Preguntas examenes otros años

Explica brevemente los motivos por los que Python es un entorno relevante para el desarrollo de proyectos de Inteligencia Artificial, aportando observaciones tanto de sus ventajas como inconvenientes.

Python es un lenguaje de programación interpretado que busca desarrollar una sintaxis que priorice la legibilidad del código. Entre las ventajas generales de Python estaría que se trata de un lenguaje multiparadigma ya que permite programación orientada a objetos, programación imperativa y funcional. En cuanto a los motivos de uso en el campo de la inteligencia artificial destaca la sencillez en su interpretación, la posibilidad de realizar iteraciones rápidas de los datos, lo cual favorece que su uso se centre en los datos y desarrollo de algoritmos. Además, una de sus mayores virtudes es la inmensa cantidad de librerías que existen como por ejemplo NumPy o ScyPy y la capacidad de combinar las librerías entre ellas. Otras liberias muy útiles en el campo de la inteligencia artificial son Scikit-learn para aprendizaje automático, openCV para la percepción computacional, Matplotlib para la generación de gráficos, TensorFlow para redes neuronales, etc. Además, actualmente existen multitud de entornos para desarrollar software en Python, tanto en la nube como en local (jupyter notebooks). Respecto a sus desventajas se puede destacar su ejecución más lenta respecto a otros lenguajes no interpretados o el hecho de que no todos los servicios de hosting están preparados para soportar aplicaciones en Python.

Describe brevemente en qué consisten las bibliotecas NumPy y Pandas, la diferencia entre ellas y su relevancia en los proyectos de Inteligencia Artificial.

Numpy es una librería para el lenguaje de programación de Python que da soporte para la creación de vectores y matrices multidimensionales además de una gran cantidad de funciones que permiten operar con ellas. Una de las mayores virtudes de NumPy es su velocidad para los cálculos, y es por ello que ha servido como base para crear otras librerías muy utilizadas en inteligencia artificial. Por otro lado, Pandas es un paquete de Python que proporciona estructuras de datos similares a los dataframes del lenguaje R. Pandas depende en gran medida de la librería Numpy. Entre sus herramientas, destaca la lectura y escritura en distintos formatos como CSV, SQL o Excel; el filtrado de datos en tablas en función de su posición, valor o etiquetas; la fusión de datos; manipulación de series temporales y hacer graficas.

Ambas se tratan de dos liberias muy utilizadas en el campo de la inteligencia artificial, sobre todo para la ciencia de datos ya que permiten operar y estructurar los datos de forma muy sencilla.

Dado un conjunto de datos personales, demográficos y fiscales de varios usuarios, explica cómo procederías para anonimizarlos de forma fiable que no permita la identificación inversa.

Para empezar, en cuanto a los datos personales se eliminarían aquellos que pueden identificar al individuo directamente como pueden ser el nombre, apellidos o el DNI. Para datos como la edad se puede desnaturalizar, de forma que la edad venga representada por una etiqueta que corresponda a un determinado rango de edad. En cuanto a los datos demográficos se podría disociar parte de la información, lo cual consistiría en eliminar parte de los datos para evitar la identificación personal. Por ejemplo, se podría fijar la residencia de una persona indicando la provincia donde vive en lugar del pueblo o ciudad, que para aquellos de pequeño tamaño podría posibilitar el encontrar al usuario. Para los datos fiscales, como por ejemplo el número de la seguridad social o algún dato bancario se podrían anonimizar utilizando una función hash, la cual garantiza la no reversibilidad.

En el contexto de la anonimización de datos personales, explica brevemente la diferencia entre tokenizar y aplicar una función hash.

Tanto la tokenización como el uso de funciones hash son procedimientos utilizados para la anonomización de datos. La principal diferencia entre tokenizar y aplicar una función de hash es que para la primera es posible revertir el proceso si se dispone del token correspondiente a cada valor. Para las funciones de hash el proceso no es reversible, aunque podría pasar que se generara el mismo hash para distintos valores de entrada.

En el contexto de los agentes inteligentes, describe brevemente los componentes del acrónimo REAS (Rendimiento, Entorno, Actuadores, Sensores) y la relación entre ellos.

Los componentes del acrónimo REAS componen lo que se denomina entorno de trabajo, lo cual implica evaluar si una tarea se está realizando correctamente, que información se recoge del entorno, cuales son las posibles acciones y que tipos de sensores se va a utilizar. Por ejemplo, si utilizamos un agente que permita caminar a un robot el rendimiento se podría medir mediante los metros avanzados sin caerse, los rangos de los ángulos girados en las articulaciones, el consumo de baterías, etc. El entorno sería donde localizamos al robot, por ejemplo, en un terreno llano asfaltado. Los actuadores serían las posibles acciones que puede realizar el robot, por ejemplo, moverse hacia adelante, hacia atrás, saltar, agacharse, etc. Por último los sensores permitirían al sistema recibir información del entorno mediante por ejemplo una cámara de videos, acelerómetros,etc.

Estos componentes se encuentran íntimamente relacionados en el funcionamiento del agente, siendo necesario una integración entre ellos para conseguir el éxito. En el ejemplo del robot que camina, será necesario medir del entorno la información necesaria utilizando los sensores para decidir la acción que produzca un mejor rendimiento en la tarea buscada.

