Private Machine Learning Track

2022 Workshop of the High-Level Group on the Modernisation of Official Statistics

Julian Templeton, Benjamin Santos and Saeid Molladavoudi (Statistics Canada)
Massimo De Cubellis and Fabrizio De Fausti (Statistics Italy)
Mat Weldon and Owen Daniel (Office of National Statistics, UK)
Matjaz Jug (Statistics Netherlands)

Introduction

Scenario: A National Statistical Office (NSO) wants to offer a Privacy Enhancing Technique (PET)-based remote analytics service, e.g. a predictive model trained on its own internal data. The service could be used by another party (public, private or another NSO) that needs to utilize the NSO's private data while maintaining the appropriate privacy requirements. The consuming party may also use their own confidential data alongside the NSO's data.

Use Case: A university wants to infer whether a student will pass or fail using demographic data owned by NSO alongside their private data. This allows the university to predict how to prepare for the coming years (number of classrooms to prepare, number of professors, …). The university has no access to the sensitive socio-demographic data. Only by using PETs can the university have access to this analytical output.

Goal: Build a private machine learning (ML) model using the linked data, or a synthetic representation, to analyze the impact of PETs when used to protect the machine learning model from various attacks and the training data from being identified/reconstructed.

Note: This use case does not follow responsible ML principles and is meant to analyze the impact of differential privacy and various machine learning attacks. The results can translate to similar use cases in which responsible ML principles are followed.

Introduction

Dataset: The university's student performance dataset and the NSO's demographic dataset have a common anonymized key to link the datasets to then train a predictive model with the use of PETs. We assume that such linkage already exists. A small open dataset is used.

Added value per organization: To train a private ML model using data owned by the other party (NSO).

Type of processing and output data: Training of a predictive model in a privacy preserving manner with the aim to provide an inferential model as output.

Trust relationships: There is a partial trust relationship between the two parties (*Honest but Curious* scenario).

Attack models: Attacks on ML models (i.e. membership inference).

Techniques to assure compliance: Attack mitigation methods during ML training and/or to inputs.

Introduction

Identified Threat Model:

- **Assumed data access:**
  The linked dataset for training is secure and cannot be accessed
- **Knowledge of and access to the machine learning model:**
  Partial or black box access with unlimited query access
- **Relevant attacks:**
  Membership inference, model reconstruction, data linkage

Background on Differential Privacy (DP)

Differential Privacy (DP) uses a privacy budget ε to help protect individual samples in a dataset by adding noise to the input data or training process based on the ε value used.

- A higher ε → less privacy, lower reduction to training performance
- A lower ε → more privacy, reduced training performance
- Can be applied to the input data or the training process

Scenarios Investigated

1. DP training vs DP data
2. Investigation of membership inference attacks
3. Custom implementation of a membership inference attack

Each scenario will present initial results which can be expanded upon with more rigorous testing in the future.

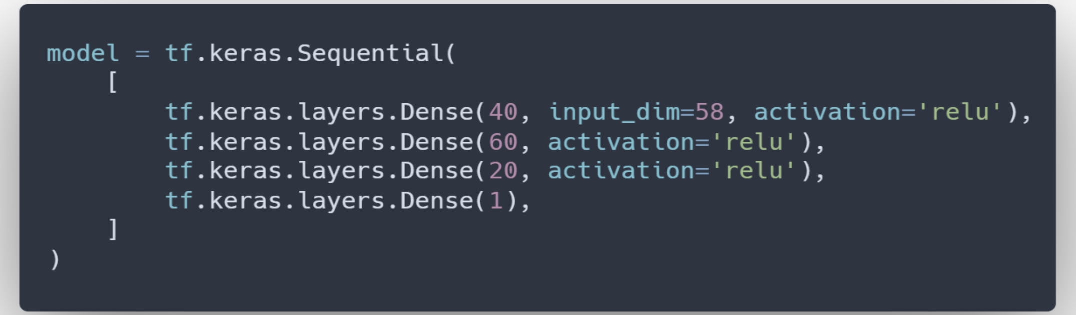
Scenario 1 - DP Training vs DP Data

We consider two approaches to achieving (ε,δ)-Differential Privacy in outputs:

- Creating (ε,δ)-DP Synthetic Data - and then apply a classical (non-private) model.
- Using non-privatised data, but applying an (ε,δ)-DP model.

