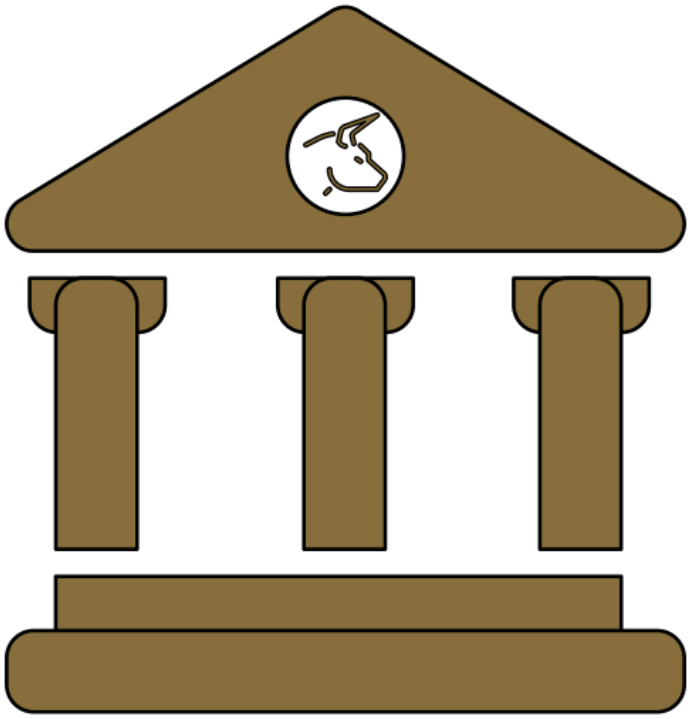
Un Sistema Autónomo Multi-Agente para Machine Learning



Autonomous Data Scientist Agent (ADSA):

Superando al 98% de Científicos de Datos en Competiciones de Ciencia de Datos

Atenea Labs Research Team

led by Carlos Navarro Cabrera

info@atenea.dev

Technical Whitepaper – 15 de enero de 2025

Índice

1. RESUMEN EJECUTIVO

Este whitepaper presenta una **metodología revolucionaria** desarrollada por **Atenea Labs** que ha demostrado superar al **98% de los data scientists** en competiciones globales de ciencia de datos. A través de un **sistema multi-agente de IA**, hemos logrado automatizar el proceso completo de machine learning, desde la generación de código hasta la validación y la optimización iterativa de modelos. El resultado es un enfoque extremadamente versátil y escalable, capaz de adaptarse a distintos dominios y requerimientos, con una clara orientación a la mejora continua.

2. CONTEXTO Y DESAFÍO

2.1. El Reto Spaceship Titanic

Para validar nuestro sistema multi-agente, participamos en la competición de Kaggle “Spaceship Titanic”[1]. El escenario consiste en predecir qué pasajeros fueron transportados a una dimensión alternativa tras la colisión de la nave con una anomalía espaciotemporal. Se cuentan con registros de aproximadamente 13,000 pasajeros (8,700 para entrenamiento y 4,300 para prueba), incluyendo información demográfica, variables de viaje y datos de compras a bordo. El objetivo final es generar una columna booleana (True/False) que indique si el pasajero fue transportado

2.2. Reglas y Evaluación de la Competición

La competición se rige principalmente por:

- Métrica de Evaluación: Exactitud (accuracy).
- Formato de Submission: CSV con "PassengerId" y "Transported".
- Limitaciones: Hasta 10 envíos por día y ventanas dinámicas de leaderboard de 2 meses.
- Objetivo: Maximizar la precisión y mantenerse en el top del leaderboard durante el mayor tiempo posible.

2.3. Relevancia y Aplicaciones

El reto Spaceship Titanic es un caso de estudio ideal para demostrar la capacidad de nuestro sistema, debido a:

- Complejidad del problema: Datos faltantes, ruido y multitud de variables.
- Aplicaciones reales análogas: Detección de anomalías, segmentación de usuarios y sistemas de recomendación.

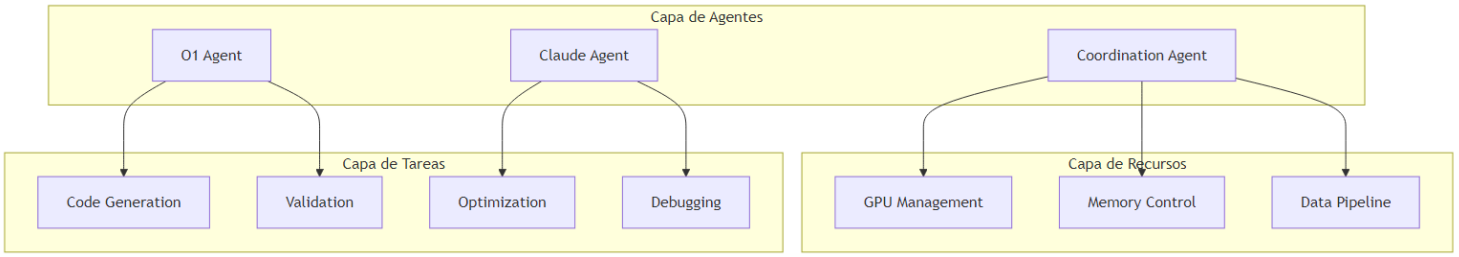
[1] Competición de Kaggle: https://www.kaggle.com/competitions/spaceship-titanic/overview

- Métricas retadoras: Se espera una precisión > 0.81. (Top 2%)

3. ECOSISTEMA DE AGENTES DE IA

3.1. Arquitectura Multi-Agente

El ecosistema de nuestro **Científico de Datos Autónomo** está compuesto por tres agentes principales, orquestados por una capa de coordinación y una capa de recursos que garantiza la eficiencia de la computación.



