CONSULTAS CON COMPARADORES DIFUSOS EN ALGORITMOS DE MINERÍA DE DATOS: EN SQL SERVER 2008 *

Angélica Urrutia[1], Claudio Gutiérrez-Soto[2] y Juan Méndez[1]

[1]Universidad Católica del Maule, aurrutia@ucm.cl.
[2]Universidad del Bío-Bío, cogutier@ubiobio.cl

Resumen

En este artículo se presenta una extensión de algoritmos de Minería de Datos existentes en SQL Server 2008, utilizando comparadores difusos de posibilidad 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 un conjunto de pasos y la aplicación de la propuesta en un caso real.

Palabras claves: Minería de datos, Comparadores difusos, Consultas complejas con algoritmos de Minería de Datos.

1. INTRODUCCIÓN

Nuestra propuesta considera una implementación que utiliza un caso de estudio, que extienden las consultas clásicas a *comparadores difusos de 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. Las consultas se ejecutan sobre la implementación de los algoritmos Árboles de decisión, Bayes Naive, y Clústeres de Minería de Datos (MD), los cuales están incluidos en SQL Server 2008.

Algunos artículos analizados muestran como la *lógica difusa* se ha utilizado para extender las Bases de Datos Relacionales (BDR) a Bases de Datos Relacionales Difusas (BDRD) [6]. Una propuesta de los comparadores difusos aplicados al resultado de datos obtenidos por un clústeres 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].

Por otro lado, la teoría de conjuntos difusos parte de la teoría clásica de conjuntos, añadiendo una función de pertenencia al conjunto a través de un número real entre 0 y 1 [6, 8]. 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) y es un conjunto de pares dado por: $A = \{\mu_A(u)/u : u \in U, \mu_A(u) \in [0,1]\}$, Donde, μ 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, 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 de comparadores, siempre y cuando los resultados entregados por el algoritmo lo permitan. Finalmente se analizan ambos resultados de las consulta y se analizan sus ventajas.

2. COMPARADORES CLÁSICOS Y DIFUSOS

Los operadores de comparación clásicos son; igual, mayor que, menor que, mayor o igual que, menor o igual que y distinto, los cuales se pueden utilizar tanto para comparar números como para comparar textos y fechas.

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

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 se establece es que el *Comparador Difuso* debe respetar los resultados de los comparadores clásicos cuando se comparan distribuciones de posibilidad que expresan valores *crisp*.

También, en [4] se proponen otros comparadores (*Mayor difuso, Mayor ó Igual difuso, Mucho mayor difuso*) en su versión de posibilidad y necesidad (ver Tabla 1), así como la comparación de distribuciones de posibilidad sobre dominios subyacentes no ordenados. Cabe destacar que al igual que en SQL, los comparadores difusos son implementados en un servidor FSQL que se presentan en [3,4,6] y 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.

Tabla 1: Comparadores difusos.

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

En general, los comparadores de necesidad exigen que la condición sea satisfecha con cierto grado, de necesidad aunque no sea completamente, mientras los comparadores de posibilidad miden en qué posibilidad de medida (o grado) es posible que la condición se cumpla.

3. COMPARADORES DIFUSOS Y SU IMPLEMENTACIÓN

Este apartado tiene por objetivo presentar la implementación de los comparadores difusos que fueron necesarios para el desarrollo de esta investigación. La implementación de cada comparador difuso está programada en código SQL para SQL Server 2008. Los comparadores difusos de posibilidad implementados son los que muestra la Tabla 1. Al igual que en [4] la sigla de cada comparador es implementado de la siguiente forma: FEQ posiblemente igual (Fuzzy EQual), FGT posiblemente mayor que (Fuzzy Greater Than), FGEQ posiblemente mayor o igual (Fuzzy Greater EQual), FLT posiblemente menor que (Fuzzy Less Than), FLEQ posiblemente menor o igual (Fuzzy Less EQual), MGT Posiblemente mucho mayor (Much Greater Than) y MLT Posiblemente mucho menor (Much Less Than). A modo de ejemplo, en este trabajo sólo se presenta el comparador FLEQ, el resto de los comparadores se encuentran en [5].

Consideremos que, este grado se utiliza para obtener una medida de compatibilidad de $\mu_R(x)$ en el cual éste es posiblemente *más pequeño o igual* en la distribución de posibilidad $\pi_S(x)$ para todo $x \in U$. En el filtrado difuso la consulta FSQL devolverá las tuplas de la base de datos (BD) con un grado 1 cuando $B_R \leq C_S$; con un grado entre 0 y 1 cuando $B_R > C_S$ y $A_R < D_S$; y con un grado 0 para los otros casos (Ver Figura 1).

