APLICACIÓN DE TÉCNICAS DE MINERÍA DE DATOS EN LA CONSTRUCCIÓN Y VALIDACIÓN DE MODELOS PREDICTIVOS Y ASOCIATIVOS A PARTIR DE ESPECIFICACIONES DE REQUISITOS DE SOFTWARE

María N. Moreno García*, Luis A. Miguel Quintales, Francisco J. García Peñalvo y M. José Polo Martín

Universidad de Salamanca. Departamento de Informática y Automática
Teléfono: 34-923-294653, Fax: 34-923-294514, *e-mail: mmg@usal.es

Resumen: La medición del software está adquiriendo una gran importancia debido a que cada vez se hace más patente la necesidad de obtener datos objetivos que permitan evaluar, predecir y mejorar la calidad del software así como el tiempo y coste de desarrollo del mismo. El valor de las mediciones aumenta cuando se realiza sobre modelos construidos en las primeras fases del proyecto ya que los resultados obtenidos permiten tenerlo bajo control en todo momento y corregir a tiempo posibles desviaciones. La proliferación actual de métricas y el gran volumen de datos que se maneja ha puesto de manifiesto que las técnicas clásicas de análisis de datos son insuficientes para lograr los objetivos perseguidos. En este trabajo se presenta la forma en que pueden aplicarse las nuevas técnicas de minería de datos en la construcción y validación de modelos de ingeniería del software, cambiando el análisis tradicional de datos *dirigido a la verificación* por un enfoque de análisis de datos *dirigido al descubrimiento del conocimiento*.

1. Introducción

Las primeras etapas del desarrollo de software son cruciales en la consecución de productos de calidad dentro de los límites de tiempo y coste establecidos para un proyecto. Los errores introducidos en las primeras etapas del desarrollo del software o durante su evolución son causa frecuente de dificultades en el mantenimiento, baja reutilización y comportamiento defectuoso de los programas. Igualmente, las malas estimaciones realizadas al comienzo del proyecto tienen consecuencias desastrosas en cuanto a costes y plazos de entrega. Estas son las principales causas por las que la medición del software en el ámbito de la especificación de requisitos (ERS) está adquiriendo cada vez mayor importancia, debido a la necesidad de obtener datos objetivos que contribuyan a mejorar la calidad desde las primeras etapas del proyecto.

Los estudios relacionados con la medición en el nivel de la especificación de requisitos se han centrado fundamentalmente en el desarrollo de métricas para determinar el tamaño y la funcionalidad del software. Entre las de mayor difusión se encuentran las métricas de puntos de función [Albrecht, 1979], métricas Bang [DeMarco, 1982] o los puntos objeto [Boehm et al., 1995]. La medición de atributos de calidad de las especificaciones de requisitos del software (ERS) ha sido también objeto de algunos trabajos que van desde la medición de especificaciones formales [Samson et al., 1990] hasta la aplicación de métricas para evaluar la calidad de especificaciones expresadas informalmente en lenguaje natural (métricas de facilidad de comprensión del texto contenido en los documentos [Lehner, 1993] o métricas de estructura y organización en documentos convencionales [Arthur y Stevens, 1989] y con hipertexto [French et al., 1997] [Roth et al., 1994]), pasando por técnicas encaminadas a

determinar el cumplimiento de los estándares, directrices, especificaciones y procedimientos, que requieren información procedente de revisiones técnicas, inspecciones**,** *Walkthrough*, o auditorías [Davis et al., 1993] [Brykczynski, 1999] [Farbey, 1990]. La creciente adopción de la tecnología de orientación a objetos en el desarrollo de software ha dado lugar a la aparición de nuevas métricas específicas para este tipo de sistemas [Chidamber y Kemerer, 1994], [Lorenz y Kidd 1994], [Churcher y Shepperd, 1995] [Binder, 1994]. Recientemente se han propuesto métricas para la evaluación de la calidad a partir de modelos producidos en etapas iniciales del ciclo de vida, como son las métricas de calidad y complejidad en modelos OMT [Genero et al., 1999], métricas de calidad de los diagramas de clases en UML [Genero et al., 1999, 2000] o las técnicas de medición de modelos conceptuales basados en eventos [Poels, 2000].
La proliferación actual de métricas y la necesidad de medir diferentes aspectos del software está contribuyendo a crear confusión sobre las relaciones entre tales medidas, así como sobre su forma y ámbito de aplicación. Este hecho ha abierto una nueva vía en la investigación orientada hacia la propuesta de modelos, arquitecturas y marcos de referencia ("framewors") que permitan la organización de las medidas y la clasificación de las entidades de software susceptibles de medir [Poels, 1998] [Briand et al., 1999] [Moreno et al. 2001]. La construcción de modelos sobre diferentes aspectos del software requiere la recolección de numerosos datos procedentes de observaciones empíricas. Los avances tecnológicos actuales posibilitan la rápida obtención de grandes cantidades de datos de fuentes muy diversas, así como el almacenamiento eficiente de los mismos. Dichos datos encierran información muy valiosa que puede tratarse mediante los métodos tradicionales de análisis de datos, sin embargo estos métodos no son capaces de encontrar toda la información útil latente en la gran masa de datos que se maneja. En ese contexto, las técnicas de minería de datos surgen como las mejores herramientas para realizar exploraciones más profundas y extraer información nueva, útil y no trivial que se encuentra oculta en grandes volúmenes de datos.
En este trabajo se muestra la aplicación práctica de técnicas de minería de datos en la construcción y validación de modelos de ingeniería del software que relacionan diferentes atributos de la ERS con el objeto de predecir características del software producido tales como calidad, funcionalidad, tamaño, complejidad, etc. Los resultados obtenidos serán indicativos tanto de la calidad como de la utilidad de los modelos de requisitos realizados.

