VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

DATA STREAMING

Missing Data Imputation using gcimpute and AutoMQ

Multidisciplinary Project - Final Report

Presenter: Group 2
Nguyễn Tấn Nhật - 2152832
Cao Võ Hoài Phúc - 2053332
Lê Hữu Anh Quân - 1952942
Nguyễn Tiến Thành - 2053437

CONTENT

1 INTRODUCTION

MOTIVATION

Reality

Solution

- The growth of IoT devices has led to vast data generation, requiring efficient streaming and management.
- ·A key challenge is handling missing data, which impacts analytics and decision-making.

Copulas

AutoMQ

Source: By AWS Marketplace

GOALS

Deploying a data streaming platform

Using gcimpute to manage incomplete data effectively.

Developing a unified system that integrates these components.

2 PRELIMINARY CONCEPTS

GAUSSIAN COPULA MODEL

The **Gaussian copula** is a **method** for **modeling dependencies** between variables, even when these variables have different marginal distributions (such as categorical and continuous).

Age	Income	MaritalStatus
25	45000	Single
32	NaN	Married
NaN	38000	Divorced
45	55000	NaN
NaN	NaN	Single

GAUSSIAN COPULA MODEL

1 Copula Basics

2 Gaussian Copula Process

GAUSSIAN COPULA MODEL

COPULA BASICS

A copula is a **function** that links **multivariate distributions** to their marginal distributions.

Example: A **copula** models the **dependency** between **rainfall** and **crop yield**, even if their individual distributions are very different.

a gamma distribution

a normal distribution

Source: By Google Image

GAUSSIAN COPULA MODEL

COPULA BASICS

In the example, **copula basics** help explain how gcimpute uses the **Gaussian copula model** to handle dependencies among the **mixed variables (Age, Income, MaritalStatus)** when some values are missing.

Age	Income	MaritalStatus
25	45000	Single
32	NaN	Married
NaN	38000	Divorced
45	55000	NaN
NaN	NaN	Single

GAUSSIAN COPULA MODEL

GAUSSIAN COPULA PROCESS

Step 1: Rank and transform the data:

- Convert Age, Income, and MaritalStatus into ranks, then transform them into uniform and standard normal distributions.

Age (Rank) Values: [25, 32, 45]	Income (Rank) Values: [45000, 38000, 55000]	MaritalStatus (Rank) Values: Single = 1, Married = 2, Divorced = 3
0	0.3333	0
0.3333	0	0.3333
0.6667	0.6667	0.6667

Age (Normal)	Income (Normal)	MaritalStatus (Normal)
-Inf	-0.4308	-Inf
-0.4308	-Inf	-0.4308
0.4308	0.4308	0.4308

GAUSSIAN COPULA MODEL

GAUSSIAN COPULA PROCESS

Step 2: Learn the dependencies using the Gaussian Copula:

- Model the correlation structure between Age, Income, and MaritalStatus in the transformed space.

Step 3: Sample and impute missing values:

- Use the copula to sample from the learned distribution and generate plausible values for the missing Age, Income, and MaritalStatus.

GAUSSIAN COPULA MODEL

GAUSSIAN COPULA PROCESS

Step 4: Reverse the transformations:

- Convert the imputed values back to their original scales (e.g., imputed Income and Age values should be realistic numbers, and MaritalStatus should map back to categorical values like "Single", "Married" or “Divorced”).

Age	Income	MaritalStatus
25	45000	Single
32	42000	Married
29	38000	Divorced
45	55000	Married
30	41000	Single

IMPUTATION MECHANISM

For imputation, gcimpute follows these steps:

Step 1: Model Training: Learning the Dependency Structure

- Objective: Train a Gaussian copula model on the observed (non-missing) data to learn the correlation structure among variables.

Example:

For the Age, Income, and MaritalStatusdata, the Gaussian copula model will learn:

- How Age relates to Income (e.g., weak positive correlation).
- How Age relates to MaritalStatus (e.g., weak positive correlation).
- How Income relates to MaritalStatus (e.g., very weak positive correlation).

IMPUTATION MECHANISM

For imputation, gcimpute follows these steps:

Step 2: Predicting Missing Values: Using the Trained Model

- Objective: Once the copula model is trained, it is used to predict the missing values in the dataset.

Example:

- If Income is missing for an individual with Age = 32 and MaritalStatus = Married, the model uses the learned correlation structure to predict the missing Income value

IMPUTATION MECHANISM

For imputation, gcimpute follows these steps:

Step 3: Confidence Intervals: Reflecting Uncertainty in Imputation

- Objective: Gaussian copula models not only provide an imputed value but also enable the estimation of confidence intervals around those imputed values.

Example:

- For missing Income (e.g., when Income is missing for a person with Age = 32), the imputed value might be 42000, with a confidence interval of [40000, 44000].

3
ALGORITHMS

Method	Description
`fit`	Fit a Gaussian copula from (incomplete) data
`transform`	Impute incomplete data using a Gaussian copula
`fit_transform`	Impute incomplete data using the Gaussian copula `fitted` from itself
`fit_transform_evaluate`	Conduct an evaluation on imputed data returned at each iteration during model fitting
`sample_evaluation`	Sample multiple imputed data using a Gaussian copula
`get_params`	Get parameters of the fitted Gaussian copula
`get_vartypes`	Get the specified variable types used in model fitting
`get_confidence_interval`	Get the confidence intervals for the imputed missing entries

Table 3: Overview of the core methods for both the '`GaussianCopula`' class and the '`LowRankGaussianCopula`' class.

