RAG Evaluation using Ragas

[ ]:
```
1. retrieval
2. generation
```

[ ]:
```
!pip install -q langchain openai weaviate-client ragas Dataset
```

[ ]:
```
import requests
from langchain.document_loaders import TextLoader
from langchain.text_splitter import CharacterTextSplitter
```

[ ]:
```
url="https://raw.githubusercontent.com/langchain-ai/langchain/master/docs/docs/modules/state_of_the_union.txt"
```

[ ]:
```
res = requests.get(url)
```

[ ]:
```
res.text
```

[ ]:
```
with open("state_of_the_union.txt", "w") as f:
    f.write(res.text)
```

[ ]:
```
loader=TextLoader("./state_of_the_union.txt")
```

[ ]:
```
documents=loader.load()
```

[ ]:
```
from langchain.text_splitter import CharacterTextSplitter
text_splitter=CharacterTextSplitter(chunk_size=500,chunk_overlap=50)
```

[ ]:
```
chunks=text_splitter.split_documents(documents)
```

[ ]:
```
chunks[2].page_content
```

[ ]:
```
'Groups of citizens blocking tanks with their bodies. Everyone from students to
retirees teachers turned soldiers defending their homeland. \n\nIn this struggle
as President Zelenskyy said in his speech to the European Parliament “Light will
win over darkness.” The Ukrainian Ambassador to the United States is here
tonight. \n\nLet each of us here tonight in this Chamber send an unmistakable
signal to Ukraine and to the world.'
```

[ ]:
```
len(chunks)
```

[ ]: 90

[ ]:
```
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import Weaviate
import weaviate
from weaviate.embedded import EmbeddedOptions
```

[ ]:
```
from google.colab import userdata
OPENAI_API_KEY=userdata.get('OPENAI_API_KEY')
```

[ ]:
```
import os
os.environ["OPENAI_API_KEY"]=OPENAI_API_KEY
```

[ ]:
```
client=weaviate.Client(
    embedded_options=EmbeddedOptions()
)
```

```
Binary /root/.cache/weaviate-embedded did not exist. Downloading binary from htt
ps://github.com/weaviate/weaviate/releases/download/v1.23.7/weaviate-v1.23.7-Lin
ux-amd64.tar.gz
Started /root/.cache/weaviate-embedded: process ID 3695
```

[ ]:
```
vectorstore=Weaviate.from_documents(
    client=client,
    documents=chunks,
    embedding=OpenAIEmbeddings(),
    by_text=False

)
```

[ ]:
```
retriever=vectorstore.as_retriever()
```

[ ]:
```
from langchain.chat_models import ChatOpenAI
```

[ ]:
```
from langchain.prompts import ChatPromptTemplate
from langchain.schema.runnable import RunnablePassthrough
from langchain.schema.output_parser import StrOutputParser
```

[ ]:
```
  llm=ChatOpenAI(model_name="gpt-3.5-turbo",temperature=0.2)
```

[ ]:
```
# Define prompt template
template = """You are an assistant for question-answering tasks.
Use the following pieces of retrieved context to answer the question.
If you don't know the answer, just say that you don't know.
Use two sentences maximum and keep the answer concise.
```

```
Question: {question}
Context: {context}
Answer:
"""
```

[ ]:

```
prompt = ChatPromptTemplate.from_template(template)
```

[ ]:

```
rag_chain=(
    {"context":retriever,   "question":RunnablePassthrough()}
            | prompt
            | llm
            | StrOutputParser()
)
```

[ ]:

```
res.text
```

[ ]:

```
questions=["what did the President say about Justic Breyer?",
           "What did the President say abou Intel's CEO?",
           "What did the President say about gun violence?"

           ]
```

[ ]:

```
groud_truths=[["The president said that Justice Breyer has dedicated his life to serve the country and thanked him for his service."],
              ["The president said that Pat Gelsinger is ready to increase Intel's investment to $100 billion."],
              ["The president asked Congress to pass proven measures to reduce gun violence."]]
```

[ ]:

```
query
```

[ ]: 'what did the President say about Justic Breyer?'

[ ]:

```
rag_chain.invoke(query)
```

[ ]: 'The President honored Justice Breyer for his service and mentioned nominating Judge Ketanji Brown Jackson to continue his legacy.'

[ ]:

```
answer=[]
context=[]
```

[ ]:

```
for query in questions:
  answer.append(rag_chain.invoke(query))
  context.append([docs.page_content  for docs in retriever.get_relevant_documents(query)])
```

[ ]:

```
answer
```

[ ]:
```
['The President honored Justice Breyer for his service and mentioned nominating
Judge Ketanji Brown Jackson to continue his legacy.',
 "The President mentioned that Intel's CEO, Pat Gelsinger, is ready to increase
their investment from $20 billion to $100 billion, which would be one of the
biggest investments in manufacturing in American history.",
 'The President called for Congress to pass measures to reduce gun violence,
including universal background checks and banning assault weapons and high-
capacity magazines. He also emphasized the need to crack down on gun trafficking
and ghost guns.']
```

[ ]:
```python
context
```

[ ]:
```python
# To dict
data = {
    "question": questions,
    "answer": answer,
    "contexts": context,
    "ground_truths": groud_truths
}
```

[ ]:
```python
from datasets import Dataset
dataset=Dataset.from_dict(data)
```

[ ]:
```python
from ragas import evaluate
from ragas.metrics import (
    faithfulness,
    answer_relevancy,
    context_recall,
    context_precision,
)
```

