Departamento de Informática
Universidad Carlos III de Madrid

Máster en Ciencia y Tecnología Informática

Programación Automática

Examen

Normas generales del examen

- El tiempo para realizar el examen es de **2 horas**
- No se responderá a ninguna pregunta sobre el examen
- Si se sale del aula, no se podrá volver a entrar durante el examen
- No se puede presentar el examen escrito a lápiz

Problema 1. (5 puntos)

La empresa *MeGustaElJamonIberico.es* se dedica a la distribución de productos derivados del cerdo ibérico mediante un portal de venta en Internet. En concreto distribuye chorizo, salchichón, morcilla, paleta y jamón. Actualmente, dispone de una base de datos de 5000 clientes, donde para cada uno de ellos, tiene almacenado el dinero que se gastó ese cliente en cada uno de sus productos en los años 2011 y 2012. La empresa va a comenzar la distribución de un nuevo producto, el morcón, en la campaña del 2013. Para ello, pretende definir una campaña de marketing basada en el envío de correos por e-mail.

En primer lugar, pretende dividir a los clientes en distintos perfiles o grupos. Para ello, pretende aplicar alguna técnica de aprendizaje automático.

1. **(1 punto)** ¿Qué tipo de algoritmo de aprendizaje sería más apropiado? Selecciona un algoritmo de aprendizaje dentro de los de ese tipo

2. **(1 punto)** ¿Qué parámetros habría que configurar de ese algoritmo? ¿Hay alguna forma razonable de configurarlos para esta tarea?

3. **(1 punto)** ¿Cómo sería la representación de los ejemplos?: ¿Cuántos atributos hay y de qué tipo son?

En segundo lugar, para cada cliente, la empresa pretende enviarles un máximo de un correo a la semana de publicidad. Se dispone de 5 posibilidades. La primera es no enviar ningún correo, para no cansar al cliente. La segunda es enviarle un bono descuento de un 5 % para compras de morcón. El tercero es enviarle un bono descuento de un 5 % para compras de cualquier producto. El cuarto es ofrecerle un pack de productos concreto sin descuento, y el quinto es una promoción para que no tenga gastos de envío. Para cada usuario, se mantiene en todo momento el dinero que se ha gastado en cada producto durante el año 2013, y en función de eso, se pretende tomar la decisión de qué tipo de correo enviarle. Además, se aplicará la misma estrategia para todos los clientes del mismo perfil (obtenido en el apartado anterior). Se desea diseñar un esquema que permita maximizar el dinero gastado por cada cliente al cabo del año en morcón.

1. **(0.5 puntos)** ¿Qué tipo de algoritmo de aprendizaje sería más apropiado? Selecciona un algoritmo de aprendizaje dentro de los de ese tipo

2. **(0.5 puntos)** ¿Qué parámetros habría que configurar de ese algoritmo? ¿Hay alguna forma razonable de configurarlos para esta tarea?

3. **(1 punto)** ¿Cómo sería la representación de los ejemplos de aprendizaje?

Problema 2. (5 puntos)

Super Mario Bros. es un videojuego de plataformas, diseñado por Shigeru Miyamoto, lanzado el 13 de septiembre de 1985 y producido por la compañía Nintendo, para la consola Nintendo Entertainment System (NES). El juego describe las aventuras de dos fontaneros, Mario y Luigi; ambos deben rescatar a la Princesa Peach, del Reino Champiñon, que fue secuestrada por el rey de los koopas, Bowser. A través de ocho diferentes niveles de juego, los jugadores pueden controlar a alguno de los dos hermanos y deben enfrentarse finalmente tras los niveles correspondientes de cada mundo a los monstruos de cada castillo para liberar a Peach. [1]

