativamente diferentes entre sí después de que una prueba global (como Friedman) haya detectado diferencias significativas. La prueba de Holm acepta o rechaza la hipótesis H_0: la media de los resultados de cada par de grupos es igual. Para ambas pruebas se utilizó la implementación en Python de STAC [Rodriguez-Fdez et al., 2015]. Dado que la prueba de Friedman está diseñada para el caso de uso de minimización, para la precisión media, se interpreta que un rango más alto es mejor calificado, mientras que para la desviación estándar y el tiempo, los rangos más bajos son mejores.

Las experimentaciones de las siguientes secciones se enfocan en el comportamiento del modelo de configuración frente a estos casos de estudio y usando distintos valores de tasa de diferenciación.

5.4. Experimentos con tasa de diferenciación 0.15

Una vez entrenado el sistema de recomendaciones debe ser capaz de encontrar las configuraciones adecuadas para los requerimientos del usuario. Las pruebas de validación del modelo de ajuste de parámetros consisten en observar la factibilidad de las respuestas de los agentes y comparar la calidad de estas con los resultados del caso de estudio. Es necesario presentar a los agentes la situación a configurar mediante las características delimitadas en la Sección 4.4. Se formularon las consultas al sistema de agentes según los datasets generados por Loredo (vea Tabla 5.4).

Los agentes generan sus preferencias y pesos guiados por el hiperparámetro tasa de diferenciación. Este modula la percepción del agente indicando que tan diferente es una característica de interés alto o medio en una configuración paramétrica. Inicialmente se fija la tasa de diferenciación en 0.15, un valor exploratorio moderadamente bajo ya que su dominio se restringe al intervalo [0,1]. De este modo podemos observar algunas respuestas preliminares de los agentes, y mas adelante se prueba con distintos valores de tasa de diferenciación para analizar su impacto en las negociaciones.

Tabla 5.4: Consultas al sistema de configuración para el caso de estudio de Loredo.

Campo característico	Consulta dataset A	Consulta dataset B	Consulta dataset C	Consulta dataset D	Consulta dataset E
Clases a considerar	4	4	4	2	2
Campos categóricos presentes	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)
Clase objetivo	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)	0 (Ninguno)
Preselección de variables	OneHot Encoder	OneHot Encoder	Ninguna	OneHot Encoder	Ninguna
Número de registros	240	270	270	270	270
Clases presentes	4	4	4	2	2
Diferencia de tamaño entre clases	82.8	88.2	88	88	88
Número de variables	38	24	11	24	11

Las configuraciones proporcionadas por los agentes se muestran en la Tabla 5.5, la primer columna identifica las distintas características de las configuraciones presentadas; la segunda columna en adelante presenta las configuraciones elegidas por los agentes. Para cada consulta de la Tabla 5.4 se obtuvo al menos una respuesta. Para el contexto del dataset A, los configuradores acordaron el ejemplo 928 del corpus de experiencia; para los contextos B, D y E, el acuerdo fue sobre el ejemplo 1633; para el contexto C, los acuerdos fueron sobre los ejemplos 851 y 1791. La información devuelta por los agentes proviene del corpus de experiencia, por lo que debe considerarse que hay partes de información que no son útiles para el usuario final o configurar el algoritmo KNN. Por ejemplo, las medidas de precisión mostradas corresponden al experimento registrado en la experiencia, y no a la obtenida de correr la configuración con el dataset de Loredo.

Tabla 5.5: Configuraciones elegidas por el sistema de configuración para el caso de Loredo.

Campo característico	ID 928	ID 1633	ID 851	ID 1791
Tipo de aprendizaje	Basado en ejemplos	Basado en ejemplos	Basado en ejemplos	Basado en ejemplos
Modificación del dataset	No	No	No	No
Tipo de asociación	Proximidad	Proximidad	Proximidad	Proximidad
Genera un modelo de generalización	No	No	No	No
Reducción de registros del dataset	Ninguna	Ninguna	Ninguna	Ninguna
Estimación de variables del dataset	Ninguna	Ninguna	OneHot Encoder	OneHot Encoder
Métrica de cercanía	Manhattan	Manhattan	Manhattan	Manhattan
Valor de parámetro K	3	3	1	1
Política para asignación de clases	Mayoría simple	Mayoría simple	Asignación directa	Asignación directa
Precisión promedio	0.7961	1	0	0.0039
Desviación estándar de precisión	0.0364	0	0	0.0078

Con cada conjunto de entrenamiento Loredo experimentó con diferentes configuraciones del algoritmo KNN, las identificaremos como Loredo + Dataset + Número de Configuración en la Tabla 5.6 (por ejemplo, LoredoA1 corresponde al dataset A con la primer configuración propuesta para ese dataset). Junto a estas, se encuentran las configuraciones propuestas por los agentes y el dataset sobre el cual deben aplicarse, indicado como ID+Número de Configuración. Esta tabla desglosa los componentes de configuración relacionados con el algoritmo KNN según [Villarreal Hernández et al., 2023], y permitiendo apreciar las diferencias entre ellas. Por ejemplo, podemos apreciar que los agentes tienden a proponer valores de K menores a los propuestos por Loredo. También, los agentes tienden a usar la política de asignación de clase por mayoría simple mientras que Loredo alterna entre conteo de frecuencia y vecinos con pesos.

Tabla 5.6: Composición de configuraciones propuestas por Loredo y por los agentes configuradores.

Dataset	Configuración del Algoritmo	Estimación de Variables	Reducción de registros	*K*	Métrica de distancia	Política para asignación
A	ID 928	Ninguna	Ninguna	3	Manhattan	Mayoría simple
	LoredoA1	OneHot Encoder	Ninguna	5	Euclidiana	Conteo de frecuencia
	LoredoA2	OneHot Encoder	Ninguna	10	Euclidiana	Vecinos con pesos
B	ID 1633	Ninguna	Ninguna	3	Manhattan	Mayoría simple
	LoredoB1	OneHot Encoder	Ninguna	3	Euclidiana	Conteo de frecuencia
	LoredoB2	OneHot Encoder	Ninguna	3	Euclidiana	Vecinos con pesos
	LoredoB3	OneHot Encoder	Ninguna	5	Euclidiana	Vecinos con pesos
	LoredoB4	OneHot Encoder	Ninguna	10	Euclidiana	Vecinos con pesos
C	ID 851	OneHot Encoder	Centroides	1	Manhattan	Asignación directa
	ID 1791	Ninguna	Centroides	1	Manhattan	Asignación directa
	LoredoC1	Ninguna	Ninguna	3	Euclidiana	Conteo de frecuencia
	LoredoC2	Ninguna	Ninguna	5	Euclidiana	Conteo de frecuencia
	LoredoC3	Ninguna	Ninguna	5	Euclidiana	Vecinos con pesos
	LoredoC4	Ninguna	Ninguna	10	Euclidiana	Vecinos con pesos
D	ID 1633	Ninguna	Ninguna	3	Manhattan	Mayoría simple
	LoredoD1	OneHot Encoder	Ninguna	5	Euclidiana	Vecinos con pesos
	LoredoD2	OneHot Encoder	Ninguna	10	Euclidiana	Vecinos con pesos
E	ID 1633	Ninguna	Ninguna	3	Manhattan	Mayoría simple
	LoredoE1	Ninguna	Ninguna	3	Euclidiana	Vecinos con pesos

Las configuraciones devueltas por los agentes (en Tabla 5.6) se probaron en un esquema de validación cruzada de 5 folds junto a las configuraciones de Loredo. Para todas las configuraciones ejecutadas se midió la precisión, la dispersión de la precisión y el tiempo en segundos. Las Tablas 5.7, 5.8, 5.9, 5.10, 5.11 muestran las mediciones ordenados por los resultados de la prueba de rango de Friedman para cada datset. Los rangos de Friedman nos indican la precedencia de la población. La ultima columna muestra el resultado de la comparación por la prueba de Holm las cuales se realizan por pares, si la prueba determina que la diferencia entre ambas poblaciones es estadísticamente significativa se rechaza la H_0. Por ejemplo en la evaluación de precisión para el dataset A (Tabla 5.6) se compara ID 928 vs LoredoA1 y les declara estadísticamente similares (H_0 Aceptada), en cambio la comparación LoredoA1 vs LoredoA2 les declara no estadísticamente similares (H_0 Rechazada).

Para el contexto del dataset A se puede apreciar en la Tabla 5.7 que la precisión de la propuesta de los agentes (ID 928) fue ligeramente mejor que las otras propuestas. Con la medición de desviación ocurre algo similar, aunque ID 928 fue la más lenta en comparación. LoredoA1 y ID 928 tienen rendimientos equivalentes en precisión y consistencia, sin diferencias estadísticamente significativas. LoredoA2 es significativamente inferior en precisión y estabilidad mientras que en tiempo, no hay diferencias importantes.

Tabla 5.7: Evaluación de las configuraciones para el caso de Loredo, dataset A.

Aspecto medido	Algoritmo	Medición	Rango F.	Holm H_0
Precisión media	ID 928	0.8776	2.6	⟩Aceptada
	LoredoA1	0.8764	2.4	⟩Rechazada
	LoredoA2	0.6048	1	
Desviación estándar	ID 928	0.0342	1.48	⟩Aceptada
	LoredoA1	0.0368	1.65	⟩Rechazada
	LoredoA2	0.0660	2.87	
Tiempo en segundos	LoredoA1	1.8741	1.68	⟩Aceptada
	LoredoA2	1.8800	1.77	⟩Aceptada
	ID 928	1.9097	2.55	

Para el contexto del dataset B (Tabla 5.8), la configuración de los agentes es ligeramente superior a las otras propuestas en términos de precisión y tiempo, y está en un segundo lugar cercano en términos de desviación. El algoritmo ID 1633 destaca en precisión y tiempo de ejecución. En precisión, solo LoredoB2 tiene un rendimiento significativamente inferior. Para desviación estándar y tiempo, no hay diferencias estadísticamente significativas entre los algoritmos. Esto sugiere que ID 1633 es el más eficiente globalmente, aunque LoredoB1 también es bastante consistente.

Tabla 5.8: Evaluación de las configuraciones para el caso de Loredo, dataset B.

Aspecto medido	Algoritmo	Medición	Rango F.	Holm H_0
Precisión media	ID 1633	0.8569	4.6	
	LoredoB1	0.8549	4.4	⟩Aceptada
	LoredoB2	0.7111	3	⟩Aceptada
	LoredoB3	0.6369	2	⟩Rechazada
	LoredoB4	0.5017	1	⟩Aceptada
Desviación estándar	LoredoB1	0.0448	2.48	
	ID 1633	0.0453	2.58	⟩Aceptada
	LoredoB2	0.0510	2.77	⟩Aceptada
	LoredoB3	0.0609	3.48	⟩Aceptada
	LoredoB4	0.0635	3.68	⟩Aceptada
Tiempo en segundos	ID 1633	2.3339	2.77	
	LoredoB2	2.3366	2.87	⟩Aceptada
	LoredoB4	2.3398	2.9	⟩Aceptada
	LoredoB3	2.3351	2.97	⟩Aceptada
	LoredoB1	2.3450	3.48	⟩Aceptada

En el contexto C (en Tabla 5.9) observamos que los agentes eligieron más de una configuración. Esto ocurre debido a que en las rondas de negociación todos los agentes hacen propuestas elegibles para acuerdo. Dado el caso en que las propuestas son suficientemente buenas para los umbrales de aceptación individuales es posible que todas las propuestas de esa ronda sean de aceptación unánime. Con la configuración actual de la negociación es con tres agentes participando, así que el máximo de propuestas que es posible acordar también es tres. En

el contexto específico del dataset C se eligieron las configuraciones ID 851 e ID 1791. Las configuraciones de los agentes con la mejor calificación en precisión y desviación fueron ID 1791, que obtuvo el segundo lugar por un margen pequeño. Mientras que la configuración ID 851 es la más rápida, tiene una baja calificación en precisión y desviación. LoredoC1 e ID 1791 tienen la mejor precisión, pero esta vez el test Holm indica que todas las diferencias son significativas, incluso entre los mejores. Destaca LoredoC4 como el de peor desempeño en precisión, pero sin penalización en estabilidad o tiempo. En desviación estándar y tiempo, no hay diferencias relevantes.

Tabla 5.9: Evaluación de las configuraciones para el caso de Loredo, dataset C.

Aspecto medido	Algoritmo	Medición	Rango F.	Holm H_0
Precisión media	LoredoC1	0.8377	5.45	
	ID 1791	0.8372	5.34	⟩Rechazada
	LoredoC2	0.8180	3.92	⟩Rechazada
	ID 851	0.8058	3.29	⟩Rechazada
	LoredoC3	0.6039	2	⟩Rechazada
	LoredoC4	0.4771	1	⟩Rechazada
Desviación estándar	LoredoC1	0.0387	2.45	
	ID 1791	0.0393	2.71	⟩Aceptada
	ID 851	0.0399	2.9	⟩Aceptada
	LoredoC2	0.0443	3.16	⟩Aceptada
	LoredoC4	0.0602	4.77	⟩Aceptada
	LoredoC3	0.0654	5	⟩Aceptada
Tiempo en segundos	ID 851	2.3358	2.32	
	LoredoC4	2.3786	2.35	⟩Aceptada
	LoredoC1	2.3962	3.52	⟩Aceptada
	ID 1791	2.3919	4.23	⟩Aceptada
	LoredoC2	2.4100	4.29	⟩Aceptada
	LoredoC3	2.4266	4.29	⟩Aceptada

En los contextos D y E (Tabla 5.10 y 5.11 respectivamente), la diferencia de precisión entre configuraciones es mayor que en los contextos anteriores. La configuración ID 1633 mejora la precisión de las propuestas anteriores con un pequeño aumento en el costo de tiempo.

Tabla 5.10: Evaluación de las configuraciones para el caso de Loredo, dataset D.

Aspecto medido	Algoritmo	Medición	Rango F.	Holm H_0
Precisión media	ID 1633	0.8788	3	⟩Rechazada
	LoredoD1	0.7422	2	⟩Rechazada
	LoredoD2	0.6314	1	
Desviación estándar	ID 1633	0.0344	1.61	⟩Aceptada
	LoredoD1	0.0491	2	⟩Aceptada
	LoredoD2	0.0554	2.39	
Tiempo en segundos	LoredoD2	2.2402	1.9	⟩Aceptada
	LoredoD1	2.2473	1.94	⟩Aceptada
	ID 1633	2.2433	2.16	

Tabla 5.11: Evaluación de las configuraciones para el caso de Loredo, dataset E.

Aspecto medido	Algoritmo	Medición	Rango F.	Holm H_0
Precisión media	ID 1633	0.8673	2	⟩Rechazada
	LoredoE1	0.7593	1	
Desviación estándar	ID 1633	0.0407	1.4	⟩Aceptada
	LoredoE1	0.0458	1.6	
Tiempo en segundos	LoredoE1	2.4975	1.4	⟩Aceptada
	ID 1633	2.5078	1.55	

En términos de precisión, en cuatro de los cinco datasets, la configuración del agente representa una mejora. Aunque esta mejora no siempre viene acompañada de una reducción en la desviación, como se observa en el contexto de los datasets B (Tabla 5.8) y C (Tabla 5.9).

Podemos notar algunos patrones que emergen de las configuraciones de los agentes. Existen casos en los que la elección de los agentes mejora en precisión sobre la de Loredo (contextos A, B, D y E), en la Tabla 5.6podemos observar sus componentes. Se observa que en todos ellos se decidió no incluir un procedimiento de estimación de variables o reducción de registros en

la configuración, lo que sugiere que para este problema particular son preferibles los datos en la forma en que fueron capturados originalmente. En cuanto al valor $K = 3$ en todas las configuraciones, se refuerza la noción de elegir un valor impar que es recomendado en la literatura. El valor de K está estrechamente relacionado con la política para asignación de clase, que es la misma en las cuatro configuraciones. La política de mayoría simple requiere que la clase mayoritaria sea al menos la mitad de los vecinos, de lo contrario devuelve una etiqueta sin clase. Así, para una clasificación exitosa, al menos dos de los tres vecinos deben ser de la misma clase; surge una estrategia interesante ya que los datasets A y B (Tablas 5.7 y 5.8) excluyen al menos una clase, reduciendo el problema de clasificación. En el caso de los datasets D y E (Tablas 5.10 y 5.11) , usar $K = 3$ evita la sensibilidad al ruido de usar solo el vecino más cercano.

En el contexto del dataset C las configuraciones de los agentes obtuvieron la calificación más baja. Se observa que este es el único caso en el que los agentes dieron más de una opción para abordar la tarea de pronóstico. También es el único caso en el que los agentes optaron por incluir opciones de estimación de variables y reducción de registros. De acuerdo con la reducción de centroides, el valor de $K = 1$ y la política de asignación directa son necesarios. La configuración ID 851 tiene calificación de precisión más baja que ID 1791 y la principal diferencia entre estas configuraciones es una estimación de variables por OneHotEncoder. Ocurre algo similar con la calificación de la desviación, esto puede indicar que usar OneHotEncoder en el dataset C es una decisión subóptima. Al observar la composición del dataset C con respecto a los otros (ver Tabla 5.4), se observa que el contexto C mantiene cuatro clases a considerar mientras que el número de variables involucradas se reduce a once.

Para los experimentos con en el caso de estudio de Zamora las consultas usadas se presentan

en la Tabla 5.12. Para el dataset F y G con una tasa de diferenciación de 0.15, los agentes decidieron unánimemente usar la configuración ID 928 que se describe en la Tabla 5.5 y la Tabla 5.6.

Tabla 5.12: Consultas al sistema de configuración con sus características para el caso de estudio de Zamora.

Campo característico	Consulta dataset F	Consulta dataset G
Clases a considerar	5	5
Campos categóricos presentes	0 (Ninguno)	0 (Ninguno)
Clase objetivo	0 (Ninguno)	0 (Ninguno)
Preselección de variables	Ninguna	Ninguna
Número de registros	9512	9512
Clases presentes	5	5
Diferencia de tamaño entre clases	2779.4242	1483.9941
Número de variables	5	5

La configuración devuelta por los agentes se probó en un esquema de validación cruzada de 5 folds, en la cual se midió la precisión, la dispersión, y el tiempo en segundos. Una vez obtenidas las mediciones se procesaron estos resultados con la prueba de rangos de Friedman y la prueba de Holm. Ya que en ambos datasets se eligió la misma configuración, el elementos distintivo de las comparaciones es el dataset de trabajo. En este caso de estudio los resultados experimentales nos indican las diferencias producidas por el etiquetado de las clases.

