Universidad Carlos III de Madrid

Aprendizaje Automático

Computer Science Engineering

Tutorial 2: Introducción a Weka

Authors:
Daniel Medina García
Alejandro Rodríguez Salamanca

February 24, 2016

Contents

Introducción

Esta práctica consiste en un primer acercamiento a Weka y su funcionamiento. El objetivo es resolver una serie de ejercicios aplicando los filtros y algoritmos que nos ofrece Weka a unos archivos de datos. El documento se encuentra dividido en siete secciones, correspondiendo cada una a un ejercicio.

1 Los ficheros de datos

¿Cuántos atributos de entrada tiene el fichero de datos? ¿De qué tipo son?

`badges`	Este archivo contiene un atributo de entrada de tipo String que se corresponde con un nombre, y un atributo de salida, la clase a la que pertenece.
`badges_plain`	El contenido de este archivo es el mismo que el de `badges`, con la diferencia de que los nombres están previamente enumerados.

¿Podría un algoritmo de aprendizaje automático identificar esa función con los datos que hay en ese fichero? ¿Por qué?

No. El fichero contiene tan solo un campo de datos, de tipo enumerado, y cada posible valor de dicho campo sólo se encuentra en una instancia. Al ser una clase con dos valores (muchos menos que valores del enumerado) y ser todas las instancias distintas, toda información que saque un algoritmo de aprendizaje automático será meramente memorización, sin valor alguno en futuras instancias con valores distintos a los actuales ya que no existe razonamiento o computación en la resolución de la clase.

2 Clasificar con ZeroR

¿Qué resultado en términos de instancias correctas ofrece el algoritmo ZeroR?

ZeroR indica un resultado del **51.0204%** de instancias correctas. Lo único de lo que nos informa este dato es de que las clases están equilibradas dentro de nuestro conjunto de datos.

¿Qué ocurre si se selecciona otro algoritmo de clasificación permitido para ese conjunto de datos?

Utilizando meta / MultiScheme **obtenemos exactamente el mismo resultado** que con ZeroR. Esto es lógico pues no se puede mejorar el resultado al carecer de información suficiente para razonar sobre los datos.

¿Cuáles son las diferencias al repetir los pasos anteriores con el otro fichero, `badges_plain`**?**

Con ZeroR **ninguna**, pues dicho algoritmo sólo tiene en cuenta el número de instancias pertenecientes a cada una de las clases. Con cualquier otro algoritmo, de nuevo **ninguna**, pero en este caso es debido a que no se posee información suficiente para aplicar ningún criterio informado en el etiquetado.

¿Qué ocurre si seleccionamos el algoritmo trees / ID3 en el segundo fichero?

Si probamos con ID3 podemos observar que el **porcentaje de instancias correctamente clasificadas es del 100%**. Si usásemos validación cruzada seguramente los resultados se parecerían más a los obtenidos con ZeroR, ya que se sigue sin tener información para clasificar más allá de la memorización y por ello no se tiene un criterio razonado para clasificar nuevas instancias.

3 Generando nuevos atributos

Propón 6 nuevos atributos y explica por qué los has elegido.

Al tener como datos de entrada nombres propios, sólo podemos basarnos en la naturaleza de las palabras que los componen para realizar algún tipo de razonamiento. Así, proponemos algunos nuevos campos a partir del original que detallan las características de dichos nombres:

- Número de **letras** del nombre.
- Número de **vocales**.
- Número de **consonantes**.
- Número de **espacios**.
- **Inicial** del nombre.
- **Inicial** del apellido.

¿Cuántos atributos tiene el fichero `badges1` y de qué tipo son?

Este archivo contiene atributos que persiguen lo mismo que planteábamos en la pregunta anterior: descubrir más datos sobre los nombres. Así, el archivo contiene **siete** nuevos campos de entrada además del nombre de tipos **numérico y nominal**.

¿Qué otro tipo de información estadística se muestra sobre los atributos? Tras pulsar el botón "Visualize all" indica qué se muestra y si hay algún atributo que no se visualice.

En cuanto a la información estadística sobre los atributos, si el atributo es de tipo **nominal**, aparece el **valor y el número de repeticiones**. En cambio, si es de tipo **numérico**, Weka nos ofrece los valores **máximo, mínimo, medio** y la **desviación estándar**.

