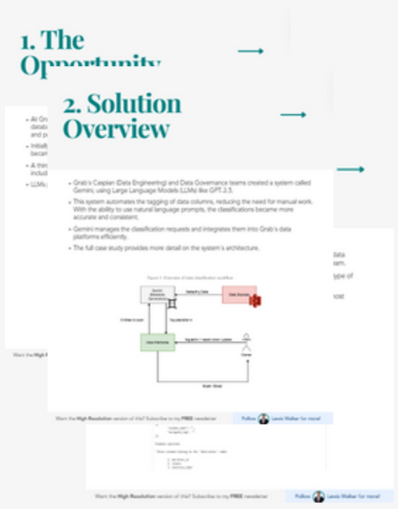


How Grab Classify Data Better with Generative AI

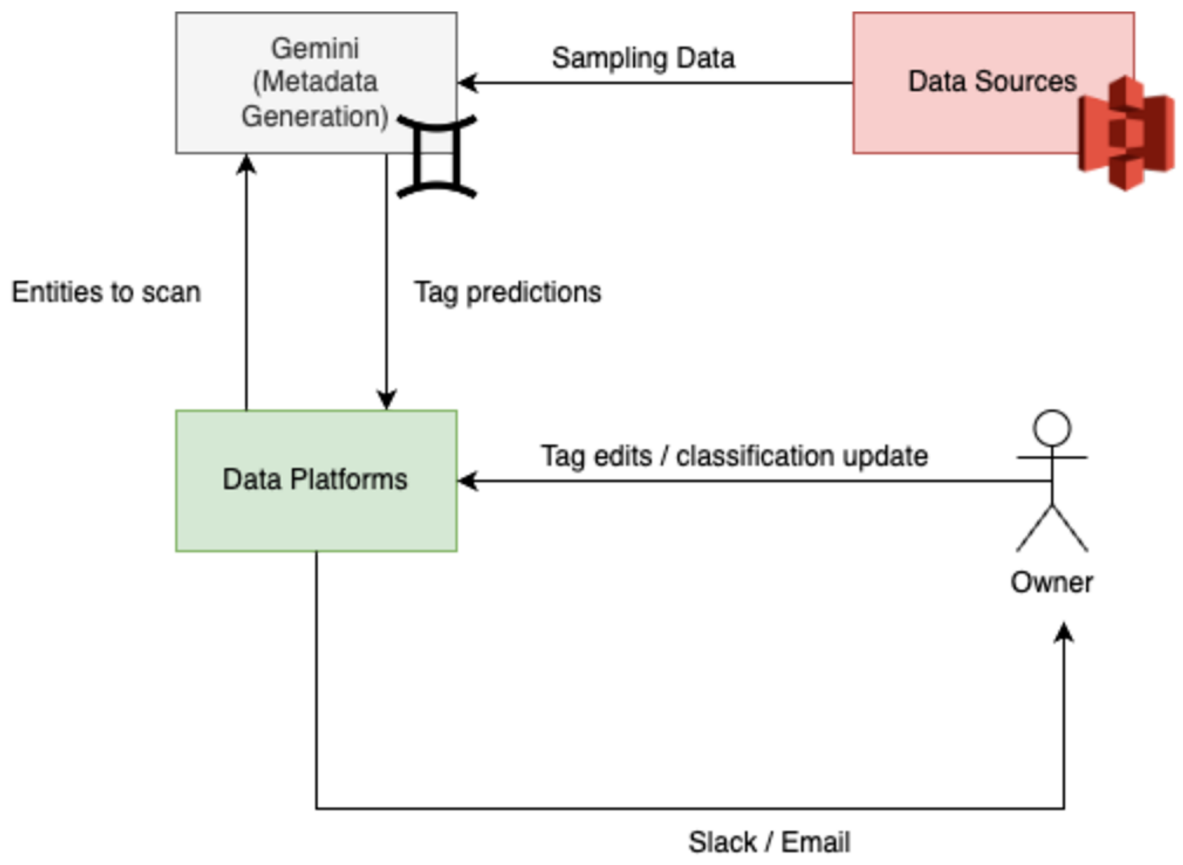
1. The Opportunity

- At Grab, 'The Uber of Southest Asia', managing and understanding vast data, from database tables to Kafka schemas, is critical for securing information for drivers, users, and partners, while ensuring appropriate access for data analysts.
- Initially, classifying Personal Identifiable Information (PII) was done manually, which became slow and inconsistent as the data grew with overly restricted data.
- A third-party classification service was leveraged but rules-based challenges included classifier customization effort and false positives.
- LLMs presented the opportunity for a more scalable and accurate solution.