Se aplicaron las pruebas estadísticas de Friedman y Holm con un nivel de significancia de 0.05 para comparar dos datasets con diferentes patrones de etiquetado de clases. Se muestran los

resultados de las pruebas en Tabla 5.13, en ella observamos que todos los casos la hipótesis nula (H_0) fue aceptada, lo que indica que no se encontraron diferencias significativas en las mediciones de precisión, desviación y tiempo usando distintos los patrones de etiquetado en los datasets.

Tabla 5.13: Evaluación de la configuración ID 928 para cada dataset en el caso de Zamora.

Aspecto medido	Dataset	Medición	Rango F.	Holm H_0
Precisión media	G F	0.6158 0.6125	34.35 28.65	⟩Aceptada
Desviación estándar	G F	0.0503 0.0508	30.06 32.93	⟩Aceptada
Tiempo en segundos	G F	2774.88 2777.61	30 33	⟩Aceptada

Aunque la precisión media es casi idéntica en ambos datasets, la distribución de clases cambió notablemente. Como muestra la Tabla 5.3 en el primer patrón, la mayoría de los datos (7421 de 9512 instancias) están en C5 ($\geq$1000). En el segundo patrón, aun que la distribución es más uniforme, la mayoría de los datos están en C3 (4539 de 9512 instancias). Este cambio pudo haber afectado la dificultad de clasificación, pero la configuración ID 928 parece haber mantenido un desempeño estable ya que el impacto a la desviación no fue estadísticamente significativo (vea Tabla 5.13).

Dado que en ninguna comparación se rechazó H_0, se puede decir que el cambio en la asignación de etiquetas no tuvo un impacto estadísticamente significativo en la precisión, variabilidad o tiempo de ejecución del modelo.

A pesar de cambiar la forma en que se agrupan las clases, la precisión promedio no varió significativamente. Una posible razón por la que el cambio de etiquetas no afectó la precisión es

que las características de los datos son lo suficientemente informativas como para permitir una buena clasificación sin importar cómo se etiqueten las clases. Una segunda razón posible es que la métrica Manhattan y usar $K = 3$ sean factores que mitigan el impacto en la redistribución de clases. Otra posible explicación es que en ambos casos la mayoría de los datos se encuentran en pocas clases dominantes, por lo que el modelo sigue clasificando de manera similar. Para estos casos específicos, la precisión del KNN con la configuración ID 928 es robusta a cambios en la agrupación de clases.

5.5. Experimentos con tasa de diferenciación variada

Los agentes generan preferencias y pesos basados en las relaciones de interés de su compromiso, que puede ser interés alto o medio. El cálculo de las preferencias y pesos son influenciados por la tasa de diferenciación, un hiperparámetro que ajusta la percepción de su experiencia y las ofertas recibidas. De esta forma la tasa de diferenciación dicta que tan amplia es la distancia del interés alto y el interés medio. En la Tabla 5.14 se encuentran los cálculos para los valores de preferencia calculados con las Ecuaciones 4.1y 4.2, mientras que el peso de variable se emplean las Ecuaciones 4.9 y 4.10.

Tabla 5.14: Preferencias y pesos calculados para las tasas de diferenciación.

	Interés alto		Interés medio	
Tasa de diferenciación	Valor de preferencia	Peso de la variable	Valor de preferencia	Peso de la variable
0.15	0.063889	0.936111	0.045139	1.045139
0.20	0.066667	0.933333	0.041667	1.041667
0.25	0.069444	0.930556	0.038194	1.038194
0.30	0.072222	0.927778	0.034722	1.034722
0.35	0.075000	0.925000	0.031250	1.031250
0.40	0.077778	0.922222	0.027778	1.027778

Se repitieron las consultas de la Tabla 5.4 y se recolectaron las respuestas del sistema de configuración. En la Tabla 5.15 reunimos las configuraciones elegidas por el modelo de configuración con cada variación de la tasa de diferenciación. Las celdas resaltadas en amarillo pertenecen a negociaciones donde los agentes terminaron la sesión (200 rondas) sin llegar a un acuerdo unánime, sin embargo se rescatan las ultimas ofertas como respuesta. En el resto de los casos los agentes llegaron al acuerdo en la primer ronda de negociación. La tendencia observada es que emergen las mismas respuestas, y al incrementar la tasa de diferenciación se usan mas rondas de negociación. Se señalan en verde las configuraciones que fueron calificadas con el mejor rango Friedman (incluyendo empates) en términos de precisión. Se señalan en amarillo las configuraciones obtenidas de negociaciones donde los agentes terminaron la sesión sin llegar a un acuerdo unánime. Se detecta que 0.2 y 0.25 son valores conservadores para la tasa de diferenciación, estos dan respuestas adecuadas sin agotar el límite de rondas. Con 0.25 y mayor hay un incremento en el número de respuestas, sin embargo estas configuraciones comparten la mayoría de sus componentes con respuestas que aparecen con tasas de diferenciación menores y por tanto con menos rondas de negociación.

Tabla 5.15: Configuraciones elegidas por el modelo de configuración con cada variación de la tasa de diferenciación para el caso de Loredo.

	Tasa de diferenciación					
Dataset	0.15	0.2	0.25	0.3	0.35	0.4
A	ID 928	ID 928	ID 928	ID 928	ID 928, 707	ID 928, 707
B	ID 1633	ID 928	ID 928, 1633	ID 928, 1633	ID 928, 1633	ID 928, 707
C	ID 851, 1791	ID 1633	ID 1633	ID 1633	ID 1633	ID 1633
D	ID 1633	ID 1633	ID 1633	ID 1633	ID 1633	ID 1633
E	ID 1633	ID 1633	ID 1633	ID 1633	ID 1633	ID 1633

Ademas de las cuatro configuraciones ya mostradas en la Tabla 5.5, en las tasas de diferenciación 0.35 y 0.4 aparece la configuración ID 707 que se describe en la Tabla 5.16. Los campos característicos relacionados con la configuración paramétrica son la preselección de variables,

y los campos desde reducción de registros a política asignación de clase. De estos campos señalados podemos observar que el único que cambia en estas tres configuraciones es la preselección de variables ya que mientras los ejemplos ID 707 y ID 1633 usan OneHotEncoder el ejemplo 928 no emplea una técnica de preselección. En cuanto al resto de parámetros, ID 707 es muy similar a los ejemplos ID 928 y ID 1633 sin embargo las condiciones experimentales que usaron son distintas. Esto nos indica que individualmente los agentes prefirieron ejemplos menos parecidos en la descripción de problema que se les asignó (las consultas de la Tabla 5.4).

Tabla 5.16: Comparativa de las configuraciones ID 707, 928 y 1633 para el caso de Loredo.

Campo característico	ID 707	ID 928	ID 1633
Clases a considerar	2	3	2
Campos categóricos	0	0	0
Clase objetivo	0	0	0
Preselección de variables	OneHotEncoder	SinEstimacion	OneHotEncoder
Numero de registros	150	178	178
Clases presentes	2	3	2
Diferencia tamaño clases	72	9.39266853573	88
Numero de variables	118	13	1262
Tipo de aprendizaje	Ejemplos	Ejemplos	Ejemplos
Requiere modificar dataset	0	0	0
Tipo de asociación	Proximidad	Proximidad	Proximidad
Produce modelo de gen.	0	0 0	
Reduccion de registros	SinReduccion	SinReduccion	SinReduccion
Metrica de cercania	Manhattan	Manhattan	Manhattan
Valor de K	3	3	3
Politica asignación de clase	MayoriaSimple	MayoriaSimple	MayoriaSimple
Precision media	0.96862745098	0.91372549019	1
Desviacion estandar	0.02352941176	0.01999615495	0

La Tabla 5.17 presenta los resultados de precisión media obtenidos de la validación cruzada. Se evaluaron mediante la prueba de rangos de Friedman y se comparan mediante la prueba de Holm. Las comparaciones de Holm se realizan de todos vs todos, pero solo presentamos

los pares en el orden del rango Friedman, es decir son del algoritmo de un renglón vs el siguiente inmediato. Por ejemplo, en el contexto A, ID 928 vs ID 707 la H_0 es Aceptada: se consideran estadísticamente similares. Esto se debe a que estos algoritmos lograron la mayor precisión promedio con una ligera ventaja entre ellos. Caso similar en el contexto B con ID 1633, 928, y 707 en que lograron precisiones parecidas y apenas mayores que LoredoB3. En el contexto C la mayoría de los algoritmos obtuvieron una precisión superior al 80 %, pero la prueba de Holm nos dice que sus desempeños son significativamente distintos indicando mejoras significativas entre los algoritmos. En el contexto D las diferencias de desempeño son más claras que en el contexto C, con ID 1633 en primer lugar al igual que en el contexto E. Las precisiones son muy similares en el contexto A, mientras que en los contextos B y C hay una mayor variación entre las mejores y peores precisiones. La configuración ID 1633 se aprecia como la más consistente, logrando altas precisiones en la mayoría de los contextos. Los resultados de la Tabla 5.17 muestran que los algoritmos tipo ID alcanzan consistentemente las mayores precisiones medias en todos los conjuntos de datos evaluados, superando en varios casos de forma estadísticamente significativa a las configuraciones alternativas. En particular, algoritmos como ID 928 y ID 1633 se destacan por su robustez, presentando diferencias significativas respecto a la mayoría de las configuraciones de Loredo, según la prueba de Holm. Aunque algunas configuraciones como LoredoA1 y LoredoB1 logran desempeños competitivos en contextos específicos, la mayoría de estas variantes muestran resultados significativamente inferiores. En conjunto, la evidencia sugiere una mejora de los algoritmos elegidos por los agentes en términos de precisión media en los cinco datasets analizados.

