

# Transformando la Representación de los Datos para Mejorar el Clasificador Bayesiano Simple

Jose Carlos Cortizo<sup>1,2</sup>, Ignacio Giraldez<sup>2</sup>, and Mari Cruz Gaya<sup>2</sup>

<sup>1</sup> Artificial Intelligence & Network Solutions S.L.

jccp@ainetsolutions.com; <http://www.ainetsolutions.com/jccp>

<sup>2</sup> Universidad Europea de Madrid

Villaviciosa de Odon, 28670, Madrid, Spain

{josecarlos.cortizo, ignacio.giraldez, mcruz}@uem.es

**Abstract.** El clasificador bayesiano simple se basa en la asunción de independencia entre los valores de los atributos dado el valor de la clase. Así pues, su efectividad puede decrecer en presencia de atributos interdependientes. En este artículo se presenta DGW (Dependency Guided Wrapper), un wrapper que utiliza la información acerca de las dependencias entre atributos para transformar la representación de los datos para mejorar la precisión del clasificador bayesiano simple. Este artículo presenta una serie de experimentos donde se compara las representaciones de datos obtenidas por el DGW contra las representaciones de datos obtenidas por 12 acercamientos previos, como son la construcción inductiva de productos cartesianos de atributos, y wrappers que realizan búsquedas de subconjuntos óptimos de atributos. Los resultados de los experimentos muestran que DGW genera representaciones nuevas de los datos que ayudan a mejorar significativamente la precisión del clasificador bayesiano simple más frecuentemente que cualquier otro acercamiento previo. Además, DGW es mucho más rápido que cualquier otro sistema en el proceso de transformación de la representación de los datos.

**Palabras Clave:** Wrapper, asunción de Independencia, Naive Bayes, Selección de Atributos

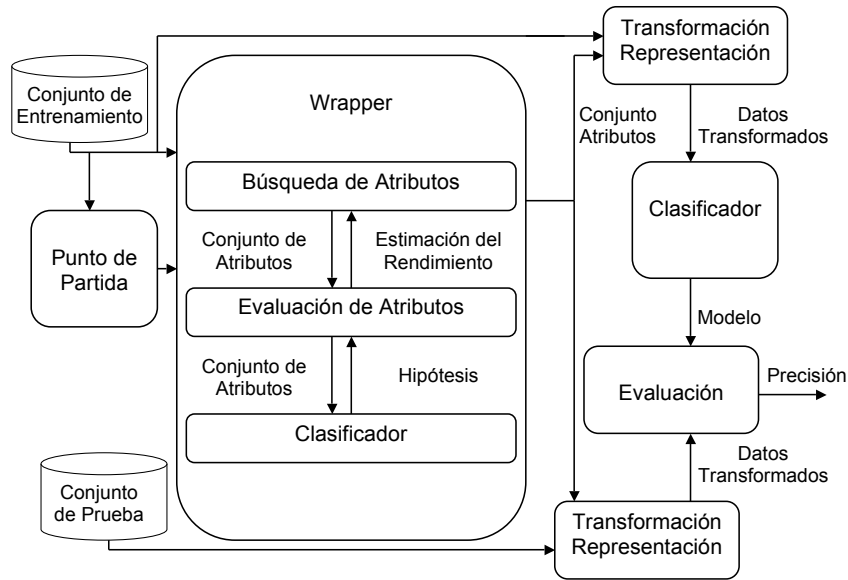
## 1 Introducción

Existen múltiples enfoques del problema de la clasificación automática supervisada, desde la inducción de árboles de decisión, enfoques basados en proximidad, etc. pero el acercamiento estadístico parece ser el más intuitivo y simple. De hecho, existen acercamientos a este problema por parte de la estadística anteriores a los enfoques del Aprendizaje Automático.

Dentro de los clasificadores estadísticos, el clasificador bayesiano simple se ha ganado una posición de privilegio [1] debido a su simplicidad, su resistencia al ruido, su eficiencia en cuanto a tiempo de ejecución y espacio [2], su comprensibilidad [3], sus resultados en cuanto a rendimiento y velocidad en el área de la recuperación de información y la categorización automática de textos [4] y, además, es bien conocido que cuándo la asunción de independencia se

cumple, ningún otro clasificador puede mejorar al clasificador bayesiano simple en términos de probabilidad de instancias correctamente clasificadas [5].