Source by: www.jstatsoft.org

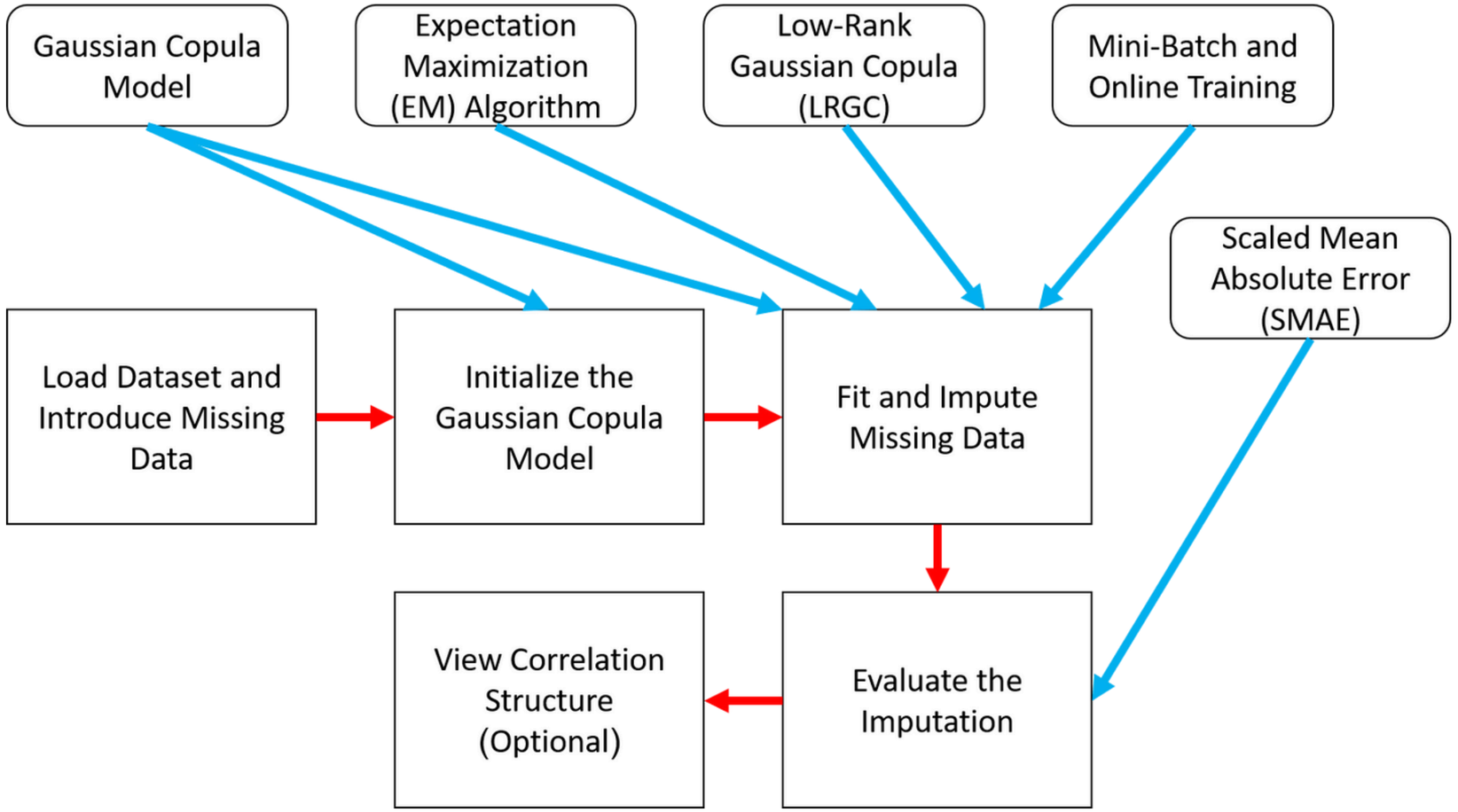
ALGORITHMS

KEY ALGORITHMIC APPROACHES

1. Gaussian Copula Model
2. Expectation Maximization (EM) Algorithm
3. Mini-Batch and Online Training
4. Low-Rank Gaussian Copula (LRGC)
5. Scaled Mean Absolute Error (SMAE)

ALGORITHMS

MISSING DATA IMPUTATION PROCESS



ALGORITHMS

EXPECTATION MAXIMIZATION (EM)

```python
from gcimpute.gaussian_copula import GaussianCopula


model = GaussianCopula(verbose=1)  # Instantiate the model
imputed_data = model.fit_transform(X=data_with_missing)  # Impute missing data
```

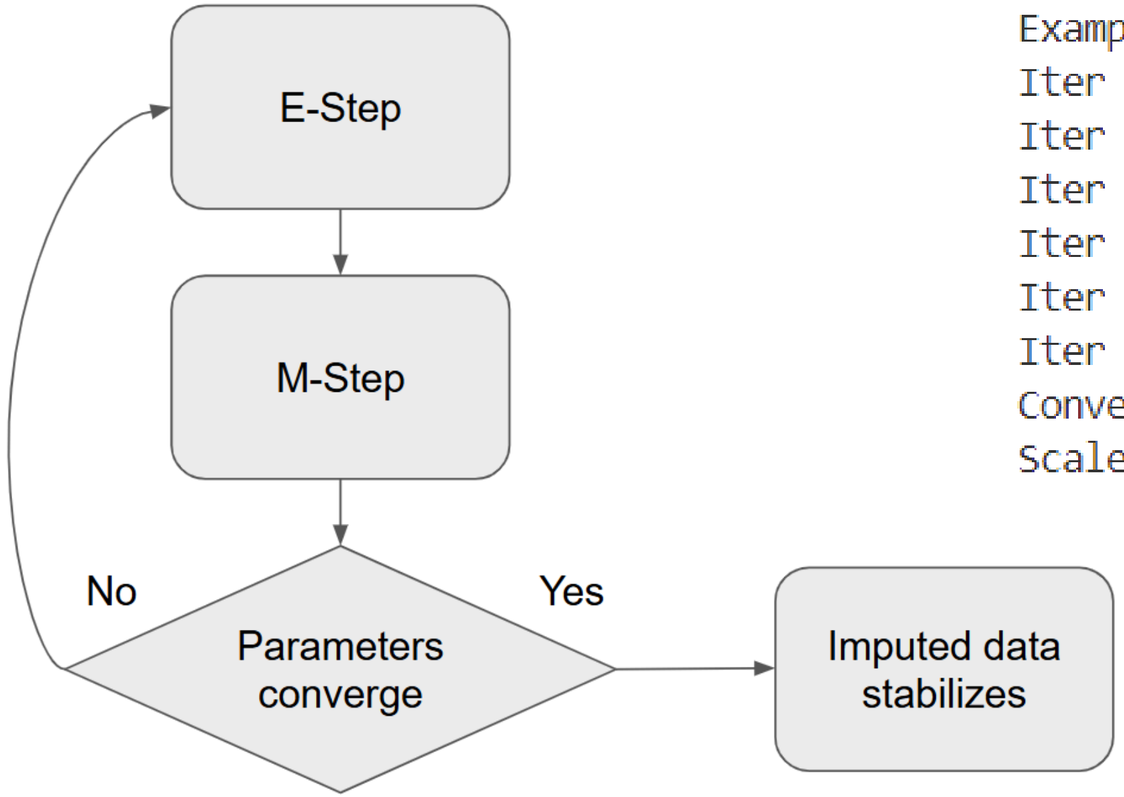
1. Initialization: The copula model initializes with parameters.
2. In each iteration:

- The E-step estimates the missing data values based on current parameters.
- The M-step updates the copula correlation matrix using Maximum Likelihood Estimation (MLE).