Tabla 5.17: Resultados de precisión media, todas las configuraciones para el caso de Loredo.

Dataset	Algoritmo	Precisión media	Rango F. precisión media	Holm H_0
A	ID 928	87.67 %	3.0323	⟩Aceptada
	ID 707	87.67 %	3.0323	⟩Aceptada
	LoredoA1	87.41 %	2.9355	⟩Rechazada
	LoredoA2	60.42 %	1.0000	
B	ID 1633	84.98 %	5.7581	⟩Aceptada
	ID 928	84.98 %	5.7581	⟩Rechazada
	ID 707	84.98 %	5.7581	⟩Aceptada
	LoredoB1	84.38 %	4.7258	⟩Aceptada
	LoredoB2	72.57 %	3.0000	⟩Rechazada
	LoredoB3	63.67 %	2.0000	⟩Aceptada
	LoredoB4	49.70 %	1.0000	
C	ID 1633	83.99 %	6.6129	⟩Rechazada
	LoredoC1	83.57 %	6.1613	⟩Rechazada
	LoredoC2	81.74 %	4.7419	⟩Rechazada
	ID 851	80.44 %	3.7419	⟩Rechazada
	ID 1791	80.44 %	3.7419	⟩Aceptada
	LoredoC3	60.20 %	2.0000	⟩Rechazada
	LoredoC4	47.42 %	1.0000	
D	ID 1633	87.89 %	3.0000	⟩Rechazada
	LoredoD1	74.23 %	2.0000	⟩Rechazada
	LoredoD2	63.14 %	1.0000	
E	ID 1633	86.74 %	2.0000	⟩Rechazada
	LoredoE1	75.94 %	1.0000	

La Tabla 5.18 se dedica a la desviación estándar de la precisión media, y al igual que la Tabla 5.17 incluye todas las configuraciones generadas. En le contexto A las configuraciones ID 707 e ID 928 resultaron tener la precisiones más consistentes, con un desempeño indistinguible para la prueba de Holm. Caso similar ocurre en el contextos B y C, ya que las configuraciones ID 707, ID 928, y ID 1633 resultaron muy similares, sin embargo la diferencia con las configuraciones de Loredo es menor. Esto es especialmente notable en el contexto C dónde LoredoC1 y LoredoC2 mejoran en cierta medida a ID 851 y ID 1791. En el contexto D si

bien la medición de desviación entre ID 1633 y LoredoD1 parece decididamente distinta, la prueba de Holm nos indica que no son estadísticamente tan distintas. Tanto en el contexto C, D y E la configuración ID 1633 obtuvo el primer lugar, sin embargo la prueba de Holm no hace distinción con el segundo lugar. De nuevo, ID 1633 muestra un comportamiento consistentemente estable en todos los datasets, siendo una de las configuraciones más confiables en términos de variabilidad de la precisión. La Tabla 5.18 revela que los algoritmos ID 1633, ID 928 y ID 707, presentan de forma consistente las desviaciones estándar más bajas en todos los conjuntos de datos, lo que indica una mayor estabilidad en su desempeño. En contraste las configuraciones de Loredo (A2, B3, B4, C3 y C4) exhiben desviaciones significativamente más altas y estadísticamente significativas. Estos resultados refuerzan la superioridad no solo en precisión media, sino también en consistencia siendo soluciones más robustas para los distintos datasets evaluados.

Tabla 5.18: Resultados de desviación estándar de la precisión, todas las configuraciones para el caso de Loredo.

Dataset	Algoritmo	σ de precisión media	Rango F. σ de precisión	Holm H_0
A	ID 707	0.0373	2.1452	⟩Aceptada
	ID 928	0.0373	2.1452	⟩Aceptada
	LoredoA1	0.0412	2.3226	⟩Rechazada
	LoredoA2	0.0634	3.3871	
B	ID 707	0.0418	3.5000	⟩Rechazada
	ID 928	0.0418	3.5000	⟩Aceptada
	ID 1633	0.0418	3.5000	⟩Aceptada
	LoredoB3	0.0489	4.0161	⟩Aceptada
	LoredoB1	0.0426	4.0323	⟩Aceptada
	LoredoB2	0.0460	4.2581	⟩Aceptada
	LoredoB4	0.0587	5.1935	
C	ID 1633	0.0414	2.8387	⟩Aceptada
	LoredoC1	0.0422	3.2581	⟩Aceptada
	LoredoC2	0.0463	3.8871	⟩Aceptada
	ID 851	0.0474	3.9032	⟩Aceptada
	ID 1791	0.0474	3.9032	⟩Aceptada
	LoredoC4	0.0628	5.0968	⟩Aceptada
	LoredoC3	0.0613	5.1129	
D	ID 1633	0.0344	1.6129	⟩Aceptada
	LoredoD1	0.0491	2.0000	⟩Aceptada
	LoredoD2	0.0554	2.3871	
E	ID 1633	0.0407	1.4032	⟩Aceptada
	LoredoE1	0.0458	1.5968	

La Tabla 5.19 se dedica al rendimiento en términos de tiempo empleado en la ejecución de la configuración. El comportamiento de todas las configuraciones probadas es muy similar, no hay diferencias significativas de tiempo debido al tipo de algoritmo. Las diferencias de medición de tiempo se aprecian entre contextos distintos, esto es un comportamiento consistente esperado. La familia de algoritmos KNN tienen en común el crecimiento de operaciones en función de la cantidad de registros y variables que se incluyen en el dataset de entrenamiento. Al menos

en las configuraciones observadas, las diferencias en los componentes de las variantes KNN no parecen afectar significativamente los tiempos de procesamiento en estos contextos. La Tabla 5.19 muestra que no existen diferencias estadísticamente significativas en los tiempos de ejecución entre los distintos algoritmos, ya que en todos los casos la hipótesis nula fue aceptada. Aunque se observan ligeras variaciones en los valores absolutos (por ejemplo los algoritmos LoredoE1 y ID 1633 mostrando tiempos marginalmente mayores en el conjunto E) estas diferencias no alcanzan significancia estadística. Por lo tanto, se concluye que el tiempo de ejecución no representa un factor diferenciador relevante entre las configuraciones analizadas.

Tabla 5.19: Resultados de tiempo en segundos, todas las configuraciones para el caso de Loredo.

Dataset	Algoritmo	Tiempo en segundos	Rango F. Tiempo en segundos	Holm H_0
A	LoredoA1	1.9226	2.0000	⟩Aceptada
	LoredoA2	1.9360	2.5806	⟩Aceptada
	ID 928	1.9268	2.6452	⟩Aceptada
	ID 707	1.9014	2.7742	
B	LoredoB4	2.2585	3.2903	⟩Aceptada
	LoredoB3	2.2625	3.4194	⟩Aceptada
	ID 707	2.2555	3.4839	⟩Aceptada
	LoredoB1	2.2638	4.1935	⟩Aceptada
	LoredoB2	2.2700	4.3226	⟩Aceptada
	ID 1633	2.2744	4.4839	⟩Aceptada
	ID 928	2.2726	4.8065	
C	LoredoC3	2.3191	3.2903	⟩Aceptada
	ID 1791	2.3154	3.3871	⟩Aceptada
	LoredoC4	2.3109	3.7419	⟩Aceptada
	ID 851	2.3114	4.1290	⟩Aceptada
	LoredoC1	2.3097	4.1613	⟩Aceptada
	ID 1633	2.3196	4.6129	⟩Aceptada
	LoredoC2	2.3302	4.6774	
D	LoredoD2	2.2402	1.9032	⟩Aceptada
	LoredoD1	2.2473	1.9355	⟩Aceptada
	ID 1633	2.2433	2.1613	
E	LoredoE1	2.4975	1.4516	⟩Aceptada
	ID 1633	2.5078	1.5484	

Podemos notar que el algoritmo ID 1633 es una opción robusta que no requiere técnicas de modificación de datos, lo que refuerza su atractivo en contextos donde se prefiere mantener la integridad del dataset original. La Tabla 5.5 muestra las condiciones del entrenamiento en los que observamos un comportamiento de alta precisión y baja variabilidad para esta configuración. Estos resultados pueden deberse al uso de la distancia Manhattan y valor de $K = 3$. A diferencia de $K = 1$ (usado en ID 851 e ID 1791), que puede sobreajustarse a datos

ruidosos o atípicos, $K = 3$ parece ofrecer un equilibrio entre sesgo y varianza, proporcionando estabilidad. La distancia Manhattan suele ser menos sensible que la euclidiana a valores extremos o variaciones dominadas por una sola dimensión, lo cual puede ser útil cuando los datos tienen atributos heterogéneos o con diferente escala.

