Segmentación de mercado explicable sobre datos de alta dimensión

Carlos Eiras-Franco
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
carlos.eiras.franco@udc.es

Bertha Guijarro-Berdiñas
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
cibertha@udc.es

Amparo Alonso-Betanzos
Grupo LIDIA. CITIC.
Universidade da Coruña
A Coruña, España
ciamparo@udc.es

Antonio Bahamonde
Universidad de Oviedo
Gijón, España
abahamonde@uniovi.es

***Resumen*—Obtener información relevante de la gran cantidad de datos que se generan en las interacciones de un mercado o, en general, de un conjunto de datos diádicos, es un amplio problema que despierta gran interés académico y en la industria. Por otra parte, la interpretabilidad de los algoritmos de aprendizaje máquina está adquiriendo relevancia hasta el punto de ser requerida legalmente, lo que dispara la necesidad de contar con algoritmos con estas características. En este trabajo proponemos una medida de calidad que tiene en cuenta el nivel de interpretabilidad de un resultado. Presentamos, además, un algoritmo de agrupamiento sobre datos diádicos que ofrece resultados con el nivel de interpretabilidad que desee el usuario y que es capaz de manejar grandes volúmenes de datos. Detallamos experimentos que avalan tanto la precisión de los resultados obtenidos, comparable a métodos tradicionales, como su escalabilidad.**

***Index Terms*—segmentación de mercado, interpretabilidad, explicabilidad, escalabilidad, aprendizaje automático, big data**

I. Introducción

Los datos obtenidos al monitorizar el funcionamiento de un mercado son, en su mayoría, diádicos, es decir, representan la relación entre dos entidades, ya sean estas usuarios y productos, compradores y vendedores u otra pareja de agentes. Más generalmente, los datos diádicos [1] recogen información sobre la interacción de dos entidades de cualquier ámbito y son prevalentes en problemas tan comunes como los sistemas recomendadores [2], lingüística computacional, recuperación de información y aprendizaje de preferencias [3], además de estar presentes en campos más específicos como la corrección automática de exámenes [4].

Un problema tradicional a tratar sobre datos de este tipo consiste en obtener agrupaciones de entidades con un comportamiento similar, dando lugar a un modelado de mayor nivel del entorno estudiado. De esta manera se podría, por ejemplo a partir de los datos de un recomendador de libros, buscar conjuntos de libros que atraen a usuarios similares o conjuntos de usuarios que tienen gustos parecidos respecto a la lectura.

Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y Competitividad (proyectos de investigación TIN 2015-65069-C2, tanto 1-R como 2-R y Red Española de Big Data y Análisis de datos escalable, TIN2016-82013-REDT), por la Xunta de Galicia (GRC2014/035 y ED431G/01) y por Fondos de Desarrollo Regional de la Unión Europea.

El estudio de los mercados con el objetivo de identificar subconjuntos de actores que se comportan como mercados más pequeños y homogéneos es la meta de la segmentación de mercados [5]. La información obtenida a partir de estas agrupaciones que se comportan de manera homogénea permite a las empresas hacer campañas de marketing dirigido a medida de cada grupo, aumentando así su efectividad y disminuyendo costes [6]. Esta información es, por tanto, una ventaja codiciada por las empresas, pero resulta complicada de obtener en la práctica aún cuando se dispone de abundantes datos.

Por otra parte, que los resultados obtenidos de la segmentación de mercados sean comprensibles por un gestor humano es un factor esencial que está atrayendo mucha atención recientemente. El concepto de interpretabilidad se refiere a esta característica, aunque no se trata de un concepto monolítico sino que, de acuerdo a [7], refleja distintas ideas. Idealmente, el modelo obtenido debe:

1. permitir a un supervisor interpretar sus resultados de manera que pueda verificar que los objetivos del modelo están alineados con los deseados (p.e. un sistema de evaluación de crédito no debe tener sesgos raciales o de género),
2. motivar sus predicciones de manera que un supervisor pueda hacer hipótesis de causalidad que luego verificará, descartando meras correlaciones fortuitas o dependientes de un tercer factor,
3. explicar sus salidas de manera que se pueda corroborar cómo de generalizables son. Por ejemplo, en el caso descrito en [8], un sistema de predicción de mortalidad por neumonía asignaba, erróneamente, menor riesgo a los pacientes que también padecían asma. Esto se debía a que estos pacientes habían recibido un tratamiento más agresivo por lo que, en el historial de datos analizados, sus porcentajes de recuperación solían mejorar respecto a otros pacientes; en consecuencia, el modelo automático asignaba un menor riesgo resultado de una mala generalización que posiblemente causaría el resultado contrario al esperado. Al utilizar el modelo derivado para asignar tratamientos, estos pacientes recibirían un tratamiento más leve, puesto que al haber obtenido históricamente mejores resultados, su riesgo estimado de mortalidad

sería menor,
4. ser informativo, es decir, ofrecer al supervisor información novedosa sobre las variables bajo estudio.

