Análisis de Datos en WEKA – Pruebas de Selectividad

María García Jiménez
Ingeniería de Telecomunicación
Universidad Carlos III
100025080@alumnos.uc3m.es

Aránzazu Álvarez Sierra
Ingeniería de Telecomunicación
Universidad Carlos III
100025202@alumnos.uc3m.es

RESUMEN
En este trabajo vamos a utilizar la herramienta de minería de datos WEKA para analizar el contenido de un fichero .arff, que contiene las muestras correspondientes a 18802 alumnos presentados a las pruebas de selectividad y los resultados obtenidos en las pruebas.

Categorías y Descripción de la Asignatura
H.2.8 Database Applications [Database Management]: Data mining.

Términos Generales.
Algoritmos, diseño, experimentación, teoría.

Palabras Claves
Aprendizaje, algoritmo, modelo, predicción.

1. INTRODUCCIÓN
WEKA (*Waikato Environment for Knowledge Analysis)* es una herramienta que permite la experimentación de análisis de datos mediante la aplicación, análisis y evaluación de las técnicas más relevantes de análisis de datos, principalmente las provenientes del aprendizaje automático, sobre cualquier conjunto de datos del usuario.

El fichero de datos seleccionado contiene datos provenientes del campo de la enseñanza, correspondientes a alumnos que realizaron las pruebas de selectividad en los años 1993-2003 procedentes de diferentes centros de enseñanza secundaria de la comunidad de Madrid. Los datos de cada alumno contienen la siguiente información: año, convocatoria, localidad del centro, opción cursada (entre 5 posibles), calificaciones parciales obtenidas en lengua, historia, idioma y las tres asignaturas opcionales, así como la designación de las asignaturas de idioma y las 3 opcionales cursadas, calificación en el bachillerato, calificación final y si el alumno se presentó o no a la prueba.

Algunos de los análisis que podemos llevar a cabo con esta herramienta puede ser el relacionar los resultados obtenidos en las pruebas con las características o perfiles de los estudiantes, cuáles son las características comunes de aquellos alumnos que superan las pruebas, hay diferencias en los resultados obtenidos según las opción elegida, las localidades de las que proceden,…

2. DESARROLLO Y RESULTADOS
2.1 Preprocesado de los Datos
2.1.1 Filtros de Atributos
WEKA permite realizar manipulaciones sobre los datos aplicando filtros. Se pueden aplicar en dos niveles: atributos e instancias. Además las operaciones de filtrado pueden aplicarse en cascada, de forma que la entrada de cada filtro es la salida de haber aplicado el anterior filtro.

Vamos a aplicar sólo filtros no supervisados sobre atributos, donde las operaciones son independientes del algoritmo análisis. El resultado de estos filtros nos servirá de ayuda para el resto de aplicaciones de la herramienta.

De entre todos los filtros que hay implementados en esta sección, hemos decidido aplicar sobre nuestros datos los filtros "Remove" y "Discretize", que eliminan atributos y discretizan atributos numéricos, respectivamente.

- "Remove": vamos a proceder a eliminar los atributos correspondientes a las calificaciones parciales y la calificación final, quedando únicamente como calificaciones las notas de bachillerato y la de selectividad.

- "Discretize": Este filtro transforma los atributos numéricos seleccionados en atributos simbólicos, con una serie de etiquetas que resultan de dividir la amplitud total del atributo en intervalos. Por ejemplo, una vez aplicado el filtro anterior, si dividimos las calificaciones en 4 intervalos de igual frecuencia, obtenemos los rangos delimitados por (4, 4.8, 5.76). Podemos observar como el 75% de los alumnos alcanza la nota de compensación, el 50% está entre 4 y 5.755, y el 25% obtiene una nota a partir del 5.755.

2.2 Visualización
La herramienta de visualización de WEKA permite representar gráficas 2D que relacionan pares de atributos. Podemos visualizar en la figura 1 el rango de calificaciones finales de los alumnos entre 1993 y 2003, especificando como color para la gráfica la convocatoria de la prueba.

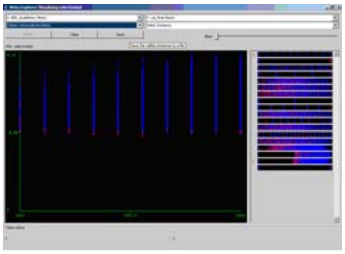


Figura1. Rango de calificaciones finales 1993-2003

Vemos que prácticamente no existen alumnos con nota inferior al 4.86, y que la mayoría de los alumnos que se presentan lo hacen en la convocatoria de Junio. Además también podemos observar que a medida que se van realizando más pruebas, el número de notas próximas al 9.72 va siendo cada vez mayor.

También podemos visualizar en la figura 2 dos variables muy relacionadas entre sí: la calificación de la prueba y la nota de bachillerato, eligiendo como color una vez más la convocatoria.

