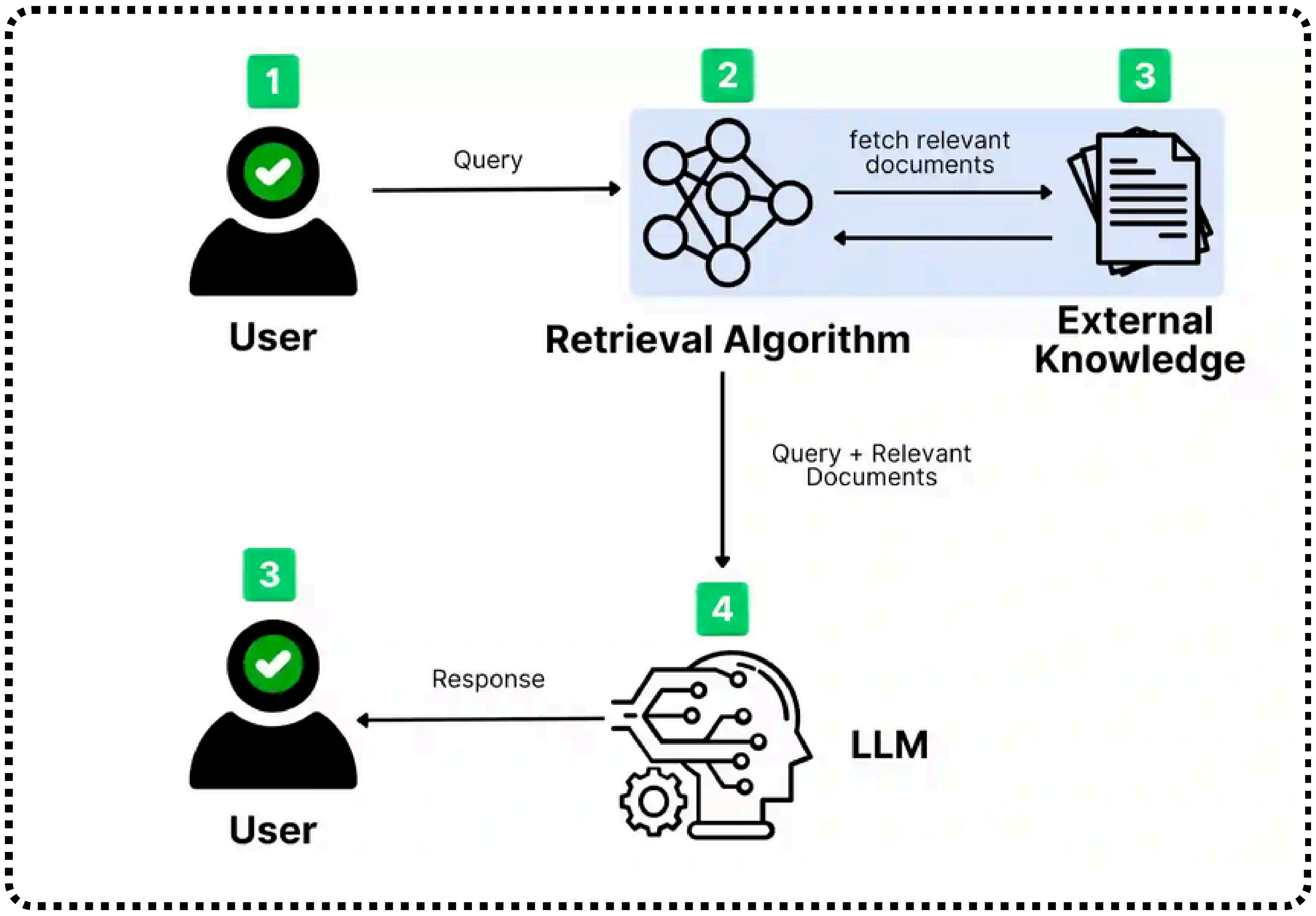
Analytics Vidhya

Mastering RAG

RAG vs Agentic RAG



1 User

Query

2 Retrieval Algorithm

fetch relevant documents

3 External Knowledge

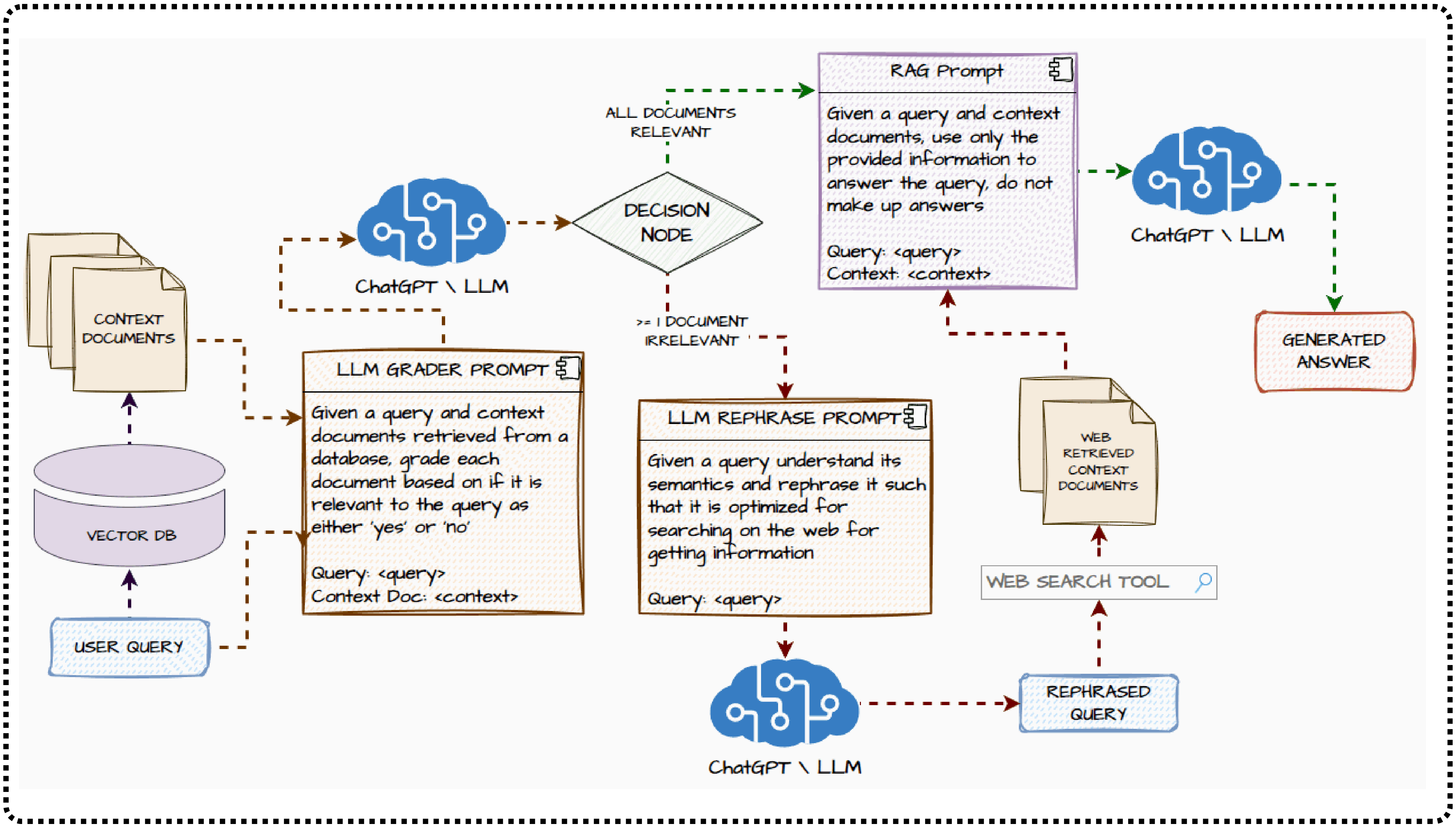
Query + Relevant Documents

4 LLM

Response

3 User

Agentic RAG introduces AI agents into the retrieval-augmented workflow, enabling iterative reasoning, planning, and self-improvement. Instead of just retrieving documents and generating a response in one step, the system evaluates, re-retrieves, and refines outputs dynamically.