2. Solution Overview

- Grab's Caspian (Data Engineering) and Data Governance teams created a system called Gemini, using Large Language Models (LLMs) like GPT-3.5.
- This system automates the tagging of data columns, reducing the need for manual work. With the ability to use natural language prompts, the classifications became more accurate and consistent.
- Gemini manages the classification requests and integrates them into Grab's data platforms efficiently.
- The full case study provides more in-depth detail on the system architecture.

Figure 1: Overview Of Data Classification Workflow



3. Classification Prompt

- The LLM-powered column-level tag classification works as follows:
  - Given a data entity with a defined schema, we tag each field with metadata classifications based on an internal scheme from the data governance team.
  - For example, a field might be tagged as a specific business metric or a type of personally identifiable information (PII).
  - The language model is used as a column tag generator, assigning the most appropriate tag to each column.
  - Below is the prompt used:

Figure 2: Classification Prompt

```
You are a database column tag classifier, your job is to assign the most appropriate tag based on table name a
nd column name. The database columns are from a company that provides ride-hailing, delivery, and financial se
rvices. Assign one tag per column. However not all columns can be tagged and these columns should be assigned
<None>. You are precise, careful and do your best to make sure the tag assigned is the most appropriate.

The following is the list of tags to be assigned to a column. For each line, left hand side of the : is the ta
g and right hand side is the tag definition

…
<Personal.ID> : refers to government-provided identification numbers that can be used to uniquely identify a p
erson and should be assigned to columns containing "NRIC", "Passport", "FIN", "License Plate", "Social Securit
y" or similar. This tag should absolutely not be assigned to columns named "id", "merchant id", "passenger i
d", “driver id" or similar since these are not government-provided identification numbers. This tag should be
very rarely assigned.

<None> : should be used when none of the above can be assigned to a column.
…

Output Format is a valid json string, for example:

[{
        "column_name": "",
        "assigned_tag": ""
}]

Example question

`These columns belong to the "deliveries" table

        1. merchant_id
        2. status
        3. delivery_time`
```

4. Classification Ouput

- Grab needed the language model's output in a fixed format for downstream processing. Therefore, they used prompt engineering to ensure the output is usable.
- Here are techniques they found effective:
  - **Clarify Requirements:** Clearly define the task; the model will only do what you ask.
  - **Few-Shot Learning:** Provide examples to guide the model's responses.
  - **Schema Enforcement:** Provide the Data Transfer Object (DTO) schema to ensure the output conforms to it.
  - **Handle Uncertainty:** Include a default <None> tag for cases where the model is unsure or confused.

5. The Impact →

- Within a month of system roll out, Grab scanned over 20,000 data entities, projected to save approximately 360 man-days annually through automated tagging.
- The classified tags enable further use cases, such as determining data sensitivity tiers and enforcing Attribute-based Access Control (ABAC) and Dynamic Data Masking.
- In addition to significant benefits, the system remains cost-effective, allowing for scalability across more data entities.

6. Moving Forward

- **Prompt Improvement:** Grab is exploring ways to feed sample data and user feedback into the system to improve accuracy. They're also testing confidence level outputs from the LLM to reduce manual verification by only involving users when the model is uncertain.
- **Prompt Evaluation:** Grab is building analytical pipelines to measure the performance of prompts, helping refine them more effectively.
- **Scaling:** Grab plans to extend this solution to more data platforms, streamlining metadata generation for additional teams. New downstream applications, including those in security and data discovery, are also in development.