Estas características en ocasiones son incluso requeridas por ley, como en el caso del "derecho a la explicación" recogido en el nuevo Reglamento Europeo de Protección de Datos de la Unión Europea[1], de manera que resultan no solo deseables en todo caso sino que también obligatorias en muchos otros.

Los algoritmos de aprendizaje automático ofrecen interpretabilidad generalmente de dos maneras distintas: (1) mediante la transparencia del modelo y/o del algoritmo, que permite al supervisor seguir la "lógica" usada para hacer las predicciones, como ocurre por ejemplo con las reglas de producción, o (2) interpretabilidad *post-hoc*, consistente en justificar las salidas bien ofreciendo casos similares o visualizaciones u otros métodos que ayuden a identificar las características de la entrada que llevan a una predicción. Esta segunda opción es la más común pero, aunque ofrece una explicación de cada caso individual, no es informativa ya que no ofrece al supervisor información general del entorno modelado.

Un problema adicional es que, si bien, el estudio de los mercados se ve potenciado por la inmensa cantidad de datos que se generan y recogen —tanto en entornos online como, en menor medida, en tiendas físicas— y la valiosa información que estos datos encierran, a su vez, su inmenso volumen dificulta la extracción de esta información. Por tanto, se requiere la utilización de algoritmos escalables que puedan procesarlos.

En este trabajo proponemos un nuevo algoritmo de agrupamiento de datos diádicos que se puede utilizar para realizar segmentación de mercado. Este algoritmo busca obtener datos fácilmente interpretables e informativos para un supervisor humano, mejorando así su proceso de toma de decisiones. Además, su implementación en la plataforma escalable Apache Spark [9] permite procesar grandes cantidades de datos para obtener información decisiva en un tiempo práctico. En la Sección II detallamos los trabajos previos de otros autores en este campo; en la Sección III se definen los principales conceptos manejados por el sistema propuesto y en la Sección IV presentamos el nuevo algoritmo. En la Sección V mostramos y comparamos los resultados de aplicar nuestro algoritmo a un conjunto de datos de lectores de noticias. Por último, la Sección VI resume las conclusiones obtenidas y el trabajo futuro que proyectamos realizar.

II. Trabajo relacionado

Las técnicas de segmentación de mercado se pueden dividir atendiendo al origen de los datos que utilizan [10]:

1. *A priori*. Utilizan información de los compradores o de los productos que está disponible antes de su interacción en el mercado. Adolecen de no prestar atención al comportamiento de los actores en el mercado y ofrecer, por tanto, información limitada.
2. *Post-hoc*. Se basan en el estudio de los datos que se generan durante la actividad en el mercado. Es un enfoque más explorado para el que se han utilizado algoritmos de agrupamiento [11], árboles de clasificación y regresión [12], mapas autoorganizativos [13], algoritmos de reducción de la dimensionalidad [14] o algoritmos de co-agrupamiento [15].

Desde el punto de vista de la interpretabilidad, los algoritmos mencionados muestran características dispares. Mientras los resultados de algoritmos evolutivos y de co-agrupamiento no son apropiados si la interpretabilidad es un objetivo, los mapas autoorganizativos, y los algoritmos de reducción de la dimensionalidad y de agrupamiento ofrecen explicaciones *post-hoc* al supervisor. No obstante su capacidad de relacionar de manera sencilla las variables de entrada que describen a cada grupo es limitada, lo que reduce su utilidad para motivar sus predicciones y explicar sus salidas. Los árboles de clasificación y regresión, por otra parte, sí relacionan de manera clara las variables de entrada con las predicciones efectuadas y han sido utilizados con este fin sobre datos no diádicos [16], aunque su tamaño debe ser controlado para ofrecer explicaciones verificables por un supervisor.