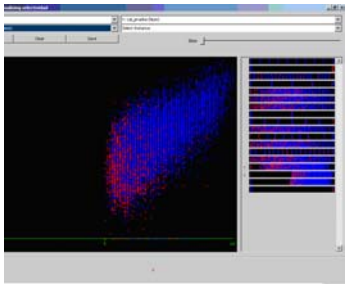


Figura 2. Relación calificación de la prueba y nota de bachillerato

Vemos que la relación no es totalmente directa, aunque presenta una cierta tendencia creciente: si la nota obtenida durante el bachillerato es elevada, la nota en la prueba de selectividad tiende a serlo también.

Si lo que queremos visualizar son atributos simbólicos, sus posibles valores se representan a lo largo del eje. Puesto que aquellas instancias que comparten cada valor de un atributo simbólico serían un único punto, lo que se hace es introducir un desplazamiento aleatorio (ruido), mediante el botón Jitter. Como ejemplo, vamos a relacionar la nota de selectividad con la localidad de origen, especificando como color la nota de bachillerato. Previamente vamos a realizar una discretización de las calificaciones en intervalos de amplitud 2. El resultado lo podemos visualizar en la figura 3.

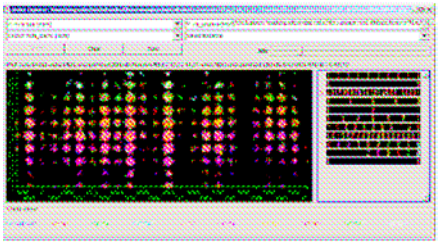


Figura 3. Relación nota de selectividad y localidad

2.3 Asociación

Mediante algoritmos de asociación podemos realizar la búsqueda automática de reglas que relacionan conjuntos de atributos entre sí. Son algoritmos no supervisados, ya que no existen relaciones conocidas a priori con las que contrastar la validez de los resultados, sino que se evalúa si esas reglas son estadísticamente significativas. El principal algoritmo implementado en WEKA es el algoritmo *"Apriori"*, el cual sólo busca reglas entre atributos simbólicos, por lo cual todos los atributos numéricos deberían ser discretizados previamente. A modo de ejemplo vamos a discretizar todos los atributos numéricos en 4 intervalos de igual frecuencia. Si aplicamos el algoritmo de asociación con los parámetros por defecto, nos aparecen una serie de reglas que relacionan las asignaturas y las opciones, suspensos en la prueba y en la calificación final, etc. Podemos visualizar de forma gráfica en la figura 4 como la gran mayoría de alumnos que se presentan a la prueba de idioma, han elegido inglés como opción.

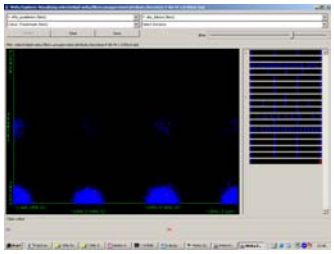


Figura 4. Relación idioma y año académico

Si lo que pretendemos es buscar relaciones no conocidas, lo que podemos hacer es aplicar un filtro para eliminar todos los descriptores de asignaturas y calificaciones parciales, quedando tan sólo los siguientes atributos: año académico, convocatoria, localidad, opción 1ª, calificación de la prueba y nota de bachillerato. Las reglas más significativas que nos proporciona la herramienta son que el 99% de los alumnos que obtienen una nota de 8 o superior, se presentaron en la convocatoria de Junio o que el 95% de los alumnos que obtuvieron en la prueba una calificación en el intervalo (5.772 – 7.696) y que además tenían una nota de bachillerato entre 6 y 8, también se presentaron en la convocatoria de Junio. Cabe destacar que si realizamos las visualizaciones en 2D de los atributos restantes, no existe ninguna relación importante entre las calificaciones, la localidad y el año de convocatoria.

Si volvemos a aplicar un filtro para eliminar de los 6 atributos anteriores las dos últimas calificaciones, y añadimos la calificación final discretizándola en 2 intervalos de igual frecuencia, llegamos a la conclusión de nuevo que las reglas más significativas relacionan la convocatoria con la calificación, pero en este caso debemos tener en cuenta otros factores: opción 1ª y localidad. Al entrar en juego todos estos atributos, la precisión de las reglas desciende. Por ejemplo, aquellos alumnos cuya localidad de origen es Leganés y su calificación final está en el rango (5.685 – inf), el 92% de los mismos se presentó en la convocatoria de Junio.

2.4 Agrupamiento

Los algoritmos de agrupamiento buscan grupos de instancias con características similares, según un criterio de comparación entre valores de atributos de las instancias definidos en los algoritmos.

2.4.1 Agrupamiento Numérico

❖ **Algoritmo K-Medias:** Se trata de un algoritmo clasificado como Método de Particionado y Recolocación. Este método es hasta ahora el más utilizado en aplicaciones científicas e industriales. El nombre le viene porque representa cada uno de los clusters por la media (o media ponderada) de sus puntos, es decir, por su centroide. Este método únicamente se puede aplicar a atributos numéricos, y los *outliers* le pueden afectar muy negativamente. Sin embargo, la representación mediante centroides tiene la ventaja de que tiene un significado gráfico y estadístico inmediato. La suma de las discrepancias entre un punto y su centroide, expresado a través de la distancia apropiada, se usa como función objetivo. La función objetivo, suma de los cuadrados de los errores entre los puntos y sus centroides respectivos, es igual a la varianza total dentro del propio cluster. La suma de los cuadrados de los errores se puede racionalizar, como el negativo del log-likelihood, para modelos mixtos que utilicen distribuciones normales.

