Analyzing the Effect of Variables in the Software Development Effort Estimation

H. Velarde, C. Santiesteban, A. García and J. Casillas

***Abstract*— Due to the variability of variables, the estimation of effort in software development is considered as a very difficult task. Maybe due to small databases, uncertain and very subjective information. For these reason, in this paper, we introduce a study of the effect of variables in the software development effort estimation process. The estimation employing lines of codes, function points and engineering task, were taken as reference. With the use of datamining and machine learning technics, were demonstrated that the effort estimation using engineering task is more accurate. And, by the other hand, was established the influence of each variable, in the software development effort estimation process.**

***Keywords*— Datamining, Effort estimation, Decision Trees, Regression Trees, Software metrics, Software projects, Variables selection.**

I. INTRODUCCIÓN

EN la actualidad, la estimación del esfuerzo de desarrollo de software es una tarea complicada debido a la gran cantidad de factores que influyen en la misma. Es necesario disponer de la información de proyectos concluidos, tal que permita establecer un modelo de relaciones entre las variables utilizadas en los proyectos y el esfuerzo necesario para su desarrollo.

Múltiples modelos de estimación del esfuerzo de desarrollo de software han sido propuestos hasta el momento. Los métodos basados en regresión son los más comunes. Mientras que le siguen los métodos basados en el criterio de expertos y en analogía [1]. Recientemente, los modelos basados en técnicas de aprendizaje automático han recibido mayor atención. Donde, las técnicas más empleadas son las redes neuronales, algoritmos genéticos, árboles de decisión y regresión, algoritmos de agrupamiento, basados en casos, multiclasificadores, entre otras técnicas de minería de datos [1]–[9].

En comparación con otros escenarios en los que han sido aplicadas exitosamente las técnicas de aprendizaje automático, la estimación del esfuerzo de desarrollo de software posee muchos otros retos como: bases de datos pequeñas y, desactualizadas, información imprecisa, incompleta y altamente subjetiva, datos cualitativos, entre otros. Así como que estas bases de datos se basan en información obtenida empíricamente y no como parte de un proceso sistemático. Es por ello que a pesar de los esfuerzos realizados, aún no se logran modelos capaces de estimar el esfuerzo de desarrollo de software con adecuada precisión [7], [10], [11].

En el presente artículo se propone realizar un estudio, a partir de la aplicación de técnicas de minería de datos y aprendizaje automático, que permita establecer cuál es la influencia de cada una de las variables utilizadas en el proceso de estimación del esfuerzo de desarrollo de software.

El artículo está estructurado como sigue: la sección de materiales y métodos donde se muestra cómo se realizó la selección de los datos empleados, el diseño de la experimentación y el método de evaluación de la influencia de las variables en la estimación del esfuerzo de desarrollo de software. La sección de análisis y discusión de los resultados, donde se mide la influencia de cada variable en la estimación y los algoritmos que mejor la describen. Finalmente, las conclusiones del trabajo resumen los resultados encontrados y su discusión.

II. MATERIALES Y MÉTODOS

La propuesta de la investigación se centra en el análisis de la influencia de las variables empleadas en la estimación del esfuerzo de desarrollo de software, a partir de un conjunto de factores conocidos en las fases iniciales del proyecto. Las variables propuestas recopilarán la información de los datos que son conocidos o estimables en las fases iniciales.

Todas las bases de datos empleadas en esta investigación, así como las pruebas estadísticas, pueden ser consultadas desde nuestro sitio web de materiales suplementarios (https://sites.google.com/site/seebetrepository/).

A. Bases de datos

Para realizar el análisis, fueron seleccionadas bases de datos públicas, extraídas del repositorio PROMISE (*PRedictOr Models In Software Engineering Software*) [12] y un repositorio de tareas de ingeniería obtenidas de aplicaciones de software de base de datos, implementadas por los centros de desarrollo de software de la Universidad de Ciencias Informáticas de las provincias de Las Villas y Ciego de Ávila en Cuba (Universidad de Ciencias Informáticas de Cuba, URL: http//:www.uci.cu). Estas bases de datos fueron separadas en tres grupos, las basadas en: conteo de líneas de código, puntos de función y órdenes de trabajo.

- **Basadas en conteo de líneas de código (LOC)**

A raíz de la aparición del modelo COCOMO [13], se ha estandarizado el empleo de una serie de variables para la estimación del esfuerzo de desarrollo de software. Éstas, describen características de los proyectos en cuanto a: tipo de software, tipo de hardware, personal que participará en el proyecto y datos propios de la organización. No obstante,

H. Velarde, Facultad de Ciencias e Ingenierías. Físicas y Formales, Universidad Católica de Santa María, Arequipa, Perú, hvelardeb@gmail.com.