El grado de posibilidad del comparador difuso FLEQ [3,4] se define de la siguiente forma:

$$\text{CDEG (R FLEQ S)} = \begin{cases} 1 & \text{para } B_R \leq C_S \\ \dfrac{D_R - A_S}{(B_S - A_S) - (C_R - D_R)} & \text{para } B_R > C_S \text{ y } A_R < D_S \\ 0 & \text{para } A_R \geq D_S \end{cases}$$

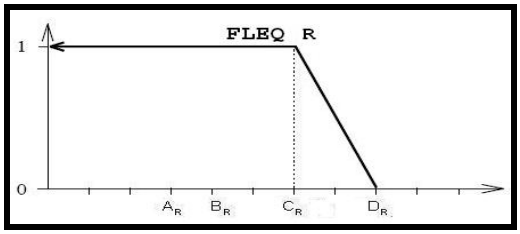

Figura 1: Representación gráfica del comparador FLEQ.

A continuación se presenta el código para SQL Server de la función FLEQ, 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 FLEQ
(@crisp FLOAT,@F1 FLOAT,@F4 FLOAT)
RETURNS FLOAT
AS BEGIN
   DECLARE @valor FLOAT
```

```
   IF (@crisp <= @F1)
      SET @valor=1;
   ELSE IF (@crisp < (@F1+@F4))
         SET @valor=ROUND((@F1 + @F4 -
@crisp)/@F4,4);
      ELSE SET @valor=0;
   RETURN @valor;
END
GO
```

4. ARQUITECTURA PROPUESTA PARA CONSULTAS FLEXIBLES EN MINERÍA DE DATOS

La arquitectura propuesta para el análisis de información (Ver Figura 2), permite satisfacer *indicadores de Gestión de Información* (GI), obtenidos desde la aplicación de un proceso de Minería de Datos. A dicho resultado, se le aplican consultas utilizando comparadores clásicos y difusos como una extensión.

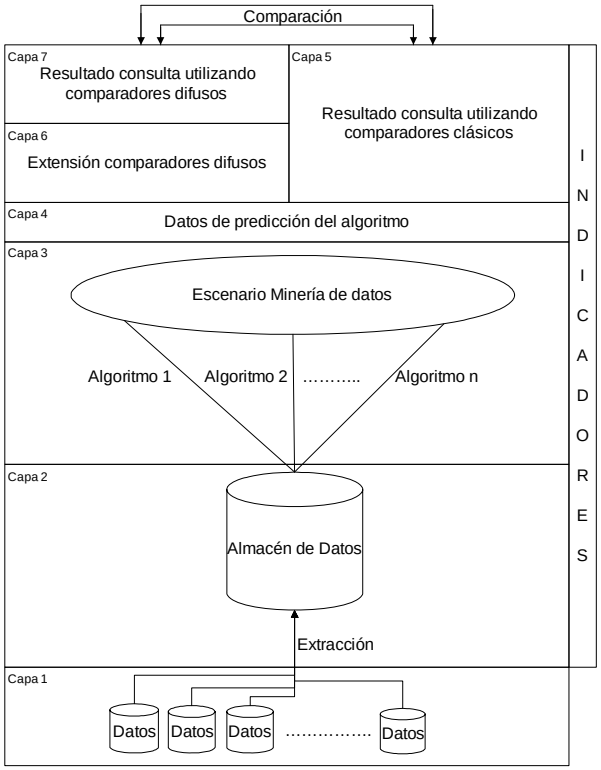

Figura 2: Arquitectura para procesos de minería de datos y consultas con comparadores clásicos y difusos.

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 escenario y aplicar uno a más algoritmos de minería de datos, para posteriormente ser consultados con comparadores clásicos y 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 de la información.

Cada capa que compone esta arquitectura tiene que realizarse en forma secuencial, sólo las consultas pueden ser tantas como el usuario final las requiera. Como en todo proceso, la principal tarea es recopilar la especificación de requerimientos. En este proceso debe quedar claro, cuáles son las necesidades de gestión del usuario final, para poder comenzar a trabajar cada una de las capas propuestas. A continuación se define cada capa.

Capa 1: En esta primera capa se encuentran todos los orígenes de datos, que pueden ser: un archivo plano, un archivo Excel, una base de datos, entre otros.
Capa 2: Tiene como objetivo la extracción de los datos, los cuales son analizados en la especificación de requerimientos, y estos formarán los datos del almacén de datos y deben ser los datos adecuado para el proceso de Minería de Datos.
Capa 3: En esta etapa se utilizan los algoritmos de Minería de Datos, que dependiendo del caso o escenario se puede aplicar el que más complemente la especificación de requerimientos de la Capa 1. Aquí debe estar definida la problemática y los indicadores de gestión a utilizar.
Capa 4: En esta etapa se definen los datos de predicción para cada algoritmo. Estos datos deben ser coherentes con la predicción de los datos e indicadores de gestión definidos en la Capa 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 resultados utilizando comparadores clásicos.
Capa 6: En esta capa se analiza cual de los comparadores de posibilidad, de los presentados en la Tabla 1, 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 o posibilidad seleccionados en la Capa 6. Las consultas ejecutadas, deben ser la mismas del la Capa 5 para ver con mayor grado de detalle el aporte de estos comparadores.

La comparación es la última capa, y tiene como objetivo analizar los resultados de las consultas obtenidos de la capa 5 y 7, para especificar por qué puede ser útil uno u otro comparados y cuál es la diferencia de sus resultados.

5. APLICACIÓN DE LA ARQUITECTURA PROPUESTA (CAPAS 1 al 4)

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

