Technical Report
EHU-KZAA-TR-1-2010

UNIVERSITY OF THE BASQUE COUNTRY
Department of Computer Science and Artificial Intelligence

Manual de prácticas de minería de datos usando el software WEKA

Abdelmalik Moujahid e Iñaki Inza

January 2010

San Sebastian, Spain
http://www.ccia-kzaa.ehu.es/

Índice general

Capítulo 1

Introducción a Weka Explorer

Weka (Waikato Environment for Knowledge Analysis) es una herramienta visual de libre distribución (licencia GNU) desarrollada por los investigadores de la Universidad Waikato en Nueva Zelanda. Es un software que define 4 entornos para el análisis y extracción de conocimiento a partir de datos:

- Simple CLI: entorno consola para acceder directamente con java a los paquetes de Weka.
- Explorer: entorno visual que ofrece una interfaz gráfica para el uso de paquetes.
- Experimenter: entorno que permite configurar un conjunto de experimentos completos y complejos de análisis mediante distintos métodos de tratamiento y sobre distintos ficheros de datos.
- KnowledgeFlow: permite generar proyectos de minería de datos mediante la generación de diagramas de flujos de información.

1.1. El entorno Explorer

El explorador de Weka se compone de varias ventanas dentro de un orden lógico correspondiente a las diferentes etapas de extracción:

- Preparación de datos;
- Algoritmos de Data Mining para la construcción de modelos: clasificación, clustering, reglas de asociación, selección de atributos;
- Visualización de datos y del modelo.

1.2. Selección y preparación de datos

Weka utiliza un formato específico de datos, el formato arff (Attribute Relation File Format). Un fichero con este formato no sólo contiene los datos que servirán de aprendizaje, sino que incluye meta-información sobre los propios datos, como por ejemplo, el nombre y tipo de cada atributo, así como una descripción textual del origen de los datos. Por defecto, algunos ficheros arff son disponible en el directorio data de Weka.

Para poder utilizar datos provenientes de otros formatos, es necesario transformar las fuentes de datos en formato arff. Podemos convertir ficheros en texto conteniendo un registro por línea y con los atributos separados con comas (formato csv) a ficheros arff.

Aparte de los ficheros que vienen por defecto con Weka, dispones de un directorio disponible en el Repositorio UCI (University California Irvine) (http://archive.ics.uci.edu/ml/datasets.html) con todas las bases de datos que utilizan los investigadores en Data Mining.

Para la fase de limpieza y transformación de los datos, Weka nos proporciona una herramienta completa. Dentro del entorno *Preprocess*, la sección *Filter* incluye una amplia variedad de algoritmos, tanto para aprendizaje supervisado como no supervisado, que permiten preparar y mejorar la

calidad de los datos. En efecto, la calidad del conocimiento descubierto no sólo depende del algoritmo de aprendizaje utilizado, sino también de la calidad de los datos. Muchas veces la existencia de **atributos irrelevantes** afecta de manera notable la precisión del algoritmo de aprendizaje. Aunque existen algoritmos, como los árboles de decisión, a los cuales no les afecta de manera grave la presencia de atributos irrelevantes, ya que en el propio mecanismo de aprendizaje realiza una selección de atributos por su relevancia.

Otra tarea de preparación de los datos es la construcción de atributos, la cual consiste en crear automáticamente nuevos atributos aplicando alguna operación o función de los atributos originales con objeto de que estos nuevos atributos hagan más fácil y preciso el proceso de aprendizaje. Por ejemplo, el precio de las viviendas de una zona se puede estimar mucho mejor a partir de la densidad de población de la zona que de la población absoluta y de su superficie.

Para aplicar un filtro sobre los datos, se elige uno en la lista que proporciona Weka (Preprocess/Filter/choose). Para acceder a los parámetros de configuración del filtro elegido, haz un clic sobre su nombre. Una vez elegido un filtro, es necesario aplicarlo al conjunto de datos (botón apply). Siempre podemos cancelar las modificaciones (botón Undo).

Selección de atributos: La selección de atributos (con el fin de centrar el aprendizaje sobre algunos atributos pertinentes en vez de utilizar todos los atributos) no requiere necesariamente la utilización de filtros. Podemos simplemente activar/desactivar las casillas de la sub-ventana Attributes de la ventana Preprocess.

Los filtros son necesarios para tareas de limpieza de los datos, pero también para realizar transformaciones de los mismos ya que algunos métodos de aprendizaje no pueden ser aplicados a ciertos tipos de datos. Principalmente, podemos distinguir dos tipos de atributos: nominales y numéricos. Conviene, entonces, en algunos casos transformar atributos numéricos en nominales y vise versa.

Ejercicio 1.1. Desde el entorno *Preprocess*, abre el data set iris.arff. Describe, con la mayor precisión posible, este fichero de datos según la información que aparece en la ventana *Preprocess*.

Ejercicio 1.2. Desde el entorno *Preprocess*, abre el data set iris.arff. Como se puede observar, todos sus atributos son numéricos. Utilizando los filtros disponibles en *Filter*, transforma este fichero de datos en otro cuyos atributos son nominales. Guarda el resultado como *irisnominal.arff*.

1.3. Tareas de procesamiento

Dentro de la minería de datos hemos de distinguir dos tipos de tareas, cada una de las cuales puede considerarse como un tipo de problema a ser resuelto por un algoritmo de minería de datos. Las distintas tareas pueden ser predictivas o descriptivas. Entre las tareas predictivas encontramos la clasificación y la regresión, mientras que el agrupamiento (*clustering*), las reglas de asociación son tareas descriptivas. Weka proporciona una amplia gama de algoritmos tanto para resolver tareas predictivas como descriptivas. Como se puede observar en la ventana principal de Weka explorer, existen 6 sub-entornos de ejecución:

- Preprocess: incluye las herramientas y filtros para cargar y manipular los datos.
- classification: Acceso a las técnicas de clasificación y regresión
- Cluster: integra varios métodos de agrupamiento
- Associate: incluye algunas técnicas de reglas de asociación
- Select Attributes: permite aplicar diversas técnicas para la reducción del número de atributos
- Visualize: incluye un entorno gráfico para analizar los datos y deducir relaciones entre sus atributos.

