GDPR Obfuscator Project

For Skills Bootcamp: Software Developer/Coding Skills Graduates - Data Engineering

Context

The purpose of this project is to create a general-purpose tool to process data being ingested to AWS and intercept personally identifiable information (PII). All information stored by Northcoders data projects should be for bulk data analysis only. Consequently, there is a requirement under GDPR to ensure that all data containing information that can be used to identify an individual should be anonymised.

Assumptions and Prerequisites

1. Data is stored in CSV, JSON, or parquet format in S3.
2. Fields containing GDPR-sensitive data are known and will be supplied in advance.
3. Data records will be supplied with a primary key.

High-level desired outcome

You are required to provide an obfuscation tool that can be integrated as a library module into a Python codebase.

The tool will be supplied with the S3 location of a file containing sensitive information, and the names of the affected fields. It will create a new file or byte stream object containing an exact copy of the input file but with the sensitive data replaced with obfuscated strings. The calling procedure will handle saving the output to its destination. It is expected that the tool will be deployed within the AWS account.

Minimum viable product

In the first instance, it is only necessary to be able to process CSV data.

The tool will be invoked by sending a JSON string containing:

- the S3 location of the required CSV file for obfuscation
- the names of the fields that are required to be obfuscated

For example, the input might be:

```
{
    "file_to_obfuscate": "s3://my_ingestion_bucket/new_data/file1.csv",
    "pii_fields": ["name", "email_address"]
}
```

The input CSV file might look like this:

```
student_id,name,course,cohort,graduation_date,email_address
...
1234,'John Smith','Software','2024-03-31','j.smith@email.com'
...
```

The output will be a bytestream representation of a file like this:

```
student_id,name,course,cohort,graduation_date,email_address
...
1234,'***','Software','2024-03-31','***'
...
```

The output format should provide content compatible with the boto3 S3 Put Object

Invocation is likely to be via a tool such as EventBridge, Step Functions, or Airflow. The invocation mechanism is not a required element of this project.

Non-functional requirements

- The tool should be written in Python, be unit tested, PEP-8 compliant, and tested for security vulnerabilities.
- The code should include documentation.

- No credentials are to be recorded in the code.
- The complete size of the module should not exceed the memory limits for Python Lambda dependencies

Performance criteria

- The tool should be able to handle files of up to 1MB with a runtime of less than 1 minute

Possible extensions

The MVP could be extended to allow for other file formats, primarily JSON and Parquet. The output file formats should be the same as the input formats.

Non-binding tech suggestions

It is expected that the code will use the AWS SDK for Python (boto3). Code may be tested with any standard tool such as Pytest, Unittest, or Nose.

The library should be suitable for deployment on a platform within the AWS ecosystem, such as EC2, ECS, or Lambda.

Although the finished product is intended to be used as a library function invoked by other code, you may want to be able to demonstrate its use by invoking it from the command line.

Due date

To be advised, but not later than four weeks from commencement.