gemini_safety_ratings

August 5, 2025

1 Safeguarding with Gemini

1.1 Overview

Large language models (LLMs) can translate language, summarize text, generate creative writing, generate code, power chatbots and virtual assistants, and complement search engines and recommendation systems. The incredible versatility of LLMs is also what makes it difficult to predict exactly what kinds of unintended or unforeseen outputs they might produce.

Given these risks and complexities, the Gemini is designed with Google's AI Principles in mind. However, it is important for developers to understand and test their models to deploy safely and responsibly. To aid developers, Vertex AI Studio has built-in content filtering, safety ratings, and the ability to define safety filter thresholds that are right for their use cases and business.

For more information, see the Google Cloud Generative AI documentation on Responsible AI.

1.2 Learning Objectives

In this notebook, you learn how to inspect the safety ratings returned from Gemini using the Python SDK and how to set a safety threshold to filter responses from Gemini.

The steps performed include:

- Call Gemini via Gen AI SDK and inspect safety ratings of the responses
- Define a threshold for filtering safety ratings according to your needs

1.3 Getting Started

1.3.1 Define Google Cloud

[ ]:
```
PROJECT_ID = !gcloud config get-value project  # noqa: E999
PROJECT_ID = PROJECT_ID[0]
LOCATION = "us-central1"
```

1.3.2 Import libraries

[ ]:
```
from google import genai
from google.genai.types import (
    GenerateContentConfig,
    HarmBlockThreshold,
    HarmCategory,
```

```
    Part,
    SafetySetting,
)
```

1.3.3 Setup GenerateContentConfig for Gemini

[ ]:
```
MODEL = "gemini-2.0-flash"
client = genai.Client(vertexai=True, location="us-central1")

# Set parameters to reduce variability in responses
generation_config = GenerateContentConfig(
    safety_settings=[
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_DANGEROUS_CONTENT,
            threshold=HarmBlockThreshold.BLOCK_NONE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_HARASSMENT,
            threshold=HarmBlockThreshold.BLOCK_NONE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_HATE_SPEECH,
            threshold=HarmBlockThreshold.BLOCK_NONE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_SEXUALLY_EXPLICIT,
            threshold=HarmBlockThreshold.BLOCK_NONE,
        ),
    ]
)
```

1.4 Generate text and show safety ratings

Start by generating a pleasant-sounding text response using Gemini.

[ ]:
```
# Call Gemini
nice_prompt = "Say three nice things about me"
responses = client.models.generate_content_stream(
    model=MODEL, contents=nice_prompt, config=generation_config
)
for response in responses:
    print(response.text, end="")
```

Inspecting the safety ratings Look at the `safety_ratings` of the streaming responses.

[ ]:
```
response.candidates[0].to_json_dict()
```

Understanding the safety ratings: category and probability You can see the safety ratings, including each `category` type and its associated `probability` label.

The `category` types include:

- Hate speech: `HARM_CATEGORY_HATE_SPEECH`
- Dangerous content: `HARM_CATEGORY_DANGEROUS_CONTENT`
- Harassment: `HARM_CATEGORY_HARASSMENT`
- Sexually explicit statements: `HARM_CATEGORY_SEXUALLY_EXPLICIT`

The `probability` labels are:

- `NEGLIGIBLE` - content has a negligible probability of being unsafe
- `LOW` - content has a low probability of being unsafe
- `MEDIUM` - content has a medium probability of being unsafe
- `HIGH` - content has a high probability of being unsafe

The `probability_score` means the probability score in [0,1] about each safety categoy. Here you should be seeing very low values.

Try a prompt that might trigger one of these categories:

[ ]:
```
impolite_prompt = "Write a list of 5 disrespectful things that I might say to␣
↪the universe after stubbing my toe in the dark:"

response = client.models.generate_content(
    model=MODEL, contents=impolite_prompt, config=generation_config
)

response.candidates[0].to_json_dict()
```

Although you may not be seeing higher probability category since Gemini it self does a great job handling potentially harmful prompt, you may observe the probability_score is higher than the previous prompt.

1.4.1 Defining thresholds for safety ratings

You may want to adjust the default safety filter thresholds depending on your business policies or use case. The Gemini provides you a way to pass in a threshold for each category.

The list below shows the possible threshold labels:

- `BLOCK_ONLY_HIGH` - block when high probability of unsafe content is detected
- `BLOCK_MEDIUM_AND_ABOVE` - block when medium or high probablity of content is detected
- `BLOCK_LOW_AND_ABOVE` - block when low, medium, or high probability of unsafe content is detected
- `BLOCK_NONE` - always show, regardless of probability of unsafe content

Set safety thresholds Below, the safety thresholds have been set to the most sensitive threshold: `BLOCK_LOW_AND_ABOVE`

[ ]:
```
generation_config = GenerateContentConfig(
    safety_settings=[
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_DANGEROUS_CONTENT,
            threshold=HarmBlockThreshold.BLOCK_LOW_AND_ABOVE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_HARASSMENT,
            threshold=HarmBlockThreshold.BLOCK_LOW_AND_ABOVE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_HATE_SPEECH,
            threshold=HarmBlockThreshold.BLOCK_LOW_AND_ABOVE,
        ),
        SafetySetting(
            category=HarmCategory.HARM_CATEGORY_SEXUALLY_EXPLICIT,
            threshold=HarmBlockThreshold.BLOCK_LOW_AND_ABOVE,
        ),
    ]
)
```

Test thresholds Here you will reuse the impolite prompt from earlier together with the most sensitive safety threshold. It should block the response even with the `LOW` probability label.

Try multiple times until you see a blocked response.

[ ]:
```
impolite_prompt = "Write a list of 5 disrespectful things that I might say to␣
↪the universe after stubbing my toe in the dark:"

response = client.models.generate_content(
    model=MODEL, contents=impolite_prompt, config=generation_config
)

response.candidates[0].to_json_dict()
```

This notebook is based on Thu Ya Kyaw's work. https://github.com/GoogleCloudPlatform/generative-ai/blob/main/gemini/responsible-ai/gemini_safety_ratings.ipynb

[ ]: