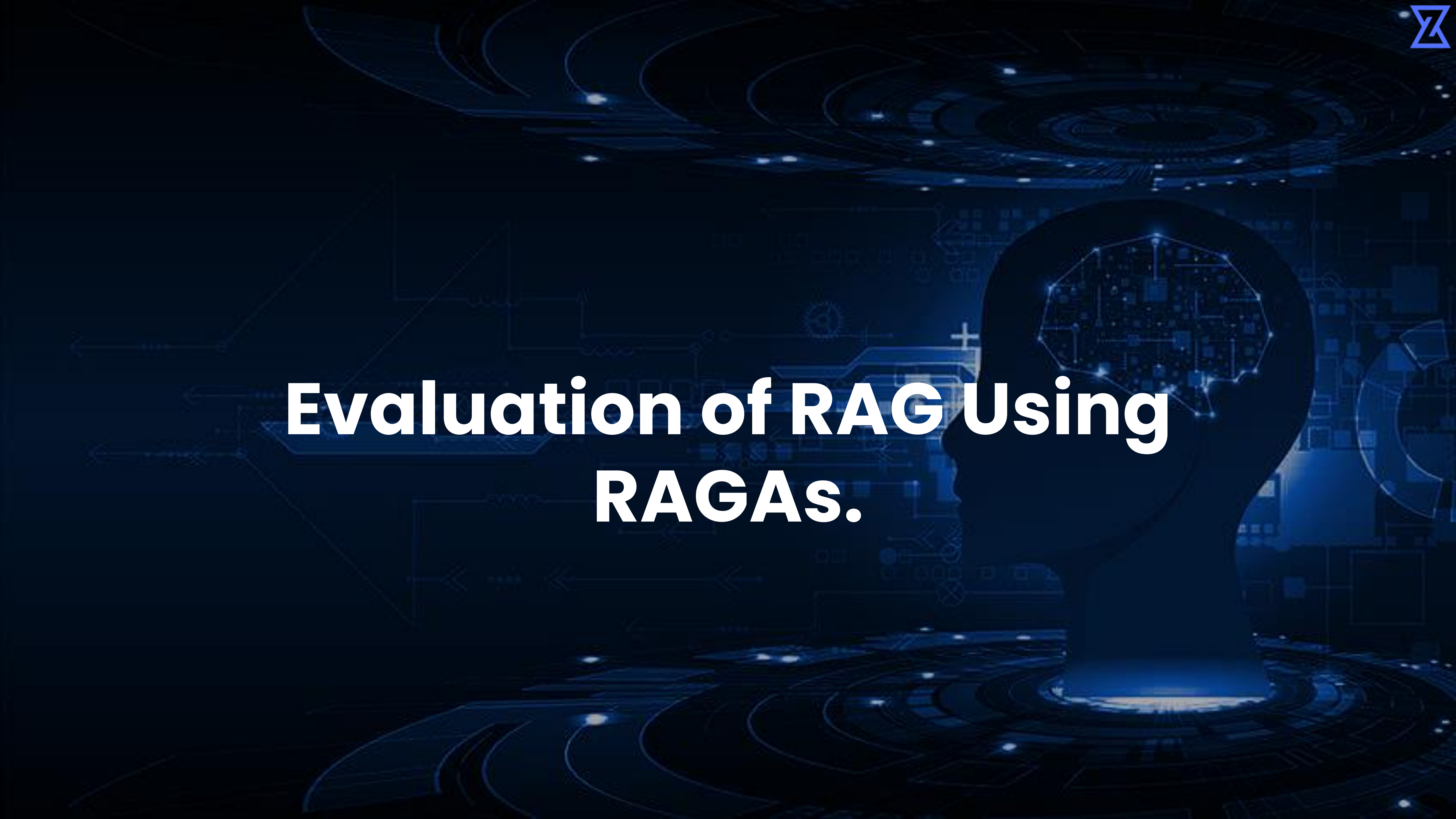


Evaluation of RAG Using RAGAs.

