Algoritmos basados en árboles de decisión para *partial* label ranking

Juan C. Alfaro
Escuela Superior de Ingeniería Informática
Universidad de Castilla-La Mancha
Albacete, 02071, España.
Email: JuanCarlos.Alfaro1@alu.uclm.es

Juan A. Aledo
Dpto. de Matemáticas
Univ. de Castilla-La Mancha
Albacete, 02071, España.
Email: JuanAngel.Aledo@uclm.es

José A. Gámez
Dpto. de Sistemas Informáticos
Univ. de Castilla-La Mancha
Albacete, 02071, España.
Email: Jose.Gamez@uclm.es

***Resumen*—Por *Label Ranking* (*LR*) se denomina a un problema de clasificación supervisada no estándar, en el sentido de que el objetivo no es predecir una etiqueta de la variable clase, si no un ranking completo de las posibles etiquetas. Además, las instancias del conjunto de entrenamiento también están etiquetadas con rankings, no con una única etiqueta. En este trabajo proponemos extender este problema de clasificación al *Partial Label Ranking* (*PLR*) en el que tanto los rankings asociados a las instancias de entrenamiento como a la predicción, puede contener etiquetas empatadas, es decir, para las que no hay preferencia entre ellas. Este escenario, que se da frecuentemente en el mundo real, evita tener que introducir desempates artificiales. Siguiendo uno de los trabajos seminales de LR, proponemos algoritmos de aprendizaje automático basados en los vecinos más cercanos y en árboles de clasificación para abordar el PLR. La experimentación realizada muestra la competencia de los algoritmos propuestos en la resolución del PLR.**

I. Introducción

El problema de la clasificación supervisada (o simplemente clasificación) es probablemente el más estudiado y aplicado en el ámbito del aprendizaje automático y ciencia de datos [1]. En un problema de clasificación, se dispone de un conjunto de variables predictoras $\{X_1, \ldots, X_m\}$ que pueden ser numéricas o discretas, y de una variable distinguida, la *clase*, C, que toma valores en un conjunto finito de etiquetas o categorías $dom(C) = \{c_1, \ldots, c_n\}$. Definido sobre estas variables se dispone de un conjunto de datos $\{(x_1^j, x_2^j, \ldots, x_m^j, c_k^j)\}_{j=1}^N$ donde N es el número de instancias y j indica el número de instancia. El objetivo de la clasificación automática es el aprendizaje a partir de los datos de un clasificador (o función de clasificación)

$$\mathcal{C} : dom(X_1) \times dom(X_2) \times \cdots \times dom(X_m) \longrightarrow C$$

que generalice bien a partir de los datos de entrada y pueda, por tanto, aplicarse exitosamente para clasificar datos nuevos.

En los últimos años han aparecido una serie de problemas aplicados, que si bien corresponden a tareas de clasificación supervisada, no encajan en la definición anterior, dando lugar a los conocidos problemas de clasificación *no estándar* [2]. Este grupo de tareas de clasificación se caracteriza por una débil supervisión en las instancias del conjunto de datos, la existencia de una estructura interna en el *objetivo* a predecir, o ambas cuestiones simultáneamente.

En este trabajo tomamos como base uno de estos problemas, de aparición bastante reciente, como es el de los clasificadores de tipo *Label Ranking (LR)* [3], cuyo objetivo es predecir un ranking total (todas las etiquetas se ordenan y el orden es estricto) entre las etiquetas de la variable clase a partir de los valores de las variables observadas. A diferencia de problemas relacionados como puede ser la clasificación ordinal, en la tarea de LR las instancias vienen etiquetadas con un ranking de las etiquetas de la variable clase, no con una única etiqueta. Esta información es explotada para construir el clasificador LR. A modo de ejemplo, supongamos que queremos recomendar a un futuro estudiante de nuestra universidad una lista ordenada de los estudios que pensamos le pueden ir bien en función de su historial de notas en Bachillerato, sus hobbies, etc. Por ejemplo, podríamos recomendar matemáticas $\prec$ informática $\prec$ biología $\prec$ medicina a un estudiante con excelentes habilidades matemáticas y cuyo hobby sea la programación. Basándose en la activa investigación en el campo de la agregación de rankings [4] y en los modelos clásicos de aprendizaje automático, en [3] se proponen dos algoritmos para aprender clasificadores LR, uno basado en los vecinos más cercanos (*IBLR*) y otro en árboles de clasificación/regresión (*LRT*) (ver Sección II).