C. Santiesteban, Centro de Bioplantas, Universidad de Ciego de Ávila, Cuba, cosme@bioplantas.cu.

A. García, División Territorial Villa Clara, DESOFT, Cuba, anamaria.garcia@vcl.desoft.cu.

J. Casillas, Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, España, casillas@decsai.ugr.es.

debido a que los problemas relacionados con el hardware ya no se consideran una limitación, las variables relacionadas con estos rasgos fueron excluidas en esta investigación. Por tal motivo serán empleadas solo 11 variables:

1. **Rely**: Garantía de funcionamiento requerida al software.
2. **Data**: Tamaño de la base de datos en relación con el tamaño del programa.
3. **Cplx**: Representa la complejidad del producto.
4. **Acap**: Calificación de los analistas.
5. **Aexp**: Experiencia del personal en aplicaciones similares.
6. **Pcap**: Calificación de los programadores.
7. **Vexp**: Experiencia del personal en la máquina virtual.
8. **Lexp**: Experiencia en el lenguaje de programación a usar.
9. **Modp**: Uso de prácticas modernas de programación.
10. **Tool**: Uso de herramientas de desarrollo de software.
11. **Sced**: Limitaciones en el cumplimiento de la planificación.

En la Tabla I, se muestra el conjunto de datos que recoge un amplio número de proyectos y variables, los cuales no solo representan un elevado número de casos útiles para la estimación del esfuerzo de desarrollo de software, sino que permiten la comparación con puntos de función y órdenes de trabajo.

TABLA I
CARACTERIZACIÓN DEL SET DATOS EMPLEADOS PARA EL ANÁLISIS BASADO EN LÍNEAS DE CÓDIGO

Base de datos	Dimensiones		Esfuerzo			
	Proyectos	Variables	mínimo	máximo	Media	dev-est
Cocomo81	63	17	5,9	11400	683,3	1821,6
Nasa_1	60	16	8,4	3240	406,4	657
Nasa_2	93	24	8,4	8211	624,4	1135,9
Cocomo_Sdr	12	25	1	22	5,7	6,8
Total a usar	**228**	**11**	**1**	**11400**	**550,8**	**1252,9**

- **Basadas en puntos de función (PF)**

El cálculo de los puntos de función conlleva un nivel elevado de definición y maduración del proyecto. Es una métrica que permite traducir en un número, el tamaño de la funcionalidad que brinda un producto de software. Se estima desde el punto de vista del usuario, a través de una suma ponderada de las características del producto. Resulta independiente de la tecnología utilizada para la construcción y explotación del software. Las variables empleadas para este estudio fueron cuatro:

1. **Input**: Procesos en los que se introducen datos
2. **Output**: Procesos en los que se envía datos al exterior de la aplicación
3. **Enquiry**: Procesos consistentes en la combinación de una entrada y una salida
4. **File**: Grupos de datos relacionados entre sí internos al sistema

En la Tabla II, se muestra el conjunto de datos empleados para el análisis de los atributos relacionados con puntos de función. Ésta, reúne una amplia cantidad de proyectos, útiles para la estimación del esfuerzo y la comparación con las estimaciones basadas en líneas de código y órdenes de trabajo.

TABLA II
CARACTERIZACIÓN DEL SET DATOS EMPLEADOS EL ANÁLISIS BASADO EN PUNTOS DE FUNCIÓN

Base de datos	Dimensiones		Esfuerzo			
	Proyectos	Variables	mínimo	máximo	media	dev-est
Albrecht	24	8	0,5	105,2	21,88	28,42
China	499	19	26	54620	3921,05	6480,86
Total a usar	**523**	**4**	**0,5**	**54620**	**3742,12**	**6382,58**

- **Basadas en tareas de ingeniería (TI)**

El cálculo del esfuerzo basado en tareas de ingeniería de proyectos de software también requiere de un elevado nivel de definición del proyecto. Sin embargo, está orientado mayormente a su empleo en prácticas de programación ágiles. Donde, es imprescindible tener una medida de la productividad de los integrantes del equipo. Es una métrica que permite computar el esfuerzo en un producto de software independientemente de la tecnología y la secuencia empleada durante la construcción del mismo. Tomando elementos del modelo COCOMO [13], se han estandarizado algunas variables como parte de la estimación del esfuerzo de desarrollo de software y se agregaron otras a partir de la experiencia de los equipos de desarrollo. Éstas, describen características de los proyectos en cuanto a: tipo de software y personal integrante del proyecto:

1. **complejidad**: Cuan simple o compleja es la tarea a desarrollar.
2. **conReuso**: Porcentaje de código existente que puede ser utilizado en el desarrollo de la tarea.
3. **conocimientoTarea**: Experiencia con la que cuenta el programador en el desarrollo de aplicaciones.
4. **conocimientoLenguaje**: Experiencia con la que cuenta el programador en el uso del lenguaje empleado.
5. **fiabilidad**: Grado de tolerancia de errores en la tarea.
6. **herramientasSoftware**: Empleo de herramientas que faciliten y reduzcan el tiempo de desarrollo.
7. **tiempoR**: Tiempo estimado, número de horas empleadas.

Esta base de datos contiene 700 registros de órdenes de trabajo. Donde, el esfuerzo está calculado en horas/hombre (h/h) y la media es 35,96 h/h con una desviación estándar de 12,24 h/h. Con un valor mínimo de 10 h/h y máximo de 85 h/h.

B. Diseño experimental

La estimación del esfuerzo de un proyecto de software se considera compleja, debido a la gran variabilidad de los datos que intervienen en el proceso. Es por ello que, para el desarrollo de la investigación, se siguió el diseño experimental que se muestra en la Fig. 1. El cual cuenta de los pasos siguientes:

1) Análisis de la capacidad discriminante de las diferentes variables sobre la estimación del esfuerzo. Mediante la comparación los perfiles de las variables en cada uno de los proyectos.
2) En caso de que los perfiles no sean discriminantes, determinar la influencia de cada variable sobre estimación del esfuerzo.
   a) Emplear herramientas de teoría matemática de la

información:
- Ganancia de la información.
- Razón de ganancia.

b) Emplear herramientas de aprendizaje automático para la selección de variables.

