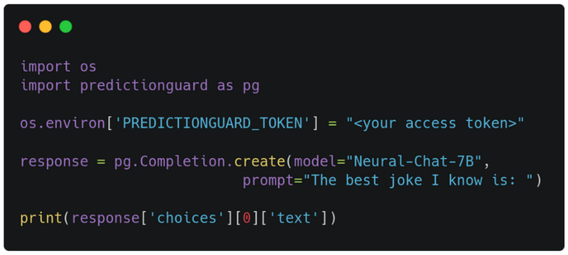
prediction
Guard

intel. liftoff

LLM CHEAT SHEET

BASIC PROMPTING

Single input prompt, output completion



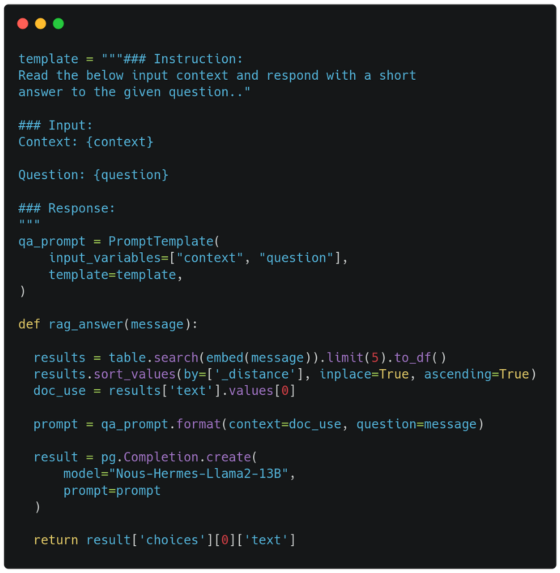
```python
import os
import predictionguard as pg

os.environ['PREDICTIONGUARD_TOKEN'] = "<your access token>"

response = pg.Completion.create(model="Neural-Chat-7B",
                                prompt="The best joke I know is: ")

print(response['choices'][0]['text'])
```

MODEL OPTIONS

And corresponding prompt formats

RETRIEVAL AUGMENTED GENERATION (RAG)

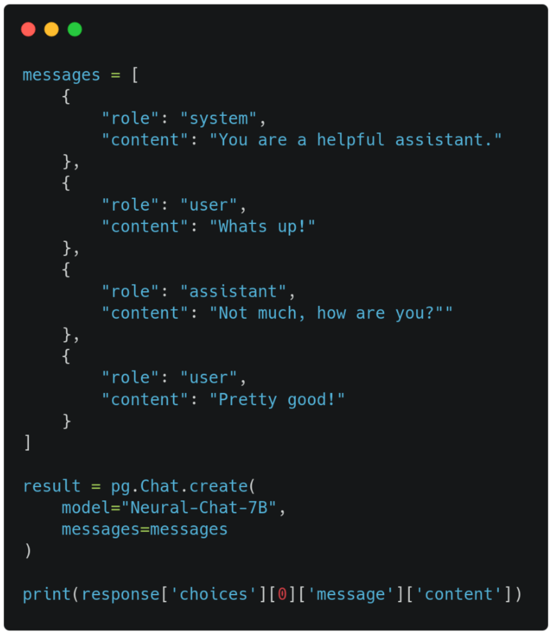
Retrieve and inject data to ground LLM outputs



```python
template = """### Instruction:
Read the below input context and respond with a short
answer to the given question.."

### Input:
Context: {context}

Question: {question}

### Response:
"""
qa_prompt = PromptTemplate(
    input_variables=["context", "question"],
    template=template,
)

def rag_answer(message):

  results = table.search(embed(message)).limit(5).to_df()
  results.sort_values(by=['_distance'], inplace=True, ascending=True)
  doc_use = results['text'].values[0]

  prompt = qa_prompt.format(context=doc_use, question=message)

  result = pg.Completion.create(
      model="Nous-Hermes-Llama2-13B",
      prompt=prompt
  )

  return result['choices'][0]['text']
```

CHAT COMPLETIONS

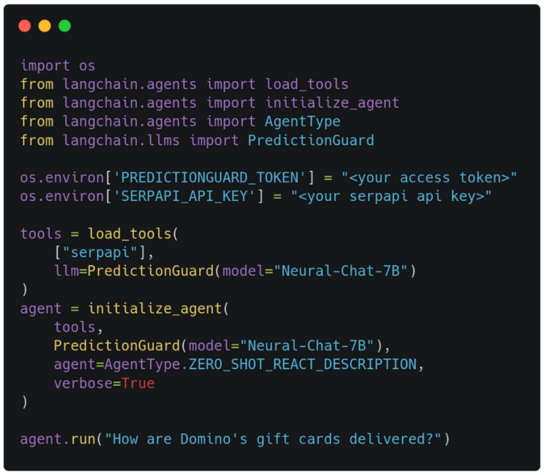
Message thread input, chat response



```python
messages = [
    {
        "role": "system",
        "content": "You are a helpful assistant."
    },
    {
        "role": "user",
        "content": "Whats up!"
    },
    {
        "role": "assistant",
        "content": "Not much, how are you?""
    },
    {
        "role": "user",
        "content": "Pretty good!"
    }
]

result = pg.Chat.create(
    model="Neural-Chat-7B",
    messages=messages
)

print(response['choices'][0]['message']['content'])
```

AGENTS

Choose and complete a sequence of actions



```python
import os
from langchain.agents import load_tools
from langchain.agents import initialize_agent
from langchain.agents import AgentType
from langchain.llms import PredictionGuard

os.environ['PREDICTIONGUARD_TOKEN'] = "<your access token>"
os.environ['SERPAPI_API_KEY'] = "<your serpapi api key>"

tools = load_tools(
    ["serpapi"],
    llm=PredictionGuard(model="Neural-Chat-7B")
)
agent = initialize_agent(
    tools,
    PredictionGuard(model="Neural-Chat-7B"),
    agent=AgentType.ZERO_SHOT_REACT_DESCRIPTION,
    verbose=True
)

agent.run("How are Domino's gift cards delivered?")
```

FREE access token for usage at: tinyurl.com/pg-intel-hack

More details at docs.predictionguard.com