The Post-processing Theorem of Differential Privacy ensures that the same privacy guarantee will hold - but we anticipate accuracy will vary.

We ran two experiments on the same Feedforward Neural Network (NN): varying where privacy was added.



```
model = tf.keras.Sequential(
    [
        tf.keras.layers.Dense(40, input_dim=58, activation='relu'),
        tf.keras.layers.Dense(60, activation='relu'),
        tf.keras.layers.Dense(20, activation='relu'),
        tf.keras.layers.Dense(1),
    ]
)
```

Scenario 1 - Experiment 1: DP Synthesised Data with non-DP Training

In this experiment we create (ε,δ)-DP synthetic data, and train the NN without DP. We varied $\varepsilon = 1, 10, 1000$.

For synthesis, we used a base implementation of MST (McKenna et al. 2021), which won the 2018 NIST DP Challenge.

This is an ‘unsupervised’ method: it aims to recreate general statistical information (high fidelity), rather than the information relevant to a particular task (high utility).

UNECE

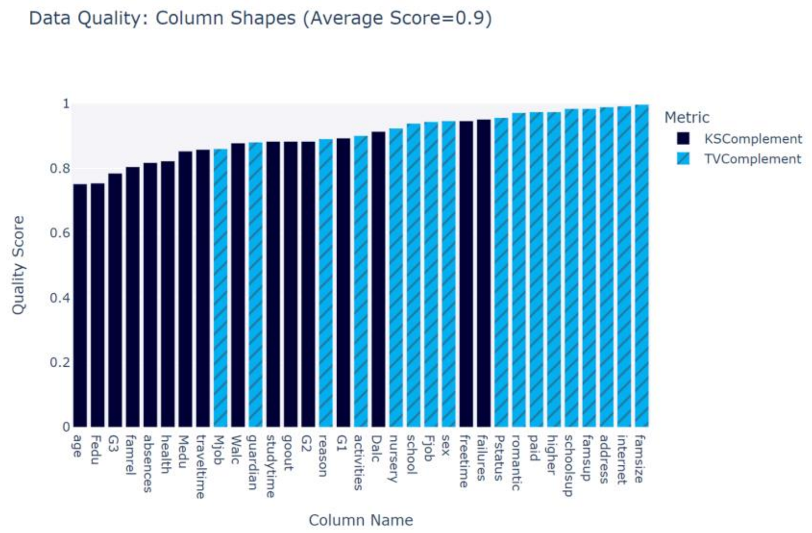
modernstats by HLG - MOS

Scenario 1 - Experiment 1: Synthetic Data Cont.

The dataset is synthesised three times: with 1, 10 and 1,000 epsilon budget, respectively.

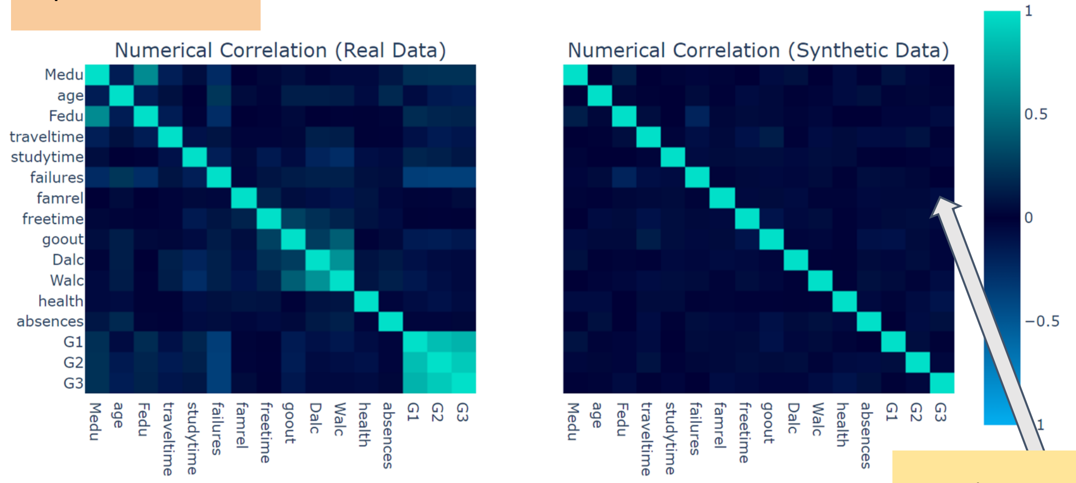
The quality of the synthetic data on univariate margins (epsilon=1, image below) is reasonably high.