Vamos a aplicar el algoritmo de agrupamiento K-medias, por ser uno de los más veloces y eficientes, aunque también hay que decir que es uno de los más limitados. Este algoritmo precisa únicamente del número de categorías similares en las que queremos dividir el conjunto de datos.

Vamos a comprobar si el atributo "opción 1ª" divide naturalmente a los alumnos en grupos similares, para lo cual seleccionamos el algoritmo SimpleKMeans con un número de clusters igual a 5. Nos aparecen los 5 grupos de ejemplos más similares, y sus centroides: promedios para atributos numéricos y valores más repetidos en cada grupo para atributos simbólicos. Por ejemplo, para 3 de los 5 clusters, la localidad más repetida es Leganés y para los otros 2, es Getafe. En todos los clusters, la convocatoria más repetida es la de Junio. El número de instancias agrupadas en cada cluster es:

Clustered Instances

0	2474 (13%)
1	6030 (32%)
2	5236 (28%)
3	2475 (13%)
4	2587 (14%)

También resulta de gran interés el analizar gráficamente cómo se distribuyen diferentes valores de los atributos en los clusters generados. Podemos seleccionar a modo de ejemplo las combinaciones del atributo "opción 1ª" con "localidad", "cal_final", "convocatoria" y "año_académico". En las figuras 5, 6, 7 y 8 podemos observar el resultado.

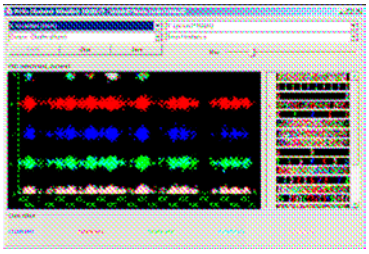


Figura 5. Relación localidad y opción 1ª

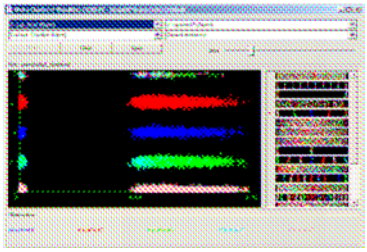


Figura 6. Relación calificación final y opción 1ª

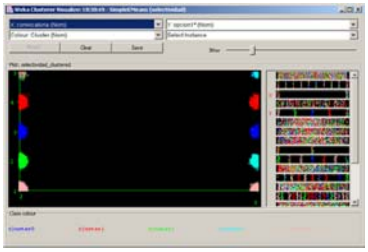


Figura 7. Relación convocatoria y opción 1ª

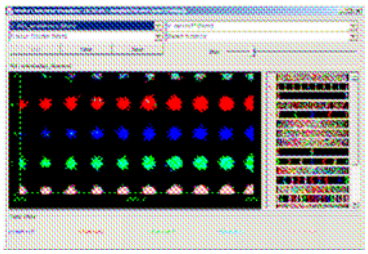


Figura 8. Relación año académico y opción 1ª

Viendo el resultado obtenido, podemos concluir que el parecido entre casos viene dado fundamentalmente por las opciones seleccionadas. Los clusters 0, 1 y 4 se corresponden con las opciones 3, 4 y 1, respectivamente. Los clusters 2 y 3 representan la opción 3 en las convocatorias de Junio y Septiembre.

Vamos a realizar un análisis más exhaustivo particularizando a las dos localidades mayores: Leganés y Getafe. En primer lugar preparamos los datos aplicando los filtros necesarios para quedarnos únicamente con los atributos "localidad", "opción 1ª" y "cal_final". Además discretizamos las calificaciones en dos grupos de la misma frecuencia, y nos quedamos sólo con los estudiantes de esas dos localidades. A continuación, aplicamos el algoritmo K-Medias para 4 grupos.

```
Cluster centroids:

Cluster 0
        Mean/Mode:  GETAFE 2 '(-inf-5.685]'
        Std Devs:   N/A   N/A   N/A
Cluster 1
        Mean/Mode:  LEGANES 4 '(-inf-5.685]'
        Std Devs:   N/A   N/A   N/A
Cluster 2
        Mean/Mode:  LEGANES 1 '(5.685-inf)'
        Std Devs:   N/A   N/A   N/A
Cluster 3
        Mean/Mode:  GETAFE 4 '(5.685-inf)'
        Std Devs:   N/A   N/A   N/A
```

Vemos que existen buenos alumnos en Leganés para la opción 1 y en Getafe para la opción 4.

❖ **Algoritmo EM:** El algoritmo EM asigna a cada instancia una distribución de probabilidad de pertenencia a cada cluster. El algoritmo puede decidir cuántos clusters crear basado en validación cruzada o se le puede especificar a priori cuantos debe generar. Utiliza el modelo Gaussiano finito de mezclas, asumiendo que todos los atributos son variables aleatorias independientes.