En el caso de los datasets F y G (caso de estudio de Zamora) se realizaron las consultas de la Tabla 5.12 con distintos valores de tasa de diferenciación, dando como resultado las configuraciones de la Tabla 5.20. Para las consultas F y G y las tasas de diferenciación 0.25, 0.4, 0.5, 0.6, y 0.7 se obtuvo la configuración ID 928; para las tasas de diferenciación 0.8, y 0.9 se obtuvo la configuración ID 62. Al igual que con el caso de estudio de Loredo, aumentar la tasa de diferenciación produce acuerdos distintos a los de tasas mas bajas pero muestran la tendencia a tener configuraciones similares.

Tabla 5.20: Configuraciones elegidas por el modelo de configuración con cada variación de la tasa de diferenciación para el caso de Zamora.

	Tasa de diferenciación							
Dataset	0.15	0.25	0.4	0.5	0.6	0.7	0.8	0.9
F	ID 928	ID 928	ID 928	ID 928	ID 928	ID 928	ID 62	ID 62
G	ID 928	ID 928	ID 928	ID 928	ID 928	ID 928	ID 62	ID 62

Como se observa en la Tabla 5.21) los ejemplos ID 928 y ID 62 comparten los componentes de configuración del algoritmo KNN, de modo que no es necesario realizar comparaciones entre estas configuraciones. Note que los experimentos con la configuración ID 928 se desglosan en la Sección 5.4.

Tabla 5.21: Ejemplos del sistema de configuración, con sus características, para las consultas F y G, una configuración por columna.

Campo característico	ID 928	ID 1633
Tipo de aprendizaje	Basado en ejemplos	Basado en ejemplos
Modificación del dataset	No	No
Tipo de asociación	Proximidad	Proximidad
Genera un modelo de generalización	No	No
Reducción de registros del dataset	Ninguna	Ninguna
Estimación de variables del dataset	Ninguna	Ninguna
Métrica de cercanía	Manhattan	Manhattan
Valor de parámetro K	3	3
Política para asignación de clases	Mayoría simple	Mayoría simple
Precisión promedio	0.7961	0.9372
Desviación estándar de precisión	0.0364	0.0337

CAPÍTULO 6

Conclusiones y trabajos futuros

6.1. Conclusiones

La arquitectura para la configuración de parámetros está diseñada para incorporar constantemente nuevas opciones de conjuntos de datos y algoritmos, promoviendo su adaptabilidad y evolución. Si bien llevar a cabo estudios de reutilización con familias completas de algoritmos es lo ideal, el método propuesto también admite la integración de algoritmos aislados, facilitando su implementación en escenarios específicos.

Esta arquitectura representa un marco innovador para explorar diferentes configuraciones de algoritmos predictivos. El uso del paradigma de agentes inteligentes permite evaluar estas configuraciones desde múltiples enfoques, equilibrándolos mediante consenso. Los algoritmos

de minería de datos y aprendizaje automático han demostrado ser efectivos en la predicción de tendencias y la resolución de problemas complejos en procesos químicos y de materiales [Goh et al., 2017]. Ampliar las capacidades de los agentes configuradores propuestos permitirá un despliegue más rápido, flexible y eficiente de estas técnicas, fomentando su aplicación en áreas científicas y tecnológicas avanzadas.

Los elementos de negociación multiagente en la arquitectura son altamente adaptables, lo que permite la incorporación de nuevo conocimiento mediante la captura de descriptores sin requerir el uso obligatorio del generador de experimentos. El método de generación de preferencias garantiza que, para cualquier entrada del corpus de experiencia, sea posible calcular la utilidad asociada a los compromisos. Además, el esquema de relaciones de interés añade una capa de abstracción que facilita la modificación y adición de variables o compromisos a la arquitectura, asegurando su evolución mediante estudios futuros.

Este trabajo cumple tres los objetivos específicos planteados: primero, analizar y seleccionar técnicas de minería de datos y aprendizaje automático relevantes para problemas químicos. Segundo, diseñar una metodología basada en preferencias y consenso multiagente para evaluar su desempeño predictivo. Y tercer objetivo, implementar un sistema multiagente adaptable capaz de recomendar configuraciones paramétricas óptimas en algoritmos predictivos. Estas metas se materializan en la arquitectura propuesta, la cual integra conocimiento técnico con capacidades deliberativas para su aplicación en contextos de predicción mediante clasificación.

En conclusión, el uso de agentes inteligentes en el ajuste de parámetros representa una contribución novedosa en el campo de la química computacional, ofreciendo una solución innovadora y eficiente para optimizar los modelos predictivos. Este enfoque no solo mejora el rendimiento de los algoritmos, sino que también abre nuevas oportunidades para el desarrollo de herramientas avanzadas de pronóstico y análisis.

6.2. Trabajos futuros

La arquitectura propuesta utiliza un esquema de aprendizaje perezoso que facilita la incorporación de nuevas opciones de algoritmos y tipos de conjuntos de datos, incrementando así la experiencia de los agentes configuradores. Aunque actualmente incluye un generador de experimentos que requiere la implementación directa de algoritmos, se sugiere ampliar el corpus de experiencia utilizando datos extraídos de la literatura científica. Esto permitiría incluir configuraciones y resultados previamente documentados sin necesidad de realizar experimentos adicionales. Asimismo, es posible enriquecer los roles de los agentes mediante la creación de nuevas relaciones de interés, la adición de compromisos, o la incorporación de descriptores que incrementen la diversidad de enfoques.

Desde una perspectiva experimental, una línea futura de trabajo es explorar el impacto de la selección de diferentes conjuntos de datos semilla en la generación del corpus de experiencia. Se identificaron tres estrategias para su selección que se podrían explorar: (1) conjuntos de datos directamente relacionados con el área química, cuyos descriptores permiten al sistema aprender de soluciones previas; (2) conjuntos de datos provenientes de otras disciplinas, siempre que sus descriptores sean lo suficientemente variados para ampliar las perspectivas del sistema; y (3) conjuntos de datos artificiales, asociados o no con problemas reales, que ofrecen beneficios como el control estadístico y la flexibilidad en su generación.

Otras direcciones futuras incluyen la expansión del conjunto de algoritmos soportados por la arquitectura, así como el desarrollo de estrategias de negociación que integren características como personalidades o emociones en los agentes. Incorporar estos elementos podría mejorar la dinámica de negociación y aumentar la capacidad de adaptación del sistema en escenarios más complejos.

La captura de la consulta del usuario final representa un desafío interesante, ya que describir un problema de pronóstico en términos lógicos y computables constituye un problema por sí mismo. La definición de este problema podría influir en la dificultad de encontrar configuraciones paramétricas adecuadas, dado que los algoritmos de pronóstico tienen una complejidad computacional asociada que depende de dicha configuración. Para abordar este desafío, se puede desarrollar de una interfaz asistida que permita al usuario final generar una versión simplificada y estructurada de su problema de pronóstico. Esto evitaría que el problema sea demasiado complejo o ambiguo para los agentes configuradores, mejorando la interacción entre el usuario y el sistema.

En resumen, las futuras mejoras en la arquitectura buscarán no solo ampliar la capacidad del sistema para integrar nuevas configuraciones y enfoques, sino también explorar cómo optimizar las interacciones entre los usuarios, los agentes configuradores y los problemas de pronóstico abordados. Estas líneas de investigación prometen fortalecer la utilidad y el impacto práctico de la arquitectura.

6.3. Contribuciones

Aunque existe una amplia variedad de técnicas y enfoques para resolver el problema del ajuste de parámetros, la revisión del estado del arte no identificó trabajos que aplicaran el paradigma de agentes inteligentes a este problema en el contexto de la química computacional. Este vacío en la literatura resalta la originalidad del enfoque propuesto, que busca aprovechar las capacidades de los agentes inteligentes para abordar problemas complejos de pronóstico y optimización en esta disciplina.

El ajuste de parámetros mediante agentes inteligentes permite identificar relaciones clave entre los parámetros y los problemas químicos investigados, lo cual no solo mejora la calidad de las clasificaciones y pronósticos, sino que también optimiza el uso de recursos energéticos y reduce el tiempo de cómputo requerido en las investigaciones. Este enfoque es particularmente valioso en la química computacional, donde los modelos predictivos son cada vez más complejos y demandan configuraciones precisas para obtener resultados efectivos.

Esta arquitectura incluye compromisos configurables, esquemas de preferencias por roles y estrategias de negociación basadas en agentes inteligentes. Además, se exploran conceptos avanzados como el uso de funciones de distancia mixtas y la capacidad de pronóstico del sistema, los cuales contribuyen a maximizar la flexibilidad y la adaptabilidad del modelo a diferentes escenarios. Experimentos realizados con variantes del KNN demuestran la eficacia de esta aproximación y su potencial para abordar problemas reales en química computacional y otras disciplinas científicas.