III. Definiciones

Los datos diádicos X describen una interacción entre dos entidades $\mathcal{U}$ e $\mathcal{I}$. Cada dato $x \in \mathcal{X}$ tiene la forma (u, i, v) donde $u \in \mathcal{U}$ y $i \in \mathcal{I}$ son los elementos de cada entidad que se relacionan y v es un valor que informa respecto a esa interacción. Para cualesquiera u, i existirá un dato x que describirá su relación (ya sea observado o predicho), por lo cual se puede representar $\mathcal{X}$ como una función $f : (\mathcal{U}, \mathcal{I}) \rightarrow \{-1, +1\}$ que se denomina función de utilidad. Nótese que aunque v puede tomar cualquier valor, para este trabajo hacemos la simplificación de transformarlo a -1 o 1. Obtener una predicción del valor de esta función de utilidad es un problema muy habitual que se intenta resolver sobre estos datos y que ha sido repetidamente resuelto en la literatura mediante métodos como la factorización matricial [2]. Por otra parte, definiremos una agrupación $Cl(\mathcal{U})$ sobre un $\mathcal{U}$ como un conjunto de m grupos disjuntos que abarcan todos los elementos de $\mathcal{U}$. En fórmula:

$$Cl(\mathcal{U}) = \{Clu_1, \ldots, Clu_m\}. \tag{1}$$

Se puede utilizar la homogeneidad de la función de utilidad dentro de cada grupo Clu_k para establecer la idoneidad de la agrupación [17]. Definiremos con este objetivo la proporción p de elementos positivos en un grupo se como

$$p_{kj} = Pr(+1|Clu_k, i_j) = \frac{|\{u \in Clu_k : f(u, i_j) = +1\}|}{|Clu_k|} \tag{2}$$

donde i_j representa el elemento j-ésimo de $\mathcal{I}$.

Diremos que un grupo Clu_k es *coherente* cuando haya gran consenso entre sus valores de f, es decir, que p se acerque a 0 o a 1. Podemos medir esta coherencia, como anunciamos, utilizando la entropía de p.

$$H(p) = -p \log_2(p) - (1 - p) \log_2(1 - p). \tag{3}$$

[1] https://ec.europa.eu/info/law/law-topic/data-protection_en

$H(p_{kj})$ nos permite medir la coherencia de un único grupo Clu_k respecto al elemento i_j. Para extender esta medida a todo el agrupamiento $Cl(\mathcal{U})$ debemos tener en cuenta todos sus grupos y todos los elementos de $\mathcal{I}$. Para conseguir esto, utilizaremos la media ponderada. Formalmente, definimos la *entropía ponderada (EP)* del agrupamiento $Cl(\mathcal{U})$ como

$$EP(Cl(\mathcal{U})) = \sum_{k,j} \frac{|Clu_k|}{|\mathcal{U}||\mathcal{I}|} H(p_{kj}). \quad (4)$$

Con las características de interpretabilidad descritas en la Sección I en mente, en este trabajo nos enfocaremos en la capacidad de los algoritmos para motivar sus predicciones de manera sencilla utilizando las variables de entrada, con el objetivo de que un supervisor pueda formular y verificar hipótesis, así como también de resultar informativo respecto a la situación general del mercado en un instante dado. Debemos, por tanto, incluir en nuestra medida de idoneidad la complejidad de la explicación que requiere cada grupo, que mediremos con el número de variables que lo describen. Definimos, en consecuencia, la *calidad* de un agrupamiento como

$$calidad(Cl(\mathcal{U})) = -EP(Cl(\mathcal{U})) - \lambda \sum_{Clu_k \in Cl(\mathcal{U})} NV(Clu_k). \quad (5)$$

donde NV se refiere al número de variables necesarias para describir Clu_k y λ es un hiperparámetro que permite al supervisor balancear la entropía del agrupamiento con su explicabilidad. Se puede comprobar que existe un balance entre la complejidad de la explicación y la precisión del agrupamiento obtenido, lo cual es esperable intuitivamente: para obtener grupos uniformes generalmente será preciso formar gran número de ellos, lo que obligará a describirlos con más variables; ambas cosas reducen la explicabilidad del modelo. Por el contrario, para obtener un modelo más explicable será necesario obtener un menor número de grupos descritos con menos variables, que consecuentemente tendrán que ser menos uniformes. El hiperparámetro λ permite al supervisor decidir cómo manejar esta disyuntiva.

Cabe destacar que, aunque la *calidad* así descrita es un número negativo, esto no es relevante. El objetivo del algoritmo será maximizar su valor, acercándolo a 0. Esta medida de *calidad* se puede aplicar a agrupamientos construidos con cualquier algoritmo sobre una entidad $\mathcal{U}$ de un conjunto de datos diádicos.

