CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

Redes Neuronales CMAC como Modelo de Clasificación en Minería de Datos

Tesis que presenta

Palacios Hernández Francisco

Para obtener el grado de

Maestro en Ciencias

En la especialidad de

Ingeniería Eléctrica

Opción Computación

Director: **Dra. Xiaoou Li Zhang**
Co-director: **Dr. Luis E. Rocha Mier**

México, D.F., Agosto de 2006

Agradecimientos

Quiero agradecer en especial a mis padres por su apoyo incondicional, por su ejemplo y dedicación que han sido motivo de mi admiración todos estos años.

A mis hermanos y familiares por su aliento y cariño, siempre me han hecho sentir muy afortunado de tenerlos como familia.

A mis abuelitas Cari y Martha, aunque ya no estén aquí para verme se que estarían muy orgullosas de mi, gracias por todas sus enseñanzas.

A mis grandes amigos Jorge y Gil, que siempre estuvieron dispuestos a ayudarme y con quien he compartido toda esta jornada.

A mis asesores la Dra. Xiaoou Li y el Dr. Luis E. Rocha, por su colaboración en esta tesis, por su tiempo y guía para la finalización exitosa de este trabajo.

Al Dr. Wen Yu por su valiosa colaboración y comentarios en beneficio del contenido de este trabajo de tesis.

A todos mis maestros de la seccción de Computación por sus enseñanzas y a todo el personal de la sección, en especial a Sofi por su amabilidad y ayuda incondicional.

A todos mis compañeros de la sección por su ayuda y consejo.

Al CINVESTAV.

Al CONACyT.

Resumen

Los nuevos requerimientos en Minería de Datos demandan modelos con periodos de entrenamiento corto. Los modelos clásicos de redes neuronales que se vienen utilizando para la clasificación en Minería de Datos como MLP se encuentran en clara desventaja, dado que una de sus principales limitantes es el largo periodo que se requiere para su entrenamiento.

En este trabajo de tesis, proponemos como alternativa un modelo de clasificación utilizando una red neuronal CMAC, cuya ventaja es el corto periodo de entrenamiento que requiere. Dadas estas características se puede entender a CMAC como un clasificador que podría ser utilizado casi en tiempo-real. Se describen las implicaciones y consideraciones en el diseño e implementación que se tienen al utilizar a la red CMAC como modelo de clasificación en el área de Minería de Datos.

Para evaluar el desempeño de la alternativa propuesta, se realizaron una serie de casos de estudios en donde se lleva a cabo la comparación con otros modelos bien conocidos de clasificación utilizando bases de datos de la vida real. Al final los resultados obtenidos nos permitieron identificar el buen desempeño que presentan las redes CMAC como modelo de clasificación en Minería de Datos.

Abstract

The new requirements in Data Mining demand models with short training periods. Classical models of neural networks used for classification in Data Mining such as MLP, they are in clear disadvantage, because their training process is very slow.

In this thesis, we propose an alternative classification model, which uses a CMAC neural network. The main advantage of CMAC is its short training period. In this way CMAC can be understood as a classifier that can be used almost in real-time. We discuss the design and implementation of CMAC as a classification model.

In order to compare CMAC's performance with other well-known classification models, we performed a series of case studies with real-life datasets. The obtained results show that CMAC is a viable model for classification in Data Mining.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1.

Introducción

El avance de la tecnología en los últimos tiempos, el crecimiento de los mercados y de las propias sociedades en sí, ha acarreado entre sus múltiples consecuencias, la generación de grandes volúmenes de información.

En cada ámbito del quehacer humano la cantidad de información recolectada es enorme, pero aún más grande es la necesidad de obtener ventaja y conocimiento útil a partir de ella. Existen muchas cosas que se desearía conocer a partir de los datos con que cuenta un negocio o industria. Por ejemplo, en una campaña de mercadotecnia nos interesaría conocer por anticipado a aquellos clientes que responderán a la promoción de un producto en específico. Los beneficios económicos que se obtendrían al tener información de este tipo claramente serían considerables. Es por ello que la necesidad de obtener información útil a partir de los datos con que se cuenta es un punto de gran interés en nuestros tiempos [6].

1.1. Antecedentes

Al ser la Minería de Datos el proceso que nos permite descubrir nuevas y útiles correlaciones, patrones y tendencias dentro de grandes cantidades de datos, el interés en ella por parte de campos económicos, sociales y de investigación la han llevado a ser considerada como uno de los desarrollos más revolucionarios de la siguiente década [23]. Además, la Minería de Datos se proyecta como una de las diez tecnologías emergentes que cambiarán al mundo en los próximos años [2].

La clasificación es una de las principales aplicaciones de la Minería de Datos para el análisis y modelado de los datos. Dada su importancia, la clasificación es una de las técnicas de Minería de Datos más estudiada encontrándose en la literatura una gran cantidad de trabajos y desarrollos, que describen nuevos modelos o estudios acerca del

comportamiento de los mismos.

De este modo la clasificación es una técnica de análisis de datos que puede ser utilizada para la extracción de modelos que describan a las clases dentro de un conjunto de datos. Diferentes métodos de clasificación han sido propuestos por investigadores de distintos campos como inteligencia artificial, sistemas expertos, estadística y neurobiología [24].

Más recientemente, un nuevo requerimiento en torno a la obtención de conocimiento a través del uso de Minería de Datos ha tomado fuerza. Así, se ha manifestado una clara necesidad de realizar Minería de Datos en aplicaciones que deben cumplir con restricciones tiempo. Así surge el término de Minería de Datos en tiempo-real, que se refiere a realizar el proceso de Minería de Datos pero cumpliendo adicionalmente con compromisos en cuanto al tiempo de respuesta.

Dada esta nueva restricción, los modelos tradicionales no han sido del todo capaces de presentar una solución adecuada. Por ejemplo, en clasificación, el modelo de red neuronal más ampliamente utilizado, conocido como MLP, tiene como desventaja importante que su tiempo de entrenamiento es largo, lo que lo hace inadecuado para aplicaciones donde el principal requerimiento es la velocidad en la obtención de la información.

Es por eso que en esta tesis se presenta una nueva alternativa para resolver el problema asociado al largo periodo de entrenamiento requerido por los modelos clásicos de redes neuronales utilizadas para clasificación en Minería de Datos.

1.2. Trabajo Relacionado

En la literatura se pueden encontrar muchos ejemplos de la aplicación de las redes neuronales en el campo de Minería de Datos. Las redes neuronales proporcionan como gran ventaja y atractivo a la Minería de Datos su gran poder predictivo, aunque de igual forma presentan como principal desventaja la poca claridad e interpretabilidad del modelo generado. La mayoría de los trabajos en donde se utilizan redes neuronales en aplicaciones de Minería de Datos emplean el modelo de redes multicapas con retropropagación (MLP).

En [19] se presenta un enfoque combinado de redes neuronales, conjuntos difusos y algoritmos genéticos para la clasificación de donadores en una campaña de caridad. El

enfoque combinado de las tres técnicas busca reducir la desventaja de las redes neuronales en cuanto a la obtención de reglas. El modelo de redes neuronales empleado en este trabajo es la red MLP.

Otro trabajo relacionado con el uso de redes neuronales y su aplicación en Minería de Datos se presenta en [30]. En este trabajo, a diferencia de algunos otros, más allá de tener como principal objetivo el determinar qué personas responderán a la invitación de enviar su donativo, busca seleccionar a aquéllos cuyo monto de donación sea mayor. Así mismo, este trabajo no sólo es un ejemplo del uso de redes neuronales para este tipo de problemas, sino es más bien un estudio comparativo de distintas técnicas de clasificación en Minería de Datos como lo son: los árboles de decisión, métodos de regresión, redes neuronales y conjuntos difusos. De la misma forma que en el trabajo previo, el modelo de red neuronal empleado es MLP por considerarlo un aproximador universal, lo que significa que teóricamente es capaz de aproximar cualquier función continua a cualquier grado de precisión.

La combinación de técnicas de clasificación en busca de mejorar el desempeño individual de cada una de ellas, ha sido una alternativa comúnmente empleada. En [21] se presenta una combinación de redes neuronales y árboles de decisión para abordar el problema de la clasificación de los clientes de un banco y así poder otorgarles un crédito determinado. Al igual que en los trabajos previamente descritos el modelo neuronal empleado es MLP.

Si bien MLP es un modelo con un buen desempeño en cuanto a la precisión de los resultados generados, tiene como una de sus principales desventajas el largo periodo de entrenamiento requerido. Es por ello que en este trabajo se pretende evaluar el desempeño de una alternativa de red neuronal cuya principal ventaja bien conocida es la rapidez de entrenamiento que presenta. Este modelo es conocido como CMAC [5] y ha sido ampliamente utilizado en el área de control, precisamente como una alternativa al modelo clásico MLP y a su limitante de emplearse en aplicaciones que requieren de un entrenamiento en tiempo-real [32].

1.3. Motivación

La acumulación de grandes cantidades de datos en los ámbitos del desarrollo humano y la necesidad de herramientas que de manera automática analicen estos datos en busca de información útil o la obtención de modelos que nos proporcionen un soporte para la

toma de decisiones más adecuadas es un demanda global que va en crecimiento.

La clasificación es una las tareas más solicitadas dentro de la Minería de Datos, debido a que los beneficios que proporciona se pueden traducir fácilmente en beneficios económicos.

Dada la nueva tendencia de la Minería de Datos de aplicarse en áreas donde se tienen compromisos de tiempo, es clara la necesidad de nuevos modelos cuyo periodo de entrenamiento sea corto.

Como ya mencionamos, el modelo CMAC es una red neuronal bien conocida en control por sus características de rápido aprendizaje. Por ello resulta atractivo evaluar su desempeño sobre el tipo de problemas que se presentan en clasificación en el área de Minería de Datos.

1.4. Objetivos

Los objetivos principales de esta tesis son:

- Desarrollar una implementación de la red neuronal CMAC que pueda ser aplicada a la clasificación en Minería de Datos.
- Evaluar el desempeño de la red CMAC implementada en el tipo de problemas clásicos de la clasificación en Minería de Datos.
- Analizar los resultados del modelo de clasificación CMAC y así contar con más información acerca de que tan buena alternativa es al modelo clásico MLP utilizado ampliamente en el campo de la Minería de Datos.

1.5. Organización

El resto del contenido de esta tesis esta organizado de la siguiente manera:

En el capítulo 2 se proporcionan los conceptos básicos de Minería de Datos y clasificación, los nuevos requerimientos del área y las características de los modelos clásicos utilizados en clasificación.

En el capítulo 3 se describe el modelo de red neuronal CMAC, sus características, funcionamiento, ventajas y desventajas con otros modelos de clasificación. Además de mostrar su aplicación como aproximador de funciones.

En el capítulo 4 se detallan los aspectos de implementación a tomar en cuenta para aplicar la red CMAC a la tarea de clasificación en Minería de Datos.

En el capítulo 5 se presentan los casos de estudio sobre los cuáles se evaluó el desempeño de la red CMAC para la clasificación en Minería de Datos y su comparación con otros modelos.

En el capítulo 6 se presentan las conclusiones acerca de la aplicación de CMAC a la clasificación en Minería de Datos. También se mencionan los posibles trabajos futuros a realizar entorno al tema presentado en este trabajo.

Capítulo 2.

Minería de Datos y Clasificación

De acuerdo con el Gartner Group [1], "La Minería de Datos es el proceso de descubrir nuevas y útiles correlaciones, patrones y tendencias dentro de grandes cantidades de datos almacenadas en repositorios, utilizando tecnologías para el reconocimiento de patrones así como técnicas matemáticas y estadísticas". También existen otras definiciones:

> "Minería de Datos es el análisis de conjuntos de datos (comúnmente grandes) de observaciones para encontrar relaciones inesperadas y presentar los datos en formas que sean tanto entendibles como útiles para el dueño de la información [14]".

> "Minería de Datos es un campo interdisciplinario que conjunta diferentes técnicas desde inteligencia artificial, reconocimiento de patrones, estadística, bases de datos y visualización (figura 2.1) para realizar la extracción de información dentro de grandes cantidades de datos [8]".

Se prevé que la Minería de Datos sea "uno de los desarrollos más revolucionarios de la siguiente década" de acuerdo con [23]. Así también dentro del MIT Technology Review [2], se elige a la Minería de Datos como una de las diez tecnologías emergentes que cambiarán al mundo.

Existen una gran cantidad de áreas en donde se puede aplicar la Minería de Datos, algunas de ellas se muestran a continuación:

- Medicina: Determinar si se padece alguna enfermedad y la efectividad de los tratamientos, a través de analizar el historial clínico del paciente para encontrar alguna relación entre sus padecimientos.
- Industria farmacéutica: Identificar nuevos medicamentos.

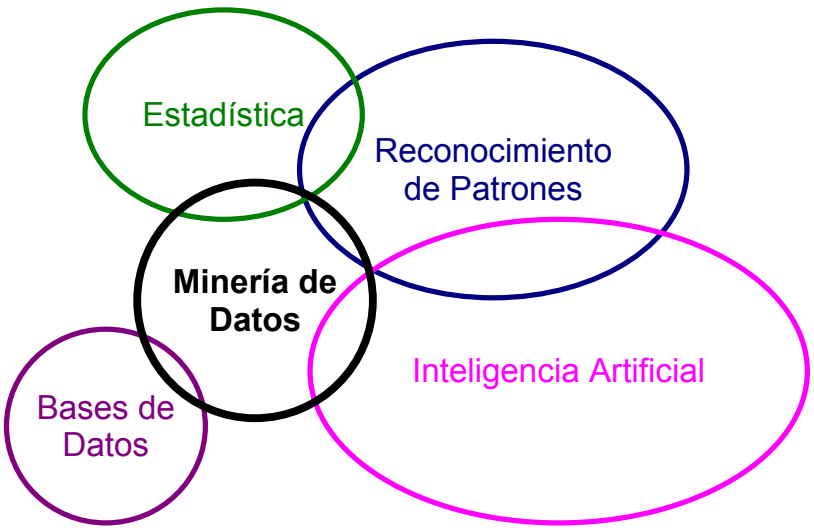


Figura 2.1.: La Minería de Datos es un campo multidisciplinario (adaptado de [45]).

- Biometría: Identificación de personas a través de imágenes, huellas digitales o bases de datos de voz.
- Multimedia: Búsqueda e identificación de imágenes, video, voz y texto dentro de una base de datos multimedia.
- Análisis científico: Identificar nuevas galaxias a través de la búsqueda de cúmulos.
- Mercadotecnia: Distinguir grupos de consumo y sus hábitos, para así diseñar campañas de mercadotecnia mejor enfocadas.
- Legal: Búsqueda y acceso a datos históricos de juicios con casos similares.
- Utilización del suelo: Identificar las áreas con utilización del suelo similar en una base de datos de observaciones de la Tierra.

2.1. Motivación de la Minería de Datos

Imaginemos que mientras esperamos en la fila de un supermercado, cerramos los ojos y escuchamos atentamente a nuestro alrededor. Además de las múltiples conversaciones

y los innumerables sonidos, se escucha el beep, beep, beep de los scaners de las cajas leyendo los códigos de barras de los productos. Esta información, además de contabilizarse en la propia caja registradora, se almacena en los servidores de la cadena de supermercado. Cada beep se traduce en un nuevo registro en la base de datos y más aún se traduce en información acerca de los hábitos de consumo de cada familia que realiza sus compras en ese supermercado (adaptado de [24]).

Es claro que la cantidad de información recolectada es enorme. Pero, ¿Qué es lo que se está aprendiendo sobre el negocio a partir de esta información?, ¿Qué ventaja se obtiene de esta información?. En muchos casos la respuesta a estas preguntas nos muestra que muchas organizaciones no están aprovechando adecuadamente la información de sus negocios. En 1984, John Naisbitt mencionó "Nos estamos inundando en información, pero nos encontramos hambrientos de conocimiento"[33]. Así hoy en día el problema no radica en la falta de información, porque realmente estamos inundados de ella, sino en la falta de analistas lo suficientemente capacitados para traducir toda esta cantidad de datos en conocimiento.

El crecimiento del campo de la Minería de Datos y del Descubrimiento de Conocimiento se ha visto impulsado por una serie de factores:

- El crecimiento exponencial de datos recolectados en las empresas.
- El almacenamiento de los datos en Data Warehouses, de modo que la empresa entera tenga acceso a una base de datos actual y confiable.
- La creciente disponibilidad de información sobre la navegación en Internet.
- La presión del mercado de ser más competitivos en una economía globalizada.
- El desarrollo de herramientas comerciales para llevar a cabo la Minería de Datos.
- El enorme crecimiento en el poder de cómputo y en la capacidad de almacenamiento.

2.2. El Proceso de la Minería de Datos

En múltiples ocasiones se trata de reinventar la rueda respecto a la manera de realizar el proceso de Minería de Datos lo que en muchos de los casos conduce a un doble esfuerzo. De tal manera que es clara la necesidad de un estándar general, sin considerar herramientas o aplicación que guíe el proceso de la Minería de Datos. Así, en 1996 fue

desarrollado el Cross-Industry Standard Process for Dataming (CRISP-DM) [9] como un esfuerzo conjunto de DaimlerChrysler, SPSS y NCR para proveer un estándar libre para el proceso de la Minería de Datos.

De acuerdo con CRISP-DM, todo proceso de Minería de Datos tiene un ciclo de vida que consiste de seis fases, como se muestra en la figura 2.2. Nótese que la secuencia de fases es adaptiva. Es decir que la siguiente fase en la secuencia depende de los resultados obtenidos en la fase anterior. Las dependencias más importantes entre fases se indican con flechas. Por ejemplo supongamos que nos encontramos en la fase de modelado. Dependiendo del comportamiento y características del modelo, tal vez sea necesario regresar a la fase de preparación de datos y realizar algunas correcciones o ajustes antes de continuar con la fase de evaluación del modelo.

La naturaleza iterativa de CRISP se representa por el ciclo exterior que se muestra en la figura 2.2.

Las seis fases de CRISP-DM son:

1. Fase de Comprensión del Negocio

   - Describir los objetivos del proyecto, así como establecer claramente los requerimientos.
   - Trasladar estas metas y restricciones en la definición del problema de Minería de Datos.
   - Preparar una estrategia preliminar para lograr los objetivos definidos.

