Departamento de Informática
Universidad Carlos III de Madrid

Grado en Ingeniería Informática

Aprendizaje Automático

Curso 2017-2018

Tutorial 3

Experimentación múltiple

Interfaces KnowledgeFlow y Experimenter de Weka.

- El objetivo de este tutorial es familiarizarse con dos interfaces de Weka que facilitan el diseño y la ejecución de múltiples experimentos de una sola vez.
- La interfaz KnowledgeFlow es una alternativa al Explorer manejado en el tutorial anterior, aportando algunas capacidades adicionales. La interfaz Experimenter permite ejecutar de una sola vez distintos algoritmos de aprendizaje automático sobre varios conjuntos de datos y comparar los resultados usando tests estadísticos.

1. Ejercicio 1: KnowledgeFlow

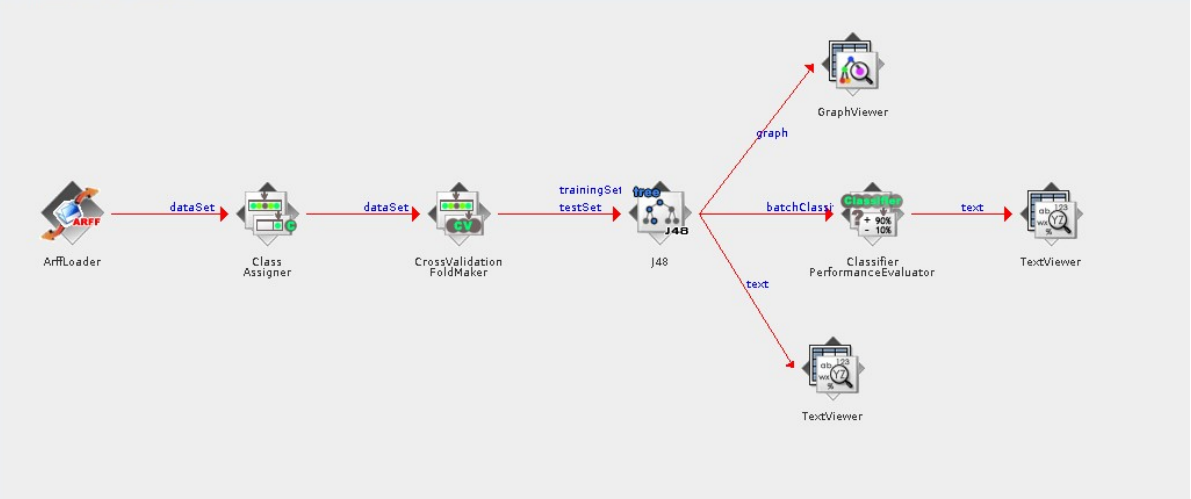

Figura 1: Flujo de conocimiento para analizar los datos en `adult-data.arff`.

1. Ejecutar Weka con `java -jar weka.jar -Xmx500m` para forzar la reserva de 500 MB de RAM.
2. Iniciar el módulo KnowledgeFlow de Weka. Los próximos apartados explican cómo construir el flujo de conocimiento que se muestra en la Figura 1.
3. Insertar un nodo del tipo `Arff Loader` (pestaña `DataSources`). Configurarlo para que lea el fichero `adult-data.arff` (botón derecho sobre el nodo opción `Configure`).

4. Insertar un nodo del tipo `Class Assigner` (pestaña `Evaluation`). Unirlo con el nodo anterior enviando el `dataSet` (botón derecho en el nodo anterior opción `dataSet`). Configurar el nodo para que la clase sea la variable llamada `salary` (debería ser la opción por omisión).

5. Insertar un nodo del tipo `Cross Validation FoldMaker` (pestaña `Evaluation`). Unirlo con el nodo anterior enviando el `dataSet` (botón derecho en el nodo anterior opción `dataSet`). En la configuración se puede elegir el número de `Folds` (por defecto son 10) y la semilla aleatoria (por defecto a 1).

6. Insertar un nodo del tipo J48 (pestaña `Classifier`). Unirlo también con el nodo anterior enviando `trainingSet` (botón derecho en el nodo anterior, opción `trainingSet`). Unirlo con el nodo anterior enviando `testSet`. Hay que enviar tanto el conjunto de entrenamiento como el conjunto de test, porque, si no, la validación cruzada no funcionará correctamente.

7. Insertar un nodo del tipo `Classifier PerformanceEvaluator` (pestaña Evaluation). Unirlo con el nodo anterior enviando `batchClassifier`. Dejar el resto de opciones por defecto.

