Predicción de rankings usando *bagging* y árboles de decisión

Juan A. Aledo[1] and Jose A. Gámez[2] y David Molina[3]

[1] Departamento de Matemáticas, Albacete, UCLM (juanangel.aledo@uclm.es)
[2] Departamento de Sistemas Informáticos, Albacete, UCLM (jose.gamez@uclm.es)
[3] Departamento de Matemáticas, Ciudad Real, UCLM (david.molina@uclm.es)

Resumen Este artículo está dedicado al estudio del problema de predicción de rankings (LRP). En concreto, se propone la aplicación de la técnica bagging en algoritmos basados en árboles de decisión para resolver el problema LRP. Partiendo de un algoritmo conocido (LRT) de aprendizaje de árboles de decisión para el problema LRP, se proponen dos modificaciones más simples, pero también más rápidas, basadas en el uso de discretización no supervisada para seleccionar los umbrales de decisión para las variables predictoras. A continuación se experimenta el uso de ensembles de árboles usando la técnica de bagging.
Las propuestas son validadas mediante un estudio experimental basado en 16 conjuntos de datos tomados de [12] y comparando con los dos algoritmos presentados en dicho artículo. Se han considerado dos casos distintos, atendiendo a la obligatoriedad de que todos los rankings sean completos o a que se permitan rankings incompletos (no todas las etiquetas de la variable clase son rankeadas). Los resultados obtenidos indican que los ensembles funcionan mejor que los algoritmos en [12], con especial enfásis en el caso incompleto. Además, se observa que el uso conjunto de los clasificadores más simples con bagging da lugar a resultados competitivos en exactitud pero mucho más rápidos que los basados en el algoritmo de partida (LRT).

Keywords: Rankings, árboles de decisión, bagging, aprendizaje automático, problema de predicción de rankings.

1. Introducción

En este artículo nos centramos en la tarea de clasificación supervisada *no estándar* conocida como *problema de predicción de rankings* [12], cuyo objetivo es predecir un ranking entre las diferentes etiquetas que puede tomar la variable clase a partir del valor de una serie de variables predictoras. Para ello, nos hemos basado en el trabajo de Cheng et al. [12], donde los autores diseñan dos algoritmos para este problema basados en técnicas de aprendizaje automático, uno basado en instancias (IBLR) y

otro basado en árboles de decisión/regresión (LRT). Los resultados muestran que IBLR supera en tasa de acierto a LRT [12]. Sin embargo, desde el punto de vista computacional, IBLR no escala bien a problemas con muchas instancias y necesita mucho más tiempo de inferencia que LRT.

Nuestro objetivo es mejorar el rendimiento de las propuestas presentadas en [12], especialmente en el caso incompleto, desarrollando dos nuevos métodos basados en el algoritmo LRT. En particular:

- Consideramos un *ensemble* de árboles basados en la combinación del algoritmo LRT y la popular técnica *bagging* [2].
- Proponemos dos modificaciones al algoritmo LRT basadas en discretización no supervisada. Su combinación con bagging da lugar a algoritmos más rápidos (escalables) y con resultados competitivos en cuanto a tasa de acierto con respecto a LRT.
- Desarrollamos una extensa experimentación para validar empíricamente nuestra propuesta.

2. El problema de predicción de rankings

Los rankings representan preferencias de forma natural. Específicamente, dado un conjunto de elementos $\mathcal{I} = \{1, 2, \ldots, k\}$, un ranking π es un orden de preferencia de (algunos de) esos elementos. Los rankings pueden ser clasificados como *completos* (los k elementos están ordenados) o *incompletos* (sólo hay p elementos ordenados, con $2 \leq p < k$). Los rankings completos son permutaciones de los elementos de $\mathcal{I}$. El conjunto de todas las permutaciones de k elementos es el grupo simétrico $\mathbb{S}_k$.