2. Técnicas de minería de datos

La minería de datos ha dado lugar a una paulatina sustitución del análisis de datos *dirigido a la verificación* por un enfoque de análisis de datos *dirigido al descubrimiento del conocimiento*. La principal diferencia entre ambos se encuentra en que en el último se descubre información sin necesidad de formular previamente una hipótesis. La aplicación automatizada de algoritmos de minería de datos permite detectar fácilmente patrones en los datos, razón por la cual esta técnica es mucho más eficiente que el análisis dirigido a la verificación cuando se intenta explorar datos procedentes de repositorios de gran tamaño y complejidad elevada. Dichas técnicas emergentes se encuentran en continua evolución como resultado de la colaboración entre campos de investigación tales como bases de datos, reconocimiento de patrones, inteligencia artificial, sistemas expertos, estadística, visualización, recuperación de información, y computación de altas prestaciones.

Los algoritmos de minería de datos se clasifican en dos grandes categorías: supervisados o predictivos y no supervisados o de descubrimiento del conocimiento [Weiss y Indurkhya, 1998].

Los algoritmos **supervisados o predictivos** predicen el valor de un atributo (*etiqueta*) de un conjunto de datos, conocidos otros atributos (*atributos descriptivos*). A partir de datos cuya etiqueta se conoce se induce una relación entre dicha etiqueta y otra serie de atributos. Esas relaciones sirven para realizar la predicción en datos cuya etiqueta es desconocida. Esta forma de trabajar se conoce como *aprendizaje supervisado* y se desarrolla en dos fases: Entrenamiento (construcción de un modelo usando un subconjunto de datos con etiqueta conocida) y prueba (prueba del modelo sobre el resto de los datos).
Cuando una aplicación no es lo suficientemente madura no tiene el potencial necesario para una solución predictiva, en ese caso hay que recurrir a los métodos **no supervisados o de descubrimiento del conocimiento** que descubren patrones y tendencias en los datos actuales (no utilizan datos históricos). El descubrimiento de esa información sirve para llevar a cabo acciones y obtener un beneficio (científico o de negocio) de ellas. En la tabla siguiente se muestran algunas de las técnicas de minería de ambas categorías.

Supervisados	**No supervisados**
Árboles de decisión	Detección de desviaciones
Inducción neuronal	Segmentación
Regresión	Agrupamiento ("clustering")
Series temporales	Reglas de asociación
	Patrones secuenciales

Tabla 1. Clasificación de las técnicas de minería de datos

La aplicación de los algoritmos de minería de datos requiere la realización de una serie de actividades previas encaminadas a preparar los datos de entrada debido a que, en muchas ocasiones dichos datos proceden de fuentes heterogéneas, no tienen el formato adecuado o contienen ruido. Por otra parte, es necesario interpretar y evaluar los resultados obtenidos. El proceso completo consta de las siguientes etapas [Cabena et al., 1998]:

1. **Determinación de objetivos**
2. **Preparación de datos**
   - Selección: Identificación de las fuentes de información externas e internas y selección del subconjunto de datos necesario.
   - Preprocesamiento: estudio de la calidad de los datos y determinación de las operaciones de minería que se pueden realizar.
3. **Transformación de datos**: conversión de datos en un modelo analítico.
4. **Minería de datos**: tratamiento automatizado de los datos seleccionados con una combinación apropiada de algoritmos.
5. **Análisis de resultados**: interpretación de los resultados obtenidos en la etapa anterior, generalmente con la ayuda de una técnica de visualización.
6. **Asimilación de conocimiento**: aplicación del conocimiento descubierto.

Aunque los pasos anteriores se realizan en el orden en que aparecen, el proceso es altamente iterativo, estableciéndose retroalimentación entre los mismos. Además, no todos los pasos requieren el mismo esfuerzo, generalmente la etapa de preprocesamiento es la más costosa ya que representa aproximadamente el 60 % del esfuerzo total, mientras que la etapa de minería sólo representa el 10%.

3. Aplicaciones de la minería de datos en la medición del software

Las técnicas de minería de datos se están utilizando desde hace varios años para la obtención de patrones en los datos y para la extracción de información valiosa en el campo de la Ingeniería del Software. Entre estas aplicaciones podemos citar la utilización de árboles de decisión en la construcción de modelos de clasificación de diferentes características del desarrollo de software [Khoshgoftaar y Allen, 1999] [Porter y Selby, 1990] [Tian y Palma, 1998], la aplicación de técnicas de "*clustering*" en la planificación del mantenimiento [Krohn y Boldyreff, 1999] y en la estimación de la fiabilidad del software [Podgurski et al., 1999] o el uso de redes neuronales en la predicción de riesgos de mantenimiento en módulos de programa [Khoshgoftaar y Lanning, 1995]. La mayor parte de los trabajos realizados están dirigidos a la obtención de modelos de estimación de esfuerzo de desarrollo [Srinivasan y Fisher, 1995] y modelos de predicción de diferentes aspectos de la calidad del software [Khoshgoftaar et al., 1997]. En ambos casos, las métricas tanto de productos como de procesos juegan un papel importante, constituyendo la base para la construcción de los modelos y posterior validación de los mismos. En publicaciones recientes aparece la introducción de algoritmos de minería en la realización de validaciones de modelos obtenidos mediante otras técnicas. En estos trabajos se comprueba la validez de modelos de estimación mediante métodos de regresión, redes neuronales, algoritmos genéticos, etc. [Dolado, 2000], se validan métricas, e incluso "*frameworks*" de medición [Mendonça y Basili, 2000].

4. Ejemplos de aplicación

En este apartado se muestran algunos ejemplos de aplicación de las técnicas mencionadas en la construcción de modelos de estimación del tamaño de un proyecto. Se ha utilizado la herramienta MineSet de Silicon Graphics sobre un conjunto de datos procedentes de experimentos llevados a cabo por Dolado y descritos en [Dolado, 2000].

4.1. Estudio previo de los datos utilizados

Se dispone de datos referentes a 42 proyectos implementados con un lenguaje de cuarta generación (Informix-4GL). Los sistemas desarrollados son aplicaciones de contabilidad con las características de sistemas comerciales, cada uno incluye alguno de los siguientes subsistemas. Compras, ventas, inventario, finanzas y ciclos de producción. Esta información se ha dividido en dos grupos, en el primero se ha realizado la descomposición del proyecto en módulos o componentes de tres tipos diferentes siguiendo las pautas de Verner y Tate [Verner y Tate, 1992] y se han obtenido atributos de cada uno de los módulos. El segundo grupo contiene datos globales de cada uno de los proyectos. En el primer caso contamos con 1537 registros con datos referentes a los módulos, mientras que en el segundo caso se trabaja con 42 registros con información correspondiente a los proyectos estudiados.

Descripción de los atributos de los módulos:

TYPECOMP: Tipo de componente (1: menú, 2: entrada, 3: informe/consulta)
OPTMENU: Número de opciones (sólo para componentes de tipo 1)
DATAELEMEN: Número de elementos de datos (sólo para componentes de tipo 2 y 3)
RELATION: Número de relaciones (sólo para componentes de tipo 2 y 3)
LOC: Número de líneas de código del módulo

Con los datos de este fichero se ha realizado un estudio estadístico de cada uno de los atributos de los módulos que componen el proyecto. En las figuras 1, 2 y 3 aparece la distribución de valores de atributos para cada uno de los tipos de componente.

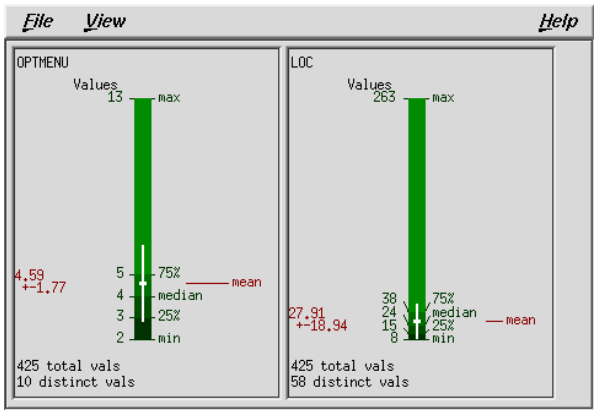

Figura 1. Estadísticas para componentes de tipo 1 (menú)

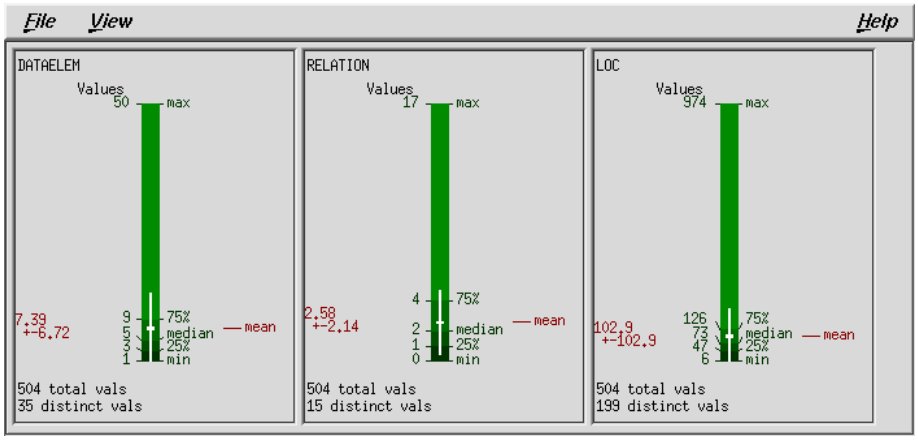

Figura 2. Estadísticas para componentes de tipo 2 (entradas)

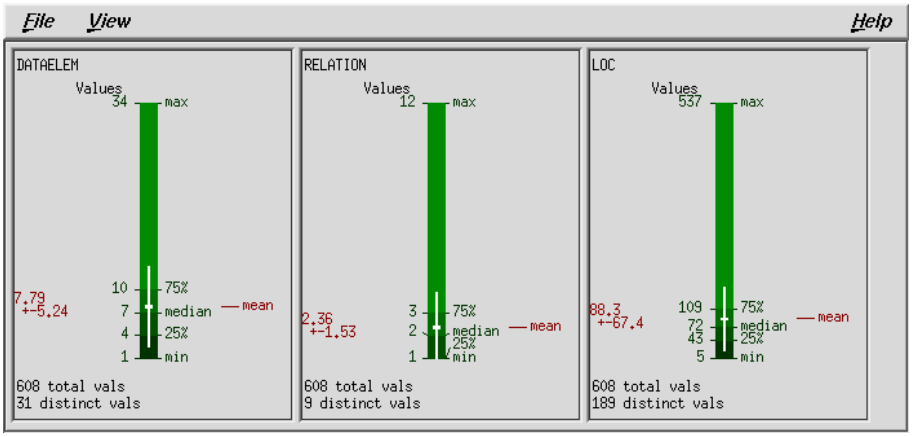

Figura 3. Estadísticas para componentes de tipo 3 (informes/consultas)

Para obtener las relaciones existentes entre los distintos atributos se han representado sus correspondientes valores mediante gráficos de dispersión (figuras 4 y 5). En ellos se puede observar que los valores de los atributos número de elementos de datos y número de relaciones tienen una clara influencia en el número de líneas de código.

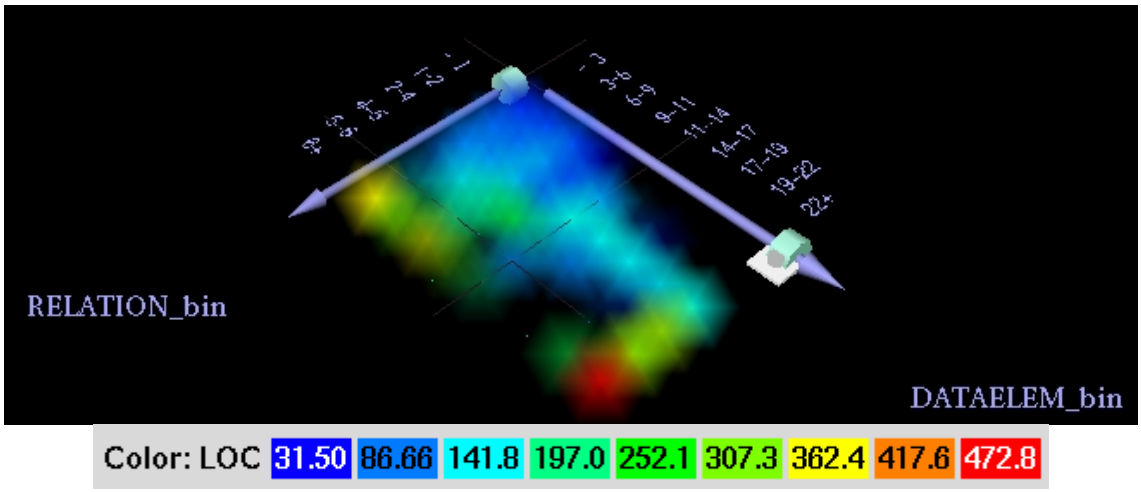

Figura 4. Gráfico de dispersión para componentes de tipo 2 (entradas)

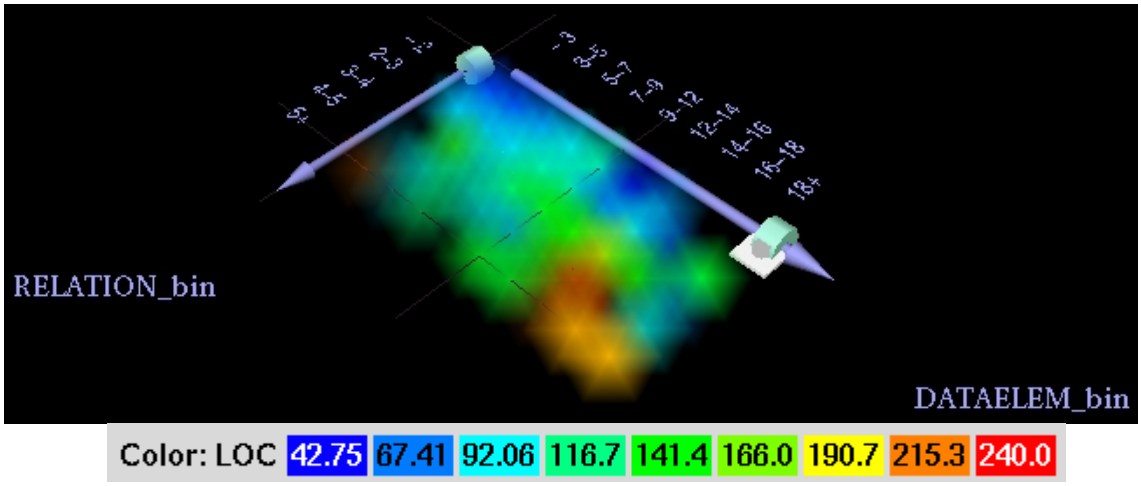

Figura 5. Gráfico de dispersión para componentes de tipo 3 (informes/consultas)

Descripción de los atributos de proyectos:

Los atributos que se describen a continuación son los utilizados en el método Mark II [Symons, 1991] para el cálculo de los puntos de función. El método considera el sistema compuesto de transacciones lógicas. Una transacción lógica es una única combinación entrada/proceso/salida activada por un único evento que tiene significado para el usuario o es el resultado de una consulta o extracción de información. El concepto de entidad sustituye al de fichero lógico.

LOC: Número de líneas de código
NOC: Número de componentes

NTRNSMKII: Número de transacciones MKII
INPTMKII: Número total de entradas
ENTMKII: Número de entidades referenciadas
OUTMKII: Número total de salidas (elementos de datos sobre todas las transacciones)
UFPMKII: Número de puntos de función no ajustados MKII

Para realizar la contabilización de entidades, éstas se dividen en primarias (aquellas para las que se construye el sistema con la intención de recolectar y almacenar datos) y no primarias (las utilizadas como referencia, validación, etc.). Todas las entidades no primarias se engloban en una única denominada entidad del sistema. Se contabiliza el numero de entidades a las que se hace referencia en cada transacción, no el número de referencias.

En cuanto al número de entradas y salidas, se cuentan los tipos de elementos de datos, no ocurrencias (aplicable también a las tablas). No se cuentan elementos de datos de cabeceras o pies de pantalla no generados específicamente para esa pantalla, tampoco se contabilizan etiquetas, cajas, etc. Los mensajes de error se consideran ocurrencias del tipo de dato "mensaje de error" (análogo para los mensajes de operador).

El estudio estadístico realizado sobre los atributos anteriores se refleja en la figura 6.

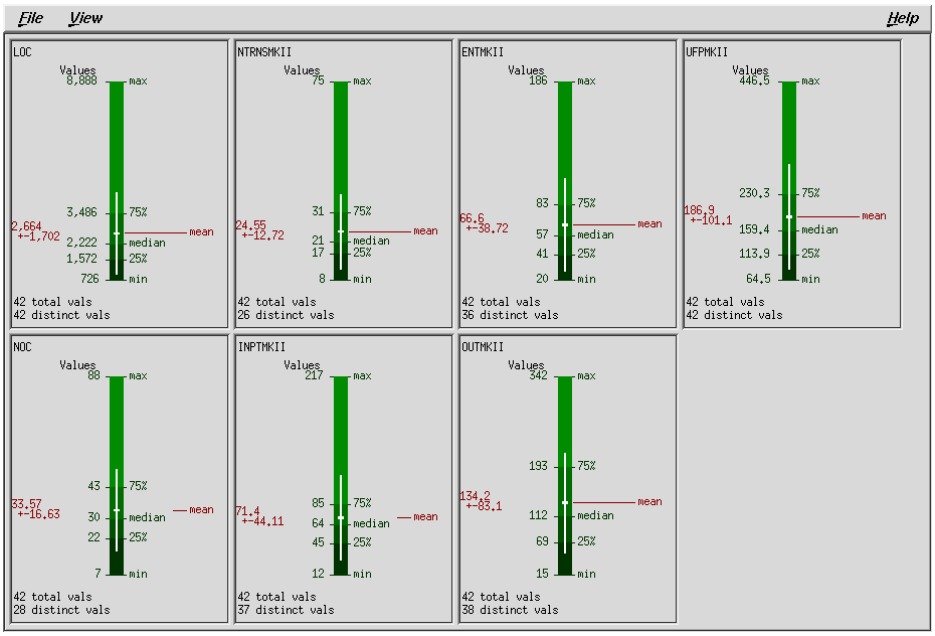

Figura 6. Estadísticas para atributos de proyectos

4.2. Aplicación de técnicas de minería de datos

Los datos anteriores se han tratado con diferentes algoritmos de minería de datos. En todos los casos se han realizado validaciones cruzadas para crear los conjuntos de prueba y entrenamiento, dividiendo de forma iterativa los datos en conjuntos mutuamente excluyentes.

Una de las técnicas de minería de datos más intuitivas es la de los **árboles de decisión**, que consiste en la creación de un modelo de clasificación a partir de un conjunto de entrenamiento y de un ***inductor***. Los registros del conjunto de entrenamiento tienen que pertenecer a un pequeño grupo de clases predefinidas, cada clase corresponde a un valor de la etiqueta. El modelo inducido (***clasificador***) consiste en una serie de patrones que son útiles para distinguir las clases. Una vez que se ha inducido el modelo se puede utilizar para predecir

automáticamente la clase de otros registros no clasificados (de etiqueta desconocida). Los algoritmos utilizados para construir el modelo de clasificación del árbol intentan encontrar valores de los atributos que proporcionen la máxima separación en los datos del conjunto de entrenamiento.
En la figura 7 se muestra un árbol de decisión que clasifica el tamaño del proyecto (LOC) en cinco intervalos, en función de los demás atributos disponibles. El atributo que más influye en la clasificación es el de mayor pureza (1 - entropía) y aparece en el nodo raíz del árbol. En el ejemplo se puede observar que el número de transacciones es la variable que produce la mayor diferenciación en la clasificación.

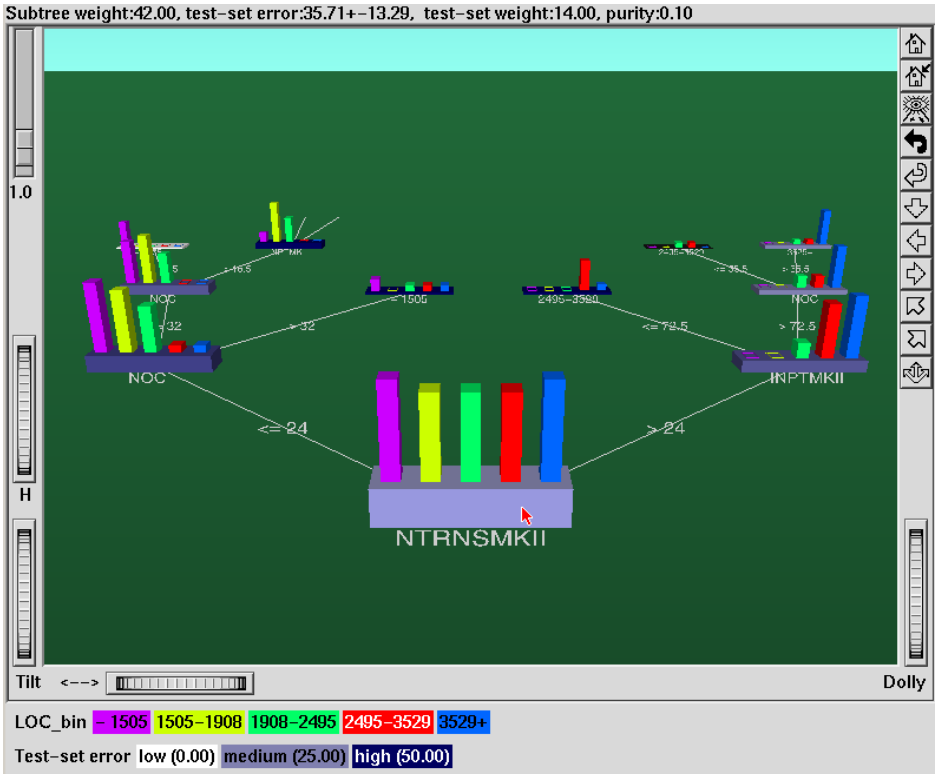

Figura 7. Árbol de decisión

Para realizar la validación del modelo inducido con los datos del conjunto de entrenamiento, aplicamos la técnica de las **matrices de confusión** que muestran el tipo de las predicciones correctas e incorrectas cuando se aplica el modelo sobre el conjunto de prueba (Figura 8). Las predicciones correctas están representadas por las barras que aparecen sobre la diagonal, mientras que el resto de las barras indican el tipo de error cometido (qué valor ha predicho el modelo y cual es el valor verdadero). La altura de las barras es proporcional al peso de los registros que representan. Hay que señalar que en caso del ejemplo que se está tratando los resultados no son muy buenos debido a que se dispone de muy pocos datos, el conjunto de entrenamiento lo componen 28 registros y el de prueba 14.

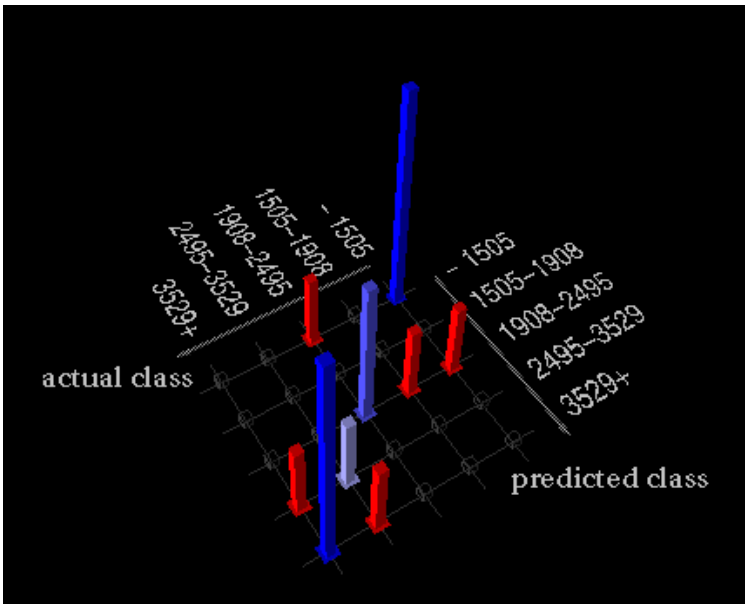

Figura 8. Matriz de confusión

Las **tablas de decisión** son otra herramienta de clasificación mediante la cual se muestran correlaciones entre pares de atributos a diferentes niveles. Los atributos continuos se separan en intervalos discretos y se induce automáticamente la tabla mediante cálculos de probabilidades a partir de los registros del conjunto de entrenamiento. En la tabla de decisión del ejemplo propuesto (figura 9) aparecen dos niveles, el primero con los atributos NOC y NTRSNSMKII cuyas celdas se descomponen en un segundo nivel en función de los valores de los atributos INTPTMKII y UFPMKII. En cada celda se muestra, mediante colores diferentes, la proporción de registros que tiene de cada una de las clases. Un buen clasificador debería conseguir el mayor número posible de celdas de un solo color que contendrían únicamente registros de una clase. En nuestro caso sólo se ha conseguido una celda de este tipo, lo que indica que se necesitan mejores atributos o mayor número de atributos para conseguir una mejor clasificación.

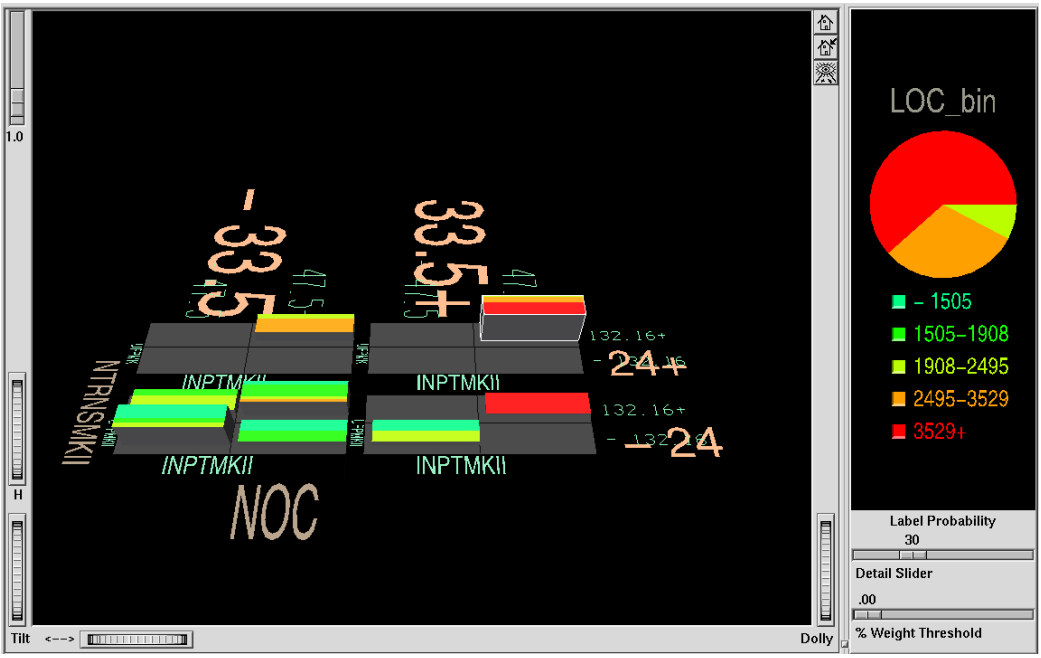

Figura 9. Tabla de decisión

Otro método útil para conocer la importancia de los valores de un atributo específico para la clasificación o la probabilidad de que un registro pertenezca a una clase cuando se conocen pocos atributos de dicho registro es el **clasificador de evidencias** que muestra la distribución de registros por valores de los atributos (figura 10).

Figura 10. Clasificador de evidencias

Los ejemplos anteriores se encuadran en la categoría de técnicas predictivas o supervisadas. Otro grupo de técnicas son las no supervisadas o de descubrimiento del conocimiento, entre las que se encuentra el análisis de asociación, que persigue el establecimiento de relaciones entre registros individuales o grupos de registros de la bases de datos. Dos especializaciones del análisis de asociación son reglas de asociación y el descubrimiento de patrones secuenciales. Una regla de asociación tiene la forma "**Si X entonces Y**", donde:

- ***Cuerpo de la regla***: X (LHS)
- ***Cabeza de la regla***: Y (RHS)
- ***Factor de confianza o previsibilidad***: grado en que la regla es verdad en relación con los registros individuales.
- ***Factor de soporte o prevalencia***: relación entre el número de veces que tiene lugar la regla y el número total de transacciones.
- ***Previsibilidad esperada***: número de veces que ocurre Y.

En la figura 11 se muestra la visualización de los conceptos anteriores relativos a las reglas de asociación que relacionan los atributos de los proyectos que se están estudiando .

