IBM Research

Diffprivlib: Privacy-preserving machine learning with Scikit-learn

Naoise Holohan
IBM Research Europe – Ireland

Traditional anonymisation overtaken by 21st Century data

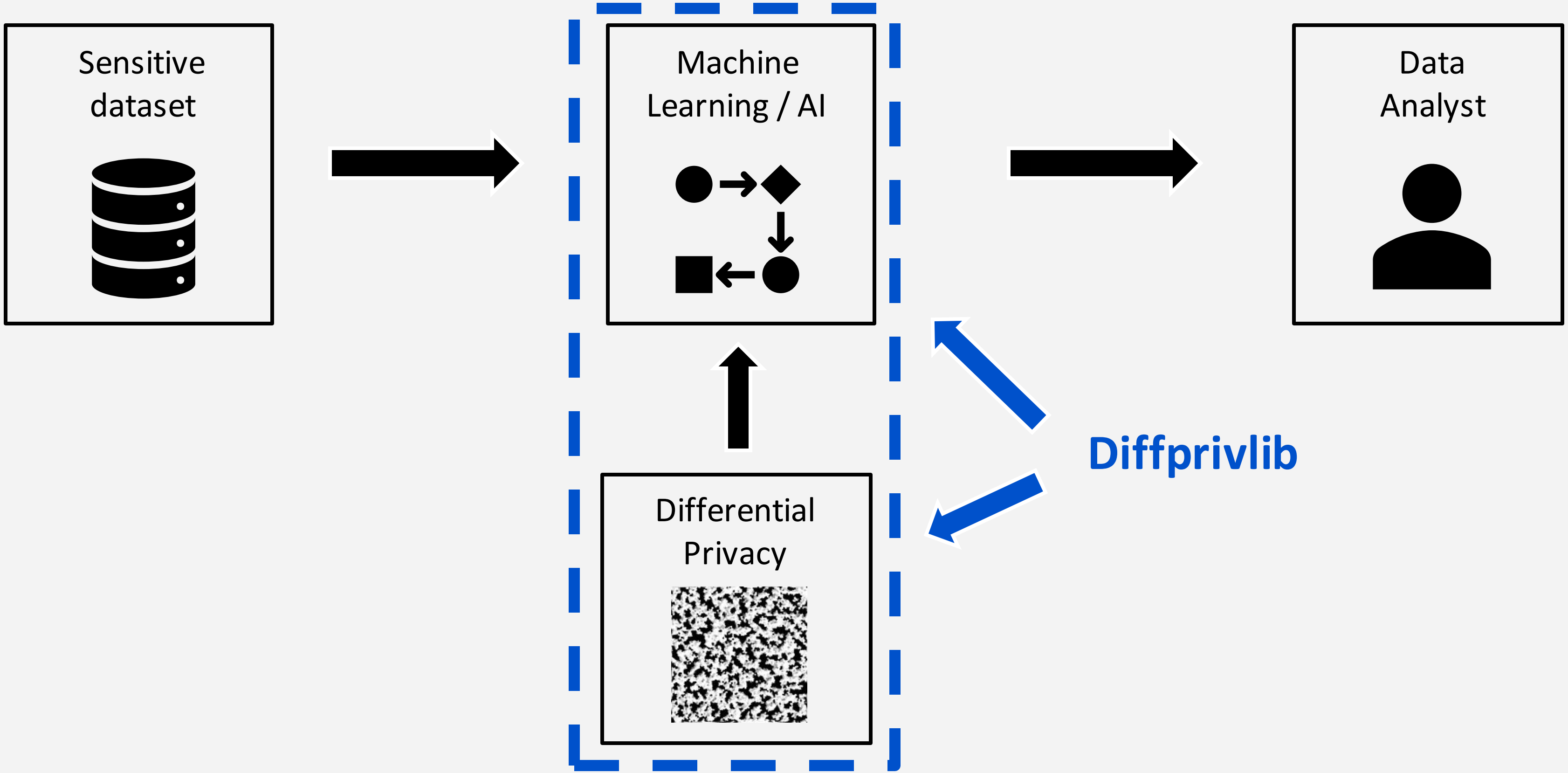
- Traditional anonymisation is crucial to safeguard sensitive data
- Risk of de-anonymisation when linked with external datasets
- Many examples of attacks on release of “anonymised” data
- Statistics are also vulnerable to database reconstruction and model inversion attacks

Privacy for 21st Century Big Data: Differential Privacy

Key Idea: Blur the data

- Individual privacy preserved
- Population trends still observable
- Privacy is <u>future proof</u>
- Queries have a privacy budget ϵ

Example use-case



Our Approach

- Python is popular for machine learning
- NumPy and Scikit-Learn are standard for data analytics and machine learning
- Require a virtually identical user experience to Numpy and Scikit-Learn
- Default privacy parameter setting
- Ensure users are already familiar with diffprivlib before using it

Diffprivlib in a nutshell

```
In [3]: from diffprivlib.models import GaussianNB

        bounds = ([4.3, 2, 1, 0.1], [7.9, 4.4, 6.9, 2.5])

        clf = GaussianNB(bounds=bounds)
        clf.fit(X_train, y_train)

Out[3]: GaussianNB(accountant=BudgetAccountant(spent_budget=[(1.0, 0)]),
                   bounds=(array([4.3, 2. , 1. , 0.1]), array([7.9, 4.4, 6.9,
        2.5])),
                   epsilon=1.0, priors=None, var_smoothing=1e-09)

In [4]: clf.predict(X_test)

Out[4]: array([0, 2, 0, 0, 2, 1, 1, 1, 2, 1, 0, 1, 1, 2, 1, 1, 2, 1, 2, 2, 1,
        1,
               1, 0, 1, 0, 1, 0, 1, 0])

In [5]: print("Test accuracy: %f" % clf.score(X_test, y_test))

        Test accuracy: 0.933333
```

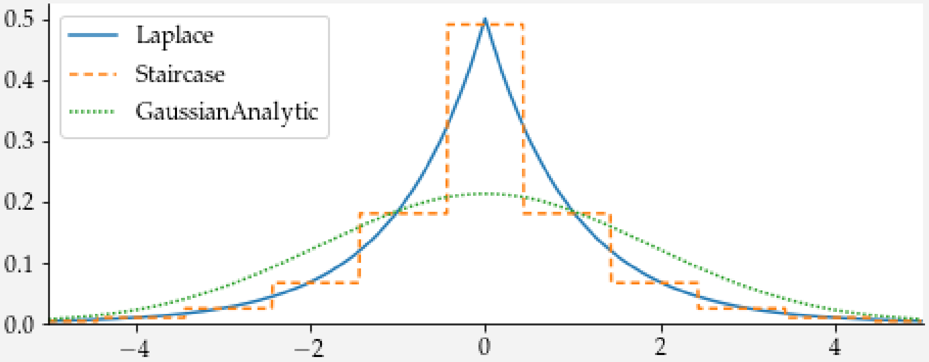
- Machine Learning with differential privacy
- No expertise required
- Open Source – free to use and modify
- Easy installation
- Integration with popular packages (Scikit-learn, NumPy)
- Easily integrated within existing applications

Modules: Mechanisms, Models, Tools, Accountant

```
>>> from diffprivlib.mechanisms import Laplace
>>> mech1 = Laplace().set_epsilon(1).set_sensitivity(1)
>>> mech1.randomise(1)
0.4371098324798539

>>> from diffprivlib.mechanisms import GaussianAnalytic
>>> mech2 = GaussianAnalytic().set_epsilon_delta(1,
  0.01).set_sensitivity(1)
>>> mech2.randomise(1)
-0.0002084664240138423
```

- Primitives for noise addition to achieve differential privacy
- Used under-the-hood in all tools/models



Modules: Mechanisms, Models, Tools, Accountant

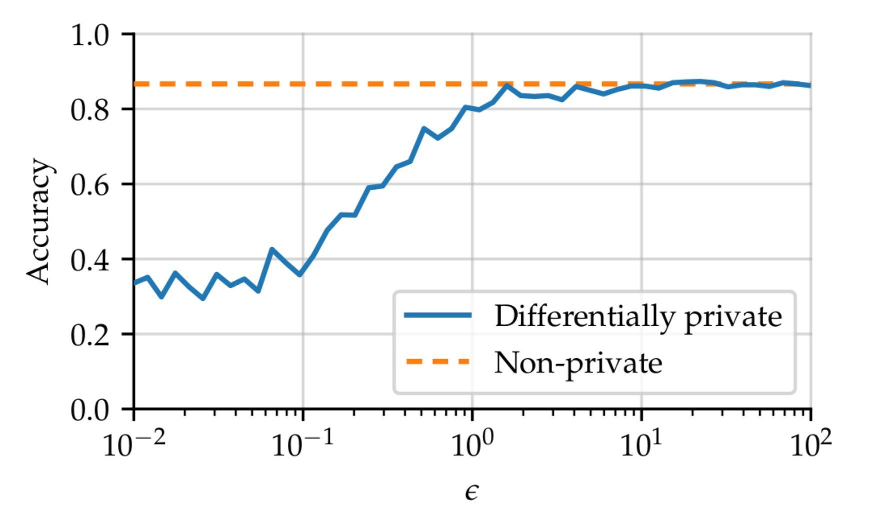
```
>>> from diffprivlib.models import GaussianNB

>>> clf = GaussianNB()
>>> clf.fit(X_train, y_train)
PrivacyLeakWarning: Bounds have not been specified and will be
  calculated from the data provided.  This will result in addi-
  tional privacy leakage.  To ensure differential privacy and
  no additional privacy leakage, specify bounds for each
  dimension.

>>> clf.predict(X_test)
array([1, 0, 2, 1, 2, 1, 2, 1, 0, 0, 1, 2, 2, 0, 0, 0, 1, 1, 1,
  1, 0, 2, 1, 1, 0, 0, 1, 0, 0, 1])

>>> (clf.predict(X_test) == y_test).sum() / y_test.
  shape[0]
0.9333333333333333
```

- Machine learning models with differential privacy built-in
- Each model inherits its Scikit-Learn equivalent as its parent class