Capa 1 (Datos): Aquí se utilizó la base de datos Adventure Works Cycles, la cual es un complemento 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 utilizando un submodelo y es la entrada para las distintas implementaciones de los 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 serían los mejores algoritmo según sea 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.* Por lo cual se debe analizar cuál de los tres algoritmos realiza esta labor de mejor manera. Para esto existe una herramienta llamada "Grafico de elevación", la cual se puede encontrar en la pestaña "Gráfico de precisión de Minería de Datos". Más información de este punto se encuentra en [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 consultas clásicas y difusas. 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).

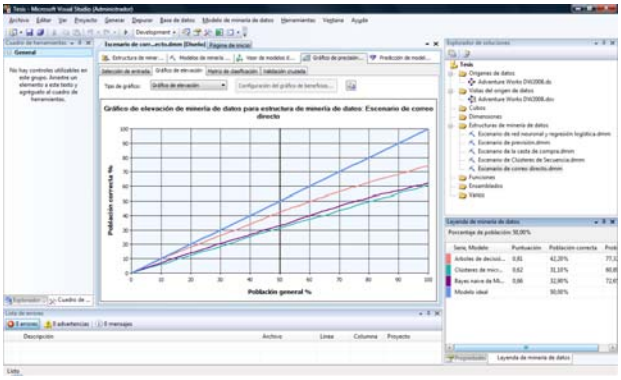

Figura 3**:** Gráfico de elevación de precisión de algoritmos de Minería de Datos.

Cuando se utiliza el algoritmo de árboles de decisión para realizar predicciones, este genera una consulta de predicción sobre una tabla de casos, dicha consulta entrega la probabilidad de que *-cada persona de la tabla de casos compre un producto-*, dicha tabla de casos contiene perfiles de probables clientes, además los resultados de predicción se almacenan en otra tabla, como resultado se obtiene un porcentaje, que se encuentra en el atributo *Expression* de la tabla en la cual se almacenaron los resultados, el cual dirá *que tan probable es que un potencial cliente se convierta en un cliente expresado de otra forma,* es la probabilidad de que cada posible comprador adquiera un producto (en este caso una bicicleta).

Otros atributos de esta tabla de resultados son: atributo *ProspectAlternateKey* que es identificador 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 (CAPAS 5 al 7)

Este apartado tiene por objetivo presentar los resultados obtenidos al utilizar los comparadores clásicos (Capa 5), los comparadores difusos (Capa 7) implementados en la capa 6, y las consultas ejecutadas sobre los resultados de predicción (Capa 4). Cada capa fue efectuada para el caso de *Escenario de Correo Directo*. El análisis de resultados fue obtenido de las predicciones realizadas por el algoritmo de *Árboles de Decisión* sobre una tabla de casos *tabla ProspectuveBuyer*. En la realización de estos pasos se recurrió a SQL Server Management Studio.

Para el análisis de los resultados se utilizó el valor *crisp* 0.51 con el difuminador +/- 0.1 (0.51 – 0.1, 0.51 + 0.1), de donde se obtiene el rango [0.41 a 0.61[, es decir de un 41% a un 61% de probabilidad, sin incluir el extremo derecho (0.61). El valor crisp escogido corresponde a -*los posibles clientes que tienen menor probabilidad de adquirir un producto*-. Este valor crisp escogido se debe a que se quiere enfocar una campaña de motivación a los clientes que tienen menos probabilidad de comprar un producto, cabe señalar que la probabilidad del cliente con menos posibilidades de comprar es de 0.506963566619447, es decir un 51% aproximadamente.

A continuación mostramos el resultado de la consulta con el comparador difuso FLEQ y la consulta clásica con comparador menor o igual con todos los pasos. El resto de los comparadores implementados se encuentran en [5].

a) Comparación usando Expression FLEQ 0.51 y Expression <= 0.51.

Para realizar esta comparación se usaron dos consultas, una consulta utilizando el comparador "<=" y otra consulta usando la función FLEQ (posiblemente menor o igual).

El siguiente código en SQL muestra las sentencias utilizada para la consulta clásica que arrojo 119 registros (la consulta corresponde a la Capa 5) y la Tabla 2 muestra el resultado parcial de la ejecución de la consulta clásica, con los atributos correspondientes de la tabla de resultados explicada en el apartado anterior.

