



Universidad Nacional del Nordeste

Facultad de Ciencias Exactas y Naturales y Agrimensura

Licenciatura en Sistemas de Información



Asignatura Inteligencia Artificial

Profesora: Dra. Sonia I. Mariño, Lic. Jaquelina Escalante

Alumno: Gonzalo Daniel Ramirez

LU:56838

DNI:44543439

gonza37754@gmail.com

Año: 2025.

Introducción

El problema específico que aborda este trabajo es la optimización de la toma de decisiones al momento de elegir un lugar de estacionamiento en la ciudad de Corrientes capital. Se busca resolver la necesidad de filtrar y priorizar opciones dentro de un conjunto de datos espaciales, considerando que no todos los estacionamientos tienen la misma "utilidad" para el usuario.

El alcance de este proyecto consiste en el desarrollo de un Agente Inteligente Racional basado en Utilidad implementado como una API REST.

- **Lo que incluye:** El sistema recibe la geolocalización del usuario y sus preferencias ponderadas, procesa un dataset de estacionamientos medidos y devuelve un ranking de las 3 mejores opciones. El alcance técnico abarca la implementación de algoritmos de búsqueda espacial eficiente (KDTree) y el diseño de una función de evaluación heurística.
- **Delimitación:** La solución propuesta opera sobre un entorno estático y determinista (basado en un dataset csv precargado).

Este trabajo se enmarca en la Inteligencia Artificial Simbólica y el diseño de Agentes Racionales. Específicamente, se implementa un Agente basado en Utilidad, el cual utiliza una función de evaluación para maximizar el rendimiento (score) basado en múltiples criterios (distancia, disponibilidad, tipo). Además, se aplican conceptos de Búsqueda Heurística para optimizar la recuperación de candidatos en el espacio de estado.

Se utiliza un dataset real ("Estacionamiento-medido.csv") proporcionado por el portal de datos abiertos de la ciudad de Corrientes [1], validando la eficacia del agente mediante pruebas de funcionalidad (endpoints de la API) y la coherencia de las recomendaciones frente a distintas configuraciones de pesos (w) proporcionados por el usuario.

Método

La construcción del IA sigue la definición teórica donde

Agente = Arquitectura + Programa.

- **Arquitectura:** Se utiliza una arquitectura de microservicios que conecta el programa con los sensores (endpoints HTTP) y actuadores (respuestas JSON)
- **Programa:** Se implementa un algoritmo de razonamiento basado en utilidad que manipula la entrada para producir una salida óptima.

El modelo de Inteligencia Artificial seleccionado es un Agente Racional Basado en Utilidad.

Se ha descartado el uso de un Agente Reactivo Simple (que actúa solo por reglas condición-acción directas) o un Agente Basado en Metas (que solo busca alcanzar un estado final), debido a que el problema requiere no solo encontrar un estacionamiento ("la meta"), sino encontrar el mejor estacionamiento posible según las preferencias del usuario.

Este modelo permite tomar decisiones racionales en situaciones donde existen múltiples alternativas para alcanzar un objetivo, permitiendo evaluar qué tan buenas son sus acciones mediante una función de utilidad y elegir la opción más valiosa. La racionalidad del agente consiste en seleccionar la acción que maximice su medida de rendimiento (el score calculado) basándose en las evidencias proporcionadas (ubicación y preferencias) y el conocimiento almacenado (dataset).

El modelo opera procesando un conjunto definido de variables que constituyen el conocimiento del entorno y las percepciones del agente:

1. **Variables de Estado** (Base de Conocimiento): Representan el conocimiento explícito del agente sobre el entorno, obtenido del dataset `Estacionamiento-medido.csv`
 - a. *Plat, Plon*: Geolocalización del estacionamiento (centroide).
 - b. *Ldisp*: Lugares disponibles (normalizado en el código como `lugares_0.norm`).
 - c. *Igarage*: Variable binaria que indica si es techado (1) o no (0)
2. **Variables de Percepción (Entrada)**: Información captada por los "sensores" del agente en cada interacción
 - a. *Ulat, Ulon*: Ubicación del usuario.
 - b. *Wdist, Wlug, Wgar*: pesos de preferencia definidos por el usuario para ponderar, distancia, disponibilidad y tipo de estacionamiento
3. **Función de Utilidad (Heurística)**: Función de evaluación que asigna un valor real a cada estado candidato. En el código, esto se modela matemáticamente como:

$$U(s) = Wdist * f(distancia) + Wlug * f(lugares) + Wgar * f(garage)$$

El modelo asume un entorno Totalmente Observable el agente tiene acceso a todo el dataset cargado Estático los datos no cambian durante el cálculo de la recomendación y Discreto cantidad finita de estacionamientos, lo que simplifica el diseño del agente en comparación con entornos estocásticos o dinámicos.

El sistema utiliza conocimiento explícito se sabe que se tiene y es consciente cuando se lo ejecuta, estructurado en el DataFrame de Pandas, el cual es manipulado computacionalmente para inferir la mejor recomendación.

Herramientas

Para la implementación del agente se utilizó el lenguaje de programación Python, seleccionado por su amplia disponibilidad de librerías para el cálculo científico y manejo de datos espaciales. El entorno de ejecución se basa en una arquitectura cliente-servidor implementada mediante el framework FastAPI [2].

Librerías utilizadas en el proyecto

- FastAPI: Utilizada para exponer la interfaz del agente al entorno (sensores/actuadores).
- Scipy [3] (módulo spatial): Fundamental para la eficiencia de la búsqueda en el espacio de estados.
- Pandas [4]: Empleada para la gestión de la Base de Conocimiento (BC).
- NumPy [5]: Utilizada para operaciones vectoriales.

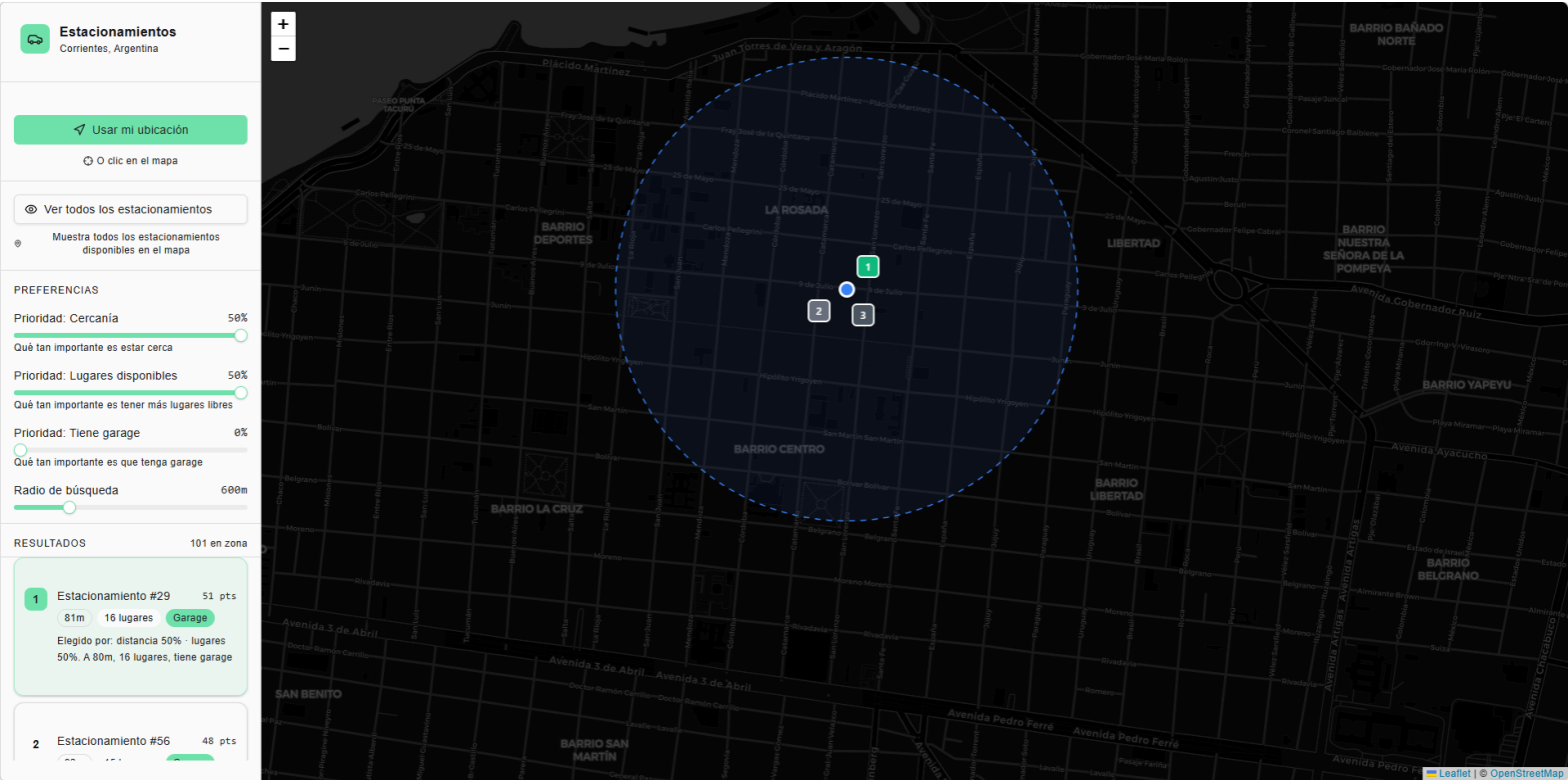
La validación no se basa en aciertos/errores, sino en verificar que el agente actúe racionalmente maximizando su medida de rendimiento según las preferencias indicadas.

Resultados

Para validar el comportamiento racional del Agente Basado en Utilidad, se diseñaron tres escenarios experimentales manteniendo constantes las variables de entorno (Ubicación del usuario fija y Radio máximo de búsqueda $R=600m$) y variando exclusivamente los pesos de preferencia en la función de utilidad.

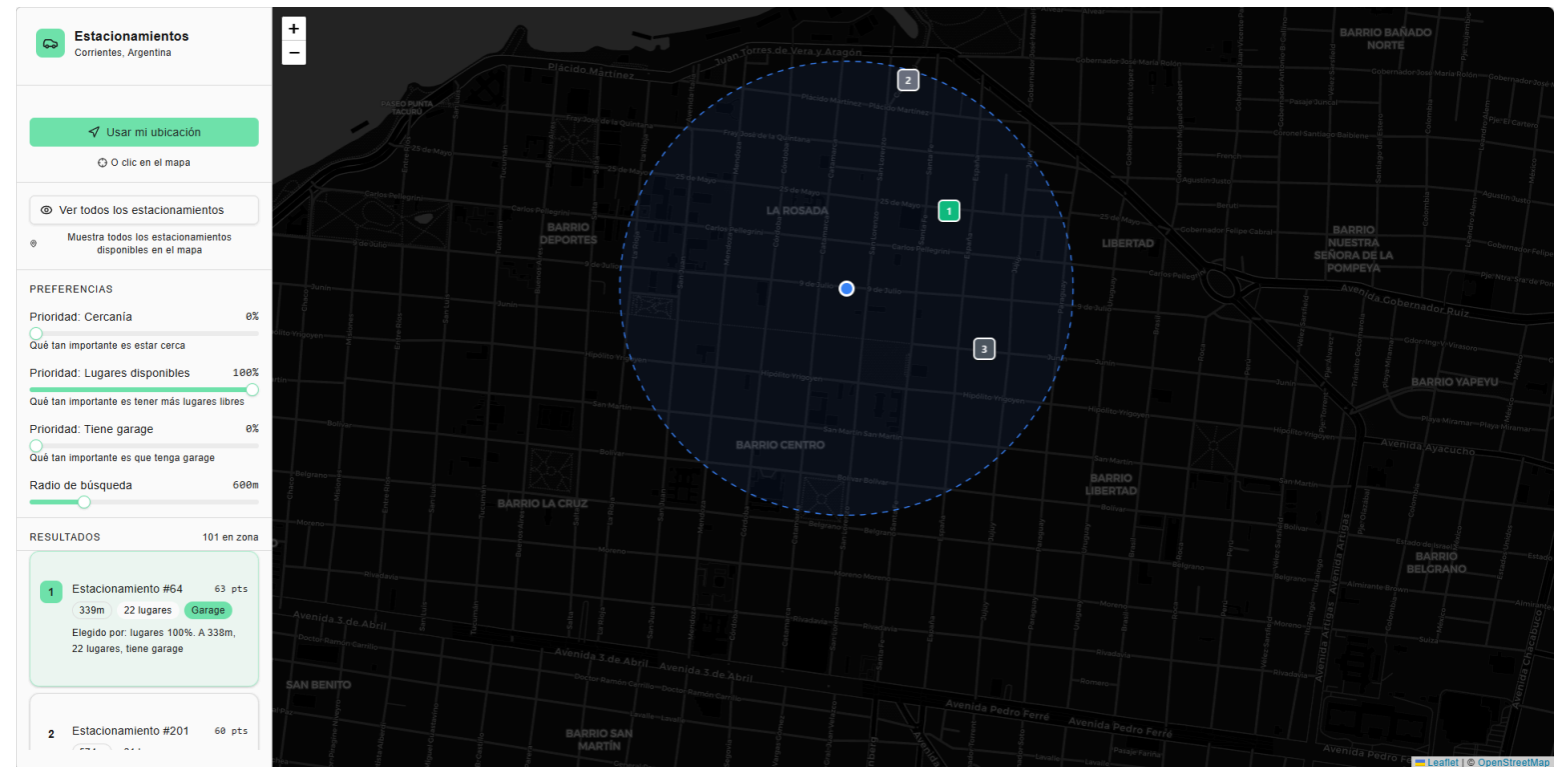
Prioridad del Usuario: Priorización máxima de cercanía y lugares disponibles sin importancia en la prioridad de garage.