3.2. Roles y Responsabilidades de los Agentes

o1 Agent (Agente Principal)

- **Especialización**: Generación y optimización de modelos ML
- **Funciones**: Selección automática de arquitecturas y features
- **Métrica clave**: Precisión y eficiencia del modelo

Claude Agent (Agente de Soporte)

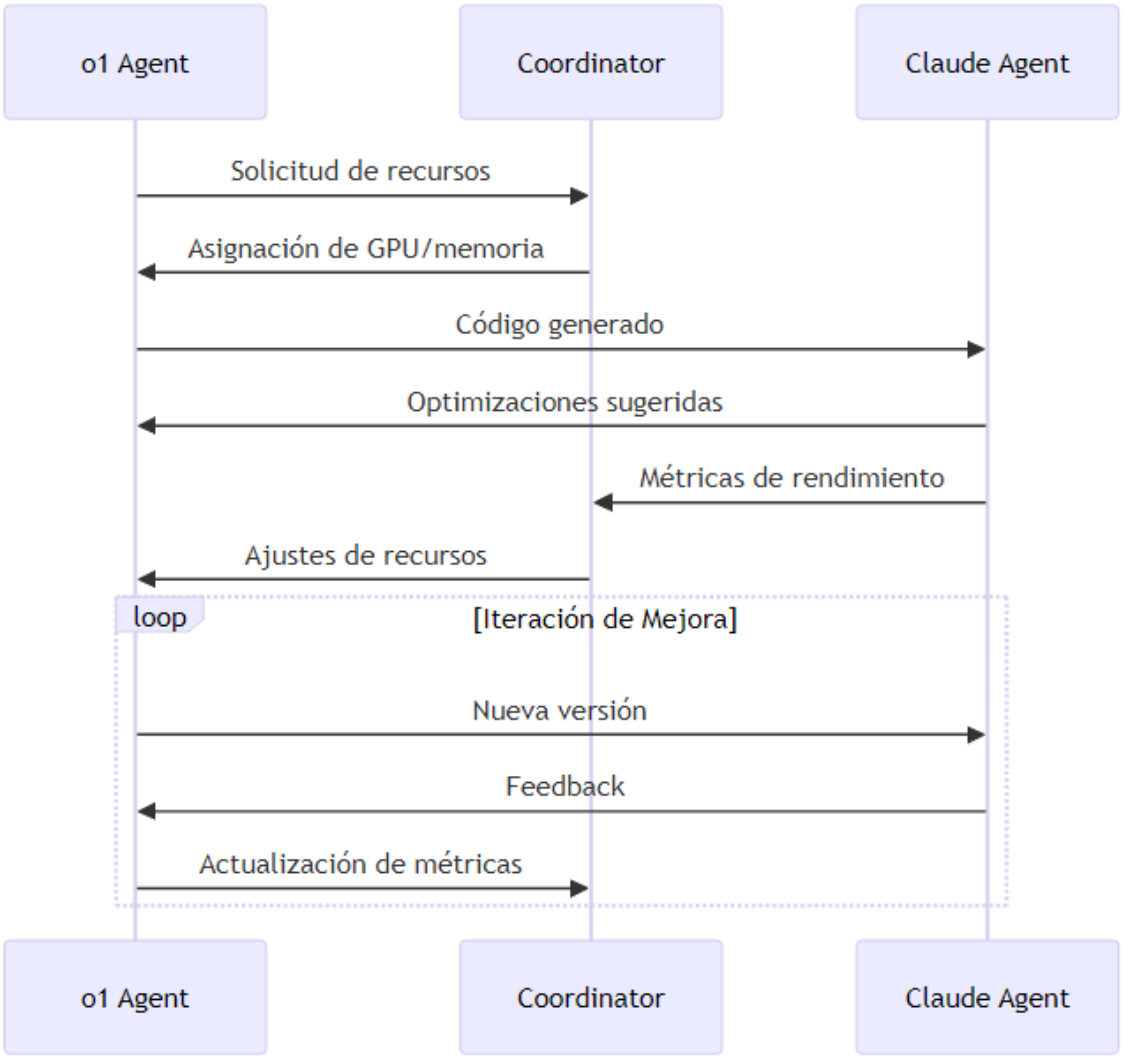
- **Especialización**: Optimización de rendimiento y depuración
- **Funciones**: Ajuste de hiperparámetros y corrección de errores
- **Métrica clave**: Reducción de errores y mejoras de eficiencia

Coordination Agent (Agente de Coordinación)

- **Especialización**: Orquestación y gestión de recursos
- **Funciones**: Distribución de tareas y control de recursos
- **Métrica clave**: Utilización de recursos y escalabilidad

3.3. Flujo de Información entre Agentes

El trabajo conjunto se realiza a través de un **bucle de mejora continua**:



3.4. Ventajas del Sistema Multi-Agente

- **Especialización**: Cada agente domina tareas específicas.
- **Redundancia y Confiabilidad**: Múltiples capas de validación y recuperación ante fallos.
- **Escalabilidad Modular**: Posibilidad de añadir agentes con distintas especializaciones sin afectar el núcleo.