"Visualize all" muestra la **correlación** entre los valores para cada atributo y su pertenencia a una u otra clase.

Genera un clasificador con ZeroR, ¿qué ocurre? Compara los resultados con los obtenidos en el ejercicio anterior.

ZeroR es un clasificador que sólo tiene en cuenta el número de instancias que pertenecen a cada clase. Es por eso que su éxito al clasificar **no varía** respecto al anterior conjunto de datos que no ofrecía información.

Genera un clasificador con trees / ID3, ¿qué ocurre? ¿Qué se podría hacer para solucionar este problema?

No es posible generar un clasificador con ID3 porque ID3 no acepta datos del tipo numérico. Así, una posible solución sería convertir el tipo de entradas a otros tipos de atributo que no sean numerales (discretización).

4 Clasificar con ID3: resolviendo problemas

¿Qué información aparece en el desplegable tras abrir la pestaña *Capabilities* de ZeroR? ¿Qué información proporciona "more"?

En el desplegable aparecen los **tipos de clase y de atributos que acepta el algoritmo**. Además, indica como información "Adicional" el hecho de que ZeroR no tiene un **número mínimo de entradas** para su funcionamiento.

"more" muestra **información no funcional** sobre el algoritmo. Incluye el **identificador** unívoco del algoritmo dentro **de Weka**, una breve **sinopsis** explicando cómo trabaja dicho algoritmo y sus posibles **opciones de configuración**.

Los atributos de entrada pueden modificarse a través de tareas de preprocesamiento. En los siguientes pasos vamos a modificar ciertos atributos de `badges1` para que pueda clasificarse con ID3.

¿Qué efecto tiene el filtro de discretización sobre el conjunto de datos con *bins* igual a 5?

Discretiza, i.e. **separa los valores de un atributo con valores continuos en su pertenencia a un rango determinado**, los atributos previamente de carácter numérico convirtiéndolos en enumerados con tantos valores distintos como rangos creados. Creará tantos rangos como le fijemos a través de *bins*.

¿Cuántas instancias clasifica bien ahora ID3 usando un conjunto de entrenamiento? ¿Qué porcentaje representa? ¿Qué crees que indica la "matriz de confusión"? ¿Cuántas instancias de cada tipo se han clasificado mal?

ID3 clasifica bien **284 de 294** instancias con *bins* igual a 10, lo que supone un **96.5986%** de éxito. Tras reducir *bins* a 5, clasifica correctamente 48 instancias menos (**236** de 294), resultando en un **80.2721%** de precisión. En ambos casos es un resultado considerablemente superior al obtenido con ZeroR, lo que nos indica que el algoritmo ha conseguido razonar sobre la información de los atributos para clasificar.

La **matriz de confusión** desglosa el éxito en la clasificación por cada una de las clases, y nos deja ver que han sido **9** las instancias **erróneamente clasificadas como** pertenecientes a la **clase negativa** mientras que tan sólo **1** instancia fue malinterpretada como **positiva**.

¿Cuál es la primera instancia del conjunto de entrenamiento que se clasifica mal cuando se usa la opción de mostrar las predicciones en la salida? ¿Por qué?

La primera instancia incorrectamente etiquetada es **la séptima**, ya que su clase es negativa y nuestro algoritmo la ha clasificado como positiva con un 57.1% de probabilidad. La opción marcada hace imprimir una tabla con tantas filas como instancias a clasificar, y los siguientes campos:

inst#	ID de la instancia.
actual	Clase a la que realmente pertenece la instancia.
predicted	Resultado de la clasificación de nuestro algoritmo.
error	Aparecerá un '+' si la instancia fue erróneamente clasificada.
prob. distr.	Porcentaje de seguridad, para cada clase, de la pertenencia de la instancia a dicha clase.

¿Cómo se clasificaría la instancia "Eloisa Figueroa"? ¿Cuáles son los atributos de este nombre? ¿Qué ocurre con los valores de esta instancia si utilizas el filtro usado anteriormente?

Siguiendo el árbol, como el nombre tiene una **longitud** entre 14.2 y 17.8 **(15)**, menos de 1.4 **espacios (1)** y menos de 10.8 **consonantes (5)**, la instancia sería clasificada como **negativa**.