3. Convergence: Repeats E and M steps until imputed data stabilizes.

ALGORITHMS

EXPECTATION MAXIMIZATION (EM)



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
Example 1: Basic usage
Iter 1: copula parameter change 0.0867, likelihood -9.6268
Iter 2: copula parameter change 0.0495, likelihood -9.5544
Iter 3: copula parameter change 0.0288, likelihood -9.5091
Iter 4: copula parameter change 0.0176, likelihood -9.4828
Iter 5: copula parameter change 0.0114, likelihood -9.4672
Iter 6: copula parameter change 0.0077, likelihood -9.4576
Convergence achieved at iteration 6
Scaled Mean Absolute Error (SMAE): 0.889
```

ALGORITHMS

SCALED MEAN ABSOLUTE ERROR (SMAE)

Formula:

For a single feature:

$$\text{SMAE} = \frac{\text{MAE}}{\text{Range of Observed Data}} = \frac{\frac{1}{n}\sum_{i \in \text{missing}} |x_{\text{true},i} - x_{\text{imputed},i}|}{\max(x_{\text{obs}}) - \min(x_{\text{obs}})}$$

Where:

- $x_{\text{true},i}$: True value of the i-th data point.
- $x_{\text{imputed},i}$: Imputed value of the i-th data point.
- x_{obs}: Observed (non-missing) values in the feature.
- n: Number of missing values.

Source: By David Andrés - @daansan_ml

ALGORITHMS

SCALED MEAN ABSOLUTE ERROR (SMAE)

```
# Assuming data is the original dataset without missing values
# Calculate Scaled Mean Absolute Error (SMAE)
smae = get_smae(imputed_data, x_true=data, x_obs=data_with_missing)
print(f"Scaled Mean Absolute Error (SMAE): {smae.mean():.3f}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
Scaled Mean Absolute Error (SMAE): 0.889
```

To compare imputation performance across scales, we use scaled mean absolute error (SMAE), which normalizes MAE by that of median imputation.

Gaussian copula imputation outperforms median imputation by 11.1% on average.

ALGORITHMS

MINI-BATCH TRAINING

Aspect	Mini-Batch Training	Standard Training
Data Processing	Processes data in smaller chunks (mini-batches).	Processes the entire dataset at once.
Memory Efficiency	More memory-efficient, suitable for large data.	Requires all data to fit into memory.
Computational Speed	Generally faster, especially for large datasets.	Slower due to processing the entire dataset.
Use Case	Large datasets, real-time learning, deep learning.	Smaller datasets, stable gradient estimation.

ALGORITHMS

MINI-BATCH TRAINING

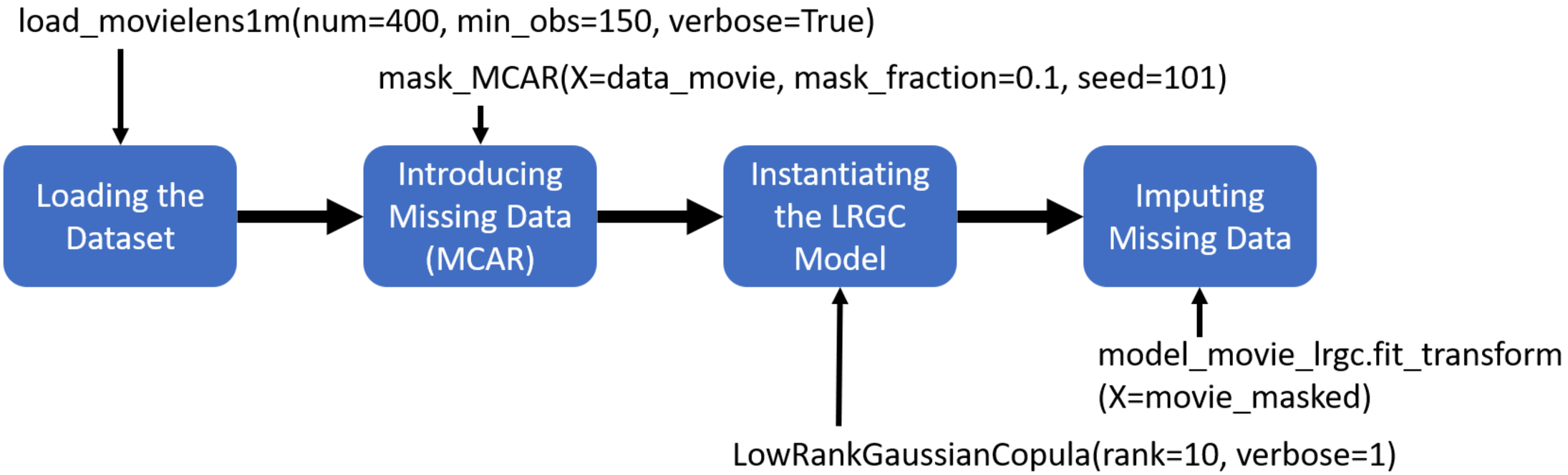
```
Example 2: Accelerating datasets with many samples: mini-batch training
We now run min-batch training with the defaults on the GSS dataset:
Runtime: 2.15 seconds
Imputation error: 0.886
Let us also re-run and record the runtime of the standard training mode:
Runtime: 6.27 seconds
```

- Mini-batch training not only reduces runtime by 66% but also improves the imputation error (from 0.889 to 0.886).

ALGORITHMS

LOW-RANK GAUSSIAN COPULA (LRGC)

- LRGC accelerates the **imputation process** for datasets with a **large number of variables** by approximating the covariance matrix with a low-rank structure.



ALGORITHMS

LOW-RANK GAUSSIAN COPULA (LRGC)