[ ]:
```python
result=evaluate(
    dataset=dataset,
    metrics=[
        context_precision,
        context_recall,
        faithfulness,
        answer_relevancy
    ]
)
```

```
WARNING:ragas.validation:passing column names as 'ground_truths' is deprecated
and will be removed in the next version, please use 'ground_truth' instead. Note
that `ground_truth` should be of type string and not Sequence[string] like
`ground_truths`
```

```
Evaluating:   0%|          | 0/12 [00:00<?, ?it/s]
```

```
[ ]: result
```

```
[ ]: {'context_precision': 1.0000, 'context_recall': 1.0000, 'faithfulness': 1.0000,
      'answer_relevancy': 0.8727}
```

```
[ ]: df=result.to_pandas()
```

```
[ ]: df
```

```
[ ]: df.iloc[0,:]
```

```
[ ]: question                both what did the President say about Justic Breyer?
     answer                  The President honored Justice Breyer for his s...
     contexts                [Tonight, I'd like to honor someone who has de...
     ground_truths           [The president said that Justice Breyer has de...
     ground_truth            The president said that Justice Breyer has ded...
     context_precision                                                     1.0
     context_recall                                                        1.0
     faithfulness                                                          1.0
     answer_relevancy                                                  0.83073
     Name: 0, dtype: object
```

```
[ ]: #data
     #this is my actual query
     df.iloc[0,:].question
```

```
[ ]: 'what did the President say about Justic Breyer?'
```

```
[ ]: #predicted
     #generation of the model
     df.iloc[0,:].answer
```

```
[ ]: 'The President honored Justice Breyer for his service and mentioned nominating
      Judge Ketanji Brown Jackson to continue his legacy.'
```

```
[ ]: #actual
     #actual answer(manually we have written this answer)(this could be llm␣
      ↪generated)
     df.iloc[0,:].ground_truth
```

```
[ ]: 'The president said that Justice Breyer has dedicated his life to serve the
      country and thanked him for his service.'
```

```
[ ]: #average value
     #retrieval result(context)
     df.iloc[0,:].contexts[0]
```

```
[ ]: 'Tonight, I'd like to honor someone who has dedicated his life to serve this
     country: Justice Stephen Breyer—an Army veteran, Constitutional scholar, and
     retiring Justice of the United States Supreme Court. Justice Breyer, thank you
     for your service. \n\nOne of the most serious constitutional responsibilities a
     President has is nominating someone to serve on the United States Supreme
     Court.'
```

```
[ ]: context_precision                                              1.0
     context_recall                                                 1.0
     faithfulness                                                   1.0
     answer_relevancy                                           0.83073
```

```
[ ]: raga:https://docs.ragas.io/en/latest/concepts/metrics/context_recall.html
     TrueLence:https://www.trulens.org/trulens_eval/getting_started/core_concepts/
     ↪rag_triad/
     ares:https://github.com/stanford-futuredata/ARES?tab=readme-ov-file#section3
```

```
[ ]: #rouge summrization
     #bleu translation
```

```
[ ]: from datasets import Dataset
     from ragas.metrics import answer_similarity
     from ragas import evaluate


     data_samples = {
         'question': ['When was the first super bowl?', 'Who won the most super␣
     ↪bowls?'],
         'answer': ['The first superbowl was held on Jan 15, 1967', 'The most super␣
     ↪bowls have been won by The New England Patriots'],
         'ground_truth': ['The first superbowl was held on January 15, 1967', 'The␣
     ↪New England Patriots have won the Super Bowl a record six times']
     }
     dataset = Dataset.from_dict(data_samples)
     score = evaluate(dataset,metrics=[answer_similarity])
     score.to_pandas()
```

```
Evaluating:   0%|          | 0/2 [00:00<?, ?it/s]
```

```
[ ]:                           question  \
     0   When was the first super bowl?
     1     Who won the most super bowls?

                                                    answer  \
     0         The first superbowl was held on Jan 15, 1967
     1   The most super bowls have been won by The New ...
```

```
                                        ground_truth  answer_similarity
0  The first superbowl was held on January 15, 1967           0.996378
1  The New England Patriots have won the Super Bo...          0.924285
```

```
[ ]: from datasets import Dataset
     from ragas.metrics import faithfulness, answer_correctness
     from ragas import evaluate

     data_samples = {
         'question': ['When was the first super bowl?', 'Who won the most super␣
       ↪bowls?'],
         'answer': ['The first superbowl was held on Jan 15, 1967', 'The most super␣
       ↪bowls have been won by The New England Patriots'],
         'ground_truth': ['The first superbowl was held on January 15, 1967', 'The␣
       ↪New England Patriots have won the Super Bowl a record six times']
     }
     dataset = Dataset.from_dict(data_samples)
     score = evaluate(dataset,metrics=[answer_correctness])
     score.to_pandas()
```

```
Evaluating:   0%|          | 0/2 [00:00<?, ?it/s]
```

```
[ ]:                          question  \
     0  When was the first super bowl?
     1   Who won the most super bowls?

                                                  answer  \
     0      The first superbowl was held on Jan 15, 1967
     1  The most super bowls have been won by The New ...

                                             ground_truth  answer_correctness
     0  The first superbowl was held on January 15, 1967            0.749095
     1  The New England Patriots have won the Super Bo...           0.731078
```