2. Fase de Comprensión de los Datos

   - Recolectar los Datos.
   - Evaluar la calidad de los datos.

3. Fase de Preparación de los Datos

   - Preparar a partir de los datos iniciales el conjunto de datos que se utilizará en el resto de las siguientes fases. Esta parte es tal vez la más laboriosa del proceso de Minería de Datos.

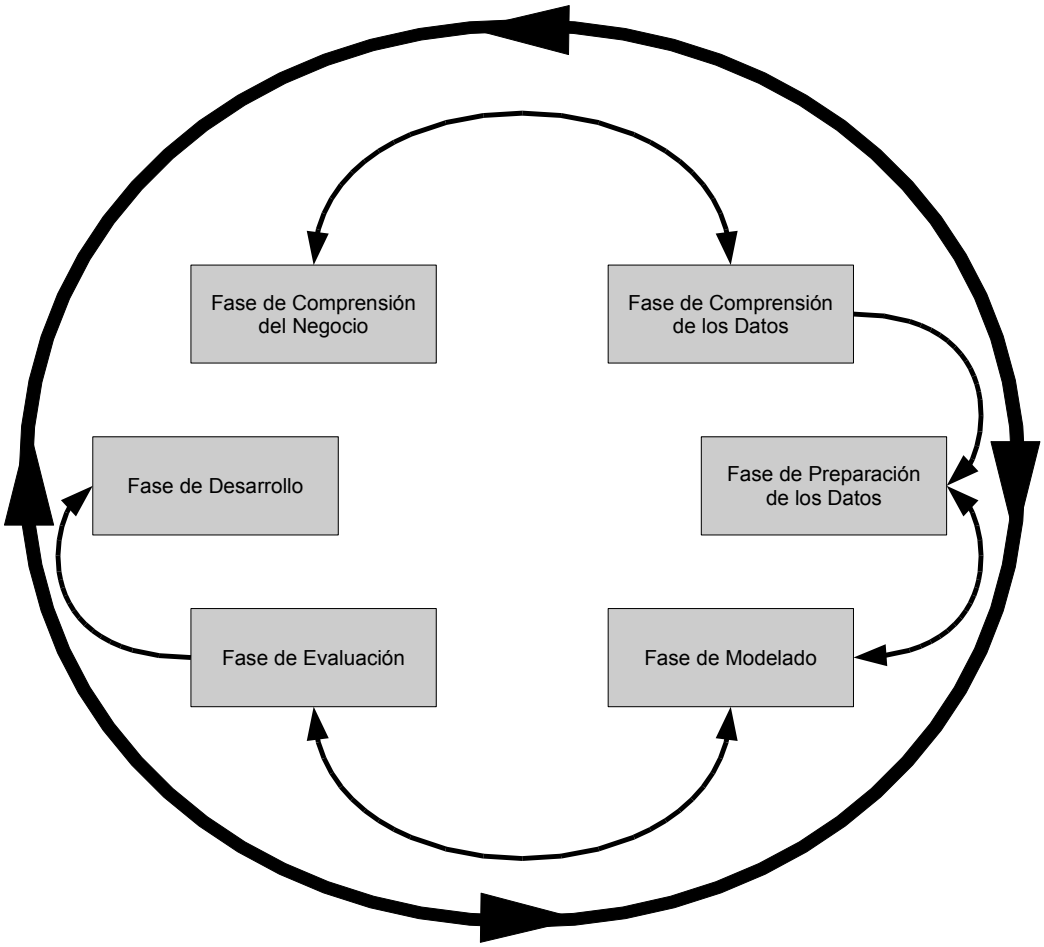


Figura 2.2.: CRISP-DM.

- Seleccionar los casos y variables que se desea analizar y que son los más apropiados para el análisis.
- Realizar las transformaciones necesarias sobre las variables requeridas.

4. Fase de Modelado

   - Seleccionar y aplicar las técnicas de modelado adecuadas.
   - Ajustar los parámetros del modelo para obtener los resultados óptimos.
   - Tener en cuenta, que regularmente se utilizan diferentes técnicas para el mismo problema de Minería de Datos.
   - Si es necesario, se debe regresar a la fase de Preparación de los Datos para contar con los datos adecuados para cierta técnica de Minería de Datos requerida.

5. Fase de Evaluación

   - Evaluar el o los modelos generados durante la fase de modelado.
   - Determinar si el modelo realizado consigue los objetivos planteados en la primera fase.

6. Fase de Desarrollo

   - Hacer uso de los modelos creados: La creación del modelo no significa la terminación del proyecto.
   - Realizar un ejemplo sencillo: Generar un informe de la prueba.

2.3. Minería de Datos en Tiempo-Real

La Minería de Datos se ha enfocado mayormente hacia aplicaciones analíticas que no requieren de una respuesta en tiempo real. Sin embargo, en la actualidad existe una clara necesidad de realizar Minería de Datos para aplicaciones que deben cumplir con determinados compromisos de tiempo [44] [43] [27]. Por ejemplo, una agencia gubernamental posiblemente necesite determinar en qué lugar se presentará algún tipo de actividad terrorista, pero una condición importante para que esta información sea

de utilidad es el periodo de tiempo en el cual sea posible obtener dicha respuesta. Es por eso que en la actualidad existe una gran demanda de herramientas y técnicas para la Minería de Datos en tiempo-real. Imaginemos como otro ejemplo una aplicación médica en donde es necesario realizar el análisis de la imágenes provenientes de determinado equipo médico para que el cirujano pueda realizar la operación adecuada. Es claro que el análisis de las imágenes en un ambiente como éste se debe realizar en tiempo-real.

La Minería de Datos en tiempo-real debe cumplir con una serie de compromisos adicionales a la Minería de Datos común. Es por ello que se presentan una serie de modificaciones o restricciones en su ciclo normal. La figura 2.3 muestra el ciclo de la Minería de Datos en tiempo-real.

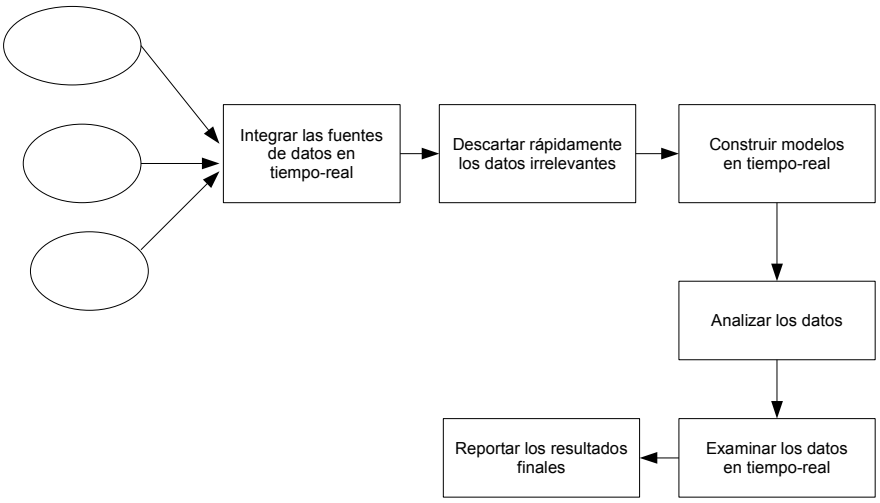


Figura 2.3.: El ciclo de la Minería de Datos en tiempo-real.

2.4. Técnicas de la Minería de Datos

Las principales técnicas que utiliza la Minería de Datos para resolver la mayoría de los problemas de investigación, economía y negocios se muestran en la siguiente lista:

- Descripción
- Estimación

- Predicción
- Clasificación
- Clustering
- Asociación

Descripción

En muchas de las ocasiones, los investigadores o analistas se encuentran tratando de encontrar únicamente diferentes formas o maneras de describir los patrones o tendencias dentro de los datos. Por ejemplo, un análisis de este tipo puede evidenciar que aquellas personas cuya situación financiera actual no es favorable tenderían a no apoyar al nuevo candidato presidencial del partido en el poder. Las descripciones de los patrones y tendencias comúnmente sugieren una explicación para dichos patrones o tendencias.

Los modelos de Minería de Datos deberían ser lo más claros posibles. Esto es, los resultados del modelo de Minería de Datos deberían describir claramente los patrones que sean más favorables para la explicación e interpretación intuitivas. Algunos métodos o modelos de Minería de Datos son más favorables para obtener una interpretación mucho más clara. Por ejemplo, los árboles de decisión proporcionan una explicación de sus resultados mucho más amigables e intuitiva para las personas. Por otra parte, las redes neuronales, son menos claras para aquellos que no son especialistas, debido a la complejidad del modelo.

Estimación

La estimación es muy parecida a la clasificación excepto porque la variable objetivo es numérica en lugar de categórica. Los modelos se construyen utilizando los registros completos, incluyendo el valor de la variable objetivo como se utiliza en los predictores.

Así, para nuevas observaciones se estima el valor de la variable objetivo basados en los valores de los predictores. Por ejemplo, se podría estar interesado en predecir la presión sanguínea de un paciente, basados en datos como su edad, sexo, índice de masa corporal y niveles de sodio en la sangre. La relación entre la presión sanguínea y las variables de predicción en el conjunto de datos de entrenamiento nos proporcionarán un modelo de estimación que se podrá aplicar a nuevos casos.

Predicción

La predicción es muy similar tanto a la clasificación como a la estimación, excepto porque en predicción, los resultados se calculan a futuro. Entre algunos de los ejemplos de la aplicación de predicción en campos como investigación o negocios se incluyen:

- Predicción de precios en los próximos tres meses.
- Predicción del aumento en el porcentaje de muertes por accidentes de tráfico en el siguiente año si se aumenta el límite de velocidad.
- Predecir si el descubrimiento de una nueva molécula conducirá al desarrollo de un nuevo medicamento que proporcione ganancias a la compañía farmacéutica.

Clasificación

En clasificación existe una variable objetivo categórica como *nivel de ingreso*, la cual, por ejemplo se podría dividir en tres clases o categorías: ingreso alto, ingreso medio y bajo ingreso. El modelo de Minería de Datos analiza un gran número de registros. Cada registro contiene información sobre la variable objetivo así como de un conjunto de variables de entrada. Por ejemplo, consideremos el extracto de datos mostrado en la tabla 2.1.

Suponga que se desea clasificar el ingreso de las personas que no se encuentran registradas en la base de datos, basados en algunas características de la persona como edad, sexo y ocupación. Para realizar esta clasificación, el algoritmo debe primero examinar el conjunto de datos que contiene tanto las variables de entrada como la variable objetivo. En este sentido el algoritmo aprende cuales combinaciones de variables están asociadas con cada nivel de ingreso. Así, cuando el algoritmo recibe un nuevo registro para el cual se desconoce el nivel de ingreso, éste deberá de asignarle una categoría. Por ejemplo, una mujer de 63 años que se dedica a la enseñanza superior posiblemente sea clasificada con un nivel de ingreso alto.

Algunos ejemplos de aplicaciones de la clasificación son:

- Determinar cuando una transacción de tarjeta de crédito es fraudulenta.
- Determinar cuando una nueva solicitud de crédito hipotecario tiene un alto o bajo nivel de riesgo.

- Diagnosticar si se padece una enfermedad en particular.

Cliente	**Edad**	**Sexo**	**Ocupación**	**Nivel de ingreso**
001	47	F	Ingeniero de Software	Alto
002	28	M	Consultor de Marketing	Medio
003	35	M	Desempleado	Bajo
.				
.				
.				

Tabla 2.1.: Extracto del conjunto de datos para clasificar el nivel de ingreso.

Clustering

Clustering se refiere a la agrupación en clases de registros, observaciones o casos similares. Un cluster es una colección o conjunto de registros que son similares entre sí y diferentes a los registros en otros clusters. Clustering se diferencia de la clasificación en el hecho de que no existe una variable objetivo. Clustering no trata de clasificar, estimar o predecir el valor de una variable objetivo. Los algoritmos de clustering buscan cómo segmentar todo el conjunto de datos en subgrupos o clusters relativamente homogéneos, maximizando la similitud de los registros dentro del clúster y minimizando la similitud de los registros fuera del clúster.

Asociación

La asociación en Minería de Datos se refiere a encontrar cuáles atributos van juntos. En el mundo de los negocios se le conoce como análisis de afinidad. La asociación se encarga de descubrir reglas que cuantifiquen la relación entre dos o más atributos. Las reglas de asociación son de la forma "Si *antecendente*, entonces *consecuente*" acompañadas de una medida del nivel de soporte y confianza asociado a la regla. Por ejemplo, un supermercado podría descubrir que 200 de 1000 de sus clientes que acuden a comprar los jueves por la noche compraron pañales y de éstos, también 50 compraron cerveza. Así, la regla de asociación debería ser "Si compra pañales, entonces compra cerveza" con un soporte de $200/1000 = 20\,\%$ y una confianza de $50/200 = 25\,\%$.

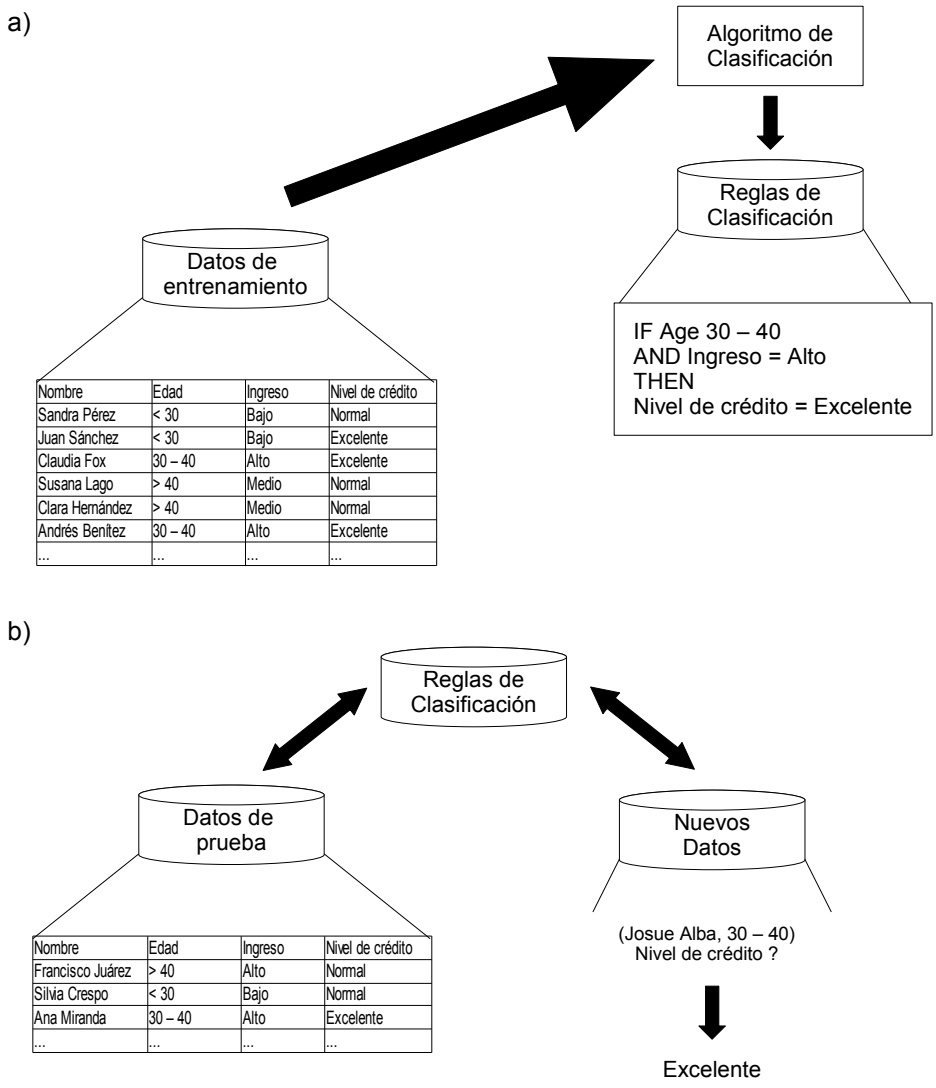
2.5. Clasificación

Las bases de datos contienen una gran cantidad de información oculta que puede ser utilizada para tomar mejores decisiones de negocio. La clasificación es una técnica de análisis de datos que puede ser utilizada para extraer modelos que describan a las clases dentro de un conjunto de datos. Por ejemplo, se puede construir un modelo de clasificación para categorizar las solicitudes de crédito hipotecario como segura o riesgosa. Diferentes métodos de clasificación han sido propuestos por investigadores en inteligencia artificial, sistemas expertos, estadística y neurobiología.

La clasificación es un proceso de dos pasos (figura 2.4). En el primer paso, se construye un modelo que describe a un conjunto determinado de clases de datos. El modelo se construye a través del análisis de las tuplas de una base de datos formadas por atributos. Se asume que cada tupla tiene una clase predefinida, que es determinada por uno de los atributos llamado atributo de clase. Dentro del contexto de clasificación a las tuplas de datos también se les conoce como muestras o ejemplos. Las tuplas de datos analizadas para construir el modelo forman el conjunto de datos de entrenamiento. A cada una de las tuplas utilizadas para realizar el entrenamiento se les conoce como muestras de entrenamiento y son elegidas de manera aleatoria del total de muestras con que se cuenta. Dado que se proporciona la clase de cada muestra de entrenamiento, a este paso se le conoce como aprendizaje supervisado (el aprendizaje del modelo es supervisado ya que para cada muestra de entrenamiento se dice a qué clase pertenece). Esto contrasta con el aprendizaje no supervisado (o clustering), en el cual no se conoce la clase de las muestras de entrenamiento.

Comúnmente el modelo aprendido se representa en forma de reglas de clasificación, árboles de decisión o fórmulas matemáticas. Por ejemplo, dada una base de datos con información de crédito de clientes, se pueden obtener reglas de clasificación para identificar a los clientes que tienen un nivel de crédito excelente o aquellos que tienen un nivel normal (figura 2.4.a). Estas reglas se pueden utilizar para categorizar futuras muestras de datos, además de proporcionar una mejor comprensión de la información en la base de datos.