8. Insertar un nodo del tipo `TextViewer` (pestaña `Visualization`). Unirlo con el nodo J48 enviando text.

9. Insertar un nodo del tipo `TextViewer` (pestaña `Visualization`). Unirlo con el nodo `Classifier Performance Evaluator`, enviando `text`.

10. Insertar un nodo del tipo `GraphViewer` (pestaña `Visualization`). Unirlo con el nodo J48, enviando `graph`.

11. Ejecutar el flujo de conocimiento. Para ello ejecutar la opción Start loading del nodo `Arff Loader`. Seleccionar la opción `Show Results` en los nodos `TextViewer`.

    Preguntas:

    a) ¿Qué se muestra en cada uno de ellos?

    b) ¿Cuál es el porcentaje de instancias clasificadas correctamente?

12. Guardar el diagrama de flujo en formato .kfml como `e1-crossvalidation.kfml`.

13. Introducir el filtro de balanceo de datos aplicado en el tutorial anterior, situado en *Filters/Resample* y cambiar el modo de evaluar a "Supplied test set" para ello copia el diagrama de la Figura 2

14. Guardar el diagrama de flujo en formato .kfml como `e1-suppliedtestset.kfml`.

    Pregunta:

    c) ¿Cuál es la utilidad de crear flujos de conocimiento con esta interfaz de Weka?

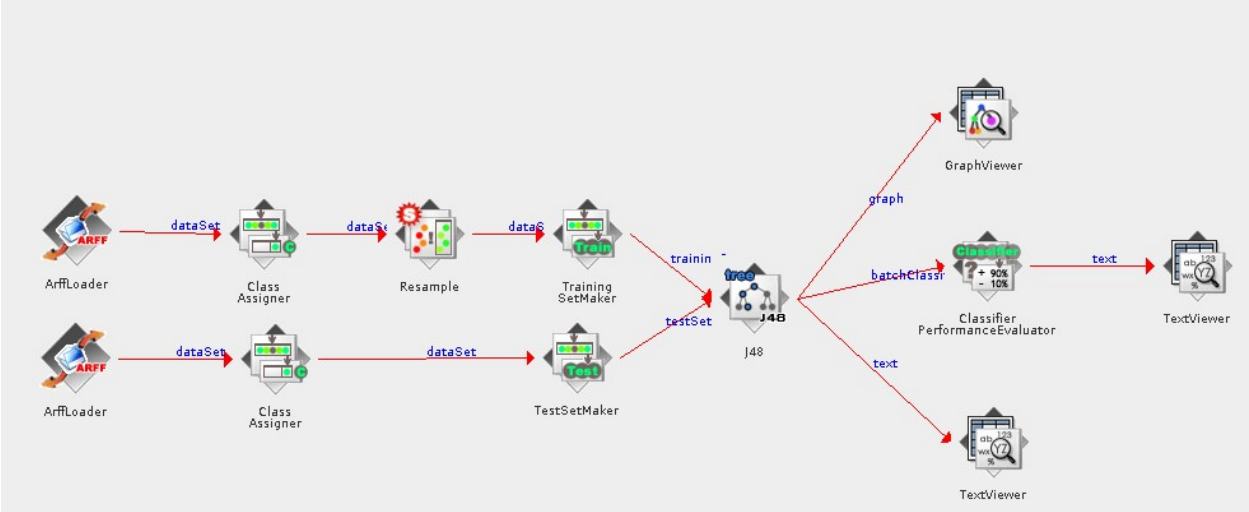

Figura 2: Flujo de conocimiento con Supplied test set.

2. Ejercicio 2: Experimenter

Este ejercicio se apoya en la función básica de extracción de características programada en el Tutorial 1 para los agentes T1HumanAgent y T1BotAgent, actualizando algunas características para dejarlos completamente preparados para las próximas prácticas. Se centra en modelos que permitan predecir el número de celdas recorridas por Mario (`distancePassedPhys`) que se puede encontrar en `environment.getEvaluationInfoAsInts[1]` en cada tick.

2.1. Actualización de los agentes de MarioAI

1. Crear dos nuevos agentes T3HumanAgent y T3BotAgent basados en los que se hicieron para el Tutorial 1. Deben incluir todas las mejoras y correcciones que se consideren oportunas y también las características descritas a continuación.

2. El número de celdas recorridas (`distancePassedPhys`) debe colocarse en el último lugar de la línea que representa la instancia de aprendizaje.

