Mario AI: Tutorial 3
Experimentación múltiple

Grado en Ingeniería Informática

Aprendizaje Automático

Aitor Alonso Núñez NIA 100346169 Gr. 83
Daniel Gak Anagrov NIA 100318133 Gr. 83

Aitor Alonso Núñez NIA 100346169 Gr. 83
Daniel Gak Anagrov NIA 100318133 Gr. 83

uc3m | Universidad Carlos III de Madrid

Índice

Aitor Alonso Núñez NIA 100346169 Gr. 83
Daniel Gak Anagrov NIA 100318133 Gr. 83

uc3m | Universidad Carlos III de Madrid

Ejercicio 1: KnowledgeFlow

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

En el nodo `TextViewer` que está unido al nodo `J48` se muestra el árbol creado por el clasificador J48. En el nodo `TextViewer` que está unido al nodo `Classifier PerformanceEvaluator` se muestra el porcentaje de instancias clasificadas correctamente y diversa información adicional como la matriz de confusión.

Por último, en el nodo `GraphViewer` que está unido al nodo `J48` se muestra una lista con los árboles de decisión que ha generado cada uno de los *folds* de la *cross validation*. Si hacemos clic sobre un elemento de esta lista nos abre la representación gráfica del árbol de decisión, con las reglas que se aplican para clasificar en cada nodo.

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

Se clasifican correctamente el 86,2105 % de las instancias (28.071 instancias bien clasificadas).

Guardar el diagrama de flujo en formato .kfml como `e1-suppliedtestset.kfml`.
c) ¿Cuál es la utilidad de crear flujos de conocimiento con esta interfaz de Weka?

Con esta interfaz podemos visualizar las distintas fases por la que pasa el estudio y filtrado de los datos, y las fases que producen el aprendizaje. Si no se utilzara esta herramienta, habría que estar repitiendo análisis, fitros, preprocesados y construcciones de modelos en el explorador de weka, mientas que al crear diferentes flujos de conocimiento, defines lineas de trabajo reutilizables permitiéndote constuir un árbol de flujos complejo, pero fácil de gestionar.

Ejercicio 2: Experimenter

Tras la ejecución del test sonre los resultado del análisis de los datos se ha obtenido la siguiente salida:

Dataset	J48	PART	ZeroR	NaiveBayes	IbK1	IbK3	IbK7
T3BotAgent_Discr	100.00	100.00	28.80 *	99.40 *	99.51 *	98.79 *	98.20 *
T3BotAgent_Discr_Selec	99.70	99.52	28.80 *	98.67 *	99.35	98.99 *	98.48 *
T3BotAgent_Discr_noObs	100.00	100.00	28.80 *	99.49 *	99.79	99.52 *	99.55 *
T3BotAgent_Discr_noObs_noEva	100.00	100.00	28.80 *	99.00 *	99.96	99.42 *	99.14 *
T3HumanAgent_Discr	99.90	99.90	25.00 *	98.89 *	99.20 *	98.48 *	96.68 *
T3HumanAgent_Discr_Selec	99.40	99.40	25.00 *	98.55 *	99.05	98.67	98.23 *
T3HumanAgent_Discr_noObs	99.90	99.90	25.00 *	99.20 *	99.30 *	99.55	99.40
T3HumanAgent_Discr_noObs_noEva	99.90	99.90	25.00 *	99.89	99.39	99.25 *	99.14 *
	(v/ /*)	(0/8/0)	(0/0/8)	(0/1/7)	(0/5/3)	(0/2/6)	(0/1/7)

Tabla 1: Test Output: Analysis result

a) ¿Hay algún agente que parezca más adecuado?

Parece que el agente *T3BotAgent* tiene un rendimiento mejor que el humano *T3HumanAgent*, pero la diferencia es irrisoria.

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

Parece ser que los mejores resultados se obtienen en los conjuntos *noObs* y *noObs_noEva* con independencia del algoritmo utilizado.

c) ¿Qué algoritmo parece más adecuado?