CONTEXT DOCUMENTS

VECTOR DB

USER QUERY

LLM GRADER PROMPT

Given a query and context documents retrieved from a database, grade each document based on if it is relevant to the query as either 'yes' or 'no'

Query: <query>
Context Doc: <context>

ChatGPT \ LLM

DECISION NODE

ALL DOCUMENTS RELEVANT

>= 1 DOCUMENT IRRELEVANT

RAG Prompt

Given a query and context documents, use only the provided information to answer the query, do not make up answers

Query: <query>
Context: <context>

ChatGPT \ LLM

GENERATED ANSWER

LLM REPHRASE PROMPT

Given a query understand its semantics and rephrase it such that it is optimized for searching on the web for getting information

Query: <query>

ChatGPT \ LLM

REPHRASED QUERY

WEB SEARCH TOOL

WEB RETRIEVED CONTEXT DOCUMENTS

What is Agentic RAG?

Agentic RAG introduces AI agents into the retrieval-augmented workflow, enabling iterative reasoning, planning, and self-improvement. Instead of just retrieving documents and generating a response in one step, the system evaluates, re-retrieves, and refines outputs dynamically.

How Agentic RAG Works:

- User Query → LLM Agent
- LLM Agent determines what knowledge it needs
- Retrieval System performs multiple retrieval passes
- Agent reviews retrieved data, verifies relevance, and may refine queries
- LLM generates an initial response
- Agent critiques, refines, and iterates until high-quality output is achieved

Key Features of Agentic RAG:

Iterative Retrieval: Agent re-queries if the initial retrieval is weak

Self-Reflection & Reasoning: Adjusts based on response quality

Autonomous Decision-Making: Can decide when to retrieve, refine, or stop

Multi-Step Query Optimization: Adapts dynamically to complex tasks

Advantages of Agentic RAG:

Improved accuracy: Dynamically corrects errors & retrieves more precise data

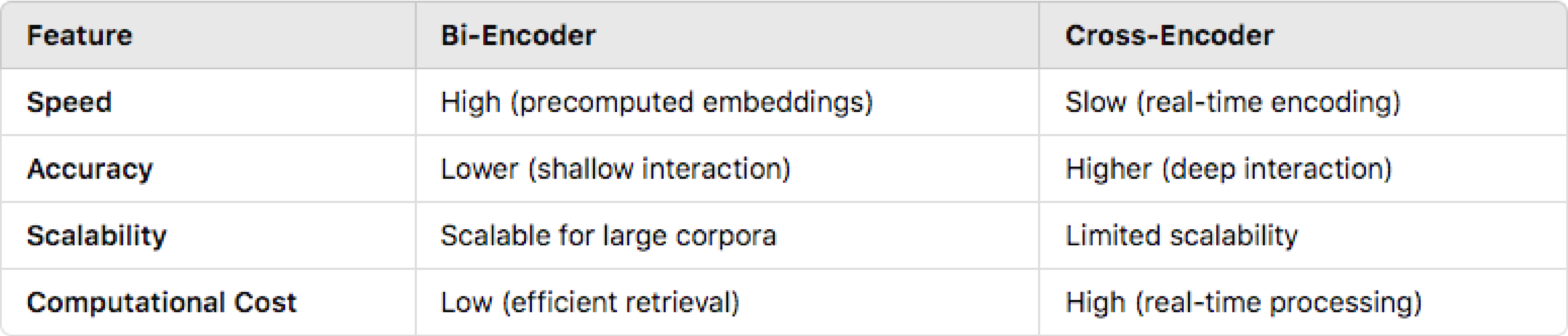
Better context understanding: Can break down complex questions

More reliable in real-world applications: Can self-correct hallucinations

Less reliance on prompt engineering: Learns and adapts autonomously

Agentic RAG vs. Standard

Documents & queries are converted into high-dimensional vectors before retrieval.