El objetivo del problema LRP (Label Ranking Problem) es aprender un clasificador-LR a partir de los datos [12]. Es importante notar que un clasificador-LR siempre produce una permutación o ranking completo. Sin embargo, como detallamos en secciones posteriores, permitiremos la presencia de rankings incompletos en las instancias de entrenamiento. Las dos siguientes definiciones formalizan nuestro problema:

Definición 1 (Clasificador-LR) Dado un conjunto de n atributos predictivos, $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ y una variable de clase C, con dominio $dom(C) = \{1, \ldots, k\}$, un clasificador Label Ranking (clasificador-LR) $\mathcal{C}$ es un modelo del tipo:

$$\begin{aligned} &\mathcal{C} : dom(X_1) \times dom(X_2) \times \cdots \times dom(X_n) \longrightarrow \mathbb{S}_k, \\ &\mathcal{C} : \qquad\qquad\qquad\qquad\qquad\qquad x_1, x_2, \ldots, x_n \longmapsto \pi \end{aligned}$$

es decir, una función que asigna un ranking de los valores de $dom(C)$ a cualquier configuración de $\prod_{i=1}^{n} dom(X_i)$.

Definición 2 (Problema de predicción de rankings) Dado un conjunto de datos $\mathbf{D} = \{(x_1^j, \ldots, x_n^j, \pi^j)\}_{j=1}^N$ de N instancias, entonces el problema de predicción de rankings (LRP) consiste en aprender un clasificador-LR a partir de $\mathbf{D}$ que generalice bien datos no entrenados previamente.

Un ejemplo claro de aplicación de este problema podría ser un sistema de recomendación de películas, en el que según las preferencias presentadas por un usuario se le indicaría un ranking de películas según sus afinidades con otros usuarios y las preferencias de éstos.

3. Algoritmo basado en árboles de decisión para el problema LRP

La inducción usando árboles de decisión es una de las técnicas más utilizadas en aprendizaje automático, tanto para tareas de clasificación [10] como de regresión [8]. Esta sección describe el algoritmo propuesto en [12] basado en árboles de decisión aplicado al problema LRP.

3.1. Entrenamiento

Se trabaja de forma recursiva. El algoritmo recibe un conjunto de s instancias $\mathbf{R} = \{(x_1^j, \ldots, x_k^j, \pi^j)\}_{j=1}^s$, siendo $k \leq n$ y $s \leq N$. A partir de los datos, el algoritmo decide si terminar el proceso creando un nodo hoja o usar un atributo predictivo X_i para partir $\mathbf{R}$ en dos conjuntos.

El criterio de parada se alcanza si todos los rankings del conjunto son el mismo o si $s \leq 2n$. En el primer caso, el ranking asociado a la hoja es el asociado a las instancias y en el segundo caso, se obtiene usando el método de recuento de Borda [4,11] para todos los rankings del conjunto.

Si no se alcanza el criterio de parada, el algoritmo evalúa todos los (N) atributos y todos los posibles puntos de corte buscando aquel que minimiza la incertidumbre asociada a la partición. Formalmente, dado un atributo X_i y un umbral t, su *incertidumbre* asociada es evaluada como:

$$f(X_i^t) = f(\mathbf{R}_{\leq t}, \mathbf{R}_{>t}) = \frac{|\mathbf{R}_{\leq j}| \cdot \theta_{\leq t} + |\mathbf{R}_{>t}| \cdot \theta_{>t}}{|\mathbf{R}|}. \quad (1)$$

donde $\mathbf{R}_{\leq t}$ y $\mathbf{R}_{>t}$ son las particiones de $\mathbf{R}$ inducidas por el umbral t para X y $\theta_{\leq t}$ (resp. $\theta_{>t}$) es el parámetro de dispersión asociado a la distribución de Mallows [9] aprendida a partir de los rankings en $\mathbf{R}_{\leq t}$ (resp. $\mathbf{R}_{>t}$) (Para más detalles ver[12]). Una vez seleccionado el atributo y su umbral, se subdivide el conjunto de instancias en función de ellos y se continua con el proceso recursivo de construcción del árbol.

