Mundo Fesc, 8 (16) (2018), 18-24 Julio /Diciembre 2018, ISSN 2216-0353 (impreso) - 2216-0388 (Online)

Implementación de Programación Genética en problemas de Clasificación Binaria

Implementation of Genetic Programming in Binary Classification problems

Joel Favila-Navarro[a], Alejandro Alvarado-Iniesta[b], Roberto Romero-López[c], Ivan Pérez Olguín[d*]

[a] Ingeniero en Mecatrónica, yogui16@live.com.mx, orcid: 0000-0001-9031-2494, Universidad Autónoma de Ciudad Juárez, Chihuahua, México

[b]Doctor en Ingeniería, alejandro.alvarado@uacj.mx, orcid: 0000-0002-3349-4823, Universidad Autónoma de Ciudad Juárez, Chihuahua, México

[c]Doctor en Ingeniería, rromero@uacj.mx, orcid: 0000-0003-0859-327X, Universidad Autónoma de Ciudad Juárez, Chihuahua, México

[d] Doctor en Ingeniería, ivan.perez@uacj.mx, orcid: 0000-0003-2445-0500, Universidad Autónoma de Ciudad Juárez, Chihuahua, México

Recibido: 16 de febrero de, 2018, Aceptado: 25 de mayo de 2018

Forma de citar: J. Favila-Navarro, A. Alvarado-Iniesta, R. Romero-López y I. Pérez-Olguín, "Implementación de Programación Genética en problemas de Clasificación Binaria", *Mundo Fesc*, vol. 8, no. 16, pp. 18-24, 2018.

Resumen

El presente trabajo muestra la implementación de programación genética para resolver problemas de clasificación binaria. Uno de los objetivos del presente trabajo es evidenciar el uso de la programación genética en este tipo de problemas; es decir, típicamente se utilizan otro tipo de técnicas, e.g., regresión, redes neuronales artificiales. Programación genética presenta una ventaja en comparación con estas técnicas, la cual es que no necesita una definición a priori de su estructura. El algoritmo evoluciona de manera automática hasta encontrar un modelo que mejor se adapte a un conjunto de datos de entrenamiento (aprendizaje supervisado). Así entonces, la programación genética puede ser considerada como una alternativa de uso para el desarrollo de sistemas inteligentes principalmente en el reconocimiento de patrones.

Palabras clave: Algoritmos evolutivos, aprendizaje de máquina, clasificación binaria, programación genética

Abstract

This work shows the implementation of genetic programming to solve binary classification problems. One of the objectives of this work is to demonstrate the use of genetic programming in this type of problems; that is, other types of techniques are typically used, e.g., regression, artificial neural networks. Genetic programming presents an advantage compared to those techniques, which is that it does not need an a priori definition of its structure. The algorithm evolves automatically until finding a model that best fits a set of training data (supervised learning). Thus, genetic programming can be considered as an alternative option for the development of intelligent systems mainly in the pattern recognition field.

Keywords: Binary classification, evolutionary algorithms, genetic programming, machine learning

Autor para correspondencia:
*Correo electrónico: ivan.perez@uacj.mx

Introducción

Hoy en día es más frecuente encontrar computadoras que integran la inteligencia artificial en sus operaciones, que generan como resultado aplicaciones que pretenden facilitar la vida diaria del ser humano a través del uso de las TICs como herramientas facilitadoras [1], i.e., teléfonos inteligentes, sistemas de seguridad, etc. Según [2], se puede decir que la virtualización, "es una abstracción de los recursos tecnológicos en donde se puede llegar a utilizar un servidor o muchos servidores siendo invisible para el usuario final". Una de estas operaciones es el reconocimiento de patrones, como la voz. El reconocimiento de voz que viene integrado casi en todos los teléfonos inteligentes modernos, aunado a eso el reconocimiento dactilar y facial. En la industria, el reconocimiento de patrones ha permitido el desarrollo de sistemas automatizados que cada vez son más utilizados y demandados.

