II CONFERENCIA INTERNACIONAL DE PROCESAMIENTO DE LA INFORMACIÓN (CIPI 2019)

Implementación de métodos para el preprocesamiento de datos usando la Teoría de Conjuntos Aproximados (RST) en Python

Implementation of methods for data preprocessing using Rough Set Theory (RST) in Python

Beatriz Bello García[1], María Matilde García Lorenzo[2], Carlos Morell Pérez[3], Rafael Bello Pérez[4]

1-Universidad Central Marta Abreu de Las Villas, Cuba, bbgarcia@uclv.cu

2-Universidad Central Marta Abreu de Las Villas, Cuba. mmgarcia@uclv.edu.cu

3-Universidad Central Marta Abreu de Las Villas, Cuba. rbellop@uclv.edu.cu

4-Universidad Central Marta Abreu de Las Villas, Cuba. cmorellp@uclv.edu.cu

Resumen:

La comunidad científica reconoce el papel de la Teoría de los Conjuntos Aproximados(RST) para el análisis inteligente de los datos. En este trabajo, se describe la implementación en Python de una biblioteca para el preprocesamiento de datos, etapa previa determinante para el descubrimiento de conocimiento. En particular, son implementados las definiciones y medidas principales basadas en RST, así como métodos de selección de rasgos y ejemplos. Se verifica la eficacia de los métodos a partir de las pruebas realizadas desde bases de datos del UCI repositorio.

Abstract: *The scientific community recognizes the role of the Rough Set Theory (RST) for the intelligent data analysis . In this paper, we describe the Python implementation of a library for data preprocessing, a decisive preliminary stage for data mining. In particular, the definitions and main measures based on RST are implemented, as well as the methods of selection of features and examples. The accuracy of the methods is verified from the tests performed from repository UCI databases .*

Palabras Clave: Aproximación inferior; Aproximación superior; Metaheuristicas; Reducción de datos; Consistencia; Aprendizaje automatizado.

Keywords: *Rough set theory; Metaheuristics; Feature learning, Data reduction; Consistency; Machine learning.*

1. Introducción

En la actualidad la disponibilidad de los volúmenes de datos e información genera la necesidad de desarrollar procesos de aprendizaje para la extracción y descubrimiento de conocimiento. Para extraer de forma automatizada conocimiento subyacente en la información se requiere el desarrollo de métodos y técnicas del Aprendizaje Automatizado (Mitchell, 1997). Para el desarrollo de funciones tales como la clasificación, la predicción numérica, la asociación o el agrupamiento que facilitan la minería de datos y el descubrimiento de conocimiento es imprescindible preprocesar la información o la muestra de datos de donde se quiere aprender para elevar la calidad del proceso.

El preprocesamiento de datos, engloba a todas las técnicas de análisis de datos que permite mejorar la calidad de los datos(García, Luengo, Herrera, 2015). La preparación de los datos puede generar un conjunto más pequeño de datos que el original con una mayor calidad a partir de técnicas como selección de datos relevantes, eliminación de anomalías, eliminación de datos duplicados, etc.. Está demostrado que una baja calidad de los datos conduce en la mayoría de los casos a una baja calidad del conocimiento extraído o a un mayor costo computacional del procesamiento (García, 2016). De modo que es importante desarrollar métodos que faciliten el procesamiento de datos y utilizar adecuadamente los que ya existen.

Una de las teorías empleadas para el análisis inteligente de los datos es la Teoría de los Conjuntos Aproximados (Rough Set Theory, RST) (Bello, Nowe et al., 2006). RST abrió una nueva dirección en el desarrollo de teorías sobre la información incompleta y es una poderosa herramienta para el análisis de datos. Esta teoría es de gran utilidad en el preprocesamiento de los datos, tanto para encontrar conjuntos reducidos de atributos y editar conjuntos de entrenamiento para resolver problemas de clasificación supervisada, como para generar conocimiento a priori sobre un conjunto de datos.

