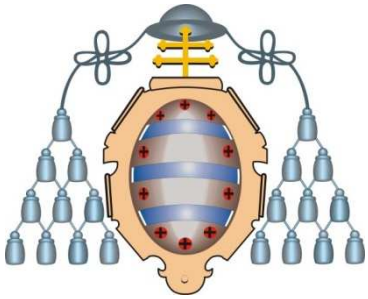


UNIVERSIDAD DE OVIEDO

DEPARTAMENTO DE EXPLOTACIÓN Y PROSPECCIÓN DE MINAS

TESIS DOCTORAL

PREDICCIÓN Y CLASIFICACIÓN DEL NIVEL DE RIESGO EN PROYECTOS DE SISTEMAS DE INFORMACIÓN

AUTOR: Carlos Alba González-Fanjul

DIRECTORES: Dr. Francisco Ortega Fernández

Dr. Vicente Rodríguez Montequín

SEPTIEMBRE, 2008

Agradecimientos

Quiero agradecer enormemente toda la ayuda recibida para la realización de esta tesis doctoral por parte de mis directores, Francisco Ortega y Vicente Rodríguez. He tenido la suerte de poder trabajar a su lado unos cuantos años y de aprender infinidad de cosas. Por todo ello, muchas gracias.

En estos años de trabajo siempre me he sentido arropado y parte de "algo" junto con el resto de compañeros del Área de Proyectos de Ingeniería: Joaquín, Cristina, Aida, Alberto, Mayte, Fernando, Valer, Nieves, Gemma, Roberto y todo el resto de nombres me han hecho feliz cada jornada laboral, a todos vosotros ¡Gracias!.

Agradezco al *International Software Benchmarking Standards Group* el esfuerzo realizado para la obtención de una base de datos acreditada con históricos de cierres de proyectos y el envío de la misma a cambio de un precio no significativo debido a la finalidad de uso que es la investigación.

Agradezco a mis padres y mi hermana los momentos que hemos pasado juntos paralelamente al desarrollo de este trabajo, en los que siempre han intentado transmitirme tranquilidad.

Doy las gracias a mi queridísima Mati que durante tanto tiempo ha aguantado firmemente a mi lado en todo momento, sin muchos fines de semana, sin muchos puentes, sin muchas noches … y siempre con una sonrisa y un gesto amable. El tiempo ha ido transcurriendo, las clases en la Universidad y los proyectos desarrollados en el Centro de Desarrollo Tecnológico de ArcelorMittal Asturias han provocado retrasos en la defensa de esta tesis, pero ello me permite dedicársela a mi hijo recién nacido Carlos, espero que el día de mañana se sienta orgulloso de esta dedicatoria.

Índice general

Índice de tablas:

Índice de figuras:

Capítulo 1: Introducción y descripción del problema

Predicción y clasificación del riesgo en los proyectos software.

1. INTRODUCCIÓN

Desde que en 1968 se comenzó verdaderamente con el desarrollo de proyectos dentro del sector informático, se han detectado graves problemas para cumplir las premisas principales de un proyecto: cumplir con los objetivos de calidad dentro de un tiempo predefinido y con unos costes prefijados, tal como ocurre en cualquier otro sector. Este fenómeno se conoce como "Crisis del software" y se define como la dificultad existente en desarrollar programas sin errores o defectos, fácilmente comprensibles, y que sean verificables. Las causas de la crisis del software parecen ser, entre otras, la complejidad que supone la tarea de programar y los cambios a los que se tiene que ver sometida una aplicación para ser continuamente adaptada a las necesidades de los usuarios.

No obstante el esfuerzo realizado para intentar solventar la crisis del software ha sido tradicionalmente escaso e insuficiente. A finales de la década de los 80 se comenzaron a realizar algunos estudios relacionados y sólo desde finales del siglo XX se ha invertido un gran esfuerzo en determinar las causas y proponer las soluciones para la denominada crisis del software.

En 1986, Alfred Spector, presidente de *Transarc Corporation* publicó un artículo comparando la construcción de puentes con el desarrollo software [Spector et al 1986]. La premisa consistía en que los puentes normalmente se construyen en el tiempo establecido, con el presupuesto asignado y no se caen. Por el contrario, el desarrollo software nunca se completa en los plazos asignados, ni de acuerdo con el presupuesto asignado, y por lo tanto fracasa. La razón fundamental de esta diferencia está en el diseño extremadamente detallado. El diseño de un puente permanece inalterable y no admite cambios y por lo tanto el contratista tiene pocas posibilidades de cambiar las especificaciones. Otra diferencia es que cuando un puente se cae, se investigan las causas y se acumulan para otras futuras construcciones, mientras que en el desarrollo software los fracasos se ocultan y no se obtiene el beneficio producido por las lecciones aprendidas.

Tras esta primera aproximación, se comenzó de forma más rigurosa a estudiar el problema y sus posibles soluciones. En los últimos años surgen distintas instituciones que realizan informes y análisis estadísticos, como pueden ser: GAO (Government Account Office) [GAO] que analiza proyectos de desarrollo de software para el

Gobierno Americano o ESPITI (European Software Process Improvement Training Initiative) [ESPITI] que realiza estudios sobre los principales problemas en el desarrollo de software a nivel europeo, y cuyos resultados son muy similares a los obtenidos en otro de los informes más aceptados, el CHAOS [Standish Group], indicando que los mayores problemas están relacionados con la especificación, la gestión y la documentación de los proyectos.

Los resultados de las investigaciones CHAOS [CHAOS ART] son los más contrastados a nivel mundial en la industria de las Tecnologías de la Información, a partir de ahora TI, y representan una década de datos que incluyen más de 50.000 proyectos y que indican los niveles de éxito o fracaso de los proyectos informáticos. El objetivo de estas investigaciones es proporcionar una comprensión de las razones por las que fracasan los proyectos, así como de los principales factores de riesgo y analizar las claves que pueden reducir los fracasos. El objeto de investigación del grupo *Standish Group* se centra en identificar el alcance de los fracasos del software, los factores principales que causan el fracaso de los proyectos software y los ingredientes clave que pueden reducir el fracaso de los proyectos.

Fruto de este esfuerzo es una mejora significativa de los resultados de los proyectos TI y, muy especialmente la disminución de los niveles de fracaso. A mediados de los años 90 empezaron a aparecer iniciativas de aplicación de procesos de mejora, utilizando los modelos CMM [CMM 1987], SPICE (ISO/IEC 15504) [Van Loon 2005], BOOTSTRAP [Bootstrap 1993], etc. Como consecuencia de la aplicación de estos procesos de mejora en los proyectos, los datos reflejan una mejora considerable [CHAOS 2003], con un incremento del 100% en proyectos que finalizan exitosamente respecto a los datos de 1994 [CHAOS 1994]. En la Fig. 1 se observa un incremento continuado del éxito de los proyectos entre los años 1994 y 2004, al mismo tiempo que se reducen de la misma manera los proyectos fracasados.

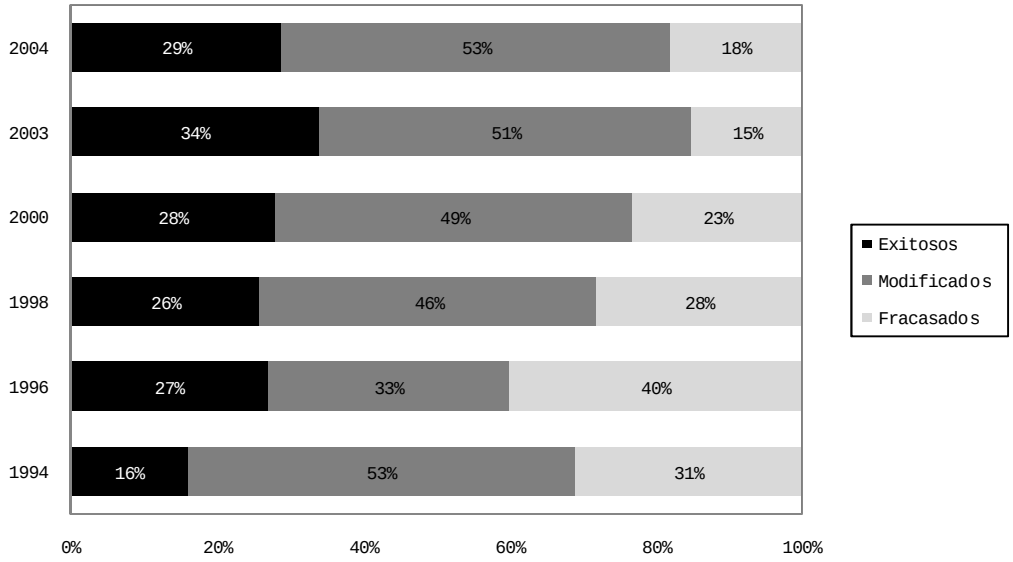


Fig. 1. Evolución del éxito de los proyectos TI. Fuente: Informe CHAOS - Standish Group

Ya en el análisis CHAOS hasta el año 2000 [CHAOS 2000] se revelaba una mejora en la gestión de los proyectos de TI con la implantación de estándares y mejores prácticas, con un crecimiento en proyectos exitosos y una caída en proyectos fracasados, mientras que los proyectos que sufren muchos cambios tendían a estabilizarse.

Esta evolución se confirma en el 2004 [CHAOS 2004], aunque con incertidumbres y parece alcanzarse una barrera que impide el crecimiento de los proyectos exitosos y la tendencia es el mantenimiento de los niveles conseguidos en los últimos años. Dada la ralentización de la mejora, se comienza la búsqueda de las causas con más exhaustividad, detectándose como factores críticos de los proyectos con problemas los siguientes:

- Falta de información por parte de los usuarios
- Especificaciones y requisitos incompletos o cambiantes
- Falta de apoyo de los directivos
- Incompetencia tecnológica
- Falta de recursos
- Expectativas no realistas
- Objetivos poco claros
- Plazos temporales no realistas
- Uso de tecnología novedosa

Por otra parte se determina que los proyectos se cancelan debido, entre otras, a las siguientes causas:

- Requisitos incompletos
- Falta de participación de los usuarios
- Falta de recursos
- Expectativas no realistas
- Falta de apoyo de los directivos
- Incompetencia tecnológica
- Falta de gestión de las TIC
- Desconocimiento de la tecnología.

El informe ESPITI (European Software Process Improvement Training Initiative) también propone la influencia de los distintos factores en el éxito o fracaso de los proyectos, realizando para ello un muestreo bastante significativo. Según un informe de la Universidad de Sevilla [GIS_US 2005] basado en dichos resultados, la gestión inadecuada de los proyectos supone el 30% de los grandes problemas que se tienen al desarrollar proyectos tecnológicos. Atendiendo a este informe, los factores de éxito de los proyectos TIC están muy relacionados con la implicación de los usuarios, el apoyo de los directivos, un enunciado claro de los requisitos, la planificación adecuada, las expectativas realistas, el personal competente, el sentimiento de propiedad, la visión y objetivos claros y el trabajo duro y concentrado del personal del equipo de proyecto.

Sobre ésta misma base de datos de ESPITI, otro grupo de investigadores [Lee et al 99] realizó un análisis desde la perspectiva de los perfiles de problemas de producción del software y su relación con el contexto organizativo de las unidades europeas de producción software. A partir de los resultados obtenidos se observa que, en general para todos los sectores, las unidades de producción se ven afectadas por deficiencias en las especificaciones, sin embargo el grado de problemas varía de acuerdo con el tamaño de la empresa, mientras que el uso de Normas ISO no muestra impacto alguno en los problemas de producción. También concluyen con unos resultados significativos en cuanto al uso de métodos de calidad, como por ejemplo que el 65% de las compañías europeas no utilizan procesos de mejora del software, el 86% de las compañías europeas no emplean métodos de valoración del software y el 80% no siguen ISO 9000. Sin

embargo la disposición general de los equipos técnicos y de gestión en el uso de herramientas y métodos de calidad, de desarrollo de proyectos e ISO 9000 es positiva.

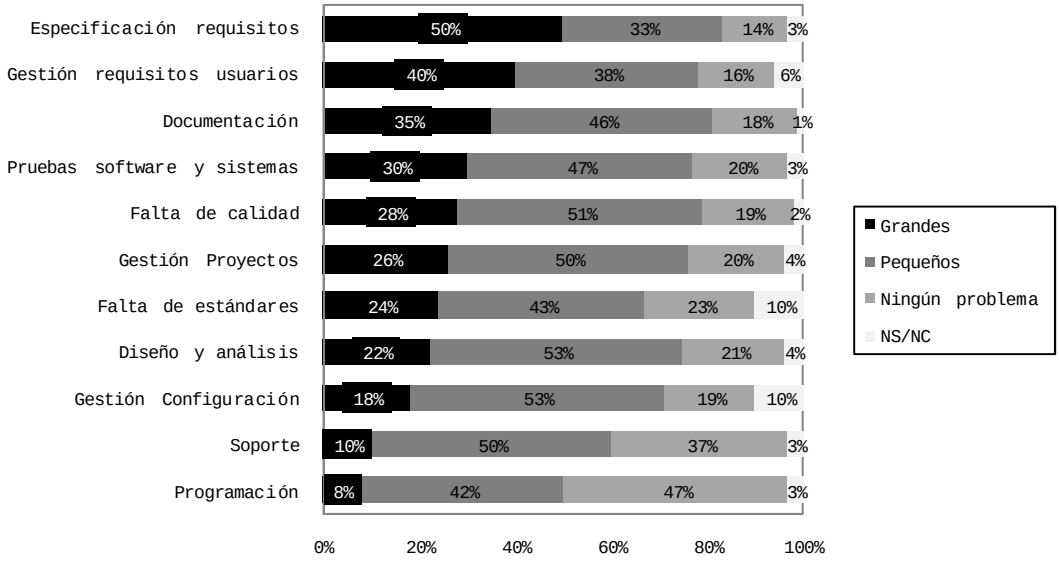


Fig. 2 Influencia de los factores de fracaso en los proyectos. Fuente: ESPITI

Desafortunadamente, a pesar de que los informes reflejan niveles de fracasos inaceptables y muy superiores a otros sectores, no parece que las organizaciones hayan decidido atacar el problema de forma rotunda. Así el informe CHAOS [CHAOS Q 2004] que refleja los planes de inversión en calidad, pruebas e inspección previstos para el 2005 por parte de las organizaciones encuestadas y su incremento respecto a lo invertido en el 2004, según se observa en la Tabla 1, es menos esperanzador.

	PREVISTOS PARA 2005				RESPECTO A 2004		
	ALTA	**MEDIA**	**BAJA**	**NINGUNA**	**MAYOR**	**IGUAL**	**MENOR**
Habilidades	12%	20%	51%	18%	22%	71%	8%
Hardware	0%	25%	37%	37%	14%	76%	10%
Software	6%	24%	43%	27%	27%	65%	8%
Servicios	10%	20%	43%	27%	27%	63%	10%

Tabla 1: Planes de inversión en calidad, pruebas e inspección. Fuente: CHAOS. Standish Group

La inversión en esta área, para la mayor parte de las compañías es mínima y los gastos que preveían realizar en el 2005 eran similares a los empleados en el 2004. Es de destacar que el 27% de las organizaciones no se planteaban inversiones en calidad para el 2005, ni en el apartado de software ni en el de servicios.

Otro estudio interesante es el elaborado por The Financial Express [Financial Express] que calculaba para el 2005 una desviación en tiempo del 36% de los proyectos y un 38% de proyectos que sufren desviaciones en costes.

2. LA CRISIS DEL SOFTWARE Y LA DIRECCIÓN DE LOS RIESGOS

El desarrollo de un proyecto de sistemas de información está sometido a multitud de riesgos y amenazas de diversa naturaleza, como pueden ser posibles "bugs" introducidos por las casas fabricantes de lenguajes de desarrollo en sus productos, pasando por fallos de hardware en los dispositivos de almacenamiento utilizados en el desarrollo del proyecto, ataques producidos por virus o incluso cambios políticos en proyectos desarrollados para las Administraciones Públicas, que pueden ocasionar cancelaciones de contratos dejando a la empresa en situaciones económicas inestables o con un exceso de plantilla sin trabajo que producir. Se puede deducir de los resultados de los informes anteriores que la mayor parte de los problemas detectados que provocan graves desviaciones de las expectativas iniciales no son de tipo técnico, sino factores de gestión, metodológicos o relacionados con las personas. La introducción de metodologías de calidad antes citadas, si bien supuso una mejora inicial, no es capaz por sí misma de conseguir niveles aceptables de errores, similares a los de otros sectores.

El principal problema está en la carencia, dentro de estas metodologías, de aspectos de la Dirección de Proyectos diferentes de la calidad o la gestión de costes y plazos. Si bien estos factores son fundamentales, la dirección de proyectos requiere la consideración de otros condicionantes, indispensables para la perfecta ejecución del proyecto. Aunque existen diversas aproximaciones a estos factores, las dos más reconocidas a nivel internacional son las propuestas por IPMA (International Project Management Association) [IPMA] y PMI (Project Management Institute) [PMI].

IPMA, agrupación internacional de asociaciones nacionales, realiza una enumeración detallada en 40 componentes en su ICB (International Components Baseline) [IPMA 2006], 12 de los cuales son adaptados por cada país en su NCB (National Components Baseline)[1]. Específicamente esto es realizado en España por AEIPRO (Asociación Española de Ingeniería de Proyectos) [AEIPRO]. Es inmediato observar cómo los aspectos técnicos o incluso los relacionados con calidad son una minoría frente al total de factores a contemplar en la dirección de un proyecto.

La aproximación de PMI en su PMBoK (Project Management Body of Knowledge) [PMBoK 2004], es más aceptada dada la mayor difusión de esta organización, principalmente en EE.UU y su sencillez. Además dichos factores sirvieron como base a la redacción de la norma ISO 10006 [ISO10006] con lo que se convirtieron en un estándar *de facto*. PMI considera nueve factores a dirigir: costes, plazo, alcance, integración, recursos humanos, aprovisionamiento, calidad y riesgo.

Es precisamente este punto, el riesgo, el que se detecta como una carencia generalizada en los proyectos software. La omisión de la gestión de estos riesgos provoca, en gran número de ocasiones, entregas de productos finales fuera de plazo y con un mayor coste respecto a lo planificado. Todo esto sin olvidar que, en muchos casos, el producto final obtenido no cumple con las especificaciones y requerimientos establecidos por el cliente antes o durante el desarrollo del proyecto. Según se publica en [Cutter 2000] respecto a los grandes proyectos Web, el 85% no cumplen con las expectativas de negocio, el 53% no disponen de la funcionalidad requerida, el 63% sufren retrasos en el desarrollo y un 58% sobrepasan ampliamente el presupuesto inicialmente planificado.

[1] Los NCB contemplan el entorno cultural adaptado a cada país en particular

Estos niveles de fracaso se deben a la aparición durante el desarrollo del proyecto de imprevistos, problemas que no se habían detectado en las fases iniciales de análisis (por otra parte muy breves e incompletas en este tipo de proyectos) y que al aparecer plantearon la necesidad de cambios o adaptaciones que condujeron a fuertes retrasos, incrementos de costes o variaciones en las funcionalidades inicialmente previstas.

Los equipos de desarrollo software siguen centrando sus esfuerzos en el propio desarrollo, en la integración de sistemas, en las comunicaciones, en definitiva, en los aspectos puramente técnicos de los proyectos de sistemas de información, dejando totalmente al margen los aspectos no técnicos, que resultan imprescindibles para el buen desarrollo del proyecto, por mucha calidad técnica que aporte el equipo. Es evidente que una mejor gestión del riesgo del proyecto podría disminuir la aparición de estos problemas y minimizar sus consecuencias, siendo conscientes de que su eliminación total es inviable, puesto que el riesgo es una de las características básicas de los proyectos. No obstante, desafortunadamente en la actualidad no se hace, excepto en casos puntuales, una buena gestión completa del riesgo a lo largo de la vida del proyecto. En el mejor de los casos el esfuerzo es limitado y se plasma en un Plan de Contingencia básico, generalmente realizado por analogía con otros proyectos y no específico del proyecto al que va dirigido. Esta forma de actuar puede ser debida al hecho de que controlar todos los posibles riesgos a los que está sometido un proyecto software es difícil, debido a la escasa formación en estos temas de los directores del proyecto o simplemente a la falta de confianza en los resultados: las escasas mejoras obtenidas no parecen compensar el esfuerzo inicial.

Es por esto que todo aquello que pueda demostrarle al director de proyecto que gestionar los riesgos disminuye los problemas es esencial para el objetivo global de disminuir el nivel de fracaso de los proyectos.

Predecir o estimar el nivel de riesgos a los que un proyecto software está expuesto (en cuanto a gravedad y frecuencia) en función de sus características (tipo de lenguaje de programación, aplicación cliente/servidor, aplicación Web, número y experiencia del equipo de desarrollo, utilización o no de herramientas CASE, etc.) es sin duda un avance, permitiendo a los equipos de desarrollo software centrar esfuerzos en prevenir ese tipo de riesgos y dejando quizá más al descubierto aspectos con probabilidad de ocurrencia baja o casi nula. Por otra parte es evidente que no todos los proyectos son

iguales. Por su duración, tipo de trabajo, coste, organizaciones implicadas, etc., algunos proyectos tienen un nivel de riesgos más severo que otros. Sin embargo no resulta fácil realizar esta clasificación puesto que depende de factores muy diversos, tal como se ha dicho anteriormente.

Considerando ambas circunstancias, se considera de gran utilidad poder encontrar un modelo que, partiendo de información básica del proyecto sea capaz de predecir su dificultad y, a partir de dicha predicción, clasificarlo en función de su riesgo, de modo que se puedan proponer medidas coherentes y proporcionadas al nivel de problemas esperado durante su desarrollo.

La predicción de los riesgos en los trabajos relacionados con sistemas de información presenta dificultades particulares si la comparamos con la predicción en otros sectores. En sectores industriales es habitual producir el mismo tipo de producto una y otra vez, y con los mismos métodos. En el software, por el contrario, es habitual desarrollar un nuevo producto cada vez, empleando distintas técnicas y herramientas, es decir, está más orientado a proyecto. Como conclusión de lo anterior, el grado de repetitibilidad existente dentro de un proyecto software (reutilización de código, diseño modular, utilización de patrones de diseño) debería reducir el riesgo del proyecto. La predicción de riesgos en los proyectos de software es más inexacta, aunque no imposible. Existen además otras razones que dificultan la estimación y gestión de riesgos en los proyectos de software, entre ellas las presiones en la empresa para disminuir el coste (común a proyectos en otros sectores como la obra civil) y el hecho de que existe una carencia generalizada de datos sobre proyectos terminados (entorno del proyecto y problemas ocurridos) que podrían guiar a los profesionales a la hora de realizar estimaciones. Esta situación se agrava debido a actitudes optimistas, bien sea de la alta Dirección, del director del proyecto o del equipo de desarrollo, los cuales en muchas ocasiones deciden no intentar prevenir riesgos potenciales debido a que es una tarea que cuesta tiempo y dinero con el fin de evitar situaciones que todavía no han tenido lugar.

Entendiendo el nivel de riesgo del proyecto como la probabilidad de que el proyecto fracase a la hora de alcanzar los objetivos especificados, puede ser representado por una función de densidad que define cómo de arriesgado es el proyecto. De esta forma, si un proyecto presenta un nivel de riesgo igual al 30%, se entiende que tiene un 70% de probabilidad de tener éxito. El nivel de riesgo ayuda a los directores de proyecto a poder

realizar comparaciones de proyectos basadas en esta probabilidad. Así, cuantificando los riesgos y teniendo en consideración los aspectos del proyecto que presenten valores más altos en dicha puntuación, los directores de proyecto pueden identificar dónde aplicar y asignar los recursos, normalmente limitados, para intentar paliar de forma preventiva estos riesgos incrementando la probabilidad de éxito del proyecto.

3. OBJETO Y ALCANCE DEL TRABAJO

En este trabajo se propone la utilización de estrategias de modelado utilizando técnicas basadas en datos. Esta aproximación ha sido utilizada con buenos resultados desde hace más de dos décadas en la resolución de muchos problemas complejos, dando lugar a una disciplina conocida en la comunidad científica con el término "data mining". Se analiza el desarrollo de un sistema basado en técnicas de inteligencia artificial capaz de seleccionar las variables que afectan a los defectos potenciales en un proyecto de desarrollo de software a partir de un conjunto de datos históricos. Este tipo de aproximación plantea dos requisitos imprescindibles:

- La necesidad de un número importante de datos que sean compatibles con las técnicas a utilizar y
- La utilización de datos representativos de proyectos diversos en características y disponibles para su contraste por otros investigadores.

Un factor importante a la hora de poder procesar un conjunto de datos, es la calidad de la información que se va a procesar. Cuando se habla del concepto de calidad de información se hace referencia a que se ha de disponer de la información necesaria para el objetivo; por eso la medición de las variables ha de recoger las características propias del atributo que se pretende medir, en este caso, número de defectos y gravedad de los mismos en los resultados de un proyecto software.

Por ello, para el desarrollo del trabajo se utiliza un conjunto de datos pertenecientes al *International Software Benchmarking Standards Group (ISBSG)*, recopilados a partir de la información extraída de la ejecución de más de 3.000 proyectos. Estos datos contienen valores numéricos y categóricos, existiendo un gran porcentaje de valores perdidos. Por este motivo, los datos son sometidos a un exhaustivo preprocesamiento y se seleccionan las técnicas de inteligencia artificial que mejor se ajustan a la condición de los datos.

4. CONCLUSIONES

Todos los estudios realizados y mencionados anteriormente coinciden en que la industria del software obtiene productos con gran número de deficiencias y de forma generalizada, con desviaciones en tiempo y coste respecto a lo planificado.

Debido principalmente a la ocultación de estos defectos en los proyectos software y a la ausencia en muchos de los casos de intentar aprender del pasado mediante lecciones aprendidas, la evolución o posible mejora es realmente lenta e insuficiente.

Conocer, prevenir e intentar mitigar cualquier riesgo al que pueda estar sometido un proyecto software es probablemente, una tarea inabordable, debido principalmente a la enorme cantidad de amenazas y diversas naturalezas de las mismas presentes desde antes del comienzo del proyecto. No obstante lo anterior, la realización de un modelo que permita a los directores de proyectos software conocer los principales riesgos a los que puede estar sometido el proyecto, así como la gravedad de los mismos, desde antes del comienzo del proyecto en función de cierta información básica y en la mayor parte de los casos, disponible, permitiría emplear los pocos recursos normalmente asignados a la gestión de riesgos en los problemas potenciales con mayor probabilidad de ocurrencia teniendo en consideración a su vez la gravedad de los mismos.

Una reciente investigación sobre técnicas y procedimientos que la organización debe conocer para predecir el coste que conlleva el desarrollo de un proyecto de sistemas de información fue la tesis presentada en el año 2005 "*Estimación de costes y plazos en proyectos de sistemas de información*" [Villanueva 2005] en la Universidad de Oviedo. Como línea de investigación derivada de esa tesis, surgió el aspecto del riesgo, que se intenta cubrir en este trabajo.

Esta Tesis Doctoral centra sus esfuerzos en la predicción y clasificación de los riesgos a los que estará sometido un proyecto de sistemas de información y cómo pueden afectar los mismos a la calidad del producto obtenido en forma de número de fallos y gravedad de los mismos, sin olvidar en ningún momento las posibles desviaciones en plazo que puedan llegar a tener lugar.

Capítulo 2: Estado del arte

Predicción de Riesgos en Proyectos de Sistemas de Información.

1. INTRODUCCIÓN

El interés de las empresas por la mejora de la calidad de sus productos de sistemas de información, así como los requerimientos cada vez más exigentes de sus clientes, han impulsado el desarrollo de numerosas investigaciones destinadas a la estimación de riesgos, lo que implica identificar los parámetros que afectan a esta estimación y recoger datos históricos del cierre de proyectos para disponer de información útil para su posterior análisis.

Este capítulo está dedicado a mostrar la evolución y diferentes aproximaciones utilizadas para estimar los riesgos a los que estará sometido un proyecto software, riesgos con los que los directores de proyecto tendrán que enfrentarse a lo largo de la ejecución del mismo.

2. EVOLUCIÓN DE LA GESTIÓN DE RIESGOS EN PROYECTOS SOFTWARE

Durante los últimos años la gestión de riesgos en proyectos ha visto incrementada su relevancia, especialmente en los proyectos software. La mayor parte de los investigadores y los profesionales que trabajan en el campo de la gestión de proyectos informáticos asumen que las incertidumbres afrontadas en los proyectos software deberían ser consideradas a la hora de planificar y controlar el desarrollo de un proyecto de este tipo [Costa et al 2007].

Sin embargo, la gestión de los riesgos en el software es un tema cuya evolución va pareja a la evolución de las técnicas y métodos del software. Así, McFarlan ya identificó en 1981 tres grandes grupos de factores de riesgo en los proyectos informáticos: el tamaño del proyecto, la experiencia en la tecnología y la estructura del proyecto [McFarlan 1981]. McFarlan sugirió que los directores de proyecto necesitaban desarrollar un perfil global de riesgos para los proyectos de software. [Boehm 1991] encuestó a varios directores de proyecto experimentados y desarrolló una lista de chequeo para identificar los 10 riesgos más significativos. [Barki et al 1993] llevaron a cabo una revisión completa de diferentes estudios relacionados con riesgos software provenientes de 120 proyectos en 75 organizaciones, proponiendo 35 medidas para riesgos software que fueron

clasificadas en 5 grupos: novedad tecnológica, tamaño de la aplicación, experiencia, complejidad de la aplicación y entorno de la organización.

[Sumner 2000] hizo uso de una entrevista estructurada para comparar las diferencias entre los riesgos software dentro de los proyectos MIS (Management Information systems) y ERP (Enterprise Resource Planning), y propuso 9 riesgos que eran únicos en los proyectos ERP. [Longstaff et al 2000] propusieron un marco de trabajo denominado *Hierarchically Holographic Modeling – HMM* (Modelado holográfico jerárquico) e identificaron 7 puntos de vista distintos para la integración de sistemas incluyendo 32 riesgos. [Kliem 2001] identificó 38 riesgos en proyectos BPR que fueron clasificados en 4 categorías: recursos humanos, gestión, negocio y tecnología.[Addison 2003] utilizó la técnica Delphi para recolectar la opinión de 32 expertos y propuso 28 riesgos relacionados con los proyectos de comercio electrónico. Al mismo tiempo, [Carney et al 2003] diseñaron una herramienta llamada *COTS Usage Risk Evaluation – CURE* (Evaluación del riesgo al usar herramientas comerciales) para predecir las áreas de riesgo cuando se adquieren herramientas comerciales para el desarrollo de un proyecto, identificando 4 categorías conteniendo 21 áreas de riesgo.

Autor (año)	Área de investigación	Categorías	Riesgos software
McFarlan (1981)	General	3	54
Boehm (1991)	General	0	10
Barki et al. (1993)	General	5	35
Sumner (2000)	ERP	6	19
Longstaff et al. (2000)	Integración de sistemas	7	32
Cule et al. (2000)	General	4	55
Kliem (2001)	BPR	4	38
Schmidt et al. (2001)	General	14	33
Houston et al. (2001)	General	0	29
Murthi (2002)	General	0	12
Addison (2003)	Comercio electrónico	10	28
Carney et al. (2003)	Herramientas comerciales	4	21
Wallace et al. (2004)	General	6	27

Tabla 2 Resumen de los riesgos en el desarrollo software

Categoría	Abreviatura	Riesgo software
Usuarios	User1	Resistencia al cambio
	User2	Conflictos entre usuarios
	User3	Usuarios con actitudes negativas
	User4	Falta de compromiso
	User5	Falta de cooperación
Requisitos	Reqm1	Requisitos cambiantes
	Reqm2	Falta de identificación de requisitos
	Reqm3	Requisitos poco claros
	Reqm4	Requisitos incorrectos
Complejidad	Comp1	Uso de tecnologías novedosas
	Comp2	Complejidad tecnológica alta
	Comp3	Tecnología inmadura
	Comp4	Tecnologías no usadas previamente
Planificación & Control	P & C1	Falta de una metodología
	P & C2	Falta de seguimiento del progreso del proyecto
	P & C3	Estimación inadecuada de recursos
	P & C4	Planificación pobre
	P & C5	Hitos no claramente definidos
	P & C6	Director de proyecto inexperto
	P & C7	Falta de comunicación
Recursos humanos	Team1	Equipo de desarrollo inexperto
	Team2	Equipo de desarrollo inadecuado
	Team3	Falta de especialización en puntos clave
Entorno de la organización	Org1	Cambios organizativos durante la ejecución
	Org2	Políticas corporativas con impacto negativo
	Org3	Entorno inestable
	Org4	Reestructuración de la organización durante la ejecución

Tabla 3 Riesgos software considerados en [Wallace et al 2004]

El trabajo desarrollado por [Wallace et al 2004] define 27 riesgos software que fueron clasificados en 6 categorías y se muestran en la Tabla 3. Esta estructura se utilizó en el estudio para investigar los efectos de los riesgos en el rendimiento de los proyectos.

La importancia de los riesgos en los proyectos de software son expresados normalmente como una combinación de probabilidad de ocurrencia e impacto en el rendimiento del proyecto. En esta línea, existen tres métodos ampliamente conocidos de gestión de

riesgos software en la literatura: SRE (Software Risk Evaluation), SERIM (Software Engineering RIsk Management) y la DoD Guide (Department of Defense).

El método SRE fue desarrollado por el *Software Engineering Institute* [SEI] para identificar, analizar y desarrollar estrategias de mitigación para el control de riesgos [Williams et al 1999]. La versión 2.0 del método SRE, disponible desde 1999, describe una escala de probabilidad de ocurrencia entre uno y tres, mientras el impacto fue definido por cuatro componentes: coste, planificación, mantenimiento y rendimiento tecnológico. Estos componentes de impacto no consideraron al "equipo" habiéndose convertido hoy en día uno de los factores críticos en los proyectos de software modernos [Jiang et al 2000].

[Karolak 1997] propuso un método denominado Software Engineering RIsk Management (SERIM) para predecir riesgos y proponer acciones correctoras en la fase de desarrollo de software. El método SERIM identificó 10 riesgos software y contiene 81 preguntas relacionadas con una métrica para medir estos riesgos. Las relaciones entre los riesgos software fueron identificadas usando tres componentes de riesgo: coste, planificación y rendimiento tecnológico. Al igual que el método SRE, el método SERIM no incluyó al componente "equipo".

[DoD 2003], elaborado por la Secretaría de Defensa de EE.UU., propuso un modelo de procesos que define cómo una organización puede identificar, evaluar y gestionar riesgos durante el desarrollo de software. En este modelo se identificaron cinco niveles para evaluar la probabilidad de ocurrencia de riesgos software: remota, poco probable, probable, muy probable y casi cierto. El impacto de los riesgos software representa el grado de influencia en el rendimiento del proyecto. Los componentes de impacto de los riesgos software sobre el rendimiento del proyecto en el modelo DoD incluyen el rendimiento tecnológico, coste, planificación y equipo.

La Tabla 4 muestra una comparación de los diversos componentes de impacto utilizados por los métodos de evaluación de riesgos software comentados.

Impacto	SRE	SERIM	DoD
Coste	✓	✓	✓
Planificación	✓	✓	✓
Rendimiento tecnológico	✓	✓	✓
Equipo			✓
Mantenimiento	✓		

Tabla 4 Componentes de riesgo en la evaluación de riesgos

El método DoD incluye diferentes tipos de impacto así como unos criterios de evaluación y descripción claros para cada escala de probabilidad de ocurrencia e impacto. La evaluación de la probabilidad de ocurrencia e impacto de los riesgos software basada en el método DoD se muestra en la Tabla 5.

Nivel	Probabilidad de ocurrencia	Impacto			
		Rendimiento tecnológico	**Coste**	**Planificación**	**Equipo**
1	Remota	Mínimo o sin impacto	Mínimo o sin impacto	Mínimo o sin impacto	Ninguno
2	Poco probable	Aceptable con alguna reducción en el margen	<5%	Requiere recursos adicionales	Impacto leve
3	Probable	Aceptable con importante reducción en el margen	5–7%	Pequeños retrasos en hitos clave	Impacto moderado
4	Muy probable	Aceptable. Margen restante	7–10%	Retrasos significativos en hitos clave (Camino crítico)	Impacto grande
5	Casi cierto	Inaceptable	>10%	Hitos clave inalcanzables	Inaceptable

Tabla 5 Niveles de evaluación de riesgos DoD

El Ministerio de Administraciones Públicas de España y en particular, el Consejo Superior de Administración Electrónica elaboró en 1.997 la primera versión de MAGERIT, promoviendo su utilización como respuesta a la percepción de que la Administración (y en general toda la sociedad) depende de forma creciente de las tecnologías de la información para el cumplimiento de su misión. La razón de ser de MAGERIT está directamente relacionada con la generalización del uso de los medios electrónicos, informáticos y telemáticos, que supone unos beneficios evidentes para los ciudadanos; pero también da lugar a ciertos riesgos que deben minimizarse con medidas de seguridad que generen confianza.

En el periodo transcurrido desde la publicación de la primera versión de MAGERIT (1.997) hasta la fecha, el análisis de riesgos se ha venido consolidando como paso necesario para la gestión de la seguridad.

MAGERIT persigue los siguientes objetivos:

1. Concienciar a los responsables de los sistemas de información de la existencia de riesgos y de la necesidad de atajarlos a tiempo
2. Ofrecer un método sistemático para analizar tales riesgos
3. Ayudar a descubrir y planificar las medidas oportunas para mantener los riesgos bajo control
4. Apoyar la preparación a la Organización para procesos de evaluación, auditoría, certificación o acreditación, según corresponda en cada caso

Los informes generados recogen los hallazgos y las conclusiones de un proyecto de análisis y gestión de riesgos: modelo de valor, mapa de riesgos, evaluación de salvaguardas, estado de riesgo, informe de insuficiencias, y plan de seguridad. La versión 2 de [MAGERIT] se publicó en junio de 2.006.

[Tiwana et al 2006] realizaron un estudio basado en una investigación empírica para modelar la influencia de los factores funcionales en el riesgo de los proyectos de sistemas de información. Desarrollaron un modelo de riesgo funcional para explicar la importancia relativa de seis factores de riesgo funcional que habían sido identificados de forma consistente como de gran importancia: Conocimiento técnico relacionado, entorno del cliente, volatilidad de los requisitos, ajuste a la metodología de desarrollo, prácticas formales de dirección de proyectos y complejidad de proyecto. El modelo se probó de forma empírica con 60 directores de proyectos software experimentados de 60 organizaciones distintas. Según los trabajos comentados, en general se puede afirmar que existe un acuerdo respecto a los grandes grupos de factores de riesgos en proyectos de sistemas de información.

Ahora bien, cuando se habla del término riesgo se asocia normalmente con el concepto de éxito del proyecto. Éxito es un término muy difícil de precisar. Es algo muy subjetivo. Por supuesto, la apreciación de éxito dentro de un proyecto variará en función de la percepción de cada persona. Normalmente, las personas afectadas o

relacionadas con el proyecto desde fuera de la organización utilizarán criterios basados en el coste o en los plazos, mientras que las personas involucradas en el proyecto de forma interna coinciden en que el grado de éxito del proyecto puede medirse como el logro del alcance de desarrollo. Uno de los trabajos más interesantes a este respecto se debe a [Agarwal et al 2006]. En el estudio se sostiene que para evaluar el éxito hay que fijarse en dos aspectos diferentes de los proyectos: factores internos y externos. El interno aborda el éxito desde el punto de vista de la ejecución, seguimiento y control del proyecto a corto plazo, mientras que el segundo criterio lo aborda desde el punto de vista del valor final de los entregables que se les dará a los usuarios, teniendo un mayor impacto a más largo plazo.

La literatura clásica de gestión de proyectos considera que el coste, el plazo y la calidad son los criterios de éxito de un proyecto (de cualquier disciplina). Estos criterios de éxito se muestran en la Fig. 3 y son, por supuesto, los factores básicos a gestionar en la dirección de un proyecto. El alcance y los resultados obtenidos en la ejecución de un proyecto están condicionados por estos tres factores básicos. Cualquier modificación en alguno de ellos condiciona directamente a los otros dos.

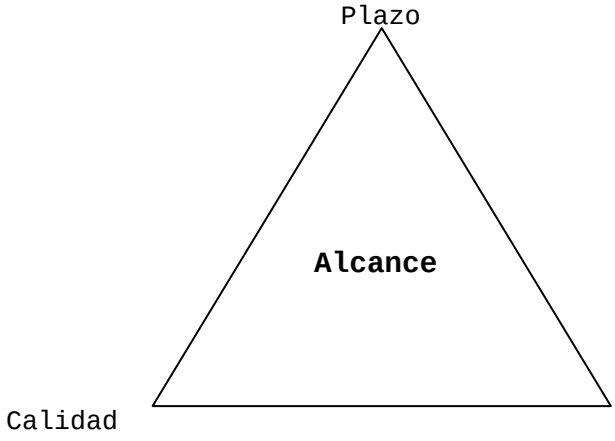


Fig. 3 Factores básicos a gestionar en la dirección de un proyecto. Fuente: [Kerzner, 2003]

Como ya se comentó en el capítulo introductorio de este trabajo, una de las referencias más asentadas refiriéndose a "éxito" dentro de los proyectos informáticos viene dada por el informe que desde 1994 publica el Standish Group, conocido como Chaos Report [CHAOS 1994]. El Standish Group califica un proyecto software como exitoso si cumple con los tres factores descritos en la Fig. 3 e impugnado si falla en conseguir cumplir alguno de los tres objetivos. En vez de declarar un proyecto como fracaso, este informe considera un proyecto como cancelado si termina antes de ser completado.

Esta perspectiva referida a los criterios de éxito del proyecto ignora el hecho de que un proyecto exitoso puede hacer entrega de un producto software no usable o que no satisfaga las necesidades funcionales del usuario. Es necesario aceptar que un proyecto puede ser evaluado desde distintos puntos de vista, por lo que restringir los criterios de éxito a coste, plazos y calidad limita la subjetividad inherente a cualquier proyecto de software.

Existen dos estudios empíricos que tratan de forma específica los criterios de éxito en los proyectos software. El primero es [Wateridge 1998], el cual realizó una investigación sobre directores de proyecto, analistas de sistemas, patrocinadores y usuarios con el objetivo de encontrar los cinco criterios de éxito más importantes en los proyectos de tecnologías de la información. Este estudio concluyó que "cumplir con los requisitos del usuario" es el criterio de éxito más importante si bien este concepto es diferente según para quién. Los usuarios calificaron el hecho de "tener a los usuarios contentos" como el segundo criterio más importante mientras que para los directores de proyecto este criterio ocupaba el séptimo lugar. Para los directores de proyecto, cumplir con el presupuesto y con los plazos es el siguiente criterio justo después de "cumplir con los requisitos del usuario". Esto significa que los usuarios asocian "cumplir con sus requisitos" con el hecho de "estar contentos" mientras que los directores de proyecto se centran en objetivos basados en presupuesto y planificación establecidos por la dirección en vez de centrarse en entregar un sistema o aplicación que mantenga contento al usuario final. El segundo estudio es [Lindbert 1999], que fue un caso de estudio basado en un proyecto software. Este estudio reveló la desviación existente en la percepción, entre los profesionales del software, de la idea de proyecto exitoso basado en la noción tradicional de cumplir con el presupuesto, plazos y calidad. Los desarrolladores unen el éxito de un proyecto terminado a la calidad del producto mientras que si un proyecto es cancelado lo unen a las lecciones aprendidas pensando en los proyectos futuros. Enlazan el éxito de sus esfuerzos de desarrollo con la satisfacción obtenida en el trabajo a través de la creatividad o lo que aprenden mientras están trabajando en el proyecto. En general, se puede afirmar que tras los estudios realizados en este campo, no es suficiente fijarse en los aspectos clásicos de control de proyectos, que podríamos denominar internos acorde a [Agarwal et al 2006], sino que se deben tener en cuenta también los factores externos.

En cuanto a las técnicas, hasta el momento se ha empleado un amplio abanico para intentar modelar/evaluar los riesgos dentro de estos proyectos. [Tuysuz et al 2006] usan un modelo basado en fuzzy y AHP (Analytic Hierarchy Process). Este método tiene la ventaja que permite evaluar el riesgo en un entorno de información vaga e incompleta. Las redes bayesianas han sido empleadas también con estos propósitos, tal como describe el modelo [Fan et al 2004]. [Fenton et al 2004] también han empleado este tipo de técnicas al igual que ocurre en [Matus 2006], tesis doctoral donde se intenta modelar mediante redes bayesianas los factores que afectan al proceso de desarrollo de software.

Las técnicas de simulación estocástica también han sido empleadas para modelar el riesgo de los proyectos informáticos [Houston et al 2001]. Uno de los modelos más claros de aplicación de las técnicas estocásticas en la gestión de riesgos dentro de los proyectos es precisamente la asignación de funciones de distribución de probabilidad a las tareas que conforman un proyecto, modeladas por ejemplo mediante la técnica PERT (Project Evaluation and Review Technique). Este mismo principio se puede aplicar a los factores que intervienen en los modelos de estimación de esfuerzo de proyectos, como el COCOMO. En el caso concreto del trabajo de [Houston et al 2001], se empleaba una red que modelaba la interdependencia entre los diferentes factores sobre el riesgo del proyecto.

También existen aproximaciones desarrolladas empleando Fuzzy Neural Networks. Así el trabajo presentado en [Benhai et al 2007], emplea esta técnica para evaluar el rendimiento de un proyecto.

Otros autores [Costa et al 2007] han realizado aproximaciones a la gestión de riesgos en este tipo de proyectos basándose en criterios económicos. Por medio de analogías con conceptos económicos, proponen una aproximación que estima la distribución de probabilidad referente a pérdidas y ganancias en la que incurrirá una organización acorde a su cartera de proyectos software. Sin embargo, estas aplicaciones van dirigidas a evaluar el riesgo de un portfolio de proyectos, no un único proyecto aislado.

Los aspectos relacionados con el ámbito geográfico donde se desarrollan los proyectos también han sido estudiados por diversos autores. Existen trabajos que incluyen dentro del planteamiento de estudio el grado de desarrollo de los países donde se realizan los proyectos. Así, [Kwan-Sik et al 2007] han particularizado los riesgos en función del entorno de desarrollo, en este caso Korea. Este tipo de análisis cobra gran importancia

desde el punto de vista de la tendencia, cada vez mayor, de externalizar el desarrollo de proyectos hacia países emergentes. A menudo es sugerido que los proyectos software externalizados muestran una necesidad de gestión y control del riesgo mayor que si se desarrollan internamente dentro de la organización [Wallace et al 2004b].

En cuanto a bases de datos para los estudios llevados a cabo, la mayor parte de los trabajos utilizan una colección propia de datos a partir de cuestionarios pasados a jefes de proyectos. Así por ejemplo, [Han et al 2007] utilizan un cuestionario que publican en una web. El cuestionario contiene cuatro secciones. La primera sección explica el propósito del estudio y anima a los directores de proyecto a participar en él. La segunda sección solicita información general sobre el último proyecto de software terminado. La tercera sección muestra 27 riesgos software y solicita que se evalúe la probabilidad de ocurrencia e impacto para cada riesgo de acuerdo con el método de evaluación de riesgos DoD mostrado en la Tabla 5. Por último, la cuarta sección muestra siete medidas usadas para medir el rendimiento del proyecto. Un total de 300 directores de proyecto fueron invitados a participar obteniéndose 135 cuestionarios retornados en el período de dos semanas. Una vez que se realizó el primer chequeo de los datos obtenidos se descartaron 20 cuestionarios por estar incompletos. Como resultado, solo 115 cuestionarios se consideraron válidos dando lugar a una tasa de respuesta del 38.33%. Los resultados aportados por estas investigaciones se basan en la confrontación de los factores considerados por los métodos tradicionales con los que consideran los expertos en gestión de proyectos. Sin embargo, los métodos basados en realización de cuestionarios de este tipo, pueden contener un alto grado de subjetividad. El resultado principal que se obtenía de este estudio era que el cumplimiento de los requisitos de usuario, programas sin defectos, es el principal criterio de éxito y medida del rendimiento de un proyecto software. Se desprende pues, que una de las formas para mejorar el rendimiento de los proyectos y el éxito es realizar una buena planificación de las actividades a desarrollar intentando reducir la complejidad del proyecto.

Otro grupo de investigadores basan sus trabajos en el análisis mantenido con grupos de expertos o personal involucrado en el desarrollo de proyectos. Así [Verner et al 2007] se reunieron con un grupo de desarrolladores americanos explorando el efecto de las prácticas de estimación a la hora de planificar y sus implicaciones en el éxito de los proyectos software. El objetivo no era sólo explorar los efectos directos de la estimación

de coste y plazos en el éxito o fracaso percibido del desarrollo del proyecto software, sino también examinar de forma cuantitativa una serie de factores relacionados con la estimación que pueden tener influencia en los productos obtenidos. Desde el punto de vista del desarrollador, los resultados sugieren que habrá más probabilidad de éxito si el director de proyecto está involucrado en la planificación, la información sobre los requisitos es adecuada y está disponible cuando las estimaciones están siendo realizadas, las estimaciones iniciales relativas al esfuerzo son buenas y se cuenta con la participación del equipo. Relacionado directamente con este estudio se encuentra [Villanueva 2005] que propone un método de estimación del esfuerzo basado en técnicas de inteligencia artifical, haciendo uso para ello de una base de datos acreditada, la perteneciente al *International Software Benchmarking Standards Group* Release 8.

3. CONCLUSIONES

La dirección del riesgo en los proyectos software destacaba tradicionalmente por su ausencia, limitándose a acciones correctoras cuando un hecho adverso tenía lugar. La dirección de riesgos en proyectos, y de forma particular, en los proyectos software ha explosionado en los últimos años debido a que los directores de proyecto, y más importante todavía, la alta dirección, se han dado cuenta de que es positivo para las organizaciones invertir dinero a priori intentando evitar que los riesgos se produzcan, dando lugar a acciones preventivas en vez de correctivas.

A lo largo de las últimas décadas, el enfoque de la dirección de riesgos ha cambiado también los objetivos. Mientras que se comenzó con un enfoque dirigido exclusivamente a cumplir con la planificación de costes, plazos y calidad, los riesgos derivados de un equipo inexperto, con malas relaciones interpersonales o con falta de comunicación empiezan a verse como riesgos muy importantes, influyendo de forma directa en los resultados del proyecto acorde al resto de factores.

La percepción del riesgo y de los criterios de éxito de los proyectos software varían enormemente en función del punto de vista desde el que se mire, bien desde la propia organización y en particular desde el equipo de proyecto, o bien desde el punto de vista del usuario. Un proyecto que cumpla en coste, plazo y calidad con la planificación realizada puede ser considerado exitoso de forma interna mientras que si no cumple con alguno de los requisitos esperados por el cliente, éste verá al proyecto como un

auténtico fracaso. De forma inversa, un proyecto que cumpla con todos los requisitos esperados por el cliente, aún significando un sobrecoste o entrega fuera de plazo, será visto a los ojos del usuario final como un éxito.

Las técnicas empleadas desde hace tiempo para la dirección del riesgo, en particular para la estimación a la hora de planificar se basan principalmente en el conocimiento experto y modelos de analogías. Técnicas más complejas basadas en datos, modelos probabilísticos o modelos basados en inteligencia artificial son usados desde hace muy poco tiempo, siendo un campo de investigación en auge como reflejan las fechas de una gran parte de los artículos referenciados.

Capítulo 3: Técnicas de Modelado

1. INTRODUCCIÓN

El desarrollo de modelos matemáticos es una aproximación ampliamente utilizada con el fin de analizar, comprender y predecir el funcionamiento de cualquier sistema en función de las condiciones que rigen su dinámica. Las herramientas y técnicas más adecuadas para el desarrollo del modelo vienen condicionadas por las características particulares de cada problema.

En este trabajo se propone la utilización de estrategias alternativas de modelado utilizando técnicas basadas en datos. Esta aproximación ha sido utilizada con buenos resultados desde hace más de dos décadas en la resolución de muchos problemas complejos, dando lugar a una disciplina conocida en la comunidad científica con el término "data mining".

Se puede definir data mining como el conjunto de técnicas y herramientas aplicadas al proceso no trivial de extraer y presentar conocimiento implícito, previamente desconocido, potencialmente útil y humanamente comprensible, a partir de grandes conjuntos de datos, con objeto de predecir, de forma automatizada, tendencias y comportamientos o descubrir modelos previamente desconocidos [Piatetski 1991].

Los orígenes del data mining se pueden establecer a principios de la década de 1980, cuando la Administración de Hacienda estadounidense desarrolló un programa de investigación para detectar fraudes en la declaración y evasión de impuestos, mediante lógica difusa, redes neuronales y técnicas de reconocimiento de patrones [DePompe 1996]. Sin embargo, la gran expansión del data mining no se produce hasta la década de 1990 originada principalmente por tres factores:

- Incremento de la potencia de los ordenadores
- Generalización del uso de ordenadores personales y de las redes corporativas
- Incremento del ritmo de adquisición de datos. El crecimiento de la cantidad de datos almacenados se ve favorecido no sólo por el abaratamiento de los discos y sistemas de almacenamiento masivo, sino también por la automatización de muchos trabajos y técnicas de recogida de datos.
- Aparición de nuevos métodos de técnicas de aprendizaje y almacenamiento de datos.

Etapa	Tecnologías	Características
Recolección de datos (Años 60)	Ordenadores, cintas, discos	Retrospectivo, datos estáticos
Acceso a los datos (Años 80)	Bases de datos relacionales, SQL, ODBC	Retrospectivo, datos dinámicos a nivel de registro
Data Warehouse y soporte a la toma de decisiones (Años 90)	OLAP, bases de datos multidimensionales, data warehouses	Retrospectivo, obtención dinámica de datos a múltiples niveles
Data Mining	Algoritmos avanzados, ordenadores multiprocesador, bases de datos masivas	Prospectivo, obtención proactiva de información

Tabla 6 Evolución histórica del data mining

En la actualidad las técnicas de data mining se utilizan en multitud de sectores, destacando especialmente su aplicación en el sector financiero, marketing y en el de las telecomunicaciones. La Fig. 4 muestra los resultados obtenidos en una encuesta web publicada en [KDNUGGETS]. La pregunta realizada fue: "¿Dónde se planea utilizar las técnicas de data mining en 2002?".

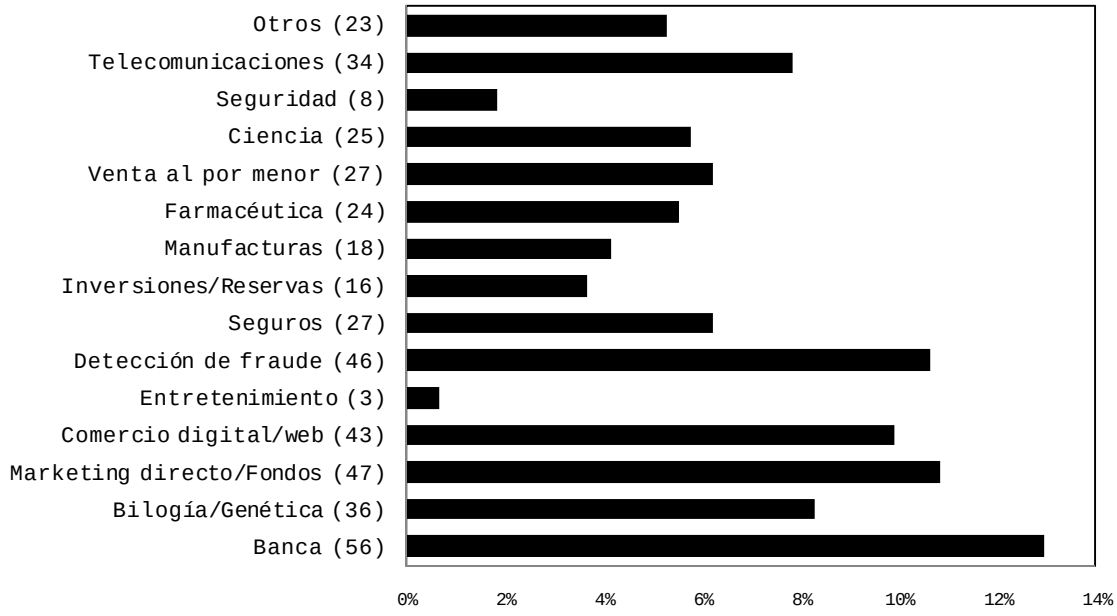


Fig. 4 Resultados de la encuesta realizada en [KDNUGGETS]

En este capítulo se describen las técnicas de data mining que, por experiencias anteriores, se consideran óptimas para ser aplicadas al proceso de predicción de riesgos en proyectos de software, así como la forma en que estas técnicas han sido combinadas de cara a integrar las ventajas de cada una de ellas y minimizar en la medida de lo posible

sus limitaciones respectivas. En ningún caso se pretende un recorrido exhaustivo de las técnicas, que puede consultarse de forma detallada en diversas referencias como [Rodríguez 2003].

2. SELECCIÓN DE LAS TÉCNICAS

Las técnicas seleccionadas en este trabajo tienen principalmente dos fines:

- realizar un preprocesado de los datos y de la información disponible, de tal forma que pudiera ser tratada de forma adecuada por las técnicas de modelado
- desarrollar modelos de tipo predictivo que permitan determinar una estimación de los riesgos en un proyecto de software.

Para realizar el preprocesado de datos, además de la utilización de técnicas de análisis de datos clásicas, tales como medidas resumen o análisis de correlaciones, ha sido necesaria la utilización de técnicas de visualización capaces de proporcionar representaciones de conjuntos multidimensionales. La dificultad inherente a visualizar datos hiper-dimensionales (superiores a tres) ha sido solventada mediante la utilización de métodos de proyección lineales y no lineales. La técnica de proyección no lineal seleccionada, **redes neuronales autoorganizadas (SOM)**, preserva en lo posible la topología de los datos multidimensionales. Es decir, si dos datos se encuentran cercanos en el espacio, sus respectivas proyecciones también se encuentran cercanas y viceversa. Esta técnica ha sido seleccionada por sus prestaciones para la visualización de espacios multidimensionales, así como por su capacidad para la agrupación de datos con características similares.

Dado que el tipo de problema abordado en esta tesis se enmarca dentro de los problemas predictivos, la característica que más se ha primado en la selección de las técnicas utilizadas para el modelado del problema ha sido su capacidad predictiva seleccionando para tal fin las **redes neuronales supervisadas perceptrón multicapa** y el **algoritmo MARS** (Multivariate Adaptive Regresion Splines) [Friedman 1991] desarrollado por Jerome Friedman en 1991, cuya eficacia ha sido sobradamente probada, tanto de forma independiente como combinadas en modelos híbridos [Rodríguez 2003].

La robustez del algoritmo MARS ante colinearidades de las variables de entrada, y su capacidad para seleccionar de forma automática variables relevantes así como interacciones entre las mismas, será utilizado para realizar la selección de variables de entrada así como para realizar un análisis de los factores más relevantes a tener en consideración para la estimación del riesgo en los proyectos de software.

A continuación se describen las técnicas mencionadas anteriormente así como las condiciones que se deben de verificar para la aplicación de las técnicas y los parámetros que deben de ser considerados durante su utilización.

3. PROYECCIONES

El método de proyecciones de Sammon [Sammon 1969] es un método de escalado multidimensional cuyo objetivo es proyectar puntos de un espacio multidimensional sobre un espacio de dimensión más baja (generalmente dos) de tal forma que las distancias mutuas se asemejen lo más posible a las del espacio original (Fig. 5)

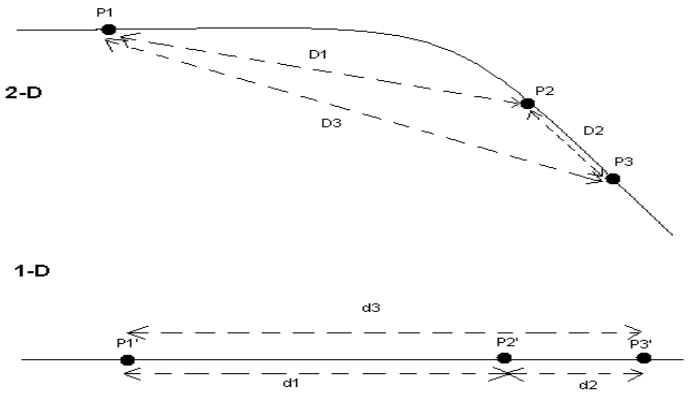


Fig. 5 Conservación de distancias cuando se proyecta un conjunto de puntos del espacio bidimensional sobre el espacio unidimensional.

El método seguido por el algoritmo de Sammon es un método iterativo que utiliza una versión del algoritmo de gradiente con un factor de paso fijo para minimizar una función de error que mide la diferencia entre la configuración de los puntos en el espacio imagen y en el espacio original. Es decir, la función de error representa hasta qué punto la configuración actual de $\boldsymbol{N}$ puntos del espacio m-dimensional coincide con los $\boldsymbol{N}$ puntos en el espacio n-dimensional. (Tabla 7).

ALGORITMO SAMMON

Paso 1: Calcular las distancias entre los puntos del espacio de entrada d_{ij}

Paso 2: Inicializar las distancias d_{ij}^{*} del espacio dimensional sobre el que se va a realizar la proyección (dimensión m=2 ó m=3) aleatoriamente

Paso 3: Calcular el error de proyección E dado por:

$$E = \frac{1}{\sum_{i<j} [d_{ij}^{*}]} \sum_{i<j}^{N} \frac{[d_{ij}^{*} - d_{ij}]^2}{d_{ij}^{*}} \qquad (3.\ 1)$$

Paso 4: Calcular el gradiente cuya dirección minimiza el error

Paso 5: Mover los puntos del espacio de dimensión menor en la dirección dada por el gradiente

Paso 6: Repetir los pasos 3 a 5 hasta que el error E es menor que una cierta tolerancia, o se realice un número máximo de iteraciones

Tabla 7 Algoritmo de Sammon

La conservación de las distancias y la topología en el algoritmo Sammon le confieren una gran capacidad para realizar tareas de agrupamiento de datos similares. Esta capacidad permite identificar datos anómalos en el espacio de partida (los *outliers* aparecen muy alejados del resto de los datos), así como detectar errores en la calidad de los datos correspondiente a falta de información relevante o errores en las medidas de los datos.

La aplicabilidad del algoritmo se ve limitada por su complejidad, que implica elevados tiempos de computación, y por su incapacidad para generalizar los resultados. Así el algoritmo de Sammon únicamente es capaz de proyectar los datos utilizados durante la construcción del algoritmo.

4. REDES NEURONALES

Una Red Neuronal Artificial es un modelo de procesamiento de la información inspirado en el sistema nervioso biológico. La pieza clave de este paradigma es la

estructura original del sistema de procesamiento, compuesta por un gran número de elementos muy simples interconectados (neuronas), cada uno con memoria local, trabajando en armonía para resolver problemas específicos.

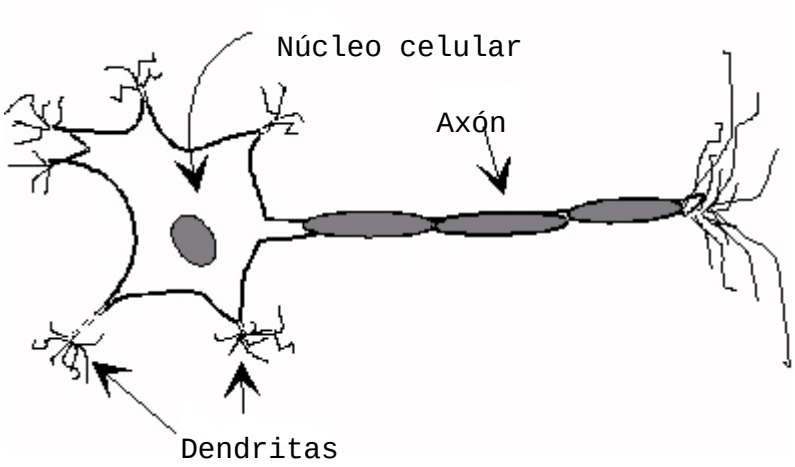


Fig. 6 Estructura básica de una neurona.

Las *dendritas* son la vía de entrada de las señales que se combinan en el cuerpo de la neurona. El *axón* es el camino de salida de la señal generada por la neurona. Las *sinapsis* son las unidades funcionales y estructurales elementales que median entre las interacciones de las neuronas.

Una neurona artificial es un elemento de procesamiento de la información que juega un papel fundamental en la red neuronal. Los tres elementos fundamentales de una neurona son:

- Un conjunto de sinapsis o conexiones, cada una de ellas caracterizada por su fuerza o peso. Así, una señal de salida x_j de la neurona j-ésima, se transmite a la neurona k-ésima que recibe como entrada $x_j w_{kj}$, donde w_{kj} es el peso o fuerza de la conexión entre ambas neuronas. De acuerdo con el signo del peso w_{kj} se tienen conexiones *excitadoras* cuando es positivo, y conexiones *inhibidoras* cuando es negativo.
- Una función de red, operador suma $\sum$, que produce la suma ponderada de todas las entradas de acuerdo a los correspondientes pesos de las conexiones.
- Una función de activación o transferencia $a(\cdot)$, que tiene como misión limitar la amplitud de la salida generada por la neurona.

Además de estos tres elementos, en el modelo de una neurona artificial es habitual incluir un *umbral o polarización* representado por θ_k, cuya misión es controlar el nivel a partir del cual la neurona produce su salida.

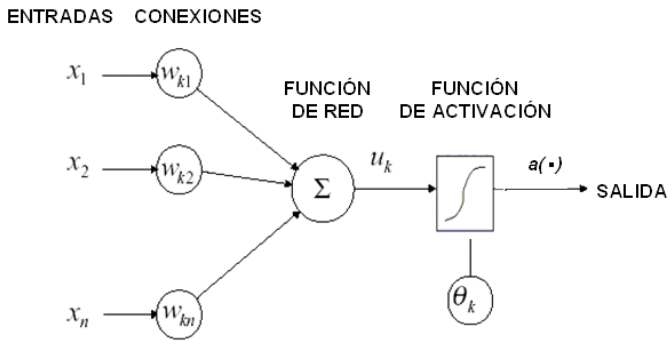


Fig. 7 Neurona artificial de McCulloch-Pitts

La función de activación representa el componente no lineal del modelo de una neurona artificial. La elección de la función de activación depende del algoritmo de aprendizaje que se vaya a utilizar, pero se pueden distinguir tres grandes clases: tipo escalón, lineales a trozos, y sigmoide, siendo esta última la función de activación más comúnmente utilizada.

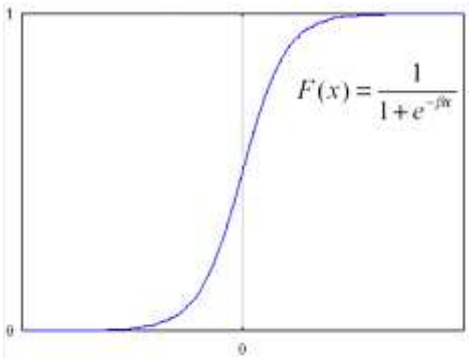


Fig. 8 Función sigmoide

La función sigmoide está en sintonía con los conocimientos fisiológicos acerca del funcionamiento de las neuronas, puesto que cuando una determinada señal nerviosa (electroquímica) no supera un cierto umbral, ésta se convierte en inhibidora y viceversa. Además la función sigmoide es una función n-diferenciable, propiedad deseable para el tratamiento numérico de los datos y la aplicación de métodos de optimización.

De todos modos, existen modelos diferentes, con mayor complejidad, que la neurona de McCulloch-Pitts, si bien todos comparten esos elementos comunes.

Evolución histórica de las Redes Neuronales		
1943	Teoría de las RN Artificiales	Walter Pitts junto a Bertran Russell y Warren McCulloch intentaron explicar el funcionamiento del cerebro humano, por medio de una red de células conectadas entre sí, para experimentar ejecutando operaciones lógicas.
1949	Conductividad de la sinopsis en las RN	El fisiólogo Donald O. Hebb expuso que las RN podían aprender
1951	Primera Red Neuronal	Minsky y Edmonds montaron la primera máquina de RN compuesta de 40 neuronas artificiales que imitaban el cerebro de una rata. La RN no tenía capacidad de retroalimentación por lo que no era capaz de trazar un plan
1956	Primera Conferencia sobre Inteligencia Artificial	Se organizó en Dartmouth donde se discutió el uso potencial de las computadoras para simular todos los aspectos del aprendizaje o cualquier otra característica de la inteligencia y se presentó la primera simulación de una RN, aunque todavía no se sabían interpretar los datos resultantes.
1959	Creación de Modelos de Adaptación (Adaline y Madaline)	Widrow publica una teoría sobre la adaptación neuronal y unos modelos inspirados en esa teoría, el Adaline (Adaptative Linear Neuron) y el Madaline (Multiple Adaline). Estos modelos fueron usados en numerosas aplicaciones y permitieron usar, por primera vez, una RN en un problema importante del mundo real: filtros adaptativos para eliminar ecos en las líneas telefónicas.
1962	Desarrollo del Perceptrón	Rosemblatt publica los resultados de un ambicioso proyecto de investigación, el desarrollo del Perceptrón, un identificador de patrones ópticos binarios, y salida binaria. Las capacidades del Perceptrón se extendieron al desarrollar la regla de aprendizaje delta, que permitía emplear señales continuas de entrada y salida.
1969	Minsky y Papert realizan una seria crítica del Perceptrón	Se revelaron serias limitaciones del perceptrón, tales como su incapacidad para representar la función XOR, debido a su naturaleza lineal. Este trabajo creó serias dudas sobre las capacidades de los modelos conexionistas y provocó una caída en picado de las investigaciones.
Años 70	Desarrollo de nuevas ideas en RN	Anderson estudia y desarrolla modelos de memorias asociativas. Destaca el autoasociador lineal conocido como modelo brain-state-in-a-box (BSB). Kohonen continua el trabajo de Anderson y desarrolla modelos de aprendizaje competitivo basados en el principio de inhibición lateral. Su principal aportación consiste en un procedimiento para conseguir que unidades físicamente adyacentes aprendieran a representar patrones de entrada similares. Grossberg realizó un importante trabajo teórico- matemático tratando de basarse en principios fisiológicos; aportó importantes innovaciones con su modelo ART (Adaptative Resonance Theory) y, junto a Cohen, elabora un importante teorema sobre la estabilidad de las RN recurrentes en términos de una función de energía.
años 80	Relanzamiento de las RN	Muchas investigaciones de la neurocomputación empezaron a ser audaces propuestas para explorar el desarrollo de neurocomputadoras y aplicaciones de RN.

		En 1983, John Hopfield, un famoso físico de reputación mundial, comenzó una interesante investigación en RN. Hopfield escribió dos grandes volúmenes de RN en 1982 y 1984, logrando persuadir a muchos físicos y matemáticos de todo el mundo a unirse a la nueva investigación de RN. En 1986 Rumelhart, McClelland and the PDP Research Group publican la aplicación del algoritmo de retropropagación del error a las RN e introducen las redes perceptrón multicapa, superando así las limitaciones de las RN señaladas por Minsky and Papert en 1969. En 1987, se realizó la primera conferencia abierta sobre RN del I.E.E.E. en San Diego, fundándose la Sociedad Internacional de RN en 1988. A principios de 1987, muchas universidades anunciaron la formación de institutos de investigación y programas de educación acerca de la neurocomputación.
Actualidad	Grandes avances y esfuerzos en el desarrollo e investigación de las redes neuronales	Las principales líneas de investigación en este campo son: La búsqueda de nuevos modelos computacionales que sintetizan el comportamiento de una neurona individual. Integración de las redes neuronales artificiales con otros modelos de procesamiento de la información: lógica difusa, algoritmos genéticos, sistemas de aprendizaje automático Formalización o identificación del proceso de desarrollo de aplicaciones basadas en redes neuronales.

Tabla 8 Tabla resumen de la evolución histórica de las redes neuronales

4.1.- CARACTERÍSTICAS DE LAS REDES NEURONALES

Hasta el momento, se han desarrollado muchos modelos de redes neuronales, pensados para una gran variedad de aplicaciones. Aunque se diferencian en su modo de activación, modelo de neurona, implementación y modo de funcionamiento, todos ellos tienen rasgos comunes, siendo sistemas de computación generados a partir de un conjunto compacto de unidades simples de procesamiento de información -neuronas artificiales- que comparten las siguientes características:

- **Codificación global.** La codificación de la información se realiza de manera global. Cada neurona representa una característica elemental y la representación global se efectúa en la red.
- **Procesamiento en paralelo**. El procesamiento de la información ocurre en paralelo, ya que cada neurona de la red actúa como una unidad de proceso independiente. Las unidades interactúan mediante señales de excitación o inhibición sobre otras neuronas, de tal forma que el estado de una neurona afecta al potencial de todas las neuronas a que está conectada, de acuerdo con los pesos de sus conexiones.

- **Memoria distribuida**. El conocimiento acumulado por la red se halla distribuido en numerosas conexiones, adaptándose los pesos sinápticos según reglas de optimización.
- **Auto-Organización y Adaptatividad**. Utilizan algoritmos de aprendizaje adaptativo y auto-organizativo.
- **Procesado No Lineal**. Las neuronas tienen funciones de activación no lineales, esto es, el nuevo estado de una neurona es una función no lineal de las señales generadas por las activaciones de otras neuronas, lo que permite captar las no linealidades inherentes al proceso que genera los datos.
- **Aprendizaje inductivo**. No se le indican las reglas para dar una solución, sino que extrae sus propias reglas a partir de los ejemplos de aprendizaje, modificando su comportamiento en función de la experiencia. Esas reglas quedan almacenadas en las conexiones y no están representadas explícitamente.
- **Generalización**. Una vez entrenada, se le pueden presentar a la red datos distintos a los usados durante el aprendizaje. La respuesta obtenida dependerá del parecido de los datos con los ejemplos de entrenamiento.
- **Abstracción o tolerancia al ruido**. Son capaces de extraer o abstraer las características esenciales de las entradas aprendidas; de esta manera, pueden procesar correctamente datos incompletos o distorsionados.

4.2.- APRENDIZAJE DE LAS REDES NEURONALES

Una de las propiedades fundamentales de las redes neuronales es la capacidad de adaptarse al entorno, aprendiendo a proporcionar la respuesta adecuada ante los estímulos que reciba de este entorno. Este aprendizaje se plasma en la modificación de los pesos de las conexiones entre los distintos elementos que forman la red. En función de la forma en que el entorno influye en el proceso de aprendizaje se distinguen dos grandes grupos de redes neuronales.

1. *Redes Supervisadas:* Durante la fase de aprendizaje, se indica a la red qué salida debe producir cada patrón, ajustando los pesos en función de ese valor.
2. *Redes No Supervisadas:* La red localiza sobre los datos de entrada unas propiedades que le sirven para la separación de los patrones en clases.

4.3.- ARQUITECTURA O TOPOLOGÍA DE LAS REDES NEURONALES

Se pueden definir las estructuras de las redes neuronales como colecciones de procesadores paralelos conectados entre sí en la forma de grafo dirigido, y organizados de tal modo que la estructura de la red sea la adecuada para el problema que se esté considerando. Cada elemento de procesamiento (o neurona) de la red se puede representar esquemáticamente como un nodo, indicando las conexiones entre nodos mediante el uso de puntas de flecha en las conexiones.

Los diferentes modos de conectar las neuronas para generar una red se denominan arquitecturas o topologías (Fig. 9). Existe una relación muy fuerte entre la arquitectura de una red neuronal artificial y los algoritmos de aprendizaje que puede usar, de tal modo que diferentes arquitecturas de redes neuronales requieren diferentes algoritmos de aprendizaje.

Una vez determinada la arquitectura a utilizar, un elemento estructural del máximo interés consiste en determinar el número de neuronas que deben ser utilizadas. Del número de neuronas depende el número de parámetros libres en el modelo y su operatividad. Si el número de neuronas es excesivamente alto en relación al problema a tratar y al número de patrones disponible, se produce un fenómeno denominado sobreajuste, mientras que si el número de unidades es excesivamente bajo se produce subentrenamiento. En el primer caso la red proporciona muy buenos resultados para el conjunto de datos de entrenamiento, pero no es capaz de generalizar los resultados para nuevos datos. En el segundo caso, los resultados generados por la red adolecen de falta de precisión.

En las redes multicapa, se debe de tener también en consideración el número de capas, así como el número de neuronas en cada capa. El número de capas está íntimamente relacionado con la velocidad de aprendizaje, por lo que comúnmente el número de capas ocultas se reduce a una o dos. Hasta el momento no hay un criterio establecido para determinar la configuración de la red, resultando la intuición y experiencia fundamentales en el desarrollo de un sistema neuronal que resuelva con eficacia una aplicación práctica determinada.

En este trabajo se propone la utilización de técnicas adaptativas como herramienta de apoyo para determinar posibles topologías de redes neuronales multicapa.

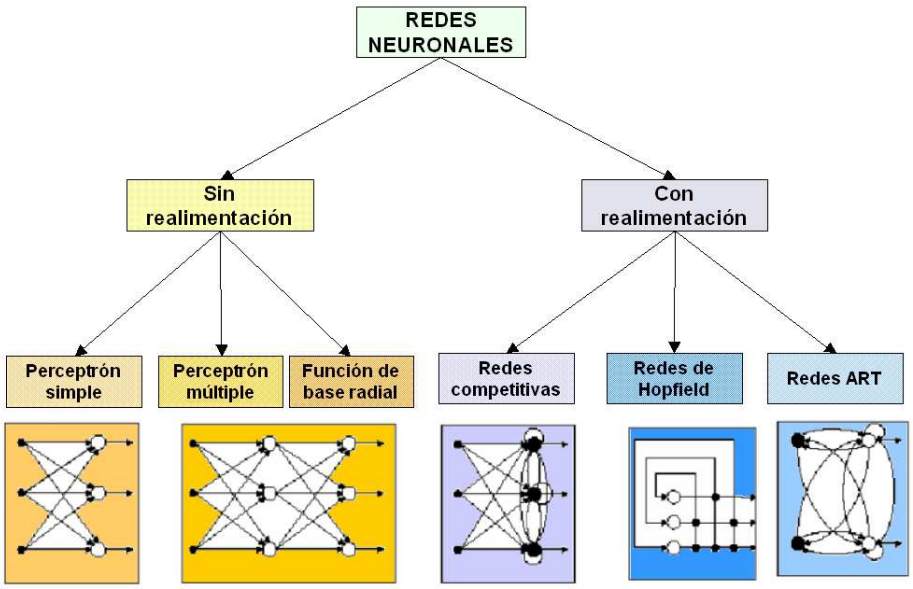


Fig. 9 Ejemplos de diferentes tipos de arquitecturas de redes neuronales.

Generalmente, una red neuronal está definida no sólo por su arquitectura, sino también por el tipo de neuronas usadas, la regla de aprendizaje o de entrenamiento, y su forma de operación. La variedad en cuanto a topologías, funciones de activación, etc., hace que la simple descripción de las distintas redes neuronales se convierta en un tema demasiado amplio como para abordarlo aquí, por lo que se presentarán únicamente los modelos de redes neuronales utilizados durante el desarrollo de este trabajo: El Perceptrón Multicapa y los Mapas Autoorganizados.

4.4.- REDES PROGRESIVAS. EL PERCEPTRÓN MULTICAPA

El paradigma más conocido dentro de las redes de propagación hacia adelante es el Perceptrón Multicapa ("Multilayer Perceptron" o MLP). El Perceptrón Multicapa es un paradigma de una red supervisada, es decir, el aprendizaje se realiza presentando a la red pares de datos de la forma *(entrada, salida_deseada)*, y el ajuste de los parámetros se realiza por el método del descenso del gradiente, a través del algoritmo de Retropropagación del Error.

Un sólo Perceptrón tiene muchas limitaciones, siendo la más importante, su incapacidad de distinguir clases que no sean linealmente separables. Si se organizan un conjunto de perceptrones en capas, esta limitación queda superada. Cada capa consta de varias neuronas cuya entrada proviene de las unidades de la capa anterior y cuya salida va a las unidades de la capa posterior, sin ningún contacto con cualquier otra capa, ni entre las

neuronas de una misma capa. La primera capa, denominada capa de entrada, recoge la señal de entrada y no realiza sobre ésta ningún tipo de procesamiento. El resultado de la red viene dado por el estado de las neuronas de la última capa, denominada capa de salida. Las capas intermedias se denominan ocultas.

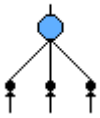
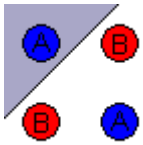
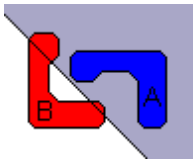
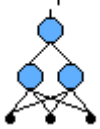
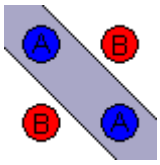
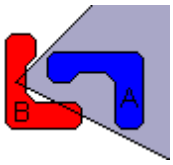
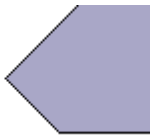
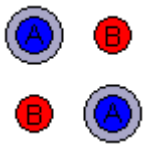
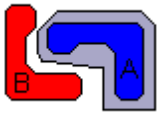
Estructura	**Regiones de Decisión**	**Problema de la XOR**	**Clases con Regiones Mezcladas**	***Formas de Regiones más Generales***
1 Capa 	Medio Plano Limitado por un Hiperplano			
2 Capas 	Regiones Cerradas o Convexas			
3 Capas 	*Complejidad Arbitraria Limitada por el Número de Neuronas*			

Fig. 10 Distintas formas de las regiones generadas por un Perceptrón multicapa

Uno de los ejes centrales de las investigaciones en el campo de las redes neuronales es cómo ajustar los pesos de los enlaces para obtener el comportamiento deseado del sistema. Esta modificación está basada a menudo en la regla de Hebb [Hebb 1949], la cual indica que el enlace entre dos unidades debe ser reforzado si ambas unidades se activan al mismo tiempo.

El entrenamiento de una red progresiva con aprendizaje supervisado consta de los siguientes pasos.

1. Primero se presenta un patrón a la capa de entrada que se propaga por la red hasta que la activación alcanza las neuronas de la capa de salida (fase de propagación).
2. Posteriormente la salida real se compara con el patrón de aprendizaje, y el error, junto con la salida de las neuronas de la capa precedente, se utiliza para modificar adecuadamente los pesos de las conexiones en la última capa.

3. Por último, los errores se van propagando hacia la capa inicial, sirviendo para modificar los pesos de las conexiones (fase de actualización) en el resto de las capas.

En general, se emplean redes neuronales multicapa con aprendizaje por retropropagación del error para simular una función desconocida a partir de los pares de señales de entrada y salida obtenidos de los patrones de aprendizaje [Sontag 1992]. Se espera en estos casos que la red sea capaz de generalizar adecuadamente, de forma que ante señales de entrada que no hayan sido presentadas previamente como patrones de aprendizaje se obtenga la salida adecuada.

4.4.1.- Redes de Retropropagación (Backpropagation)

El algoritmo de aprendizaje de **retropropagación del error** (también conocido como la *regla delta generalizada*) es una aplicación del método del descenso del gradiente.

Se establece como función objetivo a minimizar el error cuadrático:

$$E(t) = \frac{1}{2} \sum_{i \in CapadeSalida} \left(o_i(t) - \hat{o}_i(t) \right)^2 \qquad (3.\ 2)$$

donde:

$o_i(t)$ es la salida deseada para el elemento de proceso i en el instante t.

$\hat{o}_i(t)$ es la salida del elemento de proceso i en el instante t.

De acuerdo con esta regla de aprendizaje, el peso w_{ij} que conecta el elemento de proceso i con el j se ajusta por:

$$\Delta w_{ij} = \eta \delta_j(t) \bar{o}_i(t) \qquad (3.\ 3)$$

donde:

η es el denominado *coeficiente de aprendizaje*.

$\delta_j(t)$ es la señal de error del elemento de proceso j en el instante t

Si el elemento de proceso j es de la capa de salida se tiene que:

$$\delta_j(t) = \left(o_j(t) - \hat{o}_j(t)\right) a'_j(t) \qquad (3.\ 4)$$

donde $a'_j(t)$ es la derivada de la función de activación o transferencia en el instante t para el elemento de proceso j.

Si el elemento de proceso j no se encuentra en la capa de salida, sino que está en las capas ocultas, la señal de error tiene el siguiente valor:

$$\delta_j(t) = a'_j(t) \sum_{k=1}^{N} \delta_k(t) w_{kj} \qquad (3.\ 5)$$

donde N es el número de elementos de proceso en la capa destino.

En ocasiones esta fórmula se corrige para acelerar la convergencia introduciendo un término denominado *momento*. Los pesos se actualizan de acuerdo a la regla precedente, y se evalúa el error. Si el error decrece, el cambio se reitera hasta que el error comienza a aumentar. En ese momento se evalúa el nuevo gradiente y continúa el aprendizaje. El término del momento introduce el cambio de los pesos antiguos como parámetro en el cálculo del cambio en los pesos actuales. Esto evita los problemas habituales de oscilación cuando la superficie de error tiene un área mínima muy reducida. El nuevo cambio de pesos será:

$$\Delta w_{ij}(t) = \eta \delta_j(t) \hat{o}_i(t) + \alpha \Delta w_{ij}(t-1) \qquad (3.\ 6)$$

donde α es una constante comprendida entre 0 y 1, que indica la influencia del momento.

4.4.2.- Consideraciones generales sobre las redes perceptrón multicapa

Una red neuronal funciona mediante un sistema de aprendizaje a través de ejemplos. Debido a este mecanismo es posible distinguir dos partes diferentes dentro del proceso de funcionamiento de una red neuronal:

1. Periodo de entrenamiento o aprendizaje, mediante el que son presentados los ejemplos a la red, permitiendo el reajuste de sus pesos hasta que se obtiene un mínimo de error frente a los ejemplos presentados.

2. Periodo de prueba o test. Una vez realizado el aprendizaje, el mecanismo de analizar la efectividad del mismo consiste en la presentación de los patrones de prueba y analizar las respuestas de la red para cada patrón individual.

El procedimiento usual para llevar a cabo los apartados anteriores consiste en separar, en primer lugar, el conjunto de patrones en dos subconjuntos disjuntos: un grupo se utilizará como conjunto de entrenamiento, y el otro grupo como conjunto de prueba.

Una vez comprobada la efectividad en el aprendizaje, el procedimiento final será poner en explotación la red resultante con el conjunto de pesos que definen sus conexiones.

En la aplicación práctica de las redes neuronales artificiales perceptrón multicapa entrenadas con el algoritmo de retropropagación del error, se debe de tener en cuenta que el aprendizaje puede ser demasiado lento debido a la posibilidad de alcanzar un mínimo local en la superficie función del error, o bien a que, en la zona en la que se esté evaluando dicha función, la pendiente sea muy escasa (superficie de error casi plana). Este problema se puede controlar mediante la elección adecuada de los parámetros η (coeficiente de aprendizaje) y α (momento).

La elección de estos parámetros depende de cada problema específico, pudiendo presentarse la situación de que los mejores valores de estos parámetros al principio del entrenamiento no lo sean al final. Una alternativa para el ajuste del coeficiente de velocidad de aprendizaje consiste en realizar el ajuste de forma progresiva al aprendizaje mediante la comprobación de si con un valor particular de η se ha producido un incremento o decremento del error de red en la capa de salida, realizando el ajuste del parámetro mediante la regla dada por Cater [Cater 1984]:

$$\Delta\eta = \begin{cases} +a & \text{si } \Delta E < 0 \\ -b & \text{si } \Delta E > 0 \\ 0 & \text{en otro caso} \end{cases} \qquad (3.7)$$

Por su parte el parámetro α del momento es un valor comprendido entre 0 y 1. Habitualmente, el entrenamiento se comienza con valores altos del parámetro (0.6, 0.9) para evitar caer en mínimos locales, disminuyendo progresivamente el valor del momento a medida que aumenta el número de ciclos de entrenamiento.

Otra cuestión a tener en cuenta durante el entrenamiento de la red es el problema del *sobreentrenamiento*. En general, cuando el sistema (la red neuronal) es demasiado grande, el número de parámetros a ajustar durante el entrenamiento es también muy elevado. Puede que el sistema no solamente modele la relación entre entrada y salida deseada (sesgo muy pequeño), sino que también modele el ruido que se encuentra mezclado con la señal, no siendo capaz el sistema de generalizar los resultados obtenidos para nuevos casos.

Las alternativas propuestas para intentar solventar el problema de sobreentrenamiento son:

- Detener el proceso de aprendizaje en un número bajo de iteraciones, con lo que no se llega a producir el sobreajuste de los parámetros.
- Reducir el tamaño de la red mediante poda, con lo que se reduce el número de parámetros involucrados en el sistema.
- Utilizar un método incremental, aumentando el número de parámetros (elementos de proceso y conexiones) desde un nivel bajo, hasta alcanzar el grado de respuesta óptimo requerido.
- Utilizar técnicas consistentes en definir la actualización de los pesos de las conexiones entre los distintos elementos de proceso imponiendo la condición adicional de que sus correspondientes valores absolutos sean lo más pequeños posibles. De esta forma se deja tender los pesos poco a poco a cero, para que aquellos que no sean actualizados periódicamente se anulen y desaparezcan.

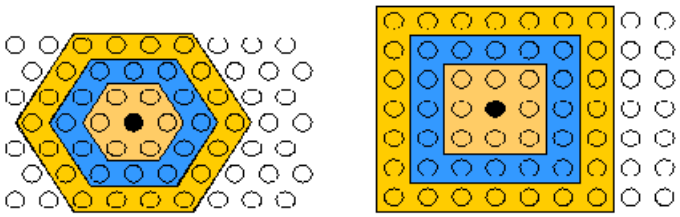
4.5.- MAPAS AUTOORGANIZADOS (SOM)

Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta; por ello, suele decirse que estas redes son capaces de *organizarse*.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

En el caso de las redes *SOM*, lo que se realiza es una identificación de características, obteniéndose en las neuronas de salida una disposición geométrica que representa un mapa topográfico de las características de los datos de entrada. De este modo si se presentan a la red informaciones similares, siempre se verán afectadas neuronas de salida próximas entre sí en la misma zona del mapa.

El aspecto geométrico de la disposición de las neuronas de una red suele ser de forma hexagonal o rectangular, formando mapas topológicos en los que se encuentran representadas las características de los datos de entrada presentados a la red, de forma que si la red recibe informaciones con características similares se generan mapas parecidos (Fig. 11).



a) b)

Fig. 11 Neuronas vecinas (nivel 1, 2 y 3) de las neuronas marcadas en negro:

Cada neurona *i* de la red SOM se representa por un vector prototipo de referencia $\boldsymbol{m}_i = [\boldsymbol{m}_{i1},...,\boldsymbol{m}_{in}]^t$, cuya dimensión coincide con la dimensión del espacio de vectores de entrada. Cada una de las neuronas se encuentra conectada a las neuronas vecinas, estableciendo diferentes niveles de vecindad. El número de neuronas utilizadas en el mapa afecta a la precisión y capacidad de generalización de la red SOM.

El aprendizaje de las redes SOM es un aprendizaje competitivo y cooperativo, de tal forma que las neuronas compiten y cooperan unas con otras para llevar a cabo una tarea dada. La competición entre neuronas se realiza en todas las capas de la red, existiendo entre las neuronas vecinas conexiones recurrentes de autoexcitación y conexiones de inhibición o de excitación. El objetivo de este tipo de aprendizaje consiste en lograr que ante cierta información de entrada, se active sólo una de las neuronas de la red, denominada *neurona ganadora*, quedando anuladas las restantes que son forzadas a sus valores mínimos de respuesta. De esta forma se logra que los datos de entrada con

informaciones similares activen la misma neurona ganadora (o neuronas cercanas a la neurona ganadora), y queden por lo tanto clasificados dentro de la misma categoría.

4.5.1.- Inicialización de las redes SOM

En la fase de inicialización se debe de seleccionar el tipo de mapa topológico que se utilizará para el entrenamiento de la red. Usualmente se utilizan mapas topológicos bidimensionales (hexagonales o rectangulares) que facilitan la visualización de los resultados, pero se pueden utilizar mapas de cualquier dimensión.

Una vez fijado el tipo de mapa a utilizar, se debe de determinar el número de neuronas y la distribución de las neuronas sobre el mapa. El número de neuronas debe de ser suficientemente grande como para ser capaz de extraer las características de los datos de entrada. La regla empírica (3. 8) puede ser utilizada como referencia para, en función del número de patrones disponibles para el entrenamiento de la red, establecer el número de neuronas del mapa topológico [Kohonen 1996].

$$\boldsymbol{N^{o}\ neuronas} = 5\,\boldsymbol{N}^{0.54321} \qquad (3.\ 8)$$

Así mismo en [Kohonen 1996], se propone distribuir las neuronas en el mapa topológico de tal forma que se mantengan las proporciones relativas al cociente de los dos autovalores mayores asociados a la matriz de datos de entrada. Es decir, si distribuimos las neuronas en *a* filas y *b* columnas, el tamaño del mapa debe verificar las condiciones (3. 9) y (3. 10).

$$\frac{\boldsymbol{a}}{\boldsymbol{b}} = \frac{\boldsymbol{\lambda}_1}{\boldsymbol{\lambda}_2} \qquad (3.\ 9)$$

$$\boldsymbol{a} \times \boldsymbol{b} \cong \boldsymbol{N^{o}\ neuronas} \qquad (3.\ 10)$$

Antes de proceder al entrenamiento de la red SOM, los vectores prototipo se inicializan siguiendo alguno de los siguientes procedimientos:

Inicialización aleatoria: los vectores prototipo iniciales se corresponden con valores pequeños generados de forma aleatoria

- *Inicialización lineal*: el método de inicialización lineal se basa en el análisis de componentes principales (PCA). Los vectores prototipo se inicializan de tal

forma que pertenezcan al mismo subespacio lineal que el generado por los dos vectores propios asociados a las dos primeras componentes principales. Esta inicialización tiene el efecto de dirigir el mapa SOM en la misma dirección que los datos que presentan las variaciones más significativas.

- *Inicialización muestral*: los vectores propios se inicializan seleccionando de forma aleatoria vectores del espacio de datos de entrada.

Aunque la red SOM es robusta respecto a los valores iniciales de partida, una elección adecuada en los vectores iniciales prototipo agiliza la convergencia hacia una buena solución.

4.5.2.- Entrenamiento

Cada elemento de proceso $\boldsymbol{i}$ o neurona que forma el mapa está definido en el instante $\boldsymbol{t}$ por un vector prototipo $\boldsymbol{m_i(t)}$. El algoritmo de aprendizaje o ajuste de los pesos hace que para cada entrada $\vec{\boldsymbol{x}}_{\boldsymbol{i}}(\boldsymbol{t})$ en el instante $\boldsymbol{t}$, se siga el siguiente proceso:

1. Determinar el elemento de proceso *ganador* (*neurona ganadora*), en el sentido de mínima distancia:

$$\boldsymbol{g} = \min_{\boldsymbol{i}}\left(\boldsymbol{d}\left(\boldsymbol{m_i(t)}, \vec{\boldsymbol{x}}_{\boldsymbol{i}}(\boldsymbol{t})\right)\right) \qquad (3.\ 11)$$

2. Adaptar los pesos de la neurona ganadora y de sus vecinos (Fig. 12) de acuerdo con la fórmula:

$$\boldsymbol{m_j(t+1)} = \boldsymbol{m_j(t)} + \boldsymbol{h(i,g,t)}\left(\vec{\boldsymbol{x}}(\boldsymbol{t}) - \boldsymbol{m_j(t)}\right) \quad \forall \mathrm{i} \in \mathrm{S} \qquad (3.\ 12)$$

donde $\boldsymbol{h(i,g,t)}$ depende de la posición del elemento de proceso $\boldsymbol{j}$ cuyos pesos se van a actualizar, del elemento de proceso ganador $\boldsymbol{g}$ y del tiempo $\boldsymbol{t}$, de tal forma que, normalmente el valor de $\boldsymbol{h}(\ )$ decrece con el tiempo.

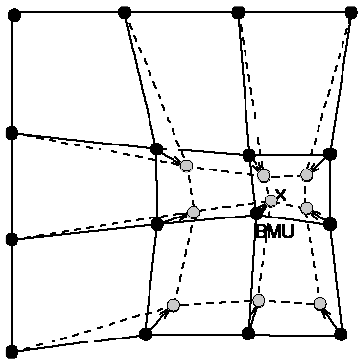
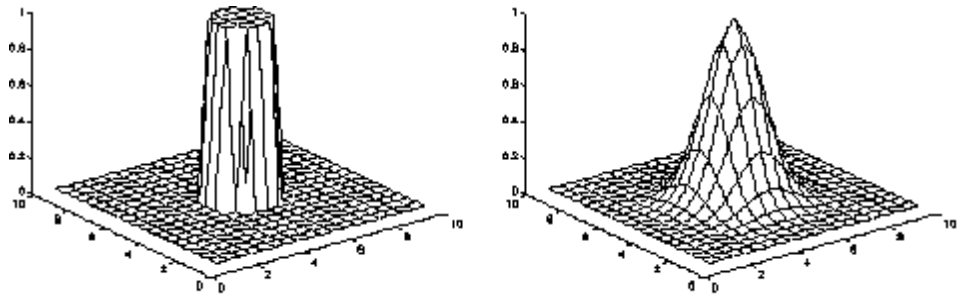


Fig. 12 Adaptación de la neurona ganadora (BMU) y de sus vecinos hacia el vector de entrada (x).

La función $h(\ )$ define la región de influencia que el vector de entrada tiene respecto al SOM, proporcionando un ajuste positivo (refuerzo) en las proximidades de la neurona ganadora, que va decreciendo hasta volverse nulo en los nodos más lejanos.



a) b)

Fig. 13 Ejemplo de dos tipos de función de influencia de la neurona ganadora respecto a las neuronas vecinas: a) Función burbuja b) Función normal

El entrenamiento de la red SOM se realiza habitualmente en dos fases: una primera fase en la que se realiza un entrenamiento grosero aplicando una gran influencia relativa de la neurona ganadora sobre las neuronas vecinas, y una fase de afino en la que la influencia de la neurona ganadora sobre sus vecinas es menor.

4.5.3.- Consideraciones generales sobre las redes SOM. Visualización

Las redes SOM buscan vectores de referencia que permitan agrupar datos similares y los sitúan simultáneamente sobre un mapa regular, combinando así la propiedad de cuantificación de los vectores con la propiedad de conservación de la ordenación topológica. Además las redes SOM presentan una relativa robustez respecto a errores en el espacio de datos de entrada, de tal forma que la presencia de *outliers* tiene únicamente un efecto local. La presencia de un *outlier* en el espacio de entrada se puede detectar con

redes SOM, por su localización en el mapa topológico sobre una neurona aislada del resto de neuronas vecinas.

Una de las propiedades más interesantes que presentan las redes SOM, y el motivo por el que han sido seleccionadas en este trabajo, es su capacidad de visualización de datos multidimensionales.

La relación entre las variables de entrada puede visualizarse mediante la representación de las componentes planas de los vectores prototipo. Para ello, las variables se representan identificando en cada vector prototipo, el valor de la componente correspondiente a la variable que se desea visualizar y asignándole un color sobre una determinada escala cromática (Fig. 14).

De esta forma se puede visualizar la distribución de las variables de entrada y encontrar relaciones asociadas a los diferentes grupos existentes dentro de las variables [Kohonen 1996].

La representación de las componentes planas, suele representarse de forma simultánea a la representación de la matriz de distancias entre las neuronas (*u-matrix*) permitiendo visualizar la estructura de agrupamientos de los mapas.

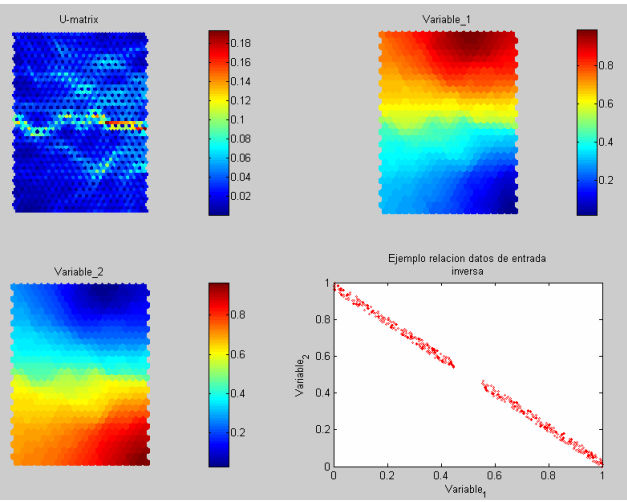


Fig. 14 Ejemplo de representación de mapas de componentes planos. Se introduce una relación inversa lineal discontinua, pudiéndose observar en la matriz de distancias la presencia de dos grupos, cada uno de ellos con relación inversa entre sí.

5. ALGORITMO MARS

El modelado de procesos a partir de un conjunto de muestras de la relación entrada/salida puede ser considerado como un problema de aproximación funcional, en el que se trata de hallar una expresión matemática,

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3, ..., \boldsymbol{X}_n) + \boldsymbol{\varepsilon}, \quad (\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_n) \in \boldsymbol{D} \subset \Re^n \tag{3. 13}$$

que sea capaz de reproducir la relación inyectiva que existe entre las entradas y las salidas del proceso $\left\{\boldsymbol{y}_j, \boldsymbol{x}_{ij}\right\}_{\substack{i=1,n \\ j=1,N}}$. (La cantidad $\boldsymbol{\varepsilon}$ representa un ruido estocástico que, de no ser nulo, refleja el hecho de que $\boldsymbol{y}$ depende de más variables, usualmente aleatorias, que las consideradas).

La computación adaptativa es aquella que ajusta dinámicamente su estrategia para tener en cuenta el comportamiento del problema específico que intenta resolver. Uno de los métodos adaptativos unidimensionales más comúnmente utilizados son las interpolaciones polinomiales a trozos.

La idea básica de la interpolación polinomial a trozos es aproximar $\boldsymbol{f}$ mediante varios polinomios de orden bajo, cada uno definido sobre una subregión (o intervalo) distinto del dominio $\boldsymbol{D}$. La aproximación obtenida es continua y, en función del orden de los polinomios, mantiene también continuas sus derivadas. El compromiso entre suavidad y flexibilidad de la aproximación se controla mediante el número de subregiones (nodos) y el orden de las derivadas más bajas que se permite sean discontinuas en los límites de las subregiones. Las funciones polinomiales segmentarias más usuales son los splines [DeBoot 1978]. Esta técnica ha proporcionado formas eficientes de resolver el problema de aproximación funcional en dimensiones bajas($n \leq 2$).

La generalización de esta técnica a espacios de mayor dimensión es directa en la teoría, pero difícil en la práctica. Las dificultades vienen dadas por el hecho de que para poder poblar de forma densa los espacios euclídeos, se necesita un número de puntos que crece exponencialmente con la dimensión del espacio. Este problema se denomina *maldición de la dimensionalidad*.

La maldición de la dimensionalidad es básica y no puede vencerse de forma directa, siendo necesaria la adaptación de los métodos de aproximación unidimensionales a

técnicas no paramétricas. Uno de los representantes de estas técnicas más extendido en la actualidad es el algoritmo MARS (*Multivariate Adaptive Regresión Spline*).

El algoritmo MARS [Friedman 1991] desarrollado por J.H. Friedman en 1991, es un método adaptativo de ajuste multidimensional. El objetivo del algoritmo consiste en aproximar una función desconocida $f : D \subseteq \Re^n \rightarrow D' \subseteq \Re$ a partir de la proyección de una muestra de datos del espacio de entrada sobre el espacio de salida. Para ello el algoritmo MARS busca la función de aproximación $\hat{f}(x_1, x_2, ...x_n)$, mediante la combinación lineal de un conjunto de funciones base $B_i(x)$ parametrizadas por la posición de los nodos:

$$\hat{f}(x_1, x_2, ..., x_n) = \sum_{i=1}^{M} a_i B_i(x) \qquad (3.\ 14)$$

donde:

$x \in D' \subset \Re^m$, donde $1 \leq m \leq n$ es el número de variables que intervienen en la construcción de la función base B_i

$a_i, i = 1, ..., M$ son los coeficientes de las funciones base B_i

M es el número de funciones base del modelo.

Las funciones base utilizadas por el MARS, se construyen como productos de funciones base del tipo:

$$b_k(x) = (x - t_k)_+ = Max(0, x - t_k) \qquad (3.\ 15)$$

donde t_k es la posición del nodo k y el subíndice $(\cdot)_+$ denota la parte positiva de $(\cdot)$

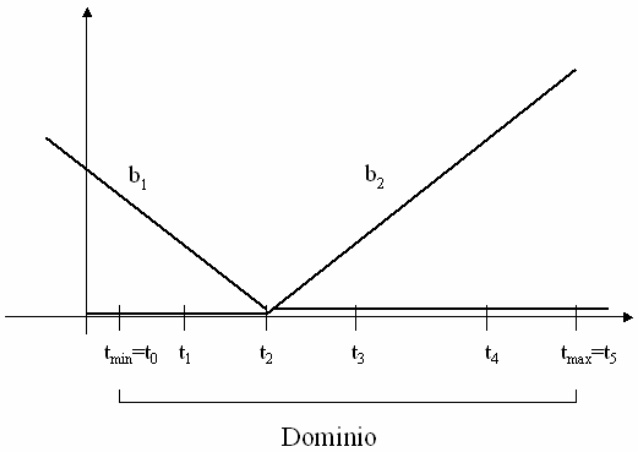


Fig. 15 Funciones base asociadas a un nodo

El número máximo de funciones base que intervienen en la construcción (número de productos), se lo denomina número de interacciones entre las variables. El valor de este parámetro lo fija el usuario, y se mantiene fijo durante el entrenamiento del algoritmo.

Así las funciones base utilizadas por el algoritmo MARS tienen la forma:

$$B_i(x) = \begin{cases} 1, & i = 1 \\ \prod_{j=1}^{J_i} \left[s_{ji} \cdot (x_{v(j,i)} - t_{ji}) \right]_+ & i = 2,3,... \end{cases} \qquad (3.16)$$

donde:

$(\cdot)_+ = max(0,\cdot)$ es decir, la parte positiva del interior del paréntesis

J_i es el grado de interacción de las funciones base B_i

$s_{ji} = \pm 1$ es el indicador del signo

v(j,i) es el índice de la variable independiente que está siendo dividida, restringiéndose a que $v(j,\cdot), j = 1, J$ sea distinto, es decir, cada variable independiente aparece únicamente una vez en cada término de interacción.

t_{ji} da la posición de la división y se denomina nodo.

Para ilustrar la notación empleada se mostrará un ejemplo. Supóngase que un modelo MARS contiene la función base B_i dada por:

$$B_i = (x_2 - 1.3)_+ \left[-(x_5 + 33.7) \right]_+$$

De forma inmediata se ve que existen dos factores en la interacción de términos, por lo tanto J_i=2. Los indicadores de signo son $s_{1,I}=1$, $s_{2,I}=-1$ y los nodos vienen dados por $t_{1,I}=1.3$, $t_{2,I}=-3.7$. Por último las etiquetas para las variables vienen dadas por $v(1,I)=2, v(2,I)=5$.

La construcción del modelo MARS para el ajuste de los datos se realiza en dos fases.

En la **primera fase**, se sitúan los nodos del recubrimiento mediante un proceso iterativo de $K_{máx}$ pasos. En cada paso, se introducen dos nuevas funciones base en el modelo del tipo:

$$\boldsymbol{B_m(x)}\left[+\left(\boldsymbol{x_v}-\boldsymbol{t}\right)\right]_+ \qquad (3.\ 17)$$

$$\boldsymbol{B_m(x)}\left[-\left(\boldsymbol{x_v}-\boldsymbol{t}\right)\right]_+ \qquad (3.\ 18)$$

donde:

$\boldsymbol{B_m(x)}$ es una función base introducida previamente en el modelo en alguno de los pasos previos, en la que intervienen como mucho $mi-1$ variables, siendo mi el número máximo de interacciones entre variables fijado por el usuario.

$\boldsymbol{x_v}$ es una de las variables de entrada, no considerada previamente en la función $\boldsymbol{B_m(x)}$. Es decir para la función $\boldsymbol{B_i}=(\boldsymbol{x}_2-1.3)_+\left[-(\boldsymbol{x}_5+33.7)\right]_+$ del ejemplo anterior, las variables $\boldsymbol{x_v}$ tienen que ser distintas de la variable número 2 y 5.

$\boldsymbol{t}$ es un nodo correspondiente a la variable de entrada $\boldsymbol{x_v}$. El valor de $\boldsymbol{t}$ se selecciona de entre el conjunto de valores distintos que toma la variable $\boldsymbol{x_v}$ para el conjunto de datos de entrenamiento.

La selección de las dos nuevas funciones base que entran en el modelo, se realiza mediante un proceso de búsqueda exhaustiva, seleccionando el par de funciones base que proporcionan un mejor ajuste.

En la **segunda fase**, se procede a eliminar de una en una las funciones base, hasta lograr que el criterio de convergencia alcance un mínimo. En cada paso se elimina la función base cuya eliminación mejora el grado de ajuste o causa la menor pérdida de

información, es decir, se utiliza un criterio de penalización del número de funciones base y del grado de convergencia.

La función de aproximación construida mediante este método es una función continua en ***D***. El modelo puede ser extendido a un modelo con derivada primera continua, reemplazando los splines lineales, por splines cúbicos en las funciones base, cuyos nodos de control vienen dados por el nodo de la función base lineal y los puntos que se encuentran situados a la mitad de la distancia del nodo situado por dicha función base, y los dos nodos adyacentes sobre la proyección de la misma componente.

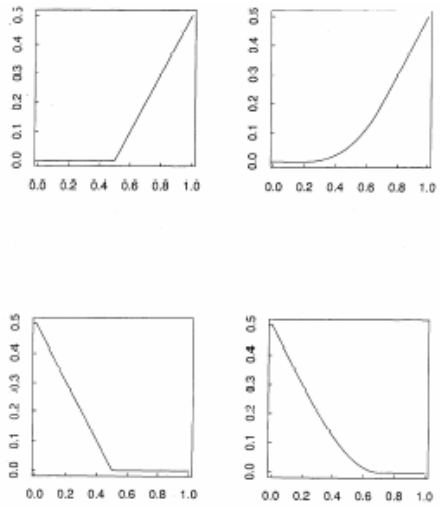


Fig. 16 Nodo de la función de base truncada lineal 0.5. Nodos de la función base spline cúbico 0.2, 0.5 y 0.7

5.1.- CONSIDERACIONES GENERALES SOBRE EL ALGORITMO MARS

Como se ha desarrollado en los apartados anteriores el algoritmo MARS utiliza una estrategia de búsqueda exhaustiva para determinar la posición de los nodos en la descomposición adaptativa. La aplicación de esta estrategia supone la resolución de aproximadamente $\boldsymbol{Nº\ Variables \cdot N}$ ajustes de mínimos cuadrados en la selección de cada nuevo nodo introducido en el recubrimiento. Si la implementación del algoritmo de mínimos cuadrados se hiciera de forma directa, la ejecución del algoritmo, para problemas relativamente grandes, sería inviable desde un punto de vista computacional.

Sin embargo la utilización de funciones base parametrizadas por un único nodo permite el desarrollo de fórmulas adaptativas relativamente simples, que permiten realizar los cálculos para la selección de los nodos de una forma muy rápida y eficiente, siendo el

coste computacional del algoritmo del orden de *KN*, con *K* función que depende del número de funciones base e interacciones entre variables utilizadas.

La base para el desarrollo de estas fórmulas se sustenta en que la matriz del sistema asociado al método de mínimos cuadrados puede definirse de tal forma que al variar la posición del nodo ***t***, dentro del conjunto de candidatos, la matriz que define el sistema permanece fija excepto para la última fila y la última columna. Además si los nodos de cada variable se encuentran ordenados, a partir de la construcción de la fila y columna asociada a uno de los nodos (el mayor o el menor) se pueden calcular las filas y columnas asociadas al resto de los nodos. Para la resolución de los sistemas planteados se utilizan las propiedades adaptativas de la descomposición de Cholesky.

La utilización de las fórmulas adaptativas hace que el algoritmo MARS, en contra de lo que inicialmente se podría suponer, sea un algoritmo con un bajo coste computacional, entrenando cientos de veces más rápido que las redes neuronales [DeVeaux 1993]. Además el algoritmo MARS selecciona de forma automática las variables relevantes e interacciones entre dichas variables. La aproximación a la solución mediante descomposición en funciones base facilita la interpretabilidad de los resultados.

En 1993 Richard DeVeux [DeVeaux 1993], realizó un estudio comparativo de la capacidad predictiva de las redes neuronales y el algoritmo MARS obteniendo como resultado que en numerosas ocasiones la capacidad predictiva del algoritmo MARS era superior a la capacidad predictiva de las redes neuronales. En este trabajo se considerarán ambas técnicas para la construcción de los modelos seleccionando finalmente la que proporcione mejores resultados.

En el siguiente subapartado se describen los parámetros utilizados por la herramienta APIMARS, desarrollada por el Grupo de Investigación en Proyectos de Ingeniería de la Universidad de Oviedo e incluye la implementación en Fortran y C de una versión modificada del algoritmo MARS.

5.1.1.- Descripción de los parámetros utilizados por el algoritmo APIMARS

Nº funciones base (FB) → Es el número máximo de funciones base utilizadas en el paso hacia delante del algoritmo. La influencia de este parámetro sobre los resultados está relacionada con la capacidad de generalización del algoritmo. En general este parámetro debe de ser suficientemente grande, para ser capaz el modelo de adaptarse a los datos,

pero no debe de ser muy grande, para evitar problemas de sobreentrenamiento. Dado que tras el paso hacia delante, para la generalización de los resultados se realiza el paso hacia atrás, la influencia de este parámetro en las aplicaciones prácticas es baja.

Nº interacciones entre variables → Es el número máximo de variables que intervienen en las funciones base. La influencia de este parámetro es similar a la influencia del grado de los polinomios utilizados en la interpolación polinomial a trozos unidimensional. Es decir, es preferible mantener el parámetro con valores tan bajos como sea posible, utilizando si es necesario un mayor número de funciones base.

% datos utilizados → Es el porcentaje de datos del conjunto total que será utilizado para crear el conjunto de patrones de entrenamiento. Si el parámetro toma el valor 100 todos los datos serán utilizados para el entrenamiento, coincidiendo la versión modificada del algoritmo MARS con el algoritmo MARS. Si el parámetro toma el valor 0 todos los datos son utilizados para validación de un modelo generado previamente.

Nº modelos generados → Es el número de iteraciones del paso hacia delante y hacia atrás, con diferentes conjuntos de entrenamiento y de prueba en cada iteración, que serán realizadas.

6. CONCLUSIONES

La estrategia de modelado propuesta en esta tesis se basa en la utilización de técnicas basadas en datos. La gran cantidad de técnicas basadas en datos existentes y la multitud de algoritmos asociados a estas técnicas, hace que la simple mención de cada una de ellas se salga de los límites de este trabajo, por lo que este capítulo se ha centrado únicamente en la descripción de las técnicas que van a ser utilizadas durante el desarrollo del trabajo.

La descripción de las técnicas ha ido enfocada a obtener un mejor conocimiento de su funcionamiento, señalando sus ventajas y limitaciones, de tal forma que su aplicación pueda realizarse de forma eficaz. Además se han intentado delimitar los parámetros que afectan a cada técnica para facilitar su configuración en la aplicación a casos concretos, e intentando disminuir de esta forma el número de pruebas necesarias.

Las técnicas seleccionadas para el desarrollo del trabajo han sido destinadas a realizar el preproceso de los datos y su posterior modelado. Las redes SOM permiten realizar

clustering de los datos de entrada así como visualizar datos multidimensionales y son utilizados básicamente para búsqueda de analogías o en el preproceso. Tanto las redes neuronales supervisadas como el algoritmo MARS permiten realizar la modelización a partir de los datos, proporcionando MARS un mejor comportamiento ante la extrapolación.

Capítulo 4: Descripción de los datos

1. INTRODUCCIÓN

Los modelos desarrollados en este trabajo pertenecen al dominio de la Minería de datos o soluciones basadas en datos, por lo que el conjunto de datos utilizado es crucial para obtener unos resultados fiables. Su selección y valoración es uno de los primeros problemas para conseguir su efectividad. Se plantean dos alternativas:

- Partir de la toma de muestras de datos procedentes del cierre de proyectos propios o
- Tomar los datos procedentes de una base de datos acreditada.

Para construir una base de datos con información procedente del cierre de proyectos de sistemas de información, se ha de coordinar a las distintas organizaciones que participen en la toma de datos para ajustar sus mediciones y los tipos de métodos empleados para la medición. Además se debe buscar que el conjunto sea representativo de todas las posibles condiciones de desarrollo, puesto que estas técnicas aportan resultados mucho mejores cuando se trata de interpolar que de extrapolar, lo que requeriría recopilar información de un importante número de organizaciones. Otro problema surge del factor tiempo, completar una muestra de datos suficiente procedente del cierre de proyectos dentro de una organización llevaría años por lo que es inviable para cubrir las necesidades y objetivos de esta investigación.

La segunda de las alternativas que se valoran para la adquisición de los datos, es basarse en una base de datos procedentes de cierres de proyectos ya construida y acreditada. Esta ha sido la solución adoptada en esta investigación utilizando la base de datos del *International Software Benchmarking Standards Group* Release 9 [ISBSG] como caso en estudio para el desarrollo del trabajo. Esta organización sin ánimo de lucro se fundó en el año 1.997. La información contenida en su base de datos es el resultado de varios años de cooperación previa de varias asociaciones nacionales de métricas de software, las cuales intentaban desarrollar y promocionar el uso de medidas para mejorar los procesos y productos software. Los datos almacenados en esta base de datos han sido recogidos y valorados por expertos en análisis y desarrollo de software, por lo que se garantiza la calidad y fiabilidad de los mismos. Se muestra a continuación una tabla que contiene las asociaciones por países que forman parte del *ISBSG* y que han colaborado

en la construcción de la base de datos, para demostrar de forma más específica la importancia y el reconocimiento internacional de este conjunto de datos.

País	Organización
Australia	ASMA / SQA (NSW) – Australian Software Metrics Association / Software Quality Association
China	CSPIU (China Software Process Improvement Union)
Finlandia	FiSMA (Finnish Software Measurement Association)
Alemania	DASMA (Deutschsprachige Anwendergruppe fur Software Metrik and Aufwandschatzung)
India	NASSCOM (National Association of Software & Service Companies)
Italia	GUFPI - ISMA (Gruppo Utenti Function Point Italia - Italian Software Metrics Association)
Japón	JFPUG (Japan Function Point User Group)
Corea	KOSMA (Korea Software Measurement Association)
Holanda	NESMA (Nederlandse Software Metrieken Gebruikers Associatie)
España	AEMES (Asociación Española de Métricas de Software)
Suiza	SwissICT
Reino Unido	UKSMA (UK Software Metrics Association)
Estados Unidos	International Function Point Users Group

Tabla 9: Asociaciones nacionales de métricas de software pertenecientes al ISBSG

2. DESCRIPCIÓN DE LOS DATOS

2.1.1.- Denominación de las variables

Las variables contenidas en la base de datos están denominadas evidentemente en inglés dada su procedencia internacional. Para el desarrollo de esta investigación se han utilizado variables en castellano y con el fin de asegurar la equivalencia de este trabajo con otros desarrollados sobre la misma información, se incluye en este apartado una

descripción de cada grupo de variables empleado para facilitar la interpretación, mostrando a su vez el nombre original en inglés.

2.1.2.- Descripción de las variables

Los datos de los que se parte para realizar el estudio proceden de 3.024 proyectos, cada uno de ellos con 100 variables o atributos. Estos atributos se describen en la Tabla 10, agrupados por categorías.

Entorno del proyecto y la calidad de los datos	
Identificador del proyecto (*Project ID*)	*Es el identificador de proyecto, la clave primaria. Este identificador ha sido generado de forma aleatoria para eliminar cualquier posibilidad de identificar a la compañía.*
Enfoque usado para recuento (*Reference Table Approach*)	*Campo de comentario que describe el enfoque usado para el recuento de las tablas de código o datos de referencia.*
Calidad de los datos (*Data Quality Rating*)	*Este campo contiene un código propio de ISBSG para caracterizar la calidad de los datos del proyecto:* *A. Datos de buena calidad, en los que no se ha detectado nada que pudiera afectar a su integridad.* *B. Fundamentalmente de buena calidad pero con algún factor que podría afectar a la integridad de los datos enviados.* *C. Debido a que no se dispone de suficiente información, no fue posible evaluar la integridad de los datos.* *D. D. Debido a uno o a una combinación de factores, se da poca credibilidad a los datos.*
Fecha de implementación (*Implementation Date*)	*Fecha real de implementación. Se muestra en formato de día, mes y año.*
Producto a generar	
Tipo de desarrollo (*Development Type*)	*Este campo describe si el proyecto consistió en un desarrollo nuevo, una mejora o rehacer el producto.*
Plataforma de desarrollo (*Development Platform*)	*Define la plataforma principal de desarrollo, determinada por el sistema operativo usado. Cada proyecto es clasificado como PC, Intermedio, Ordenador Central o Multiplataforma.*
Arquitectura (*Architecture*)	*Define el tipo de arquitectura del proyecto: Monopuesto, Cliente/Servidor, Multinivel, Multinivel con interfaz web.*
Tipo de lenguaje (*Language Type*)	*Define el tipo de lenguaje usado por el proyecto: 2GL, 3GL, 4GL o Generador de aplicaciones.*
Principal lenguaje de programación (*Primary Programming Language*)	*Principal lenguaje utilizado para el desarrollo: JAVA, C++, PL/1, Natural, Cobol, etc.*
Uso de DBMS (*DBMS Used*)	*Si el proyecto utiliza un sistema de gestión de base de datos (Data Base Management System).*
Uso de herramienta CASE (*CASE Tool Used*)	*Si se usaron herramientas CASE.*
Metodología usada (*Used Methodology*)	*Si se usó una metodología.*

Adquisición de Metodología (*How Methodology Acquired*)	*Describe si la metodología fue comprada o desarrollada en la organización.*
Defectos comunicados (*Defects Delivered*)	*Defectos comunicados en el primer mes de uso del sistema. Hay tres columnas en el conjunto de datos que indican el número de defectos comunicados graves (Extreme), medios (Major), leves (Minor).*
Unidades de negocio (*User Base - Business Units*)	*Número de unidades de negocio a los que sirve el sistema o "stakeholders" del proyecto.*
Localizaciones (*User Base – Locations*)	*Número de localizaciones físicas utilizadas por el sistema instalado.*
Número de usuarios (*User Base - Concurrent Users*)	*Número de usuarios que utilizan el sistema concurrentemente.*
Tipo de Organización (*Organisation Type*)	*Identifica el tipo de organización que propone el proyecto.*
Tipo de área de negocio (*Business Area Type*)	*Identifica el tipo de área de negocio a la que se orienta el proyecto cuando este es distinto al tipo de organización.*
Tipo de aplicación (*Application Type*)	*Identifica el tipo de aplicación a la que se orienta el proyecto. Por ejemplo, sistema de información, transacción, control de procesos.*
Paquete personalizado (*Package Customisation*)	*Indica si el proyecto es un paquete personalizado o un desarrollo de uso general.*
Grado de personalización (*Degree of Customisation*)	*Si el proyecto está basado en un paquete existente, este campo proporciona comentarios sobre cómo de complicada fue la personalización.*
Alcance del proyecto (*Project Scope*)	*El alcance del proyecto indica qué tareas fueron incluidas en el valor registrado del esfuerzo del proyecto. Estas pueden ser: Planificación, especificación, diseño, construcción, prueba e implementación.*
Métricas de tamaño	
Obtención de recuento (*Count Approach*)	*Es una descripción de la técnica usada para contar los puntos de función. Ayuda para comparar información con el mismo origen, es decir, elementos similares. Los valores pueden ser, IFPUG, MKII, NESMA, COSMIC-FFP etc.*
Puntos de función sin ajustar (*Unadjusted Function Points*)	*Los puntos de función sin ajustar, es la cantidad antes de cualquier ajuste por el valor del factor de ajuste si se utiliza.*
Valor del Factor de Ajuste (*Value Adjustment Factor*)	*El ajuste de los puntos de función tiene en cuenta varias características técnicas y de calidad, entre las que están: comunicación de datos, rendimiento del usuario final, etc. Si no se dispone de esta información toma el valor 1.*
Puntos de función (*Function Points*)	*En esta variable se registran el número de puntos de función ajustados por el valor del factor de ajuste.*
Métrica del tamaño utilizada (*Function Size Metric Used*)	*La métrica de tamaño funcional usada para grabar el tamaño de los proyectos, como por ejemplo, IFPUG3, IFPUG4, propia de la organización etc.*
Técnica de recuento (*Counting Technique*)	*La técnica de recuento es la tecnología utilizada para sostener el proceso de recuento. Ciertas tecnologías usadas para el recuento de los puntos de función pueden afectar a la potencial exactitud del recuento.*

Categorías de PF (*Function Point Categories*)	*Las categorías de los puntos de función. Se facilitan cinco campos que desglosan las cantidades de funciones en cada uno de ellos: entradas, salidas, consultas, ficheros lógicos e interfaces.*
PF de mejora (*Enhancement Data*)	*En el caso de proyectos de mejora se descomponen los puntos de función en los siguientes tres campos: añadidos, cambios, borrados.*
Clasificación PF sin ajustar (*Unadjusted Function Point Rating*)	*La clasificación de los puntos de función sin ajustar, es un campo que contiene la clasificación de calidad de los revisores de ISBSG con valores A, B, C o D aplicados a los datos de la cantidad de puntos de función sin ajustar. Los valores indican lo siguiente:* *A. Son de buena calidad y sin ningún elemento que pudiese afectar a la integridad.* *B. Parecen de buena calidad, pero la integridad no puede ser asegurada.* *C. Debido a que no se ha proporcionado información descompuesta, no fue posible facilitar la información de puntos de función sin ajustar.* *D. Debido a uno o combinación de varios factores, se podría dar poca credibilidad a la información de los puntos de función sin ajustar.*
LDC (*Source Lines of Code (SLOC)*)	*La cantidad de líneas de código fuente producidas en el proyecto.*
Mediciones de esfuerzo, tiempo, recursos y productividad	
Esfuerzo total (*Summary Work Effort*)	*Facilita el esfuerzo total en horas invertido en el proyecto por parte de la organización de desarrollo. Los tres métodos previstos son los siguientes:* • *Método-A: Horas de personal (Registrado). El esfuerzo comunicado viene de un registro diario de todo el esfuerzo gastado por cada persona en las tareas relacionadas del proyecto.* • *Método-B: Horas de personal (Deducido). El esfuerzo comunicado es deducido del tiempo registrado que señala la asignación de la gente al proyecto. Esto podría conllevar que en la estimación, sólo el 75% del tiempo asignado fue realmente asignado al proyecto; el resto es para vacaciones, formación, etc.* • *Método-C: Sólo tiempo productivo. El esfuerzo comunicado es sólo por el "tiempo productivo" gastado por cada persona en el proyecto. Este valor suele sumar sólo de 5 a 6 horas por día.*
Nivel del recurso (*Resource Level*)	*Los datos son recogidos sobre la gente cuyo tiempo está incluido en los datos del esfuerzo. Se identifican cuatro niveles como medio de toma de datos.* • *Esfuerzo del equipo de desarrollo, por ejemplo, equipo de proyecto, director de proyecto, gerente de proyectos.* • *Soporte del equipo de desarrollo, por ejemplo, administrador de la base de datos, administrador de datos, control de calidad.* • *Infraestructura, por ejemplo, soporte de software, soporte de hardware, administración de red.* • *Usuario final o cliente.* *El número significa que incluye todo este tema y los niveles precedentes. Por ejemplo, un "3" en este campo, significa que el esfuerzo del equipo de desarrollo, soporte de desarrollo e infraestructura está incluido en el valor del esfuerzo.*
Tamaño máximo del equipo (*Max. Team Size*)	*El número máximo de gente que trabajo en cualquier momento en el proyecto, (punto máximo del tamaño el equipo de trabajo).*
Técnicas en el desarrollo (*Dev. Techniques*)	*Técnicas usadas durante el desarrollo.*
Duración del proyecto (*Project Elapsed Time*)	*Duración del proyecto en meses.*
Tiempo de inactividad (*Project Inactive Time*)	*El número de meses en los cuales no existe actividad, por ejemplo, esperando por respuesta de un cliente, esperando aceptación de pruebas.*

Desglose del esfuerzo (*Work Effort Breakdown*)	*El desglose de esfuerzo. Este campo contiene la descomposición del esfuerzo de trabajo comunicado en seis categorías: Planificación, Especificación, Diseño, Construcción, Prueba e Implementación.*
Esfuerzo/actividad (*Ratio of Project Work Effort to Non-Project Activity*)	*Es la proporción del esfuerzo trabajado en el proyecto y las actividades no relacionadas con el proyecto.*
Dispersión del esfuerzo (*Percentage of Uncollected Work Effort*)	*El porcentaje disperso de esfuerzo es el porcentaje de esfuerzo no reflejado en los datos comunicados. Por ejemplo, una estimación del tiempo de esfuerzo no recopilada por el método usado. Los datos se tipifican en los siguientes términos:* • *Menos del 5% del registrado.* • *Entre el 5% y el 10% del registrado.* • *___% sobre el registrado.* • *Incapaz de estimarlo.*
Esfuerzo sin etapas (*Work Effort Unphased*)	*El esfuerzo sin etapas. Tiene lugar si no se proporciona descomposición en fases y contiene el mismo valor que el esfuerzo total. Cuando se descomponen las fases y la suma del descompuesto no es igual al total de esfuerzo mostrándose aquí la diferencia.*
Esfuerzo normalizado (*Normalised Work Effort*)	*El esfuerzo normalizado es una estimación del esfuerzo total del ciclo de vida de desarrollo. Es para los proyectos que no cubren la totalidad del ciclo de vida del desarrollo y este valor. Para los proyectos que cubren la totalidad del ciclo de vida del desarrollo, y los proyectos donde el alcance del ciclo de vida de desarrollo es desconocido, este valor es el mismo que el esfuerzo total de.*
Tasa de productividad normalizada (*Normalised Productivity Delivery Rate*)	*La tasa de productividad normalizada es un campo que contiene la tasa de productividad del proyecto en horas por punto de función calculada como el esfuerzo normalizado dividido por la cantidad de puntos de función sin ajustar.* $PDR\ \ Normalizado = \frac{Esfuerzo\ \ Normalizado}{Puntos de Función \text{sin } ajustar}$
Reparto productividad proy (*Project Productivity Delivery Rate*)	*La tasa de reparto de productividad del proyecto en horas por punto de función calculado como esfuerzo total dividido por cantidad de puntos de función sin ajustar.* $PDR\ \ \text{Pr} = \frac{Esfuerzo\ \ Total}{Puntos de Función \text{sin } ajustar}$
Tasa de productividad norm. Ajustada (*Normalised Productivity Delivery Rate (adjusted)*)	*La tasa de productividad del proyecto en horas por punto de función calculada como el esfuerzo normalizado dividido por la cantidad de puntos de función ajustados.* $PDR\ \ Normalizado\ (ajus) = \frac{Esfuerzo\ \ Normalizado}{Puntos de Función ajustados}$
Tasa de productividad comunicada (*Reported Productivity Delivery Rate (adjusted)*)	*La tasa de productividad en horas por punto de función se calcula como el esfuerzo total dividido por la cantidad de puntos de función ajustados.* $PDR \text{Rep(ajus)} = \frac{Esfuerzo\ \ Total}{Puntos de Función ajustado}$

Calidad del producto	
Total de defectos comunicados (*Total Defects Delivered*)	*Los defectos comunicados en el primer mes de uso. Muestra el total de defectos graves, medios y leves. Cuando la descomposición no está disponible se muestra aquí el valor con la suma total.*
Defectos leves (*Minor defects*)	*Número de defectos leves comunicados en el primer mes de uso.*
Defectos medios (*Major defects*)	*Número de defectos medios comunicados en el primer mes de uso.*
Defectos graves (Extreme defects)	*Número de defectos graves comunicados en el primer mes de uso.*

Tabla 10: Descripción de las variables que representan la información de los proyectos en ISBSG.

Se observa la presencia de un conjunto de variables que tienen información meramente contextual del proyecto, es decir, aportan datos sobre la ejecución o tipo de proyecto pero no se pueden utilizar en un proceso de minería de datos. En este grupo se encuentran variables como alcance del proyecto, técnicas de desarrollo o tipo de organización.

En la siguiente tabla se muestran los diferentes tipos de variables que se encuentran almacenadas en la base de datos en función de su naturaleza.

Clasificación	***Descripción***
Textos	*Variables que contienen texto descriptivo que aporta información sobre el contexto en el que se desarrolla el producto.*
Numéricas	*Variables con contenido numérico, sobre el que se pueden realizar operaciones numéricas.*
Clasificador	*Variables de valor ordinal o entre las que se puede establecer un orden o intervalos*

Tabla 11 Tipificación de los grupos de tipos de variables

En la Fig. 17 se puede ver el número de variables de cada tipo almacenadas en la base de datos. Como se puede observar un 50% de los datos pertenecen a la categoría de texto, que aportan información relativa al contexto del proyecto o la captura de datos. Un 12% de los datos son variables clasificadoras por lo que será necesario aplicar técnicas para poder ser procesadas por el modelo. También hay que destacar que del 38% de las variables de la categoría numérica, muchas de ellas se encuentran claramente relacionadas debido a que son ratios o derivaciones de otras variables.

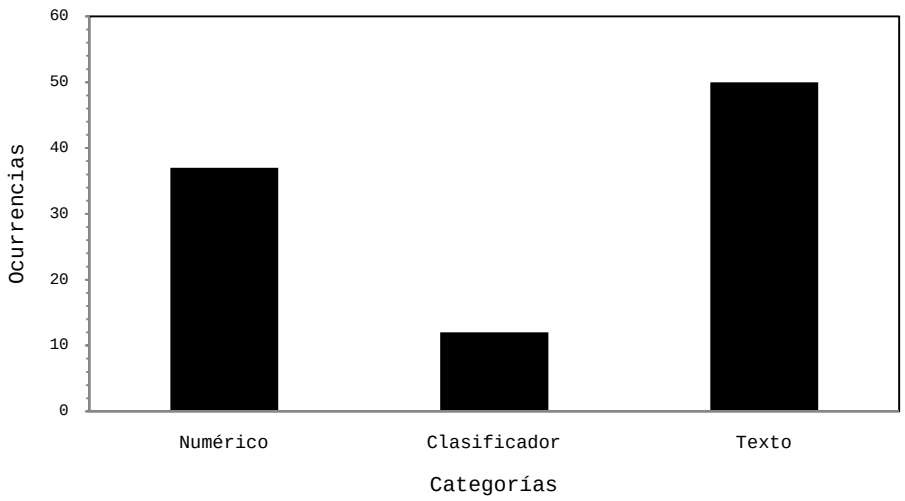


Fig. 17 Distribución de las variables agrupadas por tipo de dato.

De los grupos anteriormente citados sólo se utilizarán para el procesado de datos los grupos de variables que son numéricas o clasificadores. Solamente se excluye como información para el modelado de datos las variables de tipo texto que sólo aportan valor contextual de cómo fue tomada la muestra. Respecto a las variables categóricas, se valorará más adelante si se utilizarán para crear sub-poblaciones de la muestra o si se introducirán directamente en el modelo paramétrico.

3. GUÍA DE USO DE LOS DATOS DEL ISBSG

La base de datos del *ISBSG* R9 contiene información acerca de 3.024 proyectos de software completos. Entre los usos destacados de esta base de datos se encuentran:

- Comparar un proyecto con otros proyectos similares.
- Realizar estimaciones sobre diversos parámetros del proyecto antes del comienzo del mismo.
- Evaluar los beneficios de realizar un cambio en el software o en el hardware correspondiente al entorno de desarrollo.

Dada su disponibilidad y gratuidad a efectos de investigación, así como su internacionalidad, diversidad y representatividad internacional, se han desarrollado sobre ella multitud de trabajos relacionados con metodologías de aspectos específicos del desarrollo de proyectos, entre los que cabe destacar la tesis doctoral desarrollada por J. Villanueva [Villanueva 2005] en 2005 que, dedicada a la predicción de coste y esfuerzo, resulta complementaria de este trabajo y emplea una aproximación metodológica similar.

En los siguientes apartados se muestran unas consideraciones generales a tener en cuenta para el correcto uso de los datos del *ISBSG*.

3.1.- CONSIDERACIONES GENERALES

Independientemente de que el objetivo sea comparar, estimar o evaluar, hay que utilizar los datos de forma prudente asegurando que las comparaciones realizadas son apropiadas y que no se están mezclando o comparando cosas no posibles. El primer paso es seleccionar un conjunto de datos relevante para cada caso en particular. Este paso se realiza en dos fases: Primero, hay que estar seguro que sólo se consideran proyectos en los que se midan las distintas variables (esfuerzo, riesgos, tamaño, etc.) de la misma forma que en el proyecto a comparar, estimar o evaluar. Segundo, considerar proyectos en los que los atributos disponibles son similares a los disponibles en el proyecto a tratar.

3.1.1.- Datos comparables

TAMAÑO

Si el tamaño del proyecto es relevante para el caso de estudio, sólo puede compararse con proyectos en los que se haya usado el mismo método de medición del tamaño. Para seleccionar proyectos que usen el mismo método de medición del tamaño hay que tener en cuenta el campo *Count Approach* de la base de datos.

ESFUERZO

El esfuerzo total del proyecto viene reflejado en el campo *Summary Work Effort*. Se tiene en cuenta el esfuerzo referente a distintos niveles de detalle, en función de los campos *Resource Level* y *Recording Method*. Un valor 1 en el campo *Resource Level* significa que sólo se tiene en cuenta el esfuerzo del equipo de desarrollo, un valor 2 incluye el esfuerzo también de un equipo de soporte, el nivel 3 añade las operaciones de computación y un 4 incluye también el esfuerzo realizado por el cliente o usuario final.

CALIDAD DE LOS DATOS

Para realizar análisis estadístico debe considerarse el campo *Data Quality Rating*. Sólo deben tenerse en cuenta proyectos con valor A o B en este campo.

3.1.2.- Obtención de Datos Relevantes

A menos que la intención sea evaluar un proyecto respecto a la base de datos completa, en general no interesan los proyectos que son muy diferentes del que se quiere evaluar. Lo que interesa es seleccionar los proyectos que son similares centrándose principalmente en los atributos importantes.

El *ISBSG* sugiere que los criterios más importantes para seleccionar proyectos similares son:

- *Size*: Si el tamaño del proyecto a evaluar o comparar es muy grande, no aporta mucho valor estudiar o seleccionar proyectos pequeños y viceversa.
- *Development Type*: Hay tres valores posibles, (1) nuevo desarrollo, (2) mejora o (3) desarrollo sobre un proyecto hecho.
- *Primary Programming Language*, o *Language Type* (ej. 3GL, 4GL)
- *Development Platform* (ej. Mainframe, midrange o PC)

Otros criterios a tener en cuenta son: *Organisation Type, Business Area Type, Application Type, User base* and *Development Techniques*.

Hay que tener en cuenta que a medida que se añaden criterios de selección, el número de proyectos seleccionados se reduce.

3.1.3.- Representatividad de los datos de entrada

Como se comentó anteriormente, las técnicas empleadas en este trabajo presentan mucho mejores resultados cuando se evalúan proyectos que se encuentran incluidos dentro de los límites del espacio de datos marcados por la información utilizada para la modelización, es decir, en interpolación o extrapolación cercana. Los resultados se considerarán representativos por tanto en cualquier punto dentro de ese rango. En la base de datos se encuentran proyectos con tiempos de desarrollo entre 1 y 40 meses, con un máximo aislado de 78 meses, entre 1 y 73 personas de media en el equipo de desarrollo con un máximo de 309, esfuerzos entre 100 y 45.000 horas, con un máximo de 62.000 horas, presentando valores de defectos encontrados entre 0 y 285.

4. CONCLUSIONES

Se utiliza para la realización del estudio una base de datos acreditada y ya construida con un elevado número de proyectos muy diversos y distribuidos a escala internacional debido a que conformar una base de datos propia procedente del cierre de proyectos sería una tarea con demasiada duración, inviable para el desarrollo de este trabajo.

La utilización de los registros contenidos en esta base de datos no se puede realizar de forma directa, sino que es necesario tener en cuenta la calidad de los mismos y el tipo de proyecto que representan en relación a diversos parámetros, por ejemplo el tamaño y la forma de medirlos. Es necesario comparar proyectos cuyos atributos han sido medidos bajo los mismos criterios y utilizando las mismas métricas.

Los modelos a desarrollar están limitados por las características de los proyectos contenidos en la base, especialmente en cuanto a tamaño y tiempos de desarrollo se refiere, si bien este rango es suficientemente amplio para la mayoría de los proyectos desarrollados de forma habitual por las empresas de TI.

Capítulo 5: Metodología

1. INTRODUCCIÓN

El modelado de un problema a partir de la información contenida en un conjunto de datos es un proceso iterativo e interactivo que requiere la aplicación de una metodología estructurada para la utilización ordenada y eficiente de las técnicas y herramientas disponibles. La metodología seguida en este trabajo para la predicción y clasificación del nivel de riesgo en proyectos de sistemas de información es la metodología *CRISP-DM* (Cross-Industry Standard Process for Data Mining) [CRISP-DM] desarrollada en el año 2000 por un importante consorcio de empresas europeas [Chapman 1999] [SPSS 2001].

La metodología *CRISP-DM* consta de cuatro niveles de abstracción, organizados de forma jerárquica en tareas que van desde el nivel más general hasta los casos más específicos (Fig. 18)

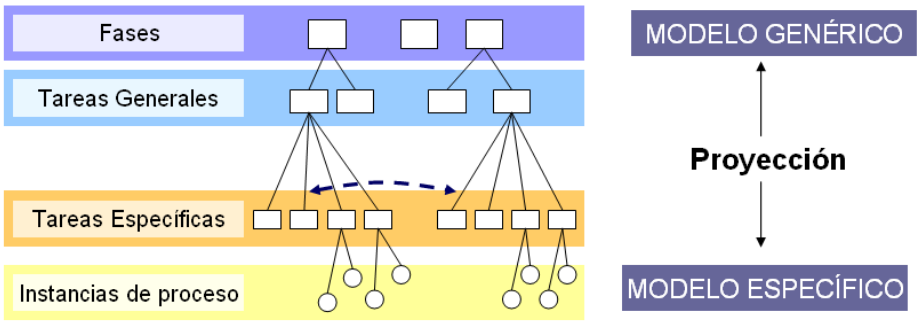


Fig. 18 Esquema de los cuatro niveles de abstracción de la metodología CRISP-DM

Las fases en el proceso global de modelado de datos no están claramente diferenciadas y lo convierte en un proceso iterativo e interactivo. Las interacciones entre las decisiones tomadas en diferentes fases, así como los parámetros de los métodos utilizados y la forma de representar el problema suelen ser extremadamente complejos. Pequeños cambios en una parte pueden afectar fuertemente al resultado final.

En la metodología *CRISP-DM* el proceso se estructura en seis fases (tal como se muestra en la Fig. 19) que serán desarrolladas con detalle durante este capítulo.

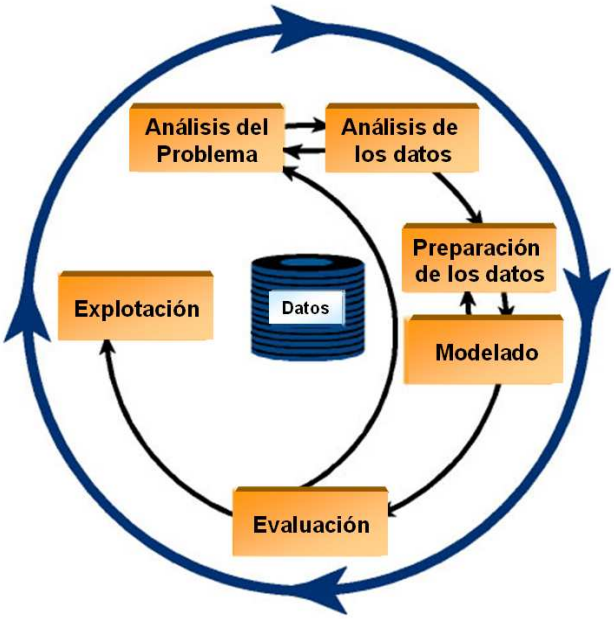


Fig. 19 Fases del proceso de modelado metodología CRISP-DM.

Las flechas indican relaciones más habituales entre las fases, aunque se pueden establecer relaciones entre cualquier fase. El círculo exterior simboliza la naturaleza cíclica del proceso de modelado.

Análisis del problema	**Análisis de los datos**	**Preparación de los datos**	**Modelado**	**Evaluación**	***Explotación***
Determinación de los objetivos empresariales • Conocimiento previo • Criterios de éxito	Adquisición de datos • Análisis fuentes datos • Estudio datos disponibles • Instalación base datos	Procesado de datos • Conversión a valores numéricos • Rellenado de datos • Identificación de valores no usuales	Selección de la técnica de modelado • Técnica de modelado • Supuestos de la técnica de modelado	Evaluación de los resultados • Valoración de los resultados • Modelos válidos	*Planificación de la explotación* • *Plan de utilización*
Evaluación de la situación • Recursos disponibles • Requerimientos, supuestos y restricciones • Terminología	Descripción de datos • Tipo • Unidades • Significado • Procedencia	Reducción de la dimensionalidad • Variables • Muestras	Diseño del método de evaluación • Medidas de error	Revisión del proceso • Detección de errores en el proceso de modelización	*Planificación de la monitorización y mantenimiento* • *Plan de monitorización y mantenimiento*

Determinación de los objetivos técnicos • Objetivos del modelado • Criterios de éxito	Exploración de datos	Transformación de datos • Normalización • Trasformaciones matemáticas • Discretización	Generación del modelo • Parámetros del modelo • Modelos • Descripción del modelo	Determinación de las siguientes acciones • Lista de las posibilidades • Decisión	*Revisión del proyecto* • *Extracción de conclusiones*
Elaboración de la estrategia • *Planificación* • *Valoración inicial* • *Téc./ herramientas*	*Verificación de la calidad de los datos*		*Evaluación del modelo* • *Verificación de los resultados* • *Obtención de más información*		

Tabla 12 Esquema de la metodología seguida durante el proceso de modelado del nivel de riesgo en proyectos de sistemas de información

2. ANÁLISIS DEL PROBLEMA

La fase inicial en el proceso de modelado consiste en realizar un análisis del problema a modelar para conocer el desarrollo y dominio de la aplicación, establecer los objetivos del usuario final, así como determinar el conocimiento relevante a usar. El desarrollo de esta primera fase establece las bases para la realización de las posteriores fases del proceso, por lo que el éxito o fracaso va a depender en gran medida de las decisiones que se adopten en esta etapa.

Las tareas que se realizarán en esta fase son:

- **Determinación de los objetivos**. A partir de la información disponible sobre el entorno en el que se desarrolla el proceso de modelado se deben de identificar los objetivos que se pretenden conseguir.
- **Evaluación de la situación**. Esta tarea implica recoger toda la información disponible sobre el entorno en el que se desarrolla el proceso que se desea modelar con el fin de identificar los factores que pueden afectar al desarrollo de cualquier fase del proceso de modelado.
- **Determinación de los objetivos técnicos.** Los objetivos empresariales establecidos en el primer punto se proyectan en objetivos técnicos identificando

el tipo de problema a resolver (predictivo, clasificación, segmentación, etc.) así como el grado de precisión requerido para los resultados.

- **Elaboración de la planificación**. Se realiza una valoración inicial de las técnicas y herramientas disponibles que se adaptan al problema específico a resolver, realizando una primera elección de las herramientas a utilizar durante el modelado.

3. ANÁLISIS DE LOS DATOS

En esta fase se realizará un estudio de la información relevante para el modelado del proceso así como de su disponibilidad. El fin de esta fase es obtener un conjunto de datos objetivo o establecer el sistema para su adquisición (creación de una nueva base de datos), que contenga la información necesaria para caracterizar el proceso a modelar. De esta forma, el proceso de búsqueda de información útil se centrará en subconjuntos de variables y/o muestras de este conjunto de datos.

Las tareas a realizar en esta fase son:

- **Adquisición de datos**. Se analizan cuáles son los datos necesarios para el desarrollo del trabajo y se estudia la forma de obtenerlos. Una vez identificados estos factores se procederá a la obtención de un conjunto inicial de datos que permita realizar un análisis preliminar de la calidad y relevancia de la información disponible.
- **Descripción de los datos**. Se describen las características fundamentales de los datos, tanto de las tablas como de cada una de las variables individuales (tipo de los datos, unidades, rangos, etc.) generando un diccionario de datos.
- **Exploración de los datos**. En esta tarea se analizan los datos en más profundidad identificando los datos clave y estableciendo las primeras relaciones entre los atributos.
- **Verificación de la calidad de los datos**. Se comprueba la existencia de errores en los datos. Los errores de datos más frecuentes son: campos vacíos, ruido e inconsistencia entre distintas fuentes de datos, datos redundantes y valores de datos inconsistentes desde un punto de vista físico.

4. PREPARACIÓN DE DATOS

A partir de los datos reunidos y analizados en la fase anterior, se generará el conjunto definitivo de datos que se usará en el modelado. Para ello se procede al preprocesado, transformación y reducción de datos.

4.1.- PREPROCESADO DE DATOS

El objetivo del **preprocesado de datos** es la transformación del conjunto original de datos en un nuevo conjunto de datos más significativo y manejable. Formalmente, [Famili 1997]:

El preproceso es una transformación $\boldsymbol{T}$ que transforma la matriz que contiene los datos reales del proceso $\boldsymbol{X}$, en una nueva matriz $\boldsymbol{Y}$ tal que:

1. $\boldsymbol{Y}$ conserva la información de $\boldsymbol{X}$
2. $\boldsymbol{Y}$ elimina al menos uno de los problemas contenidos en $\boldsymbol{X}$
3. $\boldsymbol{Y}$ es más útil que $\boldsymbol{X}$

El preproceso de los datos incluirá tres etapas principales:

- **Conversión de los datos a valores numéricos.** En general los datos simbólicos no son tratados por los algoritmos matemáticos por lo que resulta necesaria su conversión a un formato adecuado.
- **Rellenado de datos**. En los datos provenientes de procesos industriales es frecuente la ausencia o pérdida de datos debido a fallos de muestreo o a la no sincronización de variables (unas variables se muestrean a diferente velocidad que las otras). Algunas aproximaciones simples [Dixon 1979] y otras más complejas [Dempster 1977] han sido propuestas para solucionar este problema. Las alternativas que se contemplan para solucionar este problema son: Llenar los datos mediante técnicas estadísticas que calculen los valores de reemplazo a partir del resto de los datos o mantener los patrones incompletos cuando los algoritmos utilizados contemplen la situación (redes SOM) o bien descartando los patrones incompletos.
- **Identificación de los valores no usuales.** Se considera valor no usual a un ítem de datos cuyo valor cae fuera de los límites que encierran a la mayoría del resto de los valores correspondientes de la muestra adoptados por la variable.

Este tipo de dato no usual se denomina *outlier*. Los *outliers* pueden venir originados por errores en la medida o en la inserción de datos. En tales casos sería deseable que los *outlier* no afectaran al resultado de los análisis por lo que en esta situación el *outlier* puede descartarse y realizar el análisis con el resto de los datos. Sin embargo, es posible que el origen del *outlier* no sea un dato erróneo, sino que durante algún momento del proceso se ha adoptado el valor almacenado en el *outlier*. En esta situación el análisis detallado de los *outlier* suele proporcionar una valiosa información sobre el proceso en estudio. Por lo tanto los *outlier* serán estudiados y analizados en detalle para determinar su origen, siendo eliminados únicamente cuando se tenga la certeza de que representan valores erróneos.

4.2.- REDUCCIÓN DE LA DIMENSIONALIDAD DE LOS DATOS

La extracción de las características de los datos (reducción de la dimensionalidad) debe ser realizada debido a un problema común que se presenta para los modelados no lineales, la *maldición de la dimensionalidad* (*curse of dimensionality*) [Bellman 1966].

El proceso de extracción de características trata de reducir la cantidad de información (reducción de dimensionalidad) que representa a cada uno de los patrones, obteniendo un vector de atributos que represente de la mejor manera posible al patrón original prescindiendo de la información que no sea muy importante.

La reducción de la dimensionalidad puede ser de dos tipos: reducción del número de variables y reducción del número de datos a analizar.

El propósito fundamental que se busca al reducir variables es mejorar la velocidad de aprendizaje, la capacidad de generalización y/o la simplicidad de la representación, para así comprender mejor los resultados obtenidos, disminuir el volumen de almacenamiento, reducir ruidos generados por aquellas variables que no aportan información y eliminar conocimiento inútil para la solución del problema.

Si los datos proceden de un proceso del cual se posee conocimiento a priori, esta información puede ser utilizada para la extracción de los atributos, guiando y limitando la búsqueda de conocimiento, y en consecuencia mejorando la eficiencia. Sin embargo, la selección del conocimiento previo que va a emplearse ha de hacerse con sumo cuidado, pues podría descartar descubrimientos válidos no previstos, como se indica en

[Piatetski 1991] con un sencillo ejemplo: una restricción como "los camiones no circulan sobre el agua" reduce el espacio de búsqueda, pero limita algunas soluciones posibles como que los camiones vayan sobre lagos helados en invierno.

En general la extracción de las características es una tarea difícil de realizar, y que será desarrollada mediante el estudio de las relaciones de dependencia estadística entre las variables, representaciones gráficas, métodos de proyección no lineal (mapas autoorganizados SOM) y con la aplicación de una estrategia de poda iterativa realizada mediante la aplicación APIMARS.

4.3.- TRANSFORMACIÓN DE LAS VARIABLES

Los datos deben de transformarse de tal forma que su formato resulte adecuado para el algoritmo con el que sean analizados. Las principales causas para realizar transformaciones en las variables son:

- **Eliminar el efecto de los *outliers***. La presencia de un *outlier* puede alterar la convergencia de los algoritmos que minimizan el error cuadrático medio, puesto que el algoritmo será modificado para intentar mejorar el error cometido para el *outlier* empeorando el error cometido para el resto de datos.
- **Convertir los datos en más interpretables.** Si la distribución de las variables es muy sesgada su visualización resulta muy difícil de interpretar. Para garantizar la interpretabilidad de las variables muy sesgadas se puede aplicar un esquema de discretización o realizar una transformación logarítmica.

Las transformaciones más usuales son:

- **Normalización de las variables.** La normalización de las variables suele proporcionar mejores resultados que la introducción directa de los datos en los algoritmos que utilicen en su implementación fórmulas matemáticas o medidas de distancias. Las redes neuronales, por ejemplo, entrenan generalmente mejor si los valores de los datos son pequeños, por lo que se puede proceder a realizar una normalización de las variables entre un rango específico comprendido entre 0.1 y 0.9.
- **Transformaciones matemáticas.** Las transformaciones matemáticas de los datos mediante la aplicación de logaritmos (transformación logarítmica) o raíz

cuadrada resultan eficaces para estandarizar las distribuciones de variables muy sesgadas.

- **Discretización de las variables.** En ocasiones, la discretización de variables continuas proporciona una mejor interpretación de los datos. La discretización de variables puede realizarse utilizando cuantiles: Por ejemplo para la discretización de la variable, por medio de los cuantiles 10, 25, 50, 75 y 90, la variable pasará a ser discreta pudiendo tomar seis posibles valores: muy bajo, bajo, ligeramente por debajo de la media, ligeramente superior a la media, alto y muy alto.

5. MODELADO

Este punto responde a la cuarta fase de la metodología *CRISP-DM*. En ella se seleccionan de forma definitiva la técnica o técnicas a utilizar y se aplican a los datos disponibles, ajustando los parámetros a sus valores óptimos.

A continuación se incluye una descripción detallada de cada una de las tareas en las que se puede desglosar la fase de modelado.

5.1.- SELECCIÓN DE LA TÉCNICA DE MODELADO

Dado un problema de modelado una de las cuestiones que se plantean es la selección de la técnica o técnicas a utilizar. La selección de las técnicas utilizadas para el desarrollo de este trabajo ha sido presentada en el capítulo anterior. Los criterios seguidos para la selección de las técnicas utilizadas en el modelado han sido:

- Ser apropiada al problema
- Disponer de datos adecuados
- Capacidad descriptiva y predictiva de la técnica
- Cumplir los requerimientos del problema
- Tiempo necesario para obtener un modelo

6. CONOCIMIENTO DE LA TÉCNICA

6.1.- DISEÑO DEL MÉTODO DE EVALUACIÓN

Para cada algoritmo es necesario establecer criterios de evaluación que proporcionen medidas de la diferencia entre el resultado del modelo y el proceso real. Para ello se distinguirá entre el *error real* y el *error aparente*.

Si se supone que existe una población de la cual han sido extraídos los datos de entrenamiento de forma aleatoria, se define el *error real* o *error poblacional* como el error que se comete al ser evaluado el modelo sobre los datos de la población.

Se define *error aparente* o *error muestral* como el error del modelo sobre la muestra de casos utilizados en el diseño o construcción del mismo.

El objetivo del modelo es minimizar el error real cometido que al ser desconocido debe de ser estimado. Para la estimación de los errores se utilizará la distancia euclídea:

$$\boldsymbol{D}(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{i=1}^{N} (\boldsymbol{x}_i - \boldsymbol{y}_i)^2} \qquad (4.1)$$

Aunque, si fuera necesario se puede considerar la posibilidad de utilizar otras medidas de distancias alternativas.

Una vez seleccionada la medida de distancia a utilizar, se realizará una estimación del error del modelo mediante el cálculo de la distancia media entre la salida predicha por el modelo y la salida real, esta estimación es conocida como *error cuadrático medio*:

$$\boldsymbol{ECM} = \sqrt{\frac{\sum_{i=1}^{N} (\boldsymbol{y}_i - \boldsymbol{y}_i^*)^2}{\boldsymbol{N}}} \qquad (4.2)$$

Además de proporcionar esta estimación del error, se fijarán unas tolerancias de error absoluto (1, 3, 5, 7, 9, 11, 13, 15 y 17) calculando el porcentaje de aciertos obtenidos por el algoritmo para estas tolerancias.

El estimador más simple del error real es el error aparente o error cometido sobre los patrones de entrenamiento. El problema fundamental de este estimador, es que se

calcula usando el mismo conjunto de patrones que el utilizado para construir el modelo, por lo que proporciona un estimador sesgado optimista de la bondad del modelo.

Para solventar el problema de la dependencia entre el conjunto usado para construir el modelo y el usado para realizar la estimación se utilizará el método denominado *entrenamiento-prueba* que consiste en dividir el conjunto inicial de patrones, $\boldsymbol{T}$, en dos conjuntos independientes $\boldsymbol{T_a}$ y $\boldsymbol{T_p}$ de forma que:

1. Los patrones de $\boldsymbol{T_a}$ constituyen el *conjunto de aprendizaje* y se usan únicamente para construir el modelo
2. Los patrones de $\boldsymbol{T_p}$ constituyen el *conjunto de prueba* y se usan únicamente para estimar el error.

Para que el método entrenamiento-prueba resulte eficaz debe de asegurarse que los patrones de $\boldsymbol{T_a}$ sean independientes de los de $\boldsymbol{T_p}$ pero que sigan la misma distribución. Para asegurar estas condiciones se realizará una partición de $\boldsymbol{T}$ seleccionando los patrones aleatoriamente, de forma que se cumplan las condiciones (4. 3) y (4. 4).

$$\boldsymbol{T_a} \cup \boldsymbol{T_p} = \boldsymbol{T} \tag{4.3}$$

$$\boldsymbol{T_a} \cap \boldsymbol{T_p} = \varnothing \tag{4.4}$$

El tamaño de los conjuntos de prueba se seleccionará de forma que se utilicen aproximadamente el 75% de datos para entrenamiento y el 25 % de datos restantes para prueba, es decir:

$$card(T_p) = \frac{1}{4} card(T) \tag{4.5}$$

$$card(T_a) = \frac{3}{4} card(T) \tag{4.6}$$

donde ***card(T)*** es el cardinal o tamaño del conjunto $\boldsymbol{T}$.

Si el número de datos disponibles no es muy elevado se aumentará el porcentaje de datos destinados a entrenamiento a un 80%. Si esta cantidad resulta aún demasiado baja se utilizará un estimador por validación cruzada.

El *estimador por validación cruzada* con V conjuntos, [Silverman 1985], *distribuye* aleatoriamente los patrones de T en V conjuntos disjuntos $T_1, T_2,..., T_V$ de un tamaño similar:

$$(card(T_i) \approx \frac{card(T)}{V} \quad , i=1,2,...,V) \tag{4.7}$$

El procedimiento de estimación puede plantearse como sigue:

1. Para todo v, v = 1, 2,..., V, construir un modelo, que se denotará por f_v, usando $T - T_v$, como conjunto de aprendizaje y T_v como conjunto de prueba realizando la estimación del error sobre este conjunto.
   Al finalizar este paso se obtienen V modelos, f_v, con sus correspondientes estimaciones de error.
2. Construir un modelo general f, usando todos los patrones de T.
3. Los errores obtenidos en 1 se usan para estimar el error en 2 como valores de una distribución normal.

La suposición básica de la validación cruzada es que este procedimiento es "estable", esto es, que todos los modelos f_v, para $v = 1, 2,..., V$ (construidos con *casi todos* los patrones de T) tienen una tasa de error *aproximadamente igual* a la del modelo f (construido con *todos* los patrones de T).

Cuando $V = N$, el estimador por validación cruzada con N conjuntos se conoce como el *estimador "que deja uno fuera",* del inglés *leave-one-out*, [Craven 1979]. Para cada n, n = 1, 2,..., N el n-ésimo patrón es descartado y el modelo se construye utilizando los restantes N - 1 patrones. El patrón descartado se usa para prueba estimando el error cometido para ese patrón.

6.2.- GENERACIÓN DEL MODELO

Una vez fijada la representación del modelo (o familia de representaciones) y el criterio de evaluación del modelo, el problema puede ser reducido a un problema de optimización del tipo:

> "Encontrar los parámetros/modelos de la familia seleccionada que optimicen el criterio de evaluación del modelo"

Todos los parámetros de entrada de la técnica de modelado, deben de ser analizados de cara a establecer la importancia de cada uno de ellos y su influencia en el resultado del modelo. Siempre que sea posible los parámetros iniciales se asignarán en función de las características de los datos.

Con un mismo conjunto de parámetros se generarán modelos distintos, analizando los resultados de los modelos tanto desde un punto de vista numérico como desde un punto de vista gráfico.

Para cada modelo se generará un informe que recoja:

- Descripción detallada de las características del modelo (variables de entrada, rangos de variación y características de las variables de entrada, número de patrones utilizados para entrenamiento y para prueba).
- Parámetros utilizados para generar el modelo
- Exactitud y sensibilidad del modelo
- Interpretación del modelo.

6.3.- EVALUACIÓN DEL MODELO

Los modelos generados se ordenan en función de la calidad de los resultados obtenidos, comparando la evolución de los resultados, en orden a seleccionar el mejor modelo.

Se realiza un análisis de los resultados obtenidos dirigido en dos direcciones:

- **Verificación de resultados**. Se debe verificar la coherencia de la información obtenida con otros tipos de conocimiento extraídos previamente, resolviendo las posibles inconsistencias existentes.
- **Obtención de más información.** La información extraída se utilizará como información a priori para la extracción de más información. Para ello será necesario retornar a alguna de las fases anteriores del proceso de modelado y modificar algunas de las decisiones tomadas durante esas fases, haciendo para ello uso de la nueva información obtenida. Alguna de las decisiones que pueden ser tomadas para la obtención de más información son, por ejemplo: recolección de nuevos datos, separación de datos en clases, transformaciones de las variables, eliminación de datos, selección de otros algoritmos de modelado,

cambio en los parámetros introducidos en los algoritmos, delimitación del campo de búsqueda, etc.

7. EVALUACIÓN

En esta fase se evalúa el modelo, no desde el punto de vista de los datos, sino del cumplimiento de los criterios de éxito del problema. Se debe revisar el proceso seguido hasta el momento, teniendo en cuenta los resultados obtenidos, para poder repetir algún paso en el que, a la vista del desarrollo posterior del proceso, se hayan podido cometer errores.

Las tareas que se desarrollarán en esta fase son las siguientes:

7.1.- EVALUACIÓN DE LOS RESULTADOS

Se trata de comprobar que el modelo cumple los objetivos, así como de determinar si hay algún punto en el que el modelo no haya conseguido los resultados esperados. Se debe tener en cuenta también el nuevo conocimiento del problema adquirido mediante el modelo, incluso en aspectos no contemplados inicialmente.

Para ello se resumen los resultados del modelo teniendo en cuenta diversos factores:

- Comprensibilidad de los resultados
- Aplicabilidad de los mismos
- Grado de innovación de la información descubierta
- Grado de cumplimiento de los objetivos iniciales

7.2.- REVISIÓN DEL PROCESO

Se revisa el proceso seguido durante la construcción del modelo actual tratando de encontrar fases que no se hayan desarrollado correctamente y buscando alternativas que permitan construir un modelo mejor.

Finalmente, se revisan los resultados de acuerdo con el grado de cumplimiento de los criterios de éxito.

7.3.- DETERMINACIÓN DE LAS ACCIONES SIGUIENTES

De acuerdo con los puntos anteriores, se decidirá si se pasa a la fase de explotación o se repiten las fases anteriores. Para tomar esta decisión se deben considerar los siguientes puntos:

- Potencial de explotación de cada modelo
- Posibilidades de mejora del proceso

8. EXPLOTACIÓN

Durante esta fase se plantea una estrategia para la integración del conocimiento adquirido y de los modelos desarrollados en el proceso industrial.

La integración de los resultados obtenidos depende tanto de las características del modelo como del estado actual del proceso industrial. En función de estas características se toman decisiones sobre la mejor actuación del modelo (tiempo real o modelos off-line), monitorización de los resultados, estrategia de mantenimiento del modelo (necesidad de adaptación y/o reentrenamiento) e impacto de la actuación del modelo sobre las condiciones actuales de funcionamiento del proceso.

Los resultados obtenidos durante la fase de explotación pueden derivar en retornar a la fase de modelado para introducir mejoras en el modelo, incorporar conocimiento experto, o solventar posibles errores del modelado tales como por ejemplo la presentación de casos no considerados durante la fase de análisis de los datos (nuevas regiones de trabajo).

9. CONCLUSIONES

Para abordar con éxito cualquier trabajo de investigación, se hace imprescindible una metodología que permita seguir unos pasos preestablecidos y garantice las máximas posibilidades de éxito. La metodología adoptada en este trabajo de investigación se basa en la metodología *CRISP-DM*.

Así, el trabajo se ha estructurado en diferentes fases, que marcan las líneas generales de investigación, subdivididas en tareas que constituyen un nivel de abstracción más específico. La metodología tiene en consideración la interacción entre las diferentes

fases, de tal forma que se facilite la incorporación progresiva del nuevo conocimiento adquirido a la investigación.

El seguimiento de una metodología estructurada facilita además la adaptación de los resultados.

Capítulo 6: Análisis de resultados

Predicción y Clasificación del Riesgo por modelado

1. INTRODUCCIÓN

En este capítulo se persigue aplicar las técnicas de minería de datos al proceso de predicción y clasificación del riesgo en gestión de proyectos de sistemas de información, de manera que se aporte una visión innovadora al proceso tradicional de resolución de este problema.

La aplicación de las técnicas de minería de datos no se puede realizar de forma directa, sino que es necesario realizar un proceso que permita transformar la información relativa al proceso de predicción del riesgo en un formato que pueda ser tratado por dichas técnicas con el fin de poder definir un modelo de estimación. En esta investigación se analizarán las diferentes etapas a realizar mediante el estudio de un caso facilitado por el *International Software Benchmarking Standards Group* Release 9 [ISBSG].

Para realizar el proceso de predicción y clasificación del riesgo en proyectos de sistemas de información se ha seguido la metodología estructurada presentada en el capítulo anterior. Así, se comienza el capítulo con un breve análisis del problema, utilizando como referencia la base de datos facilitada por el *ISBSG* y definiendo los objetivos a alcanzar por las técnicas de minería de datos para esta base. A continuación se procede a realizar un análisis de los datos almacenados en la base que permita identificar las características de los datos asociados al problema de predicción y clasificación de riesgos en proyectos de sistemas de información. Posteriormente se procede a realizar un preproceso de los datos de manera que puedan ser tratados por las técnicas de minería. Sobre el conjunto de datos preprocesados se definieron varios modelos que, a partir de la información disponible al comienzo de un proyecto de sistemas de información, permitan realizar una estimación del riesgo existente para la realización del mismo en cuanto a calidad (defectos del producto software y clasificación cualitativa de los mismos) y plazos. Una vez definidos los modelos se procede a una evaluación de los resultados de los mismos, seleccionando el modelo que mejor se adapta a las características de los datos en estudio.

2. ANÁLISIS DEL PROBLEMA

La descripción del contexto en el que se desarrolla el problema planteado en este trabajo, así como las diferentes aproximaciones con las que se ha abordado la estimación

del riesgo en proyectos software se han desarrollado en los capítulos uno y dos respectivamente.

En este apartado se procederá a transformar el problema, de forma que pueda ser tratado desde un punto de vista matemático. Para ello se exponen los objetivos generales que se pretenden lograr y se realiza una evaluación de la situación para, a partir de los resultados obtenidos, transformar los objetivos generales en objetivos técnicos, y plantear la estrategia a seguir para alcanzarlos.

2.1.- DETERMINACIÓN DE LOS OBJETIVOS

El objetivo general de este trabajo consiste en desarrollar un método o sistema que facilite el proceso de estimación de riesgos en proyectos de sistemas de información. Este objetivo general debe de transformarse en objetivos específicos que sean medibles y alcanzables mediante la aplicación de técnicas de minería de datos. Por lo tanto, del objetivo general de esta investigación se derivan los siguientes objetivos específicos:

- Definir tres modelos predictivos basados en datos capaces de estimar el número de defectos potenciales que contendrá el sistema de información a desarrollar (leves, medios y graves).
- Desarrollar un sistema que permita extraer el conocimiento implícito contenido en una base con datos procedentes de cierre de proyectos.

2.2.- EVALUACIÓN DE LA SITUACIÓN

Dado que se ha optado por realizar una solución basada en datos, uno de los recursos que más importancia tiene en el trabajo es el conjunto de datos.

En el capítulo 4 se ha explicado de forma detallada las posibilidades de adquisición de datos, ventajas e inconvenientes, así como la elección realizada para el desarrollo del presente trabajo, tomando como base de partida del conjunto de datos una base de datos acreditada. En el mismo capítulo se ha realizado una descripción detallada de la organización que provee esos datos, el *International Software Benchmarking Standards Group* así como el contenido de los mismos. Los datos almacenados en la base de datos han sido recogidos y valorados por expertos en métricas de software, por lo que se garantiza la calidad y fiabilidad de los mismos.

Sin embargo, la principal limitación de la utilización de la base de datos *ISBSG* consiste en que los proyectos almacenados en la misma se han desarrollado en un entorno de trabajo muy distinto al que se puede vivir en una organización en particular por lo que los resultados pueden no ser extrapolables directamente a la situación local.

Como modelo de referencia para evaluar los resultados obtenidos se ha desarrollado un modelo de regresión multivariante. Los resultados obtenidos por este modelo de regresión no son malos, por lo que se dispone de un modelo de referencia no trivial permitiendo realizar comparaciones de sus resultados con los obtenidos por los modelos inteligentes basados en datos desarrollados en este trabajo.

La Fig. 20 muestra los resultados obtenidos por el modelo de referencia conteniendo el porcentaje de aciertos para diferentes tolerancias de error absoluto.

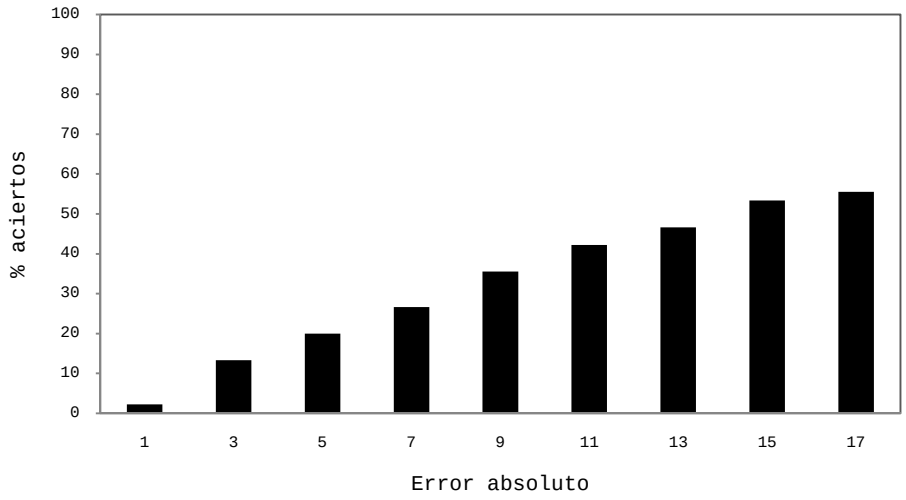


Fig. 20 Resultados del modelo de referencia

En la siguiente tabla se muestran los resultados obtenidos en el modelo de referencia que servirá para la evaluación de los nuevos modelos paramétricos desarrollados.

Error absoluto	Porcentaje de aciertos
1	2%
3	13%
5	20%
7	27%
9	36%
11	42%
13	47%
15	53%
17	56%

Tabla 13 Error absoluto y porcentaje de aciertos del modelo de referencia

2.3.- DETERMINACIÓN DE OBJETIVOS TÉCNICOS DEL MODELO

El tipo de problema analizado en este trabajo se enmarca dentro de los modelados predictivos con una única variable de salida. Como el objetivo perseguido es realizar una predicción del número de defectos leves, medios y graves que se cometerán en el desarrollo de un proyecto software, en realidad se obtendrán tres modelos, uno para cada tipo de defecto. Se predecirá el número de defectos a través de un modelo paramétrico basado en datos, que tomando como atributos la información disponible del proyecto, será capaz de predecir con mejor precisión que los modelos actuales el número de defectos y tipo de los mismos que se cometerán.