Si programamos una IA para un equipo de futbol digital siguiendo el paradigma de los agentes inteligentes (con un agente para cada jugador), describe el problema en términos del contexto REAS

Los componentes del acrónimo REAS (Rendimiento, Entorno, Acciones, Sensores) componen lo que se denomina entorno de trabajo. En el caso de un agente inteligente que controla a un jugador de fútbol, el rendimiento podría medir parámetros como la velocidad que aplica al jugador cuando corre, la proporción de regates o recuperaciones realizadas con éxito o la tasa de éxito en los disparos a puerta. El entorno correspondería al mundo donde se mueve el agente, que en este caso sería un campo de fútbol, el cual se encuentra delimitado. Dependiendo de la posición del futbolista, el agente debería lograr situar al futbolista en su posición en el campo. Además, en este caso es importante destacar que se tratará de un entorno donde coexistirán multitud de agentes. En cuanto a las acciones, en un juego de futbol además del movimiento a diferentes velocidades, el agente podría pasar el balón a otro agente, disparar a portería, realizar una entrada, etc. Por último, en cuanto a sensores se podría usar una barra de resistencia para medir el esfuerzo realizado y el cansancio del jugador al realizar las diferentes acciones posibles.

¿Qué aplicaciones, ventajas y desventajas tienen los enjambres (swarms) de Agentes Inteligentes frente sistemas inteligentes individuales?

Los enjambres son una rama de la inteligencia artificial que estudia el comportamiento colectivo de los sistemas descentralizados, autoorganizados, naturales o artificiales. Estos sistemas están inspirados en sistemas biológicos y se encuentran formados por una población de agentes

simples que interactúan localmente entre ellos y con su medio ambiente. Los agentes siguen reglas simples y aunque no exista una estructura de control centralizado, las interacciones locales entre los agentes conducen a un comportamiento global complejo. Una de las principales aplicaciones son los algoritmos de colonias de hormigas que permiten encontrar el camino más corto entre dos puntos de forma automática a través de la computación de un equivalente a las feromonas expulsadas por las hormigas reales. Entre las ventajas frente a agentes individuales destacan por ejemplo la posibilidad de realizar una tarea compleja a través de varias tareas simples y como desventaja que, en general, la programación de estos agentes que interaccionan entre ellos es más compleja que la de un agente individual.

¿Por qué hacemos búsquedas heurísticas? Explica brevemente la motivación para usarlas y aporta algún ejemplo de técnica de búsqueda heurística.

La heurística se puede definir como la manera de alcanzar la solución de problemas a través de la evaluación de los progresos alcanzados durante la búsqueda del resultado definitivo. En inteligencia artificial la heurística representa el conocimiento extraído de la experiencia dentro del dominio del problema. Se utilizan por ejemplo en algoritmos evolutivos como último recurso para resolver un problema cuando los algoritmos convencionales no funcionan o tienen un coste demasiado alto. Como ejemplo de algoritmo de búsqueda heurística se puede destacar el algoritmo A* para problemas con coste no uniforme y de los que desconocemos a priori el coste de llegar al estado meta. Otro tipo de heurísticas son las búsquedas por subobjetivos, las cuales tratan de reducir la complejidad de los algoritmos de búsqueda.

Explica la diferencia entre los algoritmos voraces y la búsqueda con backtracking. Ejemplifícalo con una discusión sobre cómo enfocarías el problema clásico de la mochila.

Indica distintas formas (al menos 3) en las que los servicios cognitivos de IBM Watson pueden ayudar a un proyecto de aprendizaje automático con Big Data.

En primer lugar, la computación en la nube permite entrenar modelos mucho más complejos venciendo las limitaciones de rendimiento que puede tener un ordenador personal. En segundo lugar, Watson cuenta con multitud de algoritmos de aprendizaje automático ya diseñados como por ejemplo distintas arquitecturas comunes en los entrenamientos con redes de neuronas profundas. Esto permite a los desarrolladores acelerar la creación de software. En tercer lugar, destaca IBM Watson cuenta con Watson Analytics para la gestión de grandes volúmenes de datos (Big Data)

Describe en qué consiste el algoritmo MINIMAX y sus campos de aplicación. Aporta un ejemplo

El algoritmo minimax es un algoritmo de búsqueda multiagente que permite minimizar la pérdida máxima esperada en juegos con adversarios y con información perfecta. Se trata de un algoritmo recursivo cuyo funcionamiento se resumen en elegir el mejor movimiento para uno mismo partiendo de la suposición de que el contrincante escogerá el peor para el adversario. Este algoritmo tiene algunas variantes como minimax con poda alfa-beta, que permite reducir la complejidad del árbol o expectminimax que añade una componente de azar.

El algoritmo minimax se puede utilizar para juegos donde se quiere enfrentar a una máquina contra un usuario, como por ejemplo el tres en raya o el ajedrez. Otro ejemplo de aplicación es su uso aplicado a vigilancia con robots móviles, como se describe en el artículo:

Vera S., Jiménez-González A., Heredia G., Ollero A. Algoritmo Minimax Aplicado a Vigilancia Con Robots Móviles. Proceedings of the 3rd Spanish Experimental Robotics Conference (ROBOT2011); Seville, Spain. 28–29 November 2011.