Nota: *la herramienta Weka está implementada en Java. Es importante remarcar que dado que se trata de una herramienta bajo licencia GNU, es posible actualizar su código fuente para incorporar*

nuevas utilidades o modificar las ya existentes, de ahí que podamos encontrar toda una series de proyectos asociados a Weka (Spectral clustering, Kea, WekaMetal, etc.) que permiten garantizar la continua evolución y adaptación de dicha herramienta.

1.4. Visualización de los datos

Para realizar una tarea de minería de datos es necesario realizar un reconocimiento o análisis exploratorio de los datos con el objetivo de detectar correlaciones y dependencias. En esta sección vamos a introducir una técnica simple para realizar este análisis exploratorio: técnica de visualización previa (entorno Visualize de Weka).

Para visualizar los datos lo podemos hacer desde la ventana *Preprocess* que nos da información sobre el número de registros y atributos del fichero de datos. Además, si seleccionamos cada uno de los atributos, conoceremos más información del atributo en cuestión: tipo (nominal o numérico), valores distintos, registros que no tienen información de ese atributo, el valor máximo y mínimo (caso de atributos numéricos), y finalmente un histograma con información sobre la distribución de los ejemplos para ese atributo. También, podemos analizar visualmente los datos desde la ventana *Visualize* que nos permite obtener tablas (o gráficos) cruzadas entre pares de atributos y decidir si tienen alguna relación con las clases.

Ejercicio 2.1. Abre el fichero de datos *drug1n.arff*. Este fichero recoge los datos para la predicción del tipo de fármaco que se debe administrar a un paciente afectado de rinitis alérgica según distintos parámetros/variables. Los datos que se recogen en los historiales clínicos de cada paciente son: la edad, el sexo, la tensión sanguínea (BP), nivel de colesterol, nivel de sodio (Na) y nivel de potasio (K). Hay cinco fármacos posibles: DrugA, DrugB, DrugC, DrugX, DrugY.

a) Utilizando los entornos de visualización comentados arriba, determina qué fármacos son más efectivos en general

b) A partir del entorno Visualize analiza las gráficas de dispersión obtenidas para cada par de atributos. Deduce qué par de atributos tienen una relación significativa.

Ejercicio 2.2. Abre el fichero de datos *iris.arff*.

a) Utilizando los entornos de visualización, determina qué atributo permite discriminar linealmente entre la clase iris-setosa y las dos otras clases

b) ¿Es posible separar linealmente la clase iris-versicolor de la clase iris-virginica?

Capítulo 2

Clasificación-Introducción

El objetivo de este capítulo es entender, en términos generales, las tareas de clasificación y la utilización del entorno explorer de Weka para la inducción y evaluación de modelos de clasificación supervisada.

2.1. Algoritmos de clasificación en Weka

Weka incorpora una lista amplia de algoritmos para resolver tareas de clasificación, aquí nos centraremos en los siguientes clasificadores:

- ZeroR: predicción de la clase mayoritaria;
- OneR: regla mayoritaria sobre un solo atributo;
- Ibk: método del vecino más próximo (KNN);
- NaiveBayes: método probabilístico;
- JRip: inducción de reglas;
- Id3 y j48: árboles de decisión;
- MultilayerPerceptron: red neuronal multicapas.

2.2. Clasificación en Weka

Los algoritmos de clasificación pueden ser utilizados después de una selección y limpieza de los datos (véase sección anterior), pulsando el botón *classify* de la ventana explorer. Esta ventana se divide en 4 entornos:

- *Classifier*: permite seleccionar y configurar el algoritmo de clasificación;
- *Test options*: permite elegir y configurar el método de evaluación del modelo inducido;
- *Result list*: muestra la lista de los modelos inducidos;
- *Classifier output*: muestra los resultados de la clasificación y la información sobre la definición del modelo.

En la parte *Classifier* del entorno *Classify* podemos ver que el clasificador por defecto es el *ZeroR*. Para cambiarlo, pulsar el botón *Choose*, y elegir uno de la lista desplegable. Para configurarlo hacer un clic sobre el texto indicando su nombre.

Una vez el clasificador es elegido y configurado, es necesario definir un método de validación del modelo en la ventana *Test options*. Los diferentes métodos de validación se abordarán con más

detalle en el capítulo 3. De momento, elegimos el método *Use traning set*, lo cual indica que vamos a usar el mismo conjunto de datos tanto para la fase entrenamiento como la de testeo.

Para iniciar el proceso de inducción del modelo, se debe pulsar en el botón *start* situado de bajo de la ventana *Test options*.

El modelo inducido (árbol de decisión, KNN ...) aparece en la ventana *Classifier output*, así como los detalles de configuración y medidas de validación del modelo definidas anteriormente.

Ejercicio 3. Abre el fichero de datos *weather.nominal.arff*. Ejecuta los algoritmos de clasificación siguientes usando *Use training set* como método de validación. Adopta los parámetros por defecto de todos los algoritmos:

- ZeroR;
- OneR;
- Ibk;
- NaiveBayes;
- Id3;
- j48.

Ejercicio 4. Compara la calidad de los modelos obtenidos (tasa de error). Cuales son los clasificadores que proporcionan los mejores resultados para este conjunto de datos.

