Examen intermedio del curso Inteligencia Artificial

Julio Waissman, 2025a

Para hacer de tarea

Agentes inteligentes

Un equipo de investigadores está diseñando un agente de inteligencia artificial para gestionar emergencias en un hospital. Este agente debe asignar recursos médicos (como personal, camas y equipos) en función de la gravedad de los pacientes y la disponibilidad de recursos.

1. Describe el problema utilizando el método de PEAS (MEAS) y porqué podría ser un problema a realizar con algoritmos de inteligencia artificial.
2. Explica qué tipo de agente racional sería el más adecuado para este escenario (reactivo, basado en objetivos, basado en utilidad o de aprendizaje) y justifica tu elección considerando las características del entorno hospitalario.
3. Supón que el agente debe decidir entre asignar el último respirador disponible a un paciente joven con baja probabilidad de supervivencia o a un paciente mayor con una mayor probabilidad de recuperación. ¿Cómo debería tomar la decisión un agente racional? ¿Crees que una decisión basada solo en una función de utilidad sería suficiente o deberían incorporarse otros criterios? Argumenta tu respuesta considerando aspectos éticos y limitaciones de la IA.

Árboles de decisión

La empresa *Chinchesoft* está desarrollando un modelo de clasificación basado en un *bosque aleatorio* para predecir si un cliente realizará una compra en un sitio web, basado en variables como el tiempo en la página, la cantidad de clics y el historial de compras.

Dado que un bosque aleatorio está compuesto por múltiples *árboles de decisión*, la forma en que se construyen estos árboles influye en la capacidad del modelo para generalizar bien a datos nuevos.

1. En cada nodo de un árbol dentro del Bosque Aleatorio, se selecciona aleatoriamente un subconjunto de características para decidir el mejor

punto de división. ¿Por qué esta estrategia mejora la generalización del modelo en comparación con entrenar un solo árbol de decisión sin restricciones?

2. Supongamos que entrenas dos versiones del Bosque Aleatorio:
   - *Versión A:* Árboles con profundidad ilimitada.
   - *Versión B:* Árboles con una profundidad máxima de 5 niveles.

   ¿Cómo afectará esta diferencia a la capacidad de generalización y el riesgo de sobreajuste del modelo?

3. Si aumentamos el número de árboles en el bosque, ¿cómo cambia el desempeño del modelo? ¿Existe un punto en el que agregar más árboles ya no mejora la precisión? Explica tu respuesta con base en la teoría estadística del aprendizaje vista en clases.

4. Si los datos del sitio web tienen un fuerte sesgo hacia clientes frecuentes (es decir, la mayoría de los ejemplos de compra provienen de clientes recurrentes), ¿qué ideas tienes que podrías aplicar para asegurarte de que el modelo no genere predicciones sesgadas hacia este grupo?

Regresión lineal

En la regresión lineal estándar, la función de pérdida más utilizada es el **Error Cuadrático Medio (MSE)**:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \hat{y}^{(i)})^2$$

donde

$$\hat{y}^{(i)} = \sum_{j=1}^{n} w_j x_j^{(i)} + b = w^T x^{(i)} + b$$

Sin embargo, esta función de pérdida es sensible a valores atípicos (*outliers*), ya que penaliza los errores grandes de manera cuadrática. Una alternativa es utilizar la **pérdida de Huber**, que combina MSE y el error absoluto medio (MAE) para ser más robusta ante valores atípicos:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} L_\delta(y_i, \hat{y}_i)$$

donde la función de pérdida L_δ se define como:

$$L_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & \text{si } |y - \hat{y}| \leq \delta \\ \delta(|y - \hat{y}| - \frac{1}{2}\delta), & \text{si } |y - \hat{y}| > \delta \end{cases}$$

donde δ es un parámetro que controla el umbral a partir del cual los errores dejan de penalizarse cuadráticamente.

1. Explica cómo la pérdida de Huber mejora la regresión lineal en presencia de valores atípicos en comparación con MSE. ¿En qué situaciones podría no ser conveniente usar Huber?
2. Analiza la derivada de la función de pérdida de Huber y encuentra $\partial L_\delta((y, \hat{y}))/\partial w_j$ y $\partial L_\delta((y, \hat{y}))/\partial b$
3. ¿Cómo se podría implementar un algoritmo de descenso de gradiente para minimizar esta nueva función de pérdida? Explica qué modificaciones serían necesarias en comparación con un modelo que usa MSE.

Búsquedas

Un robot se encuentra en un laberinto cuadrado de dimensión $N \times N$ y debe recoger cuatro banderines ubicados en distintas posiciones antes de llegar a la meta. Puede moverse en las cuatro direcciones cardinales (arriba, abajo, izquierda, derecha), pero algunas casillas están bloqueadas por paredes. El objetivo es encontrar la ruta más corta que le permita recoger todos los banderines y llegar a la meta. El robot puede iniciar en cualquier posición del laberinto.

Para hacerlo más interesante, en el laberinto hay un destructor de robot con una ruta preprogramada, que inicia en el estado s_0, y después de m movimientos regresa a su posición inicial y se repite en ciclos. Y para que esto sea un poco más interesante, una vez que un robot recoge un banderin, por 10 movimientos va a ser inmune al destructor de robots.

1. Define claramente cómo representar el estado del problema. Calcula la cardinalidad del espacio de estados en función del tamaño del laberinto.
2. Propón dos heurísticas admisibles para este problema. Justifica por qué son admisibles y si una es dominante sobre otra. No es necesario una demostración formal, solo un esbozo.
3. Una de las limitaciones del algoritmo A* es que su rendimiento puede verse afectado en laberintos grandes con múltiples objetivos (como en este caso, donde el robot debe recoger cuatro banderines antes de llegar a la meta). Se te pide ahora proponer y justificar una modificación al algoritmo A* que pueda mejorar su eficiencia en este problema sin comprometer la optimalidad de la solución. Algunas ideas que podrías considerar incluyen:
    - En lugar de buscar una única meta, adapta A* para manejar múltiples objetivos intermedios (los banderines) de manera más eficiente. ¿Cómo podrías modificar la función de evaluación para priorizar la

recolección de banderines sin explorar innecesariamente grandes partes del laberinto?

- Introducir un criterio para descartar ciertos caminos de la búsqueda si parecen ser significativamente más largos que otros. ¿Cómo podrías hacerlo sin perder soluciones óptimas?
- Algunas variantes de A* ajustan la heurística a medida que el entorno va cambiando (por ejemplo, que el robot vaya recogiendo banderines). ¿Cómo podrías diseñar una heurística adaptable que mejore la eficiencia sin volverse inconsistente?

Explica la modificación que propones y analiza su impacto en términos de tiempo de cómputo, uso de memoria y optimalidad de la solución.

Búsqueda con adversarios

El algoritmo de poda alpha-beta se usa principalmente en juegos de dos jugadores con información perfecta, como el ajedrez o el tic-tac-toe. Sin embargo, en juegos con más de dos jugadores, como el tres en raya para tres personas o ciertos juegos de estrategia por turnos, la poda alpha-beta en su forma clásica no se aplica directamente.

1. Explica por qué la poda alpha-beta tradicional no funciona bien en juegos con más de dos jugadores. ¿Qué problemas surgen al evaluar nodos en un árbol de juego donde hay más de un oponente?
2. Diseña una versión del algoritmo alpha-beta que pueda aplicarse a juegos con tres o más jugadores. Algunas ideas que podrías considerar incluyen:
   - Adaptar la función de minimización para manejar múltiples adversarios en lugar de un solo "min" como en juegos de dos jugadores.
   - Implementar un esquema de poda que descarte ramas que son claramente peores para el jugador actual, pero considerando que hay más de un adversario con distintos objetivos.
   - Redefinir los valores alpha y beta para cada jugador en lugar de un único límite de poda.
3. Pon un ejemplo sencillo en papel que ilustre como funcionaría el algoritmo.