El clasificador bayesiano asume que los valores de los atributos son condicionalmente independientes dado el valor de la clase (ver [6] and [7] para un análisis de mayor profundidad a este respecto). Esta asunción no es realista, ya que en muchas situaciones reales aparecen atributos que, en alguna medida, son dependientes entre si.



**Fig. 1.** Estructura general de un Wrapper.

Existen muchos acercamientos que tratan de aliviar esta asunción, dentro de los cuáles podemos distinguir tres líneas principales: ① acercamientos que tratan de relajar esta asunción modificando el clasificador [8], [9], ② extracción de características para lograr una representación de los datos dónde los atributos sean independientes (o pseudo-independientes) [2], [10], [7] y ③ enfoques que infravaloran la asunción de independencia [11], [12], [13].

Este artículo presenta un acercamiento que se sitúa, cláramente, en la línea de ② y se organiza de la siguiente manera. En la siguiente sección se tratan cuestiones generales relacionadas con los métodos wrapper y su aplicación al problema de la asunción de independencia. En la sección 3 se presenta un wrap-

per que guía su búsqueda en el espacio de atributos en las dependencias existentes entre los atributos (DGW, Dependence Guided Wrapper). La sección 4 explica los experimentos realizados para probar el DGW y compararlo con otros acercamientos previos. La última sección contiene los resultados y algunas conclusiones.

## 2 Wrappers para Selección de Atributos

La precisión predictiva de los clasificadores puede degradarse cuándo se enfrentan con atributos irrelevantes. La explicación a este fenómeno puede encontrarse en la “Maldición de la Dimensionalidad” [14] que se refiere al crecimiento exponencial del número de instancias necesarias para describir los datos en función de la dimensionalidad (número de atributos). La selección de atributos trata de conseguir un subconjunto de los atributos originales de un conjunto de datos, de tal forma que si un algoritmo de aprendizaje se ejecute sobre dicho subconjunto de los datos logre la mayor precisión posible. Existen dos corrientes generales dentro de la selección de atributos: el **modelo basado en filtros** y el **modelo basado en wrappers**.

Mientras que el **modelo basado en filtros** [15] se basa en la idea de Relevancia [16] y selecciona los atributos sin tener en cuenta el rendimiento del clasificador que se utilizará posteriormente, el **modelo basado en wrappers** [17] conduce una búsqueda en el espacio de posibles atributos utilizando el rendimiento del clasificador como parte de la función de evaluación de cada subconjunto de atributos. La Figura 1 muestra la estructura general de un wrapper, dónde se le ofrece al wrapper el conjunto de entrenamiento y un punto de partida (todos los atributos, ninguno o una selección aleatoria) para que el wrapper realice una búsqueda acorde con el algoritmo de búsqueda seleccionado, dónde el algoritmo de aprendizaje se utiliza como una caja negra para medir la adecuación de cada subconjunto de atributos.

En [18] se estudia un wrapper (FSS, Forward Sequential Selection) que realiza una búsqueda avariciosa utilizando el clasificador bayesiano simple como algoritmo de aprendizaje tratando de lidiar con el problema de los atributos altamente correlacionados introduciendo en el clasificador final únicamente algunos de ellos. [19] compara FSS y una adaptación del trabajo de Kittler [20] que llama BSE (Backward Sequential Elimination), muy similar al FSS pero realizando una búsqueda hacía atrás, partiendo de todos los atributos iniciales y eliminando uno de ellos en cada iteración. En [10] se propone el producto cartesiano de atributos como una operación para crear nuevos atributos compuestos que sustituyan los pares de atributos dependientes. A partir de esta operación, Pazzani propone variaciones (FSSJ y BSEJ) al FSS y BSE, dónde en cada iteración del algoritmo se estudia tanto el eliminar/agregar un atributo como realizar el producto cartesiano de dos de ellos.

Los resultados experimentales muestran que la construcción inductiva de atributos ayuda a que el wrapper consiga mejores precisiones. FSS, BSE, FSSJ, y BSEJ consiguen representaciones de los datos donde el clasificador bayesiano

simple logra mejores precisiones. Pero debido a la estructura general de un wrapper (utiliza la precisión del clasificador como métrica para evaluar cada subconjunto de atributos), parece lógico pensar que otros wrappers también pueden mejorar el rendimiento de este clasificador.