Otros campos no utilizados son la **paridad y la primera letra vocal**, de fácil deducción con valores **0 y 1 respectivamente**, la inclusión de **puntos en el nombre (0)** y el número de **palabras (2)**.

Si nos fijamos en el reparto de instancias por clase dentro de un mismo valor para los atributos recién mencionados podemos observar cómo el porcentaje de cada clase ronda el 50%, lo que indica que estos campos no añaden información para el algoritmo, ocurriendo lo mismo con el número de espacios.

Por otro lado, la longitud sí parece jugar un papel importante en la decisión. Echando un breve vistazo al reparto de clases en los **valores discretizados** de la longitud, vemos que **según crece la longitud del nombre** encontramos más instancias pertenecientes a la clase **negativa**, mientras que los nombres **más cortos** tienen más probabilidades de pertenecer a la **positiva**. Parece haber una cierta correlación con el número de consonantes, pero ésta no es tan clara.

A continuación modificamos el fichero original introduciendo el nombre anterior y clasificándolo como positivo, teniendo en cuenta que si contiene enumerados y se introduce un nuevo valor hay que especificarlo también en la definición de los valores posibles del enumerado. Después volvemos a generar el clasificador con ZeroR y training set seleccionado.

¿Cómo se clasifica la instancia nueva?

Al haber una ligera mayoría de instancias positivas, ZeroR elige la positiva como la clase dominante. Así, la nueva instancia es clasificada correctamente como **positiva**.

5 Clasificar con J48 (C4.5)

¿Cuántas hojas tiene el árbol para `badges1` generado con J48 usando conjunto de entrenamiento? ¿Cuántas instancias del conjunto de entrenamiento clasifica bien? ¿Qué porcentaje representa? ¿Cuántas instancias de cada tipo se han clasificado mal? ¿Cómo se clasificaría la instancia "Eloisa Figueroa"?

El árbol generado con J48 tiene **20 hojas**. Este árbol **clasifica correctamente 287 instancias**, que equivale a un **97.619%**. Quedan **incorrectamente clasificadas 7 instancias: 3 en la clase negativa y 4 en la clase positiva.**

Este árbol es bastante más simple que el generado por ID3, ya que al aceptar valores continuos la clasificación utiliza intervalos abiertos que facilitan su lectura e interpretación. Vemos que "Eloisa Figueroa" tiene más de 13 caracteres, ningún punto, 15 o menos caracteres y 8 o menos consonantes, por lo que la instancia se etiquetará como **negativa** con una probabilidad de 27 a 1.

¿Elegirías este modelo o el generado por ID3? ¿Por qué?

El éxito en la clasificación es un 1% mayor respecto ID3 con $bins = 10$ en la discretización necesaria para aplicarlo y la diferencia asciende hasta casi un 18% si comparamos con el mismo algoritmo con $bins = 5$. La clara diferencia entre los dos algoritmos que muestran estos datos nos hace en efecto elegir **preferentemente a J48** frente a ID3.

¿Hemos encontrado la función exacta para generar las etiquetas? ¿Por qué lo sabes?

Si hubiésemos encontrado la función exacta de etiquetado, no tendríamos ese margen de error de aproximadamente un 2.5%, por lo que por el momento **no hemos encontrado dicha función**.

6 Utilizando más atributos con J48 (C4.5)

Es momento de volver a la pestaña de preproceso y generar un nuevo atributo que calcule el número de vocales. Después se grabará el conjunto de datos como `badges1-2`, y con él se construirá un clasificador con J48. Una vez generado, se anotan el porcentaje de instancias bien clasificadas y la matriz de confusión, tras lo cual visualizaremos el árbol generado.

Anota el porcentaje de instancias bien clasificadas y la matriz de confusión. ¿Qué indican los números que aparecen en las hojas del árbol?

El **100%** de las instancias se clasifican bien, por lo que la matriz de confusión tan solo tiene dos campos distintos de cero: el de instancias correctamente etiquetadas como negativas (en este caso, **144**) y el de instancias acertadamente clasificadas como positivas (sumando estas **150**).

El número que aparece en cada hoja del árbol es la cantidad de instancias pertenecientes a esa clase dentro de nuestro conjunto de entrenamiento.