El desarrollo de un modelo de este tipo, permitirá la identificación de los atributos más influyentes en la estimación de defectos, y servirá como herramienta de apoyo del estimador o gestor de proyectos para determinar el número de defectos y tipo de los mismos que potencialmente se van a cometer, permitiendo centrar los esfuerzos humanos y económicos disponibles para el control de riesgos en aquellos apartados más influyentes en la obtención de un producto de mala calidad.

Para fijar los criterios de éxito del modelo se tomará como referencia el modelo de regresión desarrollado.

De cara a utilizar una medida de error para realizar las comparativas entre los diferentes modelos o con el anterior, se utilizará el error cuadrático medio. Las mejoras de unos modelos respecto a otros se medirán a partir de los porcentajes relativos de sus respectivos errores cuadráticos medios, comparando los resultados obtenidos con los reflejados en la Tabla 13.

2.4.- ELABORACIÓN DE LA ESTRATEGIA

Se procede a elaborar el plan inicial de trabajo que permita conseguir los objetivos fijados. Tomando como referencia los resultados de la evaluación de la situación y los conocimientos aportados por los primeros capítulos del trabajo, se procede a elaborar el siguiente plan de trabajo.

- Al disponer de un conjunto de datos cuya recolección está avalada por una organización internacional de prestigio como es *ISBSG*, se realizarán tres modelos paramétricos basados en datos: uno para predecir el número de defectos leves, otro para los medios y otro para los graves.

- El proceso de creación de los modelos seguirá los pasos y fases que se proponen en el capítulo de metodología.
- En el proceso de modelado se tomará como variable objetivo el número de defectos, leves, medios y graves en cada caso, para lo cual habrá que decidir qué variables disponibles y bajo qué unidades, es mejor candidato como variable objetivo.
- Después del análisis y el preproceso de los datos y en función de los problemas detectados en el conjunto de datos, se tomará la decisión de la técnica de modelado a utilizar.
- Se intentarán conseguir tres únicos modelos genéricos para el conjunto de datos. En caso de no cumplir los criterios de éxito se buscarán grupos o subconjuntos de datos para crear un modelo para cada subconjunto, por lo que habrá que decidir cuál es la variable o grupo de ellas que dividirá los grupos.

2.4.1.- Valoración inicial de técnicas

En relación al plan inicial para la valoración de las técnicas a utilizar en este trabajo, se ha de tener muy presente el conjunto de datos. En este trabajo se pretende utilizar redes neuronales o MARS y esto dependerá de la calidad de los datos después del preproceso.

En este trabajo se ha seleccionado como técnica de modelado una técnica ya considerada como clásica dentro de las técnicas basadas en datos: la red neuronal perceptrón multicapa. Este tipo de redes proporciona en general unos modelos con una alta capacidad predictiva, por lo que pueden ser adecuadas para el tratamiento del problema planteado en esta investigación. Sin embargo, para la aplicación de las redes perceptrón multicapa deben de tenerse en consideración las siguientes características:

- Fuerte influencia de los datos anómalos (*outliers*) sobre los resultados de los modelos neuronales: Dado que la mayoría de los algoritmos de entrenamiento están basados en minimizaciones de errores cuadráticos, la presencia de *outliers* desvía la aproximación del modelo hacia el *outlier* empeorando los resultados globales.
- Selección de un conjunto reducido de variables relevantes. Dado que el número de parámetros de la red neuronal depende del número de neuronas, y el número de datos de entrenamiento suele ser limitado, para evitar problemas de

sobreajuste se deben introducir un número reducido de variables de entrada correspondiente a las variables más relevantes.

- Configuración de los parámetros y la topología de la red. Aunque algunos investigadores recomiendan utilizar una capa oculta compuesta por el doble de neuronas menos una, esta regla no resulta efectiva para la mayoría de las situaciones, siendo necesario recurrir a la experiencia del analista y a la realización de numerosas pruebas para la determinación de la topología adecuada al problema.

Por lo tanto en este trabajo para el desarrollo del modelo paramétrico se utilizará también la versión modificada APIMARS del algoritmo MARS, dado que su robustez frente a la calidad de los datos y su interpretabilidad de los resultados puede proporcionar una herramienta de apoyo a las redes neuronales e incluso puede ser utilizada como herramienta para el desarrollo de los modelos paramétricos.

2.4.2.- Valoración inicial de herramientas de modelado

Las herramientas de modelado utilizadas en este trabajo varían desde las herramientas comerciales pasando por las de libre distribución hasta las desarrolladas por el propio Grupo de Investigación en Proyectos de Ingeniería de la Universidad de Oviedo.

Herramientas comerciales

Las herramientas comerciales utilizadas han sido:

- *MATLAB*: Como plataforma para el análisis y tratamiento de los datos.
- *SPSS*: Para el análisis y visualización de datos.
- *EXCEL*: Formato de los datos enviado por el *ISBSG,* para el filtrado inicial de los datos.

Herramientas de distribución libre

Las herramientas de este tipo utilizadas en el desarrollo de este trabajo vienen siendo utilizadas por el equipo de investigación multidisciplinar integrado por investigadores de las Áreas de Proyectos de Ingeniería, Matemática Aplicada, Expresión Gráfica en Ingeniería e Ingeniería Telemática de la Universidad de Oviedo, y por otros equipos de

investigación nacionales e internacionales que avalan la fiabilidad de los resultados obtenidos.

- *SOM Toolbox,* librería de Matlab desarrollada por la Universidad de Helsinki para la implementación y visualización de mapas SOM.
- *MARS,* versión libre distribuida por Jerome Friedman en código Fortran: Código desarrollado por Jerome Friedman para la implementación del algoritmo MARS. Se ha utilizado una versión de prueba comercial de MARS (restringida a unos días) para verificar los resultados obtenidos.

Herramientas de desarrollo de la Universidad de Oviedo

- *XDPM*: Herramienta desarrollada por el Grupo de Investigación en Proyectos de Ingeniería de la Universidad de Oviedo para la preparación y preproceso de datos.
- *APIMARS*: Herramienta desarrollada por el Grupo de Investigación en Proyectos de Ingeniería de la Universidad de Oviedo para la combinación de técnicas adaptativas y redes neuronales. La herramienta incluye la implementación en Fortran y C de una versión modificada del algoritmo MARS.

3. ANÁLISIS DE LOS DATOS

Dado que para abordar la solución del problema se ha decidido utilizar técnicas de minería de datos, en este apartado se centra el aspecto principal que es el conocimiento de los datos sobre los que se va a trabajar.

En este apartado se procede a familiarizarse con los datos, identificar su calidad e identificar las relaciones que definan unas primeras hipótesis.

3.1.- ADQUISICIÓN DE DATOS

En esta ocasión la fuente de datos es de una única procedencia, la base de datos del *International Software Benchmarking Standards Group* Release 9, por lo tanto el proceso de integración de las diferentes fuentes y orígenes de datos ya ha sido realizado por los expertos de métrica de sistemas de información pertenecientes a dicha organización.

Como almacén para la información se ha decidido utilizar un soporte de hoja de cálculo, debido a que el volumen de información es de 3.024 proyectos y 100 atributos, lo que

implica que no es necesario un soporte de base de datos para almacenar un volumen de información no excesivamente elevado.

Los proyectos de los que se han recopilado los datos cubren una amplia gama de productos de sistemas de información, aunque con un claro enfoque comercial.

Los datos se recopilan procedentes de proyectos que vienen de 20 países diferentes. Los colaboradores más importantes son Australia, Canadá, Reino Unido, India y Brasil.

Para contextualizar los datos se puede comentar que el tipo de organización de la cual proceden los datos varía desde empresas de comunicaciones hasta gobiernos, seguros y banca.

3.2.- DESCRIPCIÓN DE DATOS

La descripción detallada de los datos contenidos en la base del *ISBSG* se encuentra ya realizada en el Capítulo 4: Descripción de los datos.

3.3.- EXPLORACIÓN DE DATOS

3.3.1.- Inicial

Se ha procedido a realizar una exploración de los datos con el fin de extraer sus principales características, habiéndose obtenido la siguiente información:

Contexto del Proyecto

- Tipo de organización: los tipos importantes son las comunicaciones (26%), Administración Pública (11%), servicio comercial (10%), seguro (10%), fabricación (8%), y banca (7%).
- Área de negocio: las mayores áreas son telecomunicaciones (26%), banca (13%), seguros (13%), finanzas (9%), fabricación (8%), contabilidad (5%), ventas y marketing (4%) e ingeniería (4%).
- Ubicaciones de negocio: el 33% de los proyectos se utilizan en una sola ubicación, el 35% de 2 a 5 ubicaciones, y el 32% en más de 5 ubicaciones.
- Número de unidades de negocio: el 46% de los proyectos son empleados por una única unidad de negocio, el 38% de 2 a 5 unidades, y el 16% por más de 5 unidades de negocio.

Tipo de proyecto:

- El 57% son proyectos de desarrollo de aplicaciones que ya existían y debían ser mejoradas, el 41% son nuevos desarrollos, y el 2% son proyectos que desarrollan de nuevo en su totalidad.

Tipo de producto:

- Tipo de aplicación: el 16% son Sistemas de Información generales, el 22% son Sistemas de Transacción/Producción, el 24% son Sistemas de Finanzas o Banca y el 3% son Sistemas de Oficina de Información. Alrededor del 2% son Sistemas en Tiempo Real. El porcentaje restante se encuentra repartido entre otros tipos de aplicación.
- El 47% tiene una arquitectura cliente-servidor.
- En cuanto a usuarios concurrentes: el 14% son sistemas monousuario, el 10% soportan entre 2 y 5 usuarios concurrentes y el 76% soporta más de 5 usuarios concurrentes. De hecho, el 43% soporta a 10 o más usuarios concurrentes.

Medio de desarrollo:

- Lenguaje: se representan alrededor de 70 lenguajes de programación. 3GLs representa el 60% de los proyectos, 4GLs el 35%, y los generadores de aplicaciones el 5%. Los lenguajes más presentes son Cobol (14%), C/C++ (12%), Java/J2EE (8%), Visual Basic (13%), PL/I (6%), Oracle (5%), SQL (5%), Natural (3%), Cobol II (3%) y Telon (1%).
- Plataforma: el 60% son proyectos de ordenador central, el 17% de medio alcance, y el 23% microordenadores.

Métodos y herramientas de desarrollo:

- Metodología: el 69% de los proyectos utilizan una metodología estándar que fue desarrollada de forma interna y el 22% utiliza una metodología adquirida (de estos un 14% realizó adaptaciones). Sólo el 9% no sigue una metodología.
- El uso de herramientas CASE oscila entre el 18% de proyectos que usan herramientas de alto nivel, y el 9% para las herramientas CASE integradas. En el 27% de los proyectos se usan algún tipo de herramientas CASE.

- Las técnicas orientadas a objetos se usan en el 26% de los proyectos.
- El prototipo son empleados en el 20% de los proyectos.

3.3.2.- Correlaciones

Se procedió a realizar un análisis de correlaciones entre las variables numéricas para establecer las primeras hipótesis iniciales. La Fig. 21 muestra el gráfico de correlaciones elaborado.

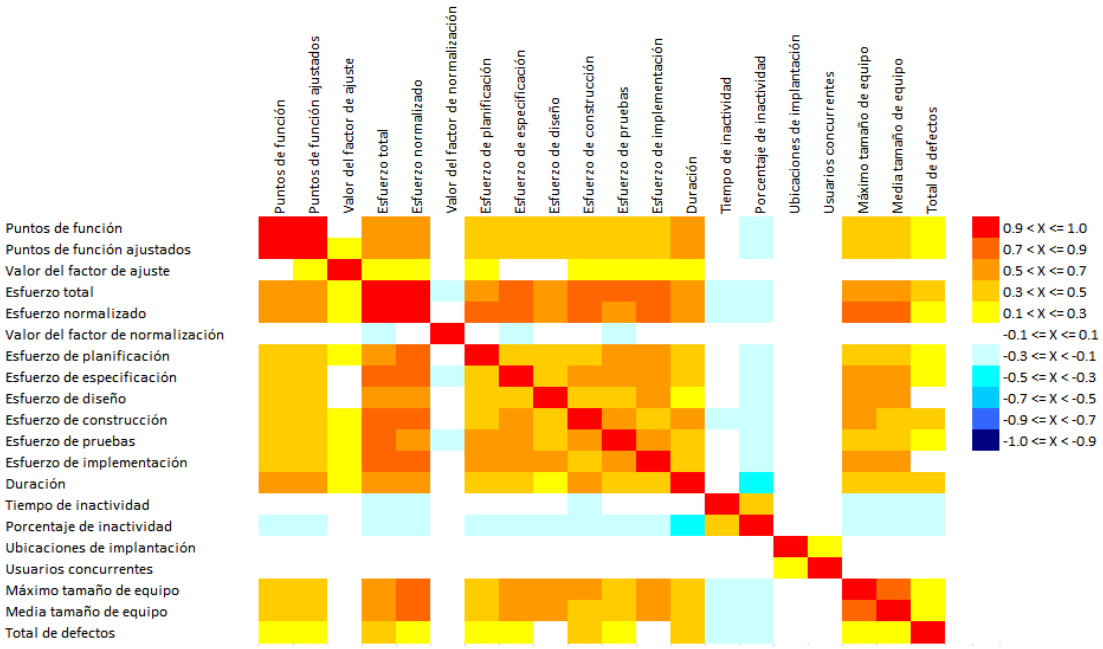


Fig. 21 Gráfico de correlaciones entre variables numéricas

En el gráfico de correlaciones se pueden observar las siguientes relaciones:

- Esfuerzo detallado por fases.

  Existe una altísima correlación entre el esfuerzo realizado en la fase de especificación y el esfuerzo total realizado. Esta correlación existe también con el esfuerzo realizado en las fases de construcción, pruebas e implementación, no habiendo una relación tan fuerte del esfuerzo total realizado con los esfuerzos realizados en las fases de planificación y diseño.

  Existe una alta correlación entre el esfuerzo realizado en la fase de planificación y el esfuerzo realizado en las fases de pruebas e implementación. Así mismo, el esfuerzo realizado en la fase de especificación está íntimamente ligado a los esfuerzos realizados en las fases de construcción, pruebas e implementación.

- Esfuerzo.

  El esfuerzo total realizado está altamente relacionado con el número de puntos de función de la aplicación, así como con el tamaño del equipo de desarrollo.

- Tamaño del equipo de desarrollo.

  Existe una relación alta entre el equipo de desarrollo y el esfuerzo realizado, y en particular con el esfuerzo dedicado a las fases de especificación, diseño, construcción e implementación.

- Número total de defectos.

  Existe correlación entre el número total de defectos encontrados con el esfuerzo realizado, y de forma específica con el esfuerzo dedicado a la fase de construcción. El mismo nivel de correlación se da con la duración total del proyecto.

3.3.3.- Valores perdidos

Una vez realizado el análisis de las relaciones entre las variables se ha procedido a realizar un análisis de la presencia de valores perdidos o ausencia de información.

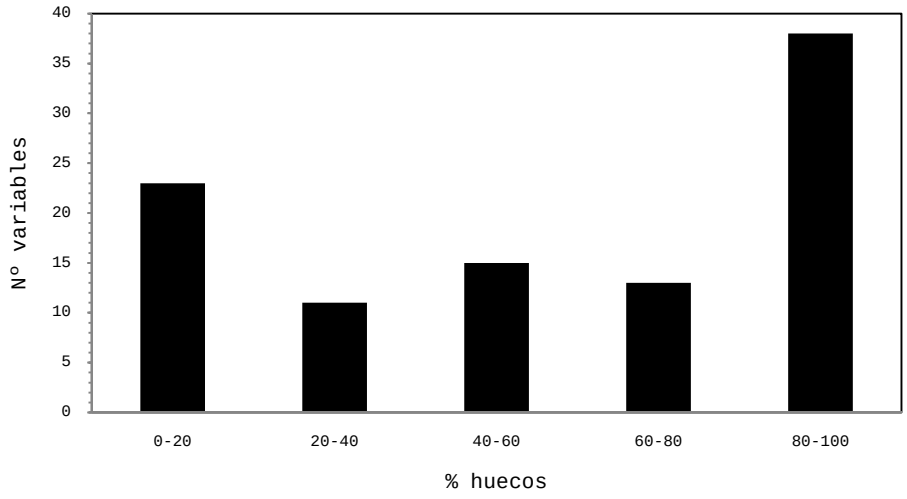


Fig. 22 Distribución de las variables según % de valores perdidos

La Fig. 22 muestra el número de variables que presentan distintos intervalos de porcentaje de huecos o valores perdidos. Se detecta una alta presencia de valores perdidos o falta de información dentro de las variables de la base de datos. Únicamente hay 5 variables que contienen toda la información respecto a todos los proyectos

recogidos, mientras que 77 variables tienen pérdidas de información superiores a un 20% de los casos. En la fase de preparación de datos se ha realizado un estudio considerando diferentes alternativas para el tratamiento de los datos vacíos.

3.3.4.- Relaciones no lineales

Para la visualización multidimensional de los datos se han utilizado mapas auto-organizativos SOM. Este método de visualización permite identificar agrupaciones de datos correspondientes a proyectos con características similares, así como encontrar relaciones no lineales dentro del conjunto de variables en exploración.

La Fig. 23 muestra un mapa SOM generado con los datos de trabajo.

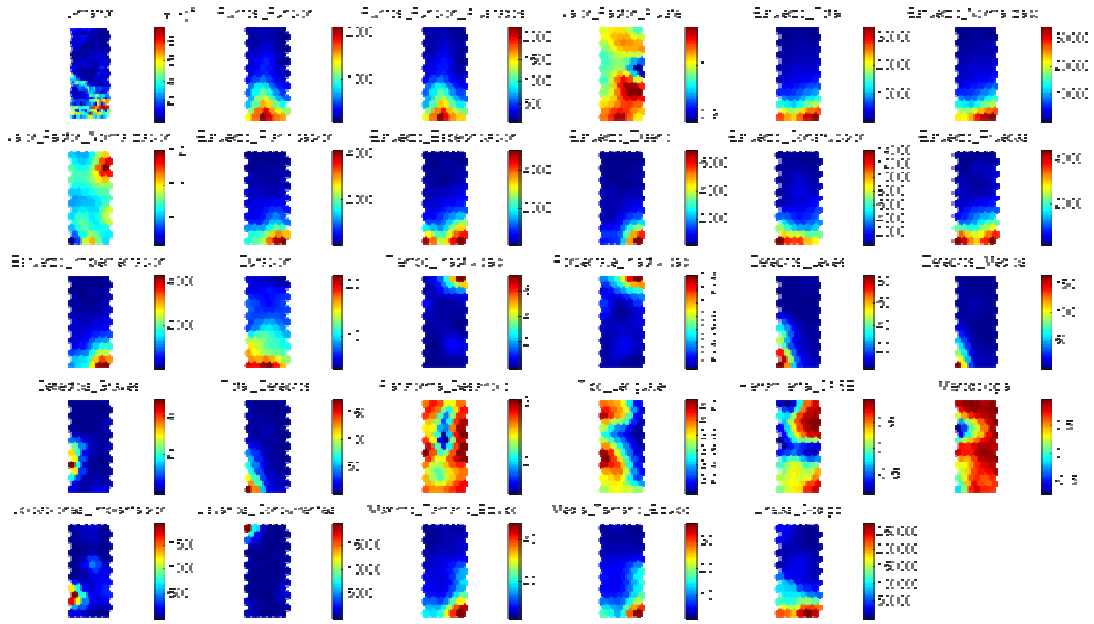


Fig. 23 Mapa SOM con las variables del problema

La utilización del mapa SOM valida algunas de las relaciones existentes ya conocidas previamente entre las variables tales como cierta relación entre el número total de defectos encontrados en la aplicación, el esfuerzo total realizado y el tiempo de duración del proyecto. De igual forma, esfuerzos altos en la fase de especificación coinciden con esfuerzos totales altos. Se cumple así mismo que tamaños de equipo de desarrollo elevados implican esfuerzos totales altos en el desarrollo del proyecto y respecto a los objetivos principales de este trabajo, se puede observar claramente que los proyectos en los que se ha realizado un gran esfuerzo en la fase de construcción presentan un elevado número de defectos totales encontrados.

3.4.- VERIFICACIÓN DE LA CALIDAD DE LOS DATOS

En este apartado se presentan las conclusiones derivadas de la exploración inicial de los datos, así como un análisis de los problemas que deberán de ser tratados en la siguiente fase:

- Presencia de *outliers* que pueden provocar la distorsión de los datos o del modelo que se genere, para lo cual se ha de analizar si se eliminan esos proyectos por considerarlos casos aislados para alguna de las variables o si se genera un nuevo modelo para ese subconjunto de proyectos.
- La gran cantidad de valores perdidos que presentan la mayoría de las variables, por lo que se ha de preparar un preproceso especial para este tipo de datos. Solucionar esta carencia de información es un paso necesario para conseguir un modelo fiable.
- Presencia de un elevado número de variables categóricas. Se ha de analizar cómo tratar estas variables para poder introducir la información contenida en dichas variables en la fase de modelado.

4. PREPARACIÓN DE LOS DATOS

En esta fase se ha procedido a la preparación de los datos de tal forma que puedan ser procesados por las técnicas adaptativas y redes neuronales, seleccionadas para el desarrollo de los modelos.

Las diferentes características de las técnicas implican que la preparación de los datos sea ligeramente distinta para su tratamiento mediante técnicas adaptativas que mediante redes neuronales, necesitando una menor preparación de los datos para ser utilizados por las técnicas adaptativas.

Ambas técnicas utilizan algoritmos basados en minimización de errores cuadráticos, por lo que resulta necesaria la identificación previa de valores no usuales asociados a errores en la inserción o medida. La identificación de los valores no usuales se realizará mediante la aplicación de técnicas de proyección no lineal (Proyección de Sammon) [Sammon 1969]. Para la aplicación de este tipo de proyecciones será necesario a su vez proceder a una preparación de datos que permita el tratamiento de los datos por el algoritmo.

Otras tareas necesarias en la preparación de los datos para las redes neuronales y las técnicas adaptativas son el rellenado de datos incompletos, la adaptación de las variables categóricas y la normalización o escalado de los datos. Esta última tarea se ha implementado en el algoritmo APIMARS para realizarse de forma automática.

4.1.- PREPROCESADO DE DATOS

Sobre los 3.024 proyectos de la base de datos de partida, se ha realizado un meticuloso preproceso con el objetivo de obtener un conjunto de datos de trabajo comparable, de calidad, sin valores perdidos y sin presencia de espúreos. Esta tarea es necesaria en cualquier proyecto de Minería de Datos para obtener buenos resultados y llegar a conclusiones válidas y fiables.

Se procede a continuación a explicar los pasos realizados en el preproceso de datos:

- Filtrado de datos con el objeto de obtener un conjunto de proyectos comparables.
- Eliminación de variables categóricas innecesarias y conversión de variables categóricas ordinales a variables numéricas.
- Preproceso de espúreos (*outliers*).
- Rellenado de huecos o valores perdidos.

4.1.1.- Filtrado de datos

De acuerdo a lo comentado en el Capítulo cuatro, apartado 3, sobre cómo deben usarse los datos del *ISBSG* y, en particular, en lo referente a obtener un conjunto de datos comparable, se realizó un filtrado seleccionando exclusivamente aquellos casos en los que los datos se consideren de alta calidad, al igual que su caracterización en puntos de función (valores 'A' o 'B' en las variables *Data Quality Rating* y *UFP Rating*). Este filtrado asegura que el conjunto de proyectos resultante presenta buena calidad e integridad en sus distintas variables, así como en el cálculo de los puntos de función en los proyectos seleccionados, dando lugar a un conjunto de datos formado por 2.179 proyectos.

Con el fin de mantener esa homogeneidad, se escogieron además los proyectos caracterizados mediante puntos de función clásicos (*Count Approach* presenta el valor 'IFPUG'), de forma que se asegura que el tamaño de todos los proyectos seleccionados

ha sido medido con la misma técnica y siguiendo los mismos criterios. El conjunto de datos resultante está formado por 2.150 proyectos.

Se procedió a eliminar los proyectos con duración desconocida, dada la importancia del plazo que se considera uno de los tres factores fundamentales, como ya se mencionó anteriormente (*Project Elapsed Time* desconocida), y la imposibilidad de obtener conclusiones acerca de los posibles riesgos existentes relacionados con la gestión de plazos del proyecto. Este filtrado dio lugar a un conjunto de datos compuesto por 1.840 proyectos.

El último filtrado realizado fue eliminar los proyectos en los que se desconociese el número total de defectos encontrados debido a que eso implica de forma automática que se desconoce el número de defectos de cualquier nivel: leves, medios y graves. El conjunto de datos resultante lo componen 303 proyectos.

4.1.2.- Preproceso de variables categóricas

Las variables categóricas o cualitativas son aquellas que no aparecen en forma numérica, sino como categorías o atributos como, por ejemplo, el tipo de lenguaje de programación de un proyecto. En dichas categorías puede haber un orden subyacente (variable ordinal) o no (variable nominal), es decir que las variables categóricas se pueden clasificar en los siguientes grupos:

- **Nominal**, en la que estarán las variables de categorías no numéricas sin relación de orden.
- **Ordinal**, en las que estarán las variables categóricas no numéricas con una relación de orden.

Las variables categóricas nominales contenidas en la base de datos utilizada para este trabajo han sido descartadas de cara a ser tenidas en cuenta en los modelos matemáticos. Dichas variables aportan información contextual a los proyectos e informan acerca de la forma en que los datos han sido recopilados o qué técnicas se han usado para medir otras variables. Una vez que esta información ha sido útil para realizar un primer filtrado, se descartan por la imposibilidad de ser manejadas de forma numérica respecto al número de datos disponibles.

Las variables categóricas ordinales se han sustituido por valores numéricos y se han empleado en los modelos desarrollados.

La Tabla 14 muestra las variables categóricas seleccionadas y los valores por los que se ha sustituido su contenido. La descripción completa y exhaustiva de las variables de la base de datos puede consultarse en el capítulo cuatro de este trabajo.

Nombre	Valores
Plataforma_Desarrollo	PC(1), MR(2), MF(3), Multi (4)
Tipo_Lenguaje	2GL(2), 3GL(3), 4GL(4) y ApG(5)
Herramienta_CASE	Sí(1), Desconocido(0), No(-1)
Metodología	Sí(1), Desconocido(0), No(-1)

Tabla 14 Variables cualitativas ordinales seleccionadas

4.1.3.- Preproceso de valores anómalos (*outliers*)

El estudio de los valores anómalos se realizó en dos fases:

- Análisis directo de las parejas de variables *Functional Size* (Puntos de función) y *Adjusted Function Points* (Puntos de función ajustados), *Summary Work Effort* (Esfuerzo total) y *Normalised Work Effort* (Esfuerzo normalizado) y estudio directo de la variable *Max Team Size* (Tamaño máximo de equipo).
- Realizando una proyección del conjunto de datos n-dimensional a dos dimensiones manteniendo las relaciones métricas para detectar proyectos lejanos teniendo en cuenta el conjunto de variables.

La Fig. 24 muestra la relación entre los puntos de función sin ajustar y los ajustados. Se observa claramente la relación directa entre ambas variables con un coeficiente de correlación R^2=0.99. Asimismo todos los proyectos contienen un valor en estas variables menor que 6.000 excepto uno que se ve alejado pero manteniendo la relación. Se ha excluido este proyecto del conjunto de datos final por la lejanía del mismo en tamaño funcional manteniendo en el conjunto de datos en estudio todos los proyectos agrupados.

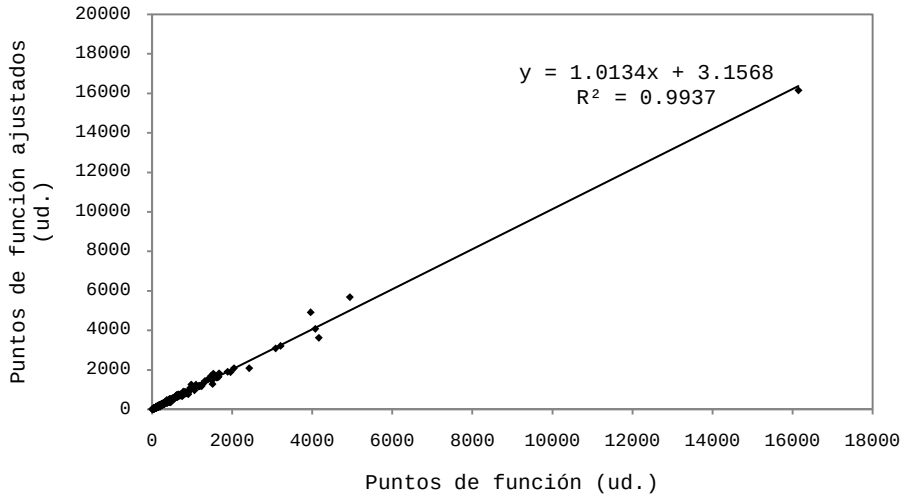


Fig. 24 Relación entre los puntos de función ajustados y sin ajustar

La Fig. 25 muestra la relación entre el esfuerzo total empleado en el proyecto en horas frente al esfuerzo normalizado. De nuevo se observa una relación muy alta entre ambas variables con un coeficiente de correlación R^2=0.95. Se ha procedido a eliminar del conjunto de datos final el proyecto aislado con un valor de esfuerzo igual a 150.040 horas y mantener el resto de los proyectos con esfuerzos menores de 7.000 horas y que se encuentran claramente agrupados.

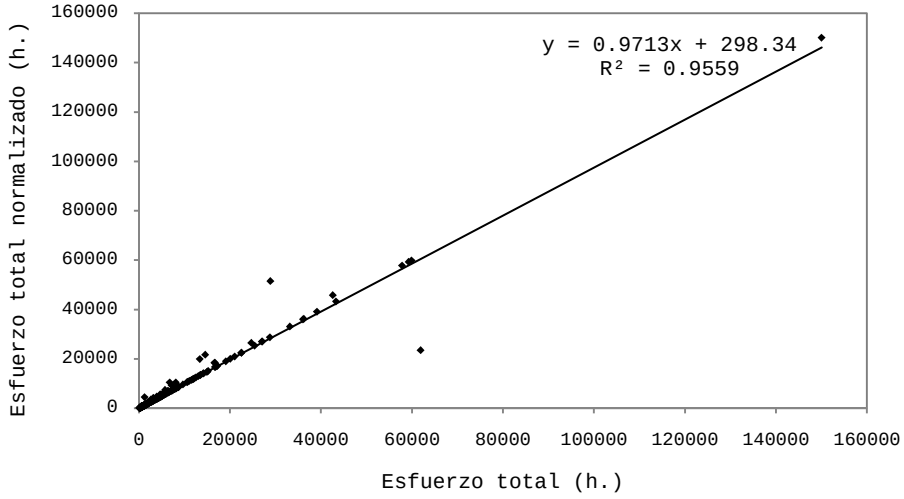


Fig. 25 Relación entre el esfuerzo total y el normalizado

La Fig. 26 muestra el tamaño máximo del equipo de desarrollo en los proyectos que conforman el conjunto de datos en estudio hasta el momento. Se puede observar la existencia de un proyecto con un tamaño máximo de equipo claramente superior al resto (309 personas). Observando en detalle el resto de variables asociadas a este

proyecto y teniendo en cuenta los objetivos de este estudio, se descartó este proyecto debido a que el número de defectos encontrados en esta aplicación fue 0 (defectos medios) y el número de defectos leves y graves que se habían producido son desconocidos.

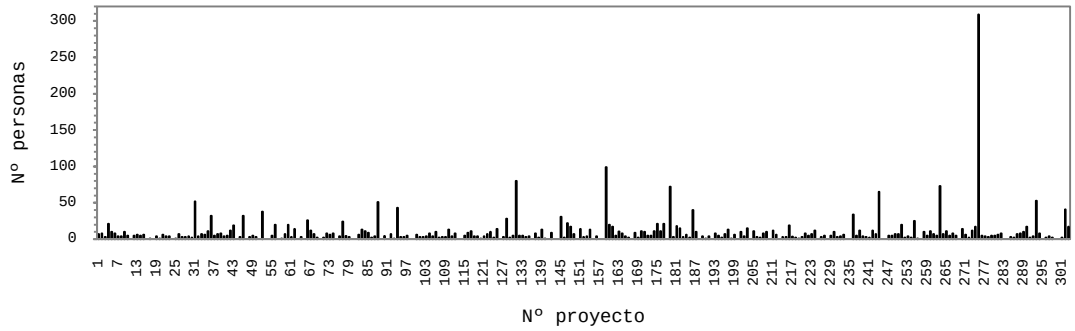


Fig. 26 Tamaño máximo del equipo de desarrollo

Antes de tomar la decisión de cuál es la acción que se ha de aplicar a los valores anómalos se procesará los datos con un algoritmo que proyecta en dos dimensiones un conjunto de datos *n*-dimensional, manteniendo las relaciones métricas, es decir, los puntos que estén cerca en *n*-dimensiones deben estar también cerca en 2 dimensiones.

Para utilizar una técnica de proyección n-dimensional basada en distancias como Sammon ha sido necesario previamente realizar una preparación de los datos que permita su tratamiento por dicha técnica.

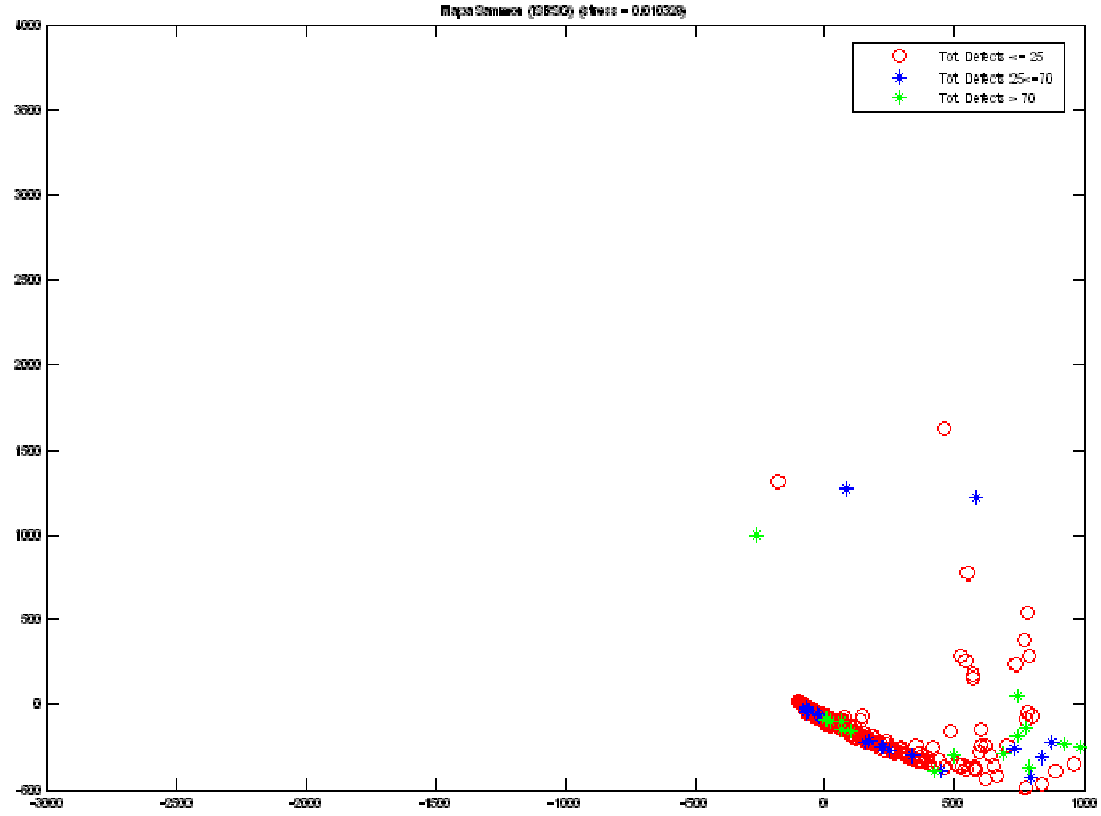


Fig. 27 Proyección Sammon del espacio n-dimensional a dos dimensiones del conjunto de datos en estudio

En la representación gráfica de la proyección Sammon se ha utilizado una gama de colores para identificar los proyectos en función del número total de defectos encontrados según los siguientes intervalos:

- Rojo: Menor o igual que 25 defectos.
- Azul: Entre 26 y 70 defectos.
- Verde: Más de 70 defectos.