3) Realizar un análisis sistemático de la influencia de las variables en la estimación del esfuerzo:
   a) Algoritmos basados en reglas.
   b) Algoritmos basados en árboles.
4) Determinar la mejor estrategia de estimación del esfuerzo de desarrollo de software empleando herramientas de aprendizaje automático:
   a) Algoritmos de aprendizaje más comunes:
      - Redes neuronales.
      - Máquinas de vectores soporte.
      - Combinación de clasificadores.
   b) Algoritmos interpretables:
      - Basados en reglas.
      - Basados en árboles.
5) Aplicar pruebas de significación estadística que permitan evaluar dichas estrategias.

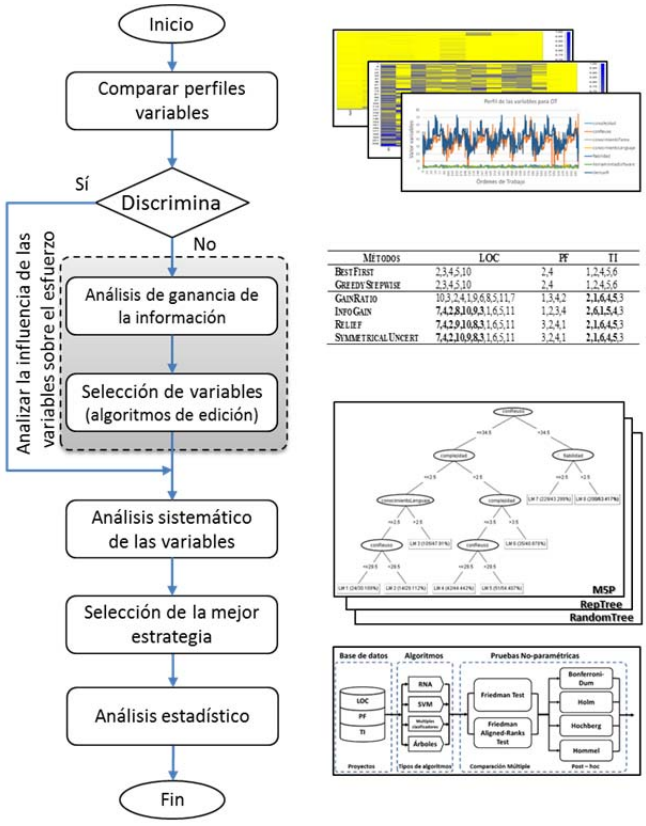

Figura 1. Diseño experimental. El diagrama de flujo describe la secuencia de pasos a seguir para el desarrollo de la investigación. A la derecha, una representación simbólica del proceso.

C. Evaluación de resultados

Con el objetivo de evaluar el comportamiento de las variables seleccionadas, se hace necesario contar con los estadígrafos que permitan medir adecuadamente el desempeño en cada uno de estos resultados. Para ello se realizó el procedimiento de validación cruzada y fueron analizadas las magnitudes del error [14].

Dentro de las medidas continuas se emplearon: el error relativo medio (1), y el coeficiente de correlación (***CC***), el cual toma valores en el rango de 1 para los resultados perfectamente correlacionados, 0 para la no correlación, y –1 para los perfectamente correlacionados negativamente [15].

$$MRE = \sum_i \left| \frac{p_i - r_i}{r_i} \right| \quad (1)$$

donde ***p*** son los valores predichos y ***r*** son los valores reales.

Con la intención de determinar las diferencias en el comportamiento entre los clasificadores, se realizan varias pruebas no paramétricas (Fig. 2). Primeramente, se aplica la prueba para múltiples muestras relacionadas de Friedman, como procedimiento de comparación múltiple. Ésta, prueba la hipótesis nula de que todas las bases de datos permiten la estimación del esfuerzo de forma similar. Luego, si la hipótesis es rechazada, se aplican las pruebas de *post-hoc* Bonferroni–Dunn, Holm, Hochberg y Hommel [16].

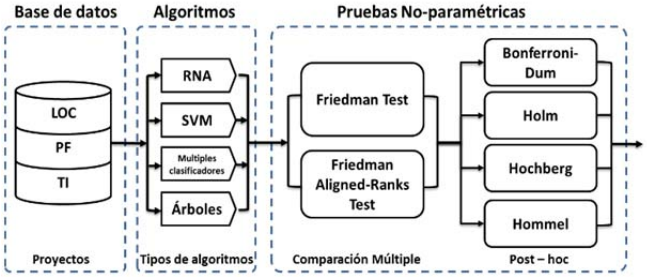

Figura 2. Evaluación de los resultados.

III. ANÁLISIS Y DISCUSIÓN

Para determinar la influencia de las variables sobre la estimación del esfuerzo, se compara el nivel de expresión de cada una por separado y los niveles de entropía. Luego se mide el nivel de correlación que pueda existir entre ellas, empleando el análisis de correlación.