```
Example 3: Accelerating datasets with many variables: low rank structure
The loaded dataset consists of 914 users and 400 movies with 53.3% ratings observed
Interation 1: noise ratio 0.6464, copula parameter change 0.0967, likelihood -411.3549
Interation 2: noise ratio 0.6368, copula parameter change 0.0428, likelihood -410.6035
Interation 3: noise ratio 0.6299, copula parameter change 0.0249, likelihood -410.0521
Interation 4: noise ratio 0.6247, copula parameter change 0.0167, likelihood -409.5867
Interation 5: noise ratio 0.6208, copula parameter change 0.0121, likelihood -409.2098
Interation 6: noise ratio 0.6178, copula parameter change 0.0093, likelihood -408.9108
Convergence achieved at iteration 6
LRGC runtime 0.09 mins.
Iter 1: copula parameter change 0.2972, likelihood -395.2120
Iter 2: copula parameter change 0.1265, likelihood -388.8526
Iter 3: copula parameter change 0.0688, likelihood -383.2187
Iter 4: copula parameter change 0.0428, likelihood -378.8466
Iter 5: copula parameter change 0.0299, likelihood -375.2380
Iter 6: copula parameter change 0.0222, likelihood -372.2628
Iter 7: copula parameter change 0.0174, likelihood -369.7298
Iter 8: copula parameter change 0.0140, likelihood -367.5678
Iter 9: copula parameter change 0.0116, likelihood -365.6854
Iter 10: copula parameter change 0.0098, likelihood -364.0453
Convergence achieved at iteration 10
GC runtime 2.50 mins.
LRGC imputation MAE: 0.581
GC imputation MAE: 0.615
```

- LRGC reduces runtime by 96% compared to the standard Gaussian copula for p=400 variables, with greater speedups for higher dimensions.
- It also lowers imputation error from 0.615 to 0.581, demonstrating small errors (~0.5 stars) on a 1–5 rating scale.

4

CASE STUDY

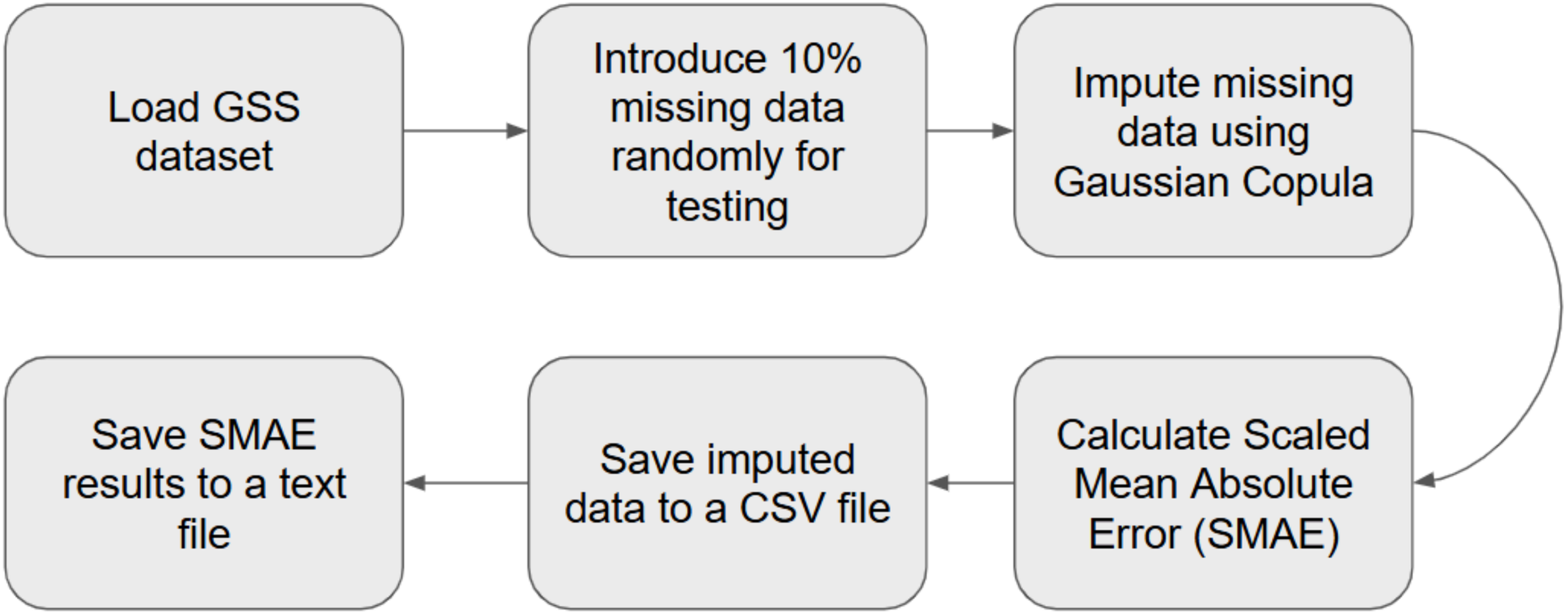
DATASET: GSS_2014_18VAR.CSV
18X2538

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1		DEGREE	PEOCNTC	STRESS	SLPPRBLM	WKSMOO	UNEMP	SATFIN	CLASS_	SATJOB	WEEKSWI	LIFE	HEALTH	HAPPY	RINCOME	INCOM16	SEXFREQ	SOCBAR	AGE	
2	1	3	NA	4	3	2	1	2	3	2	52	1	1	1	12	2	3	NA	53	
3	2	3	NA	2	2	1	1	1	3	2	52	1	1	1	12	3	6	NA	26	
4	3	1	1	NA	NA	NA	NA	3	2	1	13	3	2	3	NA	2	3	2	59	
5	4	3	3	4	3	2	2	1	3	1	52	NA	NA	1	9	2	0	5	56	
6	5	3	NA	NA	NA	NA	2	1	3	NA	0	1	1	1	NA	4	NA	NA	74	
