WorkShop
EIG2009

**WORKSHOP INTERNACIONAL**
***EIG2009***
*Departamento de Ingeniería de Sistemas*
*Universidad de La Frontera – Chile*
*Diciembre 3 y 4, 2009*

# Algoritmos de Minería de Datos Extendidos con Comparadores Difusos y su Impacto en los Indicadores de Gestión*

Angélica Urrutia[1], Juan Méndez[1] y Claudio Gutiérrez-Soto[2]
[1]Universidad Católica del Maule, aurrutia@ucm.cl.
[2]Universidad del Bío-Bío, cogutier@ubiobio.cl

**Resumen:** La Minería de Datos es utilizada en diferentes disciplinas para la búsqueda de patrones y modelos ocultos en las Bases de Datos. Esta generalmente es aplicada en las áreas de negocios y marketing. Sin embargo, su aplicación y uso quedan finalmente a disposición de quienes manejan este conocimiento, por lo que debe de ser trasformado en información útil para los niveles superiores. En este artículo se presenta una extensión de algoritmos de Minería de Datos existentes en SQL Server 2008 utilizando comparadores difusos, que permitirán al usuario final contar una herramienta la cual aporta información útil, oportuna; así como también conocimiento de las variables que influyen de manera directa en los indicadores de gestión. Los aportes principales de este trabajo son; en primer lugar el análisis de los algoritmos que pueden ser extendidos utilizando comparadores difusos, segundo la aplicación en casos reales de los algoritmos extendidos utilizando comparadores difusos en una Base de Datos de prueba. Los resultados de los indicadores de gestión extendidos se pueden visualizar tanto como Cliente-Servidor como en Web.
**Palabras claves**: minería de datos, extensión de algoritmos con comparadores difusos, indicadores de gestión, comparadores difusos, Consultas complejas.

## 1. Introducción

La investigación de este trabajo, considera la implementación de algunos de los algoritmos de minería de datos que están incluidos en SQL Server 2008, dichos algoritmos son: Árboles de decisión, Bayes Naive, y Clústeres (existen cinco algoritmos más que no son trabajados aquí). Nuestra propuesta considera una implementación, que utiliza un caso de estudio, que extienden las consultas clásicas a *comparadores difusos de necesidad y posibilidad* utilizando la lógica difusa, de tal forma de prestar un mejor servicio de información al usuario final que requiere análisis de datos e indicadores de gestión.

En [6] se muestra como la *lógica difusa* se ha utilizado para extender las Bases de Datos Relacionales (BDR) a Bases de Datos Relacionales Difusas (BDRD). Algunas propuestas de uso de comparadores difusos aplicados al resultado de datos obtenidos por un Clúster de minería de datos, sobre un caso del área de turismo, se encuentra en [1], aquí los autores extienden el FSQL de [4] para utilizarlo en consulta difusas potenciando los resultados de los datos obtenidos por un Clúster. Otras extensiones de conjuntos difusos aplicados a casos de minería de datos se encuentran en [2].

La teoría de conjuntos difusos parte de la teoría clásica de conjuntos, añadiendo una función de pertenencia al conjunto, definida ésta como un número real entre 0 y 1 [7]. Así, se introduce el concepto de conjunto o subconjunto difuso asociado a un determinado valor lingüístico, definido por una palabra, adjetivo o etiqueta lingüística $A$. Para cada conjunto o subconjunto difuso se define una *función de pertenencia o inclusión* $\mu_A(u)$, donde se define un conjunto difuso $A$ sobre un universo de discurso $U$ (dominio ordenado) es un conjunto de pares dado por: $A = \{\mu_A(u)/u : u \in U, \mu_A(u) \in [0,1]\}$, Donde, $\mu$ es la llamada función de pertenencia y $\mu_A(u)$ es el *grado de pertenencia* del elemento $u$ al conjunto difuso $A$. Este grado oscila entre los extremos 0 y 1, considerando que $\mu_A(u) = 0$, indica que $u$ no pertenece en absoluto al conjunto difuso $A$ *y que* $\mu_A(u) = 1$, indica que $u$ pertenece totalmente al conjunto difuso $A$. El mismo concepto se aplica a los comparadores de necesidad y posibilidad expuestos en [3,4,6].