Agenda

- What is RAGAs?
- Confusion Matrix
- Evaluation of Data
- RAGAs Expects following Information
- Evaluation Matrix
- Different types of Evaluation Matrix
- Evaluation Matrix (Retrieval)
- Evaluation Matrix (Generation)
- Evaluation Matrix (end-to-end)
- Colab Implementation of RAGAs
- END

What is RAGAs?

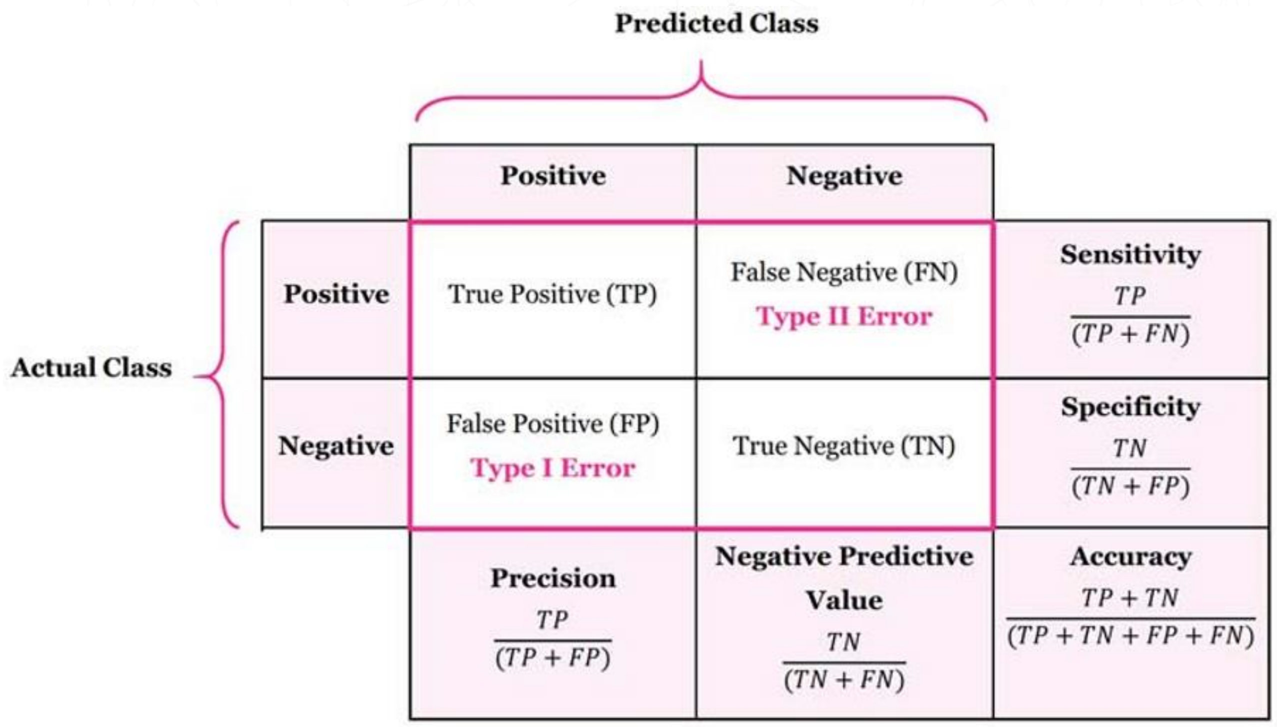
RAGAs

- RAGAs (Retrieval-Augmented Generation Assessment) is a framework (GitHub, Docs) that provides you with the necessary ingredients to help you evaluate your RAG pipeline.
- Ragas is a library that provides tools to supercharge the evaluation of Large Language Model (LLM) applications.
- It is designed to help you evaluate your LLM applications with ease and confidence.