En el ámbito de la agregación de rankings podemos encontrar problemas cuyo objetivo es descubrir ranking parciales (alias débiles) [5], [6], i.e., con empates. Estos rankings son más generales puesto que no fuerzan a tomar decisiones de desempate de forma poco natural y rigurosa. Continuando con nuestro ejemplo anterior, supongamos que dadas las variables predictoras para un estudiante determinado, basamos nuestra recomendación o predicción en la selección que en su día hicieron los dos estudiantes más parecidos a nuestro sujeto:

$est.1$: matemáticas $\prec$ informática $\prec$ biología $\prec$ medicina
$est.2$: matemáticas $\prec$ biología $\prec$ informática $\prec$ medicina

Es indudable que matemáticas debería ser la primera opción del ranking y medicina la última, pero no tenemos evidencia para ordenar entre informática y biología, por lo que lo más *natural* sería dejarlas empatadas, produciendo así un ranking parcial:

predicción: matemáticas $\prec$ informática $\sim$ biología $\prec$ medicina

Nuestro objetivo en este trabajo es extender el problema de la clasificación LR a este marco más general que denominamos *Partial Label Ranking* (*PLR*), y en el que si bien todas las etiquetas de la variable clase son ordenadas, el ranking no es estricto, si no que puede haber etiquetas empatadas. Tomando como base [3] diseñamos dos algoritmos de clasificación PLR basados en el método de los vecinos más cercanos (*IBPLR*) y en árboles de clasificación/regresión (*PLRT*) (ver Sección III).

El resto del artículo se organiza como sigue. En la Sección II repasamos los conceptos básicos para trabajar con rankings e introducimos los clasificadores principales para LR. En la Sección III describimos nuestra propuesta de algoritmos de clasificación para el PLR. En la Sección IV describimos el estudio empírico llevado a cabo para evaluar los métodos diseñados en este artículo. Finalmente, en la Sección V proporcionamos algunas conclusiones.

II. Preliminares: clasificadores Label Ranking

Comenzamos por introducir algunos conceptos relativos a la agregación de rankings, para después revisar brevemente los algoritmos de clasificación LR propuestos en [3]. Nos ceñimos al caso de rankings completos, que es el contemplado en este artículo, aunque en [3] se proporcionan herramientas para gestionar el caso de que las instancias puedan contener rankings incompletos. Tampoco se describen algoritmos más avanzados como pueden ser los basados en ensembles [7], [8].

A. Agregación de rankings

Dado un conjunto de items $\mathcal{I} = \{1, 2, \ldots, n\}$, un ranking π expresa un orden de preferencia sobre (algunos de) esos items. El conjunto de rankings completos y estrictos (permutaciones) de $\mathcal{I}$ es el grupo simétrico $\mathbb{S}_n$. Denotamos con $\widetilde{\mathbb{S}}_n$ el conjunto de rankings (completos o incompletos, totales o parciales) definidos sobre $\mathcal{I}$.

Dado un conjunto o muestra de N rankings $\{\pi_1, \pi_2, \ldots, \pi_N\}$, $\pi_i \in \widetilde{\mathbb{S}}_n$, el *problema de agregación de rankings* (*RAP*) consiste en obtener la permutación $\pi_0 \in \mathbb{S}_n$ que mejor representa a los rankings contenidos en la muestra, es decir, el que minimiza la suma de las distancias a los rankings contenidos en la muestra. π_0 se conoce como el *ranking de consenso*.

Dependiendo del tipo de rankings a considerar en la muestra, en la salida (consenso) y de la distancia utilizada, aparecen distintas variantes del RAP, siendo probablemente el más estudiado el caso de la agregación de ranking completos y totales (permutaciones), conocido como *Kemeny Ranking Problem* (*KRP*). Los problemas de agregación de rankings son NP-completos y por ello han proliferado los métodos heurísticos para abordarlos. En el caso de KRP, el más usado tanto por su sencillez como por su rapidez, es el algoritmo de conteo de Borda [9]. Además, en el caso de permutaciones, un conjunto de rankings puede modelarse probabilísticamente mediante la distribución de Mallows, la cual se especifica con dos parámetros: la moda o permutación de consenso π_0; y θ, un número real que mide la dispersión en torno a la moda, de forma que un mayor valor de θ indica una mayor concentración de los rankings alrededor de la moda. Cuando estos valores deben ser estimados un número importante de veces, el algoritmo de Borda es usado para estimar π_0 y θ se estima a partir de π_0 y de los datos mediante métodos iterativos.