### 3 Wrapper Guiado por Correlaciones

Este artículo presenta un wrapper que realiza una búsqueda, en el espacio de atributos, guiado por la información de las dependientes existentes entre los mismos. Como resultado, el DGW (Dependency Guided Wrapper) obtiene un subconjunto de los atributos utilizados para representar los datos con el que se transforma la representación original de los mismos.

Ya que el clasificador bayesiano simple se basa en la asunción de atributos independientes. DGW trata de encontrar una representación de los datos libre de dependencias para satisfacer esta restricción. La Figura 1 muestra la estructura general de un Wrapper y, a partir de la misma, se procede a presentar el DGW, describiendo cómo implementa cada una de las partes del wrapper.

- ▷ **Punto de Partida:** DGW es un wrapper backward, es decir, parte del conjunto inicial de atributos y, en cada paso, estudia qué atributos sobran en la representación de los datos.
- ▷ **Clasificador:** El clasificador utilizado para estimar el rendimiento de cada subconjunto de atributos es el clasificador bayesiano simple.
- ▷ **Selección de Atributos:** DGW realiza dos fases para la selección de los atributos.

- La primera etapa es el wrapping basado en dependencias. Sea  $S = \{A_1, A_2, \dots, A_n\}$  el conjunto original de atributos y  $V = \{v_1, v_2, \dots, v_m\}$  los posibles valores de la clase.  $e_j = \{\mathbf{x}, v\}$  es un ejemplo de entrenamiento donde  $\mathbf{x} = (a_1, a_2, \dots, a_n)$  es un punto que pertenece al espacio de entrada ( $X$ ) y  $v \in V$  es un punto del espacio de salida ( $V$ ).

DGW intenta evitar las dependencias existentes en el conjunto original de atributos. Para ello, DGW comienza estudiando las dependencias lineales existentes entre cada par de atributos para, posteriormente, calcular el Coeficiente de Determinación ( $R^2$ ) de cada una de estas dependencias lineales como medida de estimación de lo dependientes que son cada par de atributos. Cada dependencia,  $d_i$ , queda definida como los dos atributos relacionados y el valor de  $R^2$  para la dependencia lineal entre ambos atributos,  $d_i = (A_i, A_j, I(A_i, A_j))$ . En este punto, DGW construye una lista  $L_{d_i}$ , conteniendo cada posible dependencia  $d_i$ , que se encuentra inversamente ordenada en función del valor de  $R^2$  de cada dependencia entre atributos. Finalmente,  $L_{d_i}$  se utiliza para obtener los atributos finales al utilizarse como guía para eliminar (o no) los atributos más dependientes hasta que no se consiga una mejora en la clasificación final.

En cada paso, el algoritmo considera un número dado de dependencias ( $N$ ), cogiendo las  $N$  primeras dependencias de  $L_{d_i}$ <sup>3</sup>. En cada paso se estudia cómo afecta el eliminar cada uno de los atributos de las dependencias a la clasificación final. Teniendo en cuenta que en cada paso se toman  $N$  dependencias y que cada dependencia esta formada por dos atributos, en cada paso se evaluarán  $2 * N$  modificaciones del espacio de atributos. Por cada uno de estos atributos, DGW calcula el impacto que produce su eliminación en la precisión final. Al final de cada iteración se eliminará aquel atributo que cause un mayor impacto positivo, es decir, aquel atributo que, cuándo se elimina del conjunto de atributos, haga que el clasificador bayesiano simple obtenga una mejor precisión. Cuándo en un determinado paso no se consigue mejorar la precisión clasificadora eliminando cualquiera de los atributos, DGW detiene y concluye esta fase.

- La segunda fase es el **Ranking mediante Evaluador de Atributos**. DGW recibe como parámetro un determinado evaluador de atributos (por ejemplo Information Gain, Chi Cuadrado, etc.) y, para cada uno de los atributos que no hayan sido eliminados en la fase previa, calcula el valor de esta metrica. Con esto, DGW genera una lista  $L_{EA}$  conteniendo todos los atributos supervivientes, ordenándolos en función de los valores ofrecidos por el evaluador de atributos. Entonces, se analiza el impacto de eliminar o no cada uno de los atributos de esta lista. Si eliminar el atributo contribuye a mejorar (o mantener) la precisión, entonces se elimina, en caso contrario se conserva el atributo.