Confusion Matrix

Confusion Matrix

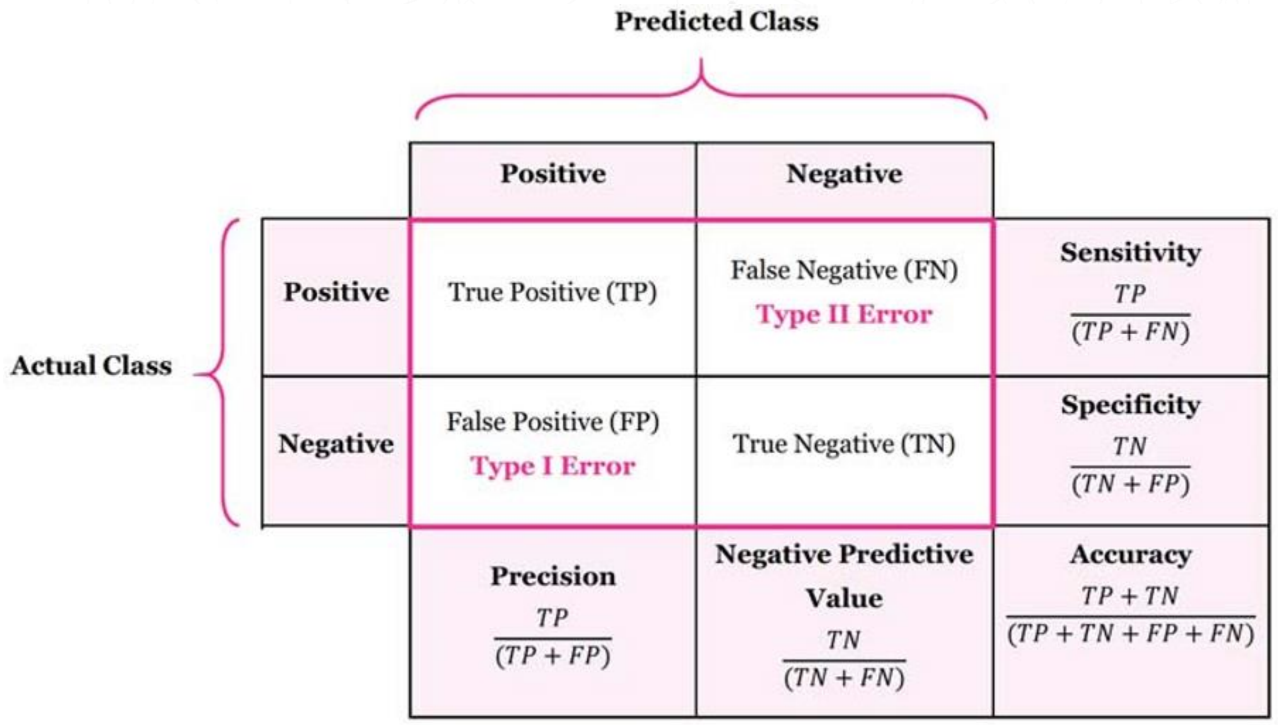
- The confusion matrix is a tool used to evaluate the performance of a model and is visually represented as a table.
- This allows for data practitioners to further analyze their model through fine-tuning.



		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN + FP)}$
		Precision $\frac{TP}{(TP + FP)}$	**Negative Predictive Value** $\frac{TN}{(TN + FN)}$	**Accuracy** $\frac{TP + TN}{(TP + TN + FP + FN)}$

Confusion Matrix

- True Positive (TP) : The actual value is **Yes** and our model is also predicted **Yes**.
- False Negative (FN) : The actual value is **Yes** and our model is predicted **No**.
- True Negative (TN) : The actual value is **No** and our model is predicted **No**.
- False Positive (FP) : The actual value is **No** and our model is predicted **Yes**.



		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Confusion Matrix (Accuracy)

- It measures the total number of correct classifications divided by the total number of cases.

OR

- It is the proportion of the total number of prediction that are correct.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Confusion Matrix (Recall/Sensitivity)

- It measures the total number of true positives divided by the total number of actual positives.

OR

- Out of total positive **(truth)** value, how many were actually predicted to be **positive**.

$$\textbf{\textit{Recall}} = \frac{\text{TP}}{\text{TP + FN}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{Actual Results}}$$

Confusion Matrix (Precision)

- It measures the total number of true positives divided by the total number of predicted positives.