Feature	Bi-Encoder	Cross-Encoder
Speed	High (precomputed embeddings)	Slow (real-time encoding)
Accuracy	Lower (shallow interaction)	Higher (deep interaction)
Scalability	Scalable for large corpora	Limited scalability
Computational Cost	Low (efficient retrieval)	High (real-time processing)

Key Use Cases for Agentic RAG

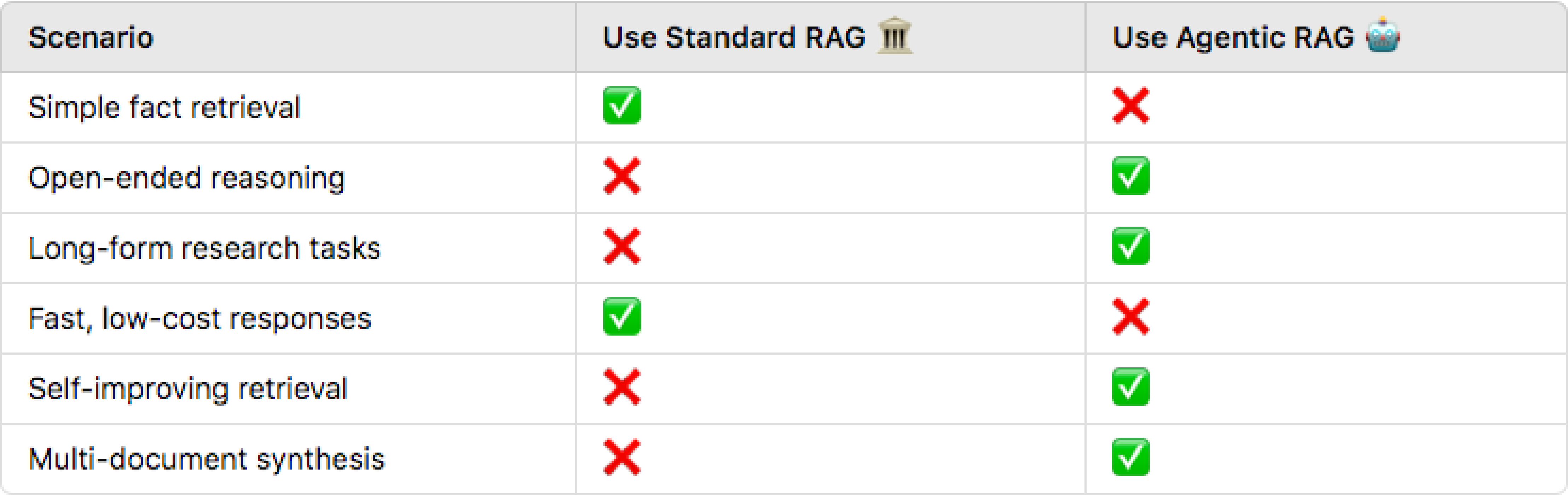
Agentic RAG is particularly beneficial for complex, multi-step reasoning tasks where a single retrieval cycle isn’t sufficient.

Best Use Cases for Agentic RAG

- **Scientific & Technical Q&A**(Iterative fact-checking)
- **Legal & Compliance Checks** (Analyzing multiple regulations)
- **Financial & Market Analysis**(Synthesizing real-time data)
- **Code Generation & Debugging** (Stepwise improvement)
- **Customer Support AI** (Adaptive problem-solving)
- Standard RAG retrieves one batch of legal clauses and generates a response.
- Agentic RAG analyzes gaps, re-retrieves related case laws, and refines its legal opinion.

When to use Agentic RAG vs RAG

Re-ranking **improves retrieval results** by scoring and ordering retrieved documents before feeding them to the LLM.



Scenario	Use Standard RAG 🏛️	Use Agentic RAG 🤖
Simple fact retrieval	✅	❌
Open-ended reasoning	❌	✅
Long-form research tasks	❌	✅
Fast, low-cost responses	✅	❌
Self-improving retrieval	❌	✅
Multi-document synthesis	❌	✅