RAG + LLM Evaluation

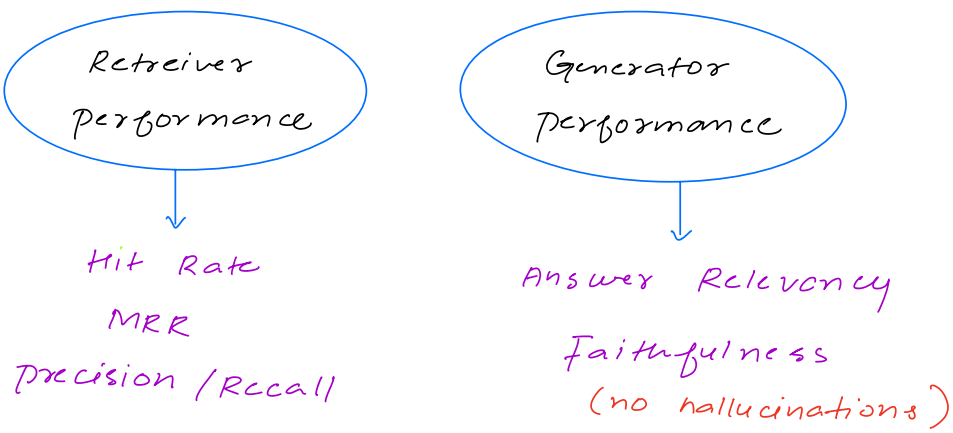
RAGAS = *Retrieval-Augmented Generation Assessment Suite* — a framework to **evaluate retrieval-augmented generation (RAG) systems**.

In a typical RAG pipeline:

- **Retriever** → fetches potentially relevant documents for a query.
- **Generator** (usually an LLM) → uses those documents to produce the final answer.

RAGAS helps track how well each component performs by providing **automatic evaluation metrics**. Many of these rely on LLMs as judges, so you don't need to label everything manually.

Q. what does RAGAS help us Track ?



1. Retriever performance

- **Hit Rate** – Did the retriever return at least one document containing the ground-truth (gold answer)?
- **MRR (Mean Reciprocal Rank)** – How high up was the *first relevant document* ranked?
  - If the first relevant doc is usually at the top, MRR is high (close to 1).
  - If relevant docs appear later, MRR drops.
- **Context Precision/Recall** – Measures how many retrieved docs are truly relevant (precision) and how many relevant docs were missed (recall).

👉 These metrics tell you if the **retriever is surfacing the right context** and putting it near the top.

Actual Answer

Rank of Actual Answer

2. Generator performance

- **Answer Relevancy** – Does the generated answer actually address the question?
  - Checked by comparing the answer with the query (using LLM-as-judge or semantic similarity).
  - High relevancy = the model stays on-topic.
- **Faithfulness** – Is the answer *grounded in retrieved documents* (i.e., no hallucinations)?
  - Checked by verifying whether the answer's statements are supported by the retrieved context.
  - Low faithfulness means the LLM is inventing facts not present in the context.

👉 These metrics tell you if the **generator uses the retrieved context properly** and avoids hallucinations.

LLM judges whether ans. is Relevant

whether content RAG is used.

🔷 Why this matters

- If **MRR is low** → retriever finds relevant docs but ranks them poorly → need better ranking.
- If **Answer Relevancy is low** → generator isn't focusing on the query → prompt/model issue.
- If **Faithfulness is low** → generator hallucinates even when retriever worked → grounding issue.

By combining retriever + generator metrics, RAGAS tells you **where your pipeline is failing**:

- Bad retriever?
- Good retriever but bad generator?
- Or both?

Evaluation Process

🔷 Step 1: You need a benchmark dataset

For evaluation, you can't just run on random queries. You need a **test set** that has:

1. **Question (query)** – what the user would ask.
2. **Gold answer (ground truth)** – the correct, authoritative answer.
   - This is the reference against which you'll compare your model's output.
   - It may come from a curated dataset or be annotated manually.
3. (Optionally) **Gold documents** – the reference passages that actually contain the answer.