3. Incluir también estos atributos antes del anterior: posición de Mario (`MarioFloatPos`), la recompensa intermedia (`IntermediateReward`), información completa de la pantalla (`MergedObservationZZ`), la información de evaluación (`infoEvaluation`), número de monedas en pantalla, número de bloques/ladrillos en pantalla y número de enemigos en pantalla.

4. Modificar el método para guardar instancias de T3HumanAgent y T3BotAgent de forma que, al crear un fichero nuevo, añada automáticamente una cabecera .arff compatible que permita a Weka abrir e interpretar el fichero de ejemplos de entrenamiento completo. La última línea que podría quedar a medio escribir debería eliminarse antes de abrir el fichero con Weka.

5. Actualizar la captura de datos "pasados" que se realizó en el Tutorial 1. Reutilizando el mecanismo ya programado, ahora se busca invertir el procedimiento para poder hacer predicciones (regresión) en las próximas prácticas del curso:

   - A partir de ahora, la línea será lo que se reserve durante unos ticks y se irá introduciendo información "futura" cuando esté disponible.
   - La información futura serán las monedas recogidas y enemigos eliminados en los próximos $n+6$, $n+12$ y $n+24$ ticks (1/4 de segundo después, 1/2 de segundo después y 1 segundo después con la velocidad del juego por defecto). Por ejemplo, para calcular las monedas recogidas en los próximos 12 ticks habrá que restar el total de monedas hasta el tick $n+12$ menos el total de monedas hasta el tick actual n. Podría ser útil generalizar esta operación para un número arbitrario de ticks futuros.
   - La Figura 3 muestra un esquema (organizado temporalmete) que representa los distintos atributos de una instancia de aprendizaje. El orden de los mismos no importa, salvo en el ya mencionado `distancePassedPhys` que para este ejercicio debe ir en último lugar.

Figura 3: Estructura de una instancia de aprendizaje (el orden de los atributos no es relevante salvo en el final `distancePassedPhys`).

2.2. Generación y preprocesado de datos de MarioAI

1. Recolectar 1000 instancias de entrenamiento con T3HumanAgent y guardar el fichero de entrenamiento como `T3HumanAgent.arff`.

2. Recolectar 1000 instancias de entrenamiento con T3BotAgent y guardar el fichero de entrenamiento como `T3BotAgent.arff`

3. Abrir estos ficheros con el Explorer de Weka. Discretizar todos los atributos en cinco clases usando el filtro no-supervisado Discretize. Grabar los datos como `T3BotAgent_Discr.arff` y `T3HumanAgent_Discr.arff`.

4. También utilizando el Explorer, eliminar todos los atributos referidos a la matriz de observación. Grabar los datos resultantes en el fichero `T3BotAgent_Discr_noObs.arff` y `T3HumanAgent_Discr_noObs.arff`.

5. Realizar otra vez el apartado anterior, pero eliminando también la información de evaluación. Grabar los datos resultantes en el fichero `T3BotAgent_Discr_noObs_noEva.arff` y `T3HumanAgent_Discr_noObs_noEva.arff`.

6. Con el fichero inicial de datos discretizados hacer una selección distinta a las anteriores en función de los atributos que los estudiantes consideren más oportunos. Grabar los datos resultantes de estas operaciones en el fichero `T3BotAgent_Discr_Selec.arff` y `T3HumanAgent_Discr_Selec.arff`. Ambos ficheres deben tener la misma selección de atributos.

7. Como resultado de este apartado se debe terminar con una serie de ficheros de datos que se introducirán en la interfaz Experimenter para hacer experimentos automáticamente con varias configuraciones de clasificadores sobre cada fichero discretizado. Los ficheros .arff generados en este subapartado son:

   - `T3BotAgent.arff`
   - `T3BotAgent_Disc.arff`
   - `T3BotAgent_Disc_noObs.arff`
   - `T3BotAgent_Disc_noObs_noEva.arff`
   - `T3BotAgent_Disc_Selec.arff`
   - `T3HumanAgent.arff`
   - `T3HumanAgent_Disc.arff`
   - `T3HumanAgent_Disc_noObs.arff`
   - `T3HumanAgent_Disc_noObs_noEva.arff`
   - `T3HumanAgent_Disc_Selec.arff`

2.3. Experimentación

1. Abrir el *Experimenter*.

2. Pulsar el botón *New* para generar un nuevo experimento.

3. Seleccionar *Classification* en el tipo de experimento.

4. Seleccionar solamente los conjuntos de datos anteriores que fueron discretizados (8 en total).

5. Seleccionar los algoritmos J48, PART, ZeroR, NaiveBayes e IbK para los valores 1, 3 y 7 de K.

6. En el apartado *Results Destination* seleccionar .csv y grabar el fichero como `e2.csv`. Este fichero contendrá los datos y resultados del experimento y se podrá abrir en una hoja de cálculo cuando termine el *Experimenter*.