IV. Algoritmo propuesto

En este trabajo proponemos un algoritmo de agrupamiento sobre datos diádicos (*post-hoc*) que obtenga grupos lo más homogéneos posibles y que, al mismo tiempo, se expliquen con el mínimo número posible de variables que describen a las entidades de la díada (*a priori*). De esta manera, en el escenario de la segmentación de mercado, el supervisor puede obtener información sobre cómo se dividen sus usuarios/productos y la relación entre estos grupos para así tomar decisiones informadas en consecuencia. Para ello se construirá un árbol de decisión que describirá una agrupación $Cl(\mathcal{U})$ en la que cada hoja del nodo representa un grupo descrito por las variables que conducen por el árbol hasta esa hoja.

El proceso global consiste en, a partir de la función de utilidad que relaciona a ambas entidades de la díada, realizar una exploración de los posibles árboles de decisión formados sobre las variables de entrada buscando maximizar la *calidad* de la agrupación resultante definida según la Ecuación 5.

Para poder explorar de manera eficiente el espacio de soluciones es necesario realizar ciertas simplificaciones. En primer lugar, para simplificar el cómputo de *calidad* de una agrupación y dado que el número de elementos de $\mathcal{I}$ puede hacer impracticable su cálculo, en lugar de realizar el cómputo de entropía respecto a cada elemento de $\mathcal{I}$, consideraremos una muestra aleatoria representativa de los elementos de $\mathcal{I}$. Una manera de hacer esto es haciendo un agrupamiento previo de $\mathcal{I}$ por procedimientos convencionales y tomando los centroides ci_j como representantes. Una vez calculados estos centroides, la proporción p descrita en la Ecuación 2 se aproximará como

$$p_{kj} = Pr(+1|Clu_k, ci_j) = \frac{|\{u \in Clu_k : f(u, ci_j) = +1\}|}{|Clu_k|} \quad (6)$$

y la Ecuación 4 deberá modificarse para tener en cuenta el tamaño del grupo Cli_j representado por ci_j

$$EP(Cl(\mathcal{U})) = \sum_{k,j} \frac{|Clu_k||Cli_j|}{|\mathcal{U}||\mathcal{I}|} H(p_{kj}). \quad (7)$$

Por otra parte, dado que el problema de explorar todos los posibles árboles de decisión anteriormente descritos es generalmente inabarcable, se hace necesario establecer una estrategia de búsqueda. En primer lugar, para evitar que el árbol de decisión pueda bifurcarse en cualquier posible valor de cada variable de entrada, debe reducirse el número de estos puntos de bifurcación. Se establecerá, por tanto, un máximo de puntos de bifurcación por variable y estos se determinarán mediante discretización en el caso de las variables continuas.

A continuación se construirá el árbol de decisión realizando una búsqueda voraz. Para ello en cada nodo se tomará el candidato más prometedor, que se determinará utilizando una heurística. Por tanto, siendo v_i el i-ésimo punto de corte de la variable V, calcularemos la heurística del agrupamiento $L = \{u \in \mathcal{U} : V(u) < v_i\}, R = \{u \in \mathcal{U} : V(u) > v_i\}$ asociado a v_i con la siguiente fórmula, derivada de manera empírica:

$$\eta(L, R) = -EP(L) - EP(R) - (|L| - |R|)^2 \quad (8)$$

Realizando una búsqueda voraz en la que para cada nodo se tome el punto de bifurcación candidato con mejor heurística y repitiendo el proceso recursivamente para los grupos L y R obtenidos hasta alcanzar un nivel L_{MAX} preestablecido por el usuario, se construirá un único árbol de decisión de L_{MAX} niveles. Para expandir la cantidad de espacio de búsqueda explorado y mejorar así las posibilidades de obtener una buena solución, el algoritmo propuesto explora en cada paso no solo el candidato con mejor valor de la heurística, sino los N mejores, dando lugar a N árboles posibles. Cada uno de

esos árboles, a su vez, darán lugar a otros $2 * N$ árboles al explorar el nivel siguiente, con lo cual el número de árboles explorados aumenta exponencialmente con el L_{MAX}. Deberá, en consecuencia, elegirse un valor bajo tanto para N como para L_{MAX}. El árbol de decisión obtenido será un árbol binario de L_{MAX} niveles y, por tanto, delimitará $2^{L_{MAX}}$ grupos.