```
select * from
[AdventureWorksDW2008].[dbo].[Resultado
CorreoDirecto]
where Expression <= 0.51
```

Tabla 2: Resultados de la consulta usando Expression <= 0.51.

	ProspectAlternateKey	Bike Buyer	Expression
1	3003	0	0,506963566619447
2	54107006788	0	0,506963566619447
…	**…**	**…**	**…**
119	70093423128	0	0,506963566619447

Luego ejecutamos la consulta difusa (corresponde a la Capa 7), para este tipo de consulta se tuvo que utilizar la condición "> 0" usada al final de la cláusula *where*, ya que la función FLEQ retorna un grado de pertenencia entre 0 y 1, es decir, retorna el grado de pertenencia que posee el valor *crisp* al valor 0.51 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. El resultado parcial 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, explicado en el apartado 5.

```
select * from
[AdventureWorksDW2008].[dbo].[Resultado
CorreoDirecto]
where (select
[AdventureWorksDW2008].[dbo].[FLEQ]
(Expression,0.51,0.1)) > 0
```

Tabla 3: Resultados de la consulta usando Expression FLEQ 0.51.

	ProspectAlternateKey	Bike Buyer	Expression
1	3003	0	0,506963566619447
2	54107006788	0	0,506963566619447
…	**…**	**…**	**…**
670	37111264500	1	0,550329734038698

El análisis de resultados usando, *Expression FLEQ 0.51 y Expression <= 0.51,* considera que una consulta clásica entrego 119 tuplas de resultados y la consulta difusa entrego 670 tuplas. En la Figura 4 se muestra la gráfica que apoya los resultados obtenidos con la consulta clásica, de color plomo se puede apreciar todo el conjunto de resultados incluidos por el comparador clásico. En la Figura 5 se muestra la gráfica que apoya los resultados obtenidos por la consulta difusa, de color plomo se puede observar todo el conjunto de resultados incluidos por el comparador difuso, aquí se debe recordar que el extremo 0.61 no es incluido.

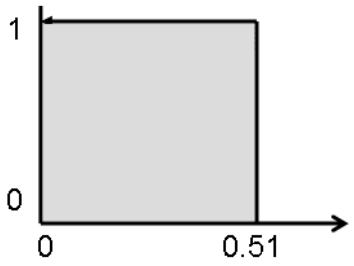

Figura 4: Gráfico de valores para el comparador clásico <=.

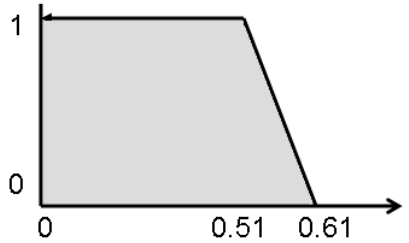

Figura 5: Gráfico de valores para el comparador difuso FLEQ.

El uso de los comparadores EQ y FEQ se encuentran en [7].

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.

Los comparadores difusos entregan más tuplas, en algunos casos, para el análisis de datos obtenidos por un algoritmo de Minería de Datos, lo que puede apoyar mejor la toma de 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.

Actualmente estamos trabajando en incorporar etiquetas lingüísticas a la consultas de lo que dará otro espectro de análisis a las consultas de algoritmos de Minería de Datos. También estamos trabajando en dar grados de importancia a los datos de cada cluster de Minería de Datos, o grados de pertenencia de los datos al cluster.

Trabajos Futuros: Se propone extender otro tipo de componentes de la lógica difusa a los datos o algoritmos de Minería de Datos. También analizar diferentes tipos de difuminación de las expresiones de las consultas.

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.

[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 en microsoft SQL Server 2008". 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.

[7]. Urrutia A, Mendez J., Gutierrez C. (2009). "Algoritmos de minería de datos extendidos con comparadores difusos y su impacto en los indicadores de gestión". Encuentro de Informática de Gestión Ediciones Universitarias de la Fontera. Universidad de la Frontera Chile, Diciembre de 2009. En edición.

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