Considerando lo anteriormente expuesto, presentamos en los siguientes apartados la propuesta de una arquitectura para procesos de minería de datos y consultas con comparadores clásicos y difusos trabajados con la teoría de conjuntos difusos, además de analizar el resultado diferentes escenarios con los algoritmos de minería de datos que más se ajusten al problema. La implementación de las consultas tiene el objetivo de proponer una extensión, siempre y cuando los resultados entregados por el algoritmo lo permitan.

* Los resultados mostrados en este trabajo son financiados por proyecto interno 2009-2010 de la UCM.

Finalmente se analizan ambos resultados de las consulta y se ven sus ventajas.

## 2. Comparadores Clásicos y Difusos

Los operadores de comparación clásicos son 6 (ver Tabla 1), los cuales se pueden utilizar tanto para comparar números como para comparar textos y fechas.

| Comparador | Significado |
|---|---|
| = | Igual |
| > | Mayor que |
| < | Menor que |
| >= | Mayor o igual que |
| <= | Menor o igual que |
| <> o != | Distinto |

**Tabla 1:** Comparadores clásicos.

Una extensión de los comparadores clásicos son los comparadores difusos los que han sido definidos como *Comparadores Difuso Generalizado* del modelo GEFRED [4], aquí se define un tipo de comparador general basado en *comparador clásico* existente (=, >, <...), el único requisito que establece es que el *Comparador Difuso* debe respetar los resultados de los comparadores clásicos cuando se comparan distribuciones de posibilidad que expresan valores *crisp*.

En [4] se propone otros comparadores (*Mayor difuso, Mayor ó Igual difuso, Mucho mayor difuso*) en su versión de posibilidad y necesidad (ver Tabla 2), así como la comparación de distribuciones de posibilidad sobre dominios subyacentes no ordenado. Cabe destacar que al igual que en SQL, los comparadores difusos, que son implementados en el servidor FSQL de [3,4,6], pueden comparar una columna (o atributo) con una constante o dos columnas del mismo tipo. Los comparadores de necesidad son más restrictivos que los de posibilidad, por lo que su grado de cumplimiento es siempre menor que el grado de cumplimiento obtenido por su correspondiente comparador de posibilidad. Los comparadores de necesidad recuperan menos tuplas y estas tuplas cumplirán necesariamente con las condiciones impuestas en la consulta.

| Comparador Difuso | Significado |
|---|---|
| FEQ, NFEQ | Posiblemente, Necesariamente Igual |
| FGT, NFGT | Posiblemente, Necesariamente Mayor que |
| FGEQ, NFGEQ | Posiblemente, Necesariamente Mayor o Igual que |
| FLT, NFLT | Posiblemente, Necesariamente Menor que |
| FLEQ, NFLEQ | Posiblemente, Necesariamente Menor o Igual que |
| MGT, NMGT | Posiblemente, Necesariamente Mucho Mayor que |
| MLT, NMLT | Posiblemente, Necesariamente Mucho Menor que |

**Tabla 2:** Comparadores difusos de posibilidad y necesidad.

En general, los comparadores de necesidad exigen que la condición sea satisfecha con cierto grado de necesidad, mientras los comparadores de posibilidad miden en qué posibilidad medida (o grado) es posible que la condición se cumpla.

## 3. Arquitectura Propuesta para Consultas Flexibles en Minería de Datos

La arquitectura propuesta, permite satisfacer *indicadores de Gestión de Información* (GI), obtenidos desde la aplicación de un proceso de minería de datos, (ver Figura 1). A dicho resultado, se le aplican consultas utilizando comparadores clásicos y difusos como una extensión.

La aplicación de Minería de Datos que utiliza comparadores clásicos y difusos, está sustentada en una taxonomía de 7 capas, lo que permite comparar, qué aportes o en qué casos se obtiene un mejor resultado de consultas. Las capas se inician desde los datos existentes en una base de datos fuente extraídos y almacenados en un Almacén de Datos, a partir de aquí, se pueden implementar distintos escenarios y aplicar uno o más algoritmos de minería de datos, para posteriormente ser consultados con comparadores clásicos y una extensión de comparadores difusos. Finalmente, ambos resultados son analizados desde el punto de vista descriptivo, lo cual entrega información valiosa al usuario final desde el punto de vista de la gestión