Este algoritmo es bastante más elaborado que el K-Medias, ya que requiere muchas más operaciones. Debido a esta mayor complejidad, lo que vamos a hacer en primer lugar es reducir el número de instancias a 500 (con filtro de instancias Resample: 3%). Este algoritmo permite buscar el número de grupo más apropiado. Una vez aplicado, se nos muestra que el número de clusters significativos en la muestra de los 500 alumnos es de 5. Vamos a ver cuál ha sido el resultado del agrupamiento sobre diferentes combinaciones de atributos: "opción 1ª" con "localidad" y "cal_final".

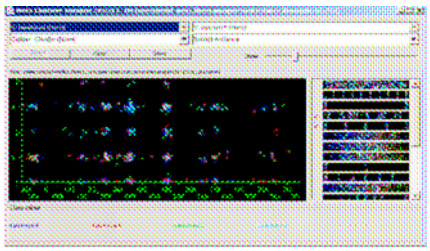


Figura 9. Relación localidad y opción 1ª

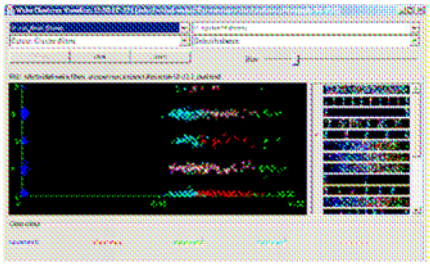


Figura 10. Relación calificación final y opción 1ª

Para este segundo algoritmo de agrupamiento por criterios estadísticos y no mediante distancias entre vectores de atributos, predomina el agrupamiento de los alumnos por tramos de calificaciones, independientemente de la opción elegida. En el algoritmo anterior veíamos que predominaba más el perfil de las asignaturas que las calificaciones.

2.4.2 Agrupamiento Simbólico

El algoritmo simbólico tiene la ventaja sobre los anteriores de realizar un análisis cualitativo que construye categorías jerárquicas para organizar los datos. Estas categorías se forman con un criterio probabilístico de "utilidad", llegando a las que permiten homogeneidad de los valores de los atributos dentro de cada una y al mismo tiempo una separación entre categorías dadas por los atributos, propagándose estas características en un árbol de conceptos.

❖ **Algoritmo COBWEB:** Se trata de un algoritmo de *clustering jerárquico*. Se caracteriza porque utiliza aprendizaje incremental, esto es, realiza las agrupaciones instancia a instancia. Durante la ejecución del algoritmo se forma un árbol (*árbol de clasificación*) donde las hojas representan los segmentos y el nodo raíz engloba por completo el conjunto de datos de entrada. Al principio, el árbol consiste en un único nodo raíz. Las instancias se van añadiendo una a una y el árbol se va actualizando en cada paso. La actualización consiste en encontrar el mejor sitio donde incluir la nueva instancia, operación que puede necesitar de la reestructuración de todo el árbol (incluyendo la generación de un nuevo nodo *anfitrión* para la instancia y/o la fusión/partición de nodos existentes) o simplemente la inclusión de la instancia en un nodo que ya existía. La clave para saber cómo y dónde se debe actualizar el árbol la proporciona una medida denominada *utilidad de categoría*, que mide la calidad general de una partición de instancias en un segmento. La reestructuración que mayor utilidad de categoría proporcione es la que se adopta en ese paso. El algoritmo es muy sensible a otros dos parámetros:

a) Acuity: Este parámetro es muy necesario, ya que la utilidad de categoría se basa en una estimación de la media y la desviación estándar del valor de los atributos, pero cuando se estima la desviación estándar del valor de un atributo para un nodo en particular, el resultado es cero si dicho nodo sólo contiene una instancia. Así pues, el parámetro acuity representa la medida de error de un nodo con una sola instancia, es decir, establece la varianza mínima de un atributo.

b) Cut-off: Este valor se utiliza para evitar el crecimiento desmesurado del número de segmentos. Indica el grado de mejoría que se debe producir en la utilidad de categoría para que la instancia sea tenida en cuenta de manera individual. En otras palabras: cuando no es suficiente el incremento de la utilidad de categoría en el momento en el que se añade un nuevo nodo, ese nodo se corta, conteniendo la instancia otro nodo ya existente.

Si aplicamos este algoritmo con los parámetros por defecto sobre la muestra reducida de instancias, el árbol generado contiene 800 nodos. Para poder obtener un árbol más manejable, podemos modificar el parámetro cut-off (0.45). El resultado generado por la herramienta WEKA es el siguiente:

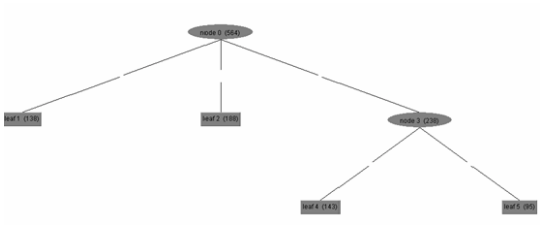
```
Number of merges: 36
Number of splits: 34
Number of clusters: 6

node 0 [564]
|  leaf 1 [138]
node 0 [564]
|  leaf 2 [188]
node 0 [564]
|  node 3 [238]
|  |  leaf 4 [143]
|  node 3 [238]
|  |  leaf 5 [95]

Clustered Instances

1     138 (24%)
2     188 (33%)
4     143 (25%)
5      95 (17%)
```