En la versión que se plantea en este problema, se desea desarrollar un agente que controle únicamente a Mario, de forma que sea capaz de superar un determinado nivel de juego. La figura 1 muestra una escena del juego. En la escena se muestra a Mario, enmarcado en una rejilla de tamaño $N \times M$, que describe su rango de visión. En cada paso, el agente recibe la matriz, donde cada una de las celdas puede contener cuatro valores distintos: 0 si no hay nada, 1 si hay un obstáculo, 2 si hay un enemigo, y 3 si hay una moneda. En función de esta percepción, el agente debe decidir qué acción ejecutar. El agente debe proporcionar un vector de cinco posiciones binarias indicando qué acciones básicas se deben ejecutar a la vez: Moverse hacia la derecha, moverse hacia la izquierda, agacharse, saltar y correr. Por lo que existen 32 posibles combinaciones, aunque no tiene demasiado sentido contemplarlas todas. Por ejemplo, no es muy inteligente decir que en el mismo paso Mario se mueva hacia la derecha y hacia la izquierda (acción (1,1,0,0,0)), ya que el estado final sería el mismo que el inicial. Otras combinaciones de tecla sí pueden tener sentido, como saltar hacia la izquierda (acción (0,1,0,1,0)).

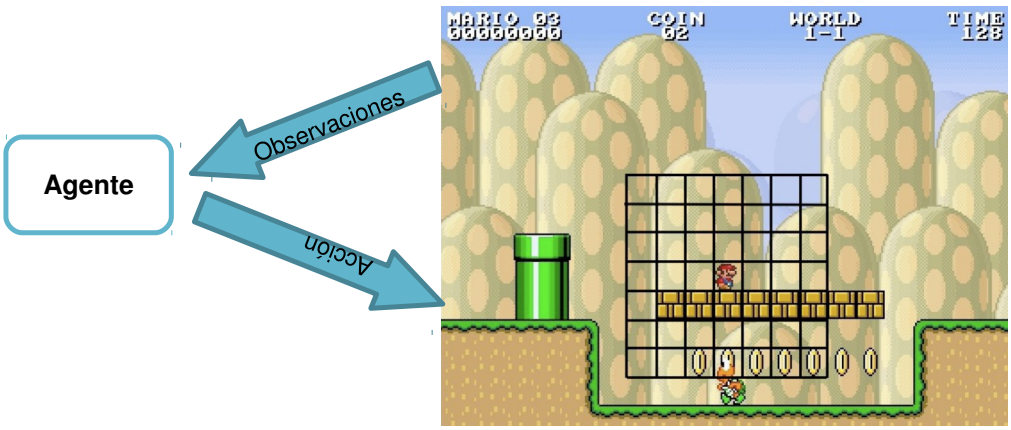

Figura 1: Imagen del juego de Mario Bros.

En este escenario, el objetivo de Mario es llegar a la meta, que está colocada en algún lugar hacia la derecha. Además, se pretende llegar a dicha meta en el menor tiempo posible, adquiriendo el mayor número de monedas posible, y sin chocar nunca contra un enemigo (pues eso mata a Mario). Por tanto, el objetivo de este ejercicio es diseñar el agente que sea capaz de resolver este problema. Para ello, se debe:

1. (1 punto) Determinar el espacio de estados y el espacio de acciones.

[1] Datos obtenidos de Wikipedia

2. Diseñar una solución a la construcción del agente controlador de Mario basada en aprendizaje automático. Para ello, define:

   - (1 punto) Qué tipo de aprendizaje automático (supervisado, no supervisado o por refuerzo) vas a utilizar y por qué. Ten en cuenta que el objetivo es llegar a la meta en el menor tiempo posible y con el mayor número de monedas
   - (1 punto) Qué algoritmo concreto planteas, y por qué. Define todos los elementos necesarios para poder ejecutar dicho algortimo. ¿Cómo sería el proceso de aprendizaje?
   - (0,5 puntos) Cómo son los datos de entrenamiento
   - (0,5 puntos) Cómo obtener los datos de entrenamiento
   - (1 punto) Cómo evaluar el agente obtenido

1. Solución al ejercicio 1

1. Dado que es un problema donde se desea agrupar a los clientes, se utiliza una técnica de agrupación o *clustering*. De esta forma, los usuarios quedarán organizados y agrupados en función de sus gastos a través del portal. En cuanto a los algoritmos concretos, se puede utilizar cualquier algoritmo de agrupación, como *k-medias*, *E-M*, o la construcción de dendogramas con las técnicas aglomerativas. En este caso, aplicaremos *k-medias*.