3.2. Clasificación

Dada una instancia la clasificación es, en general, muy sencilla: empezando por el nodo raíz se recorre el árbol siguiendo el camino apropiado que marca el valor de los atributos predictivos de la instancia. Al llegar a un nodo hoja, se devuelve su ranking asociado. En el caso de rankings incompletos, si el nodo hoja contiene un ranking incompleto, se retrocede en el árbol hasta lograr suficiente información para completarlo [12].

4. Propuesta: Un conjunto de clasificadores simples aplicado al problema LRP

El uso de un conjunto de clasificadores en lugar de uno, en general, se traduce en un incremento en la tasa de acierto [7]. Una de las técnicas más simples de construir el conjunto de clasificadores es mediante *bagging* [2], que en su forma canónica consiste en:

- Usar muestreo con reemplazo para generar b conjuntos distintos de instancias $\{\mathbf{D}_1, \ldots, \mathbf{D}_b\}$ a partir de conjunto inicial $\mathbf{D}$ y con su mismo número de instancias.
- Usar un algoritmo de aprendizaje automático para aprender un modelo M_i para cada conjunto $\mathbf{D}_i$.
- Dada una nueva instancia a clasificar $\mathbf{x} = (x_1, \ldots, x_n)$, devuelve como predicción $g(M_1(\mathbf{x}), \ldots, M_b(\mathbf{x}))$, donde $M_i(\mathbf{x})$ es la predicción del modelo M_i y g es una función de agregación (por ejemplo, el voto mayoritario).

El objetivo de este artículo es estudiar el beneficio de aplicar bagging al problema LRP usando como clasificadores base LRT y propuestas derivadas a partir de él. En particular, la idea es entrenar b árboles de decisión y después usar la permutación de consenso (la que mejor representa un conjunto de permutaciones [11]) de $\{M_1(\mathbf{x}), \ldots, M_b(\mathbf{x})\}$ como función de agregación. Sin embargo, al mismo tiempo que esperamos un incremento en la tasa de acierto usando bagging, también aumenta la complejidad computacional. Por ello, proponemos el uso de árboles sencillos, que reducen la complejidad de calcular el punto de corte en cada paso. De este modo obtenemos clasificadores más simples y rápidos, que combinados con bagging producen resultados competitivos.

4.1. W-LRT y F-LRT

En esta sección mostramos dos modificaciones a LRT. El criterio de partición usado en LRT es de carácter supervisado y produce puntos de

corte *óptimos* para el problema LRP. Sin embargo, el tiempo empleado para calcularlos es alto, puesto que para cada nodo recorre todos los atributos y todos los valores de dicho atributo en el conjunto de datos recibido. Proponemos simplificar este proceso evitando evaluar todos los puntos de corte posibles para cada atributo y sustituir el criterio de decisión por uno no-supervisado que permite establecer el punto de corte de forma directa:

- W-LRT. Este algoritmo escoge como umbral para cada atributo X_i el valor que parte su dominio en dos intervalos de igual anchura (Width-LRT), es decir, el punto medio entre el máximo y el mínimo valor del atributo.
- F-LRT. Este algoritmo escoge como umbral para cada atributo X_i el valor que parte su dominio en dos intervalos de igual frecuencia (Frequency-LRT), es decir, con el mismo número de valores en cada intervalo (mediana de la distribución).

Una vez establecido el umbral de cada atributo, se evalúan usando la ecuación 1 y se selecciona la variable que minimiza la incertidumbre (al igual que ocurría en el algoritmo LRT).

5. Evaluación experimental

Esta sección desarrollamos una extensa evaluación experimental para validar empíricamente nuestra propuesta. A continuación, describimos las bases de datos utilizadas, los algoritmos involucrados y la metodología seguida en los experimentos.

5.1. Bases de datos

Utilizamos como punto de referencia las 16 bases de datos[4] consideradas en [12]. En todos los casos, las instancias están etiquetadas con permutaciones. La Tabla 1 muestra las principales características de cada base de datos.

5.2. Algoritmos

En este estudio hemos considerados los siguientes algoritmos:

- El algoritmo LRT [12].