Dentro de las etapas del reconocimiento de patrones la clasificación es una fase primordial, la cual se puede definir como la asignación de un valor a algo acorde a sus cualidades o características. La clasificación ha sido utilizada en distintos problemas de ingeniería, finanzas y medicina [3-11-25], por mencionar. La clasificación binaria es aquella donde la salida puede tomar solamente dos posibles valores, verdadero o falso, {0,1}. Algunos ejemplos son para el desarrollo de sistemas de diagnóstico, E.G., para la detección de enfermedades como la diabetes, se padece o no se padece. En el ámbito financiero, para el desarrollo de modelos para predecir si un negocio tiene riesgo financiero, es decir, si es factible invertir en el negocio o no [11]. Otra de las aplicaciones es en el ramo agrícola, mediante el uso de imágenes para clasificar campos y determinar si este campo es un campo potencial para cultivo [13-26].

Se han desarrollado diferentes algoritmos (los llamados algoritmos de aprendizaje de máquina) para lidiar con problemas de clasificación; entre los más comunes se encuentran los algoritmos de regresión, las redes neuronales artificiales, la lógica difusa y las máquinas de soporte vectorial. Una alternativa de uso a estos algoritmos de aprendizaje de máquina es la programación genética (PG). La PG es una técnica de aprendizaje de máquina que forma parte de los algoritmos evolutivos y, que está inspirada en procesos evolutivos los cuales emplean principios Darwinianos de supervivencia y reproducción del más apto [13].

Una de las ventajas de la PG, es que ésta no requiere de una definición a priori de su estructura, i.e., en redes neuronales, se requiere definir el número de capas y neuronas antes de aplicar algún algoritmo de entrenamiento. Por lo tanto, este trabajo se enfoca en la implementación de la PG en problemas de clasificación binaria. Es importante desarrollar evidencia de la implementación de esta técnica para que pueda ser aplicada principalmente en problemas de la vida real.

Antecedentes

Programación genética. Programación genética es un algoritmo evolutivo que automáticamente evoluciona programas, funciones o cualquier otro tipo de expresión simbólica, que lleve a cabo algún tipo de cálculo, para resolver una tarea o problema en específico [14,15]. Generalmente, estos programas son representados como estructuras de longitud variable tal como un árbol de sintaxis,

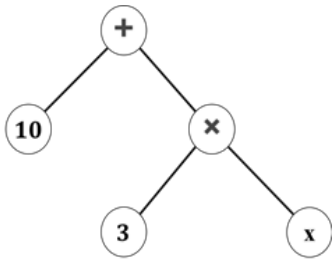

Figura 1. Representación de árbol suma(10,3*x)

Además, un conjunto de símbolos es definido el cual es nombrado conjunto primitivo P. Este conjunto contiene un subconjunto de funciones de paridad diferente, llamado el conjunto de funciones, e.g., F={+,-,×,÷}, y un subconjunto de elementos como las variables de entrada del problema u otros como valores constantes, llamado

el conjunto de terminales, e.g., T={x_i,R^m}, de tal modo que P=F U T.

La aplicación más común de PG, quizás, es la regresión simbólica la cual intenta encontrar una expresión matemática que mejor se ajuste a un conjunto dado de datos de entrenamiento (aprendizaje supervisado) [16]. Así entonces, el objetivo es buscar esa expresión K^O:R^n→R que ajuste un conjunto T={(x_1,y_1),…,(x_p,y_p)} de p datos de entrada/salida con xERn y yER o y E{0,1} para un problema de clasificación binaria.

El problema general de regresión simbólica puede ser definido en (1):

$$K^{O} \leftarrow \underset{K \in G}{arg\,min}\, f(K(x,\beta),y) \qquad (1)$$

Donde G es el espacio de búsqueda definido por el conjunto primitivo de funciones y terminales, P=F U T; f es la función objetivo la cual puede ser la diferencia entre la salida del programa K(x,β) y la salida deseada y.

Como cualquier otro algoritmo evolutivo en PG la población inicial es generada aleatoriamente. Así mismo, se aplican operadores genéticos tales como la selección, cruza y mutación. Aunque los operadores de cruza y mutación difieren de otros algoritmos evolutivos, la idea básica es la misma. Mayores detalles e información acerca de PG puede ser encontrada en [15].