¿Qué efecto tiene aumentar el valor de "Jitter" en la gráfica que relaciona el nuevo atributo con la clase?

Los puntos, inicialmente alineados y ordenados, se desagrupan formando una nube al distribuirse verticalmente. Es una forma de mostrar los datos coincidentes con mayor claridad.

¿Podrías decir cuál es el rango de vocales más común en el fichero proporcionado? ¿Se te ocurre algún otro atributo relacionado que pueda aportar información?

Los nombres con `4 vocales` son los más comunes, seguidos de aquellos con `5`. Observando la representación gráfica del campo, la cantidad de vocales parece seguir una distribución normal, y siguiendo los datos que nos aporta Weka decimos que tiene su media en 4.643 y una desviación estándar de 1.397.

En apartados anteriores comentábamos la significancia de los campos *longitud* y *consonantes*. Sin embargo, ahora entendemos que tan solo era un reflejo de que, a mayor número de letras, más posibilidades teníamos de encontrar una mayor cantidad de vocales en el nombre. Por lo tanto, y volviendo al dato del éxito total del nuevo clasificador, podemos decir que el único campo relevante a la hora de etiquetar nuevas entradas es el del número de vocales.

Tras todos estos resultados, ¿qué características o cualidades crees que deben tener los atributos para maximizar el éxito de los algoritmos de aprendizaje automático?

Dado que los algoritmos de aprendizaje automático trabajan comparando las distintas instancias a través de los atributos, debemos de dotarles de valores (ya sean numéricos o simbólicos) cuantificables en cuanto a número de repeticiones y/o proximidad.

7 Balanceado de datos, selección de características y otros filtros

¿Cuántos atributos de entrada tiene `adult-data`? ¿Cuántas instancias de entrenamiento?

El fichero consta de 32561 intancias de entrenamiento con 14 atributos de entrada, siendo la comparación respecto a 50K en el salario la clase de salida.

¿Qué resultados aparecen ejecutando ZeroR con validación cruzada? Explica el resultado.

ZeroR obtiene un **75.919% de éxito en la clasificación**. Esto nos indica que tenemos un conjunto de datos desequilibrado, i.e. con más ejemplos de instancias de una clase de la otra, lo que potencia el funcionamiento de ZeroR.

¿Qué resultados aparecen sólo con las instancias del fichero `adult-test.arff`? ¿Son estos resultados comparables a los anteriores? ¿Por qué?

En este caso la precisión del etiquetado es del **76.3774%. Sí, son comparables a los anteriores** y nos indican una tendencia global a pertenecer a la clase mayoritaria (en este caso, a recibir un salario menor a 50K).

¿Qué resultados aparecen repitiendo el proceso con J48? ¿Qué porcentaje de mejora ha obtenido respecto a los resultados del ZeroR?

Los resultados de J48 con validación cruzada ascienden al **86.2105%** de éxito en la clasificación, alcanzando un **85.8485%** con un conjunto independiente para test. Esto supone una **mejora del 10%** respecto a ZeroR. En cuanto a tiempo de modelado, ZeroR empleó 0.4 segundos en construir el modelo frente a los 4.14 que tardó J48. Aún siendo un incremento de un orden de magnitud en cuanto a tiempo para una mejora del 10%, hablando de tiempos que medimos en segundos consideramos aceptable el empeoramiento en la ejecución del cómputo del árbol.

¿Qué proporción de datos hay de cada clase? ¿Crees que este porcentaje es apropiado para que un algoritmo de aprendizaje automático aprenda bien?

En el archivo `adult-data`, un **75.919%** de las personas recopiladas reciben un salario inferior o igual a 50 K. Para aprovechar los algoritmos de aprendizaje automático al máximo es conveniente tener un conjunto de datos equilibrado, por lo que **no**, no creemos que sea apropiado.

¿Qué ocurre con el atributo de salida cuando se modifican las instancias de entrenamiento para tener un porcentaje similar entre las dos clases? ¿Ha descendido el número de ejemplos de entrenamiento?

Tras aplicar el filtro correspondiente, pasamos a clasificar 16299 entradas con un sueldo mayor a 50K y 16262 igual o menor a dicha cifra. No se trata de que tengamos menos ejemplos,

sino que se han repetido algunas de las de la clase minoritaria y descartado algunas de la clase mayoritaria. Aunque esto sesga la información original, la mejora en el rendimiento de los algoritmos de aprendizaje automático compensa este sesgo.