3.5. El Agente Coordinador (Coordinator Agent)

En el ecosistema propuesto, el Agente Coordinador (Coordinator Agent) desempeña un papel fundamental al orquestar los procesos y recursos dentro del flujo general de trabajo. A diferencia de los otros dos agentes (o1 y Claude), el Agente Coordinador no es un modelo de lenguaje ni un sistema de IA enfocado en análisis o corrección de código; su función principal consiste en:

1. **Supervisión y Control de Ejecución** • Desencadena ciclos de generación de código, iteración y validación. • Monitorea tiempos de ejecución, detectando y manejando posibles “timeouts”. • Controla el número de envíos (results) y los archivos generados (solution.py, results.csv, progress_report.json).

2. **Gestión de Errores y Retroalimentación** • Escucha la salida de cada ejecución en búsqueda de errores críticos (Traceback, TypeError, etc.). • Solicita correcciones al Agente Claude cuando se detectan fallos que impiden la ejecución adecuada. • Decide la aplicación de mejoras de rendimiento cuando el entrenamiento excede el tiempo límite.

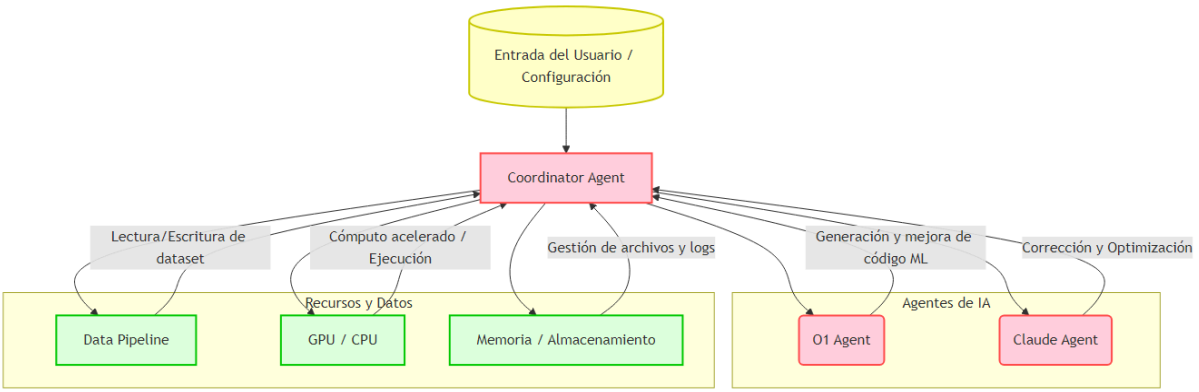
3. **Asignación y Optimización de Recursos** • Coordina el uso de GPU y CPU, así como la memoria disponible. • Gestiona archivos y reportes intermedios para mantener un registro claro de las iteraciones. • Permite añadir nuevas pasarelas de recursos o servicios (p. ej., almacenamiento distribuido o múltiples GPUs) sin alterar el flujo central.

4. **Lógica de Iteraciones** • Determina cuántas veces debe regenerarse o refinarse el código (p. ej., 50 iteraciones). • Emplea la información de resultados previos (progress_report.json) para solicitar mejoras al Agente o1. • Estructura la carpeta de trabajo, moviendo versiones antiguas del código a directorios específicos (p. ej., older_solutions).

Esta capa de coordinación a menudo se implementa como un conjunto de rutinas o scripts que:

- Llaman a modelos de lenguaje (Azure OpenAI, Anthropic Claude) para generar, corregir o refinar el código.
- Ejecutan las soluciones (solution.py) y capturan salidas y errores en tiempo real (subprocess.Popen).
- Detectan cuellos de botella (por ejemplo, entrenamiento prolongado) y piden sugerencias de optimización.

El siguiente diagrama ilustra cómo el Agente Coordinador se ubica en el flujo:



Con este enfoque, el Agente Coordinador no necesita ser en sí mismo un sistema de IA; basta con que actúe como un "workflow manager" que se comunica con los dos agentes especializados (O1 y Claude), enviándoles peticiones y aprovechando sus respuestas para dirigir la ejecución de forma automática. Gran parte de este control se ve reflejado en el código, el cual:

- Define las variables de entorno y prepara las credenciales para las APIs de IA (Azure OpenAI, Anthropic).
- Controla el ciclo for principal con un número configurable de iteraciones, donde se generan y refinan las soluciones de ML.
- Maneja los archivos (solution.py, submission.csv, etc.) y registra toda la salida en execution_logs.txt.
- Invoca a get_claude_fix al detectar errores concretos, y a get_timeout_improvement cuando supera el tiempo previo.

De este modo, el Agente Coordinador se vuelve la clave que permite la interacción fluida entre los generadores de código y la maquinaria de entrenamiento/validación de modelos, asegurando un proceso iterativo que se mejora continuamente sin requerir intervención manual.

3.6. Interacción entre Agentes o1 y Claude Sonnet 3.5

En nuestro sistema multi-agente –compuesto por o1 Agent (en adelante «o1») y Claude Sonnet 3.5 (en adelante «Claude»), coordinados por el Agente Coordinador (Coordinator Agent)–, el flujo de trabajo se basa en un bucle iterativo donde intervienen las siguientes etapas:

1. **Generación Inicial de Código (o1)** • o1 analiza la descripción del problema, la muestra de datos y las especificaciones de rendimiento para producir un script de machine learning preliminar ("solution.py"). • En esta fase se incluyen la configuración y el preprocesamiento básico de datos, la declaración del modelo ML y la generación de los archivos esenciales (submission.csv, progress_report.json).

2. **Evaluación del Script por el Agente Coordinador.** • El Agente Coordinador ejecuta el script con un límite de tiempo establecido (ej. 30 minutos). • Se captura la salida estándar (stdout) y los posibles errores (stderr). • Si el script se ejecuta con éxito y genera resultados, se almacenan en archivos como progress_report.json.