En el segundo paso (figura 2.4.b), el modelo es utilizado para la clasificación. Primero se debe de estimar la precisión del modelo (o clasificador). El método "holdout" es una técnica sencilla en la cual se utiliza un conjunto de muestras de prueba. Estas muestras se eligen aleatoriamente y son independientes del conjunto de entrenamiento. La precisión de un modelo para un determinado conjunto de prueba, es el porcentaje de muestras de entrenamiento que son clasificadas correctamente por el modelo.



Nombre	Edad	Ingreso	Nivel de crédito
Sandra Pérez	< 30	Bajo	Normal
Juan Sánchez	< 30	Bajo	Excelente
Claudia Fox	30 – 40	Alto	Excelente
Susana Lago	> 40	Medio	Normal
Clara Hernández	> 40	Medio	Normal
Andrés Benítez	30 – 40	Alto	Excelente
...	...	...	...

Nombre	Edad	Ingreso	Nivel de crédito
Francisco Juárez	> 40	Alto	Normal
Silvia Crespo	< 30	Bajo	Normal
Ana Miranda	30 – 40	Alto	Excelente
...	...	...	...

Figura 2.4.: El proceso de clasificación: a) *Aprendizaje*, b) *Clasificación*.

Si se considera aceptable la precisión del modelo, éste puede ser utilizado para clasificar futuras tuplas de datos cuya clase es desconocida. Por ejemplo, las reglas de clasificación obtenidas en la figura 2.4.a a través del análisis de los datos de clientes actuales pueden ser utilizadas para predecir el nivel de crédito de un cliente nuevo.

2.6. Modelos para la Clasificación

Existen diferentes modelos para realizar clasificación en Minería de Datos. A continuación listamos los tipos de clasificadores más comúnmente utilizados, recordando que nos estamos enfocando en clasificación supervisada.

- Clasificadores basados en árboles de decisión.
- Clasificadores Bayesianos.
- Clasificadores basados en redes neuronales.

2.6.1. Clasificadores basados en árboles de decisión

Un clasificador basado en árboles de decisión divide al conjunto de datos con base en determinadas decisiones utilizando determinados valores de umbral en los atributos. La figura 2.5 representa un típico árbol de decisión que muestra el factor de riesgo asociado a conducir a alta velocidad. En la cima de la estructura arborescente se encuentra el nodo raíz que indica la característica (o atributo) que realiza la primera división o discriminación de mayor nivel. Los nodos internos del árbol representan simples reglas de decisión de uno o más atributos mientras que los nodos hojas representan clases predeterminadas. La rama izquierda de la figura 2.5 indica que las personas con $edad < 20$ involucran la categoría de riesgo alto, mientras que la rama derecha representa que las personas con $edad > 30$ están asociadas a un riesgo bajo independientemente del tipo de auto. La tercera rama (de en medio) que representa a las personas con $20 < edad < 30$ se dirige a una segunda división basada en el atributo tipo de auto antes de llegar a una decisión final.

Por lo tanto, un objeto X es clasificado al pasarlo a través del árbol iniciando por el nodo raíz. En cada nodo interno a lo largo de la trayectoria se aplica una prueba a

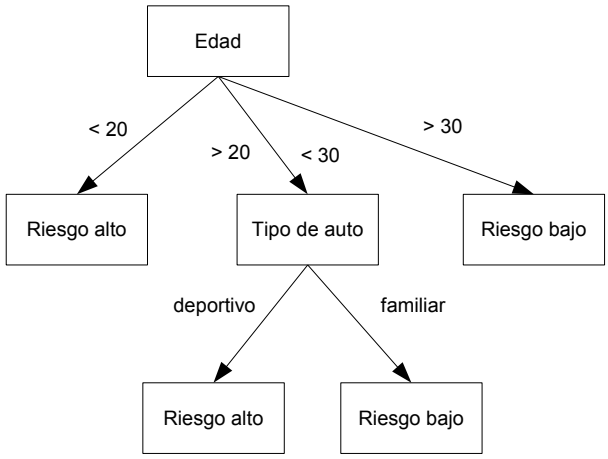


Figura 2.5.: Árbol de decisión.

los atributos de X, para determinar la siguiente rama hacia la cual se debe dirigir X. La clase de la hoja en la cual termina X se denomina como su clasificación. Un objeto es clasificado incorrectamente por el árbol si la clasificación asignada no es la clase del objeto correcta. Existen varios algoritmos de árboles de decisión bien conocidos como ID3 [35] y su sucesor C4.5 [36], CART [7], SLIQ [31], SPRINT [40], SONAR [10] y RainForest [11].

2.6.2. Clasificadores Bayesianos

Los clasificadores bayesianos son clasificadores estadísticos. Predicen la probabilidad de que una muestra dada pertenezca a una determinada clase.

Los clasificadores bayesianos se basan en el teorema de Bayes que se describirá posteriormente. Estudios comparativos de distintos algoritmos de clasificación han mostrado que un simple clasificador bayesiano conocido como el clasificador naive Bayes tiene un desempeño comparable con clasificadores basados en árboles de decisión o redes neuronales.

Los clasificadores naive Bayesian asumen que el efecto del valor de un atributo en una clase dada es independiente de los valores de los otros atributos.

Teorema de Bayes

Dado X, que es una muestra de datos cuya clase es desconocida. H es la hipótesis, de que la muestra X pertenece a una clase específica C. En un problema de clasificación se desea determinar $P(H|X)$, la probabilidad de que la hipótesis H suceda dada la muestra de datos X.

El teorema de Bayes dice:

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

donde:

$P(H)$ es la probabilidad a priori.
$P(X|H)$ es la prioridad de X dada la hipótesis H.
$P(H|X)$ es la probabilida a posteriori.

2.6.3. Clasificadores basados en redes neuronales

Una red neuronal es un conjunto de unidades conectadas donde cada conexión tiene un peso asociado a ella. Durante la fase de entrenamiento la red aprende a través del ajuste de los pesos para poder predecir la clase correcta de las muestras de entrada. La mayoría de las redes neuronales requieren de tiempos de entrenamiento largos, por lo que son más convenientes para aplicaciones en donde este hecho es permitido. Requieren de un número de parámetros que comúnmente son determinados empíricamente, como la topología de la red o estructura. Las redes neuronales han sido criticadas por su pobre interpretabilidad, puesto que es difícil para los humanos interpretar el significado simbólico detrás de los pesos aprendidos.

Sin embargo, las redes neuronales también tienen importantes ventajas como alta tolerancia al ruido en los datos, así como la capacidad de clasificar patrones con los cuales no han sido entrenadas. Además de que recientemente se han desarrollado algoritmos para la extracción a partir de redes neuronales ya entrenadas.

La red neuronal más popular es el perceptrón multicapas (MLP Multilayer Perceptron).

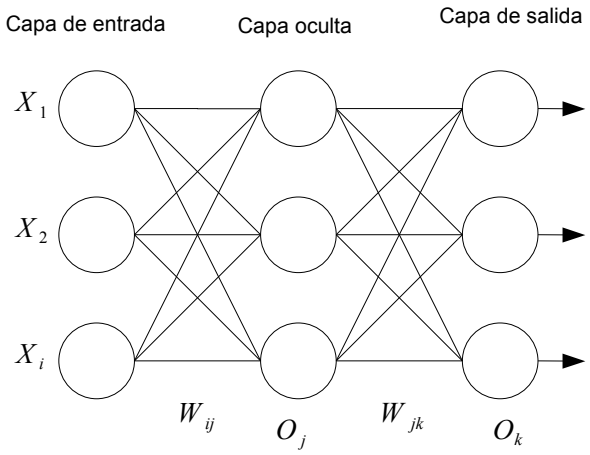


Figura 2.6.: Red Neuronal MLP.

En la figura 2.6 se muestra un ejemplo de una red MLP. Las entradas corresponden a los atributos de cada muestra de entrenamiento. Las entradas se alimentan simultáneamente en una capa de unidades que conforman la capa de entrada. Las salidas de estas unidades alimentan a una segunda capa de unidades conocida como capa oculta. Las salidas de esta capa oculta pueden ser la entrada de otra capa oculta y así sucesivamente. El número de capas ocultas es arbitrario aunque en la práctica se utiliza únicamente una. Las salidas de la última capa oculta son las entradas a las unidades que conforman la capa de salida, la cual emite la predicción de la red para las muestras dadas.

Aunque existen muchos modelos de redes neuronales que se han utilizado para clasificación, para efectos de este trabajo sólo haremos mención de MLP y de la red CMAC que se describirá a mayor detalle más adelante.

2.7. Discusión

La Minería de Datos es el proceso para el descubrimiento de información útil dentro de grandes cantidades de datos. Existen diferentes técnicas de Minería de Datos de acuerdo al tipo de información que se desee obtener. La clasificación es una de las técnicas de Minería de Datos. La clasificación tiene por objetivo la extracción de mo-

delos que permitan describir a las clases dentro de un conjunto de datos. Para llevar a cabo la clasificación existen diferentes modelos, entre los que se encuentran las redes neuronales, las cuales tienen como principal atractivo su alto poder predictivo. Las redes MLP son el modelo más comúnmente empleado para clasificación. En esta tesis presentamos como alternativa al modelo MLP a la red neuronal CMAC. La descripción, funcionamiento y características de CMAC se presentan en el siguiente capítulo.

Capítulo 3.

La Red Neuronal CMAC

Aunque el cerebelo ocupa únicamente el 10 % del volumen del cerebro, contiene más de la mitad de todas las neuronas [20]. La función del cerebelo consiste en regular la actividad motora que ocurre en otras áreas del cerebro. El cerebelo es esencial en la coordinación, destreza y sincronía de casi todos los movimientos del cuerpo, especialmente aquéllos realizados rápidamente.

Las entradas al cerebelo llegan en forma de retroalimentación sensorial proveniente de los músculos, articulaciones y la piel junto con los comandos de los centros motores acerca de qué movimiento debe ser realizado. De acuerdo con los estudios e investigaciones realizadas, estas entradas constituyen una dirección, en cuyo contenido se encuentra la señal adecuada del músculo requerido para realizar el movimiento deseado. El movimiento resultante produce una nueva entrada y así el proceso se repite. El resultado es la trayectoria del miembro a través del espacio. En cada punto de la trayectoria el estado del miembro se envía hacia el cerebelo como entrada y la memoria cerebelar responde con una señal de acción que dirige al miembro hacia el siguiente punto en la trayectoria. Estos principios básicos de operación fueron el fundamento bajo el cual se desarrolló el sistema CMAC [5].

El modelo CMAC por sus siglas en inglés Cerebellar Model Articulation Controller (figura 3.1), es un modelo adaptivo no-lineal propuesto por J. S. Albus [5] [4], con cálculos simples, generalización local y propiedades de aprendizaje rápido [32]. La red neuronal CMAC ha sido aplicada satisfactoriamente en distintas aplicaciones como: control de robots [15], deconvolución [13] [12], ecualización [38], codificación de imágenes [18] y predistorsión [39].

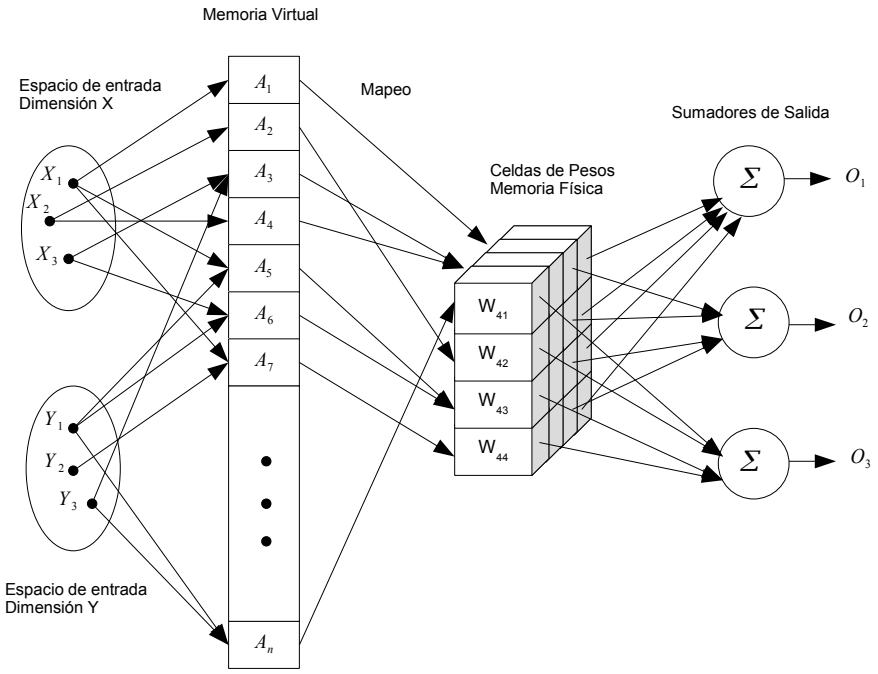


Figura 3.1.: El modelo CMAC.

3.1. Arquitectura y Funcionamiento

El modelo CMAC fue desarrollado en varios artículos en la década de los setenta [5] [4]. Originalmente fue formulado como un modelo de la corteza cerebral de los mamíferos. CMAC es una red neuronal que generaliza localmente, es decir, que las entradas que están cercanas unas de otras en el espacio de entrada, producirán salidas similares y las entradas que estén alejadas, producirán salidas no correlacionadas.

El modelo CMAC también puede verse como una tabla de búsqueda. En la figura 3.2 se muestra a la red neuronal CMAC en la forma de tabla de búsqueda.

El funcionamiento de red neuronal CMAC se describe a continuación. Primero se escalan y cuantizan las entradas utilizando:

$$q_i = \left\lfloor \frac{(x_i - \text{mín}_i)}{(\text{máx}_i - \text{mín}_i)} * resolución \right\rfloor \tag{3.1}$$

Los índices de cuantización son utilizados para localizar los pesos dentro de n_a tablas de búsqueda. Cada tabla de búsqueda se conoce como "Unidad de Asociación" (UA), etiquetadas como $UA_1 ... UA_{n_a}$. Las tablas UA almacenan el valor de un peso en cada celda. Cada UA tiene celdas que son n_a veces más grandes que el tamaño de las celdas de cuatización de las entradas, además de estar desplazadas a lo largo de cada eje por alguna constante.

El desplazamiento de la UA_i a lo largo del eje j es d_j^i. La coordenada de la celda activa de la UA_i a lo largo del eje j se calcula de la siguiente manera:

$$c_j^i = \left\lfloor \frac{q + d_j^i}{n_a} \right\rfloor \tag{3.2}$$

El algoritmo de CMAC (basado en la forma de tabla) se muestra en la tabla 3.1. La función CMACHASH toma el número de UA y las coordenadas de las celdas activas para generar un nuevo índice dentro del arreglo de pesos (memoria física). Los detalles de implementación de CMACHASH (Hashing) se describirán más adelante.

Entrenamiento.

El proceso de entrenamiento primero presenta una entrada a CMAC y se calcula su salida. Después, cada uno de los n_a pesos referenciados (uno por cada UA) es incrementado. Así la salida x_i se va acercando al valor objetivo deseado t_i. La forma en que se realiza el ajuste de pesos se muestra en la ecuación 3.3, la cual se conoce como el algoritmo LMS (Least Mean Square).

$$W_{ijk} \leftarrow W_{ijk} + \frac{\alpha a_{jk}}{n_a} (t_i - x_i) \tag{3.3}$$

donde α es la tasa de aprendizaje ($0 \leq \alpha \leq 1$).

Si $\alpha = 0$, las salidas nunca cambiarán. Si $\alpha = 1$, entonces las salidas serán iguales a los valores objetivo. Al utilizar LMS como algoritmo de entrenamiento y ($0 \leq \alpha \leq 1$), los pesos convergerán a valores que minimicen el error cuadrático medio (RMS) entre las salidas de la red y valores objetivo [17].

Tabla 3.1.: Algoritmo de CMAC.

Parámetros

n_i - Número de entradas (entero ≥ 1)
n_o - Número de salidas (entero ≥ 1)
n_a - Número de Unidades de Asociación (UA) (entero ≥ 1)
d_j^i - Desplazamiento de AU_i en el eje j (entero, $0 \leq d_j^i < n_a$)
res_i - Resolución de n_i
n_w - Número total de pesos en la memoría física

Variables internas

μ_i - Índice dentro de la tabla de pesos para la unidad de asociación $i (i = 1...n_a)$
$W_j\,[i]$ - Peso i en la tabla de pesos para la salida $x_j, i = 0...(n_w - 1), j = 1...n_x$

Entradas: $x_1...x_{n_i}$

Salidas: $y_1...y_{n_o}$

for $i = 1...n_x$
  $q_i = \left\lfloor \frac{(x_i - \text{mín}_i)}{(\text{máx}_i - \text{mín}_i)} * res_i \right\rfloor$
for $i = 1...n_a$
  for$j = 1...n_x$
    $c_j^i = \left\lfloor \frac{q_j + d_j^i}{n_a} \right\rfloor$
  μ_i =CMACHASH$(i, c_1^i, c_2^i, ..., c_{n_y}^i)$
for $i = 1...n_o$
  $x_i = 0$
  for $j = 1...n_a$
    $x_i = x_i + W_i[\mu_j]$

ENTRENAMIENTO CMAC

for $i = 1...n_o$
  incremento $= \alpha \frac{t_i - x_i}{n_a}$
  for $j = 1...n_a$
    $W_i[\mu_j] = W_i[\mu_j] + incremento$

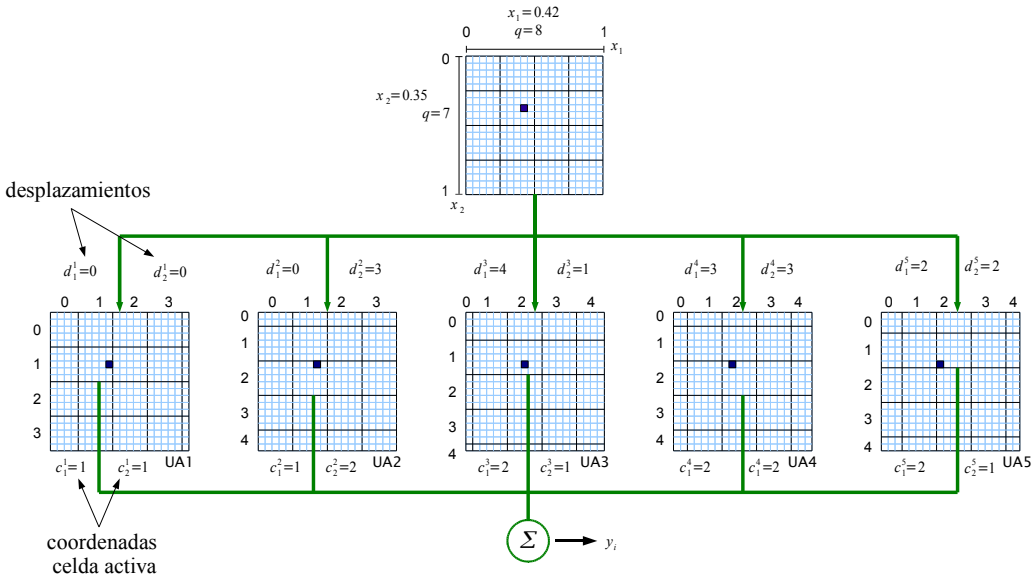


Figura 3.2.: Red Neuronal CMAC con 2 entradas, $n_a = 5$.