Cuando la entrada y la salida al problema son rankings completos y parciales (o *bucket orders*), nos encontramos ante el problema conocido como *Optimal Bucket Order Problem* (*OBOP*) [5]. El objetivo del OBOP es minimizar la distancia entre la matriz de ordenación por pares obtenida a partir de un conjunto de preferencias de distintos individuos y la matriz de buckets asociada a un bucket order. Este es un problema menos estudiado que KRP y, por ejemplo, no conocemos una distribución de probabilidad para modelar un conjunto de bucket orders. Si hay, sin embargo, distintos métodos heurísticos para resolver la agregación y obtener el bucket order de consenso como son *Bucket Pivot Algorithm* (*BPA*) [5] y versiones mejoradas del mismo, p.e. BPA multipivote [6].

B. Clasificador IBLR: Instance-Based Label Ranking

Se trata de una adaptación del método de los vecinos más cercanos (kNN) al problema del label ranking. Así, para clasificar un nuevo caso $(\mathbf{x}, ?)$ se calcula la distancia (usando las variables predictoras) a las N instancias de la base de datos y se seleccionan las k más cercanas. La adaptación al caso del LR reside en la decisión de la respuesta a generar, puesto que ahora un voto por la mayoría no es oportuno. Así, si los rankings asociados a las k instancias recuperadas son $\Pi_k = \{\pi_1, \ldots, \pi_k\}$, la respuesta a devolver es el ranking de consenso π_0 obtenido al resolver el RAP asociado a Π_k. En [3] se usa el método de conteo de Borda [9] para obtener π_0.

Algunas mejoras propuestas en [3] son la estimación del valor de k usando validación cruzada o la posibilidad de pesar los rankings por la inversa de la distancia, usando entonces el método de conteo de Borda con pesos.

C. Clasificador LRT: Label Ranking Tree

Se trata de un algoritmo estándar de construcción de un árbol de clasificación/regresión a partir de datos, usando para ello un particionamiento recursivo del conjunto de datos. Tres son las principales cuestiones que hay que resolver para adaptar el algoritmo de aprendizaje al caso LR:

- Criterio de parada. Evidentemente el proceso de ramificación se detiene si los rankings de todas las instancias que han llegado al nodo actual son iguales. Además, en [3] el proceso también se detiene, dando lugar a un nodo hoja, cuando el número de instancias que han llegado al nodo actual es menor o igual que $2n$. Esta condición se propone como un criterio de pre-poda.
- Respuesta asignada a un nodo hoja. Se asigna a la hoja el ranking de consenso de todas las instancias que llegan a ella.
- Variable a seleccionar (y umbral) para los nodos de decisión. Al igual que en los árboles de clasificación con la entropía o en los de regresión con la varianza, se toma

la decisión que genera la partición de mínima dispersión para la variable objetivo. En [3] se usa la distribución de Mallows para medir esta dispersión. Si Π es el conjunto de rankings correspondientes a las instancias que llegan al nodo actual, se estima la distribución de Mallows $\mathcal{M}(\pi_0, \theta)$ a partir de ellos. Dada una variable predictora X tomando valores en $dom(X) = \{x_1, \ldots, x_r\}$, se divide el conjunto de instancias en función de estos valores. A partir de los conjuntos de rankings $\{\Pi_1, \ldots, \Pi_r\}$ correspondientes a la partición realizada, se estiman entonces r distribuciones de Mallows $\mathcal{M}^i(\pi_0^i, \theta^i)$, $i = 1, \ldots, r$, cada una correspondiente a un conjunto de la partición (una rama). Si θ es menor que

$$\sum_{i=1}^{r} \frac{|\Pi_i| \cdot \theta_i}{|\Pi|} \tag{1}$$

quiere decir que nuestra variable ha generado una buena partición, puesto que los rankings están más concentrados en torno a su consenso (θ_i), por lo que éste es una mejor predicción. Este criterio (similar al uso de la entropía) se comprueba para todas las variables, eligiéndose la que maximiza (1). En el caso de variables numéricas se sigue el procedimiento estándar de probar todos los umbrales y elegir el que maximiza (1) para la partición $X \leq t$ y $X > t$.

III. Aprendizaje de clasificadores para LR parcial

En esta sección describimos como adaptamos el clasificador IBLR [3] para tratar con PLR (IBPLR) y nuestras propuestas para este problema basándonos en LRT [3] (PLRT).

A. Clasificador IBPLR: Instance-Based Partial Label Ranking

Al igual que el algoritmo IBLR, IBPLR es una adaptación del método de los vecinos más cercanos (kNN) para trabajar con PLR. Así, si los bucket orders asociados a las k instancias recuperadas son $\Pi_k = \{\pi_1, \ldots, \pi_k\}$, la respuesta a devolver es el bucket order de consenso π_0 obtenido al resolver el OBOP asociado a Π_k. Para resolver dicho problema, se utiliza el algoritmo BPA multipivote, que obtiene mejores resultados que BPA [6].