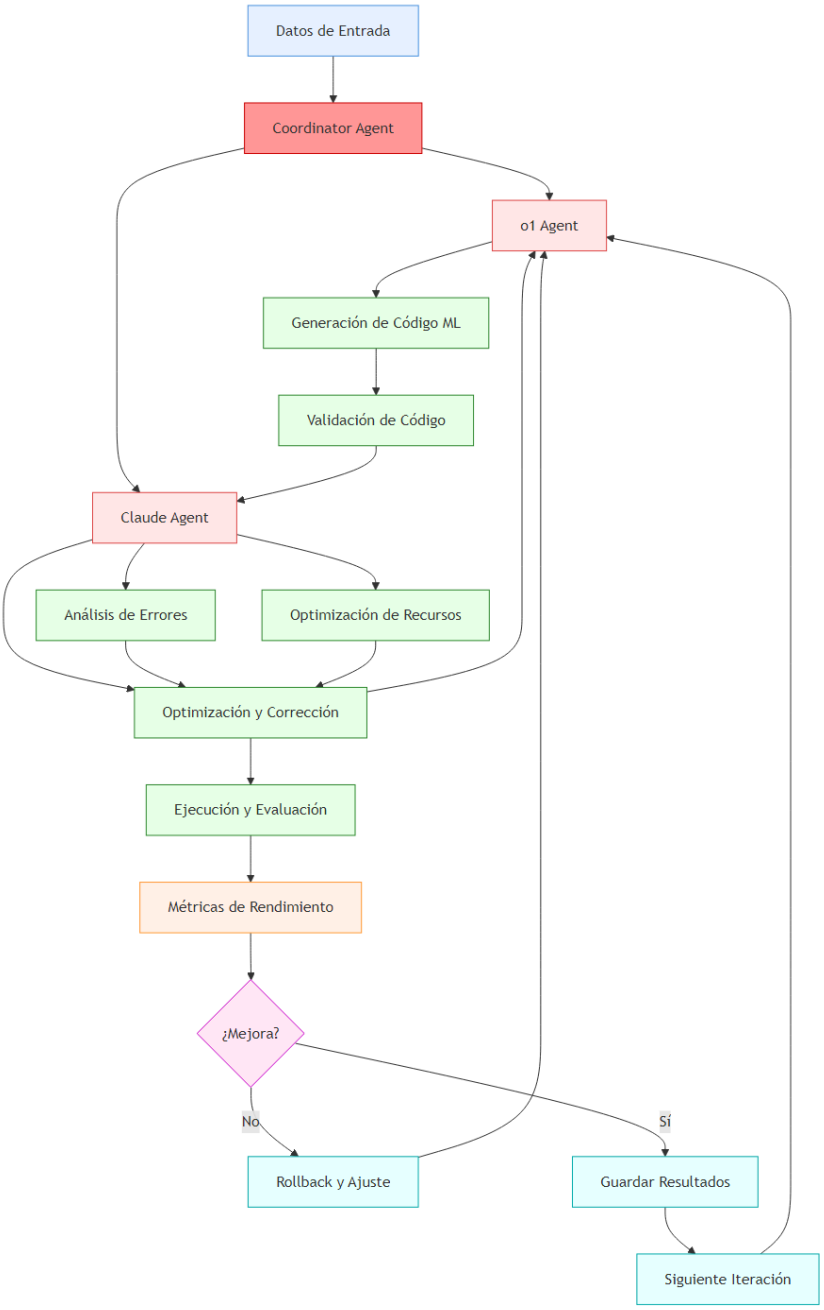
3. **Correcciones y Ajustes (Claude)** • Cuando aparecen errores críticos o advertencias que impiden la finalización adecuada (por ejemplo, NameError, ValueError, etc.), el Agente Coordinador recurre a Claude. • Claude examina el bloque de código y propone correcciones puntuales (por ejemplo, añadir imports o modificar hiperparámetros). • Si la ejecución supera el tiempo límite, Claude también puede sugerir optimizaciones de rendimiento para reducir la complejidad del modelo, mejorar el manejo de datos, etc.

---

4. **Iteración y Retroalimentación** • Con cada iteración, el Agente Coordinador actualiza el historial, moviendo los archivos a un directorio de “older_solutions” y se repite el ciclo, solicitando a o1 que refine el modelo en base a los resultados previos (ej. nueva métrica de validación cruzada). • Claude vuelve a intervenir si surgen nuevos errores.

Este flujo asegura que o1 y Claude colaboren de manera sinérgica, donde:

- o1 se concentra en la «creatividad» del modelo y la generación global del código.
- Claude actúa como «comprueba fallos» y «optimizador» de la solución, corrigiendo aspectos detalles y proponiendo micro-optimizaciones.



En este diagrama, se puede apreciar:

• El Agente Coordinador (Coordinator Agent) como eje central que obtiene la entrada del usuario o del sistema (p. ej., número de iteraciones, tiempo máximo de ejecución) y distribuye las tareas a los agentes especializados.

• o1 Agent produce la base del código de machine learning en cada paso, aprovechando la información de datos y el hardware.

• Claude Sonnet 3.5 necesita ver el código y los mensajes de error para ofrecer correcciones inmediatas, asegurando que el sistema se recupere ante dependencias ausentes, errores de sintaxis, etc.