Vemos que hay tres grupos en un primer nivel, y el tercero se subdivide en otros dos. Podemos visualizar de forma gráfica cuál es el aspecto del árbol generado:



Por último, vemos cómo quedan distribuidas las instancias por clusters, en la figura 11:

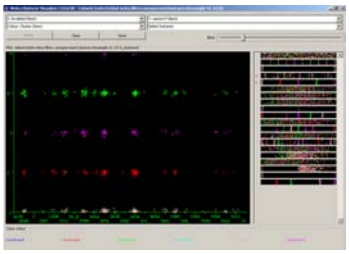


Figura 11. Relación localidad y opción 1ª

De nuevo vuelve a pesar la opción como criterio de agrupamiento. Los nodos hoja 1, 2, 4 y 5 se corresponden con las opciones 2, 4, 1 y 3, respectivamente. En un primer nivel hay tres grupos, uno para la opción 2, otro para la opción 4 y otro que une las opciones 1 y 3.

2.5 Clasificación

El problema de la clasificación es el más frecuente en la práctica. Una vez aplicados los algoritmos no supervisados de agrupamiento y asociación se aplicaría la clasificación como un refinamiento en el análisis.

De esta forma, construiremos un modelo que permita predecir la categoría de las instancias en función de una serie de atributos de entrada.

La clase se convertirá en la variable objetivo a predecir.

2.5.1 Modos de Evaluación del Clasificador

El resultado de aplicar el algoritmo de clasificación se efectúa comparando la clase predicha con la clase real de las instancias.

Existen diversos modos de realizar la evaluación:

- **Use training set**: evaluación del clasificador sobre el mismo conjunto sobre el que se construye el modelo predictivo para determinar el error, que en este caso se denomina "error de resustitución".
- **Supplied test set**: esta opción evalúa sobre un conjunto independiente. Permite cargar un conjunto nuevo de datos. Sobre cada dato se puede realizará una predicción de clase para contar los errores.
- **Cross-Validation**: evaluación con validación cruzada. Se dividirán las instancias en tantas carpetas como indica el parámetro 'Folds', y en cada evaluación se toman las instancias de cada carpeta como datos de test, y el resto como datos de entrenamiento para construir el modelo. Los errores calculados serán el promedio de todas las ejecuciones
- **Percentage split**: se dividen los datos en dos grupos, de acuerdo con el porcentaje indicado (%). El valor indicado es el porcentaje de instancias par construir el modelo, que seguidamente es evaluado sobre las que se han dejado aparte.

Además, utilizaremos otras opciones adicionales:

- **Output model**: Se visualiza el modelo construido por el clasificador.
- **Output per-class stats**: Se obtienen estadísticas de los errores de clasificación por cada uno de los valores que toma el atributo de clase
- **Output entropy evaluation measures**: se generan medidas de evaluación de entropía.
- **Store predictions for visualization**: Analiza los errores de clasificación.

2.5.1.1 Evaluación del clasificador en ventana de texto.

Si queremos predecir el atributo "presentado", con un árbol de decisión de tipo J48, obtenemos:

```
J48 pruned tree
------------------
cal_prueba <= 0: NO (153.0)
cal_prueba > 0: SI (18649.0/2.0)
Number of Leaves : 2
Size of the tree : 3
```

Obtenemos una relación trivial: excepto dos casos de error, los presentados son los que obtienen una calificación superior a 0.

Si estudiamos la matriz de confusión:

```
=== Confusion Matrix ===

   a     b   <-- classified as
18647    0 |   a = SI
    2  153 |   b = NO
```

Podemos observar que los valores de la diagonal son los aciertos, y el resto los errores. De los 18647 alumnos presentados, todos son correctamente clasificados. De los 155 alumnos no presentados, hay 153 correctamente clasificados y hay 2 con error.

2.5.2. Selección y configuración de clasificadores

Vamos a aplicar algoritmos de clasificación a diferentes problemas de predicción de atributos definidos sobre los datos de entrada.

Analizaremos la predicción de la calificación en la prueba a partir de los atributos siguientes: año, convocatoria, localidad, opción, presentado y nota de bachillerato.

2.5.2.1 Clasificador "OneR"

Es un clasificador de los más sencillos y rápidos. Sus resultados pueden ser muy buenos en comparación con algoritmos mucho más complejos. Selecciona el atributo que mejor "explica" la clase de salida.

Si lo aplicamos al problema de predicción de aprobados en la prueba a partir de los atributos de entrada, obtenemos:

=== Classifier model (full training set) ===

nota_bachi:
'(-inf-6.35]' -> '(-inf-4.795]'
'(6.35-inf)' -> '(4.795-inf)'
(13446/18802 instances correct)

Time taken to build model: 0.03 seconds

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances	**13446**	**71.513%**
Incorrectly Classified Instances	**5356**	**28.486%**
Kappa statistic	**0.4302**	
Mean absolute error	**0.2849**	
Root mean squared error	**0.5337**	
Relative absolute error		**56.9728 %**
Root relative squared error		**106.7453 %**
Total Number of Instances	**18802**	

Si analizamos los resultados, vemos que la mejor predicción posible con un solo atributo es la nota de bachillerato. Su umbral está fijado en 6.35. La tasa de aciertos sobre el propio conjunto de entrenamiento es del 71.51%.