However, the synthesiser struggles to accurately estimate correlations with lower epsilon (epsilon 1 and 10 are pictured on right).

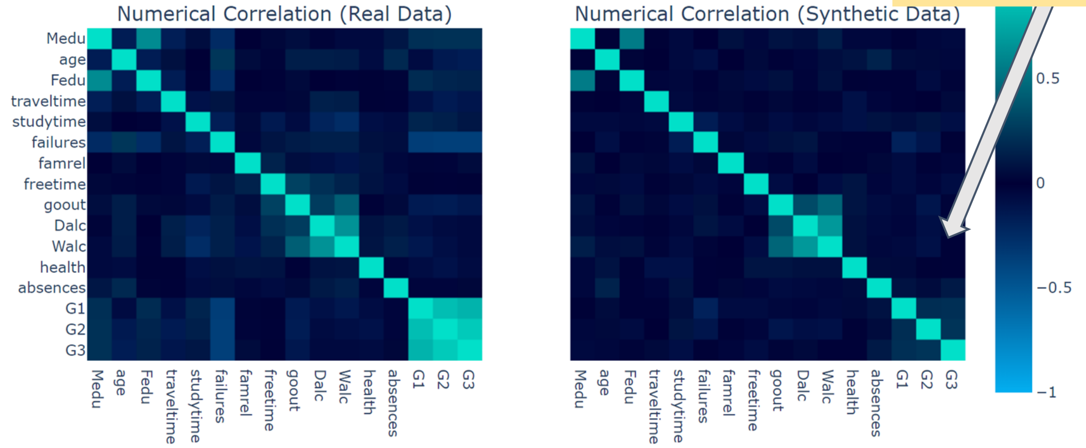


Correlations comparison to real data

Epsilon = 1

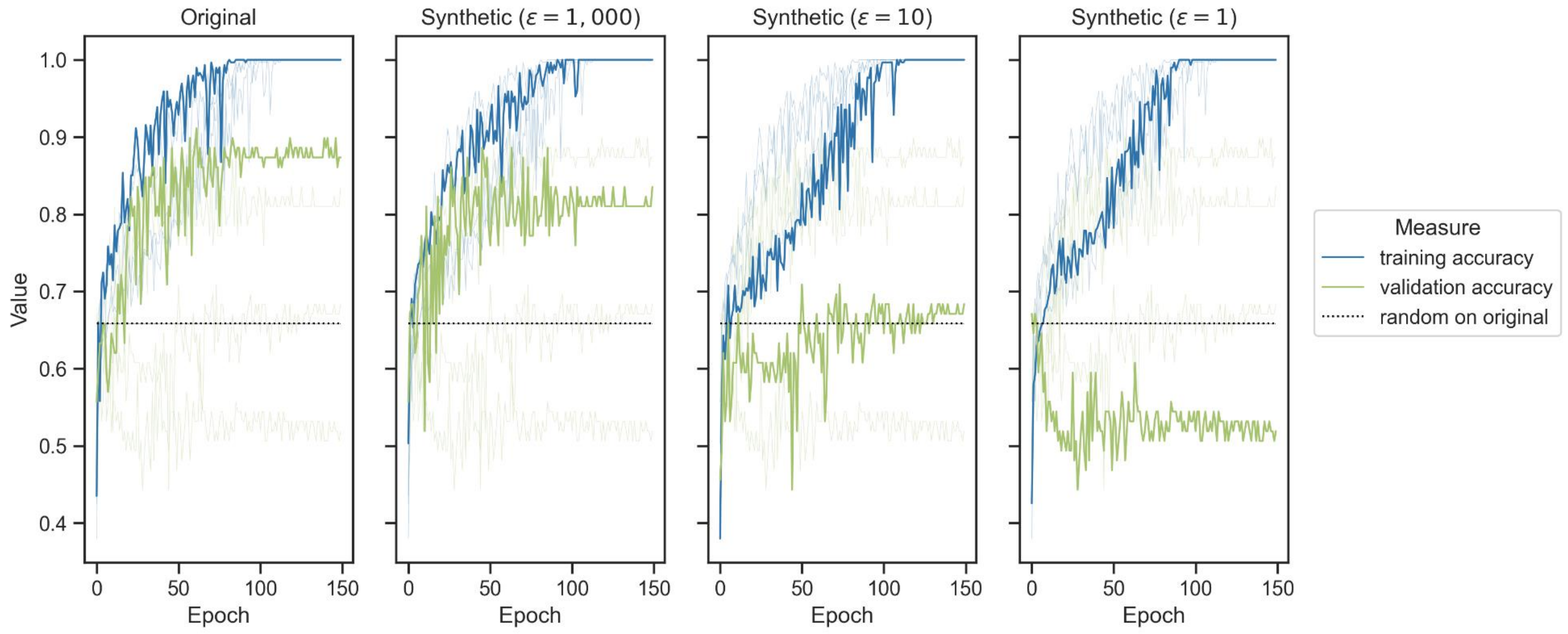


Epsilon = 10



Correlations attenuated on synthetic data, especially with G3 (target variable)

Scenario 1 - Experiment 1: Synthetic Data Results



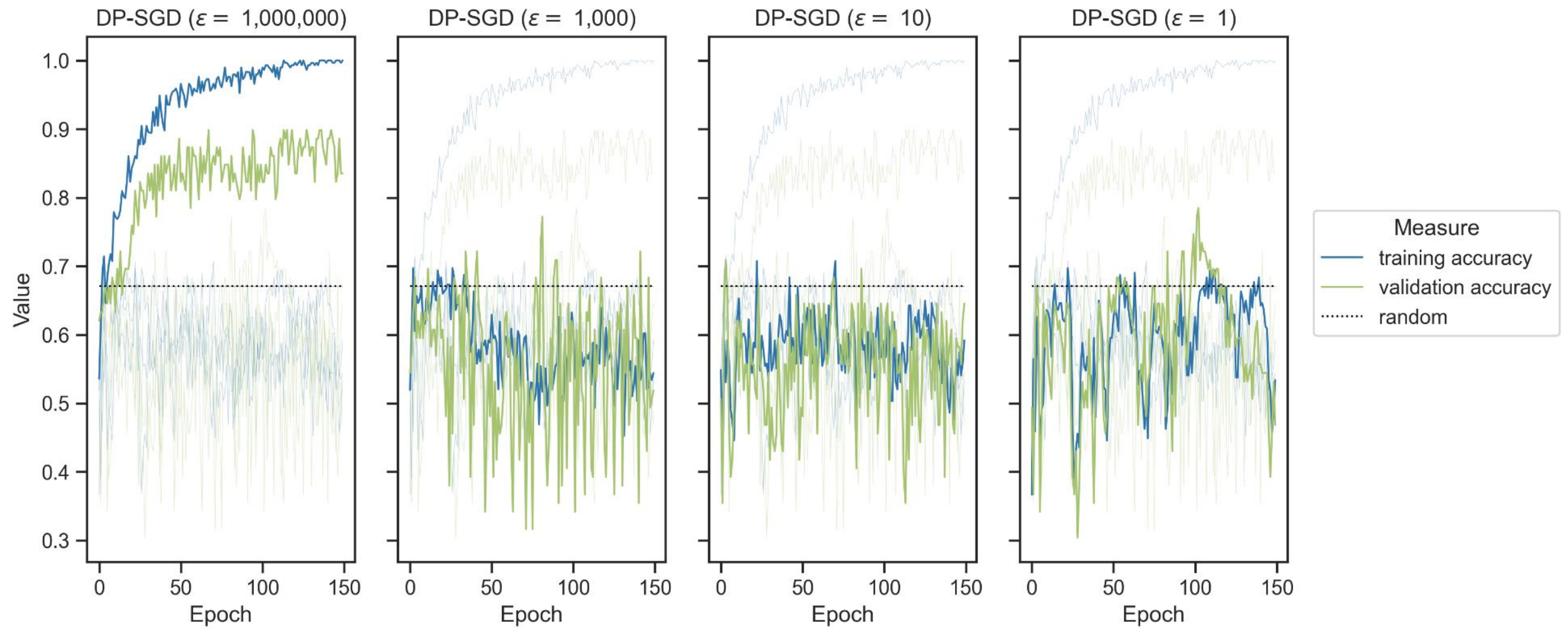
Accuracy	Training	Validation	Original data
Synthetic $\boldsymbol{\varepsilon}$ = 1	1.00	0.52	0.60
Synthetic $\boldsymbol{\varepsilon}$ = 10	1.00	0.68	0.67
Synthetic $\boldsymbol{\varepsilon}$ = 1,000	1.00	0.84	0.77
Original data	1.00	0.87	0.97

The TensorFlow model is fitted to each of the three synthetic datasets and to the original data.

The model fitted to the synthetic data seems to perform reasonably on real data when epsilon is high.

Accuracy scores of fitted models on the real data are similar to accuracy scores on validation partitions of the dataset used to train.

Scenario 1 - Experiment 2: DP-SGD Training



Accuracy	Training	Validation
DP ε = 1	0.54	0.47
DP ε = 10	0.59	0.65
DP ε = 1,000	0.56	0.52
DP ε = 1,000,000	1.00	0.84
Non DP	1.00	0.90

Here we used the original data with the same NN model, but used differentially private stochastic gradient descent: DP-SGD.

Here, the DP method does not perform well when ε is set to a low value. Accuracy improves when epsilon is set to a higher value.

Scenario 2 exhibits similar tests with some optimizations.

Scenario 1 - Conclusions

- For this task, ML on data synthesised with MST was not effective for ε values up to 10.
  - Correlation plots indicate the algorithm failed to retain relationships with the target variable.
- ML on original data with DP-SGD also performs poorly for low values of ε.
- Small size of the data probably made DP ML less effective - further work could test this on larger data sets.
- Supports findings of ONS/Alan Turing Institute (paper in review) that unsupervised synthetic data generators perform weaker than supervised methods.
  - Further work could look at synthesis methods that put higher privacy budget on key relationships of interest.

Scenario 2 - Investigation of Membership Inference Attacks

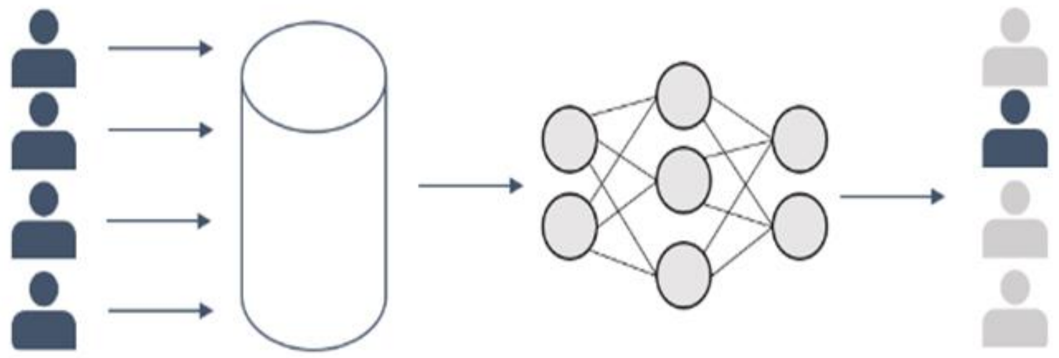
Since the tests involving input privacy were not strong, we also tested output privacy preservation.

To understand the concept of a membership inference attack we will answer these questions:

- What does a membership inference attack mean and how can be used in order to violate individual privacy?
- How do membership inference attacks work?
- How to implement membership inference attacks?
- What are the factors that increase a model's vulnerability against membership inference attacks and what are the main protective measures?

Scenario 2 - Background Information

Q: What does membership inference mean?
A: Given a trained ML model and some datapoint, decide whether this sample was part of the model's training set.

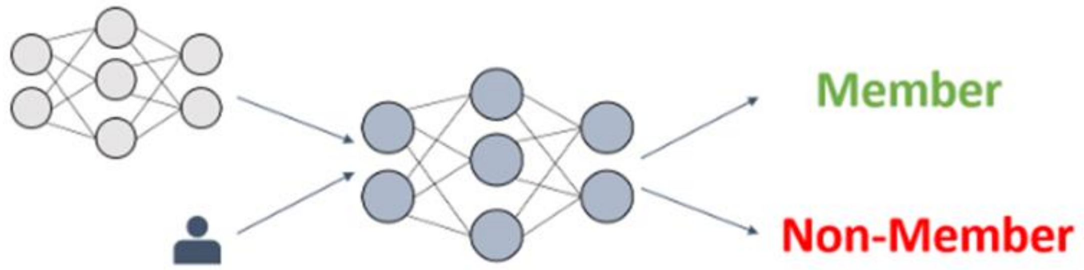


Q: How it can be used in order to violate individual privacy?
A: For example, imagine you are in a clinical context. There, you may have a ML model to predict an adequate medical treatment for cancer patients. This model, naturally, needs to be trained on the data of cancer patients. Hence, given a datapoint, if you are able to determine that it was indeed part of the model's training data, you will know that the corresponding patient must have cancer.

Scenario 2 - Background Information

Q: How do membership inference attacks work?

A: Most membership inference attacks work by building a binary meta-classifier F_{attack} which, given a model F and a data point x_i decides whether or not x_i was part of model training sample X



A: To train the binary meta-classifier, *shadow models* are built that imitate the behavior of the original ML model.
Shadow models use training datasets known to the attacker or that are generated by the attacker.
By exploiting the knowledge of the input and output data of the shadow models, the binary meta-classifier is trained.

Scenario 2 - Background Information

Q: How do you implement membership inference attacks?

A: There are several tools; two of them are:

- IBM-ART framework *(https://github.com/Trusted-AI/adversarial-robustness-toolbox)*
- **TensorFlow Privacy's Membership Inference** *(https://github.com/tensorflow/privacy/tree/master/tensorflow_privacy/privacy/privacy_tests/membership_inference_attack)*

A: Implementations can also be custom-built following the designs from research papers.

Scenario 2 - Background Information

Q: What are the factors that increase a model's vulnerability against membership inference attacks?

A: The main factors (that influence membership inference risks in ML models) are:

- **overfitting**
- classification problem complexity
- in-class standard deviation
- type of ML model targeted

Q: What are protective measures against membership inference attacks?

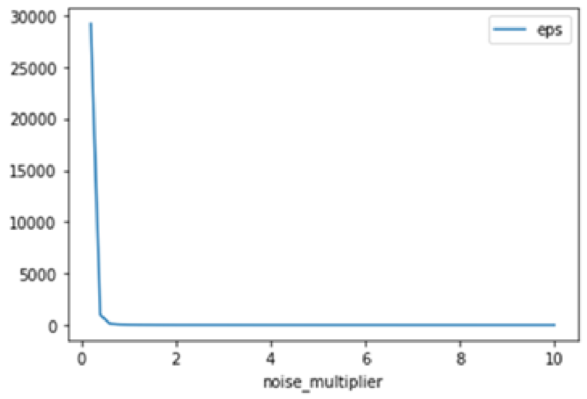
A: In addition to training models that do not fit the training data too tightly, one method that helps reduce the risk of membership inference is ***differential privacy***.

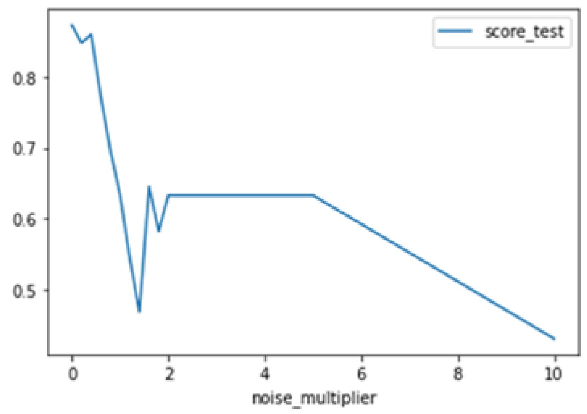
Scenario 2 - Results

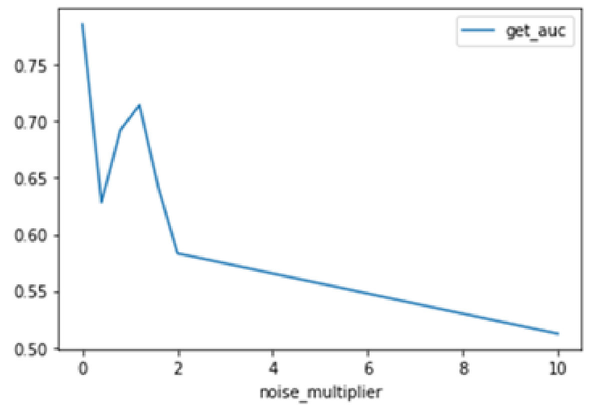
Trade-off evaluation between accuracy and utility (privacy preservation)

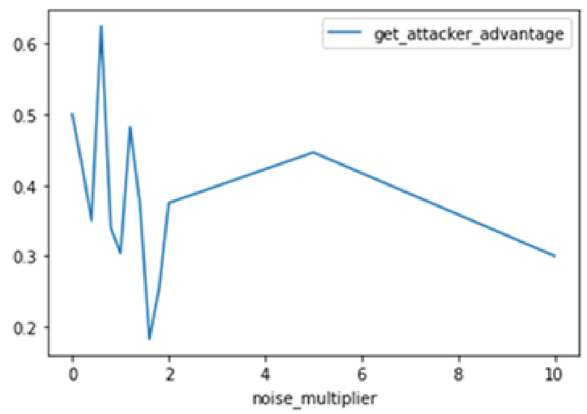
Parameters: Epochs = 200 - Learning_rate=0.1 - Noise_multiplier =[0,0.2,0.4,....]

Parameters			Scores of predictive model				Effectiveness of the attack			
epochs	noise_multiplier	learning _rate	score_train	score_valid	score_test	eps	attack_type	get_auc	get_attacker_ advantage	slice_spec
200	0	0,1	0,98016	0,79688	0,87342	inf	LOGISTIC_REGRESSION	0,73214	0,50102	CORRECTLY_CLASSIFIED=True
200	0,2	0,1	0,96032	0,76563	0,8481	2,92E+10	LOGISTIC_REGRESSION	0,73214	0,42857	CLASS=0
200	0,4	0,1	0,90079	0,78125	0,86076	1,01E+09	LOGISTIC_REGRESSION	0,66001	0,35001	Entire dataset
......	......	....	....	....	....	....	....	....	....	....









Scenario 2 - Example of Overfitting

epochs	noise_multiplier	learning_rate	attack_type	get_auc	get_attacker_advantage	slice_spec	score_train	score_valid	score_test	eps
.....	.....	.....	....	.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	....	.....	.....	.....	.....	.....	.....	.....
200	0,4	0,01	LOGISTIC_REGRESSION	0,8929	**0,75**	CLASS=0	**0,9206**	**0,7969**	0,8228	1,01E+07
200	0,5	0,01	LOGISTIC_REGRESSION	0,9643	**0,8571**	CLASS=0	**0,9008**	**0,7344**	0,8101	3,10E+06
200	0,6	0,01	LOGISTIC_REGRESSION	0,6786	**0,3571**	CLASS=0	**0,8135**	**0,75**	0,7215	1,39E+06
.....	.....	.....	....	.....	.....	....	.....	.....	.....	.....

Overfitting occurs when the model cannot generalize and fits too closely to the training dataset (high difference between score validation and score training).

When the original model (the model that classifies whether students pass or fail) is overfitted (see first two rows), there is an advantage for the attacker because the model fitted too much to the training dataset and this makes it easier to identify the data of training and the membership inference (see get attacker advantage). In the third row we have not detected overfitting and the get_advantage_attacker decreases.

Experiment 3 - Custom Implementation of a Membership Inference Attack

For this final experiment we created a custom implementation of the classic membership inference attack proposed by Shokri et al.:

1. Generate synthetic datasets to train and test the shadow models with model-based synthesis (one train/test set per shadow model)
2. Initialize, train, and test k shadow models with the synthetic datasets (one dataset per shadow model)
3. Label each sample in the synthetic datasets with a 1 if used for training a shadow model and a 0 otherwise
4. Obtain a prediction vector for each dataset, from the corresponding shadow model
5. Train an attack model for each actual class label (i.e. pass/fail) with the prediction vectors and new labels, from the synthetic datasets, to learn whether a sample of a class has been used for training
    - Φ_0 is the attack model for samples of class 0 and Φ_1 is the attack model for samples of class 1

Experiment 3 - Custom Implementation of Membership Inference Attack

Purpose: Create a codebase where we have more control over the attack and can track more outputs for evaluation, leading to more specific observations.

Implementation: Follows the exact structure from the original paper, with TensorFlow used for training the models.

Goal: Observe how the attack's performance adjusts when training normally and training with differential privacy.

Experiment 3 - Preliminary Results

Run Information	**Attack Accuracy**	**Attack Precision (Φ_0)**	**Attack Precision (Φ_1)**	**Attack Recall (Φ_0)**	**Attack Recall (Φ_1)**	**Attack F1-Score (Φ_0)**	**Attack F1-Score (Φ_1)**
Uses DP Epsilon = 1.1 Initial Model Performance: Accuracy: 68% Prec=0: 0.86 Prec=1: 0.67 Rec=0: 0.2 Rec=1: 0.98	Φ_0: 0.55 Φ_1: 0.51	Test set: 0.28 Training set: 0.85	Test set: 0.25 Training set: 0.81	Test set: 0.93 Training set: 0.13	Test set: 0.61 Training set: 0.48	Test set: 0.43 Training set: 0.23	Test set: 0.36 Training set: 0.61
No DP Initial Model Performance: Accuracy: 89% Precision = 0: 0.96 Precision = 1: 0.86 Recall = 0: 0.73 Recall = 1: 0.98	Φ_0: 0.61 Φ_1: 0.61	Test set: 0.32 Training set: 0.76	Test set: 0.2 Training set: 0.77	Test set: 0.40 Training set: 0.68	Test set: 0.24 Training set: 0.71	Test set: 0.35 Training set: 0.72	Test set: 0.22 Training set: 0.74

Experiment 3 - Outcomes

- The attack is more effective when no DP is used
- Using DP, only a smaller subset of the training data can be identified
  - Protecting more input data from being reconstructed synthetically and validated with the attack model
  - However, a small subset of data is still being identified well
- It is harder to synthesize data with high confidence for a target class when DP is used
- These tests use a basic setup, further optimizations will help better the attack model's effectiveness and the DP effectiveness

Challenges and Lessons Learned

- Input privacy techniques (i.e. differential privacy) can make machine learning models safer in terms of privacy preservation.
- Output privacy can be better secured from membership inference attacks when applying differential privacy when training.
- Input and output privacy requires thorough testing and optimization to effectively protect the private data and machine learning model.
- Synthetically generating data on smaller datasets is challenging and can negatively impact the performance of the trained model.
- Not all open source solutions offer a robust way to perform and evaluate input and output privacy against specific types of attacks.

Conclusions

- These experiments present preliminary results which exhibit the importance and utility of input and output privacy techniques.
- When collaborations are held between NSOs and/or organizations, any derived machine learning model should be secure from appropriate attacks.
- PETs can act as a way to both allow collaborations to be developed and to protect the results of the collaboration from potential threats (both for input and output privacy).

Next Steps

- Develop use cases of official statistics that require the application of private machine learning techniques.
- Carry out research and experimentation on the subject of input privacy preservation by joining international working groups (i.e. UN PET Lab).
- Continue building codebases which can be utilized to test how secure a machine learning model is from relevant attacks.

References

- Franziska Boenisch, ” Attacks against Machine Learning Privacy (Part 2): Membership Inference Attacks with TensorFlow Privacy” - Published: January 24, 2021 - https://franziska-boenisch.de/posts/2021/01/membership-inference/

- Shokri, Reza, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. “Membership inference attacks against machine learning models.” In 2017 IEEE Symposium on Security and Privacy (SP), pp. 3-18. IEEE, 2017

- https://github.com/tensorflow/privacy/tree/master/tensorflow_privacy/privacy/privacy_tests/membership_inference_attack

- https://github.com/VectorInstitute/PETs-Bootcamp/tree/main/Membership_Inference_Attacks

Questions?

(julian.templeton@statcan.gc.ca)