B. Clasificador PLRT: Partial Label Ranking Trees

Como en LRT, PLRT debe solucionar tres cuestiones principales para adaptar el algoritmo de aprendizaje al caso PLR:

- Criterio de parada. Como es usual, el algoritmo de aprendizaje debe parar la llamada recursiva y crear un nodo hoja cuando los bucket orders de las instancias que llegan al nodo actual son iguales. Además, al igual que en [3], se impone una condición de pre-poda cuando el número de instancias que llegan al nodo actual es menor que q, siendo q un parámetro a especificar al algoritmo de aprendizaje.
- Respuesta asignada a un nodo hoja. Se asigna al nodo hoja el bucket order de consenso obtenido al solucionar el OBOP con el conjunto de bucket orders de las instancias que llegan al nodo actual, usando BPA multipivote [6].
- Variable a seleccionar (y umbral) para los nodos de decisión. Como se comentó anteriormente, el problema que se plantea es que, hasta donde conocemos, no existe una distribución de probabilidad para bucket orders. Por tanto, no podemos usar la misma metodología que en [3] y proponemos métodos alternativos. Así, si tomamos el conjunto de bucket orders asociados a las instancias que llegan al nodo actual Π y una variable predictora X, la incertidumbre asociada a la partición $\{\Pi_1, \ldots, \Pi_r\}$ generada por el atributo X viene dada por

$$\sum_{i=1}^{r} \frac{|\Pi_i| \cdot \phi_i}{|\Pi|} \tag{2}$$

donde ϕ_i es un parámetro a ser estimado para la partición Π_i. Por ejemplo, en el caso de LR, ϕ_i se toma como el parámetro de dispersión θ_i. En PLR, proponemos las siguientes alternativas:

  - Entropía. Para todos los pares de etiquetas (distintas) (c_j, c_k) con $c_j, c_k \in dom(C)$, se calcula la probabilidad de que la etiqueta c_j preceda, empate y suceda a c_k de acuerdo al conjunto de bucket orders en Π_i. Con esto, se obtiene una distribución de probabilidad para cada combinación de pares. El parámetro ϕ_i se calcula como la entropía media de las distribuciones de probabilidad obtenidas.

    Ejemplo III.1. Consideramos el conjunto de bucket orders $\Pi_i = \{2 \sim 1 \prec 3, \quad 1 \prec 3 \prec 2, \quad 1 \prec 2 \sim 3, \quad 2 \prec 1 \prec 3, \quad 3 \prec 1 \sim 2\}$. Así, el parámetro ϕ_i se estima como sigue usando la alternativa basada en entropía

$$\begin{aligned}\phi_i = media\{&H(\overbrace{(0.4, 0.4, 0.2)}^{\text{Par } (1,2)}), H(\overbrace{(0.8, 0, 0.2)}^{\text{Par } (1,3)}),\\ &H(\underbrace{(0.4, 0.2, 0.4)}_{\text{Par } (2,3)})\}\\ = media\{&1.522, 0.722, 1.522\} = 1.255\end{aligned}$$

  - Desacuerdo. Utilizando el bucket order de consenso obtenido a partir de Π_i, se calcula la frecuencia (relativa) de desacuerdos entre este y los incluidos en Π_i. Un desacuerdo existe si el orden de un par de items en los bucket orders a considerar es distinto, incluyendo el caso en que un par está empatado en un bucket order, pero no en el otro.

    Ejemplo III.2. Considerando el conjunto de bucket orders del ejemplo previo y el bucket order de consenso $\mathcal{B}_0 = 1 \sim 2 \sim 3$ (obtenido mediante BPA multipivote [6]), la estimación del parámetro ϕ_i, usando el método basado en desacuerdo es

$$\phi_i = \frac{\overbrace{3}^{\substack{\text{Desacuerdos}\\ \text{par } (1,2)}} + \overbrace{5}^{\substack{\text{Desacuerdos}\\ \text{par } (1,3)}} + \overbrace{4}^{\substack{\text{Desacuerdos}\\ \text{par } (1,2)}}}{\underbrace{15}_{\text{Normalización}}} = \frac{12}{15} = 0.8$$

Cabe destacar que, para optimizar el proceso de aprendizaje, se puede construir una matriz de tamaño $n \times n$ donde cada celda contiene un par ordenado con el número de veces que un item precede y empata con otro. De este modo, lo que se consigue es que para obtener la incertidumbre usando cada una de las alternativas propuestas, no es necesario recorrer todos los bucket orders de nuevo, sino que solo es necesario recuperar la información necesaria de dicha matriz (ver [10] para detalles).