3.2. Ventajas

Debido a la naturaleza de representación local de la red CMAC, únicamente se requiere un pequeño número de n_a pesos para calcular el valor de salida o para modificarse de acuerdo a un cierto patrón de entrada. Así la red neuronal CMAC es computacionalmente barata de utilizar y es conveniente para proporcionar aprendizaje en tiempo-real. En este sentido, es comparativamente mejor que una red MLP que utiliza retropropagación como algoritmo de aprendizaje, el cual involucra a todos los pesos de la red para sus cálculos. Es por esta ventaja que la red CMAC se ha utilizado en múltiples ocasiones para el control en tiempo-real de sistemas.

La propiedad de generalización local de CMAC es de sumo provecho para el reforzamiento de aprendizaje y en aplicaciones de control. Mejora los problemas que presenta una red MLP en cuanto a la convergencia de funciones cuando se utiliza como aproximador de funciones. Puede ser una red muy grande y aún ser entrenada en un periodo de tiempo práctico, en comparación con otros modelos de redes neuronales.

Al utilizar el algoritmo LMS de Widrow y Hoff para la modificación de los pesos,

no es posible que se quede en un mínimo relativo como en el caso del algoritmo de retropropagación.

La implementación en Hardware de la red CMAC es relativamente fácil.

3.3. Desventajas

El requerimiento de memoria es significativamente mayor al de las redes MLP o RBF. Este problema se incrementa cuando la dimensionalidad del espacio de entrada aumenta.

La propiedad de generalización local no permite a CMAC descubrir relaciones globales en el espacio de entrada, lo cual no sucede en redes como la MLP.

Las colisiones que se presentan al utilizar un esquema de hashing para mapear la memoria virtual dentro de una memoria física causan interferencia o ruido durante el entrenamiento, aunque esto puede ser evitado utilizando un diseño apropiado.

3.4. Aplicaciones

Desde su aparición, CMAC ha sido ampliamente utilizado dentro del área de control debido a que sus propiedades lo convierten en un modelo de red neuronal muy útil en este tipo de aplicaciones. A continuación se muestran una serie de ejemplos de la aplicación de la red neuronal CMAC dentro del campo de control:

- Control de un manipulador robótico [29]
- Control de un propulsor eléctrico [41].
- Control de un sistema de propulsión hibrido para un vehículo aéreo no tripulado [16].
- Control de un sistema de guía de misiles [28].

Adicionalmente a su amplia aplicación dentro del área de contro, CMAC ha sido utilizado como clasificador de padecimientos del hígado [26] o para aprender a jugar futbol [47].

3.5. CMAC como Aproximador de Funciones

A partir de la descripción de la red neuronal CMAC y su funcionamiento se realizó una implementación del modelo en el lenguaje de programación *C++*. Adicionalmente, se programaron una serie de clases (*Fsin*, *Fcos*, *Norm* y *Fedge*) que nos permitirán probar el funcionamiento de la implementación realizada al utilizar a CMAC para la aproximación de funciones, que es la prueba más común para verificar el correcto funcionamiento de una red CMAC. En la figura 3.3 se muestra el diagrama de clases de la implementación de CMAC como aproximador de funciones.

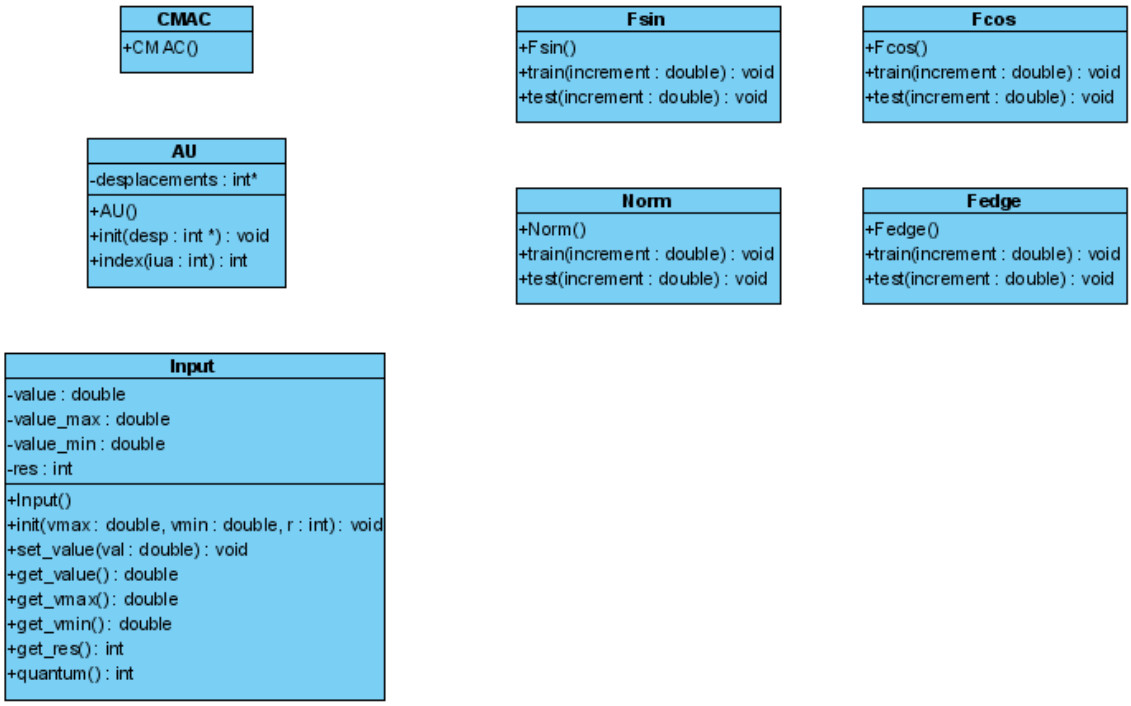


Figura 3.3.: Diagrama de clases de CMAC como aproximador de funciones.

En el diagrama de clases mostrado en la figura 3.3 se aprecian cuatro clases que no pertenecen propiamente al modelo CMAC. Cada una de estas clases: *Fsin*, *Fcos*, *Norm* y *Fedge* representa a una de las funciones que CMAC deberá aproximar. En la tabla 3.2 se muestran las cuatro funciones utilizadas.

Nombre de la función	**Fórmula**	**Clase**
$fsin(x, y)$	$sen^2(2\pi x)sen^2(2\pi y)$	Fsin
$fcos(x, y)$	$\frac{1+\cos(2\pi((2x-1)^2+(2y-1)^2))}{2\exp^{(1/4((2x-1)^2+(2y-1)^2))}}$	Fcos
$norm(x, y)$	$\sqrt{x^2+y^2}$	Norm
$fedge(x, y)$	0,75 si $x < 0{,}6$ y $y < 0{,}5$ 1 en otro caso	Fedge

Tabla 3.2.: Funciones a aproximar.

Etapas del proceso

Etapa 1:
Lectura del archivo de inicialización (ver Anexo A) en el cual se especifican los parámetros del modelo CMAC a construir como: número de entradas, valores máximos y mínimos de la entradas, número de unidades de asociación, desplazamientos para cada unidad de asociación, tasa de aprendizaje y tamaño de la memoria física del arreglo de pesos.

Etapa 2: Fase de apendizaje
Se entrena a la red CMAC con alguna de las funciones de prueba. Esto se realiza a través de presentar a la red entradas calculadas a partir de puntos en la función a aproximar. Estos puntos se van incrementando de acuerdo a un determinado paso hasta recorrer completamente el espacio de entrada.

Etapa 3: Fase de prueba

- Se prueba a la red CMAC a través de presentar nuevamente entradas de acuerdo con la función usada para entrenamiento pero se modifica el paso o incremento con el cual se recorrerá el espacio de entrada. En cada punto de prueba se calcula el error a través de la diferencia absoluta entre el valor de salida de la red y el valor esperado o real de la función.

- Se realizan las gráficas de la función aproximada por CMAC, la función real y la superficie del error de aproximación.

Las figuras 3.4 y 3.5 muestran la función *fsin* aproximada por CMAC y la función *fsin* real respectivamente. En la figura 3.6 se muestra la superficie de error de la función *fsin* aproximada por CMAC.

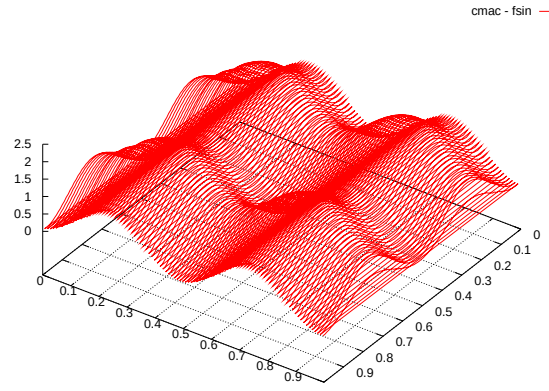


Figura 3.4.: Función *fsin* aproximada por CMAC.

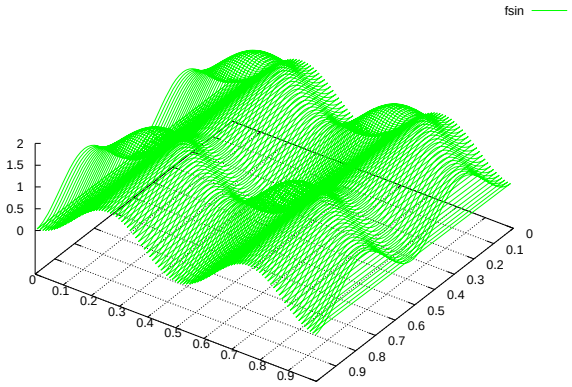


Figura 3.5.: Función *fsin* real.

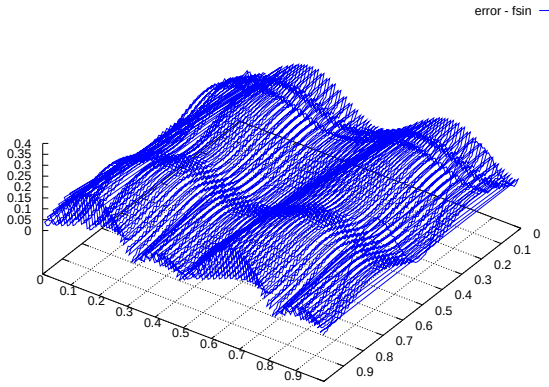


Figura 3.6.: Superficie de error de la función *fsin* aproximada.

Las figuras 3.7 y 3.8 muestran la función *fcos* aproximada por CMAC y la función *fcos* real respectivamente. En la figura 3.9 se muestra la superficie de error de la función

fcos aproximada por CMAC, en la cual es importante notar que en los bordes es donde se presenta el mayor error en la aproximación.

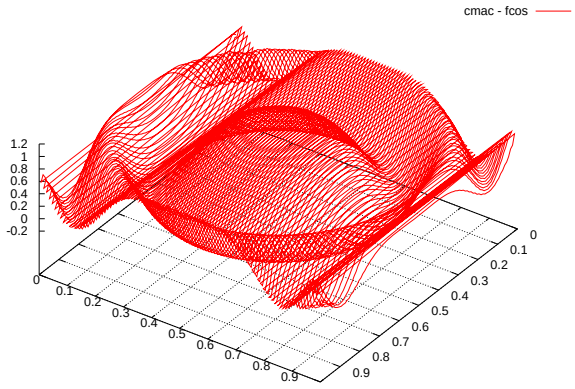


Figura 3.7.: Función *fcos* aproximada por CMAC.

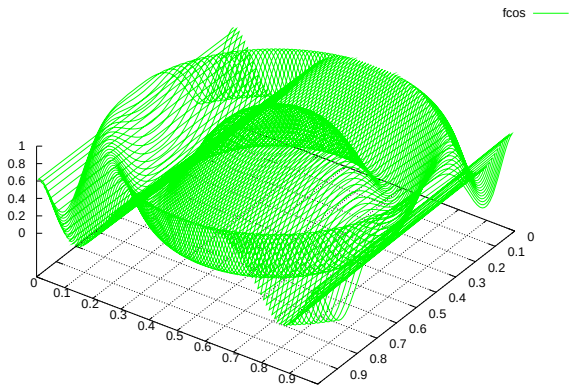


Figura 3.8.: Función *fcos* real.

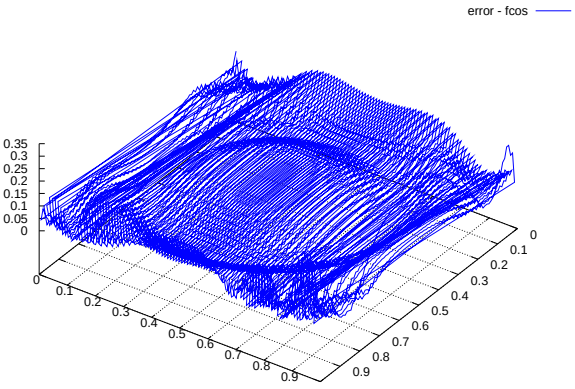


Figura 3.9.: Superficie de error de la función *fcos* aproximada.

Las figuras 3.10 y 3.11 muestran la función *norm* aproximada por CMAC y la función *norm* real, respectivamente. En la figura 3.12 se muestra la superficie de error de la función *norm* aproximada por CMAC. Si se comparan los límites del error de la figura 3.12 con los mostrados en la superficie de error acotada a un rango de $[-4, 4]$ que se muestran en la figura 3.13 es mucho más evidente la deficiencia presentada por CMAC en los bordes de la gráfica.

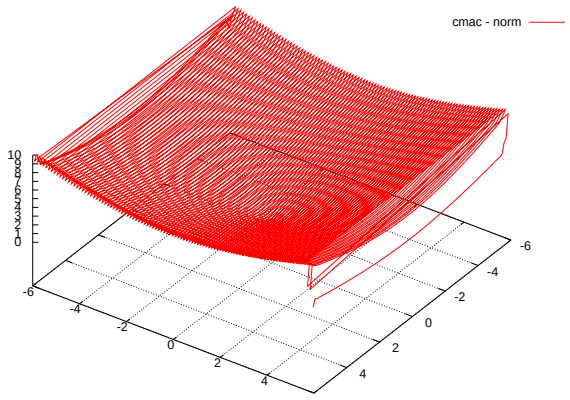


Figura 3.10.: Función *norm* aproximada por CMAC.

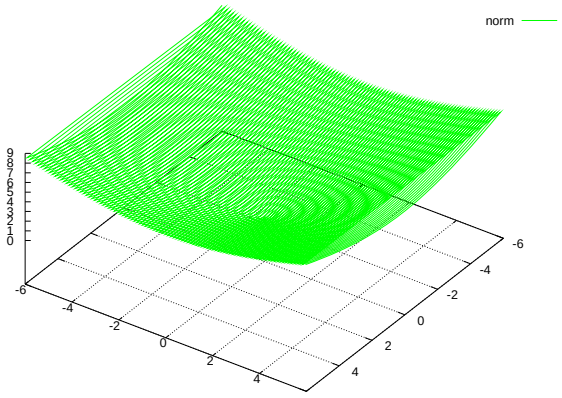


Figura 3.11.: Función *norm* real.

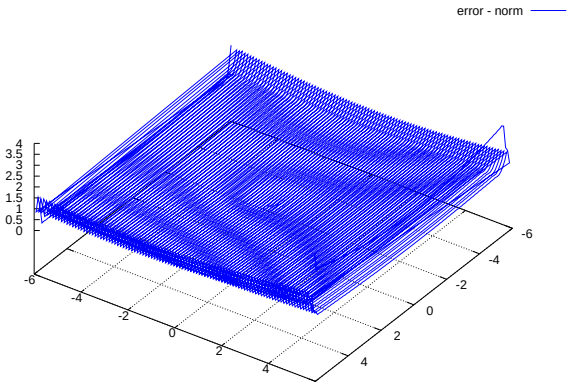


Figura 3.12.: Superficie de error de la función *norm* aproximada.

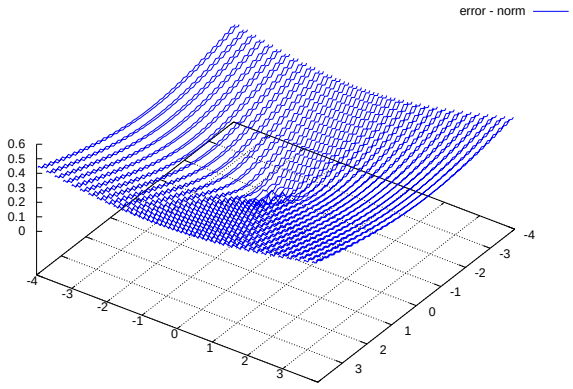


Figura 3.13.: Superficie de error recortada de la función *norm* aproximada.

Las figuras 3.14 y 3.15 muestran la función *fedge* aproximada por CMAC y la función *fedge* real, respectivamente. En la figura 3.16 se muestra la superficie de error de la función *fegde* aproximada por CMAC, en la cual es importante notar que el error más grande se presenta en la discontinuidad de la función *fedge*.