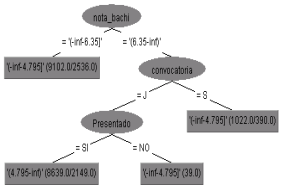
2.5.2.2 Clasificador J48

El algoritmo J48 de WEKA es una implementación del algoritmo C4.5, uno de los algoritmos de minería de datos más utilizado.

Se trata de un refinamiento del modelo generado con OneR. Supone una mejora moderada en las prestaciones, y podrá conseguir una probabilidad de acierto ligeramente superior al del anterior clasificador.

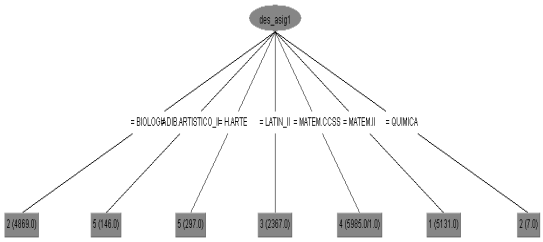
El parámetro más importante que deberemos tener en cuenta es el factor de confianza para la poda "confidence level", que influye en el tamaño y capacidad de predicción del árbol construido. Para cada operación de poda, define la probabilidad de error que se permite a la hipótesis de que el empeoramiento debido a esta operación es significativo. A probabilidad menor, se exigirá que la diferencia en los errores de predicción antes y después de podar sea más significativa para no podar. El valor por defecto es del 25%. Según baje este valor, se permiten más operaciones de poda.

Para nuestro estudio construiremos un árbol de decisión con un valor del factor de confianza para la poda del 10%:



Podemos ver cómo los atributos más importantes son la calificación de bachillerato y la convocatoria.

Otro análisis a realizar es la clasificación formulada sobre cualquier atributo de interés, por ejemplo, la predicción de la opción.



Se desvela una relación trivial entre opción y asignaturas en las opciones que se predice con prácticamente el 100% de los casos.

Si lo que predecimos es la localidad y la opción, reducimos el número de atributos a tres: localidad, opción y calificación, y buscamos un modelo de clasificación para cada uno de los atributos:

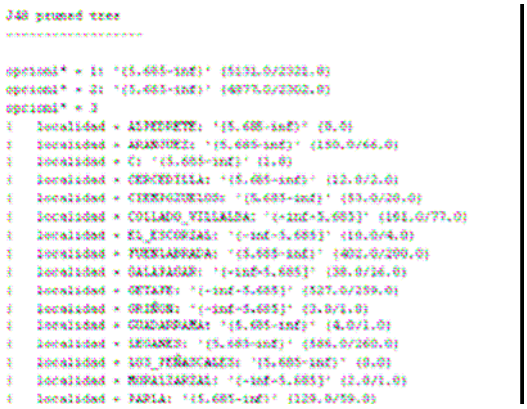
```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

opcion1* = 1: LEGANES (5131.0/3745.0)
opcion1* = 2: LEGANES (4077.0/3429.0)
opcion1* = 3
|   cal_final = '(-inf-5.685]': GETAFE (1134.0/866.0)
|   cal_final = '(5.685-inf)': LEGANES (1233.0/907.0)
opcion1* = 4
|   cal_final = '(-inf-5.685]': GETAFE (3390.0/2546.0)
|   cal_final = '(5.685-inf)': LEGANES (2594.0/1906.0)
opcion1* = 5
|   cal_final = '(-inf-5.685]': GETAFE (230.0/149.0)
|   cal_final = '(5.685-inf)': LEGANES (185.0/110.0)

Number of Leaves  :     8

Size of the tree :     12
```



Las opciones que predominan en Leganés son la 1 y la 2. Las opciones 3 y 4 son las que aparecen con mayor frecuencia entre los alumnos que aprobaron la prueba en Leganés.

LOCALIDAD

=== Summary ===		
Correctly Classified Instances	**5884**	**31.2945 %**
Incorrectly Classified Instances	**12918**	**68.705%**
Kappa statistic	**0.0899**	
Mean absolute error	**0.0679**	
Root mean squared error	**0.1842**	
Relative absolute error		**96.6808 %**
Root relative squared error		**98.335 %**
Total Number of Instances	**18802**	

OPCIÓN

=== Summary ===		
Correctly Classified Instances	**6714**	**35.709 %**
Incorrectly Classified Instances	**12088**	**64.291 %**
Kappa statistic	**0.0777**	
Mean absolute error	**0.2896**	
Root mean squared error	**0.3806**	
Relative absolute error		**97.7784 %**
Root relative squared error		**98.884 %**
Total Number of Instances	**18802**	

Pero podemos comprobar cómo hay una alta tasa de error (alrededor del 30%), que cuestionan la validez de estos modelos.

2.5.3 Predicción Numérica

Se trata de un caso particular de clasificación, en el que la clase es un valor numérico.

En los siguientes ejemplos de algoritmos de predicción numérica vamos a encontrar relaciones deterministas entre variables conocidas y buscar otros modelos de mayor interés.