7	6	3	4	NA	NA	NA	NA	2	3	2	52	1	1	2	12	4	NA	4	56	
8	7	1	1	NA	NA	NA	NA	3	NA	NA	52	NA	NA	2	NA	2	NA	4	63	
9	8	4	4	NA	NA	NA	1	2	2	2	52	NA	NA	1	12	3	NA	NA	34	
10	9	0	2	NA	NA	NA	2	3	1	NA	0	3	4	3	NA	3	3	NA	37	
11	10	2	4	NA	NA	NA	1	2	2	1	52	1	1	1	12	1	5	NA	30	
12	11	0	NA	NA	NA	NA	2	2	2	3	0	2	3	2	NA	1	3	NA	43	
13	12	0	NA	NA	NA	NA	NA	2	2	NA	0	3	4	3	NA	2	0	NA	56	
14	13	0	3	NA	NA	NA	NA	3	2	NA	0	2	2	2	NA	2	0	NA	69	
15	14	1	NA	3	1	1	NA	3	1	2	52	2	2	2	11	2	NA	5	40	
16	15	1	NA	NA	NA	NA	2	1	2	NA	52	NA	NA	2	12	2	4	2	25	
17	16	1	NA	4	1	1	2	2	2	1	40	NA	NA	2	11	3	0	5	56	
18	17	1	1	NA	NA	NA	1	3	1	1	52	1	1	2	12	2	NA	NA	51	
19	18	1	2	NA	NA	NA	2	1	2	1	52	2	2	2	12	3	NA	NA	46	
20	19	0	2	1	2	2	1	2	2	2	52	NA	NA	2	11	3	5	NA	51	
21	20	0	3	NA	NA	NA	2	2	3	2	52	2	1	1	6	3	NA	NA	39	
22	21	1	NA	5	4	1	NA	3	3	1	52	1	1	1	11	3	5	NA	30	
23	22	0	NA	NA	NA	NA	2	3	3	2	0	NA	NA	1	NA	2	4	NA	36	
24	23	1	NA	NA	NA	NA	1	2	2	4	6	1	2	2	NA	4	2	NA	42	
25	24	1	NA	NA	NA	NA	1	3	1	NA	0	2	1	2	NA	3	5	NA	38	
26	25	1	3	NA	NA	NA	2	2	2	2	52	2	2	2	12	3	4	NA	38	
27	26	4	NA	5	1	2	1	3	2	3	46	3	1	3	NA	4	0	NA	28	
28	27	1	NA	4	3	2	NA	2	3	2	52	1	2	2	11	3	5	3	35	
29	28	3	3	3	2	3	2	2	3	3	52	NA	NA	2	12	3	4	5	57	
30	29	1	4	NA	NA	NA	2	2	3	1	52	1	2	3	12	2	3	NA	50	
31	30	3	1	3	4	1	1	3	2	3	50	1	1	3	12	3	0	NA	54	
32	31	3	NA	NA	NA	NA	NA	3	3	2	0	3	2	2	NA	4	3	NA	NA	
33	32	3	NA	4	2	2	NA	1	3	2	52	1	2	2	12	1	1	NA	61	
34	33	3	NA	2	2	3	NA	1	3	4	52	2	2	2	NA	3	2	3	54	
35	34	3	4	NA	NA	NA	2	1	2	2	52	1	2	3	NA	3	6	NA	47	
36	35	1	NA	3	4	2	1	2	2	2	0	NA	NA	3	NA	2	5	NA	53	
37	36	1	NA	1	4	2	2	3	1	2	52	2	2	1	11	1	1	NA	37	
38	37	0	3	NA	NA	NA	NA	2	1	1	52	1	2	1	8	3	4	3	38	
39	38	1	NA	3	4	2	1	2	1	1	50	NA	NA	1	12	4	6	4	31	
40	39	0	NA	3	3	2	2	2	3	2	16	2	3	2	NA	2	0	NA	69	
41	40	3	NA	1	4	2	2	1	3	2	52	1	2	2	12	2	5	NA	35	
42	41	3	3	2	3	2	NA	1	3	1	48	1	2	1	NA	3	4	6	61	
43	42	2	NA	NA	NA	NA	NA	3	3	NA	0	2	2	2	NA	2	NA	1	80	
44	43	0	1	NA	NA	NA	NA	2	1	NA	0	2	4	3	NA	2	NA	NA	59	
45	44	1	NA	4	1	1	1	2	1	1	47	NA	NA	2	12	3	6	3	26	
46	45	3	NA	3	2	2	2	2	3	1	52	2	1	2	12	3	4	NA	50	
47	46	1	NA	4	4	2	2	3	1	1	52	2	2	2	12	1	4	NA	43	
48	47	2	5	4	3	3	NA	2	3	2	52	2	1	2	12	3	5	5	30	

GSS_2014_18var

Source: By gcimpute/gcimpute/data/GSS_2014_18var.csv

USING THE GCIMPUTE PACKAGE



TESTING AND RESULTS