Puesto que en la mayoría de casos se ha empeorado respecto al caso base (utilizando J48), diríamos que el algoritmo más adecuado es pues J48. Asimismo le sigue muy muy de cerca PART, solo habiendo empeorado mínimamente para el conjunto de datos *T3BotAgent_Discr_Selec*.

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

Con salvedad de los resultados obtenidos con ZeroR, que son bastante malos en comparación al resto, los resultados obtenidos con el mejor algoritmo (J48) no son 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?

Tras probar varios criterios de comparación, podemos asegurar que aquellos que tienen que ver con el número o porcentaje de instancias clasificadas correctamente o erróneamente guardan relación con los del comparador *Percent_correct*, ya que este se basa en el número de instancias clasificadas correctamente para comparar los resultados.

Entre los otros filtros que hemos probado se encuentra *UserCPU_Time_testing*, ya que consideramos que una parte muy imporante de un algoritmo de aprendizaje automático es el tiempo utilizado para clasificar las instancias y su complejidad temporal. Hemos descubierto así que el algoritmo perezoso IbK es por norma más rápido que el resto.

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?

Hemos partido del modelo *Selec* que se nos indicó que creáramos y hemos generado y evaluado los siguientes modelos:

- El modelo *Selec* base
- Un modelo que no contiene la información observada a futuro
- Un modelo que no contiene ninguna observación de `mergeObservation()`

- Un modelo que combina los dos anteriores

Tras la evaluación, hemos obtneido los siguientes resultados:

Dataset	J48	PART	ZeroR	NaiveBayes	IbK1	IbK3	IbK7
T3BotAgent_Discr_Selec	99.70	99.52	28.80 *	98.67 *	99.35	98.99 *	98.48 *
T3BotAgent_Discr_Selec_noFut	99.70	99.52	28.80 *	99.45	99.44	98.90 *	97.78 *
T3BotAgent_Discr_Selec_noObs	99.70	99.70	28.80 *	98.79 *	99.66	99.55	98.91 *
T3BotAgent_Discr_Selec_noObs_noFut	99.70	99.70	28.80 *	99.70	99.67	99.41	98.72 *
	(v/ /*)	(0/4/0)	(0/0/4)	(0/2/2)	(0/4/0)	(0/2/2)	(0/0/4)

Tabla 2: Test Output 2: Analysis result

Como se puede observar, no mejora nada el análisis, aunque mejora el rendimiento de otros algoritmos (disminuyen frente a J48 pero menos) si se compara con los resultados obtenidos anteriormente, recogidos en la tabla 1 Test Output: Analysis result.

Creemos que estos modelos son adecuados porque obvian la posición X e Y de Mario, que aunque con ella se consigue un 100 % de aciertos, es debido a que se está produciendo un sobreajuste porque el algoritmo está aprendiendo el mapa y no realmente a jugar.

g) Elegir un modelo final y justificar la respuesta.

Si tuviéramos que usar J48 y a la vista de los resultados arrojados, probablemente nos quedaríamos con el modelo que definimos inicialmente, *T3BotAgent_Discr_Selec*. Sin embargo, Nos gustaría tomar información adicional en las instancias de entrenamiento.

No se ha incluido porque para este tutorial hemos seguido fielmente las exigencias del enunciado, incluyendo en la toma de ejemplos solo los datos solicitados en el tutorial 1 y 3. Si de nosotros dependiera, tendríamos en cuenta también las teclas que pulsa Mario o si este está en el suelo o en medio de un salto, por ejemplo, además de ciertas transformaciones extra con los datos antes de escribir a los ejemplos.

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

El experimenter nos permite trabajar con varios set de datos distintos, así como con distintos algoritmos de aprendizaje automático que nos ayuden a clasificar las instancias. Esto nos permite conocer qué sets son más representativos para el parendizaje o para qué algoritmo de aprendizaje, así como qué algoritmo u algoritmos son los más útiles o eficientes dados nuestro set de datos.