2.3. Tareas de clasificación

Para empezar recordemos los siguientes conceptos acerca de algunas técnicas de muestreo de los datos:

- Muestreo aleatorio simple: la premisa de este muestreo es que cualquier instancia tiene la misma probabilidad de ser extraída en la muestra. Puede ser con o sin reemplazamiento. La manera más sencilla de realizar un muestreo sin reemplazamiento es asignar un número aleatorio a cada instancia y después ordenarlos por este valor. Se seleccionan los n primeros y tenemos una muestra de tamaño n sin reemplazamiento.
- Muestreo aleatorio estratificado: El objetivo de este muestreo es obtener una muestra balanceada con suficientes elementos de todos los estratos, o grupos. Lógicamente, para poder hacerlo es necesario conocer los estratos o grupos de interés. Esto generalmente ocurre con problemas de clasificación, donde estos estratos son precisamente las clases existentes.

Ejercicio 5. Abre el fichero de datos *weather.nominal.arff*, crea dos muestras de tamaño igual al tamaño del fichero inicial utilizando un muestreo aleatorio con reemplazo. Encuentra una manera de realizar esta operación. Guarda el resultado como *weather1.arff*.

Ejercicio 6. Utiliza el fichero *weather.nominal.arff* para generar una nueva muestra utilizando muestro aleatorio con reemplazo pero esta vez forzando una distribución uniforme del atributo clase. Guarda la nueva muestra con el nombre *weather2.arff*

Ejercicio 7. Utilizando el fichero *weather.nominal.arff*, ejecuta el algoritmo de clasificación Id3 en los 3 casos siguientes:

- Use training set;
- Supplied test set (utiliza el fichero weather2.arff obtenido en el ejercicio anterior);
- Cross validation;

- Percentage split;

Describe el árbol obtenido.

Ejercicio 8. ¿Con qué método de validación se han obtenido mejores porcentajes de bien clasificados? ¿Qué se entiende por sobreajuste (over-fitting)?

Ejercicio 9. Aplica los siguientes clasificadores sobre el fichero de datos *Drug1n.arff*:

- ZeroR;
- OneR;
- Ibk;
- NaiveBayes;
- Id3;
- j48.

La validación se realizará sobre el mismo conjunto de aprendizaje. **Cuales son los modelos que proporcionan los mejores resultados.**¿Has conseguido ejecutar todos los algoritmos? ¿Qué problemas has encontrado? ¿Cómo se pueden resolver?

Ejercicio 10. Compara las reglas obtenidas por los clasificadores OneR, Id3 y J48.

Capítulo 3

Introducción a Weka Experimenter

En este capítulo nos centraremos en las tareas de clasificación y la utilización del entorno *Weka Experimenter*. Se trata de realizar y analizar experimentos completos sobre tareas de clasificación considerando distintos algoritmos de aprendizaje y distintos conjuntos de datos.

3.1. Detalles sobre la evaluación de una tarea de clasificación

Hay que tener en cuenta varios elementos a la hora de realizar una tarea de clasificación:

- Los ficheros de datos utilizados;
- Los métodos de validación adoptados;
  - Validación sobre el mismo conjunto de datos que ha servido para inducir el modelo (Método no honesto de estimación)
  - Validación sobre un conjunto de datos distinto (Método H)
  - Validación cruzada
- Los algoritmos de clasificación utilizados.

En esta sección trataremos de observar la influencia de estos elementos sobre los resultados de la clasificación y sobre la calidad de las soluciones obtenidas.

El Weka Experimenter es un entorno gráfico que permite al usuario crear, ejecutar, modificar y analizar experimentos sobre tareas de clasificación (o regresión) de un modo ágil y eficaz comparado con el entorno Weka Explorer. Por ejemplo, puede crear un experimento que ejecuta varios clasificadores sobre varios conjuntos de datos y analizar los resultados para determinar si uno de los clasificadores es estadísticamente mejor que los demás.

3.2. Definir un experimento

El entorno Experimenter de Weka se compone de varias ventanas dentro de un orden lógico correspondiente a las diferentes fases del experimento:

- Configurar el experimento;
- Ejecutar el experimento;
- Analizar los resultados.

Ejercicio 11. Desde la ventana *Setup*, pulsa el botón *New* para iniciar un nuevo experimento. Esto permite fijar los parámetros por defecto del experimento. Ahora, para abrir el fichero de datos que servirá para aprender (y/o testar) el modelo, pulsa el botón *Add New* de la subventana *Datasets* de la ventana *Setup*. Y selecciona el dataset *iris.arff*.

3.3. Guardar un experimento

Una de las ventajas al trabajar con el entorno Experimenter es la posibilidad de guardar los resultados de un experimento.

Ejercicio 12. Para identificar el fichero que servirá de soporte para los resultados, pulsa el botón *Choose* de la subventana *Destination*, y elige *CSVResultListener*. Haz un clic sobre el texto que aparece junto al botón *Choose*, se abre una ventana de diálogo, escribe el nombre del fichero de salida *Output* , por ejemplo, *ResultExperiment1.txt* y cierra la ventana.

No sólo podemos guardar los resultados de un experimento, sino también la definición de todos los parámetros del mismo (Datasets, Destination, Result generator, Generator properties, etc.).

Ejercicio 13. Pulsa el botón *Save* situado arriba de la ventana *Setup*. Escribe el nombre del fichero con la extensión *exp* (por ejemplo, Experiment1.exp).

Para recuperar la definición de un experimento, pulsa el el botón *Open* situado arriba de la ventana *Setup* y selecciona *Experiment1.exp* desde la ventana de diálogo.

3.4. Ejecutar un experimento

Ejercicio 14. Para ejecutar el experimento que acabamos de configurar, selecciona la ventana *Run* del *Experimenter Environment* y pulsa el botón *Start*.

En este caso, se ha ejecutado un experimento por defecto sobre el dataset *iris.arff*. La definición por defecto puede ser la siguiente: un *RandomSplitResultProducer* con un parámetro fijado a 10 para el aprendizaje y testeo aleatorios efectuados sobre el dataset *iris.arff*, utilizando un 66 % de casos para el aprendizaje y el resto para el testeo, y utilizando el clasificador *ZeroR*.

- Started
- Finished
- There were 0 errors

Si el experimento se ha definido correctamente, los 3 mensajes de arriba aparecerán en el panel *Log Panel*. Los resultados del experimento son guardados en el fichero *ResultExperiment1.txt*. Es un fichero que podemos abrir en una hoja de cálculo Excel para su análisis.

3.5. Reconfigurar un experimento

Los parámetros de un experimento se pueden modificar desde la ventana *Result generator*. A partir de allí se puede elegir entre 5 generadores de resultados. Aquí nos centraremos en los siguientes:

- RandomSplitResultProducer
- CrossValidationResultProducer
- AveragingResultProducer

El RandomSplitResultProducer realiza un aprendizaje y testeo aleatorios de manera repetida. La proporción de casos reservados para el aprendizaje se especifica con el parámetro *TrainPercent*. El CrossValidationResultProducer realiza un aprendizaje y testeo basados en la validación cruzada. El número de rodajas (folds) se especifica con el parámetro *numFolds*. Y finalmente, el AveragingResultProducer proporciona resultados promediados, generalmente, sobre ejecuciones de validación cruzada.

Para todos estos generadores de resultados, el número de ejecuciones (runs) de los clasificadores (*splitEvaluator*) se define en la ventana *Setup*, opción *Runs*.

Ejercicio 15. Haz un clic sobre el nombre del generador de resultados que aparece en la ventana *Result generator*. En la ventana de diálogo, haz un clic sobre la entrada del *splitEvaluator* para abrir

las propiedades del clasificador. Para cambiarlo, utilizar el botón *Choose*, y elige, por ejemplo, el clasificador J48.

Ejercicio 16. Acepta las modificaciones (botón OK). El nombre del clasificador aparece en el panel *Result generator*. Ahora se puede reejecutar el experimento. El fichero *ResultExperiment1.txt* será sobrescrito con los resultados del J48.

Comparar varios clasificadores

Para comparar varios clasificadores, hay que definirlos en el panel *Generator properties*.

Ejercicio 17. Primero, activa la opción *Enabled* en el panel *Generator properties*. Se abre una ventana de diálogo, selecciona *splitEvaluator* y acepta (*botón Select*). El nombre del clasificador aparece en el panel *Generator properties* (por defecto, ZeroR) . Para elegir otro clasificador, haz un clic sobre el nombre del clasificador. Se abre una ventana de propiedades, a partir de allí elegir otro clasificador, por ejemplo, el J48, y aceptar (botón OK). Para añadir el clasificador elegido, haz un clic sobre el botón *Add*. Ejecuta el experimento y visualiza los resultados del fichero *ResultExperiment1.txt*. Para quitar un clasificador, usa el botón *Delete* abajo de la ventana.

Los clasificadores elegidos se pueden ejecutar a la vez sobre un conjunto de ficheros de datos, los cuales se pueden añadir desde la ventana *Datasets* usando el botón *Add New*.

3.6. Realizar un experimento

Ejercicio 18. Define un experimento con los siguientes parámetros:

- aplicando 10 ejecuciones de aprendizaje y testeo aleatorios
- utilizando los datasets *iris.arff* y *Soybean.arff*
- usando los clasificadores *ZeroR*, *OneR* y *j48*

Guarda el experimento como *Experiment2.exp*. Observa los resultados obtenidos (*ResultExperiment2.txt*) utilizando una hoja de cálculo Excel. Utiliza las funciones de Excel para rellenar la siguiente tabla:

porcentaje bien clasificados	ZeroR	OneR	J48
Iris	...	...	...
Soybean	...	...	...

Crea un gráfico basado en la tabla anterior. Qué conclusiones se pueden deducir acerca del rendimiento de los clasificadores en función de los datasets.

3.7. Analizar los resultados de un experimento

En esta sección trataremos de analizar los resultados de un experimento utilizando el entorno *Analyse* de Weka Experimenter. En las secciones anteriores, hemos utilizado el *CSVResultListener* como formato para guardar los resultados, el cual, nos genera un fichero txt que podemos abrir en Excel para su posterior análisis. Estos resultados, también, pueden ser guardados utilizando el *InstanceResultListener*. En este caso, el fichero tiene un formato arff, y por tanto, puede ser utilizado por Weka.

Ejercicio 19. Inicia un nuevo experimento. Pulsa el botón *Choose* del panel *Destination* para seleccionar *InstanceResultListener*. Utiliza el nombre *ResultExperiment3.arff*. El contenido de este fichero tiene una cabecera similar a lo siguiente:

```
@relation InstanceResultListener
@attribute Key_Dataset {iris}
@attribute Key_Run {1,2,3,4,5,6,7,8,9,10}
@attribute Key_Scheme {weka.classifiers.ZeroR}
@attribute Key_Scheme_options {''}
@attribute Key_Scheme_version_ID {-790200859438591175}
@attribute Num_Fold numeric
@attribute Date_time numeric
@attribute Number_of_training_instances numeric
@attribute Number_of_testing_instances numeric
@attribute Number_correct numeric
@attribute Number_incorrect numeric
@attribute Number_unclassified numeric
@attribute Percent_correct numeric
.
.
.
@data
```

Ejercicio 20. Añade 3 clasificadores *ZeroR, OneR y J48* utilizando *Generator properties*. Utiliza el *RandomSplitResultProducer* para un aprendizaje y testeo aleatorios con un 70 por ciento de los casos para el aprendizaje y el resto para el testeo. Ejecuta los clasificadores (10 runs) sobre los datasets *iris, Soybean y Titanic*.

Ejercicio 21. Para analizar los resultados, utiliza la ventana *Analyse* del entorno Weka Experiment. Recuerda que los resultados deben ser guardados en formato arff. Puedes utilizar el botón *Experiment* para analizar los datos del experimento actual. El número de líneas de resultados disponible (*Got 90 results*) aparecen en el panel *Source*. Este experimento se compone de 10 ejecuciones, 3 clasificadores, 3 datasets, es decir, un total de 90 líneas de resultados.

Ejercicio 22. Selecciona el clasificador ZeroR como clasificador base (*Test base*), selecciona el atributo *Percent correct* del campo *Comparison field*, y pulsa el botón *Perform test* para iniciar la comparación de los 3 clasificadores utilizando como criterio el porcentaje de bien clasificados. Los resultados de análisis pueden ser los siguientes:

Cuadro 3.1: Table Caption

Data Set	(1)	(2)	(3)
iris	94.90	94.31	33.33 •
soybean	88.74	40.18 •	13.28 •
titanic.txt	78.45	77.81	67.70 •

◦, • statistically significant improvement or degradation

Los clasificadores aparecen en las columnas y los datasets en las filas. El porcentaje de bien clasificados de cada uno de los clasificadores aparece para cada fila de los datasets. Las anotaciones 'v' y '*' indican respectivamente que un resultado específico es estadísticamente mejor o pero que el clasificador base. De bajo de cada columna (excepto la primera) aparece un contador (xx/yy/zz) del número de líneas en las cuales el clasificador ha sido mejor que (xx) , igual a (yy), o peor que (zz) el clasificador base. Para este experimento, el ZeroR ha sido 3 veces peor que el clasificador base (J48). Mientras que el OneR ha sido 2 veces igual y una vez peor que el clasificador base.

El valor 10 que aparece al inicio de cada fila indica el número de ejecuciones.

Capítulo 4

Tareas de minería de datos

4.1. El clasificador KNN

Este experimento consiste en analizar el rendimiento del clasificador KNN. Se consideran distintos datasets y se tendrán en cuenta factores propios del clasificador (valor de K).

Tarea 1 Desde el entorno Weka Experimenter, inicia un nuevo experimento. Abre los datasets *credit-g.arff, soybean.arff, Drug1n.arff y iris.arff*. Define 4 clasificadores KNN con valores de K=1,2,5,10 respectivamente. Utiliza el *RandomSplitResultProducer* para un aprendizaje y testeo aleatorios con un 70 por ciento de los casos para el aprendizaje. Ejecuta el experimento (10 runs).

Accede a la ventana *Analyse*. Recuerda que puedes utilizar el botón *Experiment* del entorno *Analyse* para acceder a los resultados del experimento actual, o bien, utilizar el botón *File* para cargar el fichero arff de los resultados del experimento (en este caso, ResultExperimento1.arff).

a) Realiza una comparación de los distintos clasificadores considerando el KNN con K=1 como clasificador base, y usando el porcentaje de bien clasificados como criterio de evaluación. Usando Excel, dibuja un gráfico que refleja los resultados obtenidos. ¿Qué conclusiones se pueden sacar?

4.2. Árboles de decisión

En este experimento trataremos de analizar los algoritmos de inducción de árboles de decisión vistos en las clases teóricas. Este análisis se llevará a cabo teniendo en cuenta factores tales como el proceso de poda (pruning), o el número de máximo de objetos por hoja.

Tarea 2 Desde el entorno Weka Explorer, abre el dataset *drug1n.arff*, es un fichero que recoge el historial clínico de los pacientes. Hay cinco fármacos posibles. DrugA, DrugB, DrugC, DrugX, DrugY. Haz un primer análisis visual de los datos para saber qué fármacos son más comunes en general. Ejecuta el clasificador J48 con los parámetros por defecto usando validación cruzada (10 folds).

a) Visualiza el árbol obtenido. ¿Cuantas reglas se han obtenido? ¿Qué representan los valores entre paréntesis al final de cada regla? Guarda el árbol obtenido y los datos correspondientes.

b) Ahora, analiza con más detenimiento los atributos de entrada del problema. Es posible mejorar el rendimiento del modelo si combinamos algunos atributos. Desde el entorno *visualize*, podemos ver que, en la gráfica que representa al atributo Na frente a K, existe una clara separación lineal entre una relación K/Na alta y K/Na baja. Este conocimiento nos servirá para mejorar nuestro modelo.

c) Desde la ventana *Preprocess*, utiliza uno de los filtros disponibles en *Filter* para crear un nuevo atributo basado en los atributos Na y K. Aplica el filtro ya configurado.

d) Ejecuta el clasificador J48 con los parámetros por defecto usando validación cruzada (10 folds). Visualiza el árbol obtenido. ¿Cuantas reglas se han obtenido?. ¿Qué conclusiones se pueden sacar?

4.3. Clasificadores Bayesianos

Notas: *El clasificador Naive-Bayes consiste en aprender, a partir del fichero de casos, la probabilidad condicional de cada atributo A_i dada la clase C. La clasificación se realiza, entonces, utilizando la regla de Bayes para calcular la probabilidad de C dada una instancia particular $A_1, A_2, ..., A_n$. Para hacer posible el cálculo de esta probabilidad, el paradigma Naive-Bayes se basa en la hipótesis de independencia; todos los atributos A_i son condicionalmente independientes dada la clase C.*

*Las redes bayesianas (RB) consisten en representar el conocimiento cualitativo del modelo mediante un grafo dirigido acíclico. Este conocimiento se centra en la definición de relaciones de independencia/dependencia entre los atributos que componen el modelo. El problema de aprendizaje de RB consiste en, dado un conjunto de datos, encontrar el grafo dirigido acíclico que mejor represente el conjunto de dependencia/independencia. Este problema se plantea como un problema de optimización. Weka nos proporciona una amplia lista de algoritmos de búsqueda (*searchAlgorithm*).*

Tarea3: Desde el entorno Weka Explorer. Abre el dataset *adult.arff*, es una base de datos original del UCI conocida también como *Census Income*. Consiste en predecir si los ingresos de un individuo serán o no superiores a 50000 dolares. Consta, además de la variable clase, de 14 atributos, 8 discretos y 6 numéricos. Define un clasificador NaiveBayes (configuración por defecto). Utilizar *Cross-Validation (10 folds)* para el aprendizaje y testeo. Ejecuta el algoritmo.

a) Comenta los resultados que aparecen en el Classifier Output.

b) ¿Crees que el NaiveBayes es competitivo a pesar de la restricción de independencia? ¿Podemos mejorar su rendimiento?

c) Abre el datasets *Breast-cancer*, es una base de datos original del UCI que consiste en predecir la recurrencia o no de un cierto tumor. Aprende un clasificador NaiveBayes aumentado a árbol TAN (tree augmented network). Utiliza *Cross-Validation (10 folds)* para el aprendizaje y testeo. Ejecuta el algoritmo. Visualiza el árbol obtenido. Accede a las tablas de distribuciones de probabilidad de los atributos pulsando los nodos del árbol.

Tarea4: Desde el entorno Weka Experimenter, abre los datasets *Breast-cancer, adult.arff, soybean.arff, iris.arff*. Define los clasificadores NaiveBayes, NaiveBayes aumentado a árbol TAN (tree augmented network) y el clasificador Bayesiano 3-dependiente (el algoritmo de búsqueda K2). Utiliza *Cross-Validation (10 folds)* para el aprendizaje y testeo. Ejecuta el algoritmo.

a) Compara los resultados en base al porcentaje de bien clasificados.

4.4. Inducción de reglas

En este experimento trataremos de analizar el algoritmo RIPPER de inducción de reglas visto en las clases teóricas. Este análisis se llevará a cabo considerando distintos datasets y teniendo en cuenta factores tales como el tamaño de los subconjuntos Growing y Pruning, o el número de mínimo de objetos en una regla.

Tarea 5 Desde el entorno Weka Explorer, abre el dataset *drug1n.arff*. Ejecuta el clasificador JRip con los parámetros por defecto usando validación cruzada (10 folds).

a) ¿Cuantas reglas se han obtenido?. Interpreta los valores entre paréntesis que aparecen al final de cada regla. ¿a que se refiere la última regla? Guarda los datos obtenidos.

b) Compara estas reglas con las obtenidas al utilizar el clasificador C45 (J48). ¿Podemos generar un árbol a partir de las reglas obtenidas por JRip?

c) Vuelve a realizar lo mismo pero esta vez aplicando al fichero de datos un filtro para añadir una nueva expresión dada por el cociente entre los atributos K y Na (punto b-tarea1-experimento2).

d) ¿Como se ven afectados los resultados al cambiar los parámetros *folds y minNo*? Encuentra valores para estos parámetros que producen los mejores resultados.

4.5. Evaluación de los clasificadores

Los métodos de aprendizaje permiten inducir modelos a partir de un conjunto de datos. En la mayoría de los casos es necesario evaluar la calidad de los modelos de la manera más exacta posible. Por ejemplo, en el ámbito de aplicación de un modelo un error en la predicción conlleva importantes consecuencias, es importante conocer el nivel de precisión del modelo aprendido. Por tanto, la etapa de evaluación de modelos es crucial para la aplicación real de las técnicas de Data Mining.

En este experimento trataremos de analizar las distintas técnicas de evaluación de modelos vistas en las clases teóricas. En particular, consideraremos los métodos basados en precisión y coste.

Tarea 6 Desde el entorno *Weka Explorer*, abre el fichero de datos *vote.arff*. Ejecuta el clasificador KNN con K igual al valor óptimo de entre k=1-5, usando cada vez uno de los siguientes métodos de evaluación:

- Use training set
- Cross-validation
- Porcentage split

Recoge los resultados en una tabla basándose en el porcentaje de bien clasificados. Comenta los resultados. ¿Se puede asociar un mejor porcentaje de bien clasificados a una buena validación del modelo? ¿Qué se entiende por sobre ajuste (overfiting)?

Tarea 7 En esta tarea vamos a abordar el aprendizaje basado en coste. Desde el entorno *Weka Explorer*, abre el fichero de datos *credit-g.arff* es un conjunto de datos que representa clientes que demandaron un crédito bancario. Suponemos que la matriz de coste asociada a este dataset es la siguiente:

	Clase real	
	Si	No
Si	0	5
No	1	0

Esta tabla indica que es 5 veces más costoso si se otorga un crédito a una persona que no lo devuelve, que la situación contraria.

a) Primero, escoge como modelo el clasificador ZeroR (es un clasificador que asigna siempre la clase mayoritaria). En la subventana *Test option* pulsa *More option*, activa la casilla *Cost-sensitive evaluation*, y introduce la matriz de coste. Ejecuta la evaluación con validación cruzada (10 Folds). Recoge la información sobre la precisión y coste obtenidos. ¿Crees que este clasificador ha tenido en cuenta la matriz de costes?

b) Realiza lo mismo que en punto (a), pero esta vez usando el clasificador KNN con k =3. ¿se ha mejorado el coste de los errores? ¿Crees que esta vez se ha tenido en cuenta la matriz de coste?

c) Desde la ventana *Classify /classifiers/meta*, encuentra uno (o varios) clasificadores que utilizan de manera explícita una evaluación basada en coste. Aplica uno de ellos utilizando como clasificadores base los vistos en los puntos (a) y (b). Comenta los resultados.

4.6. Selección de variables (FSS)

En la mayoria de las bases de datos existe mucha información que es incorrecta respecto al dominio de la realidad que se desea cubrir, y un número menor, pero a veces también importante, de datos inconsistentes o irrelevantes.

El problema de la selección del subconjunto de variables para la inducción de un modelo clasificador, se denomina FSS (*Feature Subset Selection*) y surge motivado por la *no monotocidad* de los modelos clasificatorios en relación con el número de variables predictoras, así como por la existencia de ciertas variables predictoras que pueden llegar a ser *irrelevantes* o incluso *redundantes*.

La selección de variables tiene 4 objetivos principales:

- Reduce el tamaño de los datos, al eliminar atributos de todos los ejemplos que puedan ser irrelevantes;
- Mejora la calidad del modelo, al permitir al algoritmo de minería de datos centrarse sólo en los atributos relevantes;
- Expresa el modelo resultante en función de menos variables; esto es importante cuando se desean modelos comprensibles (árboles);
- Reduce la dimensionalidad del problema

Esta práctica tiene como objetivo analizar los métodos de selección de variables utilizando las aproximaciones *Filter y Wrapper*. Recuerda que la aproximación Filter se basa en definir una medida indirecta (information gain with respect to the class, gain ratio with respect to the class, etc.) de la aportación de cada atributo, independientemente del paradigma clasificatorio a aplicar. Mientras que la aproximación Wrapper evalúa cada subconjunto de atributos candidatos directamente en el modelo clasificatorio construido con dicho subconjunto. Esta aproximación se plantea como un problema de optimización que consiste en encontrar el subconjunto óptimo de atributos que proporciona los mejores resultados.

Tarea 8 Desde el entorno Weka Explorer, abre el dataset *empleados.arff*, realiza una selección de atributos usando una aproximación filter con *GainRatioAttributeEval y InfoGainAttributeEval* como medidas de aportación de los atributos. Interpreta los resultados del *Attribute selection ouput*.

Tarea 9 Desde el entorno Weka Explorer, abre el dataset *empleados.arff*, realiza una selección de atributos usando una aproximación wrapper *WrapperSubsetEval* con Naive-Bayes como clasificador base, y usando el algoritmo de búsqueda *BestFirst*. Utiliza *Use full training set* como modo de validación. Interpreta los resultados.

a) Compara el subconjunto óptimo de atributos obtenido con los atributos mejor evaluados con el método Filter.

b) Compara el porcentaje de bien clasificados de dos clasificadores Naive-Bayes aplicados al dataset anterior en los casos siguientes:

   - se consideran todos los atributos
   - se considera sólo el subconjunto óptimo de atributos encontrado en el paso anterior. Comenta los resultados.

4.7. Metaclasificadores

Con el objetivo de mejorar la precisión de las predicciones, ha surgido un interés creciente en los últimos años en la definición de métodos que combinan varios modelos clasificatorios. Se trata, entonces, de combinar las predicciones del conjunto de modelos, normalmente por votación, para clasificar nuevos ejemplos. La precisión obtenida por esta combinación supera, generalmente,

la precisión de cada componente individual del conjunto. Este experimento tiene como objetivo utilizar los métodos de **Bagging, Boosting y Stacking** para la combinación de las salidas de varios clasificadores.

4.7.1. Bagging

Es esta sección trataremos de analizar los diferentes parámetros del algoritmo y la influyencia que tienen sobre los resultados.

Tarea 10 Desde el entorno *Explorer*, inicia un experimento sobre el fichero de datos *weather.arff*, utilizando el meta clasificador *bagging* con el J48 como clasificador base. Los parámetros de *bagging* que nos interesan son:

- la elección del clasificador base
- el parámetro *bagSizePercent*: indica el porcentaje de casos seleccionados para generar las muestras boostrap
- el parámetro *numIterations*: indica el número de iteraciones del bagging

a) Adoptando los parámetros por defecto, y seleccionando cross-validation (10 folds) como método de validación. Ejecuta el experimento.

b) Los resultados muestran los 10 árboles de decisión inducidos (*numIteration*=10). ¿Cuales son los mejores?

c) Aumenta el número de iteraciones *numIteration*. ¿Se ha conseguido reducir la tasa de error?

d) Reduce el parámetro *bagSizePercent*. ¿Qué efecto tiene esto sobre la tasa de error?

Tarea 11 Compara el rendimiento de un clasificador J48 con bagging y sin bagging sobre el fichero de datos *labor.arff*. Comenta los resultados.

4.7.2. Boosting

Tarea 12 Desde el entorno *Explorer*, inicia un experimento sobre el fichero de datos *weather.arff*, utilizando el meta clasificador *AdaBoostM1* con el J48 como clasificador base. Los parámetros de *boosting* que nos interesan son:

- la elección del clasificador base;
- el parámetro *useResampling*: indica si el boosting utiliza un pesado de las instancias y una posterior actualización de los mismos en cada iteración, o simplemente se usa un remuestreo de las instancias;
- el parámetro *numIterations*: indica el número de iteraciones del boosting

a) Adoptando los parámetros por defecto, y seleccionando cross-validation (10 folds) como método de validación. Ejecuta el experimento. ¿Crees que se ha producido un boosting?

b) En las propiedades del J48, Cambia el valor del parámetro *minNumObj* de 2 a 3. Este número indica el número de objetos (o instancias) por hoja. Ejecuta el experimento. Examina los árboles inducidos en las diferentes iteraciones del boosting. ¿cuales son los mejores?

c) Utiliza el ficher de datos *labor* para comparar los resultados obtenidos con el J48 con y sin boosting.

d) Prueba el boosting con el IBk como clasificador base. ¿Crees que se ha producido un boosting? Pon en *True* el parámetro *useResampling* y vuelve a ejecutar el experimento. Comenta los resultados.

Tarea 13 Desde el entorno *Weka Experimenter*, iniciar un experimento nuevo. Abre los datasets *Iris.arff, Soybean.arff, Labor.arff*. En el generador de resultados elige *CrossValidationResultProducer*. Define los siguientes clasificadores:

- J48;
- bagging con J48 como clasificador base y con el parámetro minNumObj=3;
- boosting con J48 como clasificador base y con el parámetro minNumObj=3;

Guarda tanto la definición como los resultados del experimento.

Tarea 14 Accede a la ventana *Analyse*. Realiza una comparación de los clasificadores definidos arriba considerando como clasificador base el J48.

4.7.3. Stacking

El *Stacking* es un meta-clasificador que combina las salidas de varios clasificadores situados en un primer nivel, y aprende a predecir la clase basándose en un clasificador simple definido en el último nivel.

Tarea 15 Desde el entorno *Weka Explorer*. Abre el dataset *credit.arff*, y selecciona Stacking pulsando el botón *Choose* en la ventana *Classify*. Define 3 clasificadores base (J48, IB1, NaiveBayes) y utiliza el clasificador *J48* como *metaClassifier*. Ejecuta el experimento. Los resultados muestran los

modelos inducidos para cada clasificador individual (excepto por IB1) y para el modelo aprendido por el meta-clasificador

a) Interpreta los resultados. ¿Crees que con el Stacking hemos mejorado la precisión de los clasificadores base para este conjunto de datos?

b) Utiliza el entorno Weka Experimenter para realizar una comparación del Stacking con cada uno de los clasificadores base.

4.8. Algoritmos de Clustering

El análisis de clusters consiste en dividir los datos en grupos de objetos (o clusters) basandose simplemente en la información contenida en los datos que describen estos objetos y las relaciones que existen entre ellos. El clustering puede ser visto como una clase de clasificación, en la cual, se realiza un etiquetado de los objetos con las etiquetas de los clusters. Es por esta razón, a menudo se refiere al clustering como clasificación no supervizada. Pero generalmente, cuando el término clasificación es utilizado sin cualificativo, entonces se refiere a la clasificación supervizada.

Recordar que todos los conjuntos de datos que ofrece Weka tienen un atributo que representa la clase. Para tareas de clustering, este atributo no tiene interés, por tanto, lo vamos a ignorar excepto para la fase de evaluación.

4.8.1. K-means

Tarea 16 Desde el entorno Weka Explorer, abre el dataset *weather.arff*. En el menu desplegable *Clusterer*, selecciona el algoritmo *SimpleKMeans* y fija:

- el parámetro *numClusters* a 3
- el parámetro *random seed* a 20

Activar la casilla *classes to clusters evaluation*, los clusters serán evaluados con respecto a la clase actual de los ejemplos que forman el conjunto de testeo. Activa la casilla *Store clusters for visualization*. Ejecuta el algoritmo.

Los resultados muestran:

- los centroides
- el número de instancias asignadas a cada uno de los clusters
- la distribución de los ejemplos del conjunto de entrenamiento dentro de los clusters versus sus clases efectivas
- tmabién proporcionan una estimación de los clusters que representan mejor las clases existentes.

Tarea 17

a) ¿cuales la razón detrás del hecho de asociar los clusters a las etiquetas de clase existentes?

b) Ejecuta el K-means varias veces, cada vez con un valor distinto del parámetro *random seed*

c) Selecciona la mejor versión basándose en la tasa de error

d) Vuelve a realizar el clustering usando *number of clusters=2*

e) Ejecuta el algoritmo para distintos valores del parámetro *random seed*

f) Haz lo mismo para *number of clusters* igual a 4, 6 y 8

f) Comenta la calidad del clustering en función del número de clusters

f) Ya que el K-means es muy sensible a la elección inicial, propone una solución a este problema.

Bibliografía

[1] C. M. Bishop (1995). *Neural Networks for Pattern Recognition.* Oxford University Press, Inc.

[2] P. R. Cohen (1995). *Empirical Methods for Artificial Intelligence.* Kluwer Academic Publishers.

[3] C. M. Bishop (2006). *Pattern Recognition and Machine Learning.* Springer.

[4] N. Cristianini, J. Shawe-Taylor (2000). *An Introduction to Support Vector Machines.* Cambridge University Press.

[5] G.J. McLahlan and T.K. Krishnan (1997). *The EM algorithm and Extension.* Wiley and Sons.

[6] J. Hand (1997). *Construction and Assessment of Classsification Rules.* John Wiley.

[7] H. Liu, H. Motoda (1998). *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer Academic Publishers.

[8] R. S. Michalski, I. Bratko, M. Kubat (1998). *Machine Learning and Data Mining. Methods and Applications.* John Wiley and Sons.

[9] D. Michie, D. J. Spiegelhalter, C. C. Taylor (1994).*Machine Learning, Neural and Statistical Classification.* Ellis Horwood.

[10] B. Mirkin (1996). *Mathematical Classification and Clustering.* Kluwer Academic Publishers.

[11] T. M. Mitchell (1997). *Machine Learning.* McGraw-Hill.

[12] R. E. Neapolitan (2003). *Learning Bayesian Networks.* Prentice Hall.

[13] S. Russel, P. Norvig (1995). *Artificial Intelligence. A Modern Approach.* Prentice Hall Series In Artificial Intelligence.

[14] S. M. Weiss, N. Indurkhya (1998). *Predictive Data Mining. A Practical Guide.* Morgan Kaufmann Publishers.

[15] S. M. Weiss, C. A. Kulikovski (1991). *Computer Systems that Learn.* Morgan Kaufmann.

[16] I. H. Witten, E. Frank (2005). *Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann. Second Edition.

[17] C.R. Reeves (Ed.)(1993). *Modern Heuristic Techniques for Combinatorial Problems.* Blackwell Scientific Publications.