Python es ideal para trabajar con grandes volúmenes de datos porque favorece su extracción y procesamiento, siendo el elegido por las empresas de Big Data (Herrera, 2014). A nivel científico, posee una amplia biblioteca de recursos con especial énfasis en las matemáticas para aspirantes a programadores en áreas especializadas (McKinney, 2012). El uso de Python como el lenguaje de programación para implementar los principales métodos computacionales facilita el desarrollo de técnicas de extracción de conocimiento o minería de datos a partir del uso de paquetes que permiten la computación de alto rendimiento y análisis de datos.

Sin embargo, en la mayoría de las herramientas para el descubrimiento de conocimiento no se incluyen métodos basados en conjuntos aproximados. De ahí la necesidad de crear una biblioteca que implemente los principales métodos de RST para el preprocesamiento de datos.

2. Preprocesamiento de datos basada en RST

RST se define a partir de dos componentes básicas: un Sistema de Información o un Sistema de Decisión y una relación de inseparabilidad. Los conceptos básicos de la RST son las aproximaciones inferiores y superiores de un subconjunto de un universo, $X \subseteq U$ (Komorowski and Pawlak, 1999). Sea el sistema de información (U,A), donde U es un universo de objetos y A un conjunto de atributos, y los conjuntos $B \subseteq A$ y $X \subseteq U$. Formalmente las aproximaciones se definen por las expresiones (1) y (2),

$$B_*(X) = \{x \in U \mid B(x) \subseteq X\} \quad (1)$$

$$B^*(X) = \{x \in U \mid B(x) \cap X \neq \phi\} \quad (2)$$

A partir de ellos se define la región límite o frontera de un conjunto por la expresión (3):

$$BN_B(X) = B^*(X) - B_*(X) \quad (3)$$

La Teoría de los Conjuntos Aproximados ofrece algunas medidas para analizar los sistemas de información (Skowron 1999; Arco, Bello et al., 2006) y entre ellas están: Precisión de la aproximación, Calidad de la aproximación, Función de pertenencia aproximada y Calidad de la clasificación, esta última describe la inexactitud de las clasificaciones aproximadas y expresa la proporción de objetos que pueden estar correctamente clasificados en el sistema.

Para el preprocesamiento de datos resulta indispensable el desarrollo métodos de reducción de datos que involucra la selección de rasgos u ordenamiento de estos y la reducción de ejemplos o instancias.

La selección de rasgos es útil para reducir la dimensionalidad del problema, y con ello simplificar el clasificador o la función de minería de datos a emplear, incrementa la velocidad de manipulación de los datos, y mejora el desempeño reduciendo la influencia de los ruidos. Se dice que la selección de rasgos es el proceso de búsqueda a través del conjunto A de n rasgos para tratar de encontrar el mejor subconjunto entre los 2^{n-1} subconjuntos candidatos de acuerdo a alguna medida de evaluación, por esta razón tenemos dos componentes: una función de evaluación usada para evaluar un subconjunto de rasgos, en cuanto a la calidad del mismo o su capacidad discriminante (ejemplos: distancia entre clases, dependencia probabilística, medida de información (Ej, medida de consistencia) y un procedimiento de generación usado para generar subconjuntos de rasgos candidatos (García, Luengo, Herrera, 2015).

El procedimiento para la generación puede ser una estrategia de búsqueda exhaustiva, a ciegas o heurística (Russell, Norvig, 2016), en este último caso pueden utilizarse metaheurísticas como Algoritmos Genéticos (Goldberg, 1989), PSO (Kennedy, 2011), Colonia de hormiga (Dorigo, Birattari, Stützle, 2010). Métodos como Focus; búsqueda exhaustiva, ABB: búsqueda completa, SetCover: búsqueda heurística, LVF: búsqueda probabilística, QBB: búsqueda híbrida se implementan en WEKA (Witten, I, Frank, E., Hall M., Pal, C., 2017) como procedimientos para la generación de candidatos.

El uso de reductos en la selección y reducción de atributos ha sido ampliamente estudiado (Ahn, 2000; Zhong, Dong et al., 2001). En (Caballero, 2005) se define el algoritmo QuickReduct para la selección de rasgos basado en estos conceptos. Estos métodos utilizan la medida calidad de la clasificación como función de evaluación heurística de un subconjunto de datos.