```
Scaled Mean Absolute Error (SMAE): 0.889
SMAE values per feature:
AGE: 0.932
DEGREE: 0.868
RINCOME: 0.944
CLASS: 0.872
SATJOB: 0.927
WEEKSWRK: 0.700
HAPPY: 0.963
HEALTH: 0.906
```

smae_results.txt

- High SMAE (Less Accurate): HAPPY (0.963), RINCOME (0.944), AGE (0.932).
- Moderate SMAE: HEALTH (0.906), SATJOB (0.927).
- Low SMAE (High Accuracy): WEEKSWRK (0.700), CLASS (0.872), DEGREE (0.868).

TESTING AND RESULTS

imputed_data.csv

Imputed data is saved to a CSV file.

	A	B	C	D	E	F	G	H
1	AGE	DEGREE	RINCOME	CLASS	SATJOB	WEEKSWR	HAPPY	HEALTH
2	53	3	12	3	3	52	3	4
3	26	3	12	3	3	52	3	4
4	59	1	12	2	4	13	2	3
5	56	3	9	3	4	52	3	4
6	74	3	12	3	4	46	3	4
7	56	3	12	3	3	52	2	4
8	63	1	12	3	4	52	2	3
9	34	4	12	2	3	52	3	4
10	53	0	8	1	3	0	1	1
11	30	2	12	2	4	52	3	4
12	43	0	9	2	2	0	2	2
13	58	0	10	2	3	0	2	1
14	69	0	11	2	4	0	2	3
15	40	1	12	1	3	52	2	3
16	25	1	12	2	3	52	2	3
17	56	1	11	2	4	40	2	3
18	51	1	12	1	4	52	2	4
19	45	1	12	2	4	52	2	3
20	51	0	11	2	3	52	2	3
21	39	0	6	3	3	52	2	4
22	30	1	11	3	4	52	3	4
23	36	0	10	3	3	0	3	3
24	42	1	11	2	1	6	2	3