Se propuso un marco teórico que incluye un framework conceptual para identificar patrones estratégicos en la configuración de variantes del algoritmo KNN. Este framework permite segmentar, documentar y organizar la estructura de una variante de KNN mediante una arquitectura modular. Este framework se implementó y empleó como un mecanismo para automatizar la generación de experimentos con el algoritmo KNN.

A continuación se listan contribuciones y productos generados durante el desempeño de este proyecto.

- Estudio de componentes estratégicos de las distintas variantes del algoritmo KNN (sección 2.4).

- Diseño e implementación de un framework que permita configurar variantes del algoritmo KNN (sección 2.4).
- Incorporación de componentes estratégicos que permiten reproducir múltiples variantes del algoritmo KNN (sección 2.4).
- Desarrollo del entorno de negociación multiagente (sección 4.2).
- Generación de un corpus de experimentos con las variantes del algoritmo KNN en una variedad de escenarios (sección 5.2).
- Diseño de estrategias para construir y explotar el corpus de experiencia (sección 4.2).
- Diseño de una estrategia de negociaciones que incorpora percepciones alteradas por esquema de pensamiento o roles (sección 4.2).
- Esquema de percepción del agente mediante compromisos específicos basado en prioridades (sección 4.4).
- Estrategia de configuración de parámetros basada en parecido de problemas tratados (sección 4.4).
- Implementación de algoritmos que calculan las preferencias y pesos de los agentes según los compromisos asignados (sección 4.4).
- Definir mecanismo de moderación de las interacciones entre agentes mediante el hiperparámetro tasa de diferenciación (sección 4.4).
- Diseño de un método para búsqueda de soluciones mediante razonamiento de agentes y toma de decisiones por consenso (sección 4.4).
- Diseño de un esquema que permita descomponer los algoritmos predictivos en componentes configurables (sección 4.2).

- Mecanismo de abstracción entre esquemas de pensamiento y parámetros concretos (sección 4.4).
- Diseño de estrategia de negociaciones multiagentes con mecanismo de presión basado en refuerzo negativo y expectativas (sección 4.4).
- Generación de un corpus experimental del desempeño de las variantes del algoritmo KNN en distintos escenarios de trabajo (sección 5.2).
- Diseño de una metodología para la evaluación del desempeño de las respuestas del sistema de configuración (sección 5.3).
- Generación de resultados experimentales del desempeño de las respuestas del sistema de configuración (secciones 5.4 y 5.5).
- Análisis compositivo de las respuestas del sistema de configuración (secciones 5.4 y 5.5).
- Análisis del desempeño de las respuestas del sistema de configuración (secciones 5.4 y 5.5).
- Análisis estadístico comparativo del desempeño de las respuestas del sistema de configuración (secciones 5.4 y 5.5).

En este apartado se enumera el material derivado de este proyecto que ha sido publicado en medios especializados.

- Publicación de Reusability analysis of K-nearest neighbors variants for classification models", capítulo de libro en DA&CI 2023: Data Analytics and Computational Intelligence: Novel Models, Algorithms and Applications (2023).

- Publicación de "Multi-Agent Architecture for Parameter Configuration in Computational Chemistry", artículo JCR en la International Journal of Combinatorial Optimization Problems and Informatics, IJCOPI (2025).

A continuación, se detallan las instancias de comunicación académica derivadas del proyecto.

- Participación en el 10th International Workshop on Numerical and Evolutionary Optimization (NEO 2022).
- Participación en el 1er Coloquio Interinstitucional de Investigación Básica y de Frontera (UAT 2023).
- Participación en el Encuentro de Jóvenes Investigadores de Tamaulipas 2023.
- Participación “IA en acción: transformando el futuro cotidiano” durante la semana vocacional (CBTIS #164 2023).

Bibliografía

[Ahmad Basheer Hassanat et al., 2014] Ahmad Basheer Hassanat, Mohammad Ali Abbadi, and Ghada Awad Altarawneh (2014). Solving the Problem of the K Parameter in the KNN Classifier Using an Ensemble Learning Approach. 12(8):7.

[Angiulli, 2005] Angiulli, F. (2005). Fast condensed nearest neighbor rule. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pages 25–32, Bonn, Germany. ACM Press.

[Angreni et al., 2019] Angreni, I. A., Adisasmita, S. A., Ramli, M. I., and Hamid, S. (2019). Pengaruh nilai k pada metode k-nearest neighbor (knn) terhadap tingkat akurasi identifikasi kerusakan jalan. *Rekayasa Sipil*, 7(2):63.

[Avrim L Bluma and Pat Langley, 1997] Avrim L Bluma and Pat Langley (1997). Selection of relevant features and examples in machine learning. page 27.

[Baarslag, 2014] Baarslag, T. (2014). *What to bid and when to stop.* Technische Universiteit Delft, Dutch research school for Information and Knowledge Systems. OCLC: 905871136.

[Baarslag et al., 2019] Baarslag, T., Pasman, W., Hindriks, K., and Tykhonov, D. (2019). Using the Genius Framework for Running Autonomous Negotiating Parties. page 30.

[Barca et al., 2020] Barca, G. M. J., Bertoni, C., Carrington, L., Datta, D., De Silva, N., Deustua, J. E., Fedorov, D. G., Gour, J. R., Gunina, A. O., Guidez, E., Harville, T., Irle, S., Ivanic, J., Kowalski, K., Leang, S. S., Li, H., Li, W., Lutz, J. J., Magoulas, I., Mato, J., Mironov, V., Nakata, H., Pham, B. Q., Piecuch, P., Poole, D., Pruitt, S. R., Rendell, A. P., Roskop, L. B., Ruedenberg, K., Sattasathuchana, T., Schmidt, M. W., Shen, J., Slipchenko, L., Sosonkina, M., Sundriyal, V., Tiwari, A., Galvez Vallejo, J. L., Westheimer, B., Włoch, M., Xu, P., Zahariev, F., and Gordon, M. S. (2020). Recent developments in the general atomic and molecular electronic structure system. *The Journal of Chemical Physics*, 152(15):154102.

[Berman et al., 2007] Berman, H., Henrick, K., Nakamura, H., and Markley, J. L. (2007). The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. *Nucleic Acids Research*, 35(Database):D301–D303.

[Birattari, 2005] Birattari, M. (2005). *The problem of tuning metaheuristics as seen from a machine learning perspective*. PhD thesis, Universite Libre de Bruxelles.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Information science and statistics. Springer, New York.

[Buitinck et al., 2013] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

[Cerecedo Cordoba, 2020] Cerecedo Cordoba, J. A. (2020). *Neuroevolución Metanivel Aplicada a la Predicción de Características Térmicas de Líquidos Iónicos*. PhD thesis, Instituto Tecnológico Nacional de México, Instituto Tecnológico de Tijuana.

[Cover and Hart, 1967] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

[Dalitz, 2009] Dalitz, C. (2009). Reject Options and Confidence Measures for kNN Classifiers. *Schriftenreihe des Fachbereichs Elektrotechnik und Informatik der Hochschule Niederrhein*, 8:16–38.

[Demšar et al., 2013] Demšar, J., Curk, T., Erjavec, A., Črt Gorup, Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., Štajdohar, M., Umek, L., Žagar, L., Žbontar, J., Žitnik, M., and Zupan, B. (2013). Orange: Data mining toolbox in python. *Journal of Machine Learning Research*, 14:2349–2353.

[E. San Fabián, 2020] E. San Fabián (2020). *Cálculos Computacionales de Estructuras Moleculares*. Universidad de Alicante, Universidad de Alicante.

[Eduardo Fernandez et al., 2019] Eduardo Fernandez, Nelson Rangel-Valdez, Laura Cruz-Reyes, Claudia Gomez-Santillan, Gilberto Rivera-Zarate, and Patricia Sanchez-Solis (2019). Inferring Parameters of a Relational System of Preferences from Assignment Examples Using an Evolutionary Algorithm. *Technological and Economic Development of Economy*, page 23.

[Eiben et al., 1999] Eiben, E., Hinterding, R., and Michalewicz, Z. (1999). Parameter Control in Evolutionary Algorithms. 3.

[Enas and Choi, 1986] Enas, G. G. and Choi, S. C. (1986). Choice of the smoothing parameter and efficiency of k-nearest neighbor classification. *Computers & Mathematics with Applications*, 12(2):235–244.

[Fehri et al., 2019] Fehri, H., Gooya, A., Lu, Y., Meijering, E., Johnston, S. A., and Frangi, A. F. (2019). Bayesian Polytrees With Learned Deep Features for Multi-Class Cell Segmentation. *IEEE Transactions on Image Processing*, 28(7):3246–3260.

[Finocchi, 2011] Finocchi, F. (2011). Density Functional Theory for Beginners. Technical report, University Pierre et Marie Curie, Institut des NanoSciences de Paris.

[Frank et al., 2005] Frank, E., Hall, M. A., Holmes, G., Kirkby, R., Pfahringer, B., and Witten, I. H. (2005). *Weka: A machine learning workbench for data mining.*, pages 1305–1314. Springer, Berlin.