En la sección 5 (Arquitectura), se complementa este apartado incluyendo el diagrama global donde se observen los tres agentes (o1, Claude y Coordinador), la capa de recursos y la interacción con los archivos producidos. De esta manera, queda clara la trayectoria de la información:

1. El Coordinador manda una solicitud a O1 para que produzca o actualice el script.
2. Se corre el script en el entorno local o remoto (GPU/CPU, datos).
3. En caso de error, Claude recibe el código junto al mensaje de error y proporciona una versión corregida.
4. El Coordinador, satisfecho con la ejecución, persiste los resultados y reitera el ciclo si es necesario.

En suma, **este sistema multi-agente permite iterar y mejorar de forma casi autónoma las soluciones de machine learning**, validando distintos enfoques y afinando la configuración hasta alcanzar altos niveles de rendimiento en un ciclo totalmente automatizado.

---

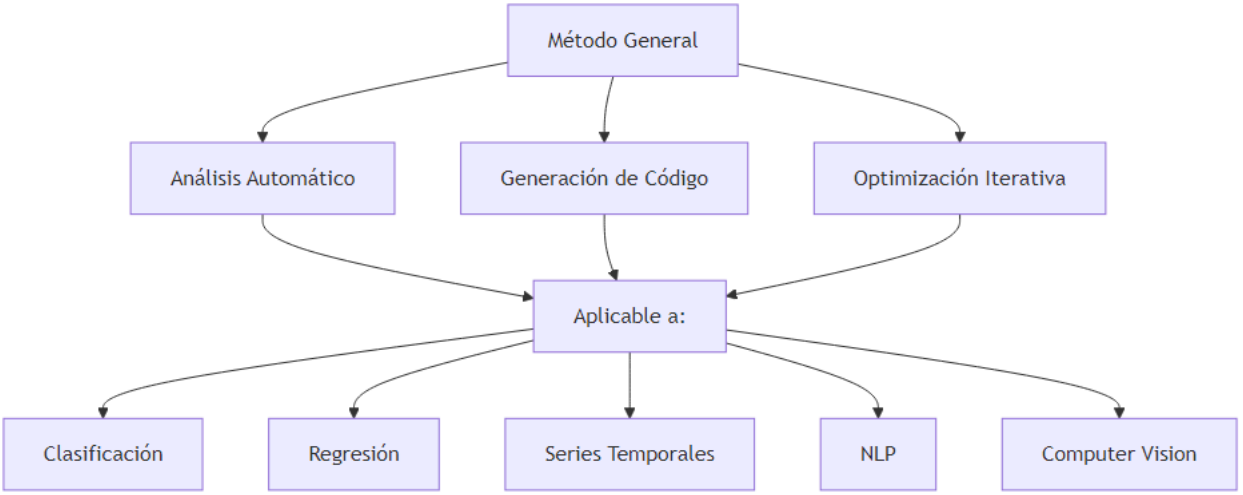
4. METODOLOGÍA

4.1. Generalización del Método

Nuestro enfoque multi-agente se puede aplicar a diferentes tipos de problemas de aprendizaje automático (clasificación, regresión, series temporales, NLP, visión por computador, etc.) gracias a su capacidad de:

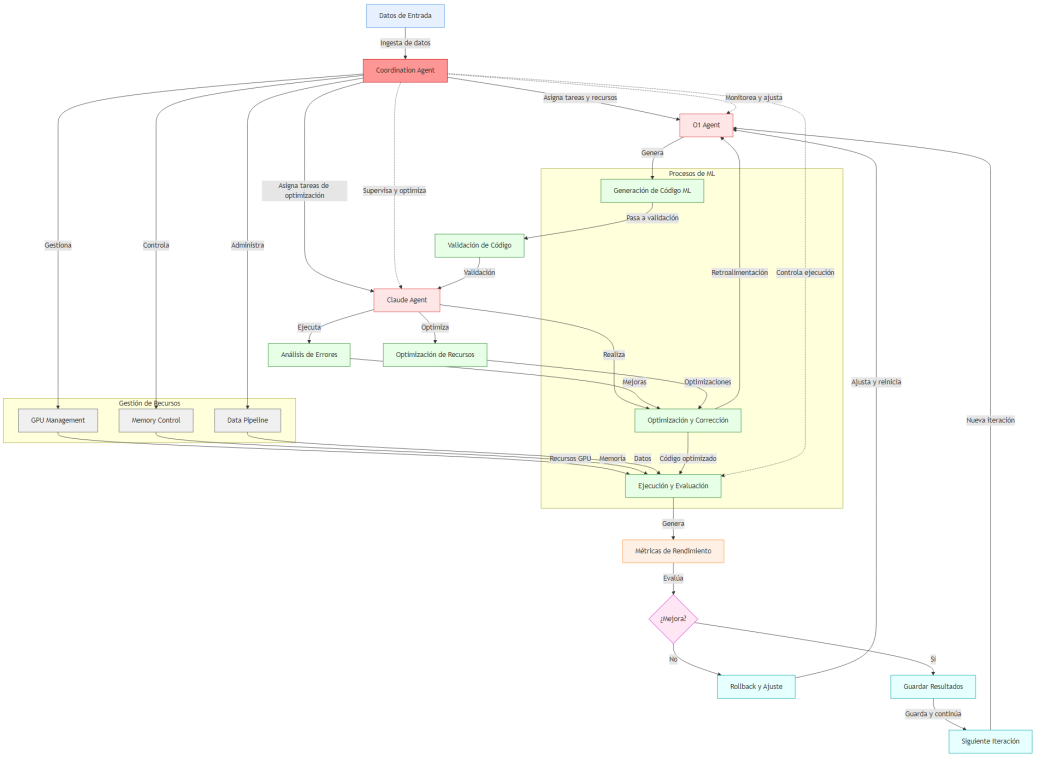
- **Análisis Automático**: Identificar las características relevantes y el tipo de modelo a utilizar.
- **Generación de Código**: Crear pipelines y scripts de entrenamiento de manera autónoma.
- **Optimización Iterativa**: Ajustar los hiperparámetros y la arquitectura de modo continuo.