2. El principal parámetro de este algoritmo es el número de clusters o k. El número de grupos a generar dependerá del número de perfiles de usuario que se deseen. Además, se puede configurar la medida de distancia a utilizar. Habitualmente, y sin tener más conococimiento, se puede usar la distancia euclídea ponderada, normalizando previamente los datos.

3. Los ejemplos se pueden representar de forma tabular, tal y como se muestra en la Tabla 1. Con esta

2011-chor	2011-salchi	2011-morci	2011-pale	2011-jam	2012-chor	2012-salchi	2012-morc	2012-pale	2012-jam
120	23	54	45	234	45	124	345	100	0
34	46	435	45	333	546	33	345	32	45
...									

Cuadro 1: Ejemplos de entrenamiento del problema 2: compras realizadas por los clientes

   representación, cada cliente está representado con una lista de 10 atributos, correspondientes a las compras realizadas en los años 2011 y 2012 de cada producto.

4. El problema de el envío de correos electrónicos, una vez por semana, se puede entender como un problema de toma de decisiones secuencial, y en este caso se puede modelar mediante un Proceso de Decisión de Markov. Por tanto, la técnica de aprendizaje a utilizar es el aprendizaje por refuerzo. Se podría utilizar gran variedad de algoritmos. En este caso, planteamos el uso de *Q-Learning*.

5. Los principales parámetros del algoritmo *Q-Learning* son el parámetro de descuento, γ, el ratio de aprendizaje α y la estrategia de exploración y explotación. En principio, no se puede predecir qué valores serían los más adecuados, por lo que se pueden usar valores clásicos, por ejemplo, de $\gamma = 0,9$ y $\alpha = 0,3$. Para α se podría utilizar un valor más alto que fuera decayendo según se van adquiriendo nuevos ejemplos de entrenamiento y se van haciendo más actualizaciones en la tabla Q. Para la estrategia de exploración tampoco se tiene un conocimiento del dominio que permita definir cuál sería la mejor opción. Por tanto, se podría comenzar con cualquiera, como una ϵ-greedy, con un $\epsilon = 0$ y que va incrementándose en cada iteración.

6. En *Q-Learning*, los ejemplos de entrenamiento vienen dados por las tuplas de experiencia, que a la vez dependen de la representación del MDP subyacente:

   - Estado: viene representado por un vector que indica las compras acumuladas por el cliente en el año 2013:
     $$2013 - chor, 2013 - salchi, 2013 - morci, 2013 - pale, 2013 - jam, 2013 - morcon$$
   - Acciones: las cinco acciones definidas: no mandar nada, enviar bono descuento morcón, enviar bono descuento general, enviar información de productos, enviar bono regalo gastos envío.
   - Función de transición de estados: desconocida, puesto que no sabemos qué efecto en las compras de los clientes tendrá cada acción
   - Función de refuerzo: Dado que la variable a maximizar es la compra de morcón, como función de refuerzo se puede utilizar el gasto realizado por el cliente en ese producto desde que se ejecutó la acción hasta la siguiente decisión.