[4] Las bases de datos están disponibles en www.uni-marburg.de/fb12/kebi/research

Tabla 1. Bases de datos.

Base de datos	# inst.	# atrib.	# clases	Base de Datos	# inst.	# atrib.	# clases
authorship	841	70	4	bodyfat	252	7	7
calhousing	20640	4	4	cpu-small	8192	6	5
elevators	16599	9	9	fried	40769	9	5
glass	214	9	6	housing	506	6	6
iris	150	4	3	pendigits	10992	16	10
segment	2310	18	7	stock	950	5	5
vehicle	846	18	4	vowel	528	10	11
wine	178	13	3	wisconsin	194	16	16

– Los dos algoritmos propuestos basados en discretización no supervisada: W-LRT y F-LRT.
– Las propuestas basadas en bagging para los tres algoritmos basados en árboles: LRTb, W-LRTb and F-LRTb, donde b denota el número de modelos considerados en bagging.
– Como referencia, también consideramos el algoritmo IBLR [12], un algoritmo que retorna la permutación de consenso de los k vecinos más cercanos, calculada utilizando el método de recuento de Borda [4,11] con un método de mejora iterativa. Se utiliza la distancia euclídea para identificar los vecinos más cercanos y se fija el número de vecinos mediante una validación cruzada, considerando todos los valores enteros del intervalo $[1, \sqrt{N}]$.

5.3. Metodología

Hemos tomado las siguientes decisiones en cuanto al diseño de los experimentos:

– En todos los casos, los algoritmos son evaluados mediante cinco repeticiones de una diez validación cruzada.
– Al igual que en [12,13] usamos el coeficiente de Kendall como tasa de acierto. Formalmente, dadas dos permutaciones π_1 y π_2 de k elementos, este coeficiente viene dado por

$$\tau(\pi_1, \pi_2) = \frac{C(\pi_1, \pi_2) - D(\pi_1, \pi_2)}{\frac{k(k-1)}{2}}, \tag{2}$$

donde $C(\pi_1, \pi_2)$ (respectivamente $D(\pi_1, \pi_2)$) es el número de pares concordantes (respectivamente discordantes) entre π_1 y π_2. Este coeficiente toma valores en el intervalo $[-1, 1]$, siendo 1 en el caso de permutaciones idéndicas y -1 cuando son inversas.

– Consideramos tres escenarios diferentes: (1) caso completo; (2) caso incompleto con el 30 % de etiquetas borradas en los rankings; (3) caso incompleto con el 60 % de etiquetas borradas en los rankings.
  Para tratar los casos incompletos, borramos etiquetas en los rankings del conjunto de entrenamiento con probabilidad 0.3 y 0.6 respectivamente. En cualquier caso, ningún ranking incompleto tiene menos de 2 etiquetas.
– Atendiendo a los clasificadores de bagging, estudiamos todos los casos desde $b = 1$ hasta $b = 100$, que fijamos como valor máximo.
– Todos los algoritmos han sido escritos en Java. Los experimentos han sido desarrollados en ordenadores con sistema operativo Linux con una memoria RAM máxima de 20 GB.

5.4. Resultados

En este sección presentamos los resultados obtenidos así como su análisis. Ante la imposibilidad, por limitación de espacio, de mostrar los resultados detallados para cada algoritmo y conjunto de datos, las figuras 1, 2 y 3 recogen los resultados promediados sobre los 16 conjuntos de datos y para los tres escenarios considerados. El eje x representa el número de modelos (b) usado en bagging.

Podemos sacar las siguientes conclusiones:

– Como se observa claramente, el uso de bagging incrementa la tasa de acierto del proceso de clasificación para los clasificadores base.
– En el caso completo, el algoritmo IBLR muestra una muy buena actuación, y los algoritmos con bagging necesitan un gran número de clasificadores para mostrar resultados competitivos. Sin embargo, en el caso incompleto, IBLR es claramente derrotado por estos algoritmos, incluso usando un número pequeño de modelos.
– El mejor método basado en bagging es el que usa el algoritmo LRT como clasificador base, aunque también es el que más recursos precisa y más tiempo emplea. Los métodos que usan W-LRT y F-LRT como clasificadores base presentan resultados competitivos con el que usa LRT, especialmente cuando se usan más de 25 modelos.

Para poder extraer conclusiones de un modo más completo, hemos realizado un análisis estadístico para cada escenario considerado atendiendo al procedimiento estándar descrito en [3] usando el software disponible en [1]. Para este análisis, consideramos los algoritmos LRT, IBLR y, además, los algoritmos basados en bagging con 25 clasificadores (LRT25b, W-LRT25b y F-LRT25b) y con 100 clasificadores (LRT100b, W-LRT100b y F-LRT100b):

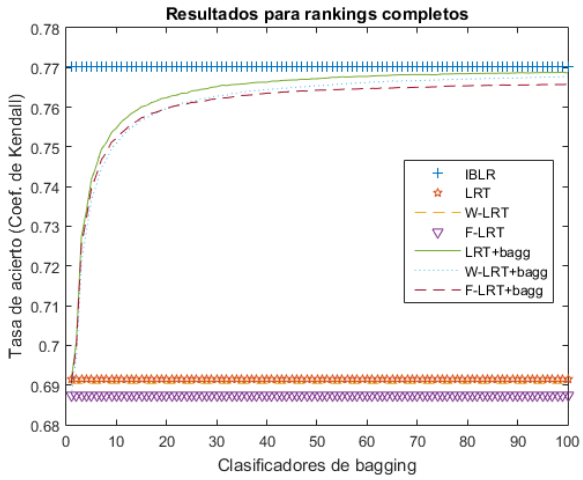

Figura 1. Rendimiento medio de cada algoritmo usando rankings completos

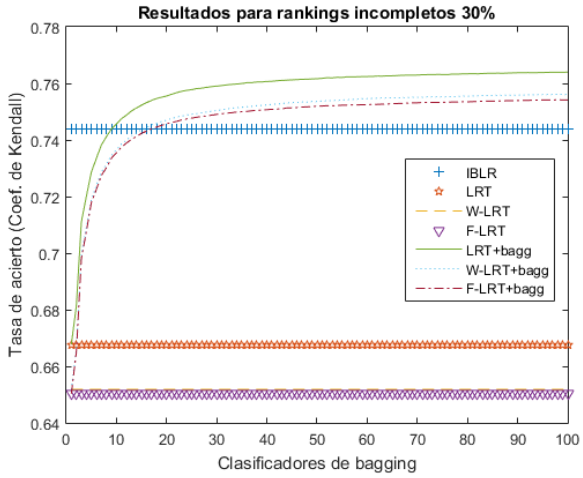

Figura 2. Rendimiento medio de cada algoritmo usando rankings con el 30 % de etiquetas borradas

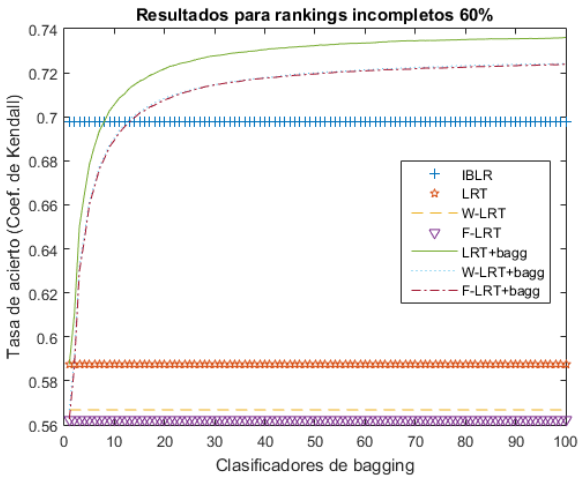

Figura 3. Rendimiento medio de cada algoritmo usando rankings con el 60 % de etiquetas borradas

- En primer lugar, realizamos un test de Friedman [5], usando un nivel de significancia del 5 %. El test rechaza las hipótesis nulas de que todos los algoritmos son equivalentes.
- En segundo lugar, realizamos un test post-hoc test usando el procedimiento de Holm [6] también con un nivel de significancia del 5 %. Este test compara todos los algoritmos con el que mejor ranking medio tiene en todos los experimentos (el primero en el ranking calculado por el test de Friedman).