25	38	1	9	1	3	0	2	4
26	38	1	12	2	3	52	2	3
27	28	4	12	3	2	46	1	4

CHALLENGES

- High SMAE for Certain Features: Features like HAPPY (0.963) and RINCOME (0.944) show low imputation accuracy due to complex dependencies.
- Mixed Data Types: Balancing continuous (AGE, RINCOME) and categorical (CLASS, DEGREE) variables is challenging.
- Outliers and Variability: Outliers in AGE and RINCOME impact accuracy.
- Model Limitations: Gaussian copula struggles with non-linear relationships and extremes.

SOLUTIONS

- Use advanced models like neural networks or hybrid methods for high-SMAE features.
- Normalize continuous features and improve encoding for categorical data.
- Refine dependency analysis to better capture relationships.
- Evaluate with additional metrics like RMSE and conduct sensitivity tests.

5 DATA STREAMING PLATFORM

AUTOMQ

Data streaming Platform

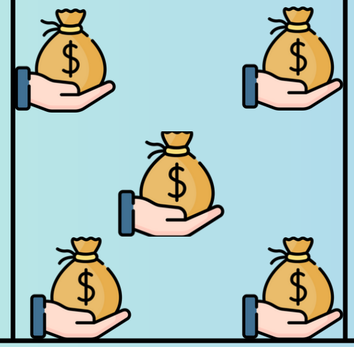
A cloud-first alternative to Kafka by decoupling durability to S3 and EBS

Maintaining 100% compatibility with Kafka

Source: By AWS Marketplace