A. Análisis de perfiles y correlación

Para analizar el perfil de las variables, en cada base de datos, se emplearon diferentes gráficos que describiesen sus características. Donde, para cada representación, se estableció que el eje de las abscisas está formado por las variables que se analizan y el eje de las ordenadas por el valor que toma cada una de las variables para cada proyecto (series). En todos los casos, los proyectos fueron ordenados en función del esfuerzo, que es la variable objetivo. Este proceso se realizó para las tres bases de datos analizadas.

LOC: El perfil de esta base de datos se puede apreciar mejor mediante el empleo de un diagrama de temperatura. El cual consiste en una matriz de colores que representa los valores que toman las variables para las series que forman cada uno de los 228 proyectos (Fig. 3). Para ello, la matriz se creó normalizando todos los valores de las variables y se ordenó ascendentemente, en función del esfuerzo (variable objetivo). Mientras más parecidos sean los patrones de colores

de las columnas, mayor será la correlación entre éstas. Patrones similares, pero en sentido inverso corresponden a una correlación negativa y patrones diferentes representan baja o ninguna correlación.

Como se puede apreciar en la Fig. 3, es muy difícil establecer una clara correlación entre el comportamiento de las series (valores de las variables en cada proyecto) y el valor del esfuerzo de desarrollo de dichos proyectos. Sin embargo, se logra identificar una correlación más directa entre el número de líneas de código generadas durante la elaboración del proyecto ***Loc*** y el esfuerzo necesario para su desarrollo.

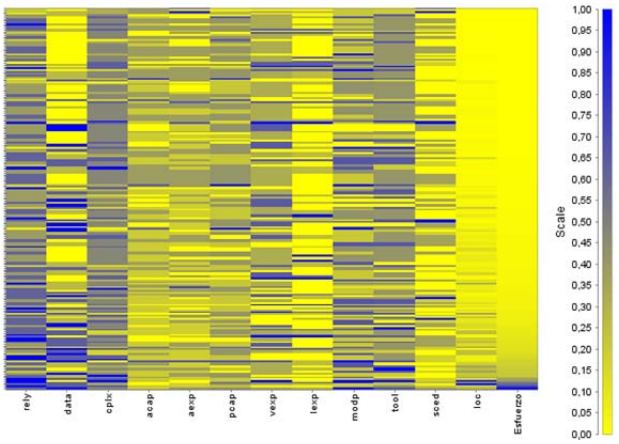

Figura 3. Diagrama de temperatura que representa el perfil de las variables para LOC. Muestra la baja correlación que existe entre las variables y el esfuerzo, el cual parece estar solo correlacionado con Loc.

Lo que evidencia que, en este caso, el análisis del perfil de las variables no es determinante para la estimación del esfuerzo de desarrollo de software. Pues, éste parece depender exclusivamente de ***Loc***.

PF: Igualmente, cada serie constituye el perfil de las variables para los 523 proyectos (Fig. 4). Esta figura muestra que el comportamiento de las variables para cada proyecto es similar. Presentándose las diferencias únicamente en la magnitud del valor de las mismas, por lo que tampoco existe correlación entre ellas. Solamente la variable RawFPcount presenta una alta correlación con el esfuerzo, pero es inversa. De lo que se puede concluir que el análisis del perfil tampoco es determinante para la estimación del esfuerzo de desarrollo de software.

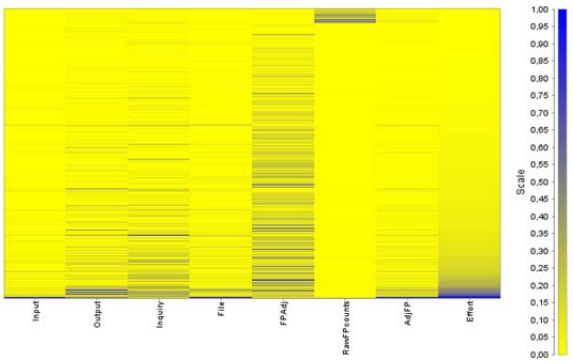

Figura 4. Diagrama de temperatura que representa el perfil de las variables para FP. Muestra la baja correlación de las variables con respecto al esfuerzo, excepto RawFPcounts que presenta una alta correlación negativa.

TI: Una alternativa para la representación de los perfiles, es emplear un diagrama de líneas. En este caso, el perfil de las variables se construye con el valor de 700 tareas de ingeniería (Fig. 5). Aquí se hace mucho más evidente la correlación que existe entre todas ellas. Además, todas siguen el mismo comportamiento, solo variando la magnitud de los valores que toman. Por lo que tampoco el empleo del perfil resulta adecuado para la estimación del esfuerzo de desarrollo de software.

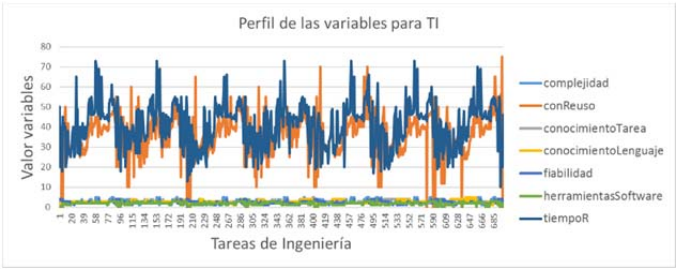

Figura 5. Perfil de las variables para TI. Muestra la alta correlación entre las variables.