En la Fig. 27 se puede observar que la mayoría de los proyectos están agrupados y la distancia existente a los proyectos ligeramente separados no es grande, por lo que se puede deducir que no existen valores anómalos en el conjunto de datos manejado en este momento.

4.1.4.- Preprocesado de valores ausentes

La presencia de valores ausentes es un problema grave ya que la primera de las alternativas a la hora de afrontarlo consiste en eliminar o rechazar los casos que presenten este problema en alguna de las variables o eliminar del modelo las variables

que no disponga de la calidad suficiente para abordarlo. Este método de afrontar los valores ausentes provocaría que sólo dispusiésemos de un caso con todos los datos. Vista la inviabilidad del método anterior se procede a estudiar otras alternativas para el procesado de este problema.

Antes de eliminar registros o variables hay que analizar si la ausencia de dato para ese campo nos proporciona en sí información, ya que la ausencia de valor puede tener significado en sí mismo, es decir, que no se quiera rellenar un valor puede ser tan significativo como la respuesta en sí misma.

El primer paso es analizar cuál pudo haber sido la causa de la ausencia de información: no respuesta por parte del encuestado, fallo en la trascripción, etc. Pero también se ha de revisar si la ausencia de datos es al azar o no ignorable. En este caso de estudio la presencia de valores vacíos en las variables de la base de datos del ISBSG es debido a la ausencia de medición o al desconocimiento del valor de la variable para el proyecto en cuestión, por lo que la presencia de huecos no es fruto del azar y es considerada no ignorable.

Para el rellenado de valores vacíos en las variables se consideraron las siguientes alternativas, en función del tipo de variable, su significado y si su valor es dependiente o no de los valores de otras variables:

- Existen variables cuyos valores deben sumar conjuntamente el valor de otra variable, por lo que se puede calcular su valor en función del número de huecos presentes entre todas ellas y el valor a sumar.
- Para las variables donde no ocurra lo anterior, buscar el conjunto de muestras más semejantes al caso en estudio, reemplazando el valor perdido en función de los valores tomados en esa variable por el conjunto de muestras semejantes.

En el trabajo que se desarrolla se han aplicado dos de las técnicas que se han comentado anteriormente.

4.1.5.- Resultado del preprocesado de los datos

Como resultado de la ejecución de los pasos anteriores se ha realizado la selección, limpieza, integrado y formateado de datos según los criterios que se indican a continuación.

En relación al proceso de filtrado de valores anómalos, se han explicado en el apartado anterior los criterios para seleccionar los casos y para tomar la decisión de filtrarlos de la muestra, así como con el proceso seguido por las variables categóricas y los valores numéricos que se han escogido para asignar a las distintas categorías ordinales. En este apartado se presenta con detalle el proceso seguido con las variables que presentaban valores perdidos.

La base de datos del ISBSG contiene un conjunto de variables relacionadas con el esfuerzo empleado en el desarrollo de los proyectos. Dispone de una variable que representa el esfuerzo total y de seis variables que son este mismo esfuerzo desagregado para las fases de planificación, especificación del sistema, diseño, construcción, pruebas e implantación. El perfil de esfuerzo que puede tener un proyecto a lo largo del ciclo de vida depende mucho del tipo de tareas que contenga. Este reparto de esfuerzo a lo largo de las tareas podría agruparse en uno de los siguientes perfiles: Uniforme  , creciente  , decreciente  , de campana  , de campana achatada  , pico inicial  , pico final  , dos picos  , personal  . Dado que los proyectos almacenados en la base de datos son muy distintos entre sí, se ha optado por el perfil uniforme, repartiendo la carga uniformemente entre las fases en las que no se dispone de información. Esta suposición se ha utilizado para rellenar los valores ausentes en aquellos casos donde no existía dato para el esfuerzo por fases, desagregando el esfuerzo total.

Según el criterio seguido para modelizar el nivel de riesgo a partir de la información disponible al inicio del proyecto, se ha realizado una selección entre las variables candidatas, concluyendo la existencia de 28 posibles variables influyentes (Tabla 15).

Entrada	Tipo	Descripción
Puntos_Funcion	Numérica	Puntos de función
Puntos_Funcion_Ajustados	Numérica	Puntos de función ajustados
Valor_Factor_Ajuste	Numérica	Valor de ajuste de los puntos de función
Esfuerzo_Total	Numérica	Esfuerzo total
Esfuerzo_Normalizado	Numérica	Esfuerzo total normalizado
Valor_Factor_Normalizacion	Numérica	Factor de normalización del esfuerzo
Esfuerzo_Planificacion	Numérica	Esfuerzo en fase de planificación
Esfuerzo_Especificacion	Numérica	Esfuerzo en fase de especificación
Esfuerzo_Diseno	Numérica	Esfuerzo en fase de diseño
Esfuerzo_Construccion	Numérica	Esfuerzo en fase de construcción
Esfuerzo_Pruebas	Numérica	Esfuerzo en fase de pruebas
Esfuerzo_Implementacion	Numérica	Esfuerzo en fase de implementación
Duracion	Numérica	Tiempo planificado para el desarrollo
Tiempo_Inactividad	Numérica	Tiempo inactivo del proyecto

Entrada	Tipo	Descripción
Porcentaje_Inactividad	Numérica	Porcentaje de tiempo inactivo
Defectos_Leves	Numérica	Defectos leves
Defectos_Medios	Numérica	Defectos medios
Defectos_Graves	Numérica	Defectos graves
Total_Defectos	Numérica	Total de defectos
Plataforma_Desarrollo	Categórica	Plataforma (PC, Mainframe, etc.)
Tipo_Lenguaje	Categórica	Tipo de lenguaje de programación
Herramienta_CASE	Categórica	Se usó una herramienta CASE
Metodologia	Categórica	Se usó una metodología
Ubicaciones_Implantacion	Numérica	Sedes para implantación
Usuarios_Concurrentes	Numérica	Usuarios Concurrentes
Maximo_Tamano_Equipo	Numérica	Máximo tamaño del equipo
Media_Tamano_Equipo	Numérica	Tamaño medio del equipo
Lineas_Codigo	Numérica	Líneas de código

Tabla 15 Listado de variables seleccionadas para preprocesar.

No todas las variables que se encuentran en la Tabla 15 serán utilizadas para modelar, pero se han añadido para el preprocesado, puesto que a priori su influencia es desconocida y, en todo caso, pueden servir para la clasificación en grupos o detección de problemas en los resultados.

Entre las variables preseleccionadas se ha eliminado alguna debido a la baja presencia de datos, es decir muchos valores ausentes. Este es el caso de *Lineas_Codigo* que presentaba un 86% de valores vacíos.

Existen dos variables, *Valor_Factor_Ajuste* y *Valor_Factor_Normalizacion* cuyos valores son obtenidos como el cociente de otras dos, en particular de *Puntos_Funcion* y *Puntos_Funcion_Ajustados* para el primer caso y de *Esfuerzo_Total* y *Esfuerzo_Normalizado* para el segundo. Las cuatro variables que se usan para calcular las otras dos, tomadas de dos en dos presentan valores en todas las muestras por lo que los huecos presentados por las variables calculadas fueron rellenados de forma directa.

Para rellenar los valores ausentes de las variables que no son parte de una desagregación ni calculados a partir de los valores de otras variables, se ha seguido el siguiente procedimiento:

1. Estimar un tamaño óptimo de clúster para un mapa SOM con la representación de las variables en estudio.
2. Ejecutar el algoritmo para obtener un mapa SOM ajustado a ese número de clústers añadiendo el código necesario para generar en la salida un fichero .csv

que identifique qué proyectos fueron agrupados en cada neurona (proyectos con características similares).

3. Para cada conjunto de proyectos similares pertenecientes a la misma neurona, ejecutar el algoritmo k-medias obteniendo un vector que contiene en cada posición el valor a insertar en los valores ausentes de cada variable.

Como resultado del paso 1 se obtiene el mapa SOM mostrado en la Fig. 28. en el cual se puede observar que el número óptimo de clústers es 19.

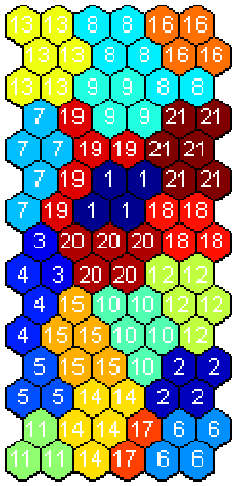


Fig. 28 Número óptimo de clústers para el conjunto de datos en estudio

A continuación se obtiene otro mapa SOM obligando a que el número de neuronas sea lo más parecido posible a 19, que sería el óptimo. Se definió un tamaño de mapa SOM formado por 5x4 neuronas, obteniendo de esta forma 20 grupos de proyectos similares. Como último paso se ejecutó el algoritmo k-medias sobre cada uno de los grupos obteniendo un vector representativo de cada grupo y utilizando los valores del vector para rellenar los valores ausentes. El conjunto de datos resultante sin valores vacíos puede ser procesado por los algoritmos de modelado utilizados en este trabajo.

5. MODELADO DEL NIVEL DE RIESGO

Tal como se explicó en el capítulo 3, en este trabajo se pretende utilizar las técnicas más prometedoras en la modelización de problemas complejos no lineales, concretamente redes neuronales o MARS, aprovechando las características distintivas de cada una de ellas y comparando o combinando los resultados producidos por ellas. Se realiza también una comparación con los mecanismos tradicionales de estimación y, concretamente con un modelo de regresión multivariante.

Dentro de las redes neuronales se opta por redes de tipo perceptrón multicapa, con algoritmos de tipo retro-propagación. Para la clasificación se utilizan además algoritmos no supervisados.

Finalmente, tras las transformaciones previamente explicadas que implican el filtrado y tratamiento de *outliers*, la eliminación de variables, sustitución de valores de las variables categóricas ordinales por valores numéricos y relleno de valores ausentes, se obtiene un conjunto de datos con 23 variables de entrada y 300 casos, así como 4 variables de salida, una para cada tipo de defecto (leves, medios y graves) y el número total de defectos presentados por el proyecto. Es importante que la base de datos disponga del número de defectos de los proyectos desglosado según sea la importancia de los mismos. Un modelo generalista que no discriminase entre los distintos tipos de defectos sería incongruente, la salida proporcionada podría ser idéntica a la hora de estimar los defectos en dos proyectos totalmente distintos, uno con cinco defectos leves y otro con cinco defectos graves. La discriminación de los resultados obtenidos por los modelos elaborados en este trabajo permitirá realizar una mejor planificación del proyecto e intentar paliar de forma preventiva los posibles defectos de las aplicaciones, mejorando drásticamente la calidad de los programas desarrollados.

Los siguientes apartados muestran los pasos realizados para el modelado tomando como variable de salida el número total de defectos encontrados, siendo análogos para las otras tres variables de salida. Al final del capítulo se muestran los resultados obtenidos por los modelos para cada una de las variables de salida.

5.1.- MODELADO MEDIANTE RED NEURONAL

Debido al problema de la dimensionalidad, no es posible introducir un número cualquiera de entradas si se pretende conseguir un modelo que sea capaz de generalizar, puesto que el número de parámetros no puede jamás exceder la mitad del número de datos o muestras disponibles. Es por ello necesario seleccionar dentro de las variables preprocesadas aquellas que resultan más prometedoras.

Para reducir este número de variables se han utilizado técnicas conjuntas antes del entrenamiento, como MARS (específicamente su modificación API-MARS) y SOM debido a que su efectividad está suficientemente demostrada en aplicaciones previas en diversos sectores [Rodríguez 2003].

El siguiente paso será seleccionar la configuración de los parámetros y la topología de la red.

5.1.1.- Reducción de la dimensionalidad

Siguiendo los pasos marcados en la metodología y utilizando combinaciones de técnicas cuya fiabilidad ha sido demostrada en otros trabajos de minería de datos se procede a reducir la dimensionalidad del conjunto de datos utilizando estrategias de selección de variables mediante la combinación de técnicas adaptativas y redes SOM.

La selección de variables se realiza mediante el seguimiento de una estrategia de poda iterativa. Para ello se parte del conjunto de datos formado por las 23 variables seleccionadas en las fases previas.

Este conjunto inicial de datos se divide aleatoriamente en dos conjuntos independientes:

- El conjunto de datos de entrenamiento formado por el 85% de los datos. Este conjunto será utilizado en el paso hacia delante del algoritmo MARS para la construcción del modelo
- El conjunto de datos de prueba formado por el 15% de los datos restantes. Este conjunto será utilizado, junto con el conjunto anterior, en la versión modificada del paso hacia atrás del algoritmo MARS (Capítulo 3).

Una vez fijado el número de interacciones entre variables, se realiza el paso hacia delante del algoritmo MARS para un número de funciones base suficientemente grande. La importancia relativa de las variables dentro de cada modelo se calcula a partir de la pérdida de precisión que sufre el modelo al eliminar la variable.

Finalmente, la importancia relativa de cada variable se calcula como el valor medio de la importancia relativa de las variables en cada modelo, ponderado por el grado de bondad del modelo. Las variables con importancia relativa ponderada inferior al 5% se eliminan del conjunto, tras verificar con mapas SOM que la información contenida en ellas es muy bajo o nulo.

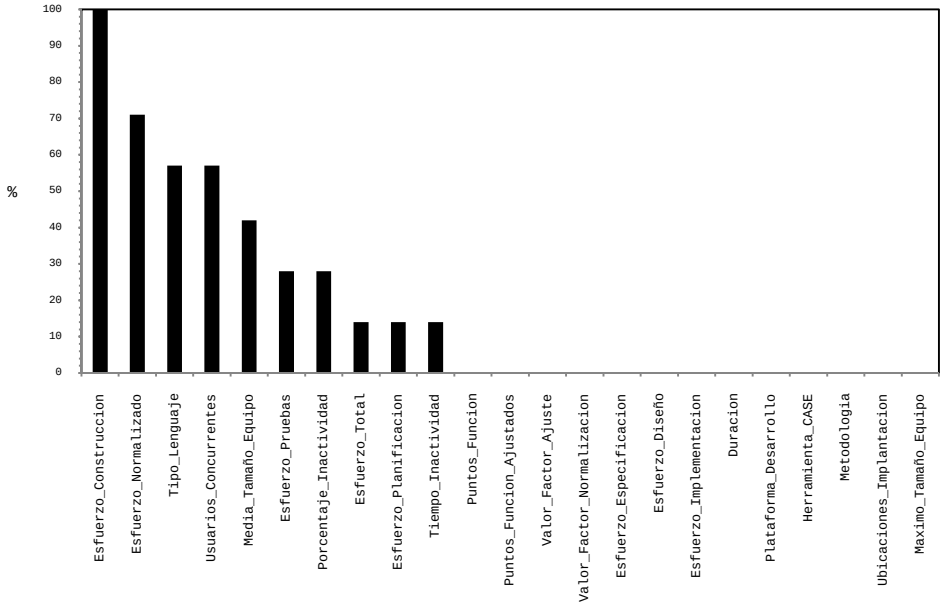


Fig. 29 Importancia relativa de las variables según MARS

En la figura anterior se muestra la importancia relativa de las entradas significativas según el método MARS. Tras su aplicación se procede a comprobar que las variables eliminadas no aportaban información relevante al modelo.

En la siguiente figura se muestra el mapa SOM con todas las variables antes de eliminar las que no son significativas para MARS. Con la eliminación de las variables no relevantes, los mapas SOM han ido transformándose mostrando paulatinamente información que originalmente se encontraba oculta por la información no relevante.

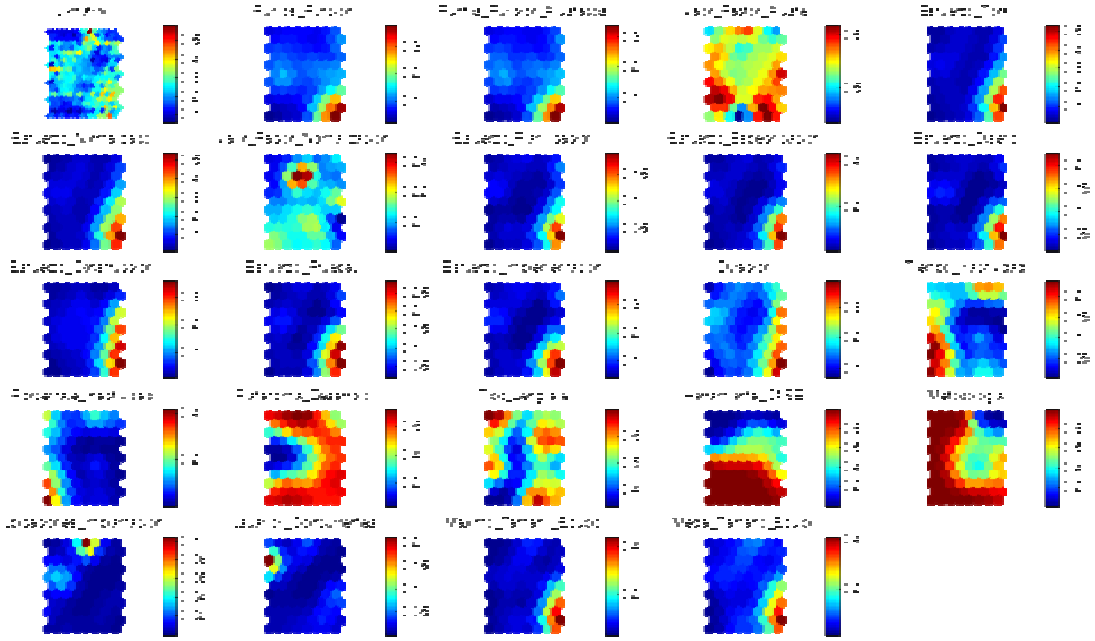


Fig. 30 Mapa SOM antes de la poda de las variables no significativas

Una vez eliminadas las variables que no tienen alta importancia relativa para el modelo el conjunto de variables restantes son *Esfuerzo_Construccion, Esfuerzo_Normalizado, Tipo_Lenguaje, Usuarios_Concurrentes, Media_Tamano_Equipo, Esfuerzo_pruebas, Porcentaje_Inactividad, Esfuerzo_Total, Esfuerzo_Planificacion* y *Tiempo_Inactividad* obteniendo el mapa SOM resultante mostrado en la siguiente figura.

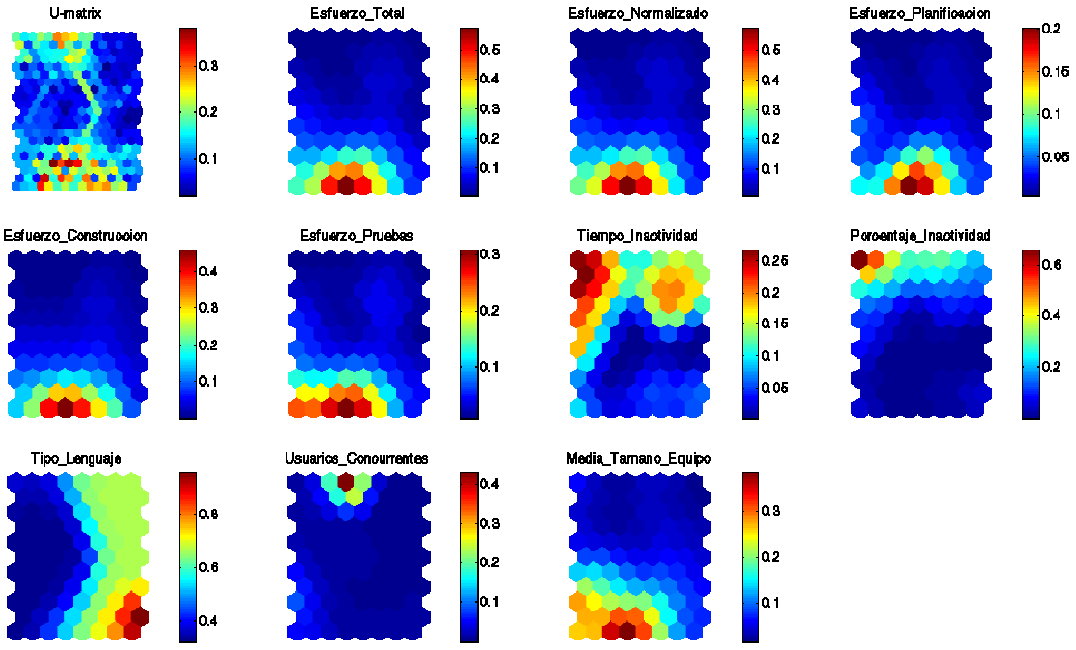


Fig. 31 Mapa SOM con variables significativas

Esta información sirve de base para el entrenamiento de la red neuronal.

5.1.2.- Modelo BackPropagation Momentum

Con las entradas anteriormente seleccionadas se procede a entrenar una red con algoritmo de aprendizaje BackPropagation Momentum.

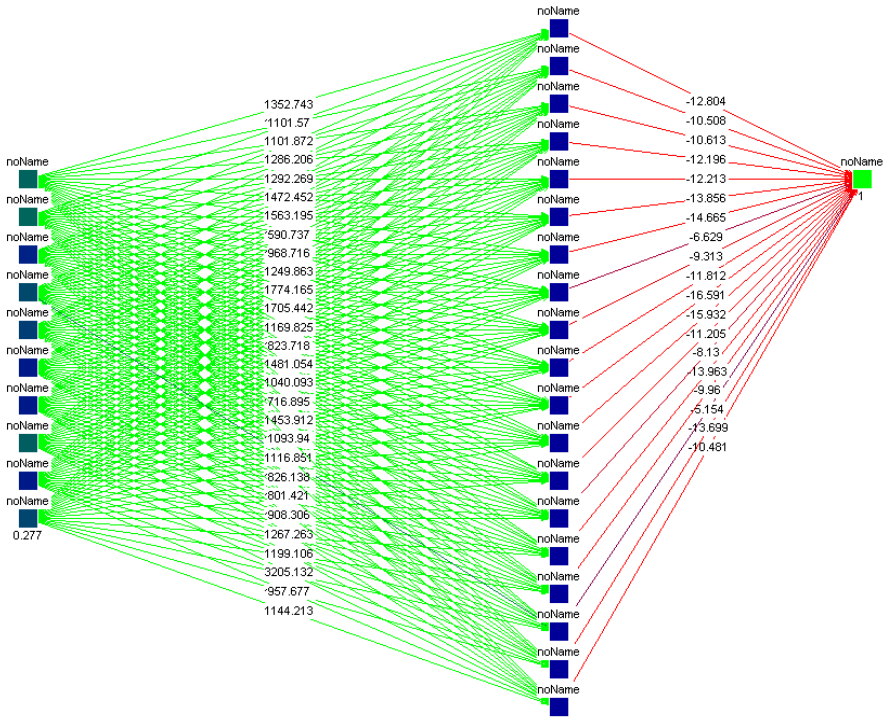


Fig. 32 Topología de la red generada por JavaNNS

En relación con la topología de la red, ésta está formada por la capa de entrada con 10 neuronas, una por cada variable de entrada, una capa oculta formada por 2n-1 neuronas, siendo n=10 el número de variables y una capa de salida formada por una neurona que es la variable de salida.

Se ha procedido a realizar el entrenamiento del modelo neuronal, ajustando los parámetros de entrenamiento, comenzando con valores altos de momentum en los primeros ciclos y valores más bajos en los ciclos finales.

A lo largo del entrenamiento de la red se han ajustado los parámetros de momentum y de entrenamiento. Los resultados conseguidos por este modelo se muestran: en la Tabla 16.

Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	10%	6%
3	23%	19%
5	37%	32%
7	45%	41%
9	57%	49%
11	61%	52%
13	69%	56%
15	73%	61%
17	*74%*	62%

Tabla 16 Porcentaje de aciertos para modelo con red neuronal y variable de salida nº total de defectos

La red neuronal aporta resultados suficientemente buenos a partir de un margen de error absoluto igual a 13.

Los resultados de los modelos correspondientes a las restantes variables de salida, defectos leves, medios y graves se encuentran en el apartado 6.

5.2.- MODELO MARS

Los modelos con redes neuronales plantean una serie de problemas conocidos que han sido descritos en el capítulo 3. Gran parte de estos problemas han sido eliminados por el exhaustivo preproceso al que han sido sometidos los datos y por la limitación de las entradas de acuerdo a las variables seleccionadas con las técnicas SOM y MARS. No obstante la utilización de algoritmos MARS para la modelización puede, en estos casos en que existen pocas variables, superar los resultados de las redes.

Se realiza por tanto modelización con técnicas adaptativas (versión modificada APIMARS del algoritmo MARS) como herramienta de apoyo para la resolución de las anteriores dificultades y también como técnica para la obtención de un modelo paramétrico. Se utilizan interrelaciones de variables hasta de nivel ternario y funciones base de segundo grado. Los resultados obtenidos por este modelo se muestran en la siguiente tabla.

Error absoluto	%media	%aciertos entrenamiento	%aciertos prueba
1	0%	8%	6%
3	4%	26%	21%
5	6%	40%	33%
7	11%	51%	42%
9	16%	60%	51%
11	21%	66%	56%
13	27%	72%	62%
15	38%	76%	67%
17	51%	79%	69%

Tabla 17 Porcentaje de aciertos para modelo con variable de salida nº total de defectos

El error cuadrático medio es 17,14.

6. EVALUACIÓN DE LOS RESULTADOS

Para comprobar la bondad de los modelos basados en técnicas híbridas multivariantes, se han comparado los resultados obtenidos por los modelos basados en redes neuronales y los modelos basados en MARS con el modelo de regresión. Esta comparativa de resultados se muestra en las siguientes tablas y figuras para las distintas variables de salida: *Total_Defectos, Defectos_Leves, Defectos_Medios* y *Defectos_Graves*. Se añaden también los resultados obtenidos por un modelo que devuelve, en cada caso, la media de los valores tomados por la variable de salida.

6.1.- MODELO CON VARIABLE DE SALIDA TOTAL_DEFECTOS

Modelo de Referencia		
Error absoluto	%aciertos	
1	2%	
3	13%	
5	20%	
7	27%	
9	36%	
11	42%	
13	47%	
15	53%	
17	56%	
Modelo Media		
Error absoluto	%aciertos	
1	0%	
3	4%	
5	6%	
7	11%	
9	16%	
11	21%	
13	27%	
15	38%	
17	51%	
Modelo red neuronal		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	10%	6%
3	23%	19%
5	37%	32%
7	45%	41%
9	57%	49%
11	61%	52%
13	69%	56%
15	73%	61%
17	74%	62%

Modelo MARS		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	8%	6%
3	26%	21%
5	40%	33%
7	51%	42%
9	60%	51%
11	66%	56%
13	72%	62%
15	76%	67%
17	79%	69%

Tabla 18 Resumen de los resultados de los modelos con variable de salida nº total de defectos

En la siguiente gráfica se muestran los resultados de los tres modelos con variable de salida *Total_Defectos* para realizar una comparación directa de los resultados.

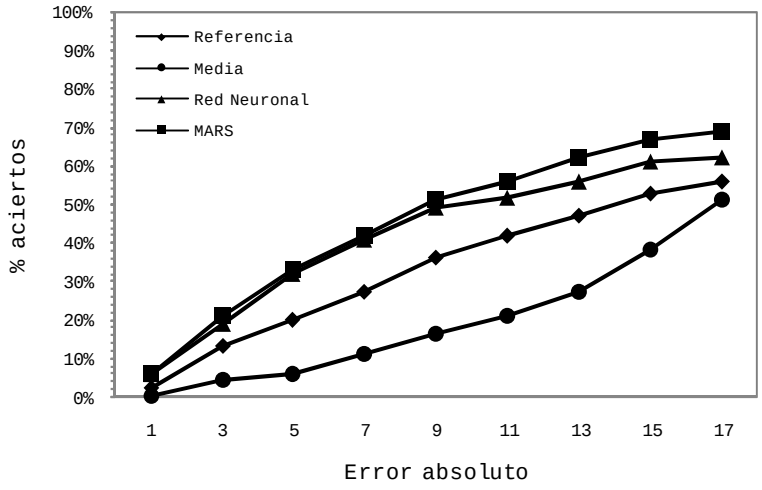


Fig. 33 Comparativa de resultados de los modelos con variable de salida *Total_Defectos*

6.2.- MODELO CON VARIABLE DE SALIDA DEFECTOS_LEVES

Modelo de Referencia	
Error absoluto	%aciertos
1	7%
3	24%
5	42%
7	47%
9	60%
11	62%
13	71%
15	76%
17	80%

Modelo Media		
Error absoluto	%aciertos	
1	3%	
3	11%	
5	21%	
7	35%	
9	42%	
11	50%	
13	56%	
15	65%	
17	73%	
Modelo red neuronal		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	10%	7%
3	35%	30%
5	52%	48%
7	69%	63%
9	75%	69%
11	81%	70%
13	88%	74%
15	90%	77%
17	92%	79%
Modelo MARS		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	19%	16%
3	48%	38%
5	66%	54%
7	77%	63%
9	84%	70%
11	89%	73%
13	92%	78%
15	94%	81%
17	*95%*	83%

Tabla 19 Resumen de los resultados de los modelos con variable de salida *Defectos_Leves*

En la siguiente gráfica se muestran los resultados de los tres modelos con variable de salida *Minor_Defects* para realizar una comparación directa de los resultados.

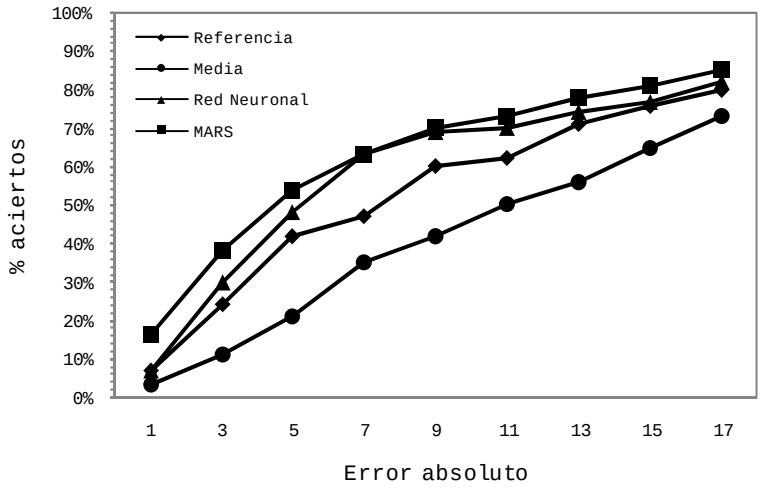


Fig. 34 Comparativa de resultados de los modelos con variable de salida *Defectos_Leves*

6.3.- MODELO CON VARIABLE DE SALIDA DEFECTOS_MEDIOS

Modelo de Referencia		
Error absoluto	%aciertos	
1	11%	
3	29%	
5	40%	
7	44%	
9	53%	
11	58%	
13	64%	
15	67%	
17	76%	
Modelo Media		
Error absoluto	%aciertos	
1	2%	
3	6%	
5	10%	
7	24%	
9	49%	
11	54%	
13	60%	
15	63%	
17	72%	
Modelo red neuronal		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	16%	14%
3	41%	38%
5	59%	53%
7	72%	60%
9	80%	65%
11	84%	70%
13	87%	72%
15	91%	75%
17	92%	76%
Modelo MARS		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	18%	17%
3	47%	43%
5	64%	55%
7	75%	63%
9	82%	68%
11	87%	74%
13	91%	77%
15	93%	80%
17	*94%*	82%

Tabla 20 Resumen de los resultados de los modelos con variable de salida *Defectos_Medios*

En la siguiente gráfica se muestran los resultados de los tres modelos con variable de salida *Defectos_Medios* para realizar una comparación directa de los resultados.

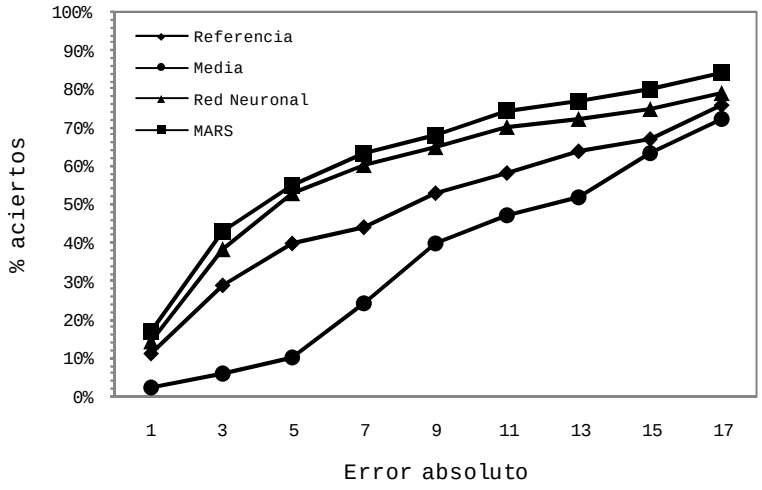


Fig. 35 Comparativa de resultados de los modelos con variable de salida *Defectos_Medios*

6.4.- MODELO CON VARIABLE DE SALIDA DEFECTOS_GRAVES

Modelo de Referencia	
Error absoluto	%aciertos
1	84%
2	91%
3	96%
4	96%
5	98%
6	98%
7	98%
8	98%
9	98%
Modelo Media	
Error absoluto	%aciertos
1	82%
2	85%
3	91%
4	91%
5	93%
6	93%
7	95%
8	95%
9	96%

Modelo red neuronal		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	85%	85%
2	90%	90%
3	90%	96%
4	92%	96%
5	95%	97%
6	100%	97%
7	100%	97%
8	100%	98%
9	100%	98%

Modelo MARS		
Error absoluto	%aciertos entrenamiento	%aciertos prueba
1	93%	86%
2	99%	93%
3	99%	96%
4	100%	96%
5	100%	97%
6	100%	98%
7	100%	98%
8	100%	98%
9	*100%*	98%

Tabla 21 Resumen de los resultados de los modelos con variable de salida *Defectos_Graves*

En la siguiente gráfica se muestran los resultados de los tres modelos con variable de salida *Defectos_Graves* para realizar una comparación directa de los resultados.