OR

- Out of the total positive **predicted** result, how many were actually **positive**.

$$\textit{Precision} = \frac{TP}{TP + FP} \quad \text{or} \quad \frac{\text{True Positive}}{\text{Predictive Results}}$$

Confusion Matrix (Specificity)

- It measures the total number of true negatives divided by the total number of actual negatives.

$$\textit{Specificity} = \frac{TN}{TN + FP}$$

Confusion Matrix (Example)

• Amongst the 200 emails, 80 emails are actually spam in which the model correctly identifies **60** of them as spam (TP).

• Amongst the 200 emails, 120 emails are not spam in which the model correctly identifies **100** of them as not spam (TN).

• Amongst the 200 emails, the model incorrectly identifies **20** non-spam emails as spam (FP).

• Amongst the 200 emails, the model misses **20** spam emails and identifies them as non-spam (FN).

Actual / Predicted	Spam (Positive)	Not Spam (Negative)
Spam (Positive)	60 (TP)	20 (FN)
Not Spam (Negative)	20 (FP)	100 (TN)

Confusion Matrix (Example)

$$\boxed{Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{60+100}{60+100+20+20} = \frac{160}{200} = 0.8}$$

$$\boxed{Recall = \frac{TP}{TP+FN} = \frac{60}{60+20} = \frac{60}{80} = 0.75}$$

$$\boxed{Precision = \frac{TP}{TP+FP} = \frac{60}{60+20} = \frac{60}{80} = 0.75}$$

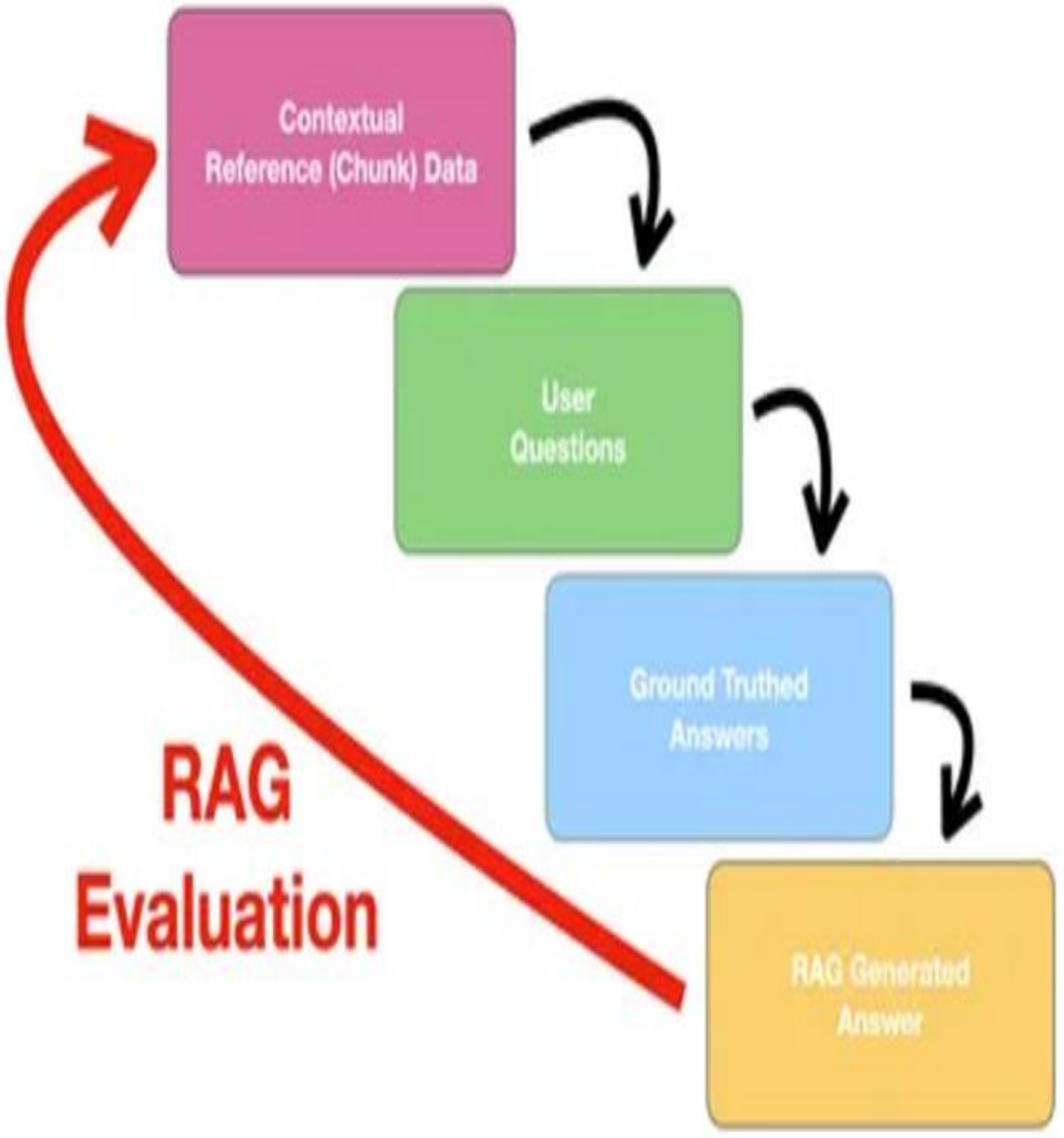
$$\boxed{Specificity = \frac{TN}{TN+FP} = \frac{100}{100+20} = \frac{100}{120} \approx 0.833}$$