Comportamiento observado: El agente deberá recomendar estacionamientos cercanos que tengan una cantidad considerable de lugares, ignorando si es garage o no.



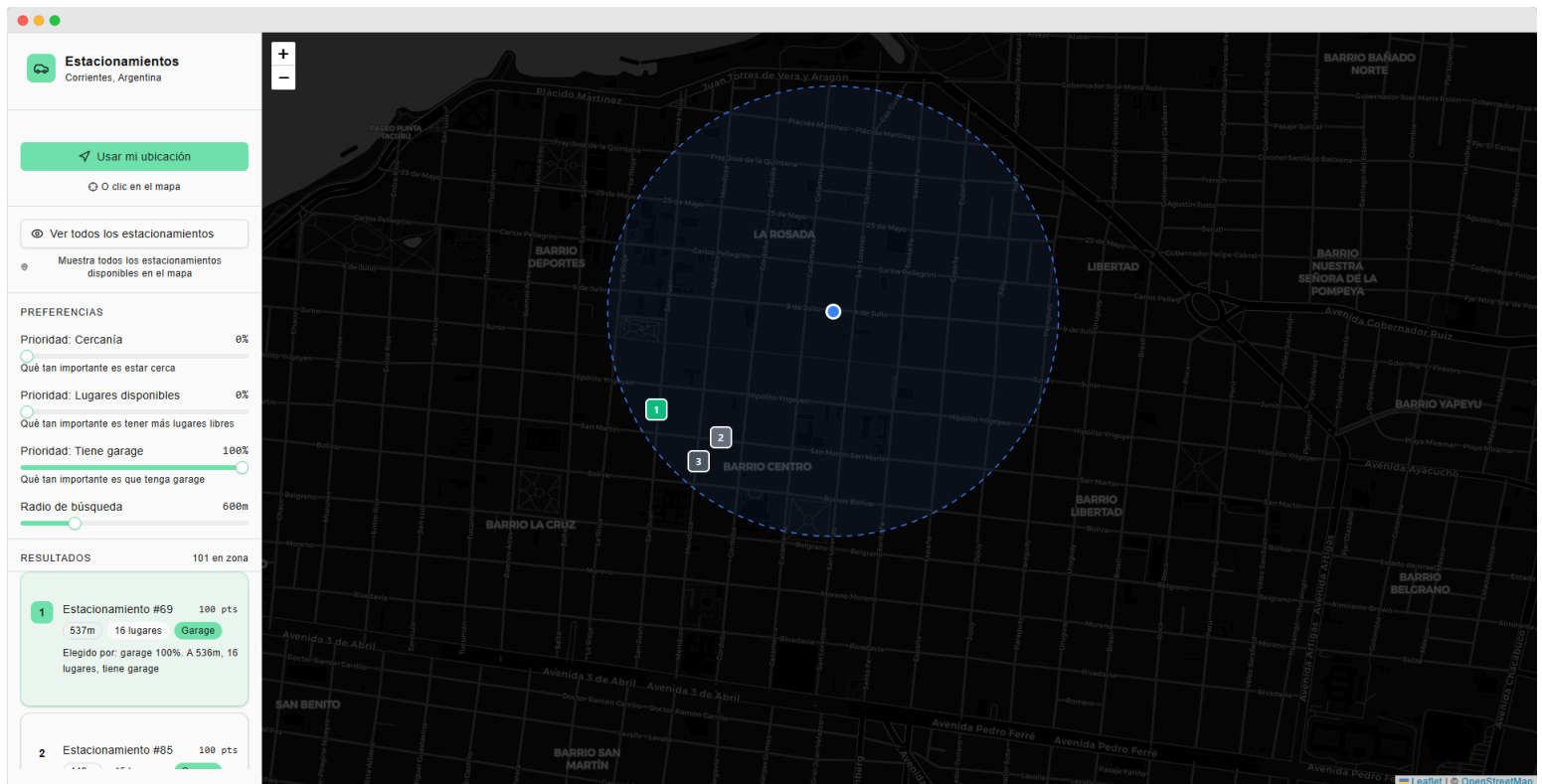
Prioridad del Usuario: Maximizar Disponibilidad

Comportamiento observado: El agente prioriza encontrar mucho lugar sin importar la distancia.



Prioridad del Usuario: Prioridad exclusiva garage

Comportamiento observado: El agente ignora la distancia y la capacidad, enfocándose únicamente en si el estacionamiento está techado (garage=1). El puntaje dependerá casi exclusivamente de este factor binario.



Los experimentos demuestran que el agente posee flexibilidad en su toma de decisiones. No ofrece una respuesta estática, sino que adapta su comportamiento racionalmente.

Conclusiones

El desarrollo de este Trabajo Práctico permitió integrar los fundamentos teóricos de la Inteligencia Artificial con una implementación práctica y funcional. Se implementó un Agente Racional basado en Utilidad, capaz de percibir preferencias de usuario y actuar en consecuencia para maximizar su rendimiento. Como limitación, se identifica que el agente opera en un entorno parcialmente observable (depende de la actualización del CSV). Como trabajo futuro, se propone buscar una solución que actualice la disponibilidad de lugares en tiempo real.

Referencias

[1] Municipalidad de la Ciudad de Corrientes, "Datasets - Portal de Datos Abiertos," *datos.ciudaddecorrientes.gov.ar*. [En línea]. Disponible en: <https://datos.ciudaddecorrientes.gov.ar/dataset>. [Último acceso: 11/11/2025].

[2]"FastAPI," *fastapi.tiangolo.com*. [En línea]. Disponible en: <https://fastapi.tiangolo.com/>. [Último acceso: 11/11/2025].

[3] The SciPy Community, "SciPy," *scipy.org*. [En línea]. Disponible en: <https://scipy.org/es/>. [Último acceso: 11/11/2025].

[4] The pandas development team, "pandas - Python Data Analysis Library," *pandas.pydata.org*. [En línea]. Disponible en: <https://pandas.pydata.org/>. [Último acceso: 11/11/2025].

[5] NumPy Developers, "NumPy," *numpy.org*. [En línea]. Disponible en: <https://numpy.org/>. [Último acceso: 11/11/2025].