Los resultados de los tests estadísticos para cada uno de los escenarios son los siguientes:

- En el caso completo, el algoritmo LRT100b es tomado como control. Los algoritmos W-LRT100b, IBLR, F-LRT100b y LRT25b no muestran diferencias significativas con sus resultados.
- En el caso incompleto-30, el algoritmo LRT100b es tomado como control. Los algoritmos W-LRT100b, F-LRT100b y LRT25b no muestran diferencias significativas con sus resultados.
- En el caso incompleto-60, el algoritmo LRT100b es tomado como control. Los algoritmos W-LRT100b y F-LRT100b no muestran diferencias significativas con sus resultados.

Como podemos ver, el algoritmo LRT100b siempre constituye el mejor método. Además, según los datos son más incompletos, los algoritmos no

basados en bagging o con un número de clasificadores pequeño quedan relegados a un segundo lugar en cuanto a tasa de acierto.

6. Conclusiones

Hemos realizado un estudio comparativo entre algoritmos en la resolución del problema LRP. Partiendo de los resultados de [12], hemos adaptado el algoritmo basado en árboles de decisión e introducido algunas versiones simplificadas para probar su eficacia al combinarlas con la técnica de bagging.

Hemos realizado experimentos con rankings completos (escenario general) y con rankings incompletos (pérdida de información) con el objetivo de probar su comportamiento en diferentes contextos. Los algoritmos propuestos obtienen buenas soluciones, mejorando en la mayoría de los casos los resultados de los algoritmos IBLR y LRT. De hecho, las versiones W-LRT y F-LRT en combinación con la técnica de bagging obtienen resultados competitivos a las basadas en LRT en un tiempo mucho más reducido.

7. Trabajos futuros

Como posibles líneas futuras de investigación, proponemos:

- Intentar la adaptación de métodos probabilísticos (por ejemplo, tipo Naive Bayes) a este problema, si bien es obvio que habría que hacer convivir nodos basados en distribuciones multinomiales o Gaussianas con el nodo clase, que estaría modelado por una distribución de Mallows.
- Abordar el problema LRP mediante el uso de clasificadores que toman como clase la relación de orden entre dos etiquetas. Por ejemplo, si tenemos tres etiquetas $\{A, B, C\}$, se construyen clasificadores que tienen como variable objetivo: $A \prec B$, $A \prec C$ y $B \prec C$. El ranking predicho para un registro, se compone entonces a partir de los resultados obtenidos en los clasificadores individuales. Nuestra idea es aplicar técnicas que han funcionado bien en problemas de clasificación estándar que descomponen el proceso en varios clasificadores pair-wise usando la técnica conocida como OvO (one vs one classifiers).

8. Agradecimientos

Este artículo ha sido parcialmente financiado por MINECO y fondos FEDER mediante el proyecto TIN2013-46638-C3-3-P.

Referencias

1. J. Arias and J. Cózar. ExReport: Fast, reliable and elegant reproducible research, 2015.
2. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
3. J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
4. P. Emerson. The original borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, 2013.
5. M. Friedman. A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.
6. S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70, 1979.
7. L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.
8. R. A. Olshen, L. Breiman, J. H. Friedma and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.
9. C. L. Mallows. Non-null ranking models. I. *Biometrika*, 44(1-2):114–130, 1957.
10. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
11. F. Schalekamp and A. van Zuylen. Rank aggregation: Together we're strong. In *ALENEX*, pages 38–51, 2009.
12. 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*, ICML '09, pages 161–168. ACM, 2009.
13. K. Dembczynski, W. Cheng and E. Hüllermeier. Label ranking methods based on the plackett-luce model. In *ICML*, pages 215–222, 2010.