La selección de los objetos de un conjunto de entrenamiento es un problema en todos los modelos que realizan inferencias a partir de ejemplos, se conoce como edición de conjuntos de entrenamiento. La edición se realiza para eliminar los casos que inducen a

una incorrecta clasificación, y selecciona un conjunto representativo y reducido. Las técnicas de edición, aunque también producen la eliminación de ejemplos, tienen como objetivo fundamental el obtener una muestra de entrenamiento de mejor calidad para una mejor precisión del sistema. Según (García, Villuendas et al., 2005) los métodos de edición pueden ser: de selección de objetos y de construcción de objetos.

La idea básica de aplicar RST para editar los conjuntos de entrenamiento es la siguiente: en el conjunto de entrenamiento se colocan los objetos del sistema de decisión inicial que pertenecen a la aproximación inferior de cada clase. Esto es igual a decir que el conjunto de entrenamiento para el primer algoritmo será la región positiva del sistema de decisión. En esta forma, los objetos que están etiquetados incorrectamente o muy cerca de la frontera de decisión pueden ser eliminados del conjunto de entrenamiento, los cuales afectan la calidad de la inferencia. Como en la aproximación inferior de cada clase estarán aquellos objetos que con certeza pertenecen a dicha clase, se garantiza la eliminación de cualquier presencia de ruido en el sistema de decisión. El algoritmo Edit1RS (Bello, García, Pérez, 2012) es un representante de estas ideas.

En (Caballero, Bello et al., 2006b) se presentan modificaciones al anterior y se presenta Edit3RS, el cual construye objetos a partir de re-etiquetar aquellos que pertenecen al conjunto frontera. El método Edit3RS es un método además de construcción de objetos, pues no solo selecciona objetos, sino que puede obtener nuevos a partir del proceso de re-etiquetamiento que se aplica a los objetos de la frontera, donde puede ser que algunos de estos cambien su clase.

En este trabajo se propone una modificación del método Edit1RS, al cual se denomina Edit2RS y en el que se cambia la forma de calcular la aproximación inferior de un objeto, de modo que a esta pertenecen los objetos que tienen un grado de pertenencia al conjunto que se aproxima mayor que un umbral dado, cuando este umbral tiene valor 1 se tiene los mismos resultados que con la definición original; esta propuesta se basa en la definición de aproximaciones con enfoque probabilístico propuesta en (Pawlak, Wong, Ziarko, 1988), ver expresión (4);

$$B_*(X)= \{x \in U \mid \mu^{\alpha,B}{}_X(x) \geq \alpha\} \qquad (4)$$

El método Edit2RS ofrece una variante intermedia entre los métodos Edit1RS y Edit3RS, el primero solo considera casos completamente consistentes, el segundo considera todos los objetos, lo que para aquellos que generan inconsistencias le trata de modificar su clase, usando el grado de membresía aproximada (aunque esto no garantiza que el conjunto de aprendizaje resultante sea totalmente consistente); mientras el nuevo método hace más flexible la definición de la aproximación inferior permitiendo cierto grado de inconsistencia (definido por el umbral que se establezca).

3. Biblioteca en Python para preprocesamiento de datos basado en RST

Se desarrolló una biblioteca para el preprocesamiento de datos usando la RST. La biblioteca está formada por varios paquetes en los que se implementan las medidas y funciones necesarias para el procesamiento de datos a partir de la RST.

En un primer paquete se implementan un conjunto de definiciones que caracterizan la Teoría de los Conjuntos Aproximados al igual que los conceptos fundamentales Aproximación Inferior y Aproximación Superior. Además de la implementación de medidas para la inferencia como Precisión de la aproximación, Calidad de la aproximación, Función de pertenencia aproximada y Calidad de la clasificación, así como medidas para la dependencia y peso de los atributos.

Para el preprocesamiento como parte de la reducción de datos se implementaron dos paquetes uno para la selección de rasgos y otro para la selección o edición de ejemplos. En el paquete de selección de ejemplos se implementaron los algoritmos Edit1RS, Edit2RS, y Edit3RS que constituyen métodos de construcción y selección de prototipos. En ambos se utilizan las funciones que permiten obtener las aproximaciones inferior y superior del conjunto de entrenamiento referido y que fueron implementadas en el paquete anterior.