Evaluation of Data

Evaluation of Data

- RAGAs (Retrieval-Augmented Generation Assessments) took an innovative approach to evaluation by eliminating the need for human created reference answers or "ground truth" labels. Instead, it uses large language models themselves to assess the quality of responses.

- This "reference-free" approach marked a significant shift from traditional evaluation methods that relied heavily on human-annotated datasets.



RAGAs Expects the Following Information

- **`question`:** The user query that is the input of the RAG pipeline. The input.

- **`answer`:** The generated answer from the RAG pipeline. The output.

- **`contexts`:** The contexts retrieved from the external knowledge source used to answer the **question**.

- **`ground truth`:** The ground truth answer the question. This is the only human-annotated information.

Evaluation Matrix

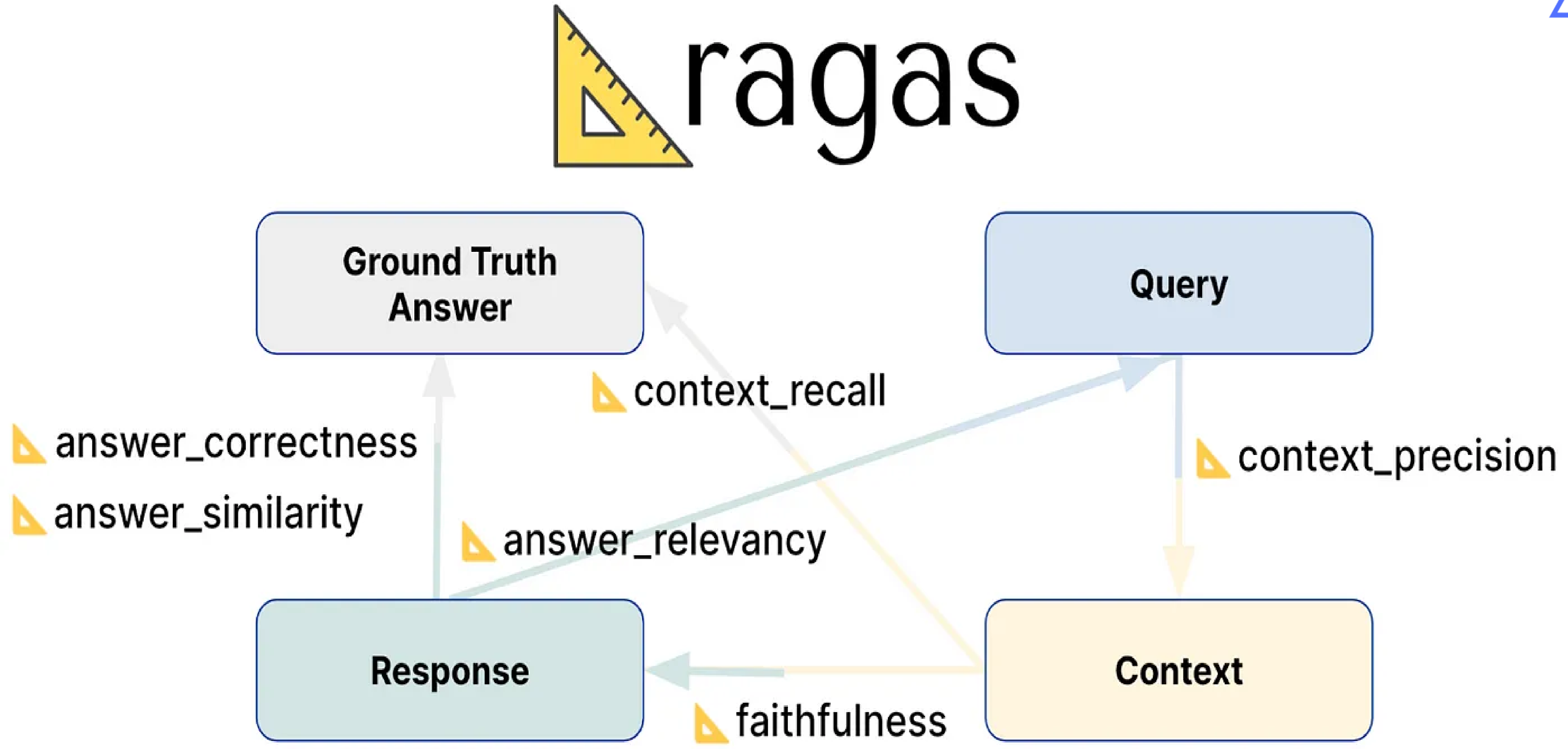
Evaluation Matrix

- RAGAs provide you with a few metrics to evaluate a RAG pipeline component-wise as well end-to-end.

- On a **component level,** RAGAs provides you with metrics to evaluate the retrieval component (`**context_relevancy**` and `**context_recall**`) and the generative component (`**faithfulness**` and `**answer_relevancy**`) separately.

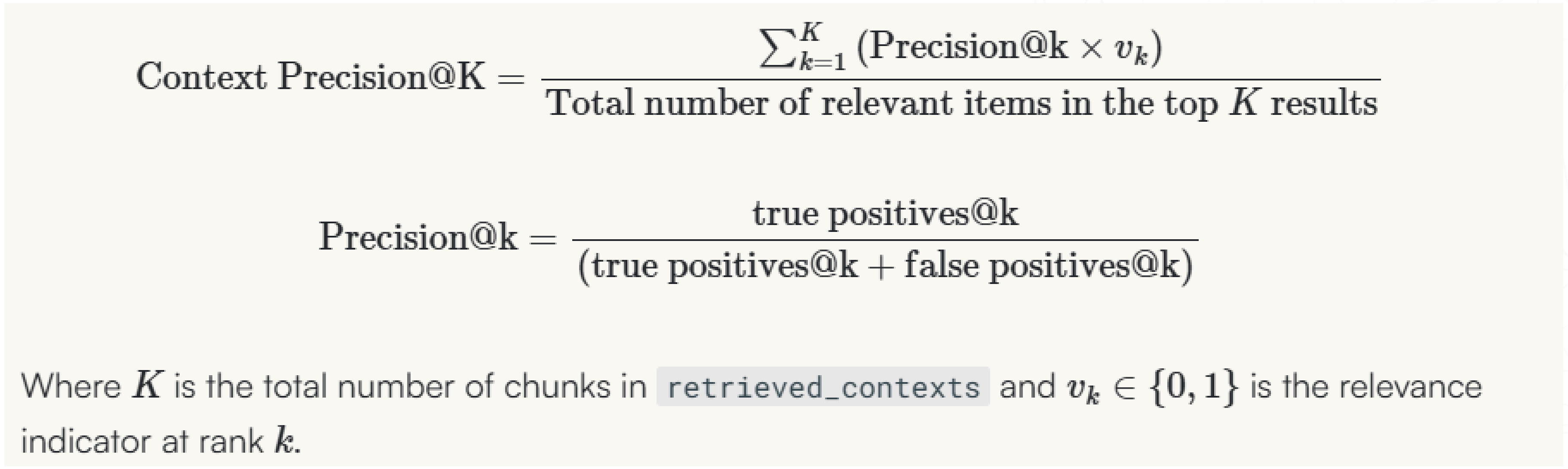
- **All metrics are scaled to the range [0, 1], with higher values indicating a better performance.**