IV. Evaluación experimental

En esta sección llevamos a cabo un estudio experimental de los métodos propuestos. A continuación describimos los conjuntos de datos utilizados, algoritmos involucrados, metodología y resultados.

A. Conjuntos de datos

Se han generado bases de datos semi-sintéticas siguiendo el esquema propuesto en [3]. Específicamente, tomamos 11 conjuntos de datos multiclase de [11] y los transformamos a bucket orders, considerándolos, por tanto, semisintéticos. La idea básica es resolver un problema de clasificación estándar, usando, en nuestro caso, Random Forest. Para cada instancia se rankean sus etiquetas de acuerdo a la distribución de probabilidad de la clase dada la instancia. Posteriormente se inicia un proceso de fusión de algunas etiquetas en buckets, siempre que la distancia entre ambas distribuciones de probabalidad (antes y después de colapsar) no superen un umbral establecido (0.05 en nuestro experimentos, ver [10] para detalles). En la Tabla I se muestran las principales características de los conjuntos de datos. Las columnas #etiquetas y #orders se corresponden con el número de etiquetas de la variable clase y el número de bucket orders distintos generados, respectivamente.

TABLA I
Descripción de los conjuntos de datos.

Cjto. de datos	#instancias	#atributos	#etiquetas	#orders
authorship	841	70	5	5
breast	106	9	6	22
ecoli	336	7	8	39
glass	214	9	6	23
iris	150	4	3	6
pendigits	10992	16	10	60
segment	2310	18	7	20
shuttle	43500	9	9	18
vehicle	846	18	4	13
vowel	528	10	11	23
wine	178	13	3	5

B. Algoritmos

Consideramos los siguientes algoritmos en el estudio:

- El algoritmo IBPLR pre-calculando (T-IBPLR) y sin pre-calcular (F-IBPLR) las distancias entre las instancias de los conjuntos de datos. Para identificar los vecinos más cercanos, utilizamos la distancia Euclídea. El número de vecinos más cercanos, k, se ajusta mediante una cinco validación cruzada (5-cv) interna sobre cada conjunto de entrenamiento correspondiente a cada uno de los folds de la validación cruzada externa, sin usar en ningún caso los registros de la partición test correspondiente para ajustar el valor de k. Dicho valor se ajusta como sigue. Primero, se va multiplicando (por 2, empezando por 5) el número de vecinos más cercanos mientras se mejore la precisión. Segundo, con los dos últimos valores probados, se hace un búsqueda binaria en ese rango. Se elige el número de vecinos más cercanos que conduzca a una mayor precisión.
- Las propuestas de PLRT, usando el método basado en entropía sin y con pre-poda (E-PLRT y PrE-PRLT, respectivamente), e idénticamente para la propuesta basada en desacuerdo (A-PLRT y PrA-PLRT). En todos los casos la pre-poda se aplica con $q = 5$.

C. Metodología

Adoptamos las siguientes decisiones de diseño:

- Todos los algoritmos han sido evaluados usando cinco repeticiones de una diez validación cruzada (5×10-cv) como en [3].
- Para evaluar la precisión de los algoritmos, utilizamos el coeficiente de correlación para rankings τ_X [12]. Dado los bucket orders $\mathcal{B}_a$ y $\mathcal{B}_b$, el coeficiente de correlación de rankings τ_X viene dado por

$$\tau_X(\mathcal{B}_a,\ \mathcal{B}_b) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \beta_{ij}^{r}\ \beta_{ij}^{p}}{n\ (n-1)} \qquad (3)$$

donde

$$\beta_{ij}^{k} = \begin{cases} 1, & \text{si } c_i \text{ precede o empata con } c_j \text{ en } \mathcal{B}_k \\ -1, & \text{si } c_i \text{ sucede a } c_j \text{ en } \mathcal{B}_k \\ 0, & \text{si } i = j \end{cases}$$

El coeficiente de correlación de rankings τ_X toma valores en el intervalo $[-1, 1]$, de tal manera que valores próximos a 1 indican una mejor correlación entre los bucket orders $\mathcal{B}_a$ y $\mathcal{B}_b$ mientras que valores próximos a 0 indican una peor correlación.
- Los algoritmos han sido implementados en Python 3.6.5 y ejecutados en ordenadores con sistema operativo Ubuntu 16.04.2 con CPU Intel Core i7-6700 a 3.40GHz y 16GB de memoria RAM.