El paquete de selección de rasgos contiene el algoritmo QuickReduct para la selección de rasgos basados en los principales conceptos y definiciones de la RST, el cual utiliza como método de búsqueda el método ascensión de colina (hill-climbing). Este paquete presenta además la solución al problema de cálculo de reductos utilizando las metaheurísticas Algoritmos Genéticos (AG) y Optimización Enjambre de Partículas (PSO).

El cálculo de reductos a partir de AG se implementó de la siguiente forma: Cada individuo es un posible reducto o un subconjunto de atributos. Los individuos se representan como cadenas de longitud n, donde n representa la cantidad de atributos, en estas cadenas si la posición i-ésima tiene valor 1 entonces, el atributo i-ésimo está en el subconjunto denotado por la cadena. La función de calidad depende de: la cantidad de unos en el cromosoma y la cantidad de pares de objetos (con diferentes valores de decisión) separados por los atributos que están en el subconjunto representado por el individuo o cromosoma. Por lo que se crean dos clases una Cromosoma y otra Población. La clase Cromosoma tiene las funciones necesarias para generar un cromosoma, el cálculo de la función de calidad para cada uno de ellos y la función que realiza la operación de mutación. En la clase Población están implementadas las operaciones selección y cruce; además de las funciones que permiten generar nuevas poblaciones.

Para realizar el cálculo de reductos usando AG es necesario crear una Población con sus atributos necesarios: cantidad de individuos por población, tamaño del cromosoma o conjunto de rasgos y cantidad de iteraciones, y la nueva población se crea llamando a la función evolución.

Para el cálculo de reductos usando PSO se tiene que cada pájaro (partícula) es tratado como un punto en un espacio N dimensional el cual ajusta su propio "vuelo" de acuerdo a su propia experiencia y la experiencia del resto de la banda. La bandada (swarm) vuela por el espacio buscando regiones prometedoras. Cada partícula tiene una posición y velocidad en el espacio de búsqueda. Tiene una medida de calidad parecida a la "fitness" de un individuo, ellas "aprenden" ajustando su posición y velocidad, teniendo en cuenta su mejor posición hasta el momento. A diferencia de utilizar AG en este se hace un ordenamiento de los rasgos (Wang, Yang, Teng, Xia, & Jensen, 2007).

En el caso del cálculo de reductos usando PSO se crearon dos clases Bandada y Partícula. En Partícula se implementó el cálculo del vector velocidad y la partícula al igual que la función de calidad de cada una de estas. Tanto en AG como en PSO la función de calidad o fitness que se utiliza es la medida de calidad de la clasificación para medir la consistencia del sistema de decisión. Esta medida expresa que si S es un conjunto de rasgos consistentes, no existen dos instancias con los mismos valores en S que pertenecen

a clase diferentes, de esta forma la medida no maximiza la separabilidad entre clases, sino que intenta conservar la potencia discriminante del conjunto de rasgos original.

Para mejorar el subconjunto de rasgos resultantes o el reducto se utilizó un regularizador con el cual no solo se tiene en cuenta calidad del conjunto de rasgos sino también se busca la menor cantidad de rasgos siempre y cuando la calidad del reducto sea igual a la calidad inicial del sistema de decisión (Wang, Yang, Jensen, & Liu, 2006).

4. Discusión de los resultados

Para evaluar la eficacia de los métodos implementados para la selección de rasgos se utilizan cinco bases de casos del repositorio UCI (Bache, Lichman, 2013). La tabla 1 muestra los resultados de evaluar la selección usando PSO sin regularizador y con regularizador.