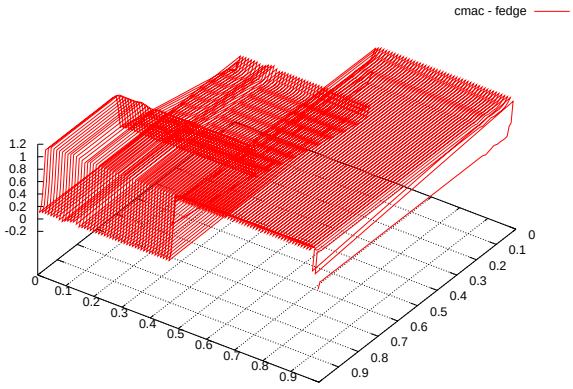


Figura 3.14.: Función *fedge* aproximada por CMAC.

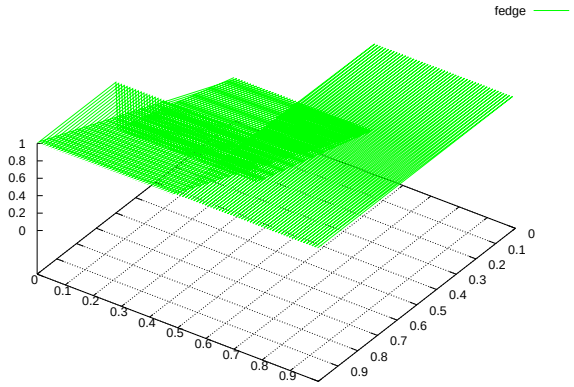


Figura 3.15.: Función *fedge* real.

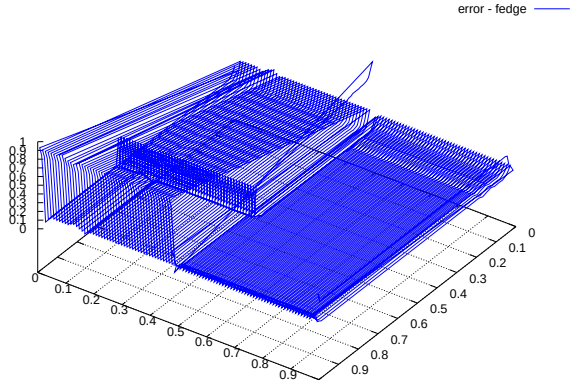


Figura 3.16.: Superficie de error de la función *fedge* aproximada.

3.6. Discusión

La red neuronal CMAC fue diseñada por J. S. Albus bajo los principios de operación del cerebelo. La principal ventaja de la red CMAC es su corto periodo de entrenamiento, además de que se requiere un número pequeño de cálculos para obtener la salida.

De acuerdo al desempeño mostrado por la red CMAC desarrollada al utilizarla como aproximador de funciones, podemos verificar que está trabajando adecuadamente. En las gráficas de error de las diferentes funciones aproximadas por CMAC, podemos notar que algunas de las limitantes de CMAC son sus fallas en los bordes de la función, así como en las discontinuidades.

En el siguiente capítulo se describirá la adaptación de la red neuronal CMAC para utilizarla como modelo de clasificación en Minería de Datos.

Capítulo 4.

CMAC para Clasificación en Minería de Datos

En este capítulo se describen los aspectos requeridos para la utilización de CMAC como clasificador. Se presenta un análisis del problema de la alta dimensionalidad que provocará la utilización de CMAC en el área de Minería de Datos donde típicamente los conjuntos de datos están formados por una gran cantidad de variables. También se presenta la forma en que se evaluará a CMAC y a otros modelos de clasificación bien conocidos.

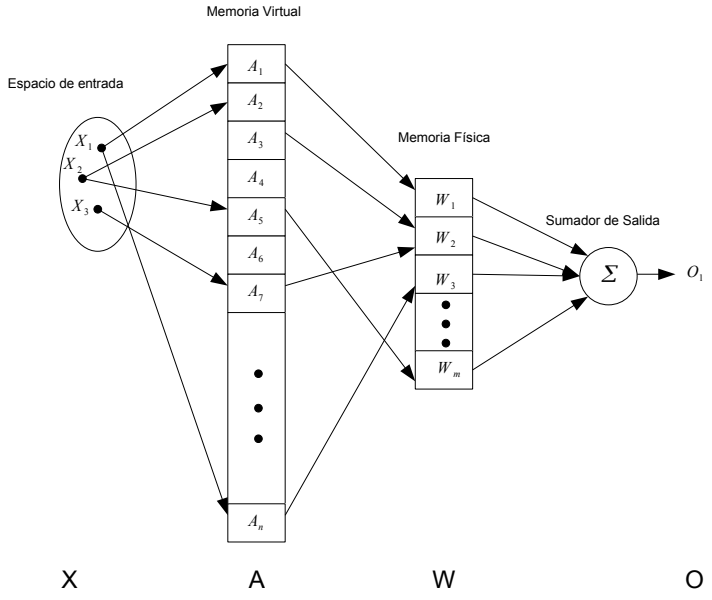


Figura 4.1.: Mapeos en CMAC.

En una red CMAC se presentan dos mapeos (figura 4.1):

El primero tiene lugar entre el espacio de entrada y la memoria virtual

$$X \rightarrow A$$

El segundo se presenta entre la memoria virtual y una memoria física en donde se almacenan los valores de los pesos

$$A \rightarrow W$$

El tamaño de esta memoria virtual es un punto a considerar cuando se desea realizar una implementación de la red neuronal CMAC. A continuación se muestra el porqué de esta observación. Supongamos que no utlizamos un esquema de hashing y que los pesos de la red se guardan directamente en locaciones de la memoria virtual. Bajo este supuesto, consideremos el número de pesos requeridos por CMAC de acuerdo con los parámetros dados en la tabla 3.1. El valor máximo de la coordenada de la celda activa c_j^i (si se considera el mayor desplazamiento posible) sería:

$$\text{máx}\, c_j^i = \left\lfloor \frac{(\text{máx}\, q_j) + \left(\text{máx}\, d_j^i\right)}{n_a} \right\rfloor \tag{4.1}$$

$$\text{máx}\, c_j^i = \left\lfloor \frac{(res_j - 1) + (n_a - 1)}{n_a} \right\rfloor \tag{4.2}$$

$$\text{máx}\, c_j^i = \left\lfloor \frac{res_j - 2}{n_a} \right\rfloor + 1 \tag{4.3}$$

Así, el número total de pesos en CMAC sería el número de pesos almacenados en cada UA, multiplicado por el número de UA's que se tenga:

$$\text{pesos en CMAC} = n_a \times \prod_{j=1}^{n_i} \left(\text{máx}\, c_j^i + 1\right) \tag{4.4}$$

Sutituyendo de la ecuación 4.3 se tiene:

$$\text{pesos en CMAC} = n_a \times \prod_{j=1}^{n_i} \left(\left\lfloor \frac{res_j - 2}{n_a} \right\rfloor + 2 \right) \tag{4.5}$$

Para simplificar esto, consideremos que res_j es lo suficientemente grande e igual para todo j. Entonces, p_j^i puede ser aproximado por:

$$\text{máx}\, c_j^i \approx \frac{res}{n_a} \tag{4.6}$$

lo cual es igual a:

$$\text{pesos en CMAC} \approx n_a \left(\frac{res}{n_a}\right)^{n_i} \tag{4.7}$$

$$\text{pesos en CMAC} = \frac{res^{n_i}}{n_a^{(n_i-1)}} \tag{4.8}$$

Es importante mencionar que en la ecuación 4.8 se puede observar que para valores típicos de res, n_a y n_i el número de pesos es enorme. Por ejemplo, supongamos los siguientes valores: $res = 200$, $n_a = 10$ y $n_i = 6$ (este valor no es tan típico). Con estos parámetros, el número de pesos requeridos sería 640 millones. Utilizando cuatro bytes por peso (considerándolo como número de punto flotante) se requerirían 2441 megabytes de memoria, lo cual es poco razonable. El problema se vuelve aún peor al observar que el número de pesos se incrementa exponencialmente al aumentar el número de entradas.

Una vez explicado esto, es claro que existen dos puntos importantes a tratar en una implementación de la red CMAC:

- Mapeo
- Hashing

4.1. Mapeo

El uso de hashing en CMAC disminuye la cantidad de memoria requerida para almacenar los pesos de la red. De esta manera, si bien no es necesario tener en memoria la cantidad de celdas que forman la memoria virtual A, sí es necesario obtener la dirección de esta celda dentro de la memoria virtual tanto para el primer mapeo $X \rightarrow A$ como para el segundo de $A \rightarrow W$.

Los rangos de los tipos de dato de C se muestran en la tabla 4.1. Es claro que si deseáramos crear una red CMAC con los siguientes parámetros, $n_i = 30$, $res = 100$, $n_a = 10$ necesitaríamos manejar números en el rango de 0 a $\frac{100^{30}}{10^{29}}$, lo cual no es posible con los tipos de dato de C cuyo tipo de dato más grande (unsigned long) únicamente tiene un rango de $[0, 2^{32})$.

Tipo	**Rango**
char	[0, 128)
signed char	(-128, 128)
unsigned char	[0, 256)
short	$(-2^{15}, 2^{15})$
unsigned short	$[0, 2^{16})$
int	$(-2^{15}, 2^{15})$
unsigned int	$[0, 2^{16})$
long	$(-2^{31}, 2^{31})$
unsigned long	$[0, 2^{32})$

Tabla 4.1.: Tipos de dato en C.

Para aplicar CMAC a la clasificación en Minería de Datos se debe trabajar con conjuntos de datos grandes y que en su mayoría tienen un número considerable de variables. Recordando que la ecuación 4.8 nos muestra que el número de direcciones requeridas (celdas o pesos) en la memoria virtual es del mismo orden en magnitud que el número de entradas, es claro que una implementación de CMAC que deba trabajar con conjuntos de datos con estas características necesitará realizar cálculos con números muy grandes.

La librería de Precisión Múltiple, también conocida como GMP, es una librería portable escrita en C para trabajar con enteros, números racionales y números de punto flotante de precisión arbitraria. Proporciona la aritmética más rápida posible para todas la aplicaciones que requieren una precisión mayor que la directamente soportada por los tipos de dato de C [3]. No existe un límite práctico de la precisión, excepto por aquel relacionado con la memoria disponible en la máquina donde GMP se ejecuta. GMP tiene un amplio conjunto de funciones, las cuales tienen una interfaz común por lo que su utilización resulta sencilla.

La velocidad proporcionada por GMP se logra a través de usar palabras completas como tipo aritmético básico, algoritmos sofisticados, la inclusión de código ensamblador cuidadosamente optimizado para los ciclos más comunes de diferentes procesadores, además de un énfasis general en la velocidad (opuesto al de la simplicidad o elegancia). Al utilizar GMP dentro del código de nuestra implementación, ésta tiene la capacidad de trabajar con un número de variables de entrada grande y así evitar alguna limitación en cuanto al número de variables en los conjuntos de datos utilizados como entrada a la red CMAC.

4.2. Hashing

Al emplear CMAC para la clasificación en Minería de Datos, el número de celdas o direcciones requeridas en A es muy grande e impráctico. Por ende es necesario proporcionar alguna alternativa para evitar este enorme requerimiento de memoria. Albus utilizó hashing para mapear la memoria A dentro de una memoria física más pequeña W [5]. Aunque al utilizar un esquema de hashing se presenta el fenómeno conocido como colisiones (figura 4.2). Una colisión se presenta cuando dos direcciones dentro de la memoria A se mapean a la misma dirección dentro de la memoria W.

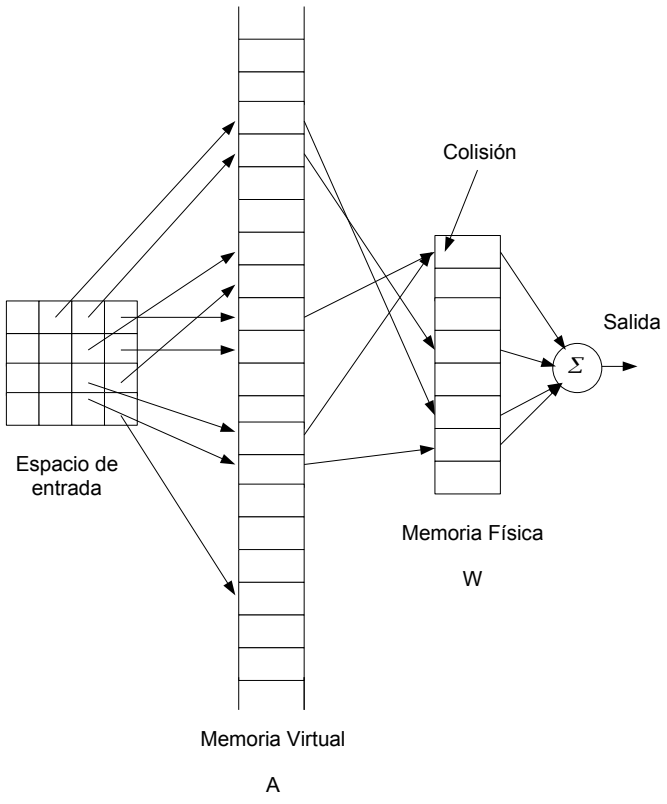


Figura 4.2.: Colisión.

Hashing es una técnica utilizada para reducir la cantidad de memoria requerida para almacenar una determinada cantidad de datos dispersos sobre un gran número de locaciones de memoria. Si se asume que las direcciones de los datos en la memoria de gran tamaño van desde 1 hasta N y las direcciones de la memoria de menor tamaño van desde 1 hasta n, entonces la función de hashing (o función hash) debe satisfacer:

$$1 \leq H(k) \leq n, \quad \forall \quad 1 \leq k \leq N \tag{4.9}$$

En CMAC se mapea cada dirección de la memoria virtual hacia una dirección en la memoria física.

El desempeño de una función hash es medido típicamente a través del número de colisiones que presentan al utilizar dicha función. Numerosas pruebas han mostrado que hay dos tipos de funciones hash que trabajan bien [22]. Una de ellas esta basada en la división, mientras que la otra esta basada en la multiplicación.

La función de hash basada en división es:

$$H_d\,(k) = 1 + k \text{ mód } n \tag{4.10}$$

Mientras que la función hash basada en multiplicación se calcula como:

$$H_m(k) = 1 + \lfloor n(kF \text{ mód } 1) \rfloor \tag{4.11}$$

Donde F es un entero que puede elegirse arbitrariamente, aunque su elección tiene una gran implicación en el número de colisiones generadas. El hashing por división distribuye uniformemente el total de direcciones de la memoria de gran tamaño dentro de las direcciones de la memoria de menor tamaño en el orden de cada dirección. Así, si la memoria de gran tamaño se encuentra completamente llena de datos, entonces el hashing por división siempre asegura el menor número de colisiones. De hecho, si $N \leq n$, no se deberían de presentar colisiones. Aunque siempre es deseable reducir el número de colisiones, el hashing por división tiene un defecto que lo hace difícilmente aplicable a CMAC. Debido al hecho de que el hashing por división distribuye uniformemente y en orden las direcciones de la memoria de gran tamaño dentro de la memoria de menor tamaño, la probabilidad de un traslape grande entre dos entradas ampliamente separadas se incrementa enormemente si se presentan colisiones. Por esta razón, el desempeño de una red CMAC que utilice hashing por división, no se espera que sea el adecuado a menos que se presenten muy pocas colisiones.

En contraste, el hashing por multiplicación revuelve las direcciones de la memoria de gran tamaño dentro de la memoria de menor tamaño. Como resultado, dos celdas

vecinas en la memoria de gran tamaño estarán separadas en la memoria de menor tamaño, así cualquier entrada será dispersada sobre la memoria de menor tamaño. De esta manera la probabilidad de tener un traslape grande se reduce. Sin embargo, aunque $N \leq n$ se pueden presentar colisiones usando hashing por multiplicación.

La función de hashing basada en división no tiene parámetros disponibles para su ajuste. En contraste, la función de hashing basada en multiplicación cuenta con el parámetro F que puede ser elegido arbitrariamente.

En este sentido se podría generar un conjunto de funciones hashing H_m que estén parametrizadas por F. Un miembro importante de este conjunto es la función de hashing Fibonacci, en la cual F se elige como r_g , donde r_g es la razón aurea $r_g = \left(1 + \sqrt{5}\right)/2$.

La razón áurea r_g tiene ciertas propiedades interesantes [22]. Consideremos el dividir el segmento de recta $[0, 1]$ sucesivamente en los puntos $r_g, 2r_g, 3r_g, ...$, donde nr_g denota la parte fraccional de nr_g, esto es nr_g mód 1. Cada nuevo punto nr_g cae siempre dentro de uno de los intervalos más grandes. Si un nuevo punto agregado nr divide al intervalo más grande restante en dos segmentos tal que uno es más largo que el doble del otro, esto se conoce como un mala división. Definitivamente, existen malas divisiones en algunos puntos para todos los r excepto por r_g y $1 - r_g$. Con r_g o $1 - r_g$, el intervalo más grande restante siempre es dividido en la razón áurea (figura 4.3). Esto significa que el hashing Fibonacci es el hashing más uniforme. Esto no sólo permite que el número de colisiones sea pequeño, también hace que las entradas se dispersen en W lo más posible. Esta última propiedad es aún más importante en una red CMAC [46], dado que esto significa que se presentarán pocos traslapes entre celdas vecinas de entradas bien separadas. Basado en este hecho, el hashing Fibonacci debería de funcionar adecuadamente en nuestra implementación de CMAC.