D. Resultados

A continuación se muestran los resultados en términos de precisión y tiempo junto con su correspondiente análisis.

1) Precisión: Los resultados en términos de precisión se muestran en la Tabla II. Cada celda contiene la media del coeficiente τ_X [12] entre los rankings reales y predichos sobre los conjuntos de datos de test para la 5×10-cv. Para hacer más fácil la interpretación de los resultados, el (los) algoritmo(s)

TABLA II
PRECISIÓN MEDIA PARA CADA ALGORITMO.

Cjto. de datos	F-IBPLR	T-IBPLR	A-PLRT	PrA-PLRT	E-PLRT	PrE-PLRT
authorship	**0.9945**	**0.9945**	0.9461	0.9462	0.9468	0.9470
breast	0.7820	0.7820	0.7896	0.7832	0.8039	**0.8058**
ecoli	**0.8994**	**0.8994**	0.8764	0.8832	0.8829	0.8805
glass	0.7787	0.7787	0.7907	0.7907	**0.8187**	0.8165
iris	0.9582	0.9582	0.9560	0.9502	**0.9613**	0.9560
pendigits	**0.9953**	**0.9953**	0.9802	0.9802	0.9838	0.9838
segment	0.9603	0.9603	0.9776	0.9777	**0.9794**	0.9782
shuttle	0.9991	-	0.9996	0.9996	0.9997	**0.9998**
vehicle	0.7426	0.7426	0.7223	0.7271	0.7479	**0.7483**
vowel	**0.9672**	**0.9672**	0.8903	0.8898	0.9133	0.9119
wine	**0.9393**	**0.9393**	0.8921	0.8901	0.8971	0.9014

TABLA III
RESULTADOS DEL POST-HOC TEST PARA LA PRECISIÓN MEDIA.

Método	ranking	p-valor	victorias	empates	perdidos
E-PLRT	2.55	-	-	-	-
PrE-PLRT	2.68	**1.0000e+00**	5	0	6
F-IBPLR	3.18	**1.0000e+00**	6	0	5
T-IBPLR	3.27	**1.0000e+00**	6	0	5
A-PLRT	4.59	4.1376e-02	11	0	0
PrA-PLRT	4.73	3.1185e-02	10	0	1

TABLA IV
TIEMPO REQUERIDO POR CADA ALGORITMO (EN SEGUNDOS).

Cjto. de datos	F-IBPLR	T-IBPLR	A-PLRT	PrA-PLRT	E-PLRT	PrE-PLRT
authorship	1379.9	**91.3**	1760.0	1721.2	1411.2	1390.7
breast	38.2	**17.9**	270.1	252.3	314.5	291.2
ecoli	321.4	**103.3**	754.1	714.5	635.5	587.7
glass	139.4	**42.8**	635.0	600.3	556.5	520.0
iris	53.4	**10.6**	17.3	16.5	17.3	16.5
pendigits	205683.2	**6060.3**	33125.0	32205.9	28231.4	27622.2
segment	9431.9	**558.3**	10033.2	9862.7	9957.3	9791.8
shuttle	3113337.0	-	28378.8	28167.7	13507.7	**13439.6**
vehicle	1282.7	**83.9**	1896.3	1828.4	1538.5	1459.5
vowel	715.9	**243.6**	9829.5	9603.5	6737.8	6636.1
wine	78.0	**13.5**	159.7	158.4	132.1	128.8

TABLA V
RESULTADOS DEL POST-HOC TEST PARA EL TIEMPO.

Método	ranking	p-valor	victorias	empates	perdidos
T-IBPLR	1.45	-	-	-	-
PrE-PLRT	2.91	**1.0541e-01**	10	0	1
F-IBPLR	3.00	**1.0541e-01**	10	0	1
E-PLRT	4.09	2.8507e-03	10	0	1
PrA-PLRT	4.18	2.5158e-03	10	0	1
A-PLRT	5.36	4.7844e-06	10	0	1

con la mejor precisión están en negrita. Cabe destacar que el valor perdido en la celda correspondiente al algoritmo T-IBPLR para el conjunto de datos *shuttle* se debe a un error en memoria cuando se intenta almacenar las distancias entre todas las instancias.