Bases de Casos	Cantidad de rasgos	Cantidad de iteraciones	Longitud promedio de los reductos sin regularizador	Longitud promedio de los reductos con regularizador
breast-cancer-wisconsin.arff	9	100	5.3442	4.7025
heart-statlog.arff	13	100	7.3540	6.3118
labor.arff	16	100	9.4784	7.5381
hepatitis.arff	19	100	9.5650	10.2337
diabetes.arff	8	100	3.8612	3.5681

Tabla 1. Resultados de evaluar PSO

De igual forma se realizó el análisis usando AG mostrándose similares resultados. Se muestra cómo se logra la reducción en la cantidad de rasgos y que el uso de un regularizador disminuye la cantidad de estos. Se analizan los resultados alcanzados con ambos métodos corroborados con el test Wilcoxon (Wilcoxon, 1945).

Seguidamente se realiza el análisis de los métodos edición a partir de comparar el número de ejemplos resultantes de aplicar los métodos a diferentes bases de datos. En la tabla 2 se muestra el efecto de reducción de los métodos, no aparece el método Edit3RS, pues este no elimina objetos sino que re-etiqueta los que pertenecen a las fronteras de las clases usando el grado de pertenencia de estos a las clases. Se puede apreciar una reducción

significativa en la mayoría de los casos de la cantidad de objetos, mayor en el caso de Edit1RS.

Para analizar en qué medida esta reducción afecta la eficacia de la clasificación se hace un estudio usando varios métodos de clasificación conocidos.

Bases de casos	# de ejemplos sin editar	# de ejemplos Edit1RS	# de ejemplos Edit2RS
diabetes	768	176	499
heart_stattlog	270	260	260
iris	150	117	147
glass	214	62	141
credit	690	623	655

Tabla 2. Resultados de evaluar métodos de reducción de ejemplos

La tabla 3 muestra los resultados de aplicar los clasificadores Naives Bayes(NV), Multilayer Percerceptron (MLP) y una versión mejorada de C4.5 (J48), implementadas todas en WEKA. Se utilizaron como conjuntos de prueba (TEST) la base de casos original(BC) con vista a tener en cuenta el valor máximo alcanzable, el 34% de la misma (34%BC), la validación cruzada con 10 particiones(CV) y la base de casos editada por Edit2RS(BCedit) y como medida para evaluar el índice de Kappa. Considerando las columnas cuarta, quinta y sexta, se pueden apreciar resultados mejores con la base editada del índice de Kappa en la mayoría de los casos; los cuales aparecen en negrita. Cuando se hace el análisis similar de la eficacia usando los métodos Edit1RS y Edit3RS, los resultados muestran que la eficacia lograda con Edit1RS es menor y con Edit3RS mayor respecto a lo que alcanza Edit2RS.

	TEST	BC	34%BC	CV	BCedit
BC	clasificador	Kappa	Kappa	Kappa	Kappa
diabetes	NB	0.416	0.45	0.402	0.40
	MLP	0.565	0.425	0.379	**0.55**
	J48	0.481	0.47	0.392	**0.49**
heart_stattlog	NB	0.707	0.63	0.663	**0.707**
	MLP	0.962	0.48	0.501	**0.947**
	J48	0.675	0.521	0.523	**0.675**
iris	NB	0.96	0.94	0.93	**0.96**
	MLP	0.97	0.94	0.93	**0.96**

	J48	0.96	0.94	0.96	**0.96**
glass	NB	0.552	0.49	0.494	**0.502**
	MLP	0.702	0.49	0.516	**0.608**
	J48	0.601	0.56	0.469	0.548
credit	NB	0.721	0.699	0.715	**0.727**
	MLP	0.924	0.802	0.689	**0.888**
	J48	0.762	0.691	0.702	**0.803**

Tabla 2: Resultados del índice de Kappa para diversos clasificadores

5. Conclusiones

Se implementó una biblioteca en Python que facilita el preprocesamiento de los datos utilizando métodos basados en conjuntos aproximados; la cual incluye un nuevo método de edicion. Se demuestra la eficacia de los métodos de reducción de datos incluidos en la biblioteca a partir de los experimentos desarrollados, demostrándose la posibilidad de utilizar esta para el análisis inteligente de datos. Como trabajo futuro se prevé la vectorización de código para mejorar la eficiencia en la ejecución de los métodos implementados.