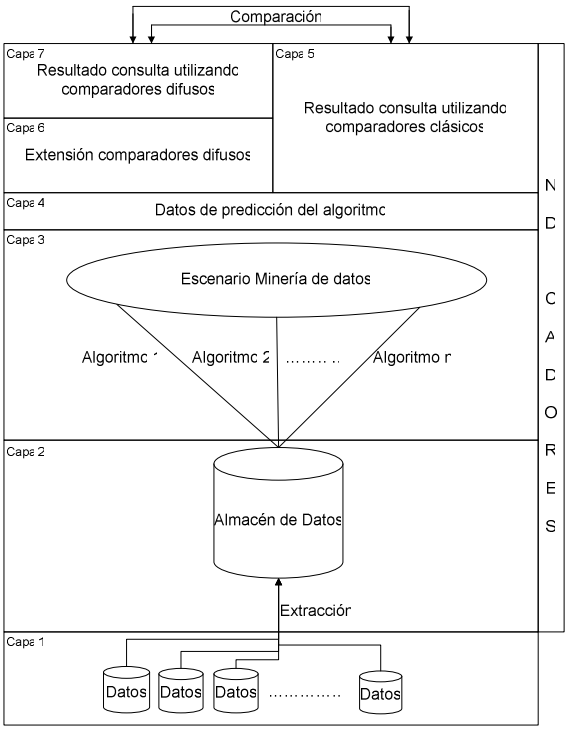


**Fig. 1:** Arquitectura para procesos de minería de datos y consultas con comparadores clásicos y difusos. El proceso se realiza de forma secuencial en cada capa.

**Capa 1:** Aquí se encuentran los datos de origen, que pueden ser: un archivo plano, un archivo excel, una base de datos, entre otros. Estos datos deben cubrir los requerimientos para los indicadores de gestión.
**Capa 2:** Tiene como objetivo la extracción de datos, los cuales son analizados en la especificación de requerimientos, y utilizados en el proceso de minería de datos, lo cual es vital para el proceso de la toma de decisiones.
**Capa 3:** En esta etapa se utilizan los algoritmos de minería de datos, que dependiendo del caso o escenario se pueden aplicar el que más complemente el caso de estudio o indicador de gestión. Aquí debe estar definida la problemática y los indicadores de gestión a utilizar.
**Capa 4:** Aquí se definen los datos de predicción para el algoritmo, estos datos deben ser coherentes con la predicción que se requiere para los indicadores de gestión definidos en la capa 2 y 3.
**Capa 5:** En esta capa se procede a realizar las consultas que solicitan los usuarios finales o que fueron especificadas en los requerimientos del problema, dichas consultas son los resultados de consultas utilizando comparadores clásicos de la tabla 1.
**Capa 6:** En esta capa se analiza cual de los comparadores de necesidad o posibilidad, de los presentados en la Tabla 2 es el más adecuado para la consulta. Estos comparadores han sido implementados en código SQL y se encuentra, explicado en el apartado 3 y en [5].
**Capa 7:** Aquí se aplican los comparadores de necesidad y posibilidad seleccionados en la capa anterior. Las consultas ejecutadas deben ser la misma de la capa 5. La comparación es la última capa, y tiene como objetivo analizar los resultados de las consultas obtenidos de las capas 5 y 6, para especificar por qué puede ser útil uno u otro comparados y cuál es la diferencia de sus resultados.

## 4. Extensión de los Comparadores Difusos y su Implementación

Aquí se presenta la implementación de los comparadores difusos. La implementación de cada comparador difuso está programada para SQL Server 2008. Los comparadores difusos implementados son los que se muestran la Tabla 2, aquí sólo se presentará uno de ellos a modo de ejemplo. La sigla de cada comparador es implementada de la siguiente forma: FEQ, posiblemente igual (Fuzzy EQual), Implementación de comparador difuso FGT, posiblemente mayor que (Fuzzy Greater Than) e Implementación de comparador difuso FGEQ, posiblemente mayor o igual (Fuzzy Greater EQual).