¿Qué resultados dan los algoritmos ZeroR y J48 repitiendo el procedimiento con las clases equilibradas? ¿Qué resultado crees que es mejor? ¿Por qué?

La siguiente tabla refleja los porcentajes de éxito en clasificación de las distintas combinaciones:

J48 supplied test	81.3279%
J48 cross validation	88.1269%
ZeroR supplied test	23.6226%
ZeroR cross validation	50.0568%

Sin duda el resultado de J48 es notablemente superior al de ZeroR. Esto es lógico dada la naturaleza de ambos algoritmos, y nos informa de que los atributos del archivo sí aportan información (si bien parcial o incompleta) para clasificar los datos. Al haber equilibrado las clases, ZeroR pierde toda efectividad alcanzando valores muy próximos al 50% (el mínimo posible en una clase binaria).

¿Qué atributos consideras inútiles para el algoritmo de aprendizaje y por qué? Elimina un par de ellos. ¿Qué es lo que ocurre al repetir la evaluación anterior?

Una vez equilibradas las clases, observamos en los gráficos de correlación que la mayoría de atributos no aportan mucha información por sí solos. Por ejemplo, la edad parece indicar que el grupo con mayor probabilidad de obtener un salario mayor está entre los 28 y los 70 mientras que el grupo con mayor probabilidad de conseguir un salario inferior se sitúa entre los 17 y los 65 años. Ahí encontramos un poco de información, pero es claro que nos hacen falta más datos además de la edad para evaluar la etiqueta.

El atributo que a priori parece menos efectivo o incluso perjudicial para el árbol es `fnlwgt`, pues tiene una gráfica muy equilibrada entre clases para un mismo valor y por tanto la variación en el árbol que pueda generar será mínima, perjudiciando al tiempo de ejecución.

`capital-gain` es otro atributo que merece mención. En su representación gráfica observamos que bastantes instancias tienen valores de bien 99999 o 0. Estos valores son potencialmente falsos ya que son los extremos y no añaden información sino que pueden generan ruido, a no ser que tengan algún tipo de valor simbólico fuera de nuestro entendimiento. `capital-loss` también sufre este problema pero mucho menos pronunciado, teniendo un conjunto de instancias bastante más amplia con valores significativos.

Tras probar otras combinaciones antes de llegar al razonamiento anterior, concluimos con la comparativa que muestra las diferencias en rendimiento tras la eliminación de los atributos mencionados. Se puede observar que los resultados son algo peores que el original, pero han reducido coste en tiempo de ejecución.

original	88.1269%
sin `fnlwgt`	86.9568%
sin `capital-gain`	86.2596%
sin ninguno de los dos	85.0588%

¿Qué resultados se obtienen aplicando el filtro de normalización para los atributos numéricos?

El filtro transforma los valores numéricos a otros únicamente comprendidos entre 0 y 1.

Después del procesamiento de datos que has realizado en este apartado, ¿crees que esto ayuda al proceso de aprendizaje? ¿Por qué? ¿Cuál es el mejor resultado obtenido? Justifícalo.

Ayuda porque ahora los valores muy grandes o muy pequeños no afectan tanto, ya que todos se encuentran en la misma escala. Con J48 y cross validation obtenemos un 82.39% de instancias correctamente clasificadas, un 6% menos aproximadamente que con los valores sin normalizar.

8 Problemas encontrados

Nuestro equipo lidió con los usuales problemas que uno se encuentra al trabajar con un entorno nuevo, como no encontrar alguna opción del menú o similar. Sin embargo, el guión era claro y suficientemente descriptivo como para no tener que acudir a ayuda ajena al equipo para solucionarlos.

También nos costó razonar sobre los atributos a eliminar, dado que el razonamiento de la eliminación de algunos atributos no tenía siempre las consecuencias que esperábamos.

9 Opinión personal sobre el tutorial

Consideramos que este tutorial ilustra claramente el cómo y el por qué del funcionamiento de los algoritmos para aprendizaje supervisado. Tras la finalización del mismo hemos adquirido nuevos e interesantes conocimientos con los que asentar lo estudiado en clase de forma teórica.