Una vez finalizado todo el proceso, DGW obtiene  $F = \{A_i, A_j, \dots A_z | A_k \in S\}$ , que es un subconjunto de los atributos originales que contiene aquellos con los que se obtiene una mejor precisión clasificadora. A partir de  $F$ , DGW transforma la representación de los datos para que todas las instancias queden representadas en función de estos atributos.

## 4 Experimentos

En esta sección se presentan los experimentos realizados para evaluar el wrapper propuesto en este artículo. Estos experimentos comparan DGW con otros métodos propuestos en el estado del arte y wrappers que realizan una búsqueda entre los posibles subconjuntos de atributos para mejorar el rendimiento del clasificador bayesiano simple. Para este propósito, se han seleccionado 23 conjuntos de datos de la UCI [21], tratando que mostraran una cierta variedad tanto en el número de atributos (entre 4 y 58) como en el número de instancias (desde poco más de 100 hasta casi 50.000).

<sup>3</sup> Resulta necesario utilizar varias dependencias en cada iteración para evitar eliminar aquellos atributos que, a pesar de ser interdependientes con otros atributos del conjunto de datos, presentan información relevante con respecto al valor de la clase

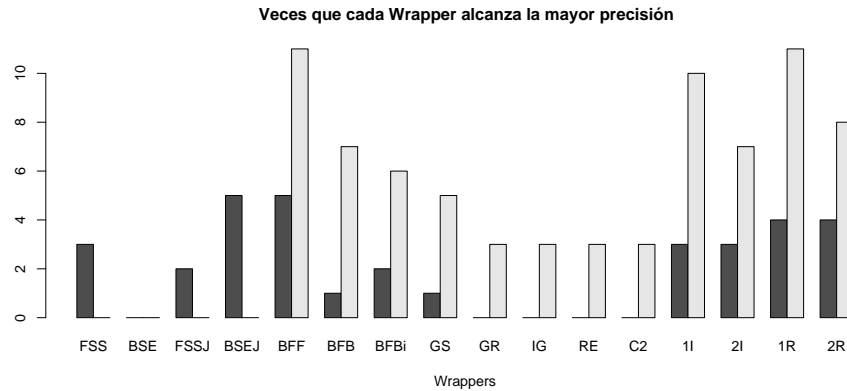
Se ha probado DGW utilizando la Ganancia de Información y Relief como evaluadores de atributos. Además, estos dos evaluadores se han combinado con la evaluación de 1 y 2 dependencias por cada paso. Esto ha permitido evaluar 4 variaciones de DGW: utilizando Ganancia de Información con 1 dependencia (1I) y con 2 dependencias (2I) por cada iteración y utilizando Relief con 1 (1R) y con 2 dependencias (2R) por iteración.

**Table 1.** Tabla resumen que muestra cuándo un wrapper obtiene una representación de los datos significativamente mejor (+) o peor (−) que la original. *N* significa que no existen datos para esa combinación de wrapper y conjunto de datos. Si no aparece nada, significa que el wrapper no consigue una representación de los datos que sea significativamente mejor o peor que la original.

| Cjto. Datos | FSS      | BSE      | FSSJ     | BSEJ     | BFF | BFB | BFBi | GS | GR | IG | RE | C2 | 1I | 2I | 1R | 2R |
|-------------|----------|----------|----------|----------|-----|-----|------|----|----|----|----|----|----|----|----|----|
| Tae         |          |          | +        | +        |     |     |      |    |    |    |    |    |    |    |    |    |
| Hayes-R.    |          |          | +        | +        |     |     |      |    |    |    |    |    | +  | +  | +  | +  |
| Haber.      |          |          |          |          |     |     |      |    |    |    |    |    |    |    |    |    |
| Iris        |          |          |          |          |     |     |      |    |    |    |    |    |    |    |    |    |
| Bupa        | +        | +        | +        | +        | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Wine        |          | +        |          | +        | +   | +   | +    | +  |    |    |    |    | +  | +  | +  | +  |
| Machine     | +        | +        | −        | −        | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Glass       | +        | +        | +        | +        | +   | +   | +    | +  | +  | +  |    |    | +  | +  | +  | +  |
| Ecoli       |          |          | −        |          |     |     |      |    |    |    |    |    |    |    |    |    |
| Pima Ind.   | +        | +        | +        | +        | +   | +   | +    | +  |    |    |    |    | +  | +  | +  | +  |
| TicTacT.    |          | +        | +        | +        | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Crx         | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Cmc         | +        | +        |          |          | +   | +   | +    | +  | +  | +  |    |    | +  | +  | +  | +  |
| Yeast       |          |          | −        | −        |     |     |      |    |    |    |    |    |    |    |    |    |
| Ionosp.     | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Wdbc.       | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Abalone     | +        | +        | +        | +        | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Segmen.     | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Spam        | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Letter R.   | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  | +  | +  | +  | +  | +  | +  | +  | +  |
| Pendigits   | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | +   | +   | +    | +  |    |    |    |    | +  | +  | +  | +  |
| Nursery     | −        |          | +        | +        |     |     |      |    |    |    |    |    |    |    |    |    |
| Adult       | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | −   | +   | −    | +  |    | +  | +  | +  | +  | +  | +  | +  |
| Total-15    | 5        | 8        | 5        | 7        | 8   | 8   | 8    | 8  | 6  | 6  | 4  | 5  | 9  | 9  | 9  | 9  |
| Total-23    | <i>N</i> | <i>N</i> | <i>N</i> | <i>N</i> | 14  | 16  | 14   | 16 | 12 | 13 | 11 | 11 | 17 | 17 | 17 | 17 |