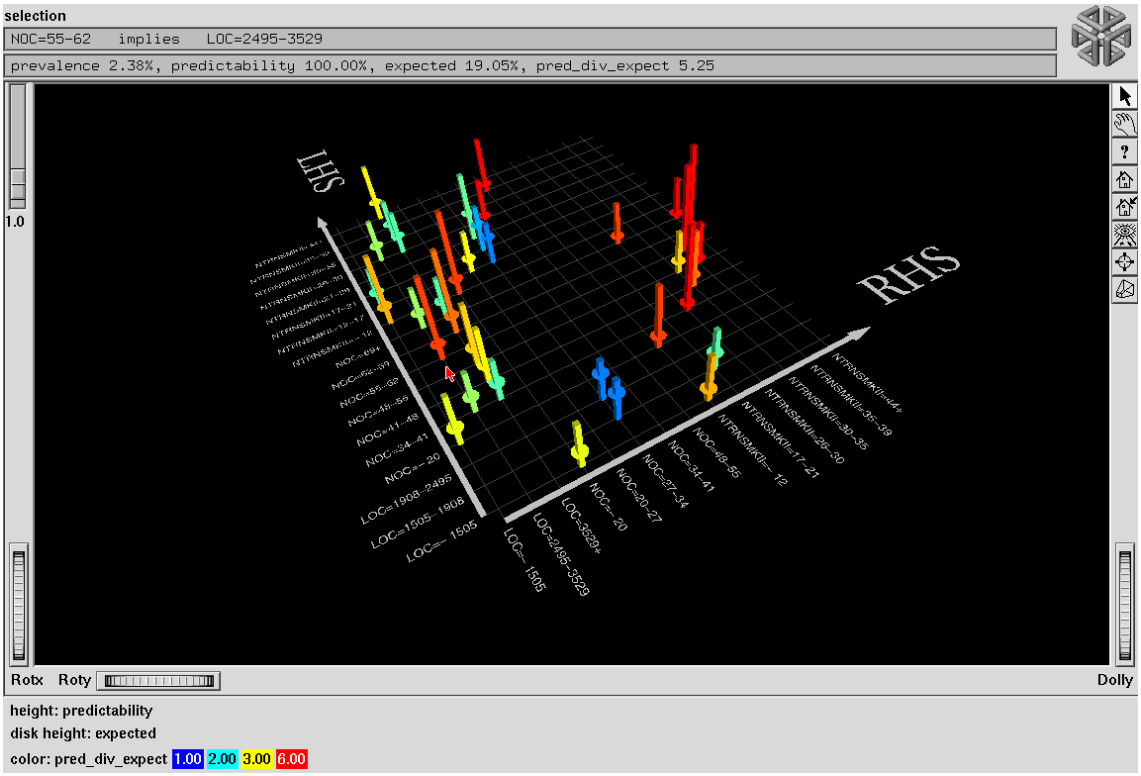

Figura 11. Visualización de reglas de asociación

La visualización es una de las técnicas más potentes para identificar patrones ocultos en los datos. Con estas técnicas se pueden detectar fenómenos que ocurren en los datos mediante representaciones n-dimensionales de datos sobre pantallas bidimensionales. En la figura 12 podemos observar un gráfico de dispersión en el que sobre los ejes del gráfico aparecen representados los valores de los dos atributos más influyentes del ejemplo, utilizando el color para representar el tamaño del proyecto. En este ejemplo se puede distinguir las combinaciones de valores de atributos que dan lugar a diferentes tamaños de proyectos

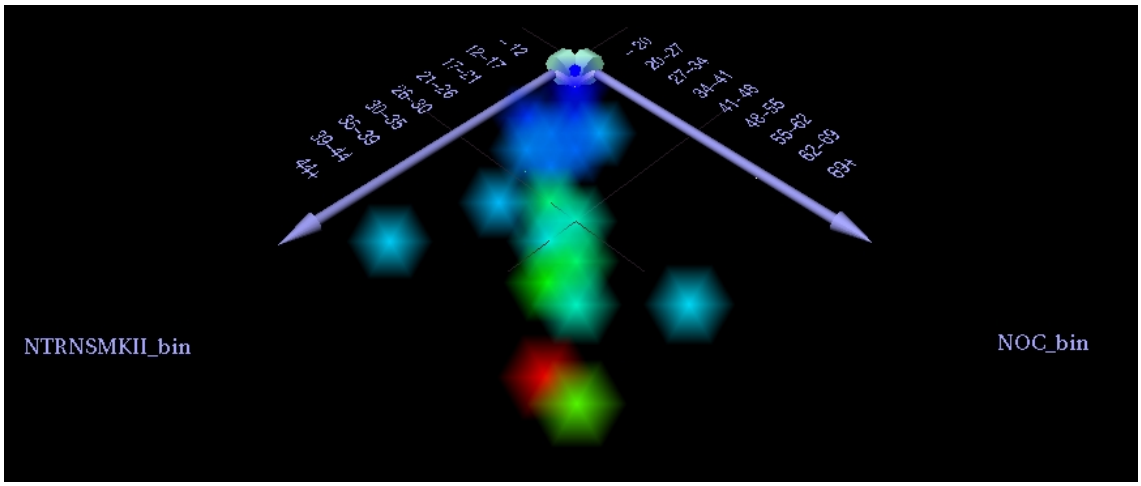

Figura 12. Gráfico de dispersión

5. Conclusiones

En el apartado anterior se han ofrecido únicamente unos pocos ejemplos del uso de técnicas de minería de datos en la construcción de modelos predictivos y asociativos con datos procedentes de mediciones realizadas en el ámbito de la especificación de requisitos, no obstante las posibilidades que ofrece este nuevo enfoque de tratamiento de datos son mucho mayores, ya que el número de técnicas que engloba es mucho más amplio. Por otra parte, los métodos de minería de datos llevan asociados una serie de mecanismos (estimación de errores, matrices de confusión, matrices de pérdida, curvas de esfuerzo y aprendizaje, análisis sensitivo de entradas...) que permiten realizar una mejor validación empírica de los modelos y un análisis de resultados más completo y fiable que el que ofrece el enfoque clásico.

6. Bibliografía

[Albrech, 1979] Albrecht, A.J., "*Measuring application development*", Proc. of IBM Applications DevelopmentJoint SHARE/GUIDE Symposium, Monterey, CA, pp 83-92, 1979.

[Arthur y Stevens, 1989] Arthur, J.D. y Stevens, K.T. , "*Assessing the adequacy of documentation through document quality indicators*", Proceedings of the IEEE Conference of Software Maintenance, pp. 40-49, 1989.

[Binder, 1994] Binder, R., "*Testing Object-Oriented Systems*", American Programmer, 7(4), 22-29, 1994.

[Boehm et al., 1978] Boehm, B.W., Kaspar, J.R. y otros "*Characteristics of Software Quality*", TRW Series of Software Technology, 1978.

[Briand et al., 1999] Briand, L.C., Daly, J.W. y Wüst, J.K. "*A unified framework for coupling measurement in object-oriented system*", IEEE Transaction on Software Engineering, 25 (1), 1999.

[Brykczynskki, 1999] Brykczynskki, B., "*A survey of software inspection checklist*", ACM Software Engineering Notes, 24(1), pp 82-89, 1999.

[Cabena et al., 1998] Cabena, P., Hadjinian, P., Stadler, R., Verhees, J. Y Zanasi, A. "*Discovering Data Mining. From Concept to Implementation*", Prentice Hall, 1998.

[Chidamber y Kemerer, 1994] Chidamber, S.R. y Kemerer, C.F., "*A Metrics Suite for Object-Oriented Design*" , IEEE Transactions of Software Engineering, 20(6), 476-493, 1994.