A continuación se presenta el código de la función FEQ, a modo de ejemplo, la cual implementa el comparador difuso como una función de SQL Server.

```
SET ANSI_NULLS ON GO
SET QUOTED_IDENTIFIER ON GO
CREATE FUNCTION FEQ
(-- Add the parameters for the
function here
@crisp FLOAT,@F1 FLOAT,@F4 FLOAT)
RETURNS FLOAT
AS
```

```
BEGIN
-- Declare the return variable here
DECLARE @valor FLOAT
-- Add the T-SQL statements to
compute the return value here
IF (@crisp > (@F1 - @F4) and @crisp
< (@F1 + @F4))
SET @valor=ROUND(((@F4 - Abs(@F1 -
@crisp)) / @F4), 4);
ELSE
SET @valor=0;
-- Return the result of the function

RETURN @valor;
END
GO
```

## 5. Aplicación de la Arquitectura Propuesta (capas 1 al 5)

Para validar nuestra propuesta, hemos utilizado un caso de estudio que nos permitió obtener los resultados de la implementación de los comparadores difusos. A continuación se describe la aplicación de cada una de sus capas de la Figura 1.

Capa 1 (Datos): Para esta se utilizó la base de datos Adventure Works Cycles, la cual es presentada como la base de datos de ejemplo de SQL Server 2008.

Capa 2 (Definición del problema): El modelo de datos con el cual se trabajó, es una parte del Data Warehouse de Adventure Works Cycles. La Figura 2, muestra un submodelo del Data Warehouse que es la entrada para el almacén de datos que implementa los distintos procesos de minería de datos.

Capa 3 (Escenarios y algoritmos de minería de datos): El escenario que se utilizó en este caso, es el de *Correo Directo*, aquí se implementaron tres algoritmos: árboles de decisión, clústeres y Bayes Naive. En el caso de SQL Server da la posibilidad de sugerir cuales serian los mejores algoritmo según el caso.

Para el *Escenario de Correo Directo* se implementaron los algoritmos mencionados anteriormente, aunque los algoritmos son completamente diferentes, en este escenario se requiere decidir, ***Qué tan probable es que una persona con ciertas características compre algún producto ofrecido.***

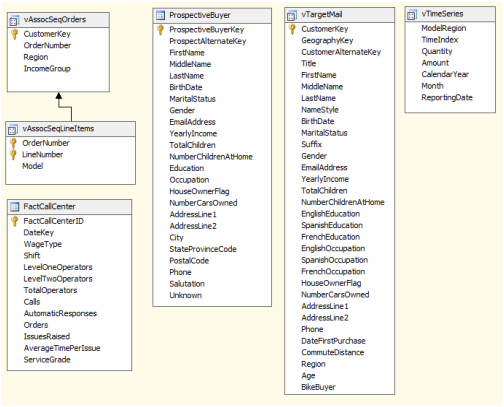


**Fig. 2:** Modelo utilizado para el proceso de minería de datos.

Por lo cual se debe analizar cuál de los tres algoritmos realiza esta labor de mejor manera, es decir satisface el indicador a cabalidad. Para esto existe una herramienta llamada "Grafico de elevación", la cual se puede encontrar en la pestaña "Gráfico de precisión de minería de datos" [5].

Capa 4 (Predicción de los escenarios de minería de datos): Para el Escenario de Correo Directo, el **indicador definido es, *Cuál es la probabilidad de que un posible cliente compre una bicicleta***.

Para este caso, se utilizó una herramienta que puede seleccionar los algoritmos que se quieren comparar o seleccionar, algún conjunto de datos específico. La Figura 3, muestra el gráfico de elevación, en el podemos observar que el algoritmo que más se ajusta a nuestro modelo ideal (predicción perfecta) es el algoritmo de Árboles de decisión, por lo cual será este el algoritmo elegido para realizar las predicciones y la posterior extensión. Cabe señalar que la extensión a realizar también es valida para los otros dos modelos (Clústeres y Bayes Naive), ya que como se mencionó anteriormente, aunque los algoritmos son completamente diferentes, en esta ocasión se utilizan con el mismo propósito (predecir un atributo).

Cuando se utiliza el algoritmo de árboles de decisión para realizar predicciones, genera una consulta de predicción sobre una tabla de casos (ver Figura 4), dicha consulta entrega la probabilidad de que -*cada persona de la tabla de casos compre un producto*-, por tanto, se realizó predicciones sobre la tabla de casos, utilizando el algoritmo de Árboles de Decisión, dicha

tabla de casos contiene perfiles de probables clientes, los cuales analizados con este algoritmo. Como resultado se obtiene un porcentaje, el cual dirá *que tan probable es que un potencial cliente se convierta en un cliente*. Cumpliendo así con el indicador definido en esta capa.