Relación entre calificación final y parciales.

Utilizamos el modelo de predicción de regresión simple, que construye un modelo lineal del atributo clase a partir de los atributos de entrada.

cal_prueba =

0.1674 * nota_Lengua +
0.1673 * nota_Historia +
0.1668 * nota_Idioma +
0.1897 * calif_asig1 +
0.1896 * calif_asig2 +
0.1193 * calif_asig3 +
0.0084

Time taken to build model: 0.41 seconds

=== Evaluation on training set ===
=== Summary ===

Correlation coefficient	**0.9984**
Mean absolute error	**0.0524**
Root mean squared error	**0.0781**
Relative absolute error	**4.8231 %**
Root relative squared error	**5.6362 %**
Total Number of Instances	**18802**

El resultado es la relación con los pesos relativos en cada una de las pruebas parciales sobre la calificación de la prueba.

Aparece en este caso el coeficiente de correlación y los errores medio y medio cuadrático, en términos absolutos y relativos. Existe una precisión muy alta, ya que el coeficiente de correlación des de 0.9984.

Si queremos estudiar ahora qué peso lleva la calificación de bachillerato y de la prueba en la nota final, aplicamos este modelo de regresión lineal a la relación entre calificación final y nota de bachillerato:

Linear Regression Model

cal_final =

1.8747 * cal_prueba +
-4.3253

Time taken to build model: 0 seconds

=== Evaluation on training set ===
=== Summary ===

Correlation coefficient	**0.8746**
Mean absolute error	**1.2189**
Root mean squared error	**1.4675**
Relative absolute error	**46.2215 %**
Root relative squared error	**48.4917 %**
Total Number of Instances	**40**

Se ha aplicado este ejemplo a una muestra con un menor número de instancias. Para ello hemos elegido únicamente a los alumnos de la población de Guadarrama. Los resultados no son nada buenos, pues la relación es no lineal.

Para solucionarlo, aplicamos el **algoritmo MP5**, que lleva a cabo una regresión por tramos, con cada tramo determinado a partir de un árbol de regresión.

```
Linear Regression Model

cal_final =

      0.3996 * cal_prueba +
      0.5997 * nota_bachi +
      0.0046
```

Vemos que la relación obtenida en la seguida en la actualidad: 40% nota de la prueba y 60% nota de bachillerato.

Correlación entre nota de bachillerato y calificación en prueba.
Si se construye un modelo entre dos variables, se puede estudiar el grado de correlación entre ellas.

Vamos a analizar las diferencias en las opciones 1 y 4. Teniendo en cuenta los atributos 'calificación en prueba' y 'nota de bachillerato', vemos los resultados del análisis de la correlación para cada caso:

- opción 1ª

```
Linear Regression Model

cal_prueba =

      0.922  * nota_bachi +
     -1.9479

Time taken to build model: 0.05 seconds

=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.6688
Mean absolute error                      0.8739
Root mean squared error                  1.1148
Relative absolute error                 73.8833 %
Root relative squared error             75.0895 %
Total Number of Instances             1745
```

- opción 4ª

```
Linear Regression Model

cal_prueba =

      0.8434 * nota_bachi +
     -0.842

Time taken to build model: 0.59 seconds

=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.586
Mean absolute error                      0.7888
Root mean squared error                  1.0441
Relative absolute error                 79.3249 %
Root relative squared error             80.8973 %
Total Number of Instances             2035
```

El grado de relación entre las variables es diferente. Los alumnos de la opción 1ª tienen una relación más lineal entre ambas calificaciones que los alumnos de la opción 4ª.

3. EVALUACIÓN

Como evaluación de todos los resultados obtenidos, podemos concluir con varias afirmaciones:
- La mayoría de los alumnos que se presentan lo hacen en Junio.
- Si la nota obtenida durante el bachillerato es alta, la nota de la prueba de selectividad tiende a serlo también.
- El inglés es la opción que eligen la mayoría de alumnos en la prueba de idiomas.
- En Leganés hay buenos alumnos en la opción 1ª, y en Getafe en la opción 4ª, que puede ser debido al impacto de la Universidad en la zona.

-Se obtiene la relación trivial de que los presentados son los que obtienen una calificación superior a 0.
-Obtención de la relación de las pruebas parciales con sus pesos relativos sobre la calificación total de la prueba.
- El peso que tiene la nota de bachillerato y la nota de la prueba sobre la nota final son del 60% y del 40% respectivamente.

4. TRABAJOS FUTUROS

4.1 Aprendizaje del Modelo y Aplicación a Nuevos Datos

Como futuros estudios a realizar, nos parece interesante la posibilidad que ofrece WEKA de construir y evaluar un clasificador de forma cruzada con dos ficheros de datos.

Un posible análisis de este tipo sería la generación con el filtro de instancias de dos conjuntos de datos correspondientes a los alumnos de dos poblaciones distintas.

Seguidamente generaríamos los modelos de clasificación de alumnos con buen y mal resultado en la prueba con el fichero de alumnos de una de estas localidades, para evaluarlos con los alumnos de la otra localidad elegida. Para ello se deberá utilizar la opción de evaluar con un fichero de datos independientes, "Supplied test set".