El análisis estadístico se ha llevado a cabo utilizando el procedimiento estándar en aprendizaje automático descrito en [13], [14], utilizando el software disponible en [15]. Este procedimiento consta de dos pasos:

- Primero, hemos llevado a cabo un test de Friedman [16] con un nivel de significación de $\alpha = 0.05$. Puesto que el p-valor obtenido es $1.6112e^{-2}$, la hipótesis nula (de que todos los algoritmos son equivalentes en términos de precisión) es rechazada.
- Tras ello, hemos realizado un post-hoc test aplicando el procedimiento de Holm [17] con un nivel de significación de $\alpha = 0.05$. Básicamente, este test toma, como control, el algoritmo rankeado primero por el test de Friedman (E-PLRT) y se utiliza para descubrir métodos destacados.

Los resultados para el post-hoc test se muestran en la Tabla III. En esta se muestra el ranking obtenido por el test de Friedman y el p-valor ajustado por el procedimiento de Holm. Las columnas victorias, empates y perdidos se corresponde con el número de veces que el método de control gana, empata o pierde con el algoritmo correspondiente. Los p-valores en negrita se corresponden con hipótesis nulas no rechazadas.

De acuerdo a estos resultados, podemos concluir que:

- El algoritmo E-PLRT está rankeado en la primera posición por el test de Friedman y, por tanto, es utilizado como control. De hecho, la versión con pre-poda (PrE-PLRT) se encuentra en la segunda posición a pesar de ganar en seis ocasiones (de un total de once) a la propuesta sin ella.
- Las propuestas A-PLRT y PrA-PLRT están rankeadas en las últimas posiciones por el test de Friedman. Por tanto, tener en cuenta la sub-solución al problema en cada uno de los nodos del árbol no parece conducir (al menos en estas bases de datos) a mejores árboles en términos de precisión.
- Los resultados del post-hoc test revelan que los métodos T-IBPLR, F-IBPLR y PrE-PLRT no muestran diferencia estadística significativa con respecto al algoritmo E-PLRT.

2) Tiempo: En este artículo hemos trabajado con dos paradigmas del aprendizaje automático cuyo tiempo requerido para inducción e inferencia es claramente distinto. Para hacer una comparación adecuada, el tiempo de la CPU para el proceso completo (aprendizaje con el conjunto de entrenamiento y validación con el de test) sobre la 5x10-cv ha sido utilizado. Los resultados se muestran en la Tabla IV.

Como antes, hemos llevado a cabo el análisis estadístico descrito en [13], [14]:

- En el test de Friedman [16] con un nivel de significación de $\alpha = 0.05$ se obtiene un p-valor de $2.8972e^{-5}$. Por tanto, la hipótesis nula (de que todos los algoritmos son equivalentes en términos de tiempo) es rechazada.
- Por otro lado, el post-hoc test usando el procedimiento de Holm [17] con un nivel de significación de $\alpha = 0.05$ se muestra en la Tabla V.

De acuerdo a los resultados obtenidos, podemos concluir que:

- El test de Friedman rankea en primera posición el algoritmo T-IBPLR y, por tanto, es utilizado como control.
- Las propuestas A-PLRT y PrA-PLRT están rankeadas en las últimas posiciones por el test de Friedman, puesto que requieren mayor tiempo de CPU dado que deben tener en cuenta la sub-solución al problema en cada uno de los nodos del árbol.
- Los resultados del post-hoc test revelan que los métodos F-IBPLR y PrE-PLRT no muestran diferencia estadística significativa con respecto al algoritmo T-IBPLR.

Cabe destacar que, en este caso, el número de instancias en la mayoría de los conjuntos de datos utilizados es pequeño. No obstante, como puede observarse en la Tabla IV, para los conjuntos de datos con un número de instancias (considerablemente) grande, el algoritmo T-IBPLR no puede almacenar todas las distancias (por lo que no acaba) y el algoritmo F-IBPLR requiere de un gran tiempo de CPU. Por tanto, si hubiéramos utilizado (en su mayoría) conjuntos de datos con un alto número de instancias, el análisis llevado a cabo en términos de tiempo mostraría resultados más favorables para nuestras propuestas basadas en árboles de decisión, pues estas escalan mejor que los algoritmos basados en el paradigma de los vecinos más cercanos al crecimiento en el número de instancias de los conjuntos de datos.

V. Conclusiones

En este trabajo hemos planteado un nuevo problema para label ranking, en el que las instancias tanto del conjunto de entrenamiento como de test están etiquetadas con rankings completos y parciales.

Basándonos en [3], hemos decidido diseñar algoritmos basados en el paradigma de los vecinos más próximos (IBPLR) y en árboles de clasificación/regresión, teniendo que lidiar con el problema de no disponer de una distribución de probabilidad para rankings completos y parciales.

