Product Demo Description: Northhaven Synthetic Data Generator (SDG)

This document outlines the demonstration flow and key technical outputs of the Northhaven Analytics Synthetic Data Generator (SDG) for investor due diligence. The goal of the demo is to prove the product's ease of use, high-fidelity replication, and enterprise-grade reproducibility.

1. Demo Walkthrough: Architecture and Setup

The demonstration begins by presenting the client's secure environment where the custom-trained ML engine resides.

1.1 Dataset Folder Structure

The demo navigates the standard project directory, illustrating the organized output of the data_manager module. The structure guarantees clarity between input and generated assets:

```
/client_project_id/
├── /01_metadata/
│   ├── schema_v1.0.json
│   └── statistical_aggregates_v1.0.json
├── /02_models/
│   ├── sdg_model_v1.0.git
│   └── logs_training_20251204.txt
└── /03_synthetic_output/
    └── run_2025-12-04_1400/
        ├── client_master_v1.0.csv
        └── account_transactions_v1.0.csv
```

1.2 JSON Metadata and Schema

The demonstration opens the schema_v1.0.json file. This is the blueprint for the generative model, defining not only data types (e.g., categorical, continuous, datetime) but also embedding **financial constraints** (e.g., minimum salary, logical relationships between Product_type and Account_balance). This file serves as the core input for the ML model training phase.

1.3 Model Training Logs

The training log (logs_training_20251204.txt) is displayed to show the technical depth of the process. The log confirms the successful execution of the C-CTGAN/TSM hybrid architecture,

tracking the convergence of the Generator (G) and Discriminator (D) and reporting key fidelity metrics at intervals:

- **Convergence Confirmation:** Shows the adversarial loss function stabilizing.
- **Fidelity Target:** Confirms that the **Synthetic-to-Real Correlation Retention** metric exceeded the target of 0.95, indicating the model is ready for production use.

2. Functionality: Two-Line Generation

The core value proposition of Northhaven is demonstrated through the modular Python library, enabling users to generate massive datasets with minimal code complexity.

The demo executes the following two lines of code, illustrating the power of the encapsulated model module:

```
# 1. Load the dedicated, versioned ML artifact
generator = model.load('sdg_model_v1.0')

# 2. Generate 1,000,000 new rows, conditioning on 'High Income' cohort
dataset = generator.generate(volume=1_000_000, condition={'Monthly_incoming': 'HIGH'})
```

Performance Confirmation: Upon execution, the system confirms the time-to-generation, validating the performance benchmark: **1 million rows generated in approximately 8 minutes**.

3. Output and Validation Proofs

3.1 Synthetic Sample Rows

The output file, client_master_v1.0.csv, is displayed. The rows are demonstrably synthetic yet logically coherent, maintaining the integrity of the original schema (as seen in the provided data):

Client_ID	Age	Country	Monthly_incoming	Credit_score	Product_type	Account_balance
SY0001	48	Poland	35820	812	Investment	245,700
SY0002	25	Germany	5950	710	Savings	31,100
SY0003	55	France	42500	805	Investme	315,500

					nt	
SY0004	39	Spain	2850	620	Checking	8,900

Demonstration Highlight: This sample was generated in approximately 10 seconds (partial generation for speed).

3.2 Statistical Validation Outputs

The fidelity of the synthetic output is the central proof point. The demonstration reviews the automatically generated Validation Report (PDF/HTML), which includes:

- **Distribution Matching (KL Divergence):** Comparison charts confirming that the marginal distribution of key features (e.g., Credit_score, Age, Monthly_incoming) in the synthetic data aligns closely with the real data distribution.
- **Correlation Matrices:** A heat map showing the pairwise correlation matrix of the synthetic dataset. A visual side-by-side comparison with the real correlation matrix confirms preservation, which is vital for risk model inputs.

4. Key Technical Strengths for Enterprise Integration

The demonstration concludes by summarizing the core technical differentiators:

- **Reproducibility (Git Integration):** The git_controller ensures that the **SDG Model Artifact** used for generation is versioned and immutable. This allows the client's Model Validation team to reproduce any synthetic dataset used by the Development team, establishing an auditable chain of custody required by regulators.
- **Performance:** Demonstrated market-leading sub-linear scaling, ensuring data constraints are never a bottleneck for large-scale stress testing (e.g., 1 Billion Rows in approximately 16 hours).
- **Modularity and Integration:** The library structure allows for easy integration into existing enterprise Python notebooks, Azure/AWS ML environments, and internal risk pipelines without requiring major refactoring.
- **High Fidelity:** The TSM component successfully replicates complex temporal dependencies (e.g., monthly Utilization Rate sequences), providing statistically robust data for advanced behavioral models.