6. Referencias bibliográficas

1. Ahn, B. S. (2000). The integrated methodology of rough set theory and artificial neural networks for business failure predictions

2.Arco, L., R. Bello, et al. (2006). On clustering validity measures and the Rough Set Theory. 5th Mexican International Conference on Artificial Intelligence, IEEE Computer Society Press

3.Bache, K., & Lichman, M. (2013). UCI machine learning repository.

4.Bello, R., A. Nowe, et al. (2006). "Two Step Ant Colony System to Solve the Feature Selection Problem" Lectures Notes on Computer Sciences. Springer-Verlag: 588- 596.

5.Bello, R., García, M., & Pérez, J. N. (2012). Teoría de los conjuntos aproximados: conceptos y métodos computacionales. Universidad Distrital Francisco José de Caldas. Bogotá, Colombia.

6.Caballero, Y. (2005). Uso de los Conjuntos Aproximados para el tratamiento de los datos. Santa Clara, Cuba, Universidad Central de Las Villas.

7.Caballero, Y., R. Bello, et al. (2006)b. Improving the k-NN method: Rough Set in edit training set. The First IFIP International Conference on Artificial Intelligence in Theory

and Practice, Springer Boston.

8. Dorigo, M., Birattari, M., Stützle, T. (2010). Ant colony optimization, in Encyclopedia of Machine Learning, Springer: New York City. p. 36-39.
9. García, M., Y. Villuendas, et al. (2005). Métodos de selección y construcción de objetos para el mejoramiento de un clasificador supervisado: estado del arte
10. García, S., Luengo, J., & Herrera, F. (2015). Data preprocessing in data mining. Springer.
11. Goldberg, D.E. (1989). Genetic algorithms in optimization, search and machine learning, Boston, USA: Addison-Wesley Longman Publishing Co., Inc.
12. Herrera, F. (2014). Big data: Procesando los datos en la sociedad digital. Revista Española de Física, 28(4), 40–44.
13. Kennedy, J. (2011). Particle swarm optimization, in Encyclopedia of Machine Learning, Springer: Boston, MA. p. 760-766.
14. Komorowski, J. and Z. Pawlak (1999). "Rough Sets: A tutorial." Rough Fuzzy Hybridization: A new trend in decision-making. Springe: 3-98.
15. McKinney, W. (2012). Python for data analysis: Data wrangling with Pandas, NumPy, and IPython. " O'Reilly Media, Inc."
16. Mitchell, T. (1997) Machine Learning, Redmond, WA, Ithaca, NY: McGraw-Hill Science/Engineering/Math. 432.
17. Pawlak, Z., S.K.M. Wong, W. Ziarko. (1988). Rough sets: probabilistic versus deterministic approach, International Journal of Man–Machine Studies 29: 81–95.
18. Russell, S. J., & Norvig, P. (2016). Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited,.
19. Skowron, A. (1999). New directions in Rough Sets, Data Mining, and Granular Soft Computing. 7th International Workshop (RSFDGRC'99), Yamaguchi, Japan, Lecture Notes in Artificial Intelligence 1711.
20. Wang, X., Yang, J., Jensen, R., & Liu, X. (2006). Rough set feature selection and rule induction for prediction of malignancy degree in brain glioma. Computer Methods and Programs in Biomedicine, 83, 147–156.
21. Wang, X., Yang, J., Teng, X., Xia, W., & Jensen, R. (2007). Feature selection based on

rough sets and particle swarm optimization. Pattern Recognition Letters, 28, 459–471.

22. WEKA https://www.cs.waikato.ac.nz/ml/weka/index.html [Consultado 8-3-2019]
23. Wilcoxon, F. (1945). Individual comparisons by ranking methods. Biometrics Bulletin, 1(6), 80–83.
24. Witten, I, Frank, E., Hall M., Pal, C., Data Mining: Practical Machine Learning Tools and Techniques, Fourth Edition, 2017. ELSEVIER, ISBN:978-0-12-804291-5
25. Zhong, N., J. Dong, et al. (2001). "Using Rough sets with heuristics for feature selection." Journal of Intelligent Information Systems 16: 199-214.