Atendiendo a esto, hemos adaptado el algoritmo IBLR al problema planteado, que hemos denotado como IBPLR, y hemos diseñado dos alternativas para PLRT, una teniendo en cuenta la sub-solución al problema en el nodo correspondiente y otra sin tener en cuenta dicha sub-solución.

De la evaluación experimental, podemos concluir que el clasificador de PLRT usando el criterio de entropía con y sin pre-poda es competitivo con el clasificador IBPLR en términos de precisión. Por otro lado, el rendimiento en términos de precisión para el clasificador PLRT usando desacuerdo como criterio, está por debajo del ofrecido por los anteriores, a pesar de tener en cuenta la sub-solución al problema en el nodo correspondiente. No obstante, cabe destacar que este último no presenta malos resultados en términos del coeficiente de correlación de rankings τ_X.

En cuanto a tiempo, el rendimiento del clasificador IBPLR pre-calculando las distancias está por encima del resto de clasificadores. No obstante, como ya se comentó, si el número de instancias en los conjuntos de datos es considerablemente grande, el clasificador IBPLR pre-calculando las distancias puede que no disponga de memoria suficientemente mientras que sin pre-calculo el tiempo requerido es excesivo.

Como trabajo futuro, planteamos un enfoque más flexible en el que se permita que los conjuntos de datos estén etiquetados con rankings parciales, posiblemente incompletos. Además, para reducir el tiempo de inducción en los clasificadores PLRT, planteamos diseñar clasificadores débiles, que, para mejorar la precisión, se combinen en multi-clasificador (p.e., bosques aleatorios). También, para tratar de seguir la metodología propuesta en [3], planteamos estudiar una distribución de probabilidad para modelar bucket orders. Por último, pensamos que una posible área de aplicación del PLR sería en el campo del meta-aprendizaje, para recomendar los clasificadores apropiados a un conjunto de datos determinado, en función de sus características.

Agradecimientos

Este artículo ha sido parcialmente financiado por fondos FEDER y la Agencia Estatal de Investigación (AEI/MINECO) mediante el proyecto TIN2016-77902-C3-1-P.

Referencias

[1] R. Duda, P. Hart, and D. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[2] J. Hernández-González, I. Inza, and J. Lozano, "Weak supervision and other non-standard classification problems: A taxonomy," *Recognition Letters*, vol. 69, pp. 49–55, 2002.

[3] W. Cheng, J. Hühn, and E. Hüllermeier, "Decision tree and instance-based learning for label ranking," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 161–168.

[4] S. Lin, "Rank aggregation methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, pp. 555–570, 2010.

[5] A. Gionis, H. Mannila, K. Puolamäki, and A. Ukkonen, "Algorithms for Discovering Bucket Orders from Data," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2006, pp. 561–566.

[6] J. Aledo, J. Gámez, and A. Rosete, "Utopia in the solution of the Bucket Order Problem," *Decision Support Systems*, vol. 97, pp. 69–80, 2017.

[7] J. Aledo, J. Gámez, and D. Molina, "Tackling the supervised label ranking problem by bagging weak learners," *Information Fusion*, vol. 35, pp. 38 – 50, 2017.

[8] C. de Sá, C. Soares, A. Knobbe, and P. Cortez, "Label ranking forests," *Expert Systems*, vol. 34, no. 1, 2017.

[9] P. Emerson, "The original Borda count and partial voting," *Social Choice and Welfare*, vol. 40, no. 2, pp. 353–358, 2013.

[10] J. Alfaro, "Label ranking classifiers to deal with partial rankings," Undergraduate Dissertation, School of Computer Science and Engineering, University of Castilla-La Mancha, 2018.

[11] D. Dheeru and E. Karra, "UCI machine learning repository," http://archive.ics.uci.edu/ml, 1987, online. Last accessed: September 4, 2018.

[12] E. Emond and D. Mason, "A new rank correlation coefficient with application to the consensus ranking problem," *Journal of Multi-Criteria Decision Analysis*, vol. 11, pp. 17–28, 2002.

[13] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[14] S. García and F. Herrera, "An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons," *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[15] J. Arias and J. Cózar, "ExReport: Fast, reliable and elegant reproducible research," http://exreport.jarias.es/, 2015, online. Last accessed: September 4, 2018.

[16] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Ann. Math. Statist.*, vol. 11, pp. 86–92, 1940.

[17] S. Holm, "A Simple Sequentially Rejective Multiple Test Procedure," *Scandinavian Journal of Statistics*, vol. 6, pp. 65–70, 1979.