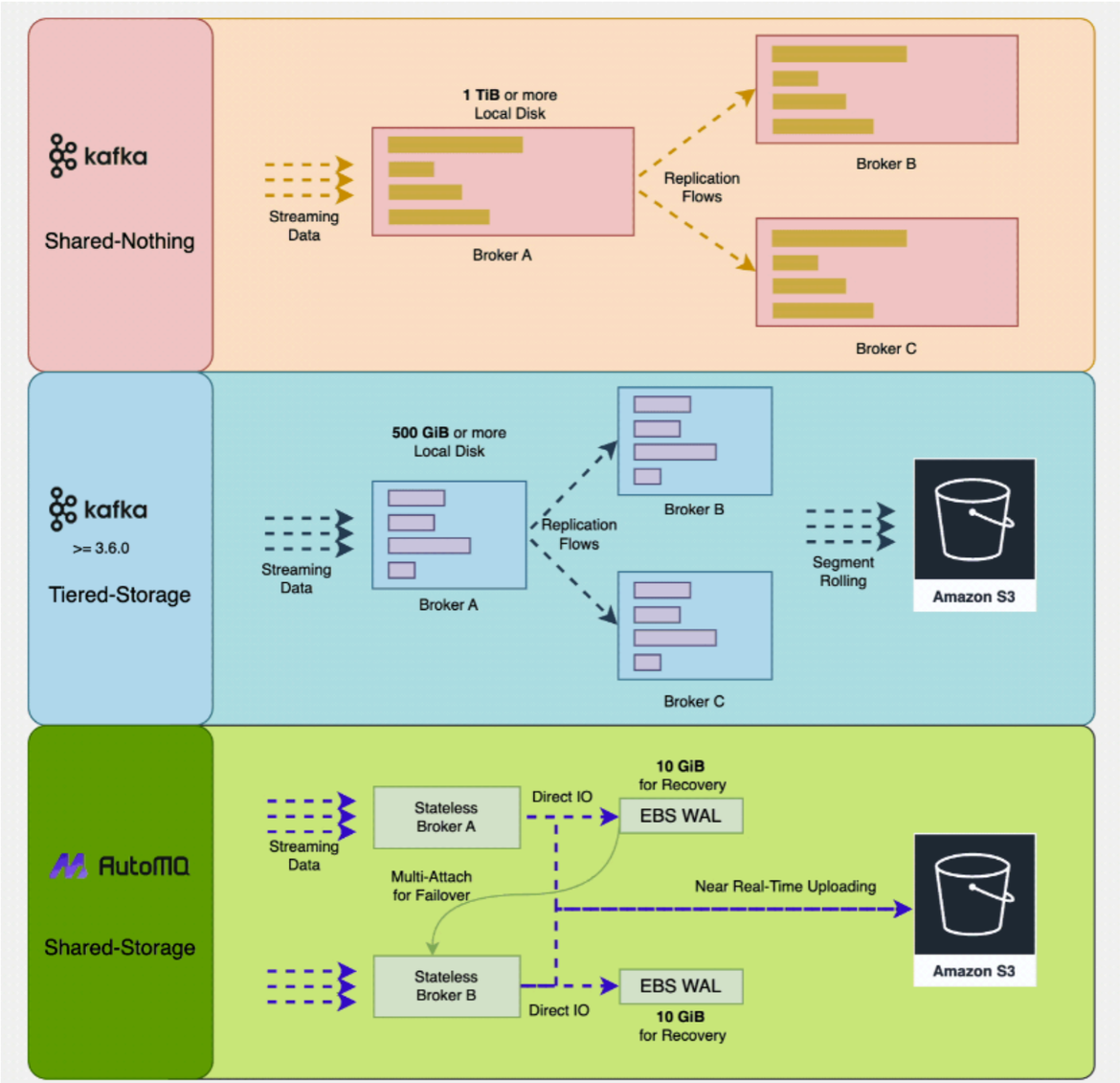
Source: By AutoMQ

AutoMQ Vs kafka

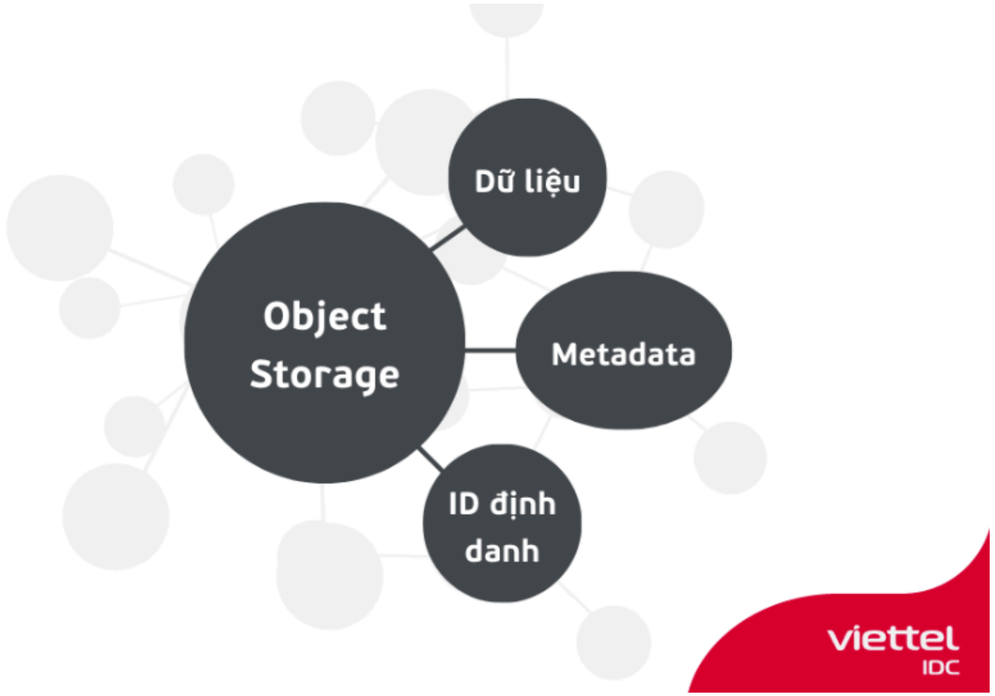
Compare Item	Scale in/out	Cost	Publish Latency During Catch-up Reads	Stateless Broker	Self-Balancing
AutoMQ	In seconds		<10ms	YES	YES
Kafka	In hours even days		>800ms	NO	NO

Source: By AutoMQ

KAFKA ARCHITECTURE EVOLUTION

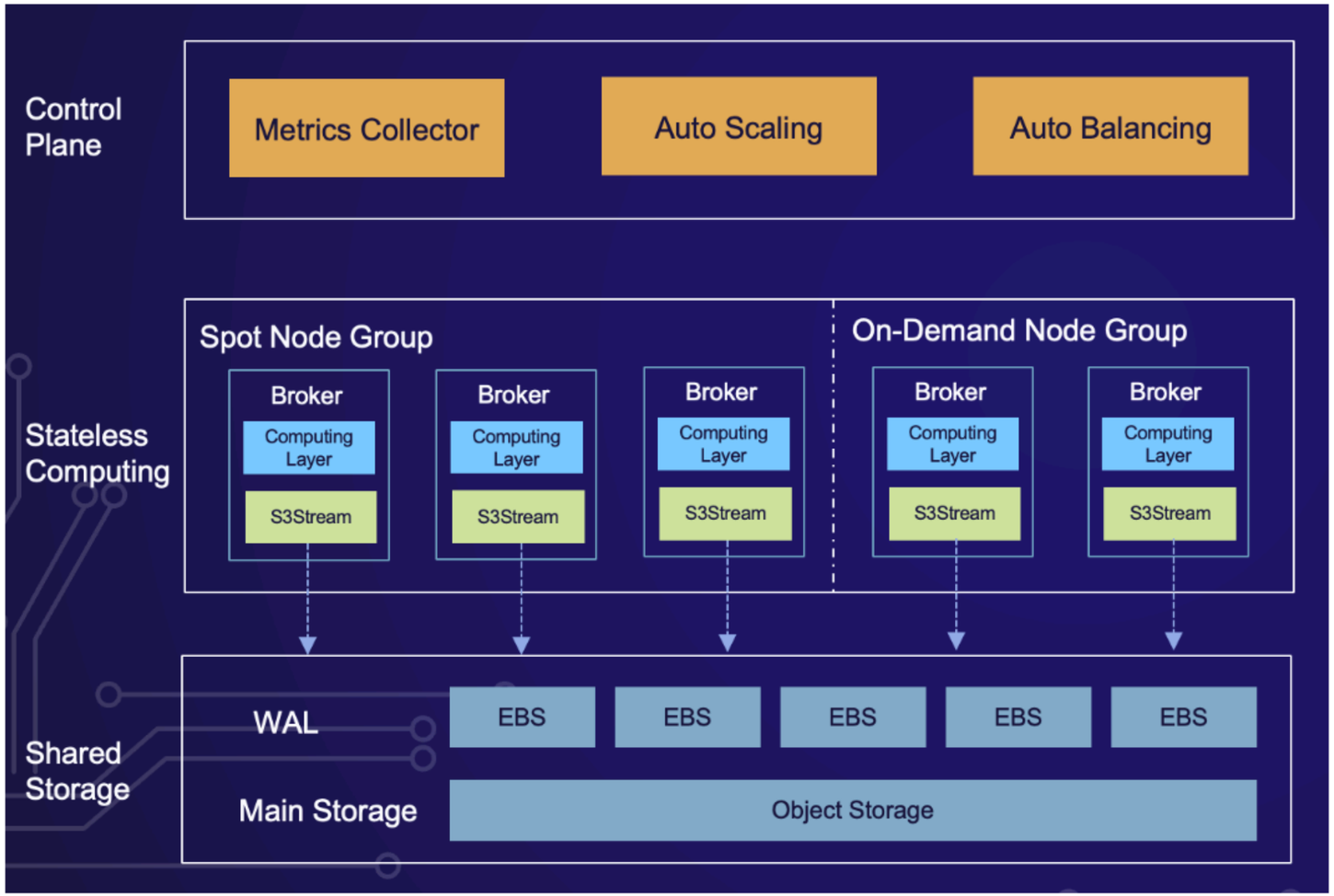


Source: By AutoMQ



Source: Viettel IDC

AUTOMQ SHARED STORAGE ARCHITECTURE OVERVIEW

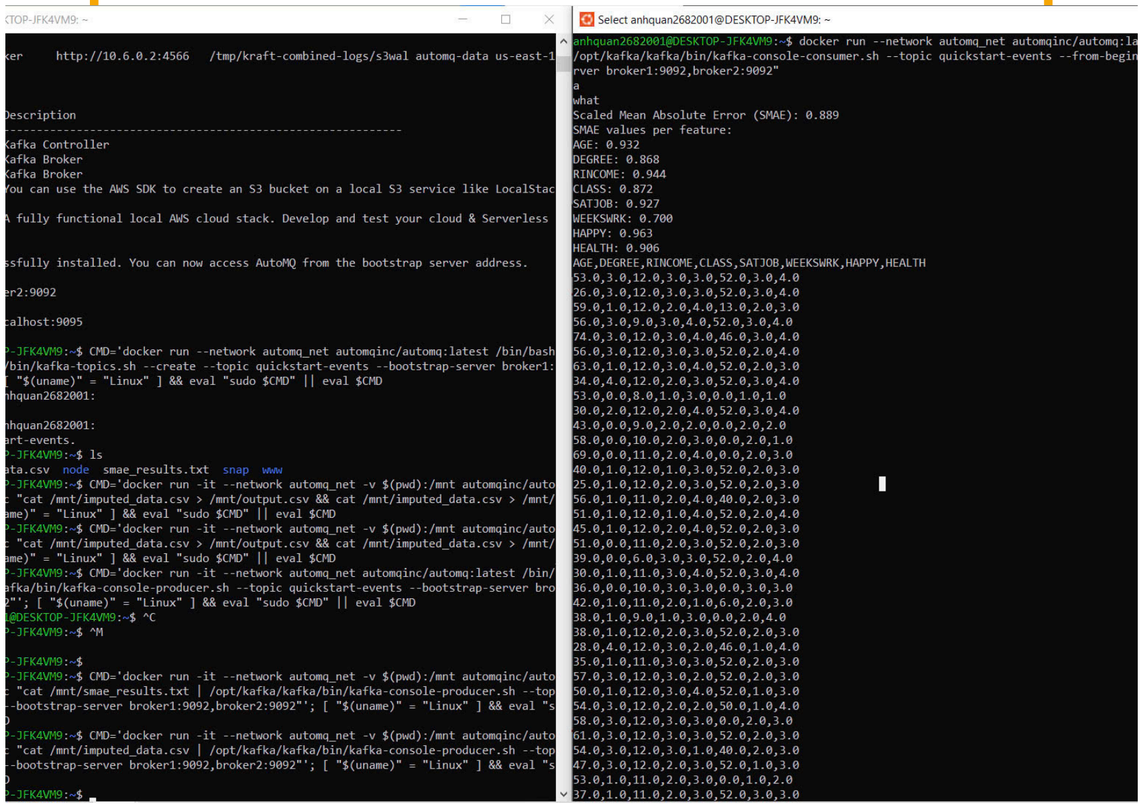


Source: By AutoMQ

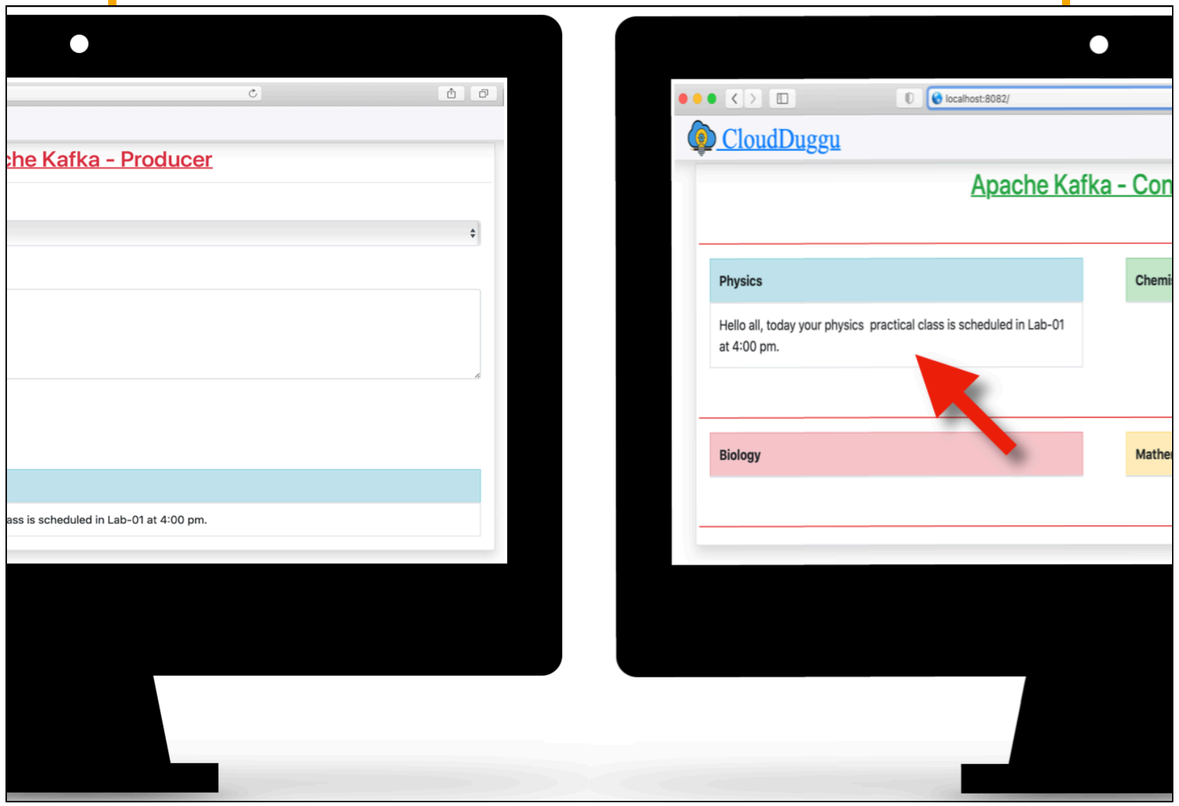
6
EVALUATION

TESTING AND RESULTS

Actual Result



Hoping result



7

CONCLUSION

CONCLUSION

Our team succeeded in combining AutoMQ platform and gcimpute package.

However, the tests provided are only at the basic data transfer level and cannot account for measurement capabilities, real-time updates, or very large data processing.

In the future, if possible, our team will research other functions of AutoMQ platform and optimize the data streaming process.