Example dataset entry:

json Copy code

```
{
  "question": "Who is the CEO of Tesla?",
  "ground_truths": ["Elon Musk"],
  "contexts": ["Tesla, Inc. is led by CEO Elon Musk..."],   # retrieved docs
  "answer": "Elon Musk is the CEO of Tesla."                # your model's answer
}
```

Gold Answer

So the **"benchmark"** you compare against is the **gold answers** (and sometimes gold docs) in this dataset.

Imp

🔷 Step 2: How Each Metric Is Checked

Let's use this same example — *"Who is the CEO of Tesla?"* — to understand each metric.

1. Hit Rate (Retriever Metric) → Hit rate is binary

Goal:
Checks whether *any* of the retrieved documents contain the **gold answer**.

True
False

How it's computed:

- For each query:
  - If at least one retrieved document includes "Elon Musk" → Hit = 1
  - Otherwise → Hit = 0
- Average across all queries → **Hit Rate**

Example:
If the retriever fetched the following top-3 docs:

sql Copy code

```
1 Tesla designs electric vehicles and energy products.
2 Elon Musk founded SpaceX and leads Tesla as CEO.   ✅ contains gold answer
3 Tesla's Gigafactories are spread across the globe.
```

Hit = 1 because one of the top-3 docs mentions "Elon Musk".

Interpretation:
👉 "Did my retriever at least find one useful doc per query?"

2. MRR (Mean Reciprocal Rank, Retriever Metric)

Goal:

Measures *how high* the relevant document appears in the ranked list.

How it's computed:

- For each query, find the **rank (position)** of the first doc containing the gold answer.
  - Reciprocal Rank = 1 / rank
- Average across queries → **MRR**

1/rank = 1/2 = 0.5

Example:

In the same top-3 list:

- The first relevant doc ("Elon Musk founded SpaceX and leads Tesla...") is at **rank 2**.
- Reciprocal Rank = 1 / 2 = 0.5

  If most queries are like this, your **MRR ≈ 0.5**.

Interpretation:

👉 "Does my retriever rank the right docs near the top?"

Higher MRR = better ranking quality.

Azure RAG + LLM Evaluation

- we will have comparison against Golden Answer / Docs or Context in LLM

🔷 Step 3: Evaluating in an Azure OpenAI RAG System

1. **Retriever** – e.g., Azure Cognitive Search or a vector DB.
2. **Generator** – e.g., Azure OpenAI `gpt-4o-mini`.
3. For each query:
   - Run the retriever → get top-k docs (`contexts`)
   - Run the generator → get the LLM answer (`answer`)
   - Collect together with the **gold answer(s)**

Then run **RAGAS evaluation**:

```python
from ragas import evaluate, EvaluationDataset

eval_dataset = EvaluationDataset.from_list([
    {
        "question": "Who is the CEO of Tesla?",
        "answer": "Elon Musk is the CEO of Tesla.",
        "contexts": ["Tesla, Inc. is led by CEO Elon Musk..."],
        "ground_truths": ["Elon Musk"]
    }
])

results = evaluate(dataset=eval_dataset, llm=evaluator_llm)
print(results)
```

Test Data captured after running { Retriever LLM }

Test Dataset

Output example:

```json
{
  "hit_rate": 1.0,
  "mrr": 0.9,
  "answer_relevancy": 0.98,
  "faithfulness": 0.95
}
```

🔷 Step 4: Understanding the Scores

Metric	What it measures	Example insight
Hit Rate	Retriever found at least one correct doc	If low → retriever misses key info
MRR	How early the correct doc appears	If low → retriever ranks docs poorly
Answer Relevancy	Whether the answer answers the question	If low → LLM not following query
Faithfulness	Whether answer is supported by context	If low → hallucination issue

Imp. we only need

- Question & Gold Answer

for evaluation