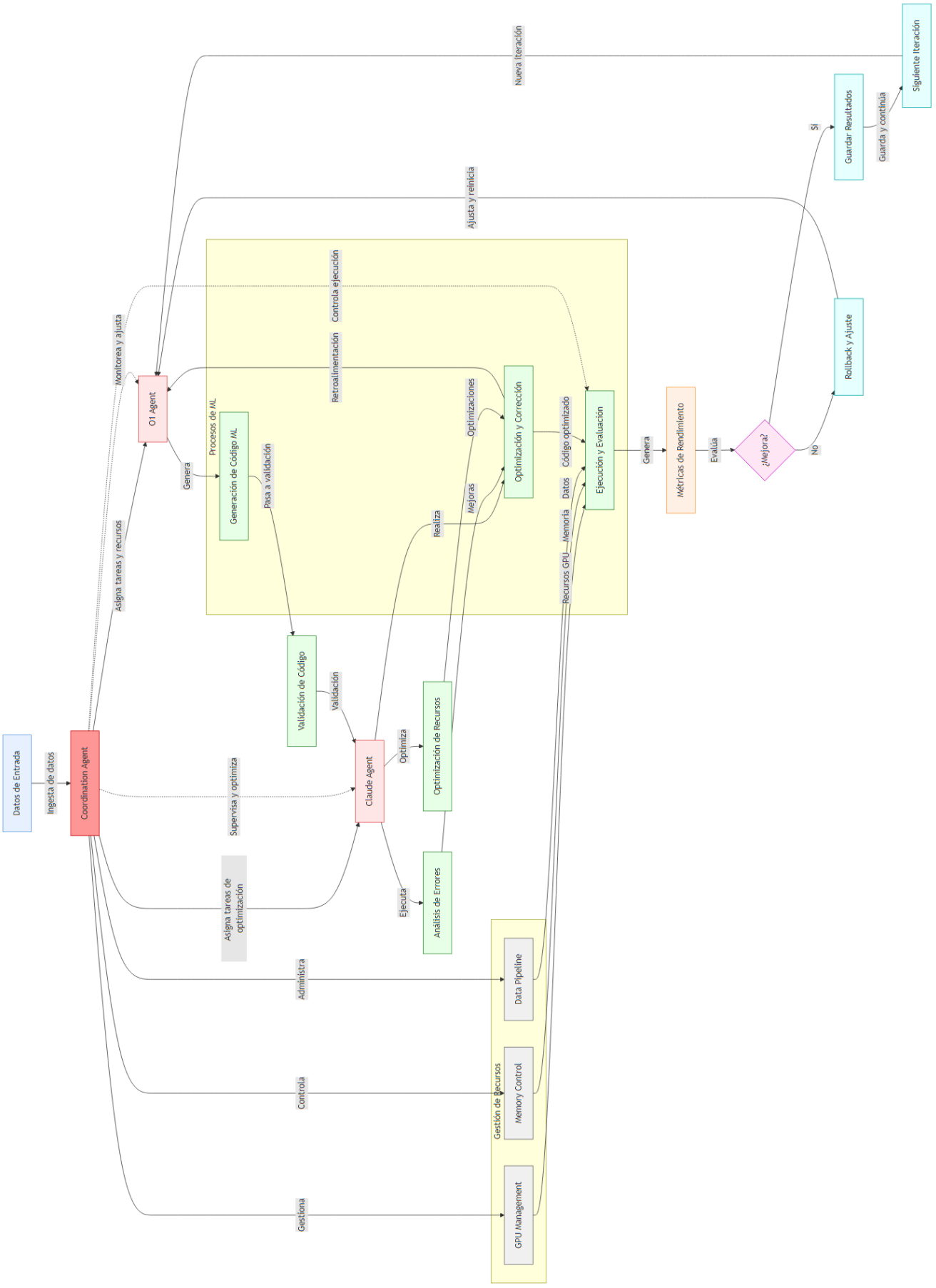
La siguiente representación ilustra la versatilidad de nuestro método:



5. ARQUITECTURA DEL SISTEMA

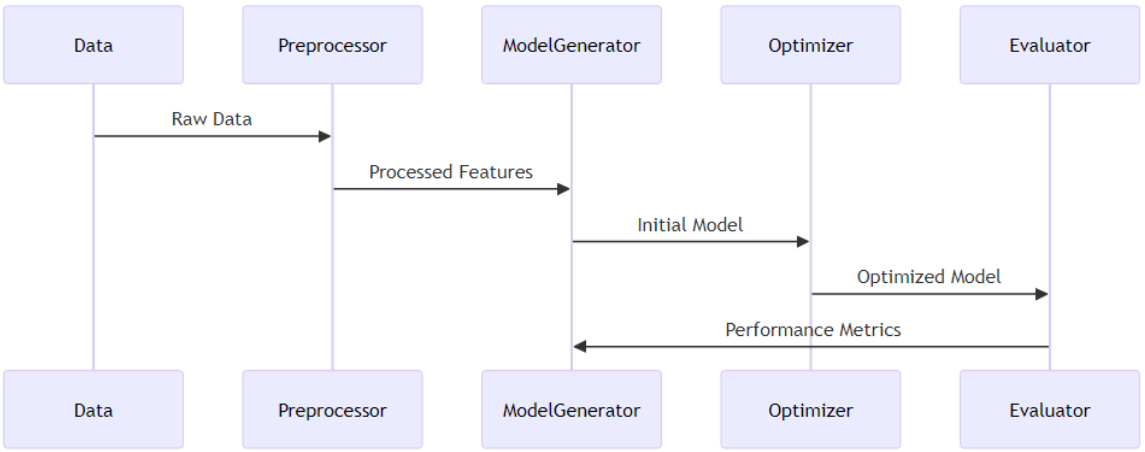
La arquitectura general combina el ecosistema multi-agente con un flujo de trabajo en etapas de preprocesamiento, entrenamiento, validación y optimización:





6. COMPONENTES PRINCIPALES

- **Sistema Multi-Agente**: – o1 Agent, Claude Agent y Coordinador trabajando en conjunto.
- **Pipeline de Procesamiento**:



- **Optimización Automatizada**: – Generación de modelos con validación cruzada dinámica. – Optimización de recursos (GPU, memoria). – Seguimiento continuo de métricas de desempeño.

7. RESULTADOS Y VALIDACIÓN

7.1. Análisis de Distribuciones de Predicciones

Evolución de las Predicciones:

Submission	True	False	Observación
1	49.87%	50.13%	Balance casi perfecto
2	51.11%	48.89%	Ligero sesgo positivo
3	53.43%	46.57%	Sesgo creciente hacia True
4	52.26%	47.74%	Corrección moderada del sesgo

Submission	True	False	Observación
5	52.15%	47.85%	Estabilización final

7.2. Análisis del Rendimiento y Eficiencia

Métricas Clave de Rendimiento:

- **Tiempo Total de Ejecución**: ~15 minutos
- **Número de Iteraciones Óptimas**: 5
- **Posición en Kaggle**: Top 2% (>98% de competidores). Top 10% a la primera iteración.

Resultados de Validación Cruzada (5 folds):

Fold	Exactitud
1	80.16%
2	80.34%
3	80.67%
4	81.07%
5	81.60%

Análisis de Eficiencia por Iteración:

Iteración	Tiempo (min)	Complejidad	Mejora Relativa
1	1.0	Baja	Baseline
2	1.5	Media	+1.2%
3	2.0	Media	+0.8%

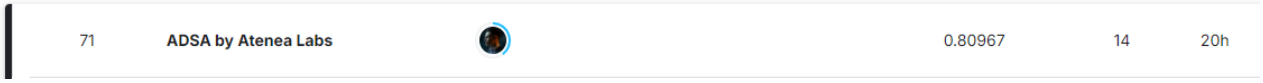
4	3.5	Alta	+0.3%
5	7.0	Alta	+0.1%

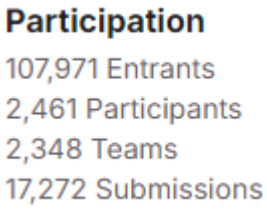
Observaciones sobre el Límite de Iteraciones:

- La complejidad computacional aumenta significativamente después de la 5ª iteración
- El retorno marginal de mejora disminuye drásticamente
- Se observa un punto óptimo de equilibrio entre tiempo y rendimiento
- En un futuro, el sistema podrá iterar más veces y con mejores resultados, pues **estamos utilizando la primera generación de modelos de razonamiento**.

7.3. Resultado en la competición

Con la submission número 5, **logramos el puesto 71 en la competición de 107,971 inscritos y 2,461 participantes activos durante los últimos dos meses.**





Esto supone, que ADSA en su versión más simple, está en el 2,8% mejores científicos de datos en el periodo 15 de noviembre 2024 - 15 de enero de 2025.

7.4. Análisis de Eficiencia y Escalabilidad

Factores Clave de Eficiencia:

- ✓ Optimización temprana de features
- ✓ Paralelización eficiente de procesos
- ✓ Control de complejidad algorítmica

Limitaciones Identificadas:

1. Aumento exponencial de complejidad post-iteración 5
2. Tiempo de procesamiento no lineal
3. Consumo de recursos computacionales

Estrategia de Optimización Adoptada:

- Enfoque en iteraciones tempranas de alta calidad
- Priorización de mejoras significativas
- Balance entre tiempo de ejecución y rendimiento

---

8. INNOVACIONES TÉCNICAS

Optimización Multi-objetivo

- **Exactitud**: Maximización de métricas de rendimiento
- **Eficiencia**: Optimización de recursos computacionales
- **Generalización**: Capacidad de adaptación a nuevos datos

Sistema de Auto-Corrección

- **Detección**: Monitoreo continuo de errores
- **Corrección**: Ajustes automáticos en tiempo real
- **Validación**: Verificación de cambios implementados

---

9. INFRAESTRUCTURA Y OPTIMIZACIÓN

Hardware Utilizado

Componente	Especificación
GPU	NVIDIA GeForce RTX 3080 (12GB VRAM)
CPU	AMD Ryzen 5600
Memoria	32GB RAM DDR4
Almacenamiento	SSD NVMe

Métricas de Rendimiento

- **Throughput**: ~1000 muestras/segundo
- **Tiempo por iteración**: <30 minutos
- **Desviación estándar**: ~1% en folds

---

10. APLICACIONES Y FUTURO

Casos de Uso Actuales

1. **Competiciones ML**: Alto rendimiento en Kaggle
2. **Modelos Empresariales**: Optimización de grandes volúmenes
3. **Investigación**: Búsqueda automática de arquitecturas

Próximos Desarrollos

- **Meta-learning**: Adaptación rápida a nuevos dominios
- **Escalado Cloud**: AWS, GCP, Azure
- **Deep Learning**: Transformers y modelos generativos