B. Influencia de las variables sobre el esfuerzo

Teniendo en cuenta que se está realizando un análisis exploratorio de datos, es necesario hallar las causas de la variabilidad del conjunto de datos y ordenar las variables por importancia para poder construir modelos predictivos. Es por ello que se realiza el análisis en función de la ganancia de la información y la razón de ganancia. Adicionalmente, fueron empleadas las técnicas de aprendizaje automático para la selección de variable: BestFirst y GreedyStepwise (Tabla III).

TABLA III
RANKING DE ATRIBUTOS PARA ESTIMACIÓN DEL ESFUERZO DE DESARROLLO DE SOFTWARE. SE RESALTAN LAS COINCIDENCIAS EN EL ORDEN DE LOS ATRIBUTOS PARA CADA BASE DE DATOS

No	Métodos	LOC	PF	TI
1	BestFirst	2,3,4,5,10	2,4	1,2,4,5,6
	GreedyStepwise	2,3,4,5,10	2,4	1,2,4,5,6
2	GainRatio	10,3,2,4,1,9,6,8,5,11,7	1,3,4,2	**2,1,6,4,5,3**
	InfoGain	**7,4,2,8,10,9,3,1,6,5,11**	1,2,3,4	**2,6,**1,**5,**4,**3**
	Relief	**7,4,2,9,10,8,3,1,6,5,11**	**3,2,4,1**	**2,1,6,4,5,3**
	SymmetricalUncert	**7,4,2,10,9,8,3,1,6,5,11**	**3,2,4,1**	**2,1,6,4,5,3**

La Tabla III, muestra el orden de importancia de las variables en el proceso de estimación del esfuerzo, establecido como resultado del empleo de los métodos de evaluación de atributos. Como puede apreciarse, en para cada base de datos, aparecen coincidencias en los rankings establecidos por estos algoritmos.

En el caso de la base de datos basada en LOC, los algoritmos de tipo goloso sugieren que las variables más influyentes son: (2) *Data*, (3) *Cplx*, (4) *Acap*, (5) *Aexp* y (10) *Tool*; mientras que los algoritmos basados en información de ganancia sugieren que el orden de importancia de estos atributos es: (7) *Vexp*, (4) *Acap*, (2) *Data*, (9) *Modp*, (10) *Tool*, (8) *Lexp*, (3) *Cplx*, (1) *Rely*, (6) *Pcap*, (5) *Aexp*, (11) *Turn*.

Por otra parte, para la base de datos basada en PF, los algoritmos golosos identifican como principales atributos a (2)

Output y (4) *File*. Y, los algoritmos basados en la teoría de la información sugieren que el orden de importancia de las variables es: (3) *Enquiry*, (2) *Output*, (4) *File* y (1) *Input*.

Para la base de datos basada en TI, los algoritmos golosos sugieren: (1) *complejidad*, (2) *conReuso*, (4) *conocimiento-Lenguaje*, (5) *fiabilidad*, (6) *herramientasSoftware*. Por su parte, el orden sugerido por los algoritmos basados en teoría de la información, establecen el siguiente ranking: (2) *conReuso*, (1) *complejidad*, (6) *herramientasSoftware*, (4) *conocimiento-Lenguaje*, (5) *fiabilidad* y (3) *conocimientoTarea*.

En sentido general, los algoritmos de evaluación de variables basados en estrategias golosas, seleccionan menos variables, pero no difieren apreciablemente de los rankings establecidos por los algoritmos basados en la teoría de la información. Por tal motivo, se asumió el orden sugerido por los algoritmos basado en ganancia.

C. Análisis sistemático del orden de las variables

Poder determinar cuál es el orden de importancia de las variables a partir de la relación que existe entre éstas y el esfuerzo real, es una tarea importante. Esto se debe a que permitiría a los directores de proyecto tomar decisiones más adecuadamente. Sin embargo, el análisis de la influencia de las variables sobre el esfuerzo, se realizó tomando en cuenta la influencia de las mismas sobre todo el conjunto de datos. Pero, esta no es la forma en que trabajan los algoritmos de aprendizaje automático. Debido a que cada vez que se realiza una partición de los datos basada en la información que aporta una de las variables, la influencia de las mismas sobre los subconjuntos resultantes varía considerablemente.

Esto justifica que se realice un estudio sistemático, a partir del análisis del orden de las variables establecido por varios algoritmos basados en reglas. Para ello, se tomaron en cuenta las reglas generadas por los árboles que mejor correlacionaron en la estimación del esfuerzo y que menor porcentaje de error relativo obtuvieron.

Para este estudio, fueron seleccionados los algoritmos basados en árboles de regresión: M5Rules [17], M5P [18], RandomTree [19] y REPTree [19]. Cada uno de los algoritmos son entrenados y evaluados en las bases de datos LOC, PF y TI y se evalúa el orden de las variables que éstos establecen. A partir de aquí, se puede sugerir cuál es el orden de importancia de dichas variables en la estimación del esfuerzo.

LOC: El resultado de la aplicación de estos algoritmos, muestran varios órdenes de importancia de las variables. Donde resaltan:

- *Rely*, (5) *Aexp*, (10) *Tool*, (7) *Vexp*, (8) *Lexp*, (4) *Acap*, (2) *Data* y (9) *Modp*.
- (9) *Modp,* (1) *Rely*, (8) *Lexp*, (2) *Data* y (10) *Tool*.
- (9) *Modp,* (10) *Tool,* (1) *Rely*, (2) *Data* y (8) *Lexp*.