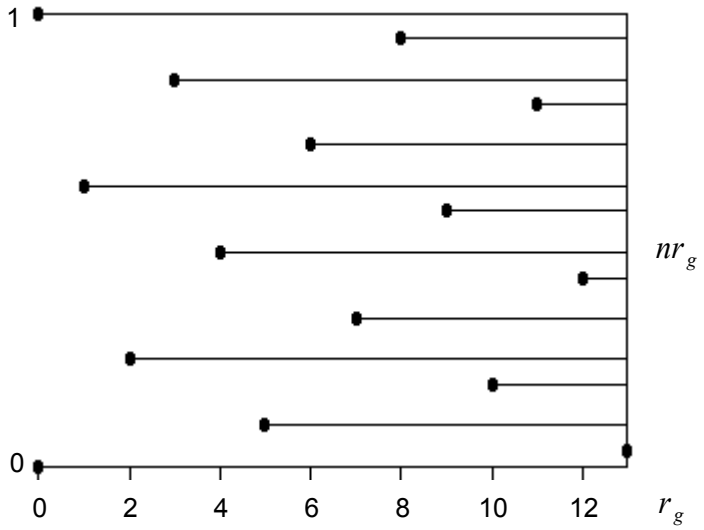


Figura 4.3.: Hashing Fibonacci.

4.3. Evaluación del desempeño de un modelo de clasificación

La evaluación del desempeño de un modelo de clasificación se basa principalmente en la cuenta del número de muestras de ejemplo correcta e incorrectamente predecidas por el modelo [42]. Estos resultados son tabulados en una tabla conocida como matriz de confusión (*confusion matrix*). La tabla 4.2 representa la matriz de confusión para un problema de clasificación binario. Cada entrada f_{ij} en esta tabla indica el número de muestras de la clase i que se han predecido como de la clase j. Por ejemplo f_{01}, es el número de muestras de la clase 0 incorrectamente predecidas como pertenecientes a la clase 1. Basados en las entradas de la matriz de confusión, el número total de predicciones correctas hechas por el modelo es $(f_{11} + f_{00})$, mientras que el número de predicciones incorrectas es $(f_{10} + f_{01})$.

		Clase Predecida	
		Clase = 1	Clase = 0
Clase Actual	Clase = 1	f_{11}	f_{10}
	Clase = 0	f_{01}	f_{00}

Tabla 4.2.: Matriz de confusión para un problema de 2 clases.

Aunque una matriz de confusión proporciona la información necesaria para determinar qué tan buen desempeño tiene un modelo de clasificación, resumir esta información en un solo número podría resultar más conveniente al realizar la comparación de desempeño de diferentes modelos. Esto se puede realizar a través de utilizar una métrica de desempeño como la precisión, la cual se define de la siguiente manera:

$$\text{Precisión} = \frac{\text{Número de predicciones correctas}}{\text{Número total de prediciones}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}} \tag{4.12}$$

De forma equivalente, el desempeño de un modelo puede ser expresado en términos de su tasa de error, la cual está dada por la siguiente ecuación:

$$\text{Tasa de error} = \frac{\text{Número de predicciones incorrectas}}{\text{Número total de prediciones}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}} \tag{4.13}$$

En muchas ocasiones es muy útil medir el desempeño de un modelo sobre un conjunto de prueba porque dicha medida proporciona una estimación más justa de su error de generalización. Así, la precisión o tasa de error calculada sobre un conjunto de prueba puede utilizarse para comparar el desempeño relativo de dos clasificadores diferentes en el mismo dominio. Sin embargo, para realizar esto es necesario conocer los valores de clase de la muestras dentro del conjunto de prueba.

El método Holdout

En el método holdout, el conjunto de datos original con las muestras etiquetadas con su valor de clase es dividido en dos conjuntos disjuntos. Así, un modelo de clasificación es entrenado utilizando el conjunto de datos de entrenamiento y su desempeño es evaluado sobre el conjunto de prueba (figura 4.4). La proporción de datos reservada para entrenamiento y para prueba comúnmente queda a criterio del analista. La precisión del clasificador se puede estimar con base en la precisión del modelo entrenado sobre el conjunto de prueba.

Selección aleatoria de muestras

El método holdout puede ser repetido un cierto número de veces para mejorar la estimación del desempeño del modelo de clasificación. Este enfoque es conocido como selección aleatoria de muestras. Dada pre_i como la precisión en la $i - esima$ iteración, la precisión conjunta está dada por:

$$pre_c = \sum_{i=1}^{k} pre_i/k \tag{4.14}$$

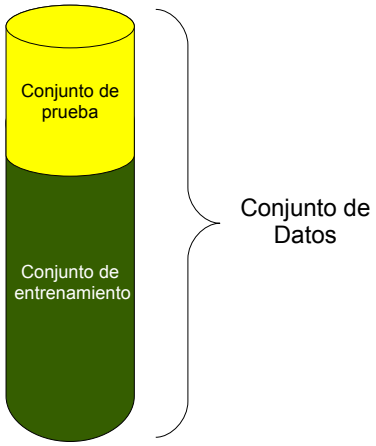


Figura 4.4.: El método Holdout: divide el conjunto de datos.

4.4. Metodología

Para evaluar el desempeño de la red neuronal CMAC para clasificación en Minería de Datos se plantearon una serie de casos de estudio en los cuales se comparará el desempeño de CMAC contra otros modelos de clasificación ampliamente utilizados en el área de Minería de Datos: MLP y C4.5. La red neuronal MLP es el modelo de redes neuronales más ampliamente utilizado tanto en trabajos de investigación como en herramientas comerciales en el campo de Minería de Datos, siendo está la principal razón para realizar su comparación con CMAC y así determinar qué tan buena alternativa es

CMAC en comparación con la red MLP dentro de la Minería de Datos. C4.5 es un algoritmo de árbol de decisión bien conocido cuya aplicación en Minería de Datos también ha sido extensa, su comparación con CMAC resulta de interés, para determinar si la red CMAC sigue conservando la propiedad de alta capacidad predictiva que presentan los modelos de redes neuronales sobre los árboles de decisión.

A continuación se describe la metodología utilizada en los diferentes casos de estudio para realizar la comparación de los modelos de clasificación a evaluar (ver figura 4.5).

Etapa 1: *Fase de preparación de los datos*

- Limpieza de los datos. Remover las muestras que contengan valores perdidos ya que estas muestras, lejos de ayudar a la mejora en el desempeño de los modelos, podrían introducir ruido y perjudicar así su precisión.
- Selección de variables. Selección de las variables más adecuadas para el análisis de los modelos, dejando de lado aquellas que no colaboren a la mejora en el desempeño de los modelos.
- Transformación de variables. Los modelos a comparar (CMAC, MLP, C4.5) requieren que las variables de entrada sean de tipo numérico por lo que si los conjuntos de datos seleccionados para realizar los experimentos contienen variables de entrada de tipo categórico éstas deberán de transformarse en variables de tipo numérico. La única variable de tipo categórico permitida es la variable objetivo o de clase. La transformación de las variables categóricas a variables de tipo numérico se realizó a través de un procedimiento empírico que efectúa la asignación del valor numérico para una variable categórica de acuerdo a la frecuencia de la misma en el conjunto de datos original.
- Obtener los parámetros de las variables de entrada (únicamente para CMAC).

Etapa 2: *Fase de división de los datos*

- Selección aleatoria de las muestras. Elección de las muestras que conformarán el conjunto de entrenamiento y el conjunto de prueba. La división se realiza de acuerdo al parámetro de porcentaje recibido para cada conjunto de acuerdo a la prueba a realizar.

Etapa 3: *Fase de evaluación*

- Creación del modelo de clasificación de acuerdo con los parámetros especificados (en el Anexo B se encuentran los parámetros de los modelos de clasificación utilizados en cada caso de estudio).
- Entrenar al modelo de clasificación con las muestras del conjunto de entrenamiento.
- Calcular la precisión del modelo con las muestras del conjunto de prueba.

Etapa 4: *Fase de resultados*

- Recopilar los datos de las pruebas realizadas.
- Generación de la gráfica comparativa del desempeño de los modelos de clasificación probados.

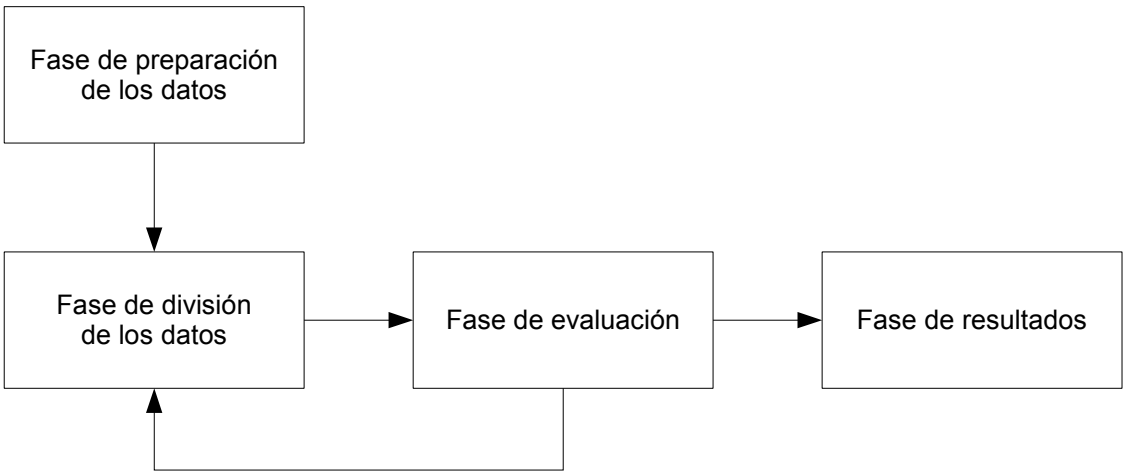


Figura 4.5.: Metodología.

Para implementar la metodología descrita se desarrollaron los métodos para llevar el tratamiento de los datos y la división de los mismos en el conjunto de datos de entrenamiento y el conjunto de prueba. La figura 4.6 muestra el diagrama de clases de la red

CMAC para la clasificación en Minería de Datos bajo el esquema de pruebas descrito. Las clases *Train* y *Test* se encargan de llevar a cabo las etapas de entrenamiento y prueba del modelo, respectivamente.

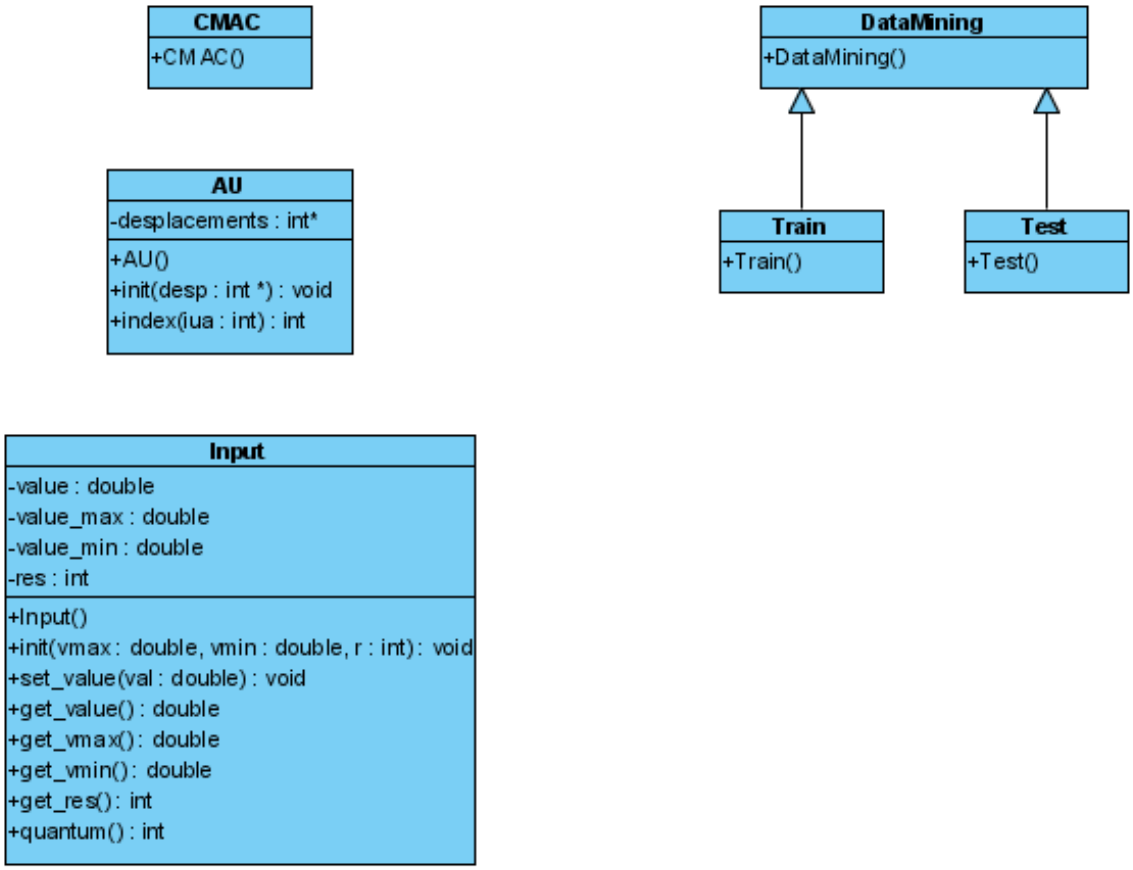


Figura 4.6.: Diagrama de Clases de CMAC para la clasificación en Minería de Datos.

Las pruebas de los clasificadores basados en MLP y C4.5 se realizaron haciendo uso de la herramienta de Minería de Datos Tanagra [37], a fin de realizar una comparación de nuestra implementación de CMAC con una herramienta bien conocida en el área.

4.5. Discusión

Al utilizar CMAC como modelo de clasificación en Minería de Datos y debido a que el número de variables en los conjuntos de datos típicos en esta área es grande, se tiene que es necesario proveer los mecanismos que permitan a CMAC recibir y manejar un número de entradas grande.

Los aspectos fundamentales para el manejo de un número de entradas grande en CMAC se centran en el mapeo y la utilización de hashing para reducir el tamaño del arreglo de los pesos de la red.

La metodología diseñada para evaluar el desempeño de la red CMAC como modelo de clasificación, se basa en el método *holdout*. En el siguiente capítulo se mostrará la evaluación de CMAC y su comparación con los clasificadores basados en MLP y el algoritmo C4.5.

Capítulo 5.

Casos de Estudio

En este capítulo se describen los casos de estudio realizados para evaluar el desempeño de la red neuronal CMAC desarrollada como modelo de clasificación en Minería de Datos. Los resultados obtenidos de los diferentes casos de estudio nos proporcionarán una idea más clara de las capacidades de la red CMAC dentro del campo de la Minería de Datos.

5.1. Caso de estudio 1

En este primer caso de estudio se muestra el desempeño de la red CMAC para clasificación en Minería de Datos y su comparación con otros modelos de clasificación (MLP y C4.5) utilizando la base de datos *Mushrooms* obtenida del *UCI Machine Learning Repository*. Esta base de datos contiene los registros de hongos obtenidos por *The Audubon Society Field Guide to North American Mushrooms* (1981) e incluye las descripciones de muestras correspondientes a 23 especies de hongos. Cada especie es identificada como definitivamente comestible o definitivamente venenosa.

La base de datos *Mushrooms* tiene 8,124 registros con 22 variables de tipo categórico. En el Anexo C se proporciona información más completa acerca de la base de datos *Mushrooms*. Únicamente se utilizaron 21 variables del total debido a que la variable *stalk-root* presenta en su mayoría valores perdidos y por lo tanto se decidió no utilizarla. La distribución de la variable objetivo (o de clase) es la siguiente:

- comestible: 4,208 (51.8 %)
- venenoso: 3,916 (48.2 %)

La evaluación del desempeño de los modelos se realizó utilizando el método de *hold-out* combinado con la *selección aleatoria de muestras*. Así, los resultados se obtienen

con base en el promedio de distintas ejecuciones de los modelos sobre conjuntos de datos diferentes (formados aleatoriamente). En la tabla 5.1 se muestran los resultados de la precisión de cada modelo, utilizando un esquema de 70 % de los datos empleados para entrenamiento y 30 % para prueba, además de estar basados en el promedio de 10 experimentos.

Modelo	**Precisión**
CMAC	99.64
MLP	99.15
C4.5	98.73

Tabla 5.1.: Precisión de los modelos utilizando la base de datos *Mushrooms* y un esquema de 70 % de los datos utilizados para entrenamiento y 30 % para prueba.

La figura 5.1 muestra la precisión de los modelos de clasificación evaluados al variar la proporción de datos utilizada para entrenamiento y la utilizada para prueba. Cada punto se obtuvo con base en el promedio de 10 experimentos.

De los resultados obtenidos en este caso de estudio podemos observar que el clasificador basado en la red CMAC presenta una precisión mayor a la red MLP, lo cual es un resultado importante, ya que para este caso en especifico, CMAC muestra que aunado a sus propiedades de rápido entrenamiento también es capaz de proveer un alto grado de precisión como clasificador en el área de Minería de Datos. Ambos modelos de redes neuronales: CMAC y MLP presentan mejores resultados lo cual corrobora el bien conocido aspecto de que en general las redes neuronales tienen un mayor poder predictivo que los árboles de decisión. Claro está que la ventaja de estos últimos sobre las redes neuronales está dada por la fácil interpretabilidad de sus resultados y reglas generadas.

Los parámetros de los modelos de clasificación utilizados en este caso de estudio se muestran en el Anexo B.