Del estado del arte, y con el fin de poder comparar DGW, se han seleccionado los wrappers estudiados en los trabajos de Langley [16] y Pazzani [19] [10]: Forward Sequential Selection (FSS), Backward Sequential Elimination (BSE), Forward Sequential Selection and Joining (FSSJ) y Backward Sequential Elimination and Joining (BSEJ). Además, se han probado otros wrappers clásicos implementados en el paquete Weka [22] como son: Best First Forward (BFF), Best

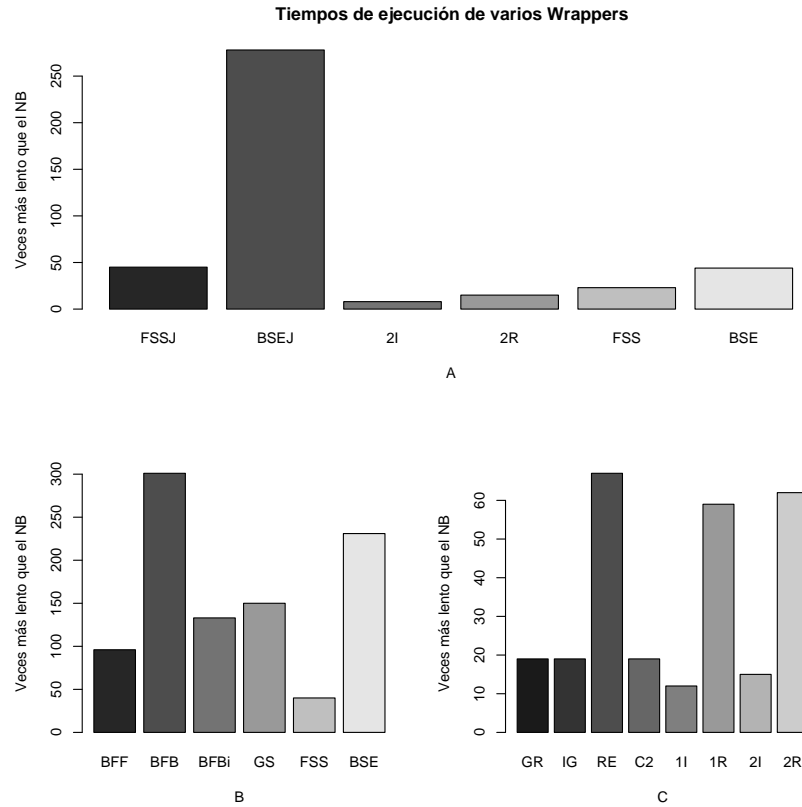
First Backward (BFB), Best First Bidirectional (BFBi), Genetic Search (GS), Ranking con Information Gain (IG), Ranking con Gain Ratio (GR), Ranking con Relief (RE), Ranking con Squared-Chi (C2).



**Fig. 2.** Número de veces que cada wrapper obtiene la mejor precisión. Las barras oscuras tienen en cuenta todos los wrappers pero solo los 15 conjuntos de datos para los que todos los wrappers tienen resultados. Las barras claras tienen en cuenta todos los wrappers menos FSS, BSE, FSSJ y BSEJ y todos los conjuntos de datos.