Este proceso aparece descrito en el Algoritmo 1.

Datos: $\mathcal{U}, f, L_{MAX}, N$
Resultado: Árbol de decisión que determina el agrupamiento.
funcion CONSTRUYEARBOL($\mathcal{U}, nivel, puntosC, f, L_{MAX}, N$)
  si $nivel >$*NIVEL_MAX* **entonces**
    devolver $\emptyset$
  fin
  $candidatos \leftarrow$ lista ordenada con capacidad N;
  para $(variable, valor) \in puntosC$ **hacer**
1   $izq \leftarrow \{u \in \mathcal{U} : u[variable] < valor\}$;
    $der \leftarrow \{u \in \mathcal{U} : u[variable] > valor\}$;
    si HEURISTICA(izq, der) $> candidatos.minimo$ **entonces**
2     $candidatos.añadir((variable, valor))$;
    fin
  fin
  $mejor \leftarrow \emptyset$;
  para $(variable, valor) \in candidatos$ **hacer**
3   $izq \leftarrow \{u \in \mathcal{U} : u[variable] < valor\}$;
4   $der \leftarrow \{u \in \mathcal{U} : u[variable] > valor\}$;
5   $arbolIzq \leftarrow$ CONSTRUYEARBOL($izq, nivel + 1, puntosCorte$);
6   $arbolDer \leftarrow$ CONSTRUYEARBOL($der, nivel + 1, puntosCorte$);
7   $nuevo \leftarrow (variable, valor, arbolIzq, arbolDer)$
    si EP($nuevo, f$) $>$ EP($mejor, f$) **entonces**
8     $mejor = nuevo$;
    fin
  fin
  devolver $mejor$;
fin
$puntosC \leftarrow$ lista de puntos de corte en todas las variables;
devolver
  CONSTRUYEARBOL($\mathcal{U}, 0, puntosC, f, L_{MAX}, N$);

Algoritmo 1: Algoritmo de construcción del agrupamiento.

Por último, es posible que tras completar la búsqueda un subconjunto de este árbol alcance mayor calidad que el árbol completo. Por tanto, se intentará mejorar la calidad de la agrupación eliminando grupos mediante un proceso de poda. Para ello se recorrerán los nodos desde $L_{MAX} - 1$ hasta la raiz y se eliminarán aquellas bifurcaciones cuyo efecto en la *calidad* general no supere un $umbral$ indicado por el usuario, tal como aparece detallado en el Algoritmo 2.

El algoritmo descrito consta de cómputos que se pueden

Datos: $árbol$, $umbral$, λ
Resultado: Árbol podado.
funcion PODA($árbol$,$umbral$,λ,$nivel$)
1   **si** ESHOJA($árbol$) **entonces**
2   **devolver** $árbol$;
    fin
3   $izq \leftarrow$PODA($árbol.ramaIzq, umbral, nivel + 1$);
4   $der \leftarrow$PODA($árbol.ramaDer, umbral, nivel + 1$);
5   $entropíaDividido \leftarrow \frac{izq.entropía * |izq| + der.entropía * |der|}{|árbol|}$;
6   $\Delta E = entropíaDividido - árbol.entropía$;
7   $\Delta NV = izq.numVars + der.numVars - árbol.numVars$;
    si $-\Delta E - \lambda \Delta NV < umbral$ **entonces**
8     $árbol.ramaIzq \leftarrow \emptyset$;
9     $árbol.ramaDer \leftarrow \emptyset$;
    devolver $árbol$;
fin
devolver PODA($árbol$,$umbral$,λ,0);

Algoritmo 2: Algoritmo de podado del árbol.

realizar en paralelo dado que son independientes. Para aprovechar esta característica se ha implementado en Apache Spark. Valiéndose de la computación distribuida que facilita Spark se aumenta la escalabilidad del algoritmo y se posibilita así el análisis de grandes conjuntos de datos en tiempos razonables.

V. Experimentación

Para comprobar la validez de este algoritmo hemos realizado dos experimentos. En primer lugar hemos calculado un agrupamiento con una entropía ponderada comparable a la obtenida en [17] para una agrupación de 100 grupos obtenidos mediante K-Means. El segundo experimento consiste en comprobar el efecto sobre el tiempo de ejecución de la adición de más nodos de cómputo para el cálculo distribuido, para establecer qué nivel de escalabilidad tiene la implementación en Apache Spark del algoritmo propuesto.

Para ello hemos utilizado un conjunto de datos de la empresa Outbrain, liberado en el contexto de una competición auspiciada por el popular sitio web Kaggle.com. Outbrain es la empresa líder en descubrimiento de contenidos, mediante la sugerencia a lectores de noticias en las que pueden estar interesados. El conjunto de datos[2] recoge las visitas de un grupo de usuarios a las páginas web de multitud de publicaciones periodísticas durante 14 días. De cada visita se registra, además de metadatos como la fecha, la localización o el medio empleado para conectarse, el documento visitado, del cual se conocen ciertos aspectos del contenido, y el resultado de su interacción con los anuncios mostrados en la página que, a su vez, enlazan a otros documentos. Aunque la competición original buscaba predecir qué anuncios tendrían más éxito en una situación dada, el problema que hemos decidido abordar

[2]El conjunto de datos está disponible para descargar en el sitio https://www.kaggle.com/c/outbrain-click-prediction

Cuadro I
Descripción del conjunto de datos

Conjunto	Atributos	Muestras
DIA1	647	1,289,506

Cuadro II
Tiempo de cómputo del un árbol de 3 niveles. Escalabilidad

	Tiempo (H:M:S)		
# cores	12	2x12	4x12
DIA1	3:19:46	2:35:33	1:33:45

es la agrupación de los pares usuario-documento en función de su comportamiento. Para ello hemos determinado las visitas correlativas de un usuario y hemos registrado la preferencia del usuario por un documento ofrecido frente a otros, obteniendo tuplas de preferencias sobre las que aprender la función de utilidad mediante factorización matricial [17]. Dado que los documentos visitados se refieren a temas de actualidad, las visitas registradas varían de temática con el tiempo y, por ello, tiene sentido agrupar los datos en subconjuntos más pequeños que abarquen menos tiempo. En particular, para este trabajo hemos utilizado los datos recogidos durante el primer día, descritos en el Cuadro I. De acuerdo a lo descrito en la Sección IV, hemos agrupado las noticias (conjunto $\mathcal{I}$) en 100 grupos utilizando K-Means sobre los valores de la función de utilidad para cada noticia. El número de candidatos (N) que hemos explorado para cada nodo es 5.

V-A. *Resultados*

Para realizar el primer experimento es necesario determinar un valor de λ y decidir qué nivel de profundidad adquirirá el árbol calculado. En este caso optamos por un árbol de 5 niveles. Al fijar este valor, la agrupación obtenida constará de 32 grupos descritos con 5 variables cada uno, con lo cual su valor NV, necesario para computar la calidad según la Ecuación 5 será de $32 * 5 = 160$. Tal como describimos en la Sección III, el valor λ debe balancear el valor de la entropía ponderada (que oscila entre 0 y 1) con el valor $NV(Cl(\mathcal{U}))$, que en este caso oscilará entre 1 y 160. El valor de λ dependerá de la importancia que dé el supervisor a la interpretabilidad de la solución. En este experimento hemos tomado $\lambda = 0{,}001$.

Tras aplicar el algoritmo descrito al conjunto *DIA1* con los parámetros mencionados. La entropía ponderada de la agrupación es de 0.244 y, en consecuencia, su calidad es de -0.375. Tal como se indica en la Sección IV, este valor de calidad se puede mejorar con el proceso de poda descrito en el Algoritmo 2. Para ello es necesario establecer un umbral de mejora de la calidad que indique qué nodos deberán permanecer sin dividir. Habiendo calculado el árbol previamente, este proceso es lo suficientemente poco costoso como para que se pueda realizar la poda con cientos de umbrales distintos y representar en una gráfica los valores obtenidos, tal como se puede apreciar en la Figura 1. Tomando el valor de umbral (0.003) que obtiene mayor calidad, se obtuvo un árbol que describe 18 grupos, 2 de los cuales se describen usando 3 variables, 8 necesitan 4 variables y los 8 restantes se describen con 5 de las 647 variables. Su valor NV es, por tanto, 78, su entropía ponderada es de 0.2529 y su calidad es -0.3308. El árbol obtenido puede observarse en la Figura 2. Comparativamente, el agrupamiento obtenido en [17] ofrece una menor entropía ponderada (0.21), pero se compone de 100 grupos que precisan las 647 variables para describir sus centroides, lo que se traduce en un valor de calidad muy bajo (-64,91). Si utilizásemos un árbol de clasificación para predecir a qué grupo pertenece cada muestra y de él extrajésemos nuevas descripciones para los grupos, aún en el caso óptimo estas nunca serían menos de 8 variables de media (en consecuencia $NV = 800$), dado que se precisa un árbol binario de más de 8 niveles para discernir entre 100 elementos. Por tanto, la calidad de una agrupación compuesta de 100 grupos nunca será mayor de -1.01. Esto nos permite apreciar la utilidad del algoritmo propuesto para encontrar una agrupación bastante homogénea y con explicabilidad alta.

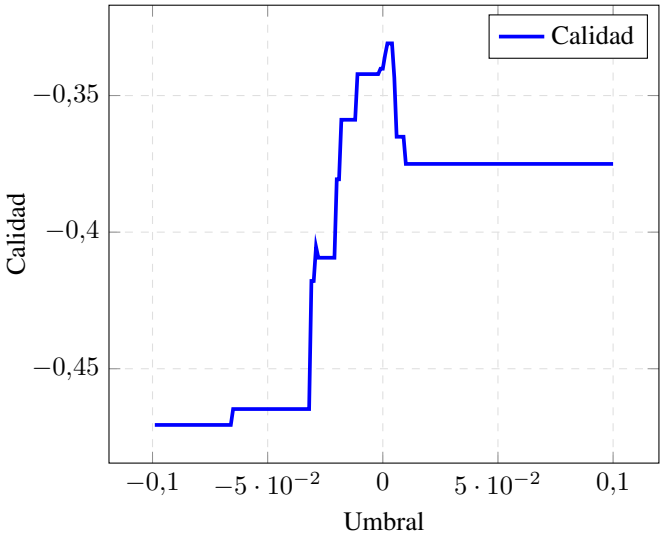

Figura 1. Calidad del agrupamiento vs umbral de bifurcación para el conjunto DIA1 con $\lambda = 0{,}001$.

En segundo lugar, para comprobar la escalabilidad del algoritmo, realizamos el mismo cómputo varias veces variando el número de núcleos de computación involucrados en el cálculo para observar su efecto en el tiempo invertido. El cómputo que realizamos se corresponde a la búsqueda del árbol óptimo de tres niveles. Los tiempos de ejecución aparecen reflejados en el Cuadro II.

Como se puede apreciar, la independencia de los cálculos realizados, que facilita su cómputo en paralelo, y la facilidad de distribución del cálculo que proporciona Apache Spark provocan que la escalabilidad sea buena, correspondiéndose casi linealmente el incremento de nodos de computación con el descenso del tiempo de ejecución.

VI. Conclusiones

En este trabajo hemos reflexionado sobre las características que debe cumplir un algoritmo para ser considerado fácilmente

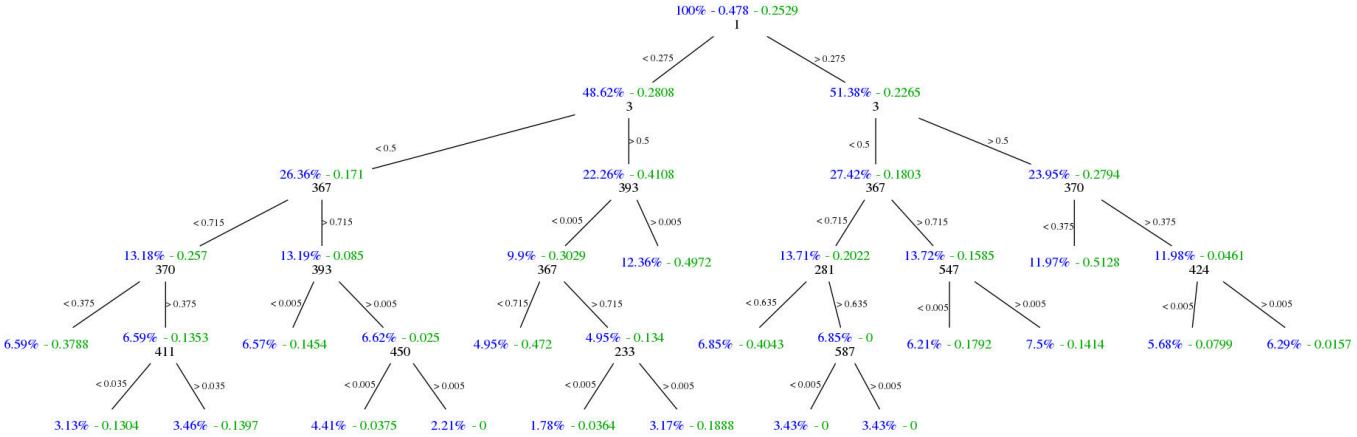

Figura 2. Árbol de decisión que determina el agrupamiento de los usuarios del conjunto de datos DIA1. En cada nodo se listan, respectivamente, la proporción de elementos que representa respecto al total (indicada en color azul), la entropía ponderada de ese nodo (verde) y la variable utilizada para la siguiente bifurcación (negro). Los valores utilizados para las bifurcaciones se indican al lado de las aristas. En la raíz se indica también en azul la entropía ponderada del conjunto entero.

interpretable y, adaptando una medida previamente utilizada en la literatura, hemos desarrollado un algoritmo de agrupamiento sobre datos diádicos que las cumple. El algoritmo propuesto permite, cuando es aplicado a datos referentes a interacciones en un mercado, realizar una segmentación de mercado a gran escala que ofrece a un supervisor información sobre qué está ocurriendo en ese medio en términos de las variables *a priori* que maneja. Asimismo, hemos realizado una implementación del citado algoritmo en la plataforma de computación distribuida Apache Spark que permite su aplicación a grandes volúmenes de datos y hemos realizado experimentos que verifican tanto la validez de este enfoque como la escalabilidad del mismo. Como trabajo futuro queremos adaptar este algoritmo para que pueda operar en situaciones en que el tiempo de cómputo es crucial. Para ello, queremos replantear la estrategia de búsqueda del árbol óptimo para que permita ser interrumpida en cualquier momento y ofrezca un resultado parcial relevante. Igualmente, creemos que la obtención de una heurística con mayor poder predictivo haría que la búsqueda fuese mucho más eficiente.

Agradecimientos

Los autores desean agradecer a la Fundación Pública Galega Centro Tecnolóxico de Supercomputación de Galicia (CESGA) el uso de sus recursos de computación.

Referencias

[1] Thomas Hofmann, Jan Puzicha, and Michael I Jordan. Learning from dyadic data. In *Advances in neural information processing systems*, pages 466–472, 1999.

[2] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[3] Oscar Luaces, Jorge Díez, Amparo Alonso-Betanzos, Alicia Troncoso, and Antonio Bahamonde. A factorization approach to evaluate open-response assignments in moocs using preference learning on peer assessments. *Knowledge-Based Systems*, 85:322–328, 2015.

[4] Oscar Luaces, Jorge Díez, Amparo Alonso-Betanzos, Alicia Troncoso, and Antonio Bahamonde. Content-based methods in peer assessment of open-response questions to grade students as authors and as graders. *Knowledge-Based Systems*, 117:79–87, 2017.

[5] Philip Kotler and Keith Kohn Cox. *Marketing management and strategy*. Prentice Hall, 1980.

[6] Philip Kotler. *Dirección de mercadotecnia: Análisis, planeación, implementación y control*. Magíster en Administración-Tiempo Parcial 29, ESAN, 2001.

[7] Zachary C Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.

[8] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730. ACM, 2015.

[9] Matei Zaharia, Reynold S Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J Franklin, et al. Apache spark: a unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, 2016.

[10] Jiapeng Liu, Xiuwu Liao, Wei Huang, and Xianzhao Liao. Market segmentation: A multiple criteria approach combining preference analysis and segmentation decision. *Omega*, 2018.

[11] Pavel Berkhin. A survey of clustering data mining techniques. In *Grouping multidimensional data*, pages 25–71. Springer, 2006.

[12] Bo Fan and Pengzhu Zhang. Spatially enabled customer segmentation using a data classification method with uncertain predicates. *Decision Support Systems*, 47(4):343–353, 2009.

[13] Melody Y Kiang, Michael Y Hu, and Dorothy M Fisher. An extended self-organizing map network for market segmentation—a telecommunication example. *Decision Support Systems*, 42(1):36–47, 2006.

[14] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[15] Hanhuai Shan and Arindam Banerjee. Bayesian co-clustering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 530–539. IEEE, 2008.

[16] Jayanta Basak and Raghu Krishnapuram. Interpretable hierarchical clustering by constructing an unsupervised decision tree. *IEEE transactions on knowledge and data engineering*, 17(1):121–132, 2005.

[17] Jorge Díez, Pablo Pérez, Oscar Luaces, and Antonio Bahamonde. Readers segmentation according to their preferences to click promoted links in digital publications. Technical report, Universidad de Oviedo, 2018.