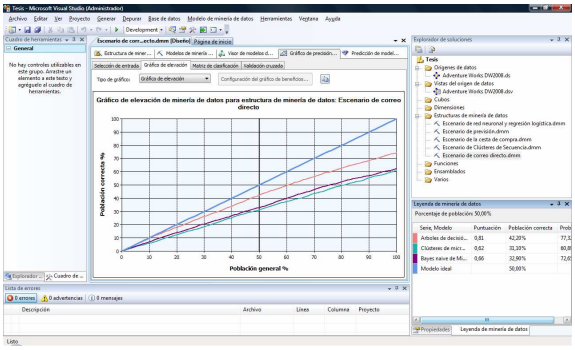


**Fig. 3:** Gráfico de elevación de precisión de algoritmos de minería de datos.

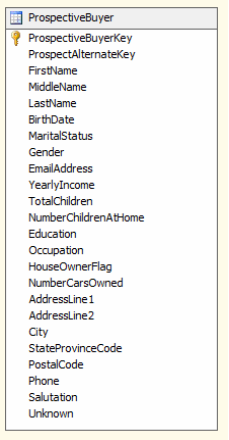


**Fig. 4:** Definición de tabla de casos.

Una vez implementado el algoritmo de *Árboles de Decisión* en el Escenario de Correo Directo, se utilizó la opción *Predicción de modelos de minería de datos* del SQL Server. El paso siguiente es la selección de la *tabla ProspectuveBuyer*. Finalmente, se selecciona la *Función de Predicción*, con el atributo *PredictProbability*, obtenido por el algoritmo de *Árboles de Decisión* que arroja como resultado, la probabilidad de adquisición de algún producto por cada comprador. La Figura 5 a) muestra la pantalla final de este proceso. La Figura 5 b), muestra los resultados arrojados en la predicción, que es la probabilidad de que cada posible comprador adquiera un producto (en este caso una bicicleta).

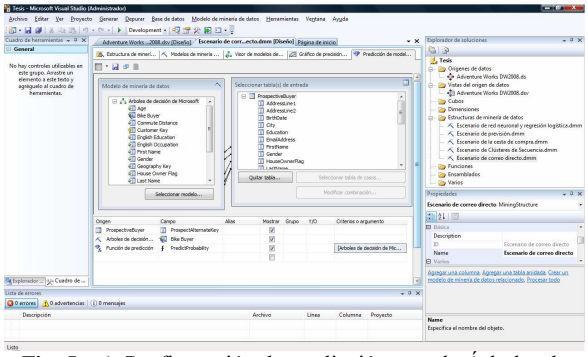

**Fig. 5:** a ) Configuración de predicción usando Árboles de decisión

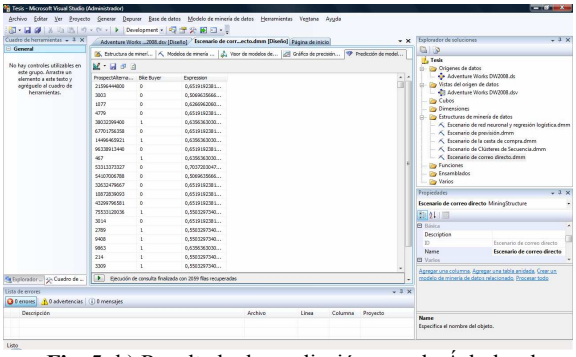

**Fig. 5:** b) Resultado de predicción usando Árboles de decisión.

El resultado se almacena en la tabla ResultadoCorreoDirecto en la base de datos Adventure Works DW 2008, con un atributo *ProspectAlternateKey* que es identidicador del posible cliente, el atributo *Bike Buyer* puede tener dos valores 0 (indica que no es comprador de bicicletas) o 1 (indica que el posible cliente es comprador de bicicletas) y el atributo *Expression* que almacena la probabilidad de que el posible cliente adquiera un producto (INDICADOR), siendo este campo el que predijo el algoritmo.