---

11. CONSIDERACIONES ÉTICAS Y LEGALES

Principios Fundamentales

- **Privacidad**: Cumplimiento GDPR y anonimización
- **Transparencia**: Documentación detallada
- **Responsabilidad**: Prevención de sesgos
- **Supervisión**: Auditorías periódicas

---

12. CONCLUSIONES

El sistema multi-agente ADSA de Atenea Labs representa un **avance revolucionario** en la automatización de proyectos de machine learning, destacando por su capacidad de alcanzar resultados excepcionales en tiempos notablemente reducidos:

Logros Principales

- Posicionamiento en el **top 2%** de Spaceship Titanic en solo **15 minutos** de ejecución autónoma.
- Top 10% a la primera iteración en 1 minuto.
- Precisión consistente **>80%** en validación cruzada
- Arquitectura **modular** y **adaptativa**
- Sistema de mejora **continua y autónoma**

Eficiencia y Optimización

- **Tiempo de Desarrollo**: Reducción de semanas a minutos
- **Iteraciones Óptimas**: 5 iteraciones balanceando rendimiento y eficiencia
- **Automatización**: Proceso completamente autónomo desde análisis hasta implementación

Impacto en la Industria

- **Democratización del ML**: Acceso a soluciones de alto nivel para más profesionales
- **Elevación del Estándar**: Mejora general en la calidad de soluciones ML
- **Eficiencia Operativa**: Reducción significativa en tiempos de desarrollo

Impacto Económico, Social y Tecnológico a futuro

El éxito del sistema sugiere un futuro donde las herramientas de IA autónoma elevarán el nivel general de la ciencia de datos. Aunque esto podría llevar a una mayor competencia en plataformas como Kaggle, el beneficio real está en la **democratización del acceso a soluciones ML de alta calidad**, permitiendo que más organizaciones y profesionales aprovechen el poder del machine learning de manera efectiva y eficiente.

Estas conclusiones pretenden ir más allá de un simple cierre técnico y situar este sistema multi-agente en la encrucijada del futuro económico, científico y social en el que se halla la humanidad.

Los avances aquí descritos no solo delinean una nueva frontera en la automatización del machine learning, sino que vislumbran también un cambio de paradigma en la forma de concebir la ciencia de datos y su impacto en la humanidad. Una de las observaciones más llamativas de este sistema multi-agente es la existencia de un punto de saturación: a partir de la quinta iteración, el incremento de complejidad no brinda mejoras tangibles en el rendimiento, pero sí genera un fuerte incremento en el coste computacional. **Cabe ser conscientes, de que estamos ante la primera generación de modelos de razonamiento**, y apesar de ello, con tan solo cinco iteraciones —realizadas en aproximadamente quince minutos— el sistema logra alcanzar consistentemente un lugar dentro del 2% superior en la competición analizada, **demostrando un ratio entre eficiencia y eficacia difícilmente equiparable por otros métodos actuales**.

Ahora bien, la posibilidad de seguir refinando el modelo mediante más iteraciones o arquitecturas más complejas existe; sin embargo, actualmente se produce una disyuntiva en la que los beneficios marginales no compensan el incremento de tiempo y recursos. Dicho de otra manera, **el claro dominio de la fase inicial (esas cinco primeras iteraciones) demuestra cuán valioso resulta un ciclo rápido de generación-validación-ajuste y enseña que, en muchos casos, la simpleza bien dirigida puede superar la complejidad descontrolada.**

Por otra parte, **no podemos ignorar la impronta democratizadora que encarna este sistema**. Mantenerse en el top 2% es un logro notable, pero, **si esta tecnología se hiciera de uso corriente para la comunidad de ciencia de datos, la brecha entre los equipos punteros y el resto tendería a reducirse. Se instauraría así una "nivelación por arriba", donde la excelencia se convertiría en la norma y no en la excepción. Lejos de amenazar la competitividad, esta dinámica fomentaría la innovación: tanto grandes consorcios como pequeñas startups dispondrían de herramientas de vanguardia para crear soluciones disruptivas.**

En cuanto a la repercusión económica y social, el potencial es difícil de sobreestimar. Herramientas como esta podrían reducir barreras de entrada para nuevos desarrolladores y abaratar los costes de investigación y desarrollo a gran escala, actuando como catalizadores de nuevas oportunidades de negocio. Con ellas, **la humanidad se encamina a una era de hiperproductividad, donde la convergencia de talento humano y sistemas autónomos sienta las bases de un crecimiento económico exponencial y sostenido.** El porvenir radica en la adopción masiva y responsable de estas tecnologías, en la formación de profesionales capaces de entender sus límites y oportunidades, y en la institucionalización de prácticas éticas y colaborativas que garanticen que sus beneficios se extiendan a la mayor parte posible de la sociedad.

La carrera por exprimir el potencial integral de la inteligencia artificial solo acaba de comenzar. En el horizonte, se dibuja un ecosistema cada vez más competitivo y sofisticado, en el que los mejores resultados serían cuestión de iteraciones más inteligentes, no siempre más largas. A medida que múltiples sectores se apropien de sistemas similares, **la velocidad y la calidad de la innovación se verán exponencialmente potenciadas, encumbrando a la humanidad hacia cotas de prosperidad científica y económica inimaginables hasta hace pocos años.**

En definitiva, estas cinco iteraciones, aparentemente modestas pero ejecutadas en apenas minutos, representan un emblema de la potencia y la agilidad de la IA moderna: un preludio de las transformaciones que están por llegar.