Revisión de literatura

En esta subsección se presentan algunos trabajos que se han desarrollado en los últimos años para resolver problemas de clasificación, tanto binaria como multiclase. Se han utilizado diferentes algoritmos de aprendizaje en diversos campos de aplicación. Por mencionar, [3] desarrollaron un sistema binario de clasificación de noticias mediante el uso de máquinas de soporte vectorial (MSVs). Autores como [17] presentan un enfoque nuevo de clasificación mediante la combinación de regresión logística con decisión-theoretic rough sets (DTRS, por sus siglas en inglés) para resolver problemas binarios y multiclase. [18] proponen la incorporación de un optimizador numérico en el algoritmo nominal de PG se hace uso de redes neuronales artificiales (RNAs) para clasificar señales cardiacas con el objetivo de demostrar si un electrocardiograma era normal o podría presentar algún problema. Autores como [12] emplearon lógica difusa y RNAs para la clasificación de campos de cultivo. Así mismo [5] presenta un método basado en regresión logística multinomial para resolver un problema multiclase de clasificación hiperespectral. En el 2016, realizan una comparación entre RNAs y MSVs para clasificar fallas en la producción de rodillos, obteniendo como resultado que las MSVs proveen mejores resultados. Similarmente, [7] y emplean MSVs para clasificar enfermedades de la piel [6] y presentan una red de creencia profunda (DBN, por sus siglas en ingles) para resolver un problema de clasificación sobre la calidad del sueño [8].

Si bien son más extensos los trabajos publicados en diferentes áreas y mediante el uso de diferentes algoritmos, se puede notar entones que las RNAs y MSVs son de las más reportadas en la literatura. Sin embargo, están requiere de una definición a priori de su estructura, i.e. en RNAs hay que establecer el número de capas y en MSVs la función kernel, antes de entrenar el algoritmo. PG es un algoritmo evolutivo que automáticamente transforma poblaciones de programas en nuevas poblaciones por medio del uso de operadores genéticos de búsqueda. Por lo tanto, no requiere del establecimiento de una estructura a priori. Si bien hay ciertos parámetros que deben de fijarse al inicio del algoritmo, la estructura final se obtiene de manera automática, es decir se genera de manera autónoma un modelo de clasificación generación tras generación.

Experimentación

Para problemas de clasificación binaria, se trató PG como un algoritmo supervisado donde un conjunto de patrones de entrenamiento, x ERn, es utilizado para mapear una función g(x) E Rn→R. Después, se inserta una función escalón unitario (2).

$$G(f(x)) = \begin{cases} 0 & \text{si } f(x) < 0 \\ 1 & \text{si } f(x) \geq 0 \end{cases} \qquad (2)$$

De tal modo que G(x) : R→{0,1}, como se muestra en la Figura 2.

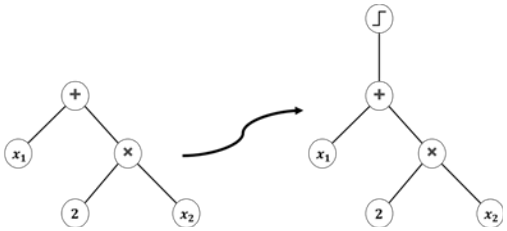

Figura 2. Transformación de árbol a salida binaria

En problemas de clasificación, se puede utilizar la exactitud como métrica para evaluar el rendimiento de los modelos. La exactitud se puede definir como la fracción de predicciones correctas sobre el número total de casos. Específicamente, para problemas de clasificación binaria se puede calcular en términos de predicciones positivas y negativas (3).

$$\text{exactitud} = \frac{VP+VN}{VP+VN+FP+FN} \qquad (3)$$

donde VP son los verdaderos positivos, VN los verdaderos negativos, FP son los falsos positivos y FN los falsos negativos [19,20].

En este trabajo se utilizó la función objetivo (4) como métrica para evaluar el rendimiento de la PG en problemas de clasificación binaria.

Función objetivo=1-exactidud (4)

Donde es deseado encontrar valores iguales o cercanos a cero.

El algoritmo estándar de PG fue implementado utilizando la herramienta de GPLAB1 Matlab. El conjunto de problemas a resolver cubre una serie de problemas de referencia de clasificación binaria tomados del repositorio del centro de aprendizaje de máquina y sistemas inteligentes UCI [21].

La Tabla 1 muestra un resumen de los problemas evaluados.

Tabla 1. Resumen de los problemas de clasificación binaria utilizados para evaluar el algoritmo de PG

Nombre	# de atributos	# de instancias
XOR	2	4
Globos	8	20
Crioterapia	6	91
Trenes	32	10
Tic Tac	9	958

Las salidas de interés reportadas, son la función objetivo en (4) y el tamaño del programa generado, empleado para evaluar el rendimiento de cada modelo en términos de los recursos consumidos. La configuración utilizada para cada uno de los problemas se muestra en la Tabla 2.