[Frisch et al., 2016] Frisch, M. J., Trucks, G. W., Schlegel, H. B., Scuseria, G. E., Robb, M. A., Cheeseman, J. R., Scalmani, G., Barone, V., Petersson, G. A., Nakatsuji, H., Li, X., Caricato, M., Marenich, A. V., Bloino, J., Janesko, B. G., Gomperts, R., Mennucci, B., Hratchian, H. P., Ortiz, J. V., Izmaylov, A. F., Sonnenberg, J. L., Williams-Young, D., Ding, F., Lipparini, F., Egidi, F., Goings, J., Peng, B., Petrone, A., Henderson, T., Ranasinghe, D., Zakrzewski, V. G., Gao, J., Rega, N., Zheng, G., Liang, W., Hada, M., Ehara, M., Toyota, K., Fukuda, R., Hasegawa, J., Ishida, M., Nakajima, T., Honda, Y., Kitao, O., Nakai, H., Vreven, T., Throssell, K., Montgomery, Jr., J. A., Peralta, J. E., Ogliaro, F., Bearpark, M. J., Heyd, J. J., Brothers, E. N., Kudin, K. N., Staroverov, V. N., Keith, T. A., Kobayashi, R., Normand, J., Raghavachari, K., Rendell, A. P., Burant, J. C., Iyengar, S. S., Tomasi, J., Cossi, M., Millam, J. M., Klene, M., Adamo, C., Cammi, R., Ochterski, J. W., Martin, R. L., Morokuma, K., Farkas, O., Foresman, J. B., and Fox, D. J. (2016). Gaussian~16 Revision B.01. Gaussian Inc. Wallingford CT.

[Gates, 1972] Gates, G. (1972). The reduced nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, 18(3):431–433.

[Goh et al., 2017] Goh, G. B., Hodas, N. O., and Vishnu, A. (2017). Deep learning for computational chemistry. *Journal of Computational Chemistry*, 38(16):1291–1307.

[Goldberger et al.,] Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. Neighbourhood Components Analysis. page 8.

[Gordon and Schmidt, 2005] Gordon, M. S. and Schmidt, M. W. (2005). Advances in Electronic Structure Theory: GAMESS a Decade Later. In *Theory and Applications of Computational Chemistry*, pages 1167–1189. Elsevier.

[Gou et al., 2012] Gou, J., Yi, Z., Du, L., and Xiong, T. (2012). A Local Mean-Based k-Nearest Centroid Neighbor Classifier. *The Computer Journal*, 55(9):1058–1071.

[Gražulis et al., 2012] Gražulis, S., Daškevič, A., Merkys, A., Chateigner, D., Lutterotti, L., Quirós, M., Serebryanaya, N. R., Moeck, P., Downs, R. T., and Le Bail, A. (2012). Crystallography Open Database (COD): an open-access collection of crystal structures and platform for world-wide collaboration. *Nucleic Acids Research*, 40(D1):D420–D427.

[Groom et al., 2016] Groom, C. R., Bruno, I. J., Lightfoot, M. P., and Ward, S. C. (2016). The Cambridge Structural Database. *Acta Crystallographica Section B Structural Science, Crystal Engineering and Materials*, 72(2):171–179.

[Gupta et al., 2016] Gupta, J. K., Adams, D. J., and Berry, N. G. (2016). Will it gel? Successful computational prediction of peptide gelators using physicochemical properties and molecular fingerprints. *Chemical Science*, 7(7):4713–4719.

[Gutiérrez, 2005] Gutiérrez, J. J. (2005). ¿qué es un framework web?

[Hall et al., 1991] Hall, S. R., Allen, F. H., and Brown, I. D. (1991). The crystallographic information file (CIF): a new standard archive file for crystallography. *Acta Crystallographica Section A Foundations of Crystallography*, 47(6):655–685.

[Han et al., 2012] Han, J., Kamber, M., and Pei, J. (2012). *Data mining: concepts and techniques*. Elsevier, Burlington, MA, 3rd ed edition.

[Hanson, 2016] Hanson, R. M. (2016). Jmol SMILES and Jmol SMARTS: specifications and applications. *Journal of Cheminformatics*, 8(1).

[Hao Li et al., 2019] Hao Li, Zhien Zhang, and Zhe-Ze Zhao (2019). Data-Mining for Processes in Chemistry, Materials, and Engineering. *Processes*, 7(3):151.

[Hart, 1968] Hart, P. (1968). The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 14(3):515–516.

[Hellenbrandt, 2004] Hellenbrandt, M. (2004). The inorganic crystal structure database (icsd)—present and future. *Crystallography Reviews*, 10(1):17–22.

[Hindriks et al., 2009] Hindriks, K., Jonker, C. M., Kraus, S., Lin, R., and Tykhonov, D. (2009). Genius: negotiation environment for heterogeneous agents. page 2.

[Hu et al., 2016] Hu, L.-Y., Huang, M.-W., Ke, S.-W., and Tsai, C.-F. (2016). The distance function effect on k-nearest neighbor classification for medical datasets. *SpringerPlus*, 5(1):1304.

[Iglesias Fernández, 1998] Iglesias Fernández, C. (1998). Fundamentos De Los Agentes Inteligentes. Technical report, Universidad Politécnica de Madrid, Madrid.

[Iswanto et al., 2021] Iswanto, I., Tulus, T., and Sihombing, P. (2021). Comparison of Distance Models on K-Nearest Neighbor Algorithm in Stroke Disease Detection. *Applied Technology and Computing Science Journal*, 4(1):63–68.

[Jacques Ferber, 1995] Jacques Ferber (1995). *Les Systèmes Multi Agents: vers une intelligence collective*. InterEditions.

[Kirdat and Patil, 2016] Kirdat, T. and Patil, V. V. (2016). Application of Chebyshev Distance and Minkowski Distance to CBIR Using Color Histogram. 2(9):4.

[Kirklin et al., 2015] Kirklin, S., Saal, J. E., Meredig, B., Thompson, A., Doak, J. W., Aykol, M., Rühl, S., and Wolverton, C. (2015). The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies. *npj Computational Materials*, 1(1):15010.

[Klaus Hechenbichler and Klaus Schliep, 2004] Klaus Hechenbichler and Klaus Schliep (2004). Weighted k-Nearest-Neighbor Techniques and Ordinal Classification. *Sonderforschungsbereich*, 386(399):17.

[Langley and Sage, 1994] Langley, P. and Sage, S. (1994). Oblivious Decision Trees and Abstract Cases. page 5.

[Le Bail, 2005] Le Bail, A. (2005). Inorganic structure prediction with GRINSP. *Journal of Applied Crystallography*, 38(2):389–395.

[Lin et al., 2014] Lin, R., Kraus, S., Baarslag, T., Tykhonov, D., Hindriks, K., and Jonker, C. M. (2014). Genius: an integrated environment for supporting the design of generic automated negotiators. *Computational Intelligence*, 30(1):48–70.

[Loredo Pong, 2022] Loredo Pong, V. (2022). Design and analysis of classification models for the gelification of alkoxybenzoates using the kNN algorithm. *International Journal of Combinatorial Optimization Problems and Informatics (IJCOPI)*.

[M. Erdem Günay et al., 2018] M. Erdem Günay, Lemi Türker, and N. Alper Tapan (2018). Decision tree analysis for efficient CO2 utilization in electrochemical systems. 28.

[Martínez González, 2001] Martínez González, J. L. (2001). Optimizacion Y Ajuste De Parametros Mediante El Metodo Simplex (Nelder-Mead). *EcosimPro user workshop*, First EcosimPro user workshop:12.

[Mehta et al., 2018] Mehta, S., Shen, X., Gou, J., and Niu, D. (2018). A New Nearest Centroid Neighbor Classifier Based on K Local Means Using Harmonic Mean Distance. *Information*, 9(9):234.

[Metcalf and Casey, 2016] Metcalf, L. and Casey, W. (2016). Metrics, similarity, and sets. In *Cybersecurity and Applied Mathematics*, pages 3–22. Elsevier.

[Michalewicz and Fogel, 2004] Michalewicz, Z. and Fogel, D. B. (2004). *How to solve it: modern heuristics*. Springer, Berlin ; New York, 2nd ed., rev. and extended ed edition.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill series in computer science. McGraw-Hill, New York.

[Molina et al., 2012] Molina, M. M., Luna, J. M., Romero, C., and Ventura, S. (2012). Meta-learning approach for automatic parameter tuning: A case study with educational datasets. In *Proceedings of the Fifth International Conference on Educational Data Mining*, volume Proceedings of the 5th International Conference on Educational Data Mining of *Proceedings of the International Conference on Educational Data Mining (EDM)*, pages 180–183.

[Muhammad Ejazuddin Syed, 2014] Muhammad Ejazuddin Syed (2014). Attribute weighting in K-nearest neighbor classification. Master's thesis, University of Tampere, School of Information Sciences.

[Mulak and Talhar, 2013] Mulak, P. and Talhar, N. (2013). Analysis of Distance Measures Using K-Nearest Neighbor Algorithm on KDD Dataset. 4(7):4.

[Pal and Shiu, 2004] Pal, S. K. and Shiu, S. C. K. (2004). *Foundations of soft case-based reasoning*. Wiley series on intelligent systems. John Wiley & Sons, Hoboken, N.J.