   Dado que se asume que cada perfil de usuario se va a comportar de forma parecida, se aprenderá una única política (por tanto, una única tabla Q) para cada perfil. Por tanto, una tupla de experiencia contendría:

   - Estado: por ejemplo, $< 100, 230, 24, 0, 45, 10 >$
   - Acción: por ejemplo, enviar bono descuento morcón
   - Estado siguiente: por ejemplo, $< 100, 230, 24, 0, 45, 30 >$ (el cliente ha comprado morcón por valor de $30 - 10 = 20$ euros
   - Refuerzo inmediato: 20

2. Solución al ejercicio 2

1. El espacio de estados se compone de todas las posibles combinaciones de estados que Mario puede percibir. En principio, dado que en cada momento recibe una matriz de tamaño $M \times N$, y que el número de posibles valores de cada elemento de la matriz es 5, su tamaño es de $5^{M \times N}$. Sin embargo, es fácil ver que no todas estas posibles combinaciones del espacio de estado se darán en el juego, por lo que en la realidad probablemente se percibirán muchos menos estados.

2. El espacio de acciones se compone de todas las posibles combinaciones de acciones que se pueden ejecutar, es decir, 2^4. Sin embargo, como bien dice el enunciado, no todas estas combinaciones tienen sentido, por lo que habría que hacer un análisis a priori de cuál es el número de accioens que de verdad tienen sentido, para así limitar el valor anterior a un número más pequeño.

3. En este caso se pueden plantear varios tipos de aprendizaje. Por ejemplo, el aprendizaje supervisado se podría plantear desde el punto de vista de aprendizaje por instrucción; es decir, un humano podría jugar un número suficientemente alto de partidas, capturar los pares estado-acción ejecutados, y aprender un modelo que, dado cualquier estado, defina qué acción hay que ejecutar. Sin embargo, este modelo adquirido sería una copia del comportamiento del humano, que no tiene por qué cumplir los objetivos planteados, es decir, no tiene por qué llegar a la meta en un tiempo mínimo adquiriendo el mayor número de monedas posible. Es decir, este tipo de aprendizaje no optimiza la función objetivo (cualquiera que sea). Otra posible opción es el aprendizaje por refuerzo, que sí permite optimizar alguna función de refuerzo. Sería, por tanto, un aprendizaje por prueba y error.

4. Se podría utilizar el algoritmo Q-Learning. Se utilizará la función de actualización de Q-Lerning estocástica, puesto que el dominio no es determinista. Para ello, debemos modelar el problema como un Proceso de Decisión de Markov:

   - Espacio de estados: definido anteriormente
   - Espacio de acciones: definido anteriormente
   - Función de transición de estados: desconocida, pero dada por el comportamiento pre-programado del juego
   - Función de refuerzo. Se podrían usar distintas funciones de refuerzo. Sea la que fuese, debería premiar cuando se coge una moneda (refuerzo +5), y cuando se llega a la meta (refuerzo +100). Se podría penalizar (con un refuerzo de -100) cuando Mario se choca con un monstruo. Distintas funciones de refuerzo pueden generar distintas políticas, por lo que se debe buscar una función de refuerzo que pondere de forma adecuada el tiempo que se tarda en llegar a la meta y la recolección de monedas, que son los dos parámetros a optimizar.
   - El resto de parámetros de aprendizaje, número de episodios, número de pasos por episodio, γ y α, se fijarían tras un estudio empírico. El proceso de aprendizaje constaría de un proceso de prueba y error, en el que se ejecutarían múltiples episodios.

5. Los datos de entrenamiento se componen de tuplas de experiencia, del tipo $< s, a, s', r >$, donde s es un estado, a es la acción que se ha ejecutado, s' sería el estado al que se llega al ejecutar a desde s, y r sería el refuerzo recibido tras esa transición.

6. Las tuplas de experiencia se recopilarían tras un clásico proceso de prueba y error. Habría que definir una estrategia de exploración, en la que podría ayudar un humano para guiar a Mario y lograr episodios en los que efectivamente llegue a la meta. La tabla Q se podría inicializar a cero, para todos los pares estado-acción.

7. La evaluación se realizaría ejecutando la política aprendida, y obteniendo el rendimiento de dicha política, es decir, el refuerzo acumulado a lo largo del episodio. Con ese valor, se pueden comparar distintas políticas para determinar cuál es la mejor.