Tabla 2. Parámetros del algoritmo de PG

Parámetro	Valor
Corridas	10
Población	200
Generaciones	150
Conjunto de entrenamiento	70%
Conjunto de prueba	30%
Inicialización de árbol	Ramped half-and-half, max. profundidad 6
Conjunto de funciones	+,-,×, sin, cos, log, sqrt, tan, tanh
Conjunto de terminales	Atributos de entrada x's, constantes
Selección	Torneo, tamaño 2
Operador de cruza	Standard sub-tree crossover, 0.8 prob
Operador de mutación	Mutation probability per node 0.15
Elitismo	Mejor individuo sobrevive

Fuente: [22]

Resultados y análisis

Las Figuras 3-6 resumen los resultados de la implementación de PG en los problemas 3 y 5 de clasificación binaria seleccionados. Los resultados muestran las gráficas de convergencia de la función objetivo y el tamaño del programa con respecto al número de generaciones, mostrando la media de las corridas. El primer problema es la compuerta XOR, considerada un problema relativamente simple. Dado que el número de muestras es pequeño, no se evalúa la exactitud en datos de prueba. Los resultados de este problema no se grafican ya que en la generación

de la población inicial se logró encontrar un modelo con ajuste perfecto, es decir una exactitud de 1 con un valor de la función objetivo igual a 0. Lo mismo sucedió con el problema de los globos, al generar la población inicial se encontró un modelo con ajuste perfecto en todas las corridas.
Para el problema de la crioterapia los resultados fueron un poco diferentes, las Figuras 3-4 muestran los resultados obtenidos para este problema. Dado que el número de instancias es mayor que los problemas anteriores, consideramos datos de prueba. Los resultados muestran que en promedio se encontró una exactitud perfecta antes de la generación 30 para el conjunto datos de entrenamiento; mientras que con una mediana de 0.80 en la exactitud para los datos de prueba.

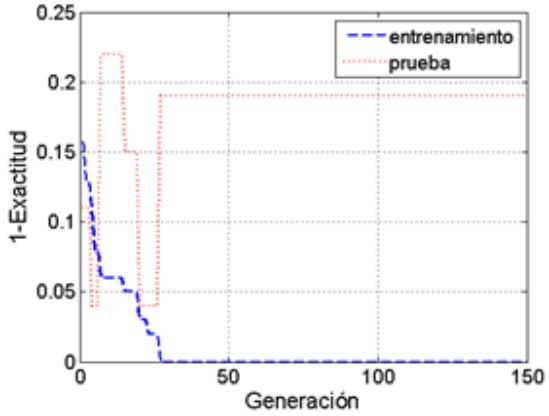

Figura 3. Resultado de la función objetivo problema crioterapia

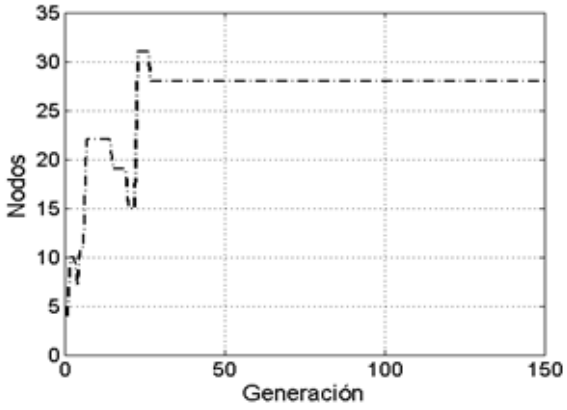

Figura 4. Tamaño de modelo problema crioterapia

Para el problema de trenes, los resultados muestran que en promedio antes de la generación 5 se encontró un modelo con exactitud perfecta para las 10 corridas, por lo que consideramos al igual que los problemas 1 y 2 no es necesario mostrar las gráficas obtenidas. Finalmente, para el problema de tic tac, que es considerado el más complejo debido al número de atributos e instancias, las Figuras 5-6 muestran las gráficas de convergencia de la función objetivo tanto para los datos de entrenamiento como para los datos de prueba, así como la mediana del tamaño de programa. Se puede ver entonces que el algoritmo es capaz de encontrar un modelo de clasificación con una exactitud mayor al 95%, en promedio, para este problema.