Different Types of Evaluation Matrix


ragas
Ground Truth Answer
Query
Response
Context
context_recall
answer_correctness
answer_similarity
answer_relevancy
context_precision
faithfulness

Context Precision (Retrieval)

- This metric measure the proportion of relevant chunks in the **retrieved_contexts**.
- It measure weather all of the ground-truth relevant items present in the `**contexts**` are ranked higher or not. Ideally all the relevant chunks must appear at the top ranks. This metric is computed using the `**question**` and `**context**`.
- It is used to measure information density.



$$\text{Context Precision@K} = \frac{\sum_{k=1}^{K} \left(\text{Precision@k} \times v_k\right)}{\text{Total number of relevant items in the top } K \text{ results}}$$

$$\text{Precision@k} = \frac{\text{true positives@k}}{(\text{true positives@k} + \text{false positives@k})}$$

Where K is the total number of chunks in `retrieved_contexts` and $v_k \in \{0, 1\}$ is the relevance indicator at rank k.

Context Recall (Retrieval)

- It measures how many of the relevant documents (or pieces of information) were successfully retrieved.
- It focuses on not missing important results. Higher recall means fewer relevant documents were left out. In short, recall is about not missing anything important.
- This metric is computed based on the `**ground_truth**` and the `**contexts**`.
- **This is the only metric in the framework that relies on human-annotated ground truth labels.**

$$\text{context recall} = \frac{|\text{GT claims that can be attributed to context}|}{|\text{Number of claims in GT}|}$$

Faithfulness (Generation)

- This metric measures the factual consistency of the generated answer against the given context.
- The number of correct statements from the given contexts is divided by the total number of statement in the generated answer.
- This metric uses the ` **retrieved _context**` and `**response**`.

Faithfulness Score = $$\frac{|\text{Number of claims in the generated answer that can be inferred from given context}|}{|\text{Total number of claims in the generated answer}|}$$

Answer Relevancy (Generation)

- This metric focuses on assessing how pertinent the generated answer is to the given prompt/question.
- This metric is computed using `**user_input**`, the `**retrieved_contexts**` and the `**response**`.
- The Answer Relevancy is defined as the mean cosine similarity of the original `**user_input**` to a number of artificial questions, which where generated (reverse engineered) based on the `**response**`

$$\text{answer relevancy} = \frac{1}{N}\sum_{i=1}^{N} cos(E_{g_i}, E_o)$$

$$\text{answer relevancy} = \frac{1}{N}\sum_{i=1}^{N} \frac{E_{g_i} \cdot E_o}{\|E_{g_i}\|\|E_o\|}$$

Where:

- E_{g_i} is the embedding of the generated question i.
- E_o is the embedding of the original question.
- N is the number of generated questions, which is 3 default.