[P.E. Hart, 1968] P.E. Hart (1968). The condensed nearest neighbour rule.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., and Cournapeau, D. (2011). Scikit-learn: Machine Learning in Python. *MACHINE LEARNING IN PYTHON*, page 6.

[Rivera Zárate, 2009] Rivera Zárate, G. (2009). *Ajuste Adaptativo de un Algoritmo de Enrutamiento de Consultas Semánticas en Redes P2P*. PhD thesis, Instituto Tecnológico de Cd. Madero, DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN.

[Rodriguez-Fdez et al., 2015] Rodriguez-Fdez, I., Canosa, A., Mucientes, M., and Bugarin, A. (2015). STAC: A web platform for the comparison of algorithms using statistical tests. In *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8. IEEE.

[Sanner, 1999] Sanner, M. (1999). Python: A programming language for software integration and development. *J. Mol. Graph. Model*, 17(1):57–61. This is the primary citation for producing molecular graphics images.

[Sarasty, 2015] Sarasty, H. (2015). Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales.

[Shi, 2012] Shi, Y. (2012). Comparing K-Nearest Neighbors and Potential Energy Method in classification problem. A case study using KNN applet by E.M. Mirkes and real life benchmark data sets. page 23.

[Sorkun et al., 2020] Sorkun, M. C., Astruc, S., Koelman, J. M. V. A., and Er, S. (2020). An artificial intelligence-aided virtual screening recipe for two-dimensional materials discovery. *npj Computational Materials*, 6(1):106.

[Stone, 1974] Stone, M. (1974). Cross-Validatory Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133.

[Stuart J. Russell et al., 2010] Stuart J. Russell, Peter Norvig, and Ernest Davis (2010). *Artificial intelligence: a modern approach*. Prentice Hall series in artificial intelligence. Prentice Hall, Upper Saddle River, 3rd ed edition.

[Tan, 2005] Tan, S. (2005). Neighbor-weighted K-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28(4):667–671.

[Villarreal Hernández, 2019] Villarreal Hernández, J. (2019). *Desarrollo De Estrategias De Negociación Para Agentes De Software Inteligentes Influenciados Por Su Emoción Y Personalidad*. PhD thesis, Instituto Tecnológico Nacional de México - Instituto Tecnológico de Ciudad Madero, División de estudios de posgrado e investigación.

[Villarreal Hernández et al., 2023] Villarreal Hernández, J. , Morales Rodríguez, M. L., Rangel Valdez, N., and Gómez Santillán, C. (2023). Reusability analysis of k-nearest neighbors variants for classification models. In Rivera, G., Rosete, A., Dorronsoro, B., and Rangel-Valdez, N., editors, *Innovations in Machine and Deep Learning: Case Studies and Applications*, Studies in Big Data, pages 63–81. Springer Nature Switzerland.

[Wahyono et al., 2020] Wahyono, W., Trisna, I. N. P., Sariwening, S. L., Fajar, M., and Wijayanto, D. (2020). Comparison of distance measurement on k-nearest neighbour in textual data classification. *Jurnal Teknologi dan Sistem Komputer*, 8(1):54–58.

[Weininger, 1988] Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36.

[Weiss, 1999] Weiss, G., editor (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, Mass.

[Wilson, 1972] Wilson, D. L. (1972). Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3):408–421.

[Xindong Wu and Vipin Kumar, 2009] Xindong Wu and Vipin Kumar (2009). *The top ten algorithms in data mining*. Data mining and knowledge discovery series. Chapman & Hall/CRC.

[Yeung and Chang, 2006] Yeung, D.-Y. and Chang, H. (2006). Extending the relevant component analysis algorithm for metric learning using both positive and negative equivalence constraints. *Pattern Recognition*, 39(5):1007–1010.

[Young, 2002] Young, D. C. (2002). *Computational chemistry: a practical guide for applying techniques to real world problems*. OCLC: 1139500297.

[Zamora García Rojas, 2021] Zamora García Rojas, D. (2021). *Estudio matemático-experimental de emulsiones W/O utilizando modificadores para el aseguramiento de flujo*. PhD Thesis, Instituto Tecnológico Nacional de México, Instituto Tecnológico de Ciudad Madero.

[Zapata-Tapasco et al., 2014] Zapata-Tapasco, A., Pérez-Londoño, S., and Mora-Flórez, J. (2014). Método basado en clasificadores k-NN parametrizados con algoritmos genéticos y la estimación de la reactancia para localización de fallas en sistemas de distribución. (70):220–232.

Anexos

ANEXO A

Anexos.

A.1. Tablas: publicaciones que relacionan el valor de K y la métrica de cercania-similitud

La Tabla A.1.1 muestra los resultados de precisión publicados asociando métricas y valor de K. Referencia A: [Iswanto et al., 2021], B: [Angreni et al., 2019], C: [Hu et al., 2016], D: [Wahyono et al., 2020], E: [Kirdat and Patil, 2016], y F: [Mulak and Talhar, 2013].

Tabla A.1.1: Resultados de precisión publicados asociando métricas y valor de *K*.

Métrica	**Artículo**	A	B	C	D	E	F
Euclidiana	Precisión %	95.9	a) 98 b) 96 c) 98	Numérico 96 Categórico 89.5	a) 85.5 b) 85	–	97.1
	K	10	a) 1 b) 8 c) 15	Numérico 3, 4, 5 Categórico 3	a) 3 b) 1	–	1
Manhattan	Precisión %	96	–	–	a) 85.5 b) 85	–	97.8
	K	6	–	–	a) 1 b) 3	–	1
Minkowski	Valor p	N/A, 2	–	–	1.5	100	–
	Precisión %	95.9	–	–	a) 85.5 b) 85	a) 100 b) 84 c) 86 d) 84.6 e) 86.8	–
	K	10	–	–	a) 1 b) 3	a)1 b)5 c)10 d)15 e)20	–

La Tabla A.1.2 muestra los resultados de precisión publicados asociando métricas y valor de *K* (continuación). Referencia A: [Iswanto et al., 2021], B: [Angreni et al., 2019], C: [Hu et al., 2016], D: [Wahyono et al., 2020], E: [Kirdat and Patil, 2016], y F: [Mulak and Talhar, 2013].

Tabla A.1.2: Resultados de precisión publicados asociando métricas y valor de *K* (continuación).

Métrica	**Artículo**	A	B	C	D	E	F
Coseno de similitud	Precisión %	– –	– –	Numérico a) 91 b) 90 c) 89 Categórico 83	– –	– –	– –
	K	–	–	Numérico a) 1 b) 2 c) 3 Categórico 1	–	–	–
Chebyshev	Precisión %	96	–	–	a) 65.8 b) 63.6	a) 100 b) 90 c) 86 d) 84.6 e) 86.8	97.6
	K	10	–	–	a) 1 b) 13, 15	a) 1 b) 5 c) 10 d) 15 e) 20	1
Chi cuadrada	Precisión %	–	–	Categórico a) 96 b) 81	89.80	–	–
	K	–	–	Categórico a) 2, 3, 4, 5 b) 1	1	–	–

A.2. Compendio de opciones de configuración del algoritmo KNN.

La Tabla A.2.1 reúne las opciones de configuración del algoritmo KNN. Referencia A: [Muhammad Ejazuddin Syed, 2014], B: [Wilson, 1972], C: [Xindong Wu and Vipin Kumar, 2009, Pal and Shiu, 2004], D: [Enas and Choi, 1986], E: [P.E. Hart, 1968], F: [Gates, 1972], G: [Avrim L Bluma and Pat Langley, 1997], H: [Klaus Hechenbichler and Klaus Schliep, 2004, Tan, 2005], I: [Angiulli, 2005], J: [Langley and Sage, 1994], K: [Stuart J. Russell et al., 2010], L: [Zapata-Tapasco et al., 2014, Gou et al., 2012], M: [Gou et al., 2012, Mehta et al., 2018], N: [Dalitz, 2009], Ñ: [Goldberger et al.,].

Tabla A.2.1: Sumario de opciones de configuración del algoritmo KNN.

1.- Reducción de registros del cuerpo de datos	2.- Estimación de variables	3.- Métrica de cercanía-similitud	4.- Estimación del valor de K	5.- Política de asignacion de clase
Edición de Wilson (B)	Correlación de Pearson (A)	Euclidiana	Elegir $K = 1$ (A).	Conteo de frecuencia (C).
Condensación de Hart (E)	Prueba Chi cuadrada	Manhattan	$n^{2/8}$ o $n^{3/8}$ Según diferencia entre clases (D).	Vecino único más cercano (E).
Reducción de Gates (F)	Cobertura de conjuntos (G)	Minkowski	Recomendado según la métrica (en el capítulo).	Puntos con pesos (H)
Condensación rápida (I)	Algortimo OBLIVION (J)	Mahalanobis	Validación cruzada (K).	Atributos con pesos (A)
–	Análisis de Componentes Principales	Hamming	–	Distancia media (L).
	Análisis Discriminante Lineal	Levenshtein	–	Distancia mínima (M).
	Análisis de Componentes del Vecindario (Ñ)	Nucleo Gaussiano		Clasificación con rechazo (N).
		Coseno de similitud		