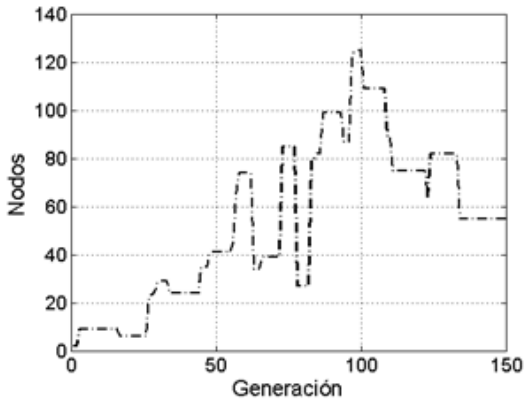

Figura 5. Resultado de la función objetivo problema Tic Tac

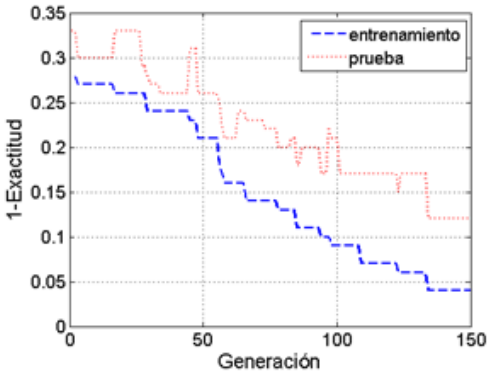

Figura 6. Tamaño de modelo problema Tic Tac

Conclusiones

En este estudio se abordó la implementación de programación genética para resolver problemas de clasificación binaria. Se agregó una función escalón unitario en la punta del árbol de tal manera que el resultado se limitará a proveer un falso o un verdadero, {0,1}. Se probó en problemas simples (considerados así por el número de instancias y atributos) donde se obtuvieron resultados inmediatos generando modelos con ajuste perfecto. Si bien los problemas son relativamente sencillos, los resultados demuestran el poder de PG para encontrar modelos de manera rápida y eficiente. El problema 5 fue el más complejo dado el número de instancias, sin embargo, se puede decir que se obtuvieron resultados satisfactorios ya que se encontró un modelo con una exactitud por arriba del 95% en promedio. Si bien los resultados son buenos en general, hay diversos puntos que pueden tratarse en específico y mejorarse; por ejemplo, el tamaño de los modelos. En problemas más complicados el tamaño del modelo generado puede incrementarse demasiado, y es un asunto que se debe tratar ya que va directamente ligado al uso de los recursos (en alguna aplicación real la rapidez de evaluación del modelo puede ser critica).

Se recomienda a futuro la incorporación de algún método numérico de búsqueda local, para optimizar las terminales numéricas y así encontrar arboles de tamaño menor. De igual manera, implementar la programación genética en problemas de la vida real como lo puede ser un sistema de visión con enfoque a la calidad en un sistema de producción.

Referencias

[1] J. Pabón-Gómez, “Las TICs y la lúdica como herramientas facilitadoras en el aprendizaje de la matemática”, *Eco Matemático*, vol. 5, n.º 1, pp. 37-48, ene. 2014. https://doi.org/10.22463/17948231.62

[2] N. Hernández y A. Flórez-Fuentes, "Computación en la Nube", *Mundo FESC*, vol. 4, n.º 8, pp. 46-51, dic. 2014.

[3] A. A. Holts-Corey y C.L. Riquelme-Jerez. “Desarrollo de un sistema de clasificación binaria automática de noticias con máquinas de aprendizaje.” Tesis de maestria, Pontificia Universidad Católica de Valparaíso, Chile, 2010.

[4] B. Mohamed, A. Issam, A. Mohamed y B. Abdellatif, “ECG image classification in real time based on the Haar-like features and artificial neural networks.” *Procedia Computer Science,* vol. 73, pp. 32-39, 2015.

[5] T.V.N. Nidhin-Prabhakar, G. Xavier, P. Geetha and K.P. Soman, “Spatial preprocessing based multiomial logistic regression for hyperspectral image classification.” *Procedia Computer Science*, vol. 46, pp. 1817-1826, 2015.

[6] K. Parikh and T.P. Shah, “Support vector machine - A large margin classifier to diagnose skin illness.” *Procedia Technology*, vol. 23, pp. 369-375, 2016.