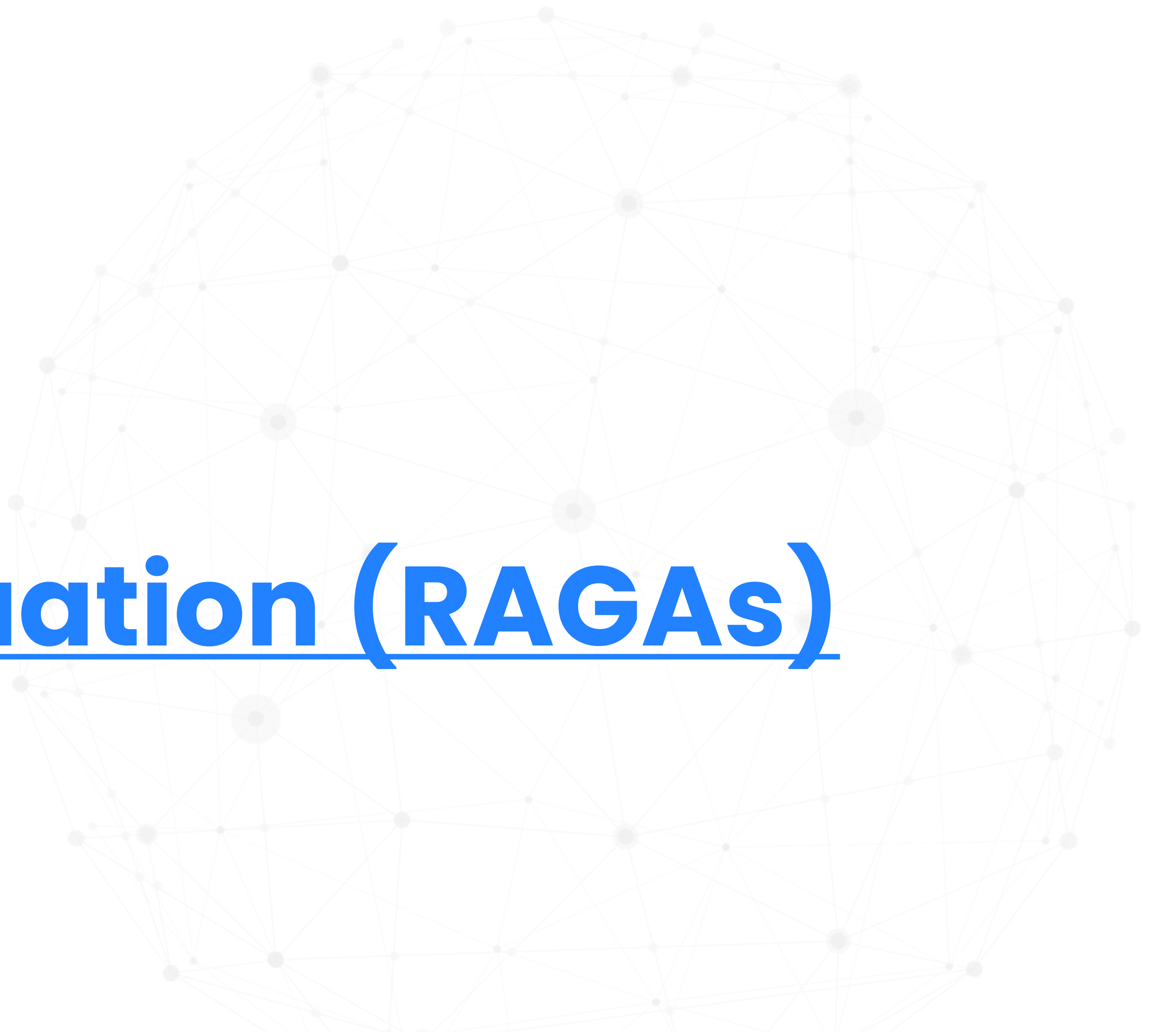
Answer Semantic Similarity (end-to-end)

- The concept of Answer Semantic Similarity pertains to the assessment of the semantic resemblance between the generated answer and the ground truth.
- This evaluation is based on the ground truth and the answer, with values falling within the range of 0 to 1.
- A higher score signifies a better alignment between the generated answer and the ground truth.
- Measuring the semantic similarity between answers can offer valuable insights into the quality of the generated response.
- This evaluation utilizes a cross-encoder model to calculate the semantic similarity score.

Answer Correctness (end-to-end)

- It involves measuring the accuracy of the generated answer when compares to the ground truth.
- This metric is computed using **`ground_truth`** and the **`answer`**.
- Answer correctness encompasses two critical aspects: semantic similarity between the generated answer and the ground truth, as well as factual similarity. These aspects are combined using a weighted scheme to formulate the answer correctness score.

Colab Implementation of RAGAs



Link ›› RAG Evaluation (RAGAs)

Thank You