De este análisis se puede inferir la importancia que juegan variables como ***Modp***, ***Rely***, ***Lexp***, ***Data*** y ***Tool***, con independencia del orden que ocupen en el proceso de la estimación. Por otra parte, estos algoritmos, en su mayoría, discriminan las variables: ***Cplx***, ***Pcap*** y ***Turn***.

PF: Cuando se analizan los resultados obtenidos de este estudio, se aprecia una ligera diferencia con los resultados propuestos por el análisis basado en teoría de la información. Donde, los árboles sugieren que el orden de importancia de las variables es: (2) *Output*, (3) *Enquiry*, (4) *File* y (1) *Input*. Intercambiándose las variables (2) *Output* y (3) *Enquiry.* Estos resultados son, relativamente, similares, sobre todo si se toma en cuenta que ambas variables logran el mayor protagonismo en la estimación del esfuerzo de desarrollo de software para cada uno de los algoritmos empleados en el estudio.

TI: En análisis de este estudio se realizó tomando como referencia, las reglas extraídas de los árboles más representativos: A partir de los cuales se puede identificar que el orden de importancia de las variables es: (2) *conReuso*, (1) *complejidad*, (3) *conocimientoLenguaje* y (5) *fiabilidad*. Si se toma en cuenta el orden establecido por los algoritmos de evaluación de variables se obtiene: (2) *conReuso*, (1) *complejidad*, (6) *herramientasSofware*, (3) *conocimientoLenguaje* y (5) *fiabilidad*. Criterios que son prácticamente coincidentes en ambos casos. Es por ello que, para la estimación del esfuerzo de desarrollo de software basado en órdenes de trabajo, se puede establecer que las variables relacionadas serían: ***conReuso***, ***complejidad***, ***conocimientoLenguaje*** y ***fiabilidad***.

D. Selección de la mejor estrategia

El último paso de este estudio es determinar cuál es la mejor estrategia para la estimación del esfuerzo de desarrollo de software. Para ello, es necesario poder determinar cuál es la variante (LOC, PF o TI) y cuál es el algoritmo más adecuado para realizar la estimación. Por tal motivo, fueron empleados algunos de los principales algoritmos de aprendizaje automático [20], los cuales fueron evaluados en cada una de estas bases de datos en cuanto a su correlación y error relativo (Tabla IV).

TABLA IV
RESULTADO DE APLICAR LOS ALGORITMOS MÁS POPULARES EN LAS BASES DE DATOS PARA ESTIMACIÓN DEL ESFUERZO DE DESARROLLO DE SOFTWARE. DONDE R REPRESENTA EL RANKING DE LOS ALGORITMOS/CONJUNTOS DE VARIABLES, CC ES EL COEFICIENTE DE CORRELACIÓN Y MRE EL ERROR RELATIVO

ALGORITMOS	LOC			PF			TI			R
	R	CC	MRE	R	CC	MRE	R	CC	MRE	
MLP	4	0,34	100,49	7	0,50	100,72	8	0,73	70,98	**7**
SVM	8	0,21	103,53	1	0,68	66,64	9	0,61	74,33	**6**
BAGGING	5	0,31	87,59	4	0,61	74,40	2	0,87	43,19	**2**
RANDOMCOMMITEE	2	0,39	70,73	6	0,50	86,77	3	0,86	43,20	**3**
M5RULES	7	0,31	104,06	3	0,62	73,00	7	0,83	50,32	**5**
M5P	6	0,31	94,52	2	0,64	71,81	4	0,84	48,62	**4**
RANDOMFOREST	1	0,42	75,49	5	0,61	75,85	1	0,89	40,95	**1**
RANDOMTREE	3	0,36	71,71	9	0,40	105,37	6	0,83	47,63	**6**
REPTREE	9	0,08	102,30	8	0,46	80,61	5	0,84	49,04	**9**
INFLUENCIA DE LAS VARIABLES	**3**	**0,30**	**90,05**	**2**	**0,56**	**81,69**	**1**	**0,81**	**52,03**	////

La Tabla IV muestra que, los algoritmos logran mejor correlación y un error más bajo en la estimación del esfuerzo cuando se emplean tareas de ingeniería. Como promedio, los algoritmos logran estimar el esfuerzo de desarrollo de software con un 81% de correlación y un error relativo del 52% para esta base de datos. Mientras que si se emplea puntos de función solo se alcanza una correlación del 56% y el error

relativo aumento al 81%, y al emplear el número de líneas de código, la efectividad de la estimación cae a solo un 30% de correlación y el error relativo llega a alcanzar el 90%.

Con el objetivo de estudiar el comportamiento del error del modelo propuesto, se construyó un diagrama de dispersión (Fig. 6). Donde se representan el valor del esfuerzo estimado en relación con el valor del esfuerzo real para cada orden de trabajo.