5.2. Caso de estudio 2

Para este segundo caso de estudio se muestra el desempeño de la red CMAC como modelo de clasificación y su comparación con otros modelos utilizando la base de datos *Adult* obtenida del *UCI Machine Learning Repository*. Esta base de datos contiene los registros de ciudadanos de los Estados Unidos obtenidos del *US Census Bureau* e

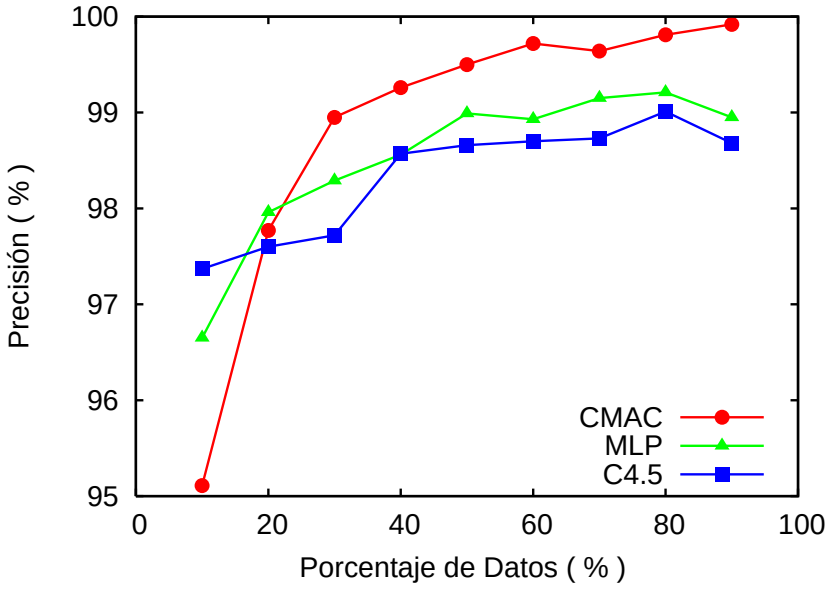


Figura 5.1.: Comparación de la precisión de los modelos de clasificación utilizando la base de datos *Mushrooms*.

incluye información sobre características de tipo social, económico y cultural de cada individuo y una clasificación basada en si ganan más o menos de 50,000 dlls anualmente.

La base de datos *Adult* contiene 48,842 registros con 6 variables de tipo numérico y 8 de tipo categórico. En el Anexo C se presentan datos más completos sobre las variables de la base de datos *Adult*. Esta base de datos presenta valores perdidos por lo que después de eliminar los registros que contuvieran algún valor perdido en alguna de sus variables el número de registros disponibles es de 45,222. Se realizó un análisis de las variables y su importancia para el modelo, por lo que únicamente se utilizaron 8 variables como entradas para los modelos. Las variables seleccionadas se muestran en la tabla 5.2. La variable objetivo o de clase tiene la siguiente distribución:

- $> 50,000$ dlls $(24.78\,\%)$
- $\leq 50,000$ dlls $(75.22\,\%)$

La evaluación del desempeño de los modelos se realizó utilizando el método de *hold-*

Nombre	**Tipo**
age	numérico
workclass	categórico
education	categórico
education-num	numérico
marital-status	categórico
occupation	categórico
hours-per-week	numérico
country	categórico

Tabla 5.2.: Variables seleccionadas de la base de datos *Adult*.

out combinado con la *selección aleatoria de muestras*. Así, los resultados se obtienen con base en el promedio de distintas ejecuciones de los modelos sobre conjuntos de datos diferentes (formados aleatoriamente). En la tabla 5.3 se muestran los resultados de la precisión de cada modelo, utilizando un esquema de 70 % de los datos empleados para entrenamiento y 30 % para prueba además de estar basados en el promedio de 10 experimentos.

Modelo	**Precisión**
CMAC	82.03
MLP	82.53
C4.5	81.30

Tabla 5.3.: Precisión de los modelos utilizando la base de datos *Adult* y un esquema de 70 % de los datos utilizados para entrenamiento y 30 % para prueba.

La figura 5.2 muestra la precisión de los modelos de clasificación evaluados al variar la proporción de datos utilizada para entrenamiento y la utilizada para prueba. Cada punto se obtuvo con base en el promedio de 10 experimentos.

De los resultados derivados del caso de estudio se puede observar que en esta ocasión el clasificador basado en CMAC obtuvo una precisión un poco menor a la obtenida por el clasificador basado en el modelo MLP, además de ser claro que sigue manteniendo un poder predictivo mayor al del clasificador C4.5.

Los parámetros de los modelos de clasificación utilizados en este caso de estudio se muestran en el Anexo B.

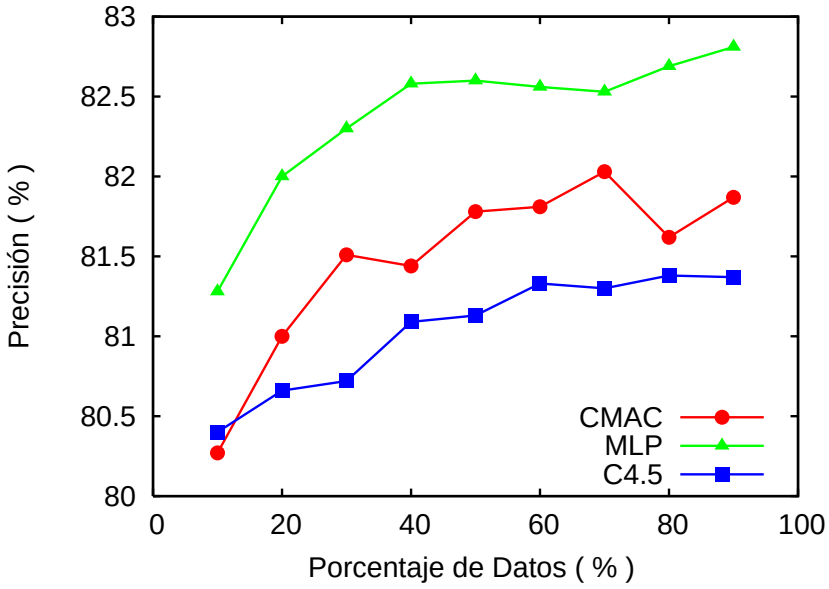


Figura 5.2.: Comparación de la precisión de los modelos de clasificación utilizando la base de datos *Adult*.

5.3. Caso de estudio 3

En este tercer caso de estudio se muestra el desempeño del clasificador basado en CMAC y su comparación con otros modelos de clasificación utilizando la base de datos *Clothing Store* obtenida de [25]. Esta base de datos representa los datos actuales de una cadena de tiendas de ropa en Nueva Inglaterra, EU.

La base de datos *Clothing Store* contiene 28,799 registros, cada uno de los cuales está compuesto de 50 variables. En el Anexo C se puede encontrar información más detallada acerca de la base de datos *Clothing Store*. La variable objetivo es de tipo binaria denotando qué cliente respondió y qué cliente no lo hizo, a una campaña de mercadotecnia a través del correo. La variable objetivo tiene la siguiente distribución:

- No respondió: 18,129 (83.39 %)
- Respondió: 21,740 (16.61 %)

En este caso de estudio se utilizo una técnica de selección de variables a través del uso de un árbol de decisión. La selección de variables es uno de los usos auxiliares de los árboles de decisión, dado que el subconjunto de variables de entrada seleccionado por el árbol puede ser un subconjunto prometedor para otros modelos, particularmente para modelos no lineales como las redes neuronales [34].

En [7] se da una medida de la importancia de una variable en un árbol. Sea $s\left(x_j, t\right)$ la mejor división (incluyendo la primera división en el árbol) en el $t - ésimo$ nodo interno usando la $j - ésima$ entrada. La importancia de la variable es el promedio de los pesos de la reducción de la impureza Gini, para las divisiones usando la $j - ésima$ entrada a lo largo de todos los nodos internos en el árbol. Los pesos son los nodos internos.

$$\text{Importancia}\left(x_j\right) = \sum_{t=1}^{r} \frac{n_t}{n} \Delta Gini\left(s\left(x_j\right), t\right) \tag{5.1}$$

Para obtener la selección de variables de la base de datos Clothing Store utilizamos el Enterprise Miner de SAS, que es una herramienta de Minería de Datos bien conocida. En la tabla 5.4 se muestran las variables seleccionadas.

Variable	**Importancia**
LTFREDAY	1.0000
DAYS	0.2835
HI	0.0996
AVRG	0.0841
PBLOUSES	0.0640

Tabla 5.4.: Selección de variables utilizando el Enterprise Miner de SAS.

La evaluación del desempeño de los modelos se realiza a través del método de *holdout* combinado con la *selección aleatoria de muestras*. Así los resultados se obtienen con base en el promedio de distintas ejecuciones de los modelos sobre conjuntos de datos diferentes (formados aleatoriamente). En la tabla 5.5 se muestran los resultados de la precisión de cada modelo, utilizando un esquema de 70 % de los datos utilizados para entrenamiento y 30 % para prueba además de estar basados en el promedio de 10 experimentos.

Modelo	Precisión
CMAC	84.79
MLP	85.40
C4.5	81.79

Tabla 5.5.: Precisión de los modelos utilizando la base de datos *Clothing Store* y un esquema de 70 % de los datos utilizados para entrenamiento y 30 % para prueba.

La figura 5.3 muestra la precisión de los modelos de clasificación evaluados al variar la proporción de datos utilizada para entrenamiento y la utilizada para prueba. Cada punto se obtuvo con base en el promedio de 10 experimentos.

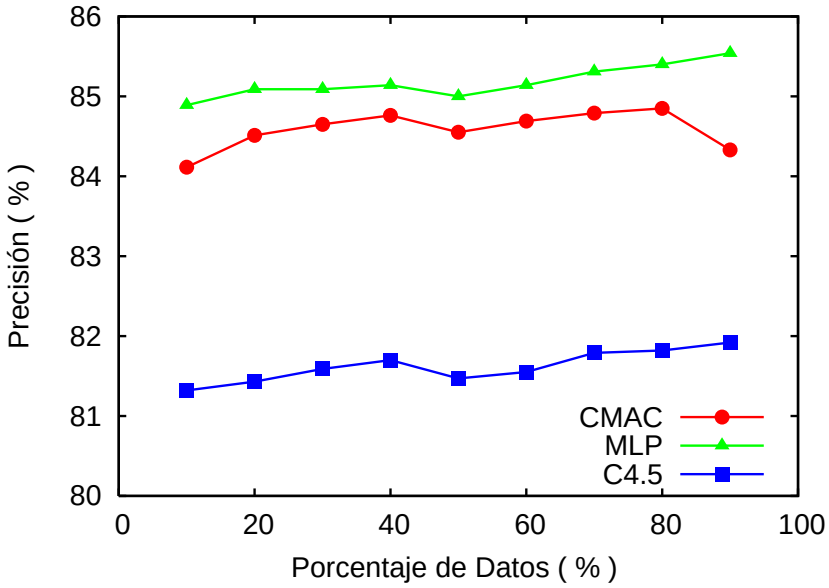


Figura 5.3.: Comparación de la precisión de los modelos de clasificación utilizando la base de datos *Clothing Store*.

De los resultados derivados del caso de estudio se observa que para el conjunto de datos *Clothing Store* el clasificador basado en la red CMAC obtiene una precisión cercana a la obtenida por el clasificador basado en el modelo MLP, además de que sigue

manteniendo una clara ventaja sobre el clasificador basado en C4.5.

Los parámetros de los modelos de clasificación utilizados en este caso de estudio se muestran en el Anexo B.

5.4. Análisis de los resultados

De los resultados obtenidos en los distintos casos de estudio realizados, es posible observar que CMAC puede ser tomado en cuenta como una opción de modelo de clasificación. Se hace notar que el desempeño de CMAC dependerá de la aplicación en específico. Así, será decisión del analista evaluar si el desempeño de CMAC para el caso en particular es aceptable.

Para el caso de la base de datos *Mushrooms* los resultados de CMAC son muy satisfactorios al presentar un mejor desempeño que el de la red MLP. Aunque para el caso de las evaluaciones realizadas con las bases de datos *Adult* y *Clothing Store* el desempeño de CMAC se mostró ligeramente por debajo del presentado por el modelo MLP, recordemos que con CMAC ganamos significativamente en el tiempo de entrenamiento y tal vez en algunos casos se sacrificará un poco en cuanto a precisión se refiere.

También es justo señalar que el poder predictivo propio de las redes neuronales se mostró en todos los casos de estudio ya que tanto CMAC como MLP mostraron mayor precisión en las clasificaciones que el algoritmo C4.5.

Capítulo 6.

Conclusiones y Trabajo Futuro

En este trabajo de tesis se presenta la implementación de una red neuronal CMAC como modelo de clasificación en Minería de Datos en busca de proveer una alternativa al modelo clásico MLP. Uno de los nuevos requerimientos del campo de Minería de Datos es el de proporcionar resultados en tiempos cada vez más cortos. Para ello, el contar con modelos que permitan un entrenamiento en tiempo real es fundamental. La red MLP ha sido el modelo más ampliamente utilizado en Minería de Datos debido a su buen desempeño en la mayoría de las aplicaciones. Sin embargo, una de sus principales desventajas es el largo periodo de tiempo requerido para su entrenamiento. Es por ello que su utilización en aplicaciones con importantes compromisos de tiempo no parece ser adecuada. Por otra parte, CMAC es una red bien conocida en el área de control cuya principal ventaja es el corto periodo de entrenamiento requerido. Así, resulta atractivo el evaluar el desempeño de una red con las propiedades de CMAC para la clasificación en Minería de Datos.

Con base a los resultados obtenidos en los distintos casos de estudio realizados, podemos decir que la red neuronal CMAC es una buena alternativa como modelo de clasificación en Minería de Datos. La precisión de la red CMAC desarrollada es muy cercana a la de una red MLP y en algunos casos hasta superior. Es importante recalcar que adicionalmente al buen desempeño mostrado por la red CMAC esta cuenta con la propiedad del corto periodo de tiempo requerido para su entrenamiento.

Es importante señalar que existen algunos puntos adicionales a tomar en cuenta tanto para la aplicación de CMAC como clasificador, como para futuros trabajos de investigación. Entre ellos está la selección de parámetros, ya que no se cuenta con heurísticas bien definidas para la selección de parámetros tales como la resolución óptima de cada entrada, los desplazamientos adecuados en cada unidad de asociación o el mismo tamaño de la memoria física adecuada para el modelo. Al no contar con un respaldo para la selección de estos parámetros, en la práctica se utiliza una selección empírica.

Obviamente, esto no es óptimo ya que el valor de estos parámetros tiene una influencia importante en el desempeño de CMAC como clasificador.

Adicionalmente, resultaría de interés el realizar la evaluación de CMAC como modelo de clasificación incorporando otro tipo de técnicas dentro del campo de softcomputing como las radial basis functions (RBF), en busca de mejorar la precisión del modelo.

Otras ideas en cuanto a trabajo futuro serían el desarrollo de un sistema experto para el ajuste de parámetros o la utilización de otra función de activación distinta a la del modelo clásico de CMAC.

Apéndice A.

Parámetros de CMAC como Aproximador de Funciones

A.1. Parámetros de CMAC para aproximar la función fsin

Número de entradas 2
Parámetros de la entrada 1 (máximo, mínimo, resolución) 1.0, 0.0, 500
Parámetros de la entrada 2 (máximo, mínimo, resolución) 1.0, 0.0, 500
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 1, 1
Desplazamientos UA 2 2, 2
Desplazamientos UA 3 3, 3
Desplazamientos UA 4 4, 4
Desplazamientos UA 5 5, 5
Desplazamientos UA 6 6, 6
Desplazamientos UA 7 7, 7
Desplazamientos UA 8 8, 8
Desplazamientos UA 9 9, 9
Desplazamientos UA 10 6, 5
Tasa de aprendizaje 0.05
Tamaño del arreglo (doubles) 6000

Tabla A.1.: Parámetros CMAC - fsin.

A.2. Parámetros de CMAC para aproximar la función fcos

Número de entradas 2
Parámetros de la entrada 1 (máximo, mínimo, resolución) 1.0, 0.0, 500
Parámetros de la entrada 2 (máximo, mínimo, resolución) 1.0, 0.0, 500
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 1, 1
Desplazamientos UA 2 2, 2
Desplazamientos UA 3 3, 3
Desplazamientos UA 4 4, 4
Desplazamientos UA 5 5, 5
Desplazamientos UA 6 6, 6
Desplazamientos UA 7 7, 7
Desplazamientos UA 8 8, 8
Desplazamientos UA 9 9, 9
Desplazamientos UA 10 6, 5
Tasa de aprendizaje 0.05
Tamaño del arreglo (doubles) 6000

Tabla A.2.: Parámetros CMAC - fcos.

A.3. Parámetros de CMAC para aproximar la función norm

Número de entradas 2
Parámetros de la entrada 1 (máximo, mínimo, resolución) 6.0, -6.0, 500
Parámetros de la entrada 2 (máximo, mínimo, resolución) 6.0, -6.0, 500
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 1, 1
Desplazamientos UA 2 2, 2
Desplazamientos UA 3 3, 3
Desplazamientos UA 4 4, 4
Desplazamientos UA 5 5, 5
Desplazamientos UA 6 6, 6
Desplazamientos UA 7 7, 7
Desplazamientos UA 8 8, 8
Desplazamientos UA 9 9, 9
Desplazamientos UA 10 6, 5
Tasa de aprendizaje 0.05
Tamaño del arreglo (doubles) 6000

Tabla A.3.: Parámetros CMAC - norm.

A.4. Parámetros de CMAC para aproximar la función fedge

Número de entradas 2
Parámetros de la entrada 1 (máximo, mínimo, resolución) 1.0, 0.0, 500
Parámetros de la entrada 2 (máximo, mínimo, resolución) 1.0, 0.0, 500
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 1, 1
Desplazamientos UA 2 2, 2
Desplazamientos UA 3 3, 3
Desplazamientos UA 4 4, 4
Desplazamientos UA 5 5, 5
Desplazamientos UA 6 6, 6
Desplazamientos UA 7 7, 7
Desplazamientos UA 8 8, 8
Desplazamientos UA 9 9, 9
Desplazamientos UA 10 6, 5
Tasa de aprendizaje 0.05
Tamaño del arreglo (doubles) 6000

Tabla A.4.: Parámetros CMAC - fedge.

Apéndice B.

Parámetros de los modelos de clasificación en Minería de Datos

B.1. Parámetros CMAC: Caso de estudio 1

Número de entradas 21
Parámetros de la entrada 1 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 2 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 3 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 4 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 5 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 6 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 7 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 8 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 9 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 10 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 11 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 12 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 13 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 14 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 15 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 16 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 17 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 18 (máximo, mínimo, resolución) 100, 0, 20

Parámetros de la entrada 19 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 20 (máximo, mínimo, resolución) 100, 0, 20
Parámetros de la entrada 21 (máximo, mínimo, resolución) 100, 0, 20
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
Desplazamientos UA 2 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
Desplazamientos UA 3 0, 9, 1, 2, 8, 7, 3, 4, 6, 5, 0, 9, 1, 2, 8, 7, 3, 4, 6, 5, 0
Desplazamientos UA 4 0, 9, 1, 2, 8, 7, 3, 4, 6, 5, 0, 9, 1, 2, 8, 7, 3, 4, 6, 5, 0
Desplazamientos UA 5 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
Desplazamientos UA 6 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
Desplazamientos UA 7 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 3
Desplazamientos UA 8 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
Desplazamientos UA 9 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 3
Desplazamientos UA 10 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
Tasa de aprendizaje 0.5
Tamaño del arreglo (doubles) 3000

Tabla B.1.: Parámetros CMAC: Caso de estudio 1.