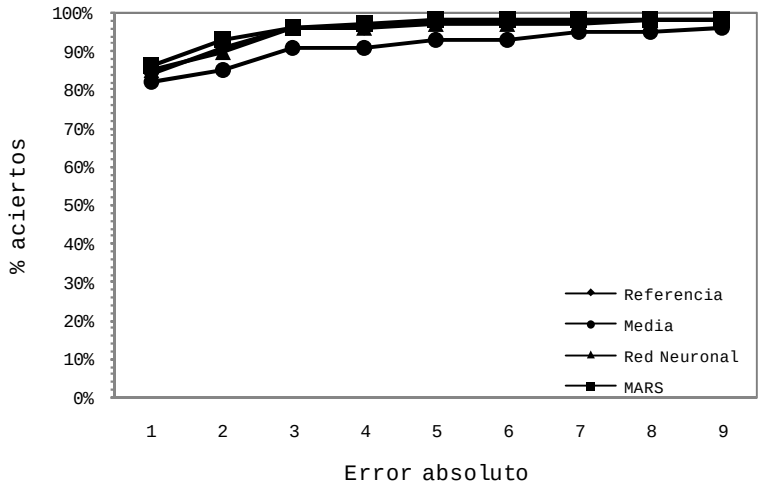


Fig. 36 Comparativa de resultados de los modelos con variable de salida *Defectos_Graves*

En las tablas y gráficos anteriores se puede observar que los modelos MARS tienen un comportamiento mejor que los basados en redes neuronales, con un porcentaje de aciertos superior. Se desprende al mismo tiempo que ambos tipos de modelos presentan porcentajes de acierto mucho mejores que el modelo de regresión utilizado como referencia, a excepción de los modelos referentes a los defectos graves en los que los tres modelos tienen un porcentaje de acierto muy alto debido a los valores que toman los datos y los márgenes de error absoluto permitido.

7. CONCLUSIONES

El resultado más significativo es que a partir de las variables aquí seleccionadas cuyas relaciones son analizadas, es posible concluir la existencia de un nivel de riesgos en el

proyecto, es decir, se identifican los aspectos a controlar en la gestión del riesgo, $R=f(x_i)$, independientemente del tipo de función f que posteriormente se aplique.

No obstante se completa la aproximación al riesgo determinando posibles funciones que aporten los mejores resultados, desde las regresiones lineales multivariantes hasta casos más complejos que han demostrado su mejor comportamiento.

Los modelos aquí desarrollados son generalistas en el sentido de que no se aplican a datos homogéneos y, en consecuencia, se suponen las soluciones generalizables. Los datos proceden de empresas distintas y muy diversas, de campos distintos y de países distintos. Toda esa diversidad impide realizar un modelo aún mejor. La aplicación de estos modelos a una empresa de forma directa seguramente no aportará los mismos resultados, sino peores, aunque siempre mejores que la aproximación general de la experiencia. Las condiciones del entorno, las especificidades de la empresa, los tipos de trabajo desarrollados, etc. son factores que deben ser personalizados.

Como los modelos están basados en datos, para su implantación dentro de una determinada organización se debe crear una base de datos local con información propia de la organización o, en su defecto, ir añadiendo a la ya existente los nuevos proyectos para ajustar la información y los modelos al comportamiento de la organización.

Los modelos desarrollados se basan en información procedente de históricos de cierres de proyectos y para generar una estimación del número de defectos leves, medios y graves requieren información del principio del proyecto, es decir, se dispone de información limitada. Pero a medida que evoluciona el proyecto se dispondría de información más ajustada que puede hacer que los resultados de los modelos sean más correctos. La información sobre las variables más influyentes en los defectos encontrados como el esfuerzo normalizado o el esfuerzo en la fase de construcción no se conoce de forma precisa en las fases iniciales de un proyecto, pero sí se puede estimar con más precisión a medida que el proyecto avanza, lo que permitirá estimar las salidas de los modelos (defectos leves, medios y graves) de forma más precisa. Otras variables también influyentes en los defectos encontrados como el tipo de lenguaje, una aproximación del número de usuarios concurrentes que usarán el sistema o una aproximación del tamaño medio del equipo sí pueden ser conocidas al principio del proyecto y usadas en los modelos desarrollados.

En todo caso resulta perfectamente aplicable para realizar una estimación del riesgo a partir de las características principales del proyecto y, sobre todo, para realizar análisis de sensibilidad de las distintas variables sobre el resultado final.

Capítulo 7: Conclusiones y líneas de futuro

1. CONCLUSIONES Y LÍNEAS DE FUTURO

Del desarrollo de esta tesis y el análisis de los resultados se pueden obtener las siguientes conclusiones generales:

1. El trabajo realizado ha permitido construir un modelo capaz de predecir el número de defectos leves, medios y graves a partir de variables que miden el tamaño del equipo, esfuerzos en diferentes fases del desarrollo, tipo de lenguaje, etc. Como consecuencia de ello se ha obtenido un mayor conocimiento del comportamiento de las variables seleccionadas para estimar los defectos de los proyectos.
2. Los defectos pueden considerarse como indicador del riesgo, si bien no ha sido posible en este caso, a partir de los datos, distinguir entre orígenes ni frecuencias de los riesgos, sino solamente de su gravedad.
3. El comportamiento de los defectos leves y medianos es muy diferente de los defectos graves, siendo estos mucho más identificables, debido en gran medida a que en su origen intervienen una serie de causas y no sólo una.
4. Las técnicas basadas en datos son capaces de estimar el riesgo superando a la experiencia y evitando la necesidad de una formulación tradicional, que no existiría dada la naturaleza del objetivo buscado.
5. El conjunto de la metodología desarrollada, producto de la combinación de diversas técnicas supervisadas y no supervisadas, proporciona un rendimiento óptimo que supera en más de un 30% los modelos tradicionales como regresión multivariante, utilizado en este trabajo como modelo de referencia.
6. El preprocesado es la parte fundamental de la modelización. La escasez de datos, los posibles datos erróneos y, sobre todo, la existencia de huecos haría imposible la creación de modelos realmente aplicables sin una buena gestión del preproceso.
7. En algunos casos, especialmente en los defectos leves, no es posible mejorar el nivel de acierto. Esto se debe a la existencia de fenómenos aleatorios no representables ni previsibles que introducen un elevado nivel de ruido en los datos.
8. La bondad de los resultados incide de nuevo en la necesidad, insistente en dirección de proyectos, de utilizar indicadores de desarrollo que puedan ser utilizados en las fases iniciales y recopilados en las finales.

Por otra parte, del desarrollo del trabajo aquí expuesto han surgido nuevas líneas de investigación sobre las que continuar el trabajo. Entre ellas se podrían destacar:

1. El trabajo se apoya en defectos como medida del riesgo, es decir, en las consecuencias de este. Sin embargo no están caracterizados los niveles de defecto, de forma que no es posible saber qué se corresponde con cada uno de los niveles. Sería importante caracterizar los defectos de forma que se delimite su nivel de gravedad y, si es posible realizar una clasificación en función del riesgo de origen, con el fin de evitar diferencias entre proyectos y poder realizar estudios de riesgos específicos.
2. Desde un punto de vista metodológico sería importante introducir este tipo de herramienta dentro de las metodologías de dirección y gestión de proyectos de sistemas de información, creando procesos específicos para esta labor de estimación inicial.
3. Los modelos basados en datos se basan precisamente en el tratamiento de estos. La recopilación de los datos es la limitación principal con el que la aplicación de estas técnicas se va a encontrar. Actualmente no existe ningún estándar de clasificación de datos para estas tareas. Sería conveniente el desarrollo de ontologías o jerarquías específicas que permitan identificar el campo y permitan compartir la información.
4. Una de las fases más importantes del proyecto es el llamado análisis post-mortem o lecciones aprendidas. Desgraciadamente y con especial hincapié en el mundo TI, no se dedican recursos a esta labor lo que, unido a la falta de una ontología del dominio específica, dificulta en gran medida la extracción de información. Sería muy deseable la investigación en formas de extracción automática de conclusiones a partir de la información recopilada durante el cierre del proyecto. Las técnicas de datamining tienen de nuevo mucho que decir ante un problema de este tipo.

Bibliografía y Referencias

[Addison, 2003] Addison, T. *E-commerce project development risks: evidence from a Delphi survey*. International Journal of Information Management 23, pp. 25–40. 2003.

[Agarwal et al 2006] Agarwal, N., Rathod, U. *Defining 'success' for software projects: An exploratory revelation*. International Journal of Project Management. Elsevier. 2006.

[Albrecht, 1979] Albrecht, A.J. *Measuring Application Development Productivity*. Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium. Octubre 1979.

[Barki et al, 1993] Barki, H., Rivard, S., Talbot, J. *Toward an assessment of software development risk*. Journal of Management Information Systems 10, pp. 203–225. 1993.

[Beecham et al, 2003] Beecham S., Hall T., Rainer A. *Software Process Improvement Problems in Twelve Software Companies: An Empirical Analysis*. Kluwer Academic Publishers, Hingham, MA, USA. 2003.

[Bellman, 1966] Bellman, R. *Adaptive Control Processes*. Princeton University Press, 1966

[Benhai et al, 2007] Yu, B., Zhang, J., Wei, S., Cong, G., Zhang, D., Chen, T. *The Research on Information System Project Performance Evaluation Based on Fuzzy Neural Network*. International Conference on Wireless Communications, Networking and Mobile Computing. pp. 6164-6168. Septiembre 2007.

[Boehm, 1981] Boehm, B. *Software Engineering Economics*. Prentice Hall PTR Prentice-Hall Inc. 1981.

[Boehm, 1991] Boehm, B. *Software risk management: principles and practices*. IEEE Software **8**, pp. 32–41. 1991.

[Boehm, 1995] Boehm, B., Clark, B., Horowitz, E. et al. *Cost models for future life cycle processes:COCOMO 2.0*. Annals Software Engineering 1. Pp. 1-24. 1995.

[Boehm, 1997] Boehm, B., Abts, C., Clark B, Devnani-Chulani, S. *COCOMO II model definition manual*. The University of Southern California. 1997.

[Boehm, 2000a] Boehm, B. et al. *Software cost estimation with Cocomo II*. Englewood Cliffs, NJ:Prentice-Hall. 2000.

[Boehm, 2000b] Boehm, B. et al. *Future Trends, Implications in Software Cost Estimation Models*. CrossTalk. pp. 4-8. Abril 2000.

[Boehm, 2000c] Boehm, B., Abts, C. *Software Development Cost Estimation Approaches – A Survey*. University of Southern California, 2000. Disponible on-line en http://sunset.usc.edu/publications/TECHRPTS/2000/usccse2000-505/usccse2000-505.pdf

[Booch et al, 1995] Booch, G., Rumbaugh, J. *Unified method for object-oriented development.* Tech. Rep., Rational Software Corporation. 1995.

[Briand, 2001] Briand, L.C., Wust, J. *Modeling development effort in object-oriented systems using design properties*. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 27. pp. 963-986. Noviembre 2001.

[Carney et al, 2003] Carney, D., Morris, E., Patrick, R. *Identifying commercial off-the-shelf (COTS) product risks: the COTS usage risk evaluation*. Technical Report, CMU/SEI-2003-TR-023. 2003.

[Cater, 1984] Cater, J. P. *Successfully using peak learning rates of 10 (and greater) in back-propagation networks with the heuristic learning algorithm.* Institute of Electrical and Electronics Engineers First International Conference on Neural Networks. San Diego, II-645-652, 1984

[Chapman, 1999] Chapman, P., Clinton, J., Khabaza, T., Reinartz, T. , Rüdiger, W. *The CRISP-DM process model*, CRISP-DM discussion paper, 1999

[Conte, 1986] Conte, E., Dunsmore, H. E., Shen, V. Y. *Software Engineering Metrics and Models*. Benjamin Cummings. 1986.

[Costa et al, 2007] Costa, HR. Barros, MDO. Travassos, GH. *Evaluating software project portfolio risks*. Journal of Systems and Software, Volume 80, Issue 1 Pages 16-31. January 2007.

[Craven, 1979] Craven, P. Wahba, G. *Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of Genrealized Cross-Validation*. Numerische Mathematic, 31, 1979

[Cutter Consortium, 2000] Research Briefs, 7 Nov. 2000.

[DeBoot, 1978] DeBoor,C. *A practical guide to splines*. Springer-Verlag. Nueva York, 1978

[DeMarco, 1982] DeMarco, T. *Controlling Software Projects: Management, Measurement and Estimation*. Englewood Cliffs, New York. Prentice Hall. 1982.

[Dempster, 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. *Maximum likelihood from incomplete data via the EM algorithm*. Journal of the Royal Statistical Society, Series B, 39:1-38, 1977

[DePompe, 1996] DePompe,B *There's gold in databases*, CMP Publications, (Jan. 8), 1996.

[DeVeaux, 1993] De Veaux R.D., Psichogios D.C., and Ungar L.H. *A Comparison of Two Nonparametric Estimation Schemes: MARS and Neural Networks*, Computers Chemical Engineering, Vol.17, No.8, 1993

[Dixon, 1979] Dixon, J. K. *Pattern recognition with partly missing data.* Rev: IEEE Transactions on Systems, Man, and Cybernetics, 9:617-621, 1979

[DoD, 2003] US Department of Defense. *Risk management guide for DOD acquisition fifth edition*. Defense Acquisition University, Defense Systems Management College. 2003.

[Dolado, 2001] Dolado J.J. *On the problem of the software cost function*. INFORMATION AND SOFTWARE TECHNOLOGY 43 pp. 61-72. Enero 2001.

[Famili, 1997] Famili, A., Shen, W.-M., Weber, R., Simoudius, E. *Data Preprocessing and Intelligent Data Analysis*. Rev: Intelligent Data Analysis Vol. 1, Nº1,3-23,1997

[Fan et al, 2004] Fan, C-F., Yu, Y-C. *BBN-based software project risk management*. The Journal of Systems and Software 73, pp. 193–203. 2004.

[Fenton, 1997] Fenton, N., Pfleeger, S.L., *Software Metrics, A Rigurous & Practical Approch*. PWS Publishing Company. 1997.

[Fenton et al, 2004] Fenton, N., Neil, M., Marsh, W., Hearty, P., Marquez, D. *Predicting Software Defects in Varying Development Lifecycles using Bayesian Nets.* Information and Software Technology, Elsevier. 2007.

[Friedman, 1991] Friedman, J. H. *Multivariate adaptive regression splines*. The Annals of Statistics, Vol. 19, Nº 1, 1-141, 1991

[Gray, 1999] Gray, A.R., McDonell, S.G. *Software metrics data analysis - exploring the relative performance of some commonly used modeling techniques*. Empirical Software Engineering 4 pp. 297-316. 1999.

[Han et al, 2007] Han, W.M., Huang, S.J. *An empirical analysis of risk components and performance on software projects*.Journal of Systems and Software. Volume 80 Issue 1 pp. 42-50. Enero 2007.

[Hebb, 1949] Hebb Donald O. *The Organization of Behavior*. Wiley. New York, 1949

[Houston et al, 2001] Houston, D., Mackulak, G., Collofello, J. *Stochastic simulation of risk factor potential effects for software development risk management*. Journal of Systems and Software 59, pp. 247–257. 2001.

[IPMA, 2006] ICB: IPMA *Competence Baseline Version 3.0*. IPMA, 2006.

[ISO10006] ISO10006. International Standardization Organization. 2003.

[Jiang et al, 2000] Jiang, J., Klein, G., Means, T. *Project risk impact on software development team performance*. Project Management Journal 31, pp. 19–26. 2000.

[Jorgersen, 1995] Jorgersen, M. *Experience with Accuracy of Software Maintence Task Effort Prediction Models.* IEEE Transaction on Software Engineering, Vol. 21. pp. 647-681. 1995.

[Karolak, 1997] Karolak, D. *Software Engineering Risk Management*, IEEE Computer Society. 1997.

[Kerzner, 2003] Kerzner, Harold. *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. Van Nostrand Reinhold, 1989.

[Kitchenahm, 1995] Kitchenahm, B., Pfleeger, S.L., Fenton, N.E., *Towards a framework for software measurement validation*. IEEE Transactions on software engineering, vol. 21, nº 12. pp. 929-944. 1995.

[Kliem, 2001] Kliem, R. *Risk management for business process reengineering projects*. Information Systems Management 17, pp. 71–73. 2001.

[Kohonen, 1996] Kohonen T., Hynninen J., Kangas J., Laaksonen J. *SOM_PAK: The Self-Organizing Map Program Package, Technical Report A31*, Helsinki University of Technology. 1996.

[Kwan-sik et al, 2007] Na, N.A., Simpson, J.T., Li, X., Singh, T., Kim, K.Y. *Software development risk and project performance measurement: Evidence in Korea*. Journal of Systems and Software Volume 80, Issue 4, pp. 596-605. Abril 2007.

[Lindbert, 1999] Linberg, K.R. *Software developer perceptions about software project failure: a case study*. Journal on Systems Software 49, pp. 177–192. 1999.

[Longstaff et al., 2000] Longstaff, T. et al. *Are we forgetting the risks of information technology?*. IEEE Computer 33, pp. 43–51. 2000.

[Matus, 2006] Matus, R. *Una herramienta para la predicción de riesgos de software usando modelos en redes bayesianas*. Tesis Doctoral, Universidad de Talca (Chile). 2006.

[McDonell, 1999] McDonell, S.G., Gray, A.R., Calvert, J.M. *FULSOME: a fuzzy logic modeling tool for software metricians*. In Proceedings of the 1999 Annual Meeting of the North American Fuzzy Information Processing Society - NAFIPS. New York NY, USA, IEEE Computer Society Press, pp. 263-267. 1999.

[McFarlan, 1981] McFarlan, F. *Portfolio approach to information systems*. Harvard Business Review 59, pp 142–150. 1981.

[Peeters, 2000] Peeters D., Dewey G. *Reducing bias in software project estimates*. Crosstalk–J. Defense Software Eng. 2000.

[Piatetski, 1991] Piatetski-Shapiro G., Frawley W.J: *Knowledge discovery in databases*. Ed. AAAI/MIT Press, 1991.

[PMBok, 2004] PMBok. *Project Management Book of Knowledge*. Project Management Institute. 2004.

[Putnam, 1992] Putnam, L., Myers, W. *Measures for Excellence*. Yourdon Press. 1992.

[Rodríguez, 2003] Rodríguez Montequín, M. T.. *Modelado evolutivo mediante técnicas adaptativas aplicado al control de inclusiones en bandas laminadas en caliente*. Tesis doctoral, Universidad de Oviedo, 2003

[Saaty, 1980] Saaty, T.L. *The Analytic Hierarchy Process*. New York: McGraw-Hill. 1980.

[Sammon, 1969] Sammon, J.W. A Nonlinear Mapping for Data Structure Analysis. IEEE Transaction on computers. Volume C-18, Nº 5. 1969.

[Shepperd, 2001] Shepperd, M., Cartwright, M. *Predicting with sparse data* IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 27, pp. 987-998. Noviembre 2001.

[Silverman, 1985] Silverman, B.W. *Some aspects of the spline smoothing approach to non-parametric regression curve fitting*. Journal of Royal Statistical Society b. 47, 1-52 ,1985

[Sontag, 1992] Sontag E. D., *Feedforward Nets for Interpolation and classification*, Computer J, System Sciences 45, 20-48, 1992

[SPSS, 2001] SPSS. *Clementine 6.0:Users guide*, SPSS, 2001

[Sumner, 2000] Sumner, N. *Risk factors in enterprise-wide/ERP projects*. Journal of Information Technology 15, pp. 317–327. 2000.

[Tiwana et al, 2006] Tiwana, A., Keil, M. Functionality Risk in Information Systems Development: An Empirical Investigation. IEEE Transactions on Engineering Management. Volume 53, Issue 3, pp. 412 – 425. 2006.

[Tuysuz et al, 2006] Tuysuz, F., Kahraman, C. *Project risk evaluation using a fuzzy analytic hierarchy process: An application to information technology projects*. International Journal on Intelligent Systems, Volume 21 Issue 6 pp. 559-584. Enero 2006.

[Verner et al, 2007] Verner, J.M., Evanco, W.M., Cerpa, N. *State of the practice: An exploratory analysis of schedule estimation and software project success prediction*. Information and Software Technology Volume 49 Issue 2 pp. 181-193. Febrero 2007.

[Villanueva, 2005] Villanueva Balsera, J. *Estimación de costes y plazos en proyectos de sistemas de información*. Tesis Doctoral, Universidad de Oviedo, 2005.

[Wallace et al, 2004] Wallace, L., Keil, M. y Arun, R. *How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model*. Decision Sciences 35, pp. 289–321. 2004.

[Wallace et al, 2004b] Wallace, L., Keil, M., Rai, A. *Understanding software project risk: a cluster analysis*. Information and Management 42, pp. 115–125. 2004

[Wateridge, 1998] Wateridge, J. *How can IS/IT projects be measured for success*. International Journal on Project Management 16, pp. 59–63. 1998.

[Williams et al, 1999] Williams, R. et al. *SRE Method Description & SRE Team Members Notebook Version 2.0*. Software Engineering Institute, Carnegie Mellon University CMU/SEI-99-TR-029. 1999.

[Wilson, 1997] Wilson, D.R., Martinez T.R. *Improved heterogenous distance functions*. Journal of Artificiall Intelligence Research 6. 1997.

[Wolverton, 1974] Wolverton, R. *The Cost of Developing Large Scale Software*. IEEE Transactions on Computers, C-23, pp. 615-636. 1974

Enlaces de Internet

[AEIPRO] Asociación Española de Ingeniería de Proyectos http://www.aeipro.com (22/02/2007)

[CHAOS 1994] The CHAOS report http://www.standishgroup.com/sample_research/chaos_1994_1.php (03/11/2006).

[CHAOS 2000] http://www.procuno.com/users/taller/Presentaciones/PresentacionIngeniA.ppt (03/11/2006).

[CHAOS 2003] http://www.standishgroup.com/press/article.php?id=2 (03/11/2006).

[CHAOS 2004] http://www.standishgroup.com/sample_research/PDFpages/q3-spotlight.pdf (03/11/2006).

[CHAOS ART] http://www.standishgroup.com/sample_research/index.php (03/11/2006).

[CHAOS Q 2004] http://www.standishgroup.com/sample_research/PDFpages/q4_quality.pdf (03/11/2006).

[CRISP-DM] Cross Industry Standard Process for Data Mining http://www.crisp-dm.org/ (05/11/2006)

[ESPITI] European Software Process Improvement Iniatiative http://ica.cordis.lu/search/index.cfm?fuseaction=proj.simpledocument&PJ_RCN=2805181&CFID=8072775&CFTOKEN=25462554 (22/03/2006).

[Financial Express] http://www.financialexpress.com/fe_full_story.php?content_id=118935 (03/04/2006).

[GAO] Government Account Office http://www.gao.gov/new.items/d06529t.pdf (03/11/2006)

[IPMA] International Project Management Association	http://www.ipma.ch/Pages/IPMA.aspx (22/02/2007)
[ISBSG] International Software Benchmarking Standards Group	http://www.isbsg.org (28/09/2006)
[KDNUGGETS]	http://www.kdnuggets.com (15/12/2002)
[MAGERIT] Metodología de Análisis y GEstión de RIesgos de los sistemas de información	http://www.csi.map.es/csi/pg5m20.htm (12/04/2008)
[PMI] Project Management Institute	http://www.pmi.org/Pages/default.aspx (22/02/2007)
[SEI] Software Engineering Institute	http://www.sei.cmu.edu/ (15/01/2007)
[Standish Group]	http://www.standishgroup.com/chaos/introduction.pdf (03/11/2006)

Anexos

1. ESTADÍSTICOS BÁSICOS DE LAS VARIABLES

Estadísticos básicos de las variables del problema.

	Media	Mediana	Moda	Desviación	Mínimo	Máximo	P25	P75
Puntos de función	472.77	268.00	119.00	618.65	6.00	4,943.00	132.25	560.00
Puntos de función ajustados	480.07	275.50	74.00	632.91	6.00	5,684.00	126.00	581.00
Valor del factor de ajuste	1.00	1.00	1.00	0.10	0.65	1.30	0.97	1.06
Esfuerzo total	5,362.47	2,207.50	700.00	8,846.88	97.00	61,891.00	783.00	5,489.50
Esfuerzo normalizado	5,475.12	2,381.00	760.00	8,689.71	97.00	59,878.00	841.00	5,636.50
Valor del factor de normalización	1.05	1.00	1.00	0.19	0.38	3.70	1.00	1.08
Esfuerzo de planificación	562.86	196.09	0.00	1,338.44	0.00	17,668.00	62.50	580.21
Esfuerzo de especificación	816.15	272.59	0.00	1,651.35	0.00	13,230.00	82.00	779.30
Esfuerzo de diseño	658.24	132.20	0.00	1,806.82	-286.00	22,117.50	14.00	580.21
Esfuerzo de construcción	1,921.75	599.50	208.00	3,957.72	0.00	31,142.00	199.04	1,609.87
Esfuerzo de pruebas	831.46	299.25	250.00	1,540.01	0.00	15,015.00	119.75	840.75
Esfuerzo de implementación	570.39	167.90	0.00	1,119.08	-286.00	9,240.00	51.90	597.50
Duración	8.39	7.00	4.00	6.07	1.00	48.00	4.00	10.00
Tiempo de inactividad	1.22	0.36	0.00	1.47	0.00	12.00	0.00	2.50
Porcentaje de inactividad	0.27	0.13	0.00	0.45	0.00	2.50	0.00	0.36
Plataforma de desarrollo	2.53	3.00	3.00	0.79	1.00	4.00	2.00	3.00
Tipo de lenguaje	3.54	3.00	3.00	0.60	2.00	5.00	3.00	4.00
Herramienta CASE	0.16	0.00	1.00	0.89	-1.00	1.00	-1.00	1.00
Metodología	0.59	1.00	1.00	0.67	-1.00	1.00	0.00	1.00
Ubicaciones de implantación	91.96	7.00	1.00	634.11	1.00	10,000.00	1.00	7.00
Usuarios concurrentes	939.98	7.50	7.00	4,549.95	1.00	38,109.00	7.00	400.00
Máximo tamaño de equipo	9.25	5.00	3.00	12.57	1.00	99.00	3.00	10.00
Media tamaño de equipo	7.31	4.00	3.00	9.15	1.00	73.00	2.20	8.00
Defectos leves	9.49	2.00	0.00	21.83	0.00	150.00	0.00	6.00
Defectos medios	9.30	1.00	0.00	33.45	0.00	285.00	0.00	4.00
Defectos graves	0.31	0.00	0.00	1.15	0.00	14.00	0.00	0.00
Total de defectos	13.89	2.00	0.00	36.81	0.00	285.00	0.00	10.00

Tabla 22 Estadísticos básicos de las variables del problema

2. CORRELACIÓN DE LAS VARIABLES NUMÉRICAS

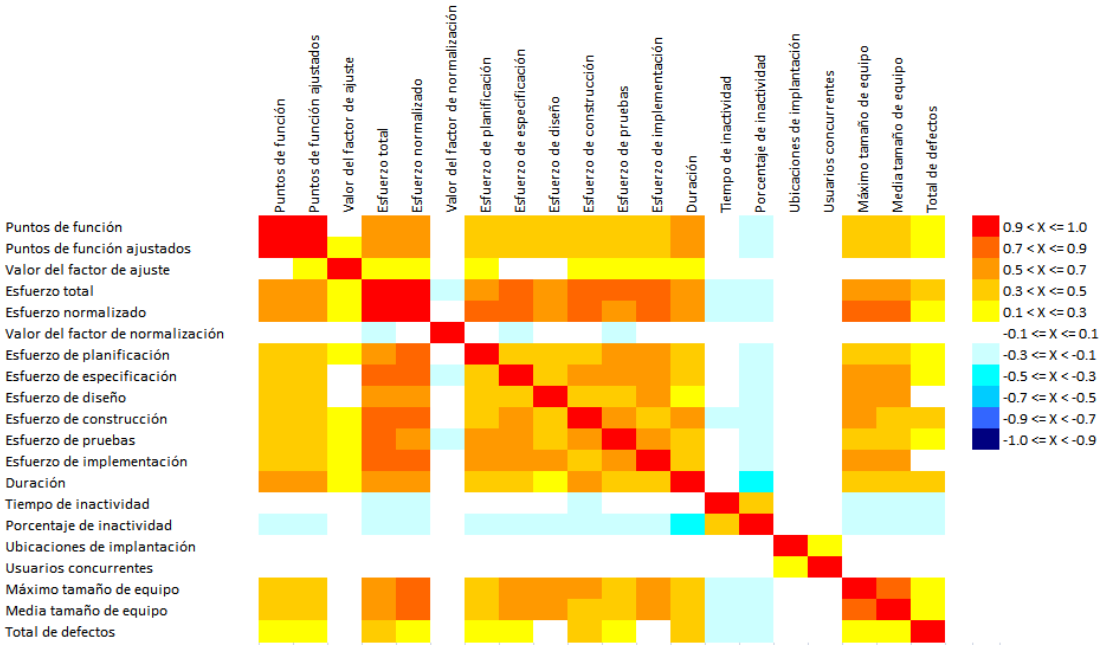


Fig. 37 Correlación de las variables numéricas

3. GLOSARIO DE TÉRMINOS

Agrupamiento – Clustering	Proceso de dividir un conjunto de datos en grupos mutuamente excluyentes de tal manera que cada miembro de un grupo esté lo "más cercano" posible a otro, y grupos diferentes estén lo "más lejos" posible uno del otro, donde la distancia está medida con respecto a todas las variables disponibles.
Algoritmo – Algorithm	Secuencia definida de operaciones
Análisis exploratorio de datos – Exploratory data analysis	Uso de técnicas estadísticas tanto gráficas como descriptivas para aprender acerca de la estructura de un conjunto de datos.
Análisis prospectivo de datos – Prospective data analysis	Análisis de datos que predice futuras tendencias, comportamientos o eventos basado en datos históricos.
Clasificación – Classification	Consiste en el etiquetado de los registros, de forma que se pueda establecer un modelo que permita clasificarlos.
Crisis del Software – Software crisis	Dificultad existente en desarrollar programas sin errores o defectos, fácilmente comprensibles, y que sean verificables
Data Mining	La extracción de información predecible escondida en grandes bases de datos.
Función de activación – Activation Function	Función utilizada por una neurona en una red neuronal para transformar datos de entrada desde cualquier dominio de valores en un conjunto finito de valores

Herramientas CASE	Aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.
Inteligencia Artificial – Artificial Intelligence	Es el campo de la ciencia de la computación dedicado a analizar y desarrollar sistemas que reproduzcan e imiten los procesos de pensamiento y razonamiento del hombre.
ISBSG	International Software Benchmarking Standars Group. Organización sin ánimo de lucro que ha creado, mantiene y explota tres repositorios con datos históricos relativos a tecnologías de la información (métricas de software) para ayudar a mejorar la gestión de tecnologías de la información de forma global.
ISO 9.000	Conjunto de normas orientadas a ordenar la gestión de la empresa y el aseguramiento de la calidad.
Limpieza de datos – Data cleaning	Proceso de asegurar que todos los valores en un conjunto de datos sean consistentes y correctamente registrados.
MARS – Multivariate Adaptive Regression Splines	Técnica de modelado basada en la construcción de modelos de regresión generados a partir de la situación de los nodos de forma adaptativa. Es capaz de seleccionar de forma automática las variables que intervienen en el modelo.
Método Heurístico – Heuristic Method	Resolución de problemas, probando diferentes métodos y comparando cual es el que ofrece la mejor solución.
Modelo analítico – Analytical Method	Una estructura y proceso para analizar un conjunto de datos.
Modelo Lineal – Linear Model	Un modelo analítico que asume relaciones lineales entre una variable seleccionada (dependiente) y sus predicadoss (variables independientes).

Modelo no lineal - Non Linear Model	Un modelo analítico que no asume una relación lineal en los coeficientes de las variables que son estudiadas.
Modelo predictivo – Predictive Model	Estructura y proceso para predecir valores de variables especificadas en un conjunto de datos.
Neurona Artificial - Artificial Neuron	Unidad básica de procesamiento de información que basa su funcionamiento tomando como modelo las células que conforman el cerebro y que procesan datos en forma de pulsos eléctricos y reacciones químicas en sus uniones con otras células similares.
Outlier	Un item de datos cuyo valor cae fuera de los límites que encierran a la mayoría del resto de los valores correspondientes de la muestra. Puede indicar datos anormales. Deberían ser examinados detenidamente; pueden dar importante información.
Red Neuronal – Neural Network	Herramienta de modelado empírico cuantitativo que se caracteriza por una red altamente interconectada de nodos que transmite valores numéricos entre ellos y calcula un resultado en base a la suma de entradas de otros nodos. Sin reglas convencionales, una red neuronal obtiene experiencia analizando automática y sistemáticamente una cantidad de datos, para determinar reglas de comportamiento. En base a estas reglas, puede realizar predicciones sobre nuevos casos. Ofrecen el potencial de identificar conexiones que otras técnicas no pueden, porque utiliza relaciones lineales y no–lineales entre los datos, puede trabajar con cualquier tipo de distribución (no solamente distribución normal) y maneja datos con redundancia y/o inconsistencia en la información
Redes SOM – Self Organised Maps	Red que estructura un mapa que preserva la topología de una representación multidimensional dentro de un vector bidimensional de neuronas, la red proyecta señales similares a posiciones similares de neuronas.

Regresión Lineal - Linear Regression	Técnica estadística utilizada para encontrar la mejor relación lineal que encaja entre una variable seleccionada (dependiente) y sus predicados (variables independientes).

4. PUBLICACIONES

La siguiente tabla muestra las publicaciones a las que ha dado lugar esta tesis.

Título	Revista o Diario	Fecha publicación
Revisión y Clasificación de las Técnicas de Análisis de Riesgos aplicadas a proyectos de ingeniería	Proceedings of the IX International Congress on Project Engineering	Junio 2005
Software project cost estimation using Multivariate Adapting Regression Splines	WSEAS Transactions on Information Science and Applications. Issue 12, Volume 2. ISSN 1790-0832. Indexada en los índices INSPEC, COMPENDEX y SCOPUS.	Diciembre 2005
Desarrollo de una Herramienta de Análisis de Riesgos en operaciones industriales	Proceedings of the X International Congress on Project Engineering	Septiembre 2006
Predicción y Clasificación de Riesgos en proyectos de sistemas de información	Proceedings of the XII International Congress on Project Engineering	Julio 2008

Tabla 23 Publicaciones a las que ha dado lugar la tesis