## 6. Análisis de resultados: Comparación consulta clásica y consulta difusa

Este apartado tiene por objetivo presentar los resultados obtenidos al utilizar los comparadores clásicos (Capa

5), los comparadores difusos (Capa 7) implementados en el la capa 6, y las consultas ejecutadas sobre los resultados de predicción (Capa 4). Cada capa fue implementada en el *Escenario de Correo Directo*. El análisis de resultados fue obtenido de las predicciones realizadas por el algoritmo de *Árboles de Decisión* sobre la tabla *ProspectuveBuyer*. Para el análisis de los resultados se utilizó el valor *crisp* 0.9 con el difuminador +/- 0.1 (0.9 – 0.1, 0.9 + 0.1), de donde se obtiene el rango ]0.8 a 1[, es decir de un 80% a un 100% de probabilidad sin incluir ambos extremos. El valor crisp escogido corresponde a los posibles clientes que tienen mayor probabilidad de adquirir un producto difuminando 0.9.

**a) Comparación usando Expression FEQ 0.9 y Expression = 0.9**

Para realizar esta comparación se usaron dos consultas, una consulta utilizando el comparador "=" y otra consulta usando la función FEQ (posiblemente igual). La siguiente consulta, es una Consulta clásica y arrojo 0 registros (Capa 5). Posteriormente se ejecutó la consulta difusa (Capa 7), aquí al igual que en los próximos comparadores difusos se utiliza la condición "> 0" usado en la cláusula *where*, la cual se debe a que la función difusa (FEQ) retorna un grado de pertenencia entre 0 y 1, es decir retorna el grado de pertenencia que posee el valor *crisp* al valor 0.9 difuminado en +/- 0.1, por lo que se debe escoger solo aquellos valores mayores a 0, considérese que el valor 0 indica que no pertenece al conjunto (Aplíquese esta lógica también para los demás comparadores difusos). El resultado de la consulta difusa se muestra en la tabla 3 con los tres atributos que se obtienen por la tabla de resultado de casos en SQL Server.

```
select * from
[AdventureWorksDW2008].[dbo].[Resulta
doCorreoDirecto]
where (select
[AdventureWorksDW2008].[dbo].[feq]
(Expression,0.9,0.1)) > 0
```

| | ProspectAlternateKey | Bike Buyer | Expression |
|---|---|---|---|
| **1** | 85459846854 | 0 | 0,926029962546816 |
| **2** | 31616555452 | 1 | 0,836826910064208 |
| **…** | … | … | … |
| **32** | 6294 | 1 | … |

**Tabla 3:** Resultados de la consulta usando Expression FEQ 0.9.

Análisis de resultados usando *Expression FEQ 0.9 y Expression = 0.9,* considera que una consulta clásica entrego **cero** tuplas de resultados y la consulta difusa entrego **32** tuplas. Que la consulta clásica halla retornado 0 registros se debe a que no existe ningún posible cliente, para este caso, que posea exactamente un 90% de probabilidad de comprar, en la Figura 6 se muestra exactamente los valores consultados de la línea punteada al 0,9. En cambio la consulta difusa retorna 32 tuplas, ya que abarco un rango entre 0.8 y 1 usando para su extracción del comparador, en la gráfica el comprador se muestra en la Figura 6 con la línea continua.

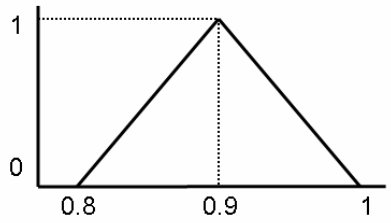


**Fig. 6:** Gráfico del valor establecido y la difuminación asignada.

Como se puede observar en la Figura 6, el resultado de usar el comparador difuso sobre el comparador clásico es notorio ya que el comparador clásico sólo abarca un elemento (0.9) del conjunto de resultados que abarca el comparador difuso (]0.8 a 1[).