Se podría hacer este procedimiento pero a la inversa, entrenar con los datos de la segunda población elegida y evaluar con los de la primera población.

Tendríamos de esta forma los dos modelos generados para ambos conjuntos de datos, que podríamos comparar y obtener diferencias entre ellos.

4.2 Selección de Atributos

Otro posible trabajo a realizar sería automatizar la búsqueda de atributos más apropiada para "explicar" un atributo objetivo, en un sentido de clasificación supervisada.

Así, podríamos explorar qué subconjuntos de atributos son los que mejor pueden clasificar la clase de la instancia.

Esta selección supervisada tendría dos componentes:

- Método de evaluación ("Atribute Evaluator"): función que determina la calidad del conjunto de atributos para discriminar la clase.

Podemos distinguir dos tipos de métodos de evaluación.

Uno de ellos son los que directamente utilizan un clasificador específico para medir la calidad del subconjunto de atributos a través de la tasa de error del clasificador. Se denomina métodos "wrapper", ya que "envuelven" al clasificador para explorar la mejor selección de atributos que optimiza sus prestaciones. Necesitan un proceso completo de entrenamiento y

evaluación en cada caso de búsqueda, por eso son muy costosos.

El otro tipo no utiliza este clasificador específico. Dentro de este tipo está el método "CfsSubsetEval", que calcula la correlación de la clase con cada atributo, y elimina atributos que tienen una correlación muy alta como atributos redundantes.

- Método de búsqueda ("Search Method"): forma de realizar la búsqueda de conjuntos. Si se quiere realizar la evaluación exhaustiva de todos los subconjuntos, aparece un problema combinatorio inabordable en cuanto crece el número de atributos. Para ello aparecen estas estrategias que permiten realizar la búsqueda de una forma más eficiente.

Uno de estos métodos, que se caracteriza por su rapidez, es el "ForwardSelection". Se trata de un método de búsqueda subóptima en escalada. El procedimiento es el siguiente: se elije primero el mejor atributo, después añade el siguiente atributo que más aporta y continúa así hasta llegar a la situación en la que añadir un nuevo atributo empeora la situación. Otro de este tipo de métodos sería el "BestSearch", que nos permitiría buscar interacciones entre atributos más complejas. Su procedimiento es ir analizando lo que mejora y empeora un grupo de atributos al añadir elementos, con la posibilidad de hacer retrocesos para explorar con más detalle. Otro de estos métodos que podríamos utilizar es el "ExhaustiveSearch", que enumera todas las posibilidades y las evalúa para seleccionar la mejor.

Deberemos, en la configuración del problema, seleccionar qué atributo se utiliza para la selección supervisada, y determinar si la evaluación se realizaría con todas las instancias disponibles, o mediante la validación cruzada.

En nuestro caso, elegiríamos los algoritmos más eficientes de evaluación y búsqueda, CsfSubsetEval y ForwardSelection.

De esta forma podremos estudiar los atributos que mejor explican otros dados, llegando a relaciones triviales u a otras no conocidas. Las confirmaríamos con la figura de mérito en cada caso, y corroboraríamos si son más o menos fiables.

5. CONCLUSIONES

En este trabajo se ha podido demostrar la gran utilidad que tiene la minería de datos al aplicarla a un caso real.

Hemos experimentado lo sencillo que es mediante WEKA el análisis y estudio estos datos, y su posterior interpretación. Hemos decidido utilizar todas las posibilidades que nos ofrece esta herramienta para hacer un estudio más completo.

El preprocesado, la clasificación, el agrupamiento, la asociación y la visualización previos de los datos de entrada nos han permitido obtener, con más facilidad, mejores resultados.

Debemos comentar también la gran diversidad de algoritmos incluidos en WEKA que se pueden utilizar según queramos obtener unos u otros objetivos.

Todo ello hace que WEKA sea una herramienta principal en las cada vez más importantes tecnologías basadas en el procesamiento de información en los distintos ámbitos de la sociedad.

6. REFERENCIAS

[1] Alonso, C.WEKA: Waitako Environment for Knowledge Analysis. Introducción básica. Departamento de Informática Universidad de Valladolid. http://www.infor.uva.es/~calonso/IAII/Aprendizaje/weka/IntroduccionWeka.pdf

[2] Ferri, C. Mi página de Weka. http://www.dsic.upv.es/~cferri/weka/

[3] García, D. Weka Tutorial (Spanish) http://metaemotion.com/diego.garcia.morate/

[4] Hernández, J. y Ferri, C. Introducción al Weka. Curso de Doctorado Extracción Automática de Conocimiento en Bases de Datos e Ingeniería del Software. Universitat Politècnica de València, Marzo 2006.

[5] Villena, J. Apuntes de la asignatura Inteligencia en redes de Comunicaciones. 5º Ingeniería de Telecomunicación.

[6] Witten, I. y Frank, E. WEKA.Machine Learning Algorithms in Java. Department of Computer Science. University of Waikato**,** Hamilton, New Zealand.

[7] Weka Documentation. The University of Waikato. http://www.cs.waikato.ac.nz/ml/weka/