[7] J.P Patel, and S.H Upadhyay, “Comparison between artificial neural network and support vector method for a fault diagnosis in rolling element bearings”. *Procedia Engineering,* 144, 390-397, 2016.

[8] I.N. Yulita, M.I. Fanany and A.M. Arymuthy, “Bi-directional long short-term memory using quantized data of deep belief networks for sleep stage classification.” *Procedia Computer Science,* vol. 116, pp. 530-538, 2017.

[9] J. Rojas Gómez, "El pensamiento Abstracto a partir de la interdisciplinariedad de las Matemáticas", *Eco Matemático*, vol. 8, pp. 51-53, jun. 2018. https://doi.org/10.22463/17948231.1382

[10] M. Largo-Leal, P. Jaimes-Espinoza, y Y. Largo-Leal, "Abordando el aprendizaje de las matemáticas", *Eco Matemático*, vol. 5, n.º 1, pp. 60-65, ene. 2014. https://doi.org/10.22463/17948231.53

[11] JC. Hernández-Suarez, L. Jaimes-Contreras, y R. Chaves-Escobar, "Modelos de aplicación de ecuaciones diferenciales de primer orden con geogebra: actividades para resolver problemas de mezclas", *Mundo Fesc*, vol. 6, n.º 11, pp. 7-15, sep. 2016.

[12] J.F. Diaz-Cordova, E. Coba-Molina and P. Navarrete-López, "Fuzzy logic and financial risk. A propossed classification of financila risk to the cooperative sector." Contaduría y Administración, vol. 62, pp. 33-34, 2017.

[13] S. Murmu and S. Biswas. "Application of fuzzy logic and neural network in crop classification." Aquatic Procedia, vol. 4, pp. 1203-1210, 2015.

[14] Y. Medina Vargas y H. Miranda Mnedez, "Comparación de algoritmos basados en la criptografía simétrica DES, AES y 3DES", Mundo Fesc, vol. 5, n.º 9, pp. 14-21, dic. 2015.

[15] P.G. Espejo, S. Ventura and F. Herrera, "A Survey on the Application of Genetic Programming to Classification." IEEE Transactions of Systems, Man, and Cybernetics, vol. 40, pp. 121-144, 2009.

[16] J. Koza. Genetic programming: On the programming of computers by means of natural evolution. Cambridge: MIT Press, 1992.

[17] R. Poli, W.B. Langdon and N.F. McPhee, A field guide to genetic programming. San Francisco: Lulu Enterprises, 2008..

[18] E.Z. Flores, L. Trujillo, O. Schütze and P. Legrand, "Evaluating the effects of local search in genetic programming." En EVOLVE-A bridge between probability, set oriented numerics, and evolutionay computation V, 2014, pp. 213-228.

[19] D. Liu, T. Li and D. Liang,"Incorporating logistic regression to decision-theoretic rough sets for classifications." International Journal of Approximate Reasoning, vol. 55, pp. 197-210, 2014.

[20] E.Z. Flores, L. Trujillo, O. Schutze and P. Legrand. "A local search approach to genetic programming for binary classification." En Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 1151-1158, 2015.

[21] J. Eggermont, A.E. Eiben and J.I. Hemert, "Adapting the fitness function in GP for data mining." En Proceedings of the Second European Workshop on Genetic Programming, pp. 193-202, 1999.

[22] S.M. Winkler, M. Affenzeller and S. Wagner. "Advanced Genetic Programming Based Machine Learning." Journal of Mathematical Modelling and Algorithms, vol. 6, pp. 455-480, 2007.

[23] D. Dua and E. Taniskidou, "UCI Machine Learning Repository." Internet: http:// archive. ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science, 2017.

[24] GPLAB A Genetic Programming Toolbox for MATLAB, 2017. [En línea]. Disponible en http://gplab.sourceforge.net/.

[25] Y. M. Moreno-Sánchez, R. M. García-Manrique, G. R. Robles-Gil y J. K. Porras-Lara,"Valoración del riesgo biopsicosocial en gestantes de Cúcuta", Aibi revista de investigación, administración e ingeniería, vol. 7, nº 1, pp. 19-22, 2019.

[26] B. N. Arias, "El consumo responsable: educar para la sostenibilidad ambiental", Revista AiBi, vol. 4, nº 1, pp. 32-37, 2016