B.2. Parámetros MLP: Caso de estudio 1

Número de capas ocultas 1
Número de neuronas 10
Tasa de aprendizaje 0.15
Regla de paro iteración máxima = 100, tasa de error = 0.01

Tabla B.2.: Parámetros MLP: Caso de estudio 1.

B.3. Parámetros C4.5: Caso de estudio 1

Tamaño mínimo de las hojas 5
Nivel de confianza 0.25

Tabla B.3.: Parámetros C4.5: Caso de estudio 1.

B.4. Parámetros CMAC: Caso de estudio 2

Número de entradas 8
Parámetros de la entrada 1 (máximo, mínimo, resolución) 90, 17, 50
Parámetros de la entrada 2 (máximo, mínimo, resolución) 8, 1, 40
Parámetros de la entrada 3 (máximo, mínimo, resolución) 16, 1, 40
Parámetros de la entrada 4 (máximo, mínimo, resolución) 16, 1, 40
Parámetros de la entrada 5 (máximo, mínimo, resolución) 7, 1, 40
Parámetros de la entrada 6 (máximo, mínimo, resolución) 14, 1, 50
Parámetros de la entrada 7 (máximo, mínimo, resolución) 99, 1, 50
Parámetros de la entrada 8 (máximo, mínimo, resolución) 41, 1, 50
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 0, 1, 2, 3, 4, 5, 6, 7
Desplazamientos UA 2 9, 8, 7, 6, 5, 4, 3, 2
Desplazamientos UA 3 0, 9, 1, 8, 2, 7, 3, 6
Desplazamientos UA 4 5, 4, 6, 3, 7, 2, 8, 1
Desplazamientos UA 5 0, 1, 2, 3, 4, 9, 8, 7
Desplazamientos UA 6 6, 7, 8, 9, 5, 4, 3, 2
Desplazamientos UA 7 0, 1, 9, 8, 2, 3, 7, 6
Desplazamientos UA 8 9, 8, 0, 1, 7, 6, 2, 3

Desplazamientos UA 9 5, 6, 7, 8, 9, 0, 1, 2
Desplazamientos UA 10 1, 1, 2, 2, 3, 3, 4, 4
Tasa de aprendizaje 0.15
Tamaño del arreglo (doubles) 12000

Tabla B.4.: Parámetros CMAC: Caso de estudio 2.

B.5. Parámetros MLP: Caso de estudio 2

Número de capas ocultas 1
Número de neuronas 10
Tasa de aprendizaje 0.15
Regla de paro iteración máxima = 100, tasa de error = 0.01

Tabla B.5.: Parámetros MLP: Caso de estudio 2.

B.6. Parámetros C4.5: Caso de estudio 2

Tamaño mínimo de las hojas 5
Nivel de confianza 0.25

Tabla B.6.: Parámetros C4.5: Caso de estudio 2.

B.7. Parámetros CMAC: Caso de estudio 3

Número de entradas 6
Parámetros de la entrada 1 (máximo, mínimo, resolución) 115.00, 1.00, 320
Parámetros de la entrada 2 (máximo, mínimo, resolución) 11476.80, 0.00, 80
Parámetros de la entrada 3 (máximo, mínimo, resolución) 2189.00, 0.00, 80
Parámetros de la entrada 4 (máximo, mínimo, resolución) 713.00, 1.00, 80
Parámetros de la entrada 5 (máximo, mínimo, resolución) 11.00, 0.00, 80
Parámetros de la entrada 6 (máximo, mínimo, resolución) 364.00, 0.09, 80
Número de salidas 1
Número de Unidades de Asociación (UA) 10
Desplazamientos UA 1 3, 2, 4, 3, 5, 2
Desplazamientos UA 2 0, 1, 2, 3, 4, 6
Desplazamientos UA 3 2, 4, 6, 8, 7, 9
Desplazamientos UA 4 5, 9, 5, 6, 9, 3
Desplazamientos UA 5 7, 5, 7, 3, 3, 0
Desplazamientos UA 6 7, 2, 9, 0, 3, 0
Desplazamientos UA 7 7, 3, 7, 9, 8, 2
Desplazamientos UA 8 8, 4, 2, 7, 6, 7
Desplazamientos UA 9 2, 2, 8, 7, 3, 1
Desplazamientos UA 10 2, 7, 0, 5, 1, 8

Tasa de aprendizaje 0.15
Tamaño del arreglo (doubles) 7000

Tabla B.7.: Parámetros CMAC: Caso de estudio 3.

B.8. Parámetros MLP: Caso de estudio 3

Número de capas ocultas 1
Número de neuronas 10
Tasa de aprendizaje 0.15
Regla de paro iteración máxima = 100, tasa de error = 0.01

Tabla B.8.: Parámetros MLP: Caso de estudio 3.

B.9. Parámetros C4.5: Caso de estudio 3

Tamaño mínimo de las hojas 5
Nivel de confianza 0.25

Tabla B.9.: Parámetros C4.5: Caso de estudio 3.

Apéndice C.

Información detallada de las Bases de Datos

C.1. Base de Datos Mushrooms

No.	Nombre	Dominio
1	cap-shape	bell, conical, convex, flat, knobbed, sunken
2	cap-surface	fibrous, grooves, scaly, smooth
3	cap-color	brown, buff, cinnamon, gray, green, pink, purple, red, white, yellow
4	bruises	bruises, no
5	odor	almond, anise, creosote, fishy, foul, musty, none, pungent, spicy
6	gill-attachment	attached, descending, free, notched
7	gill-spacing	close, crowded, distant
8	gill-size	broad, narrow
9	gill-color	black, brown, buff, chocolate, gray, green, orange,pink, purple, red, white, yellow
10	stalk-shape	enlarging, tapering
11	stalk-root	bulbous, club, cup, equal, rhizomorphs, rooted, missing
12	stalk-surface-above-ring	ibrous, scaly, silky, smooth
13	stalk-surface-below-ring	ibrous, scaly, silky, smooth
14	stalk-color-above-ring	brown, buff, cinnamon, gray, orange, pink, red, white, yellow
15	stalk-color-below-ring	brown, buff, cinnamon, gray, orange, pink, red, white, yellow
16	veil-type	partial, universal
17	veil-color	brown, orange, white, yellow
18	ring-number	none, one, two
19	ring-type	cobwebby, evanescent, flaring, large, none, pendant, sheathing, zone
20	spore-print-color	black, brown, buff, chocolate, green, orange, purple, white, yellow
21	population	abundant, clustered, numerous, scattered, several, solitary
22	habitat	grasses, leaves, meadows, paths, urban, waste, woods
23	classes	edible, poisonous

Tabla C.1.: Base de datos *Mushrooms*.

C.2. Base de Datos Adult

No.	Nombre	Tipo	Dominio
1	age	numérico	17 - 90
2	workclass	categórico	Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
3	fnlwgt	numérico	12285 - 1490400
4	education	categórico	Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool
5	education-num	numérico	1 - 16
6	marital-status	categórico	Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
7	occupation	categórico	Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces
8	relationship	categórico	Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
9	race	categórico	White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
10	sex	categórico	Female, Male
11	capital-gain	numérico	0 - 99999
12	capital-loss	numérico	0 - 4356
13	hours-per-week	numérico	1 - 99
14	native-country	categórico	United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece,South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland,France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand,vYugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands
15	class	categórico	>50K, <=50K

Tabla C.2.: Base de datos *Adult*.

C.3. Base de Datos Clothing Store

No.	Nombre	Tipo	Dominio
1	HHKEY	numérico	9.9556E12 - 9.9664E12
2	ZIP_CODE	numérico	0 - 99687
3	REC	numérico	1 - 365
4	FRE	numérico	1 - 115
5	MON	numérico	0.99 - 22511.5
6	CC_CARD	numérico	0 - 1
7	AVRG	numérico	0.49 - 1919.88
8	PC_CALC20	numérico	11 - 19
9	PSWEATERS	numérico	0 - 1
10	PKNIT_TOPS	numérico	0 - 1
11	PKNIT_DRES	numérico	0 - 1
12	PBLOUSES	numérico	0 - 1
13	PJACKETS	numérico	0 - 1
14	PCAR_PNTS	numérico	0 - 1
15	PCAS_PNTS	numérico	0 - 1
16	PSHIRTS	numérico	0 - 1
17	PDRESSES	numérico	0 - 1
18	PSUITS	numérico	0 - 1
19	POUTERWEAR	numérico	0 - 1
20	PJEWERLY	numérico	0 - 1
21	PFASHION	numérico	0 - 1
22	PLEGWEAR	numérico	0 - 1
23	PCOLLSPND	numérico	0 - 1
24	AMSPEND	numérico	0 - 10642.7
25	PSSPEND	numérico	0 - 11476.8
26	CCSPEND	numérico	0.01 - 22511.5
27	AXSPEND	numérico	0 - 4099.92
28	TMONSPEND	numérico	0 - 5562.46
29	OMONSPEND	numérico	0 - 2189
30	SMONSPEND	numérico	0 - 13224.1
31	PREVPD	numérico	0 - 2259.5
32	GMP	numérico	0 - 0.99
33	PROMOS	numérico	0 - 38
34	DAYS	numérico	1 - 713
35	FREEDAYS	numérico	1 - 713
36	MARKDOWN	numérico	0 - 0.95
37	CLASSES	numérico	1 - 37

38	COUPONS	numérico	0 - 32
39	STYLES	numérico	1 - 743
40	STORES	numérico	1 - 19
41	STORELOY	numérico	3 - 7504
42	VALPHONE	numérico	0 - 1
43	WEB	numérico	0 - 1
44	MAILED	numérico	0 - 11
45	RESPONDED	numérico	0 - 11
46	RESPONSERATE	numérico	0 - 100
47	HI	numérico	0.05 - 200
48	LTFREDAY	numérico	0.09 - 364
49	CLUSTYPE	numérico	0 - 50
50	PERCRET	numérico	0 - 40.92
51	RESP	categórico	Y, N

Tabla C.3.: Base de datos *Clothing Store*.

Bibliografía

[1] The gartner group, `www.gartner.com`.

[2] *The Technology Review Ten*. MIT Technology Review, 2001.

[3] The gnu multiple precision arithmetic library, `www.swox.com/gmp`, 2006.

[4] J. Albus. Data storage in the cerebellar model articulation controller (cmac). *ASME J. Dynamic. Syst. Meas. Contr.*, 97:228–233, 1975.

[5] J. S. Albus. A new approach to manipulator control: the cerebellar model articulation controller (cmac). *J. Dynamic Systems, Measurement an control*, 97:220–227, 1975.

[6] Michael J. A. Berry and Gordon S. Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*. Wiley, Indianapolis, Indiana, 2004.

[7] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[8] Peter Cabena, Pablo Hadjinian, Rolf Stadler, JaapVerhees, and Alessandro Zanasi. *Discovering Data Mining: From Concept to Implementation*. Prentice Hall, Upper Saddle River, NJ, 1998.

[9] Peter Chapman, Julian Clinton, Randy Kerber, and Thomas Khabaza. Crisp-dm step-by-step data mining guide, 2000.

[10] Takeshi Fukuda, Yasukiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: scheme, algorithms, and visualization. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 13–23, New York, NY, USA, 1996. ACM Press.

[11] Johannes Gehrke, Raghu Ramakrishnan, and Venkatesh Ganti. Rainforest - a framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery*, 4(2/3):127–162, 2000.

[12] F. H. Glanz and W. T. Miller. Deconvolution using a cmac neural network. In *Proc of the First Annual Meeting of the International Network Society*, page 440, 1988.

[13] F. H. Glanz and W. T. Miller. Deconvolution and nonlinear inverse filtering using a neural network. *Proc. IEEE Int. Conf. Accoust., Speech. Signal Processing. IEEE*, E7.3:2349–2352, 1989.

[14] David Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. MIT Press, Cambridge, MA, 2001.

[15] D. A. Handelman, S. H. Lane, and J. J. Gelfand. Integrating neural networks and knowledge-based systems for intelligent robotic control. *IEEE Contr. Syst. Mag*, 10:77–78, 1990.

[16] Frederick G. Harmon, Andrew A. Frank, and Sanjav S. Joshi. Application of a cmac neural network to the control of a parallel hybrid-electric propulsion system for a small unmanned aerial vehicle. In *Proceedings of International Joint Conference on Neural Networks*, pages 355–360, 2005.

[17] Chao He, Lixin Xu, and Yuhe Zhang. Learning convergence of CMAC algorithm. *Neural Processing Letters*, 14(1):61–74, 2001.

[18] Y. Iiguni. Hierarchical image coding via cerebellar model arithmetic computers. *IEEE Trans. Image Processing*, 5:1393–1401, 1996.

[19] Hemsathapat K., Dagli C.H., and Enke D. Using a neuro-fuzzy-genetic data mining architecture to determine a marketing strategy in a charitable organization's donor database. In *Proceedings of Change Management and the New Industrial Revolution*, pages 64–69, 2001.

[20] Eric R. Kandel, James H. Schwartz, and Thomas M. Jessell. *Principles of neural science*. Elsevier Science, New York, 1991.

[21] Ling Jing Kao and Chih Chou Chiu. Mining the customer credit by using the neural network model withclassification and regression tree approach. In *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, volume 2, pages 923–928, 2001.

[22] D. Knuth. *The Art of Computer Programmiing*, volume 3. Addison Wesley, 1973.

[23] Rachel Konrad. Data mining: Digging user info for gold. `http://zdnet.com.com/2100-11-528032.html?legacy=zdnn`, 2001.

[24] Daniel. T. Larose. *Discovering Knowledge in Data: An Introduction to Data Mining*. Wiley, 2005.

[25] Daniel. T. Larose. *Data Mining, Methods and Models*. Wiley, 2006.

[26] Chien-Cheng Lee, Pau-Choo Chung, and Yieng-Jair Che. Classification of liver diseases from ct images using bp-cmac neural network. In *Proceeding of the 9th IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA 2005)*, pages 118–121, 2005.

[27] Wenke Lee, Salvatore J. Stolfo, Philip K. Chan, Eleazar Eskin, Wei Fan, Matthew Miller, Shlomo Hershkop, and Junxin Zhang. Real time data mining-based intrusion detection. In *Proc. Second DARPA Information Survivability Conference and Exposition*, pages 85–100, 2001.

[28] Chih-Min Lin and Ya-Fu Peng. Missile guidance law design using adaptive cerebellar model articulation controller. *Neural Networks, IEEE Transactions on*, 16(3):636–644, 2005.

[29] Xu-Mei Lin, Tao Mei, and Hui-Jing Wang. Neural optimal control of robotic manipulators with cmac networks. In *Proceedings of the IEEE International Conference on Mechatronics and Automation*, pages 1391–1396, 2005.

[30] Sara Madeira, Joao M. Sousa, and Uzay Kaymak. Modeling charity donations using target selection for revenue maximazation. In *In Proc. of 2003 IEEE International Conference on Fuzzy Systems*, volume 1, pages 654–659, 2003.

[31] Manish Mehta, Rakesh Agrawal, and Jorma Rissanen. Sliq: A fast scalable classifier for data mining. In *Extending Database Technology*, pages 18–32, 1996.

[32] W. T. Miller and F. H. Glanz. Cmac: An associative neural network alternative to backpropagation. In *Proc. of the IEEE*, volume 78, pages 1561–1567, 1990.

[33] John Naisbitt. *Megatrends*. Warner Books, New York, 1986.

[34] William J.E. Potts. *Decision Tree Modeling, Course Notes*. SAS Institute, 2002.

[35] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106.

[36] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[37] Ricco Rakotomalala. Tanagra: a free software for research and academic purposes. In *Proceedings of EGC 2005, RNTI-E-3*, volume 2, pages 697–702, 2005.

[38] D. Reay. Nonlinear channel equalization using associative memory neural networks. In *Proc. Int. Wkshp Appl. Neural Networks Telecom*, pages 17–24, 1995.

[39] A. Artés Rodriguez, F. Gonzalez Serrano, A. Figueiras Vidal, and L. Weruaga Prieto. Compensation of bandpass nonlinearities by look-up tables and cmac. In *Proc. Int. Wkshp. Appl. Neural Networks Telecom*, pages 25–32, 1995.

[40] John C. Shafer, Rakesh Agrawal, and Manish Mehta. Sprint: A scalable parallel classifier for data mining. In T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda, editors, *Proc. 22nd Int. Conf. Very Large Databases, VLDB*, pages 544–555. Morgan Kaufmann, 3–6 1996.

[41] Weifeng Shil and Tianhao Tang. Cmac neural networks based combining control for marine diesel engine generator. In *Proceedings of the 5th World Congress on Intelligent Control and Automationn*, pages 2652–2655, 2004.

[42] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining.* Addison-Wesley, 2006.

[43] Bhavani Thuraisingham, Chris Clifton, John Maurer, and Marion G. Ceruti. Real-time data mining of multimedia objects. *isorc*, 00:0360, 2001.

[44] Bhavani Thuraisingham, Latifur Khan, Chris Clifton, John Maurer, and Marion Ceruti. Dependable real-time data mining. *isorc*, 00:158–165, 2005.

[45] Sue Walsh. *Applying Data Mining Techniques Using Enterprise Miner, Course Notes.* SAS Institute, 2002.

[46] Zi-Qin Wang, J. L. Schiano, and M Ginsberg. Hash-coding in cmac neural networks. *IEEE International Conference on Neural Networks*, 3:1698–1703, 1996.

[47] Marco Wiering, R.P. Salustowicz, and Juergen Schmidhuber. Cmac models learn to play soccer. In *In Proceedings of the 8th International Conference on Artificial Neural Networks (ICANN'98)*, volume 1, pages 443–448, 1998.