Statistics 414 From Predictive AI to Generative AI : High Machine Learning Utility for Synthetic Data

Thomas Kwok

Department of Statistics & Data Science, UCLA

March 6, 2025

Fidelity, Utility, and Privacy

Fidelity:

- Directly compare the synthetic dataset with the real dataset
- Measures how well the synthetic data statistically matches the real data

Utility:

- Determined by its effectiveness in facilitating various downstream machine learning tasks
- Contrasting the performance of models on real vs synthetic data, inspecting concrete metrics (e.g. accuracy, mse, model fairness properties)
- Often requires train on synthetic, test on real (TSTR) paradigm

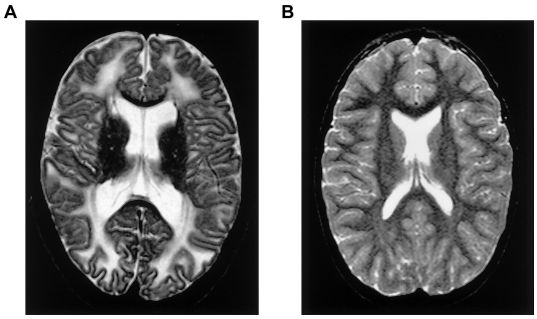
Privacy:

- Determined by the amount of information that it reveals about the real data used to produce it
- Differential privacy is required depending on the use case

Fidelity v.s. Utility: They are not synonymous nor perfectly correlated, fidelity can be reduced while leaving utility unaltered in some scenarios.
Privacy v.s. Fidelity: When fidelity increases, the privacy of synthetic data decreases. Multiple synthetic datasets might need to be generated, each with user specified privacy guarantees.

Why synthetic data?

- Question: What is the purpose of synthetic data?
- Answer: To conduct data analysis on data that
  - are sensitive / contain privacy information (e.g. clients' personal information)
  - insufficient sample size for effective data analysis (e.g. Ribose-5-phosphate isomerase deficiency (RPID) with only three known cases reported over 27 years)
- Literal definition: how well can the synthetic data be used as an alternative / supplement as the original data?



A RPID patient's brain (A) is shown with extensive abnormalities on the cerebral hemispheric white matter, as compared to that of a normal individual (B). (Huck et al. (2004))

Synthetic data use cases

- Two common use cases:

Synthetic data use cases

- Two common use cases:
  1. Replacement: If data contain sensitive information and cannot be used directly: Can using synthetic data alone train a machine learning model that functions as well as using the original data?

Synthetic data use cases

- Two common use cases:
  1. Replacement: If data contain sensitive information and cannot be used directly: Can using synthetic data alone train a machine learning model that functions as well as using the original data?
  2. Appendant: If data is insufficient: Can adding synthetic data into original data facilitate machine learning model in understanding data patterns?

How useful is synthetic data over these use cases?

Replacement of Original Data

	Datasets	Precision			Recall			F1			AUC		
		No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change
Balanced Datasets	AB	-8.82%	-3.98%	5.30%	-0.29%	-7.84%	-7.57%	-4.99%	-5.84%	-0.90%	-4.93%	-5.32%	-0.40%
	AC	-74.06%	-34.72%	151.67%	-72.22%	-16.67%	200.00%	-73.21%	-27.17%	171.88%	-18.72%	-20.58%	-2.28%
	CG	-87.24%	-5.90%	637.21%	-94.59%	-54.23%	746.67%	-92.38%	-38.27%	710.51%	-45.06%	-17.37%	50.41%
	DI	-11.10%	-1.62%	10.66%	19.32%	1.07%	-15.29%	2.15%	-0.27%	-2.37%	-8.01%	-2.64%	5.83%
	FS	1.15%	-2.86%	-3.96%	-16.69%	-4.42%	14.73%	-8.63%	-3.64%	5.46%	-3.67%	-2.37%	1.35%
	HA	-15.13%	0.19%	18.04%	-42.64%	15.33%	101.06%	-31.00%	6.92%	54.96%	-45.82%	15.49%	113.17%
	HE	-35.26%	-43.60%	-12.89%	20.56%	24.74%	3.47%	-17.71%	-24.57%	-8.34%	-13.72%	-30.64%	-19.60%
	HF	-29.73%	-12.40%	24.66%	9.62%	-18.27%	-25.44%	-12.34%	-15.74%	-3.88%	-12.77%	-14.17%	-1.61%
	LI	-12.38%	-55.96%	-49.74%	51.48%	-46.23%	-64.50%	8.77%	-51.95%	-55.82%	-4.08%	-23.48%	-20.22%
	ST	-9.30%	-7.99%	1.44%	7.50%	-2.72%	-9.50%	-1.62%	-5.43%	-3.88%	-15.44%	-12.99%	2.90%
	TI	-34.43%	-21.41%	19.86%	-55.18%	-1.52%	119.73%	-46.84%	-12.50%	64.60%	-32.53%	-13.26%	28.56%
	WQ	-2.04%	-6.05%	-4.09%	-15.63%	-54.09%	-45.59%	-9.40%	-38.51%	-32.13%	-5.77%	-14.78%	-9.56%
	Average	-26.53%	-16.36%	**13.84%**	-15.73%	-13.74%	2.37%	-23.93%	-18.08%	7.69%	-17.54%	-11.84%	6.92%
	Median	1.15%	0.19%	-0.95%	51.48%	24.74%	-17.65%	8.77%	6.92%	-1.70%	-3.67%	15.49%	**19.89%**
	SD	28.02%	18.59%	-7.37%	42.83%	25.74%	**-11.96%**	31.50%	18.07%	**-10.21%**	15.34%	11.97%	-2.92%
Imbalanced Datasets	AD	4.31%	9.01%	4.51%	-61.01%	-68.78%	-19.93%	-44.47%	-52.74%	-14.89%	-9.76%	-5.32%	4.93%
	BL	-9.26%	-7.70%	1.72%	2.73%	37.70%	34.04%	-2.64%	14.25%	17.34%	1.57%	-6.76%	-8.20%
	CR	-75.67%	10.80%	355.46%	-94.05%	-27.10%	1125.60%	-91.21%	-14.59%	871.46%	-22.58%	-4.03%	23.97%
	MC	-57.72%	-47.97%	23.04%	36.99%	26.42%	-7.72%	-30.83%	-22.24%	12.41%	-18.42%	-11.32%	8.71%
	CTR1	15.69%	50.18%	29.81%	-20.29%	-17.06%	4.05%	-4.98%	8.00%	13.65%	-9.68%	-6.83%	3.16%
	CTR2	26.12%	-25.00%	-40.53%	-70.00%	88.93%	529.76%	-52.83%	9.48%	132.09%	-17.58%	-2.27%	18.58%
	CTR3	-92.36%	5.27%	1277.85%	-55.77%	-12.03%	98.90%	-86.04%	-4.96%	580.60%	-22.90%	-3.95%	24.59%
	CTR4	-89.95%	-13.81%	757.33%	-56.86%	-9.06%	110.82%	-81.57%	-11.06%	382.58%	-21.65%	-2.32%	24.67%
	CTR5	38.21%	-36.00%	-53.70%	-26.32%	-5.26%	28.57%	2.83%	-26.65%	-28.67%	43.22%	-30.65%	-51.58%
	CTR6	-91.64%	-74.32%	207.35%	-64.71%	-58.82%	16.67%	-86.13%	-68.06%	130.34%	-28.87%	-25.36%	4.93%
	Average	-33.23%	-12.95%	**30.36%**	-40.93%	-4.51%	**61.66%**	-47.79%	-16.86%	**59.23%**	-10.67%	-9.88%	0.88%
	Best	38.21%	50.18%	8.66%	36.99%	88.93%	**37.91%**	2.83%	14.25%	**11.11%**	43.22%	-2.27%	-31.76%
	SD	53.26%	35.06%	**-11.88%**	39.06%	46.30%	5.21%	37.59%	26.86%	-7.80%	20.83%	9.99%	-8.98%
Overall Average		-29.57%	-14.81%	**20.96%**	-27.19%	-9.54%	**24.23%**	-34.78%	-17.52%	**26.45%**	-14.42%	-10.95%	4.05%
Overall Best		38.21%	50.18%	8.66%	51.48%	88.93%	**24.73%**	8.77%	14.25%	5.04%	43.22%	15.49%	-19.36%
Overall SD		40.48%	26.66%	-9.84%	42.18%	35.89%	-4.42%	35.68%	21.93%	**-10.14%**	17.93%	10.90%	-5.96%

	Datasets	Precision			Recall			F1			AUC		
		No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change	No PRRO	⇒ PRRO	Change
	AB	-1.78%	-0.04%	1.77%	0.13%	-1.10%	-1.23%	-0.89%	-0.54%	0.35%	-1.87%	-1.33%	0.55%
	AC	-19.54%	-2.93%	20.64%	-24.07%	-12.96%	14.63%	-21.77%	-8.00%	17.60%	-7.29%	-0.99%	7.76%

Utility can be good even when fidelity is not!

- Recalling from last lecture, synthetic data can be evaluated on its fidelity and utility.

Utility can be good even when fidelity is not!

- Recalling from last lecture, synthetic data can be evaluated on its fidelity and utility.
- **Fidelity v.s. Utility:** They are not synonymous nor perfectly correlated, fidelity can be reduced while leaving utility unaltered in some scenarios.

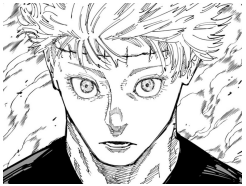
Utility can be good even when fidelity is not!

- Recalling from last lecture, synthetic data can be evaluated on its fidelity and utility.
- **Fidelity v.s. Utility:** They are not synonymous nor perfectly correlated, fidelity can be reduced while leaving utility unaltered in some scenarios.
- In fact, Xu et al. (2024) shows that the distribution of synthetic data needs not be similar to that of the original data to ensure consistent model comparison.

Utility can be good even when fidelity is not!

- Recalling from last lecture, synthetic data can be evaluated on its fidelity and utility.
- **Fidelity v.s. Utility:** They are not synonymous nor perfectly correlated, fidelity can be reduced while leaving utility unaltered in some scenarios.
- In fact, Xu et al. (2024) shows that the distribution of synthetic data needs not be similar to that of the original data to ensure consistent model comparison.
- To be more concrete,

Looks like the original data



Functions like the original data

Utility Theory

Theorem (Consistent Model Comparison)

Let $(\mathcal{F}_1, \mathcal{F}_2)$ be a pair classes of classification models for comparison in the downstream learning task. We say the synthetic distribution $\mathbb{P}_{\tilde{\mathbf{X}},\tilde{Y}}$ preserves the utility of consistent model comparison if

$$\left(R(f^*_{\mathcal{F}_1}) - R(f^*_{\mathcal{F}_2})\right)\left(R(\tilde{f}^*_{\mathcal{F}_1}) - R(\tilde{f}^*_{\mathcal{F}_2})\right) > 0$$

where $\tilde{f}^*_{\mathcal{F}_i} = \arg\min_{f\in\mathcal{F}_i} R(\tilde{f})$ for $i = 1, 2$. [a]

[a] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

- When the synthetic and real distributions are identical, the optimal models will be the same, hence it is straightforward to derive a consistent model comparison based on synthetic data.
- Even if synthetic and real distributions are not identical, consistent model comparison can still be achieved if the difference between the synthetic and real distributions vanishes.

Utility-driven Evaluation

- Common metrics used to evaluate model performance
  1. **Accuracy:** Measures the overall correctness of predictions made by a model
  2. **Precision:** Measures the proportion of true positive predictions among all positive predictions made by the model. Useful when minimizing false positives is important.
  3. **Recall:** Measures the proportion of actual positive cases that are correctly identified by the model. Useful when capturing all positive cases is important.
  4. **F1-score:** Harmonic mean of precision and recall, providing a balance between them.
- Beaulieu et al. (2019) evaluate their synthetic data generation method by measuring the accuracy of classifiers trained on synthetic datasets on the real sensitive medical data used to generate the synthetic data
- Patki et al. (2016) distribute synthetic datasets and real datasets randomly to teams of data scientists, and evaluating whether teams working on real and synthetic datasets would arrive at approximately the same conclusions
- Tao et al. (2021) trained a XGBoost classifier on synthetic data and evaluated its performance on real data for a range of different tabular datasets.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only
  - Combination of both

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only
  - Combination of both
- Evaluate model performance with the appropriate (explainable) metrics, e.g.

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only
  - Combination of both
- Evaluate model performance with the appropriate (explainable) metrics, e.g.
  - How well is the regression / classification

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only
  - Combination of both
- Evaluate model performance with the appropriate (explainable) metrics, e.g.
  - How well is the regression / classification
  - Can the metric truly reflect what we want?

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Synthetic Data Utility Evaluation Framework

- The utility level of synthetic data can be evaluated as follows: [1]

$$U(\tilde{f}, \hat{f}) = |R(\tilde{f}) - R(\hat{f})| \tag{1}$$

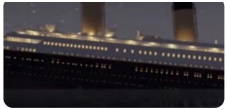
- where $R(\tilde{f})$ and $R(\hat{f})$ represent the generalization performance of $\tilde{f}$ and $\hat{f}$ on unseen real data respectively
- This can be considered as the performance difference between models trained by different datasets.
- Train ML model with different data, such as
  - Original data only
  - Synthetic data only
  - Combination of both
- Evaluate model performance with the appropriate (explainable) metrics, e.g.
  - How well is the regression / classification
  - Can the metric truly reflect what we want?
  - Is the result consistent across different scenarios?

[1] Shirong Xu and Will Wei Sun and Guang Cheng. Utility Theory of Synthetic Data Generation. Pre-print. 2024.

Example: A simple example

Titanic - Machine Learning from Disaster

Start here! Predict survival on the Titanic and get familiar with ML basics



Overview Data Code Models Discussion Leaderboard Rules

Dataset Description

Overview

The data has been split into two groups:

- training set (train.csv)
- test set (test.csv)

The training set should be used to build your machine learning models. For the training set, we provide the outcome (also known as the "ground truth") for each passenger. Your model will be based on "features" like passengers' gender and class. You can also use feature engineering to create new features.

Files
3 files

Size
93.08 kB

Type
csv

License
Subject to Competition Rules

https://www.kaggle.com/c/titanic/data?select=train.csv

Example: A simple example

- This dataset records information of passengers who boarded the Titanic and whether they survived in the accident.
  - 891 passengers with 10 passenger information features (excluding the unique ID column)
- Column features include passenger biological and ticket information.
- The key response variable (named as survived) is a binary variable recording whether the user survived in the accident.
- The survival rate is about 38%.

Example: A realistic example

CTR Prediction - 2022 DIGIX Global AI Challenge

CTR prediction through cross-domain data from ads and news feeds

Data Card | Code (3) | Discussion (0) | Suggestions (0)

About Dataset

Ad recommendation models are usually built based on historical ad impressions, clicks, and other user behavior data. If only data from the ads domain is used, user behavior data will be sparse, and the user behavior types that can be identified will be limited. However, if a user's behavior data in other domains from the same app is explored, the user's interests and behavior characteristics can be better identified. Of course, introducing user behavior data from other apps can also help enrich the data of user behavior characteristics and ad performance.You are expected to enhance ads click-through rate (CTR) prediction accuracy by leveraging ad logs, user profiles, and cross-domain data. With ads as the target domain and news feeds as the source domain, you should build user interest models through impressions, clicks, and other user behavior data obtained from the news feeds domain, thus improving the CTR prediction performance of the ads domain.

Data Description

The provided data includes data from the target domain (such as user behavior logs, user profiles, and ad material information) and that from the source domain (such as user behavior data and basic information about news items).

Usability ⓘ
5.88

License
Unknown

Expected update frequency
Not specified

Tags

Earth and Nature | Tabular | Beginner | Classification

www.kaggle.com/datasets/xiaojiu1414/digix-global-ai-challenge

Example: A realistic example

- This dataset simulates scenarios where advertisers and publishers collaborate to improve customer targeting strategies.
  - Publisher data: 28 columns and 3,227,732 rows. (1.6 GB)
  - Advertiser data: 35 columns and 7,675,517 rows. (1.56 GB)
- The dataset focuses on advertisement clickthrough rate with relevant information on user interactions with ads and news content for 7 days.
- Column features include user demographics, ad metadata, user behavior patterns, and historical CTR data.
- The key response variable (named as label) is a binary variable recording whether the user clicked into the advertisement (clickthrough rate).

Analytical difficulties when working with realistic datasets: Imbalanced class distribution

- In the sample dataset, the clickthrough rate is 1.55% (i.e. only 3 ads clicked for every 200 observations)

Analytical difficulties when working with realistic datasets: Imbalanced class distribution

- In the sample dataset, the clickthrough rate is 1.55% (i.e. only 3 ads clicked for every 200 observations)
- If you are told to make predictions on this data, by predicting on the majority class regardless, you have a 98.45% of getting it correct.

Analytical difficulties when working with realistic datasets: Imbalanced class distribution

- In the sample dataset, the clickthrough rate is 1.55% (i.e. only 3 ads clicked for every 200 observations)
- If you are told to make predictions on this data, by predicting on the majority class regardless, you have a 98.45% of getting it correct.
- **BUT** is that what you truly want? **NO**, because the original goal is to figure out how to get your ads clicked!

Potential solution: Oversampling

- Oversampling gives the machine learning model more exposure to the minority class.

Potential solution: Oversampling

- Oversampling gives the machine learning model more exposure to the minority class.
- But imagine a 50:50 distribution in training data, there is $\sim$ 50% of guessing an ad clicking behavior.

Potential solution: Oversampling

- Oversampling gives the machine learning model more exposure to the minority class.
- But imagine a 50:50 distribution in training data, there is $\sim$ 50% of guessing an ad clicking behavior.
- There will potentially be many false positives. $\Longrightarrow$ \$\$\$\$\$

Potential solution: Oversampling

- Oversampling gives the machine learning model more exposure to the minority class.
- But imagine a 50:50 distribution in training data, there is $\sim$ 50% of guessing an ad clicking behavior.
- There will potentially be many false positives. $\Longrightarrow$ \$\$\$\$\$
- Distorting the distribution may not be a good idea.

Better solution: Data Pruning

- Imagine if you see in advertisement when watching video, will you skip it directly?

Better solution: Data Pruning

- Imagine if you see in advertisement when watching video, will you skip it directly?
- These direct 'skippers' regardless of the ad do not contribute much to data analysis.

Many viewers actually skip the ad part directly, contributing to this 'peak' replay when the video goes back to the original content.

Better solution: Data Pruning

- Imagine if you see in advertisement when watching video, will you skip it directly?
- These direct 'skippers' regardless of the ad do not contribute much to data analysis.

Many viewers actually skip the ad part directly, contributing to this 'peak' replay when the video goes back to the original content.

- By considering only the users who have ever clicked into the advertisement, we can focus on understanding what features make these users click / not click into the ads.

Better solution: Data Pruning

- By considering only the users who have ever shown a clicking behavior, the clickthrough rate now is about one-third.

Better solution: Data Pruning

- By considering only the users who have ever shown a clicking behavior, the clickthrough rate now is about one-third.
- By training a machine learning model with this subgrouped dataset, the model can better grasp the data pattern.

Datasets	Without Pruning		With Pruning	
	Original	Synthetic	Original	Synthetic
D1	0.9777 %	0.0889%	4.1448%	0.4306%
D2	1.6568%	0.0000%	9.9960%	14.9338%
D3	1.2662%	0.0000%	5.0711%	0.7652%
D4	2.1325%	0.0000%	9.1426%	6.0823%
D5	4.0541%	5.4054%	26.3158%	31.5789%
D6	1.6252%	0.1912%	13.0769%	8.4615%
Average	1.9521%	0.9515%	11.2912%	10.3754%

Table: The synthesizer in fact did not generate any positive samples for D2 - D4!

Analytical difficulties when working with realistic datasets: Uninformative features

- Uninformative features are especially common when conducting behavioral analysis

	Decision Tree	Random Forest	XGBoost	GaussianNB	ComplementNB	MultinomialNB
age	0.061087	0.061558	0.073181	0.000144	0.000557	0.000357
gender	0.021990	0.020999	0.066695	0.000262	0.000179	0.000305
residence	0.070864	0.101933	0.071154	0.000096	0.001216	0.000687
city	0.185439	0.143716	0.067934	0.000082	0.002598	0.002927
city_rank	0.028471	0.031712	0.079208	0.000104	0.000316	0.000571
series_dev	0.031414	0.038351	0.076320	0.000205	0.001633	0.000814
series_group	0.030935	0.028860	0.078064	0.000190	0.000540	0.000653
emui_dev	0.063577	0.053903	0.068849	0.000334	0.001000	0.001388
device_name	0.134696	0.127471	0.068220	0.000076	0.006385	0.006859
device_size	0.127485	0.136444	0.067143	0.000054	0.006479	0.005938
net_type	0.048088	0.041858	0.051230	0.000161	0.000028	0.000023
task_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
adv_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
creat_type_cd	0.000027	0.000281	0.004193	0.000015	0.000000	0.000000
adv_prim_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
inter_type_cd	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
slot_id	0.030951	0.036544	0.041901	0.000283	0.000789	0.001271
site_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
spread_app_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
hispace_app_tags	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_second_class	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_score	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
u_refreshTimes	0.087441	0.117963	0.097212	0.004212	0.004397	0.002223
u_feedLifeCycle	0.077536	0.058409	0.088695	0.003832	0.000333	0.000856

Analytical difficulties when working with realistic datasets: Uninformative features

- Uninformative features are especially common when conducting behavioral analysis
- In this table, almost half of the features score 0 in importance score for every classification model.

	Decision Tree	Random Forest	XGBoost	GaussianNB	ComplementNB	MultinomialNB
age	0.061087	0.061558	0.073181	0.000144	0.000557	0.000357
gender	0.021990	0.020999	0.066695	0.000262	0.000179	0.000305
residence	0.070864	0.101933	0.071154	0.000096	0.001216	0.000687
city	0.185439	0.143716	0.067934	0.000082	0.002598	0.002927
city_rank	0.028471	0.031712	0.079208	0.000104	0.000316	0.000571
series_dev	0.031414	0.038351	0.076320	0.000205	0.001633	0.000814
series_group	0.030935	0.028860	0.078064	0.000190	0.000540	0.000653
emui_dev	0.063577	0.053903	0.068849	0.000334	0.001000	0.001388
device_name	0.134696	0.127471	0.068220	0.000076	0.006385	0.006859
device_size	0.127485	0.136444	0.067143	0.000054	0.006479	0.005938
net_type	0.048088	0.041858	0.051230	0.000161	0.000028	0.000023
task_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
adv_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
creat_type_cd	0.000027	0.000281	0.004193	0.000015	0.000000	0.000000
adv_prim_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
inter_type_cd	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
slot_id	0.030951	0.036544	0.041901	0.000283	0.000789	0.001271
site_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
spread_app_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
hispace_app_tags	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_second_class	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_score	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
u_refreshTimes	0.087441	0.117963	0.097212	0.004212	0.004397	0.002223
u_feedLifeCycle	0.077536	0.058409	0.088695	0.003832	0.000333	0.000856

Analytical difficulties when working with realistic datasets: Uninformative features

- Uninformative features are especially common when conducting behavioral analysis
- In this table, almost half of the features score 0 in importance score for every classification model.
- This means they are completely uninformative to the models.

	Decision Tree	Random Forest	XGBoost	GaussianNB	ComplementNB	MultinomialNB
age	0.061087	0.061558	0.073181	0.000144	0.000557	0.000357
gender	0.021990	0.020999	0.066695	0.000262	0.000179	0.000305
residence	0.070864	0.101933	0.071154	0.000096	0.001216	0.000687
city	0.185439	0.143716	0.067934	0.000082	0.002598	0.002927
city_rank	0.028471	0.031712	0.079208	0.000104	0.000316	0.000571
series_dev	0.031414	0.038351	0.076320	0.000205	0.001633	0.000814
series_group	0.030935	0.028860	0.078064	0.000190	0.000540	0.000653
emui_dev	0.063577	0.053903	0.068849	0.000334	0.001000	0.001388
device_name	0.134696	0.127471	0.068220	0.000076	0.006385	0.006859
device_size	0.127485	0.136444	0.067143	0.000054	0.006479	0.005938
net_type	0.048088	0.041858	0.051230	0.000161	0.000028	0.000023
task_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
adv_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
creat_type_cd	0.000027	0.000281	0.004193	0.000015	0.000000	0.000000
adv_prim_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
inter_type_cd	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
slot_id	0.030951	0.036544	0.041901	0.000283	0.000789	0.001271
site_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
spread_app_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
hispace_app_tags	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_second_class	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_score	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
u_refreshTimes	0.087441	0.117963	0.097212	0.004212	0.004397	0.002223
u_feedLifeCycle	0.077536	0.058409	0.088695	0.003832	0.000333	0.000856

Analytical difficulties when working with realistic datasets: Uninformative features

- Uninformative features are especially common when conducting behavioral analysis
- In this table, almost half of the features score 0 in importance score for every classification model.
- This means they are completely uninformative to the models.
- Remove these uninformative features so they will not distract the model from understanding the useful patterns.

	Decision Tree	Random Forest	XGBoost	GaussianNB	ComplementNB	MultinomialNB
age	0.061087	0.061558	0.073181	0.000144	0.000557	0.000357
gender	0.021990	0.020999	0.066695	0.000262	0.000179	0.000305
residence	0.070864	0.101933	0.071154	0.000096	0.001216	0.000687
city	0.185439	0.143716	0.067934	0.000082	0.002598	0.002927
city_rank	0.028471	0.031712	0.079208	0.000104	0.000316	0.000571
series_dev	0.031414	0.038351	0.076320	0.000205	0.001633	0.000814
series_group	0.030935	0.028860	0.078064	0.000190	0.000540	0.000653
emui_dev	0.063577	0.053903	0.068849	0.000334	0.001000	0.001388
device_name	0.134696	0.127471	0.068220	0.000076	0.006385	0.006859
device_size	0.127485	0.136444	0.067143	0.000054	0.006479	0.005938
net_type	0.048088	0.041858	0.051230	0.000161	0.000028	0.000023
task_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
adv_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
creat_type_cd	0.000027	0.000281	0.004193	0.000015	0.000000	0.000000
adv_prim_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
inter_type_cd	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
slot_id	0.030951	0.036544	0.041901	0.000283	0.000789	0.001271
site_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
spread_app_id	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
hispace_app_tags	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_second_class	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
app_score	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
u_refreshTimes	0.087441	0.117963	0.097212	0.004212	0.004397	0.002223
u_feedLifeCycle	0.077536	0.058409	0.088695	0.003832	0.000333	0.000856

Data synthesis

- To preserve utility level during data synthesis, it is important to choose the correct synthesizer.
- LLM-based synthesizer generally grasp data pattern better so that more data utility is preserved.

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

How to Implement LLM-based synthesizer [2]

- `pip install realtabformer`
- `from realtabformer import REaLTabFormer`
- Example in the link: `https://github.com/worldbank/REaLTabFormer`

[2] Solatorio, Aivin V. and Dupriez, Olivier. REaLTabFormer: Generating Realistic Relational and Tabular Data using Transformers. Pre-print. 2023.

Basic Framework: Single Table

- The REaLTabFormer (Realistic Relational and Tabular Data using Transformers) understands and generates the table with the GPT-2 framework.

Basic Framework: Single Table

- The REaLTabFormer (Realistic Relational and Tabular Data using Transformers) understands and generates the table with the GPT-2 framework.
- But as the GPT-2 was trained on a large vocabulary where most of the tokens are not needed for generating the desired tabular data, a fixed-set vocabulary is used instead to improve the training and sampling efficiency.

Basic Framework: Single Table

- The REaLTabFormer (Realistic Relational and Tabular Data using Transformers) understands and generates the table with the GPT-2 framework.
- But as the GPT-2 was trained on a large vocabulary where most of the tokens are not needed for generating the desired tabular data, a fixed-set vocabulary is used instead to improve the training and sampling efficiency.
- Unlike most tabular synthesizers, the REaLTabFormer performs minimal transformation of the raw data by adopting a full text-based strategy in handling numerical values.

Basic Framework: Single Table

- The REaLTabFormer (Realistic Relational and Tabular Data using Transformers) understands and generates the table with the GPT-2 framework.
- But as the GPT-2 was trained on a large vocabulary where most of the tokens are not needed for generating the desired tabular data, a fixed-set vocabulary is used instead to improve the training and sampling efficiency.
- Unlike most tabular synthesizers, the REaLTabFormer performs minimal transformation of the raw data by adopting a full text-based strategy in handling numerical values.
- After tokenizing every value in the dataset, the model is trained on an autoregressive task wherein the target data corresponds to the right-shifted tokens of the input data.

Basic Framework: Multi Tables

- While REaLTabFormer is able to generate tabular data just as many other synthesizers, its ability to model and generate multiple relational data makes it unique.

Basic Framework: Multi Tables

- While REaLTabFormer is able to generate tabular data just as many other synthesizers, its ability to model and generate multiple relational data makes it unique.
- In fact, this feature is mostly wanted in practical use cases as most data we possess in real life might be stored in different tables.

Basic Framework: Multi Tables

- While REaLTabFormer is able to generate tabular data just as many other synthesizers, its ability to model and generate multiple relational data makes it unique.
- In fact, this feature is mostly wanted in practical use cases as most data we possess in real life might be stored in different tables.
- Assuming we possess the health status summary of every patient in the clinic, the additional *food transaction logbook* of the patients will be highly informative in order to understand the patients more comprehensively.

Basic Framework: Multi Tables

- While REaLTabFormer is able to generate tabular data just as many other synthesizers, its ability to model and generate multiple relational data makes it unique.
- In fact, this feature is mostly wanted in practical use cases as most data we possess in real life might be stored in different tables.
- Assuming we possess the health status summary of every patient in the clinic, the additional *food transaction logbook* of the patients will be highly informative in order to understand the patients more comprehensively.
- If these two tables cannot be combined directly, we then need a multi-table synthesizer to model and generate relevant data points.

Basic Framework: Multi Tables

- In a multi-table scenario, there will be a parent table and a child table.

Basic Framework: Multi Tables

- In a multi-table scenario, there will be a parent table and a child table.
- Recall the example, the health status summary would be parent and the food transaction logbook would be child.

Basic Framework: Multi Tables

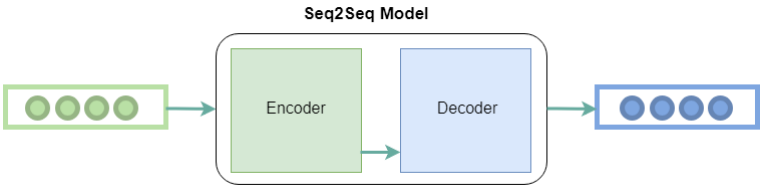
- In a multi-table scenario, there will be a parent table and a child table.
- Recall the example, the health status summary would be parent and the food transaction logbook would be child.
- After modeling the parent table with GPT-2, REaLTabFormer proceeded by modeling the child table with the sequence-to-sequence (Seq2Seq) framework.

Basic Framework: Multi Tables

- In a multi-table scenario, there will be a parent table and a child table.
- Recall the example, the health status summary would be parent and the food transaction logbook would be child.
- After modeling the parent table with GPT-2, REaLTabFormer proceeded by modeling the child table with the sequence-to-sequence (Seq2Seq) framework.
- The encoder network then uses the pre-trained weights of the parent table network to contextualize inputs for generating arbitrary-length data corresponding to observations in a child table via the decoder network.

Sequence-to-sequence Framework

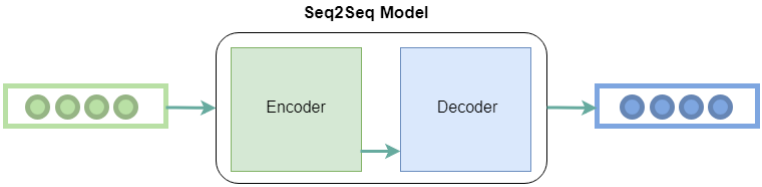
- Models that has sequential data as both input and output.



Encoder and Decoder Stack in seq2seq model

Sequence-to-sequence Framework

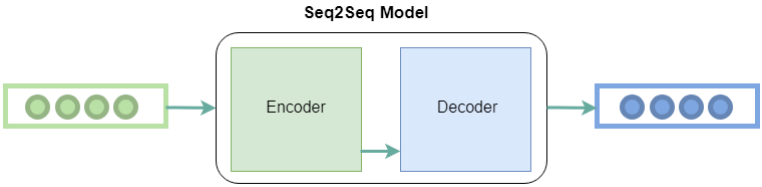
- Models that has sequential data as both input and output.
- Widely used in NLP tasks due to their ability to handle variable-length input and output sequences.



Encoder and Decoder Stack in seq2seq model

Sequence-to-sequence Framework

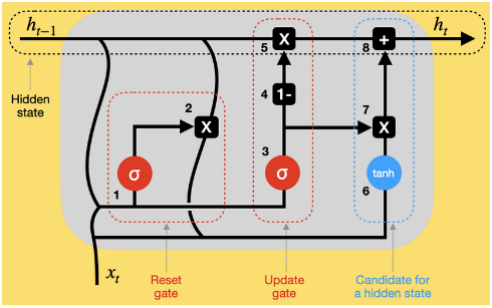
- Models that has sequential data as both input and output.
- Widely used in NLP tasks due to their ability to handle variable-length input and output sequences.
- Contains an Encoder and Decoder to convert input sequences into output sequences.



Encoder and Decoder Stack in seq2seq model

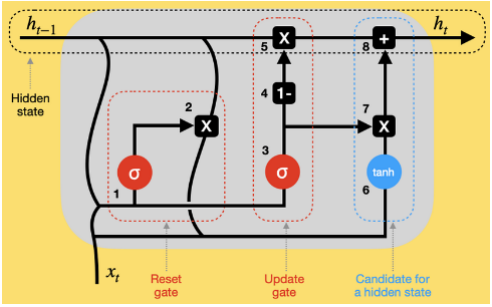
Sequence-to-sequence Framework: Encoder

- Takes the input sequence and encodes it into a fixed-length vector with neural networks / transformers.



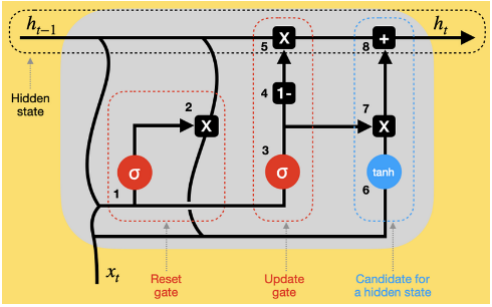
Sequence-to-sequence Framework: Encoder

- Takes the input sequence and encodes it into a fixed-length vector with neural networks / transformers.
- This ultimate encoded representation is known as *context vector*, which has captured the semantic information of the input sequence.



Sequence-to-sequence Framework: Encoder

- Takes the input sequence and encodes it into a fixed-length vector with neural networks / transformers.
- This ultimate encoded representation is known as *context vector*, which has captured the semantic information of the input sequence.
- Common encoders include recurrent neural networks (such as LSTM or GRU).



Sequence-to-sequence Framework: Decoder

- The decoder receives the context vector and the desired target output sequence

Sequence-to-sequence Framework: Decoder

- The decoder receives the context vector and the desired target output sequence
- The decoder then engages in autoregressive generation to produce individual elements step-by-step using the current hidden state, the context vector, and the previous output.

Sequence-to-sequence Framework: Decoder

- The decoder receives the context vector and the desired target output sequence
- The decoder then engages in autoregressive generation to produce individual elements step-by-step using the current hidden state, the context vector, and the previous output.
- The generation is repeated over and over by choosing the output with highest occurrence probability in that step, until the end of the output sequence is reached.

Sequence-to-sequence Framework: The Math Part

Considering the simplest vanilla seq2seq model

$$h_t = \sigma(W^{hx} x_t + W^{hh} h_{t-1})$$
$$y_t = W^{yh} h_t$$

where

x_t, y_t : input and output at time step t

h_t : hidden state at time step t

σ : sigmoid activation function

W^{hx}, W^{yh} : weight matrix for the input and output

Sampling

- The REaLTabFormer builds each observation sequentially by choosing a token from a vocabulary specific to a column in the input.

Sampling

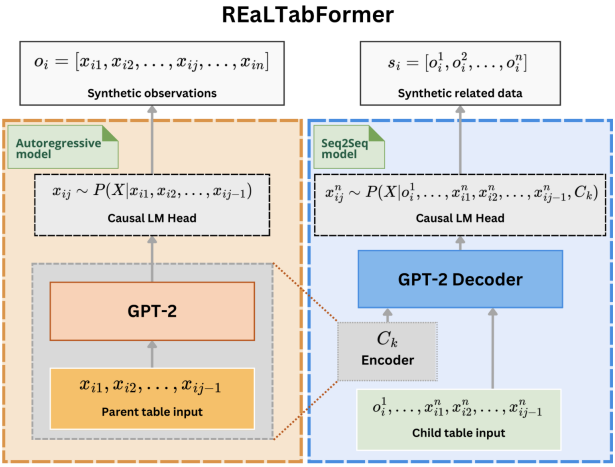
- The REaLTabFormer builds each observation sequentially by choosing a token from a vocabulary specific to a column in the input.
- Every column contains its unique token domain so that invalid tokens for that particular column will not be used for generation in the time-step.

Sampling

- The REaLTabFormer builds each observation sequentially by choosing a token from a vocabulary specific to a column in the input.
- Every column contains its unique token domain so that invalid tokens for that particular column will not be used for generation in the time-step.
- This column-specific strategy allows for efficient sampling with the probability of generating an invalid sample going close to zero.

Overall REaLTabFormer Framework

Now we have the following REaLTabFormer framework:



Weakness of REaLTabFormer

- Unable to deal with missing data.
- Unable to model non-categorical string data (e.g. ID address).
- In multi-table case, every parent observation needs at least one child observation.

Tips: coding bug in the REaLTabFormer package

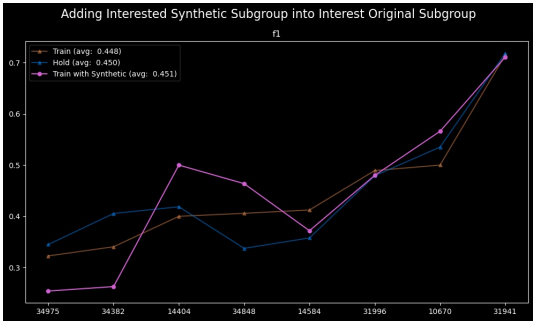
There is a bug in the package, which may cause occasional program crash. Please add the lines on the right into the program yourself.

```
if val_sensitivity < sensitivity_threshold:
    # Just save the model while the
    # validation sensitivity is still within
    # the accepted range.
    # This way we can load the acceptable
    # model back when the threshold is breached.
    trainer.save_model(bdm_path.as_posix())
    trainer.state.save_to_json((bdm_path / "trainer_state.json").as_posix())

elif not_best_val_sensitivity > (val_sensitivity - sensitivity_threshold):
    print("Saving not-best model...")
    trainer.save_model(not_bdm_path.as_posix())
    trainer.state.save_to_json(
        (not_bdm_path / "trainer_state.json").as_posix()
    )
    not_best_val_sensitivity = val_sensitivity - sensitivity_threshold

_delta_mean_sensitivity_value = abs(
    mean_sensitivity_value - val_sensitivity
)

if _delta_mean_sensitivity_value < best_mean_sensitivity_value:
    best_mean_sensitivity_value = _delta_mean_sensitivity_value
    trainer.save_model(mean_closest_bdm_path.as_posix())
    trainer.state.save_to_json(
        (mean_closest_bdm_path / "trainer_state.json").as_posix()
    )


print(
    f"Critic round: {p_epoch + n_critic}, \
        sensitivity_threshold: {sensitivity_threshold}, \
            val_sensitivity: {val_sensitivity}, \
                val_sensitivities: {val_sensitivities}"
)
```

```
if val_sensitivity < sensitivity_threshold:
    # Just save the model while the
    # validation sensitivity is still within
    # the accepted range.
    # This way we can load the acceptable
    # model back when the threshold is breached.
    trainer.save_model(bdm_path.as_posix())
    trainer.state.save_to_json((bdm_path / "trainer_state.json").as_posix())

elif not_best_val_sensitivity > (val_sensitivity - sensitivity_threshold):
    print("Saving not-best model...")
    trainer.save_model(not_bdm_path.as_posix())
    trainer.state.save_to_json(
        (not_bdm_path / "trainer_state.json").as_posix()
    )
    not_best_val_sensitivity = val_sensitivity - sensitivity_threshold

_delta_mean_sensitivity_value = abs(
    mean_sensitivity_value - val_sensitivity
)

if _delta_mean_sensitivity_value < best_mean_sensitivity_value:
    best_mean_sensitivity_value = _delta_mean_sensitivity_value
    trainer.save_model(mean_closest_bdm_path.as_posix())
    trainer.state.save_to_json(
        (mean_closest_bdm_path / "trainer_state.json").as_posix()
    )
    if not any(os.listdir(not_bdm_path.as_posix())):
        trainer.save_model(not_bdm_path.as_posix())
        trainer.state.save_to_json(
            (not_bdm_path / "trainer_state.json").as_posix()
        )

print(
    f"Critic round: {p_epoch + n_critic}, \
        sensitivity_threshold: {sensitivity_threshold}, \
            val_sensitivity: {val_sensitivity}, \
                val_sensitivities: {val_sensitivities}"
)
```

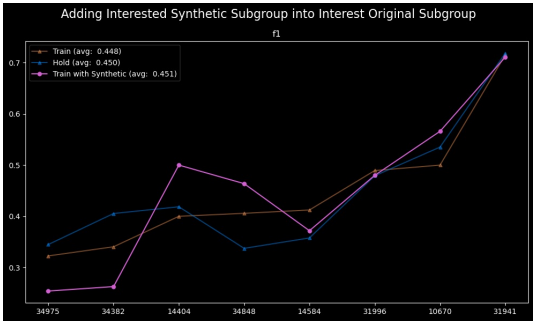
ML performance improved using synthetic data.

- Unfortunately, most synthesizers are not quite ready to produce data that can completely substitute original dataset for data analytics.



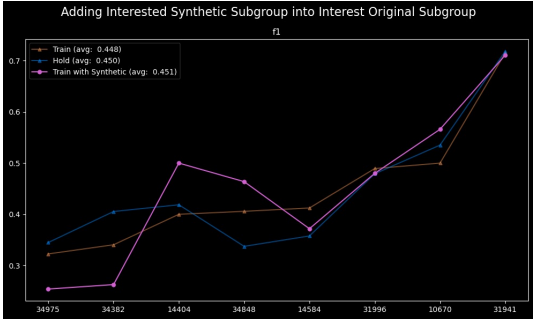
ML performance improved using synthetic data.

- Unfortunately, most synthesizers are not quite ready to produce data that can completely substitute original dataset for data analytics.
- However, can they generate synthetic data good enough for supplement of the original data?



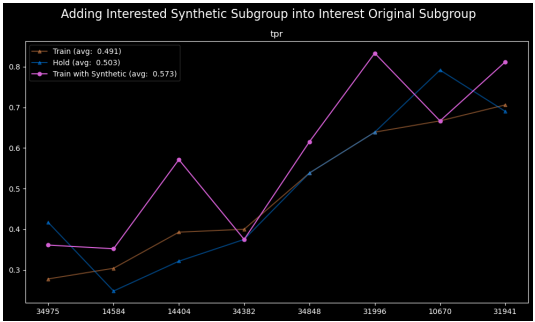
ML performance improved using synthetic data.

- Unfortunately, most synthesizers are not quite ready to produce data that can completely substitute original dataset for data analytics.
- However, can they generate synthetic data good enough for supplement of the original data?
- Data shows that the addition of synthetic data can help improving ML performance.

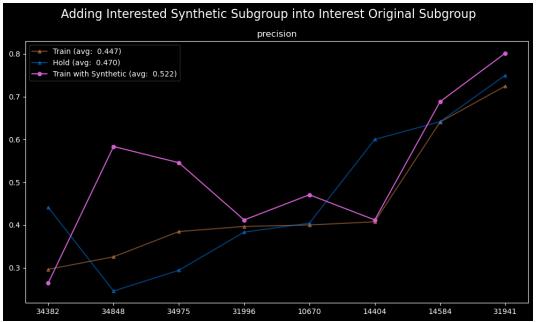


Precision-recall trade-off in adding synthetic data

- The next question goes to how to make use of synthetic data to improve utility.



By adding only the positive synthetic observations, the true positive rate increases because the model is encouraged to make positive predictions (at the expense of more false negatives).



By adding all synthetic observations, the label distribution is preserved so the model can make more precise prediction (at the expense of making less positive predictions).

Precision-recall tradeoff in adding synthetic data

- Some may argue the usage of F1-score should provide optimal balance between recall and precision, but should we seek balance between these two?

Precision-recall tradeoff in adding synthetic data

- Some may argue the usage of F1-score should provide optimal balance between recall and precision, but should we seek balance between these two?
- Example 1: in clickthrough rate case, each false positive prediction means advertisement and capital invested for nothing, so that false positive is more expensive (harmful) than false negative.

Precision-recall tradeoff in adding synthetic data

- Some may argue the usage of F1-score should provide optimal balance between recall and precision, but should we seek balance between these two?
- Example 1: in clickthrough rate case, each false positive prediction means advertisement and capital invested for nothing, so that false positive is more expensive (harmful) than false negative.
- Example 2: in epidemic control case, each false negative prediction means an undiscovered patient to spread the epidemic, so that false negative is more harmful here.

Precision-recall tradeoff in adding synthetic data

- Some may argue the usage of F1-score should provide optimal balance between recall and precision, but should we seek balance between these two?
- Example 1: in clickthrough rate case, each false positive prediction means advertisement and capital invested for nothing, so that false positive is more expensive (harmful) than false negative.
- Example 2: in epidemic control case, each false negative prediction means an undiscovered patient to spread the epidemic, so that false negative is more harmful here.
- The key idea is to identify whether false positive / negative is more harmful to the given situation to decide preference towards recall / precision.

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.
- Regardless, it is important to design an appropriate downstream task to maximize the value of your work.

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.
- Regardless, it is important to design an appropriate downstream task to maximize the value of your work.
- Motivating example:

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.
- Regardless, it is important to design an appropriate downstream task to maximize the value of your work.
- Motivating example:
  - Assuming a task to predict stock movement (a very common ML task), the common set up is a regression model

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.
- Regardless, it is important to design an appropriate downstream task to maximize the value of your work.
- Motivating example:
  - Assuming a task to predict stock movement (a very common ML task), the common set up is a regression model
  - But thinking of this problem deeper, the value change itself matters less to investors' decision, as you will buy (sell) the stock regardless of a 5% or 10% predicted growth (drop).

Design the right downstream task!

- After synthesizing the data, the final step is to train a machine learning model for prediction.
- This course focuses on marketing data, hence a common downstream task is to classify clickthrough rate.
- Regardless, it is important to design an appropriate downstream task to maximize the value of your work.
- Motivating example:
  - Assuming a task to predict stock movement (a very common ML task), the common set up is a regression model
  - But thinking of this problem deeper, the value change itself matters less to investors' decision, as you will buy (sell) the stock regardless of a 5% or 10% predicted growth (drop).
  - By converting this regression problem into a classification task between buy or not buy, can the model help us make better judgment?

Choose the right model!

- Multiple works show [3] **tree-based** models tabular data better.

[3] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

[4] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

Choose the right model!

- Multiple works show [3] **tree-based** models tabular data better.
- The outperformance of tree-based models over neural network in handling tabular data is explained as follows [4]:

[3] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

[4] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

Choose the right model!

- Multiple works show [3] **tree-based** models tabular data better.
- The outperformance of tree-based models over neural network in handling tabular data is explained as follows [4]:
  1. Neural networks are biased to overly smooth solutions.

[3] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

[4] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

Choose the right model!

- Multiple works show [3] **tree-based** models tabular data better.
- The outperformance of tree-based models over neural network in handling tabular data is explained as follows [4]:
  1. Neural networks are biased to overly smooth solutions.
  2. Tabular datasets contain many uninformative features, which MLP-like architectures are not robust to.

[3] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

[4] Léo Grinsztajn and Edouard Oyallon and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? Neurips. 2022.

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.
- Neural networks are biased to making either all positive / negative predictions, and hence are not included in the summarization table.

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.
- Neural networks are biased to making either all positive / negative predictions, and hence are not included in the summarization table.
- The comparison between tree-based and Naive Bayes model shows that while the Naive Bayes model provides a significantly higher true positive rate,

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.
- Neural networks are biased to making either all positive / negative predictions, and hence are not included in the summarization table.
- The comparison between tree-based and Naive Bayes model shows that while the Naive Bayes model provides a significantly higher true positive rate,
  - Lower AUC means low confidence in making the prediction (more likely a wild guess);

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.
- Neural networks are biased to making either all positive / negative predictions, and hence are not included in the summarization table.
- The comparison between tree-based and Naive Bayes model shows that while the Naive Bayes model provides a significantly higher true positive rate,
  - Lower AUC means low confidence in making the prediction (more likely a wild guess);
  - Lower precision means more false positives when making all the wild guesses.

Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Tree-based model is recommended for its effectiveness.

- In our sample dataset, because the features are not quite uninformative (even after removing the completely meaningless features), the classification models are unable to make a terrific task.
- Neural networks are biased to making either all positive / negative predictions, and hence are not included in the summarization table.
- The comparison between tree-based and Naive Bayes model shows that while the Naive Bayes model provides a significantly higher true positive rate,
  - Lower AUC means low confidence in making the prediction (more likely a wild guess);
  - Lower precision means more false positives when making all the wild guesses.
- This also goes back to the precision-recall tradeoff.

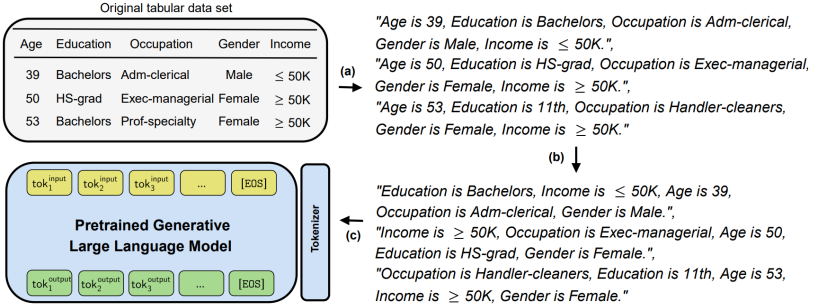
Metrics	Classification Model	Benchmark	Percentage Difference			
			Gaussian Copula	Copula GAN	CTGAN	LLM
AUC	XGBoost	61.93%	-15.48%	-16.06%	-11.34%	-11.02%
	ComplementNB	54.61%	-12.66%	-1.95%	-7.18%	-3.40%
	Random Forest	62.31%	-13.25%	-16.82%	-11.68%	-9.56%
TPR	XGBoost	6.91%	-96.34%	-94.59%	-89.83%	-62.39%
	ComplementNB	51.82%	-23.02%	-13.97%	-14.95%	-12.49%
	Random Forest	8.18%	-99.85%	-95.31%	-95.44%	-60.61%
Precision	XGBoost	42.55%	-94.60%	-95.04%	-91.87%	-78.95%
	ComplementNB	0.93%	-22.90%	-2.32%	-15.18%	-10.17%
	Random Forest	38.81%	-96.97%	-98.57%	-96.36%	-75.24%

Synthesizer advancement

- Although the LLM-based REaLTabFormer shows outperformance in data synthesis over many synthesizers, its GPT-2 backbone implies much improvement space in terms of generative potential.
- The GPT-2 backbone is fine-tuned with LoRA so that the generator can be widely used in local computers.
- Potential work direction:
  - GPT-2 is a bit out-dated, isn't it?
  - This synthesizer cannot handle missing data.

LLM framework advancement

- Existing LLM framework converts each observation into a sentence



- Is there any other conversion framework so that LLM can understand each row observation better?
  - Prompt Engineering? (Current Work)
  - LLM Interpretability? (Full of potential but it's still in a very preliminary stage)
- Should tabular data be considered similar to text data by nature?

Synthetic utility advancement

- Existing factors believed to impact synthetic utility the most:
  - Removal of uninformative features
  - A balanced / less skewed response variable distribution
- Explore on other potential data pre-processing methods to preserve / enhance synthetic data utility.
- Please inspire us more with your projects! :D