**b) Comparación usando Expression FGT 0.9 y Expression > 0.9**

Al igual que en el caso anterior se comparó una consulta clásica con una consulta difusa, la consulta difusa usará la función FGT (posiblemente mayor que). La consulta clásica, se muestra a continuación y los resultados en la Tabla 4.

```
select * from
[AdventureWorksDW2008].[dbo].[Resulta
doCorreoDirecto]
where Expression > 0.9
```

| | ProspectAlternateKey | Bike Buyer | Expression |
|---|---|---|---|
| **1** | 85459846854 | 0 | 0,926029962546816 |

| | | | |
|---|---|---|---|
| **2** | 2937 | 0 | 0,926029962546816 |
| **…** | **…** | **…** | **…** |
| **10** | 2745 | 0 | 0,926029962546816 |

**Tabla 4:** Resultado de la consulta usando Expression > 0.9.

La consulta difusa, se muestra a continuación y los resultados en la Tabla 5.

```
select * from
[AdventureWorksDW2008].[dbo].[Resulta
doCorreoDirecto]
where (select
[AdventureWorksDW2008].[dbo].[fgt]
(Expression,0.9,0.1)) > 0
```

| | ProspectAlternateKey | Bike Buyer | Expression |
|---|---|---|---|
| **1** | 85459846854 | 0 | 0,926029962546816 |
| **2** | 2937 | 0 | 0,926029962546816 |
| **…** | **…** | **…** | **…** |
| **10** | 2745 | 0 | 0,926029962546816 |

**Tabla 5:** Resultados de la consulta usando Expression FGT 0.9.

En este caso las dos consultas han retornado los mismos datos, 10 tuplas en ambos casos tanto para la condición Expression > 0.9 como para la condición Expression FGT 0.9, y son las mismas tuplas, lo que en un principio ha sido un poco sorprendente, pero analizando el código de la función FGT, este resultado es correcto, ya que el comparador difuso FGT se comporta igual que el comparador clásico ">", es decir ambos comparadores aceptan como verdadero valores mayores que 0.9 (sin incluirlo). Lo anteriormente expuesto se puede ver con mayor claridad en la Figura 7 representado en la gráfica por el color más oscuro, en ella se puede apreciar que el comparador difuso FGT no incluye los valores menores a 0.9, para el comparador clásico tampoco.

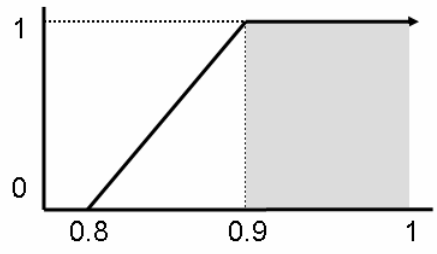


**Fig. 7:** Grafico de valores para el comparador difuso FGT y el comparador clásico >.

**c) Comparación usando Expression FGEQ 0.9 y Expression >= 0.9**

Como ya se ha expuesto en los dos casos anteriores a) y b), se compara una consulta clásica y una consulta difusa, para esta última se usa la función FGEQ (posiblemente mayor o igual que) dicha función difusa es la implementación del comparador difuso FGEQ, que ha sido implementada de la misma forma que los otros comparadores.

La consulta clásica, se muestra a continuación y los resultados en la Tabla 6.

```
select * from
[AdventureWorksDW2008].[dbo].[Resulta
doCorreoDirecto]
where Expression >= 0.9
```

| | ProspectAlternateKey | Bike Buyer | Expression |
|---|---|---|---|
| **1** | 85459846854 | 0 | 0,926029962546816 |
| **2** | 2937 | 0 | 0,926029962546816 |
| **…** | **…** | **…** | **…** |
| **10** | 2745 | 0 | 0,926029962546816 |

**Tabla 6:** Resultado de la consulta usando Expression >= 0.9.

La consulta difusa, se muestra a continuación y los resultados en la Tabla 7.

```
select * from
[AdventureWorksDW2008].[dbo].[Resulta
doCorreoDirecto]
where (select
[AdventureWorksDW2008].[dbo].[fgeq]
(Expression,0.9,0.1)) > 0
```

| | ProspectAlternateKey | Bike Buyer | Expression |
|---|---|---|---|
| **1** | 85459846854 | 0 | 0,926029962546816 |
| **2** | 31616555452 | 1 | 0,836826910064208 |
| **…** | **…** | **…** | **…** |
| **32** | 6294 | 1 | 0,836826910064208 |

**Tabla 7:** Resultados de la consulta usando FGEQ 0.9.

Las tuplas obtenidas como resultados usando Expression FGEQ 0.9 son 32 y Expression >= 0.9 son 10. En este caso se puede observar con claridad la amplitud de valores que entrega el comparador difuso

FGEQ (ver Figura 8 b)) versus el comparador clásico ">=" (ver Figura 8 a)). Se consideró todo el espectro de datos para el comparador difuso FGEQ, como se puede recordar la función en la cual está implementado el comparador retorna un grado de pertenencia y al considerar el grado de pertenencia "> 0" se considerar todos los valores. El espectro de valores para ambos comparadores se puede observar más claramente en la Figura 8.