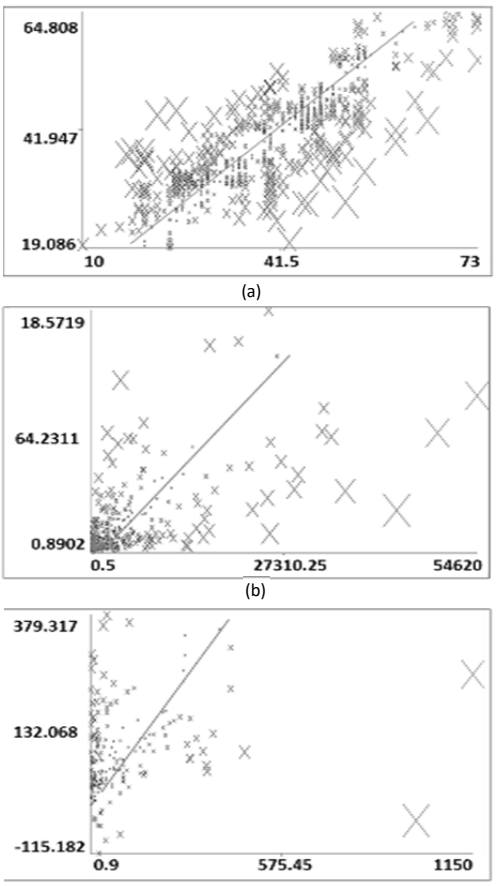

Figura 6. Diagrama de dispersión que muestra la calidad de la estimación del esfuerzo para: (a) Tareas de Ingeniería; (b) Puntos de Función; (c) Líneas de Código. Por el eje de las abscisas se encuentra el esfuerzo real y por el eje de las ordenadas, el esfuerzo estimado.

La Fig. 6a muestra que la estimación en tareas de ingeniería se concentra alrededor de la pendiente ideal. Este es un comportamiento deseable, no obstante, se observa el nivel de dispersión que genera el error en la estimación del esfuerzo de desarrollo de software. Sin embargo, algo totalmente distinto sucede cuando la estimación está basada en puntos de función o en el conteo de líneas de código.

Donde, en el caso de la estimación basada en puntos de función (Fig. 6b), se estiman valores muy bajos cuando se trata de proyectos grandes. Sin embargo, el error cometido en la estimación basada en conteo de líneas de código (Fig. 6c) muestra una gran dispersión. Cuando se trata de proyectos pequeños, los algoritmos tienden a sobrestimar el valor del esfuerzo.

Sin embargo, con el objetivo de determinar si existen diferencias estadísticamente significativas en el efecto de los diferentes conjuntos de datos en el desempeño de los algoritmos, fueron empleados múltiples pruebas estadísticas no paramétricas. Mediante la prueba de Friedman se comprueba la hipótesis nula de que las bases de datos permiten estimar de forma similar el esfuerzo. Hipótesis que fue rechazada, con un *p-value* de 0,0003. Por lo que fueron aplicadas las pruebas de post-hoc: Bonferroni-Dunn, Holm, Hochberg y Hommel, los cuales demostraron que existen diferencias estadísticamente significativas entre los algoritmos con un p-value ajustado de inferior a 0,1 para un nivel de significación igualmente de 0,1. Las cuales demostraron que estimar el esfuerzo en tareas de ingeniería resulta más adecuado que la estimación empleando puntos de función o basado en líneas de código (Fig. 7).

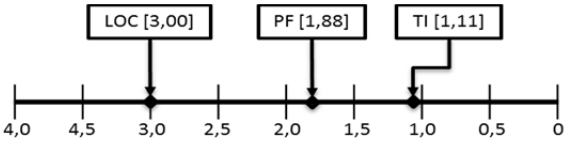

Figura 7. Representación visual de ordenamiento promedio de Friedman con un $\alpha = 0.10$.

IV. CONCLUSIONES

En el presente artículo se muestra un estudio del efecto de las variables en la estimación del esfuerzo de desarrollo de software. Para ello se tomaron como referencia las bases de datos de estimación a partir del conteo de líneas de código y de puntos de función. Además, se introdujo una base de datos de estimación del esfuerzo de desarrollo de software, a partir de tareas de ingeniería. Utilizando técnicas de minería de datos y aprendizaje automático, se logró demostrar que resulta más adecuada la estimación basada en tareas de ingeniería y se logró establecer la influencia de cada una de las variables en la estimación.

AGRADECIMIENTOS

Esta investigación se inserta en el programa doctoral Iberoamericano en Soft Computing, desarrollado por la Universidad de Las Villas, Cuba y la Universidad de Granada, España. Bajo el auspicio de la Junta de Andalucía y la Asociación Universitaria Iberoamericana de Postgrado (AUIP). Especial agradecimiento a todos los que contribuyeron con sus invaluables comentarios y consideraciones.

REFERENCIAS

[1] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Inf. Softw. Technol.*, vol. 54, no. 1, pp. 41–59, 2012.

[2] M. Algabri, F. Saeed, H. Mathkour, and N. Tagoug, "Optimization of soft cost estimation using genetic algorithm for NASA software projects," in *2015 5th National Symposium on Information Technology: Towards New Smart World (NSITNSW)*, pp. 1–4, 2015.

[3] M. G. A. C, G. Raura, J. A. R. R, and E. R. F. C, "Modelo Neuronal de Estimación para el Esfuerzo de Desarrollo en Proyectos de Software (MONEPS)," *Rev. Latinoam. Ing. Softw.*, vol. 3, no. 3, pp. 148–154, 2015.

[4] S. Waghmode and K. Kolhe, "A Novel Way of Cost Estimation in Software Project Development Based on Clustering Techniques," *Int. J.*

Innov. Res. Comput. Commun. Eng., vol. 2, no. 4, pp. 3892–3899, 2014.
[5] J. G. Borade and V. R. Khalkar, "Software Project Effort and Cost Estimation Techniques," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 8, pp. 730–739, 2013.
[6] L. L. Minku and X. Yao, "Ensembles and locality: Insight on improving software effort estimation," *Inf. Softw. Technol.*, vol. 55, no. 8, pp. 1512–1528, 2013.
[7] K. Dejaeger, W. Verbeke, D. Martens, and B. Baesens, "Data Mining Techniques for Software Effort Estimation: A Comparative Study," *IEEE Trans. Softw. Eng.*, vol. 38, no. 2, pp. 375–397, 2012.
[8] H. Najadat, I. Alsmadi, and Y. Shboul, "Predicting Software Projects Cost Estimation Based on Mining Historical Data," *ISRN Softw. Eng.*, vol. 2012, pp. 1–8, 2012.
[9] S. Kad and V. Chopra, "Software Development Effort Estimation Using Soft Computing," *Int. J. Mach. Learn. Comput.*, vol. 2, no. 5, pp. 548–551, 2012.
[10] M. Saroha and S. Sahu, "Tools & methods for software effort estimation using use case points model – A review," in *International Conference on Computing, Communication & Automation*, pp. 874–879, 2015.
[11] Danh Nguyen-Cong and De Tran-Cao, "A review of effort estimation studies in agile, iterative and incremental software development," in *The 2013 RIVF International Conference on Computing & Communication Technologies - Research, Innovation, and Vision for Future (RIVF)*, pp. 27–30, 2013.
[12] J. . Sayyad Shirabad and T. J. Menzies, "The PROMISE Repository of Software Engineering Databases," Ottawa, Canada, 2005.
[13] B. Boehm, B. Clark, S. Devnani-Chulani, E. Horowitz, R. Madachy, D. Reifer, R. Selby, B. Steece, Centre for Software Engineering USC, and S. Engineering, "COCOMO II Model Definition Manual," *Univ. South. Calif.*, vol. 4, no. 1, pp. 6–6, 2000.
[14] I. H. Witten and E. Frank, *Data Mining: Practical Machie Learning Tools and Techniques*, 2th ed. Morgan Kaufmann, 2005.
[15] E. Frank, M. Hall, L. Trigg, G. Holmes, and I. H. Witten, "Data mining in bioinformatics using Weka," *Bioinformatics*, vol. 20, no. 15, pp. 2479–2481, Oct. 2004.
[16] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci. (Ny).*, vol. 180, no. 10, pp. 2044–2064, 2010.
[17] N. Ramakrishnan, "Chapter 1. C4.5," in *The Top Ten Algorithms in Data Mining*, Taylor & Francis Group, LLC, pp. 1–19, 2009.
[18] D. Steinberg, "Chapter 10. CART: Classification and Regression Trees," in *The Top Ten Algorithms in Data Mining*, Taylor & Francis Group, LLC, pp. 179–201, 2009.
[19] Y. Zhao and Y. Zhang, "Comparison of decision tree methods for finding active objects," Advances of Space Research, pp. 10, 2007.
[20] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, p. 10, 2009.

Héctor Velarde es doctorando en TIC con Diploma de Estudios Avanzados en Soft Computing de la Universidad de Granada, España; Magíster en Ingeniería de Software de la Universidad Católica de Santa María de Arequipa y titulado como Ingeniero de Sistemas en la Universidad Católica LACh., Perú. Actualmente se desempeña como Docente Principal e Investigador en la Universidad Católica de Santa María de Arequipa, Perú. Sus intereses en investigación se centran en minería de datos e ingeniería de software.

Cosme Santiesteban es Doctor en Ciencias de la Computación en la Universidad de Granada, España en el 2015, e hizo un Máster en Inteligencia Artificial y otro en Informática Aplicada en el 2010 y 2004 respectivamente. Es Investigador a tiempo completo en el Centro de Bioplantas y Profesor Auxiliar a tiempo parcial en la Universidad de Ciego de Ávila, Cuba. Se interesa en el reconocimiento de patrones, inteligencia artificial, procesamiento digital de imágenes, técnicas de soft computing y computación paralela/distribuida.

Ana García es Doctora en Ciencias Técnicas del Instituto Politécnico José A. Echevarría de la Habana, Cuba y Licenciada en Cibernética Matemática de la Universidad Central de las Villas. Actualmente se desempeña como docente Titular en la Universidad Central de las Villas y es Sub Directora Comercial de DESOFT en Villa Clara, Cuba. Sus intereses en investigación se centran en sistemas de información, desarrollo de aplicaciones e ingeniería de software.

Jorge Casillas Recibió el Máster y el Doctorado en Ciencias de la Computación en 1998 y 2001 respectivamente, en la Universidad de Granada, España. Es Profesor Asociado en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granda, donde es miembro del grupo de investigación en Soft Computing y Sistemas de Información Inteligentes. Se interesa en *big data*, *data stream*, sistemas difusos, algoritmos evolutivos, robótica, sistemas inteligentes, entre otros.