Para poder evaluar los resultados, se ha aplicado cada wrapper a cada conjunto de datos aplicando 10 veces validación cruzada a 10 carpetas. Con esto se ha obtenido la precisión media obtenida por el clasificador bayesiano simple al realizar el aprendizaje sobre la representación de los datos obtenida por cada wrapper, así como la desviación típica de la misma. Además, se ha aplicado un análisis de hipótesis nula a nivel .05 para determinar si la representación de los datos obtenida por cada wrapper ayuda al clasificador bayesiano simple a mejorar significativamente su clasificación. En la Tabla 1 se muestra un resumen de los resultados dónde un + indica que ese wrapper, para ese conjunto de datos, obtiene una representación de los datos significativamente mejor que la original. Un - indica que la representación de los datos es significativamente peor que la original. Para algunos wrappers (FSS, BSE, FSSJ, y BSEJ), y debido al elevado tiempo de ejecución de los mismos (cientos de veces más lentos que otros wrappers) solo se han podido realizar experimentos en 15 conjuntos de datos, por lo que se reporta con  $N$  en aquellos conjuntos de datos para los que no hay resultados. Para poder realizar comparaciones tanto con aquellos wrappers para los que solo hay 15 resultados, como para aquellos que han obtenido resultados todos los conjuntos de datos, se han generado dos filas de resumen que muestran el número de veces que un determinado wrapper obtiene una representación de

los datos que es significativamente mejor que la original <sup>4</sup> : Total-15 que solo tiene en cuenta los 15 conjuntos de datos para los que todos los wrappers obtienen resultados y Total-23 que tiene en cuenta todos los conjuntos de datos pero obvia los wrappers FSS, BSE, FSSJ y BSEJ por no disponer de resultados para algunos conjuntos de datos.



**Fig. 3.** Media del número de veces que cada wrapper es más lento que el clasificador bayesiano simple. Menor significa más rápido.

La Figura 2 muestra el acumulado de veces que cada wrapper obtiene la mejor precisión en algún conjunto de datos. Las barras oscuras tienen en cuenta todos los wrappers pero solo los 15 conjuntos de datos para los que todos los wrappers tienen resultados. Las barras claras tienen en cuenta todos los wrappers menos FSS, BSE, FSSJ y BSEJ y todos los conjuntos de datos.

<sup>4</sup> Cuando se obtiene una representación de los datos significativamente peor (–) se resta 1 al acumulado.

Además de las precisiones, también se han comparado los tiempos de ejecución de los diversos wrappers. Debido a que el tiempo de ejecución varía mucho en función del tamaño del conjunto de datos, se han intentado evitar los tiempos absolutos para evitar que los conjuntos de datos grandes sean los que más influyan en los resultados. Así pues, para cada wrapper y conjunto de datos, se ha calculado cuántas veces cada wrapper tarda más que el clasificador bayesiano simple. Finalmente, con todos los resultados, se ha calculado la media de veces que cada wrapper es más lento que el clasificador bayesiano simple (ver Figura 3). Las subfiguras B y C se han calculado utilizando los 23 conjuntos de datos, mientras que la subfigura A se ha calculado utilizando solo los resultados de los 15 conjuntos de datos para los que hay resultados para BSEJ y FSSJ.

## 5 Discusión de los Resultados y Conclusiones

Comparando los resultados de la Tabla 1, se puede concluir que el DGWrapper obtiene representaciones de datos significativamente mejores, para una posterior clasificación mediante el clasificador bayesiano simple, más frecuentemente que el resto de wrappers. Además, DGW (con la variación 1R) obtiene, al igual que BFF, en 11 ocasiones una representación de los datos que hace que el clasificador bayesiano simple obtenga la mejor precisión. Pero además es un 40% más rápido que el BFF, tal y como puede verse en la Figura 3. Con estos resultados podemos concluir que DGW se muestra como el mejor wrapper en combinación con el clasificador bayesiano simple, en términos de precisión clasificadora y, también, en términos de tiempos de ejecución, ya que se muestra como uno de los wrappers más rápidos, del orden de 100 veces más rápido que los propuestos por Pazzani.

La Tabla 1 muestra que los acercamientos backward (BSE, BSEJ, BFB) obtienen representaciones de los datos significativamente mejores más frecuentemente que sus correspondientes versiones forward (FSS, FSSJ, BFF), pero no se puede concluir que estas últimas sean peores ya que en la Figura 3 se muestra que los acercamientos forward obtienen las mejores precisiones más a menudo que los acercamientos backward.

Tanto FSSJ como BSEJ necesitan que todos los conjuntos de datos tengan atributos discretos, por lo que al encontrar algún atributo numérico, éste se discretiza a 5 intervalos. Cuando se clasifica con el clasificador bayesiano simple utilizando únicamente conjuntos de datos discretizados, el resultado obtenido muestra una gran correlación con los resultados de FSSJ y BSEJ, la precisión aumenta o disminuye, con respecto a la clasificación con los atributos originales, en los mismos conjuntos de datos. Así pues, parece que gran parte del rendimiento del FSSJ y BSEJ se debe al efecto de la discretización de los atributos. A pesar de esto, el concepto de construir atributos a partir de los atributos dependientes parece bastante interesante y puede ser útil integrarlo en el DGW para no perder parte de la variabilidad de los datos debida a la eliminación de atributos parcialmente redundantes.

## References

1. Rish, I.: An empirical study of the naive bayes classifier. In: International Joint Conference on Artificial Intelligence, American Association for Artificial Intelligence (2001) 41–46
2. Kononenko, I.: Semi-naive bayesian classifier. In: EWSL-91: Proceedings of the European working session on learning on Machine learning, New York, NY, USA, Springer-Verlag New York, Inc. (1991) 206–219
3. Kononenko, I.: Inductive and bayesian learning in medical diagnosis. *Applied Artificial Intelligence* **7**(4) (1993) 317–337
4. Lewis, D.D.: Naive (Bayes) at forty: The independence assumption in information retrieval. In Nédellec, C., Rouveirol, C., eds.: Proceedings of ECML-98, 10th European Conference on Machine Learning. Number 1398, Chemnitz, DE, Springer Verlag, Heidelberg, DE (1998) 4–15
5. Mitchell, T.: *Machine Learning*. 1 edn. McGraw Hill (1997)
6. Cortizo, J.C., Giráldez, J.I.: Discovering data dependencies in web content mining. In Gutierrez, J.M., Martinez, J.J., Isaias, P., eds.: IADIS International Conference WWW/Internet. (2004)
7. Cortizo, J.C., Giráldez, J.I.: Multi criteria wrapper improvements to naive bayes learning. In Corchado, E., Yin, H., Botti, V.J., Fyfe, C., eds.: IDEAL. Volume 4224 of Lecture Notes in Computer Science., Springer (2006) 419–427
8. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. *Machine Learning* **29**(2-3) (1997) 131–163
9. Kohavi, R.: Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. (1996) 202–207
10. Pazzani, M.: Constructive induction of cartesian product attributes. *ISIS: Information, Statistics and Induction in Science* (1996)
11. Domingos, P., Pazzani, M.J.: On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning* **29**(2-3) (1997) 103–130
12. Zhang, H., Ling, C.X., Zhao, Z.: The learnability of naive bayes. *Lecture Notes in Computer Science* **1822** (2000) 432–441
13. Hand, D.J., Yu, K.: Idiot’s bayes - not so stupid after all? *International Statistical Review* **69**(3) (2001) 385–299
14. Bellman, R.: *Adaptive Control Processes: a Guided Tour*. Princeton, University Press (1961)
15. Duch, W.: Filter Methods. In: *Feature Extraction, Foundations and Applications*. Springer Verlag (2004)
16. Langley, P.: Selection of relevant features in machine learning. In: Proceedings of the AAI Fall Symposium on Relevance. (1994)
17. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artificial Intelligence* **97**(1-2) (1997) 273–324
18. Langley, P., Sage, S.: Induction of selective bayesian classifiers. (1994) 399–406
19. Pazzani, M.J. In: Searching for Dependencies in Bayesian Classifiers. 5th Workshop on Artificial Intelligence and Statistics (1996)
20. Kittler, J.: Feature Selection and Extraction. In: *Handbook of Pattern Recognition and Image Processing*. Academic Press (1986)
21. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
22. Witten, I.H., Frank, E.: *Data Mining: Practical Machine Learning Tools and Techniques*. 2nd edn. Morgan Kaufmann (2005)