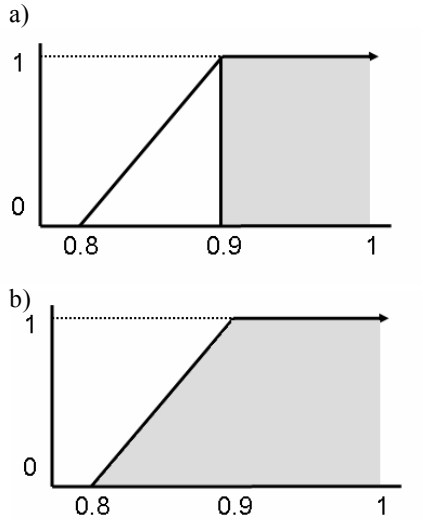


**Fig. 8:** a) Gráfico de valores para el comparador clásico >=, b) Gráfico de valores para el comparador difuso FGEQ.

Es importante notar que el comparador clásico incluye el valor 0.9 como un valor verdadero. También es importante resaltar que el comparador difuso no incluye el valor del extremo izquierdo (0.8) como verdadero al valor difuminado, pero cada comparador devuelve su propio valor entre cero y uno, aquí solo se consideran los mayores que cero para cada comparador.

## 7. Conclusiones

La lógica difusa junto con la minería de datos son herramientas muy poderosas a la hora de trabajar con indicadores de gestión para el apoyo a la toma de decisiones.

En este artículo, se ha presentado un desarrollo, análisis e implementación de algoritmos de minería de datos, con comparadores difusos. De esto es posible concluir que, los comparadores difusos entregan más tuplas para el análisis de datos obtenidos por un algoritmo de minería de datos para un indicador de gestión determinado, lo que puede apoyar mejor la toma de decisiones, ya que estos resultados podrían ser más entendibles desde el punto de vista descriptivo para quienes toman las decisiones.

Los valores a difuminar son importantes de especificar y se deben hacer en conjunto con el usuario final, ya que el resultado de las consultas, deben estar asociadas al indicador de gestión.

Como trabajo futuro se pretende extender otro tipo de componentes de la lógica difusa a los datos o algoritmos de minería de datos.

## Referencias


[1]. Carrasco R., Araque F., Salguero A., Vila M. A..(2008). Applying Fuzzy Data mining to Tourism Area. Charper XXII. In Handbook of Research on Fuzzy Information Processing in Databases, Vol. II, pp. 563-589. Information Science Reference (http://www.info-sci-ref.com).

[2]. Galindo, J. (Ed.), (2008). Section IV Fuzzy Data Mining. Handbook of Research on Fuzzy Information Processing in Databases. Hershey, PA, USA: Information Science Reference (http://www.info-sci-ref.com).

[3]. Galindo, J., Urrutia, A., Piattini, M. (2006). Fuzzy Databases: Modeling, Design and Implementation. Idea Group Publishing Hershey, USA.

[4]. Galindo, José (1999). "Tratamiento de la Imprecisión en Bases de Datos Relacionales: Extensión del modelo y adaptación de los SGBD Actuales", Tesis Doctoral, universidad de Granada, España, 1999.

[5]. Mendez J. (2009). "Extensión de algoritmos de minería de datos con comparadores difusos". Tesis de Licenciatura en Ciencias de la Ingeniería, Universidad Católica del Maule 2009.

[6]. Urrutia, A., Tineo, L., & Gonzalez, C. (2008). FSQL and SQLf: Towards a Standard in Fuzzy Databases. In Handbook of Research on Fuzzy Information Processing in Databases, Vol. I, pp. 270-298. Information Science Reference (http://www.info-sci-ref.com).

[7]. Zadeh L.A. (1965); Fuzzy Sets. Information and Control, 8, pp. 338-353.

* Los resultados mostrados en este trabajo son financiados por proyecto interno 2009-2010 de la UCM.