[Churcher y Shepperd, 1995] Churcher, N.I. and Shepperd, M.J., "*Towards Conceptual Framework for Object-Oriented Metrics*", ACM Software Engineering Notes, 20 (2), 67-76, 1995.

[Davis, A. et al., 1993] Davis, A. et al., "*Identifying and Measuring Quality in a Software Requirements Specification*" Proceedings of the First International Software Metrics Symposium, Baltimore, May 21-22, pp. 141-152, 1993.

[DeMarco, 1982] DeMarco, T., "*Controlling Software Projects*", Yourdon Press, 1982.

[Dolado, 2000] Dolado, J.J. "A *Validation of the Component-Based Method for Software Size Estimation*"; IEEE Transactions on Software Engineering 26(10), pp. 1006-1021, 2000.

[Farbey, 1990] Farbey, B., "Software Quality metrics: considerations about requirements and requirements specification", Information and Software Technology, 32 (1), pp 60-64, 1990.

[French et al., 1997] French, J.C., Knight, J.C. y Powell, A.L., "*Applying hipertext structures to software documentation", Information Processing and Management*", 33 (2), pp 219-231, 1997.

[Genero et al., 1999] Genero, M., Manso, M.E., Piattini, M. y García F.J. "*Assessing the quality and the Complexity of OMT Models*" 2nd European Software Measurements Conference-FESMA 99, Amsterdam, Netherlands, pp 99-109, 1999.

[Genero et al., 2000] Genero, M., Piattini, M. y Calero, C. "*Una propuesta para medir la calidad de los diagramas de clases en UML*", IDEAS´2000, Cancun, México, pp 373-384, 2000.

[Khoshgoftaar et al., 1997] Khoshgoftaar, T.M., Allen, E.B., Hudepohl, J.P. y Aud, S.J. "*Neural Networks for Software Quality Modeling of a Very Large Telecommunications System*". IEEE Trans.On Neural Networks, (8)4, pp. 902-909, 1997.

[Khoshgoftaar y Allen, 1999] Khoshgoftaar, T.M. y Allen, E.B. "*Modeling Software Quality with Classification Trees*".In *Recent Advances in Reliability and Quality Engineering*, Hoang Pham Editor.World Scientific, Singapore,1999.

[Khoshgoftaar y Lanning, 1995] Khoshgoftaar, T.M.y Lanning, D.L. "*A Neural Network Approach for Early Detection of ProgramModules Having High Risk in the Maintenance Phase*" .J.Systems Software ,29(1), pp. 85-91,1995.

[Krohn y Boldyreff, 1999] Krohn, U. y Boldyreff, C. "*Application of Cluster Algorithms for Batching of Proposed Software Changes*". J.Softw.Maint:Res.Pract. 11, pp. 151-165. 1999.

[Lehner, 1993] Lehner, F., "*Quality control in software documentation: Measurement of text comprehensibility*", Information and Management, 25, pp 133-146, 1993.

[Lorenz y Kidd, 1994] Lorenz, M. and Kidd, J., "Object_oriented Software Metrics", Prentice Hall 1994.

[Mendonça y Basili, 2000] Mendonça, M.G. y Basili, V.R. "*Validation of an Approach for Improving Existing Measurement Frameworks*"; IEEE Transactions on Software Engineering 26(6), pp. 484-499, 2000.

[Mendonça y Sunderhaft, 1999] Mendonça, M.G. y Sunderhaft, N.L. "*Mining Software Engineering Data: A Survey*", Technical Report, DoD Data and Analysis Center for Software, DACS-SOAR-99-3, 1999.

[Moreno et al., 2001] Moreno, M.N., García, F.J., Polo, M.J., López, V. y González, A. "*Marco de Referencia para la Gestión de la Calidad de las Especificaciones de Requisitos*", QUATIC'2001, Lisboa, Portugal, 2001.

[Podgurski et al., 1999] Podgurski, A., Masri, W., McCleese Y. y Wolff, F.G. "*Estimation of Software Reliability by Stratified Sampling*" .ACM Trans.on Soft.Eng.and Methodology, (8)3, pp.263-283, 1999.

[Poels, 1998] Poels, G., "*Towards a size measurement framework for object-oriented especifications*". H. Coombes, M. Hooft van Huysduynen, and B. Peeters (eds.), Proceedings of the 1st European Software Measurement Conference (FESMA'98), Antwerp, pp. 379-388, 1998.

[Poels, 2000] Poels, G., "*On the measurements of event-based object-oriented conceptual models*". 4th International ECOOP Workshop on Quantitative Approaches in Object Oriented Software Engineering, Cannes, France, 2000.

[Porter y Selby, 1990] Porter, A.A. y Selby, R.W. "*Empirically Guided Software Development Using Metric-Based Classification Trees*". IEEE Software ,7(2), pp.46-54, 1990.

[Roth et al. 1994] Roth, T., Aiken, P. y Hobbs, S., "*Hypermedia support for software developmemt: a retrospective assessment*", Hypermedia, 6 (3), pp 149-173, 1994.

[Samson et al. 1990] Samson, W.B., Nevill, D.G. Y Dugard, P.I., "Predictive software metrics based on a formal specification", Software Engineering Journal, 5(1), 1990.

[Srinivasan y Fisher, 1995] Srinivasan, K. y Fisher, D. "*Machine Learning Approaches to Estimating Software Development Effort*". IEEE Trans.on Soft.Eng., 21(2), pp.126-137, 1995.

[Symons, 1991] Symons, C.R. "*Software Sizing and Estimating MKII FPA*". John Wiley and Sons, 1991.

[Tian y Palma, 1998] Tian, J. y Palma J. "*Analyzing and Improving Reliability:A Tree-based Approach*" .IEEE Software , pp. 97-104,15(2), 1998.

[Verner yTate, 1992] Verner, J. and Tate, G., "A Software Size Model", IEEE Transaction of Software Engineering, 18 (4), pp. 265-278, 1992.

[Weiss y Indurkhya, 1998] Weiss, S.M. y Indurkhya, N. "*Predictive Data Mining. A Practical Guide*", Morgan Kaufmann Publishers, San Francisco, 1998.