7. Guardar el experimento seleccionando *Save* como tipo .xml introduciendo el nombre `e2.xml`.

8. Pulsar la pestaña *Run* y después el botón *Start*.

2.4. Análisis de los resultados

1. Pulsar la pestaña *Analyse* para analizar los resultados del experimento.

2. Pulsar el botón *Experiment* para seleccionar los resultados del experimento actual.

3. Seleccionar *Percent_ correct* en el *Comparison field*, y después seleccionar *Perform_ test*.

4. Los caracteres v y * indican si el resultado es significativamente mejor (v), peor (*) o igual ( ) que el esquema base, con el nivel de significación especificado (por omisión 0.05). Por otro lado, los números entre paréntesis (v/ /*) que aparecen debajo de cada esquema indican el número de veces que el esquema es mejor (v), igual y peor (*) que el esquema base.

5. Analizar qué resultados ha obtenido cada algoritmo en cada conjunto de datos y por cada agente, mostrando toda la información que se considere oportuna.

   Preguntas:

   a) ¿Hay algún agente que parezca más adecuado?

   b) ¿Hay algún conjunto de datos particular que parezca más adecuado?

   c) ¿Qué algoritmo parece más adecuado?

   d) ¿Son los resultados del mejor algoritmo mucho mejores que los del resto?

   e) Cambiar el criterio de comparación y comparar los resultados. ¿Los resultados guardan relación con los proporcionados en *Percent_ correct*? ¿Qué otra métrica habéis seleccionado? ¿Por qué?

   f) Generar con el *Explorer* de Weka los modelos que te parezcan más adecuados con los datos que correspondan. ¿Cuales habéis generado? ¿Son estos modelos tan adecuados como parecían? ¿Por qué?

   g) Elegir un modelo final y justificar la respuesta.

   h) ¿Por qué o para qué os parece adecuado el uso del *Experimenter* de Weka?

Normativa de entrega

1. Se debe entregar antes de la fecha límite indicada en Aula Global:
   - Memoria en formato .pdf (no se admite .doc, etc.) que contenga las respuestas a las preguntas y subpreguntas que se presentan en los ejercicios.
   - Fichero `T3HumanAgent.java`
   - Fichero `T3BotAgent.java`
   - Fichero `e1_crossvalidation.kfml`
   - Fichero `e1_suppliedtestset.kfml`
   - Fichero `T3BotAgent.arff`
   - Fichero `T3BotAgent_Disc.arff`
   - Fichero `T3BotAgent_Disc_noObs.arff`
   - Fichero `T3BotAgent_Disc_noObs_noEva.arff`
   - Fichero `T3BotAgent_Disc_Selec.arff`
   - Fichero `T3HumanAgent.arff`
   - Fichero `T3HumanAgent_Disc.arff`
   - Fichero `T3HumanAgent_Disc_noObs.arff`
   - Fichero `T3HumanAgent_Disc_noObs_noEva.arff`
   - Fichero `T3HumanAgent_Disc_Selec.arff`
   - Fichero `e2.xml`
   - Fichero `e2.csv`
2. Es obligatorio que la entrega se haga en grupos de 2 personas. No se admiten grupos de 1 ni de más de 2.
3. Solo es necesario que entregue uno de los miembros del grupo.
4. La entrega debe comprimirse en un fichero .zip (no se admite .rar, .7z, etc.) y entregarse por Aula Global. El nombre del fichero debe tener un formato equivalente al del siguiente ejemplo: `t3-387633-209339.zip` Donde los números son los 6 últimos dígitos del NIA de los alumnos.
5. Solo es necesario que entregue uno de los miembros del grupo.
6. No se admiten entregas fuera de plazo ni por email.
7. Se valorará la claridad de la memoria, el uso de tablas y en especial la justificación de las respuestas a la preguntas propuestas y conclusiones aportadas.
8. Se penalizará el uso de capturas de pantalla que contengan resultados de texto de la interfaz de Weka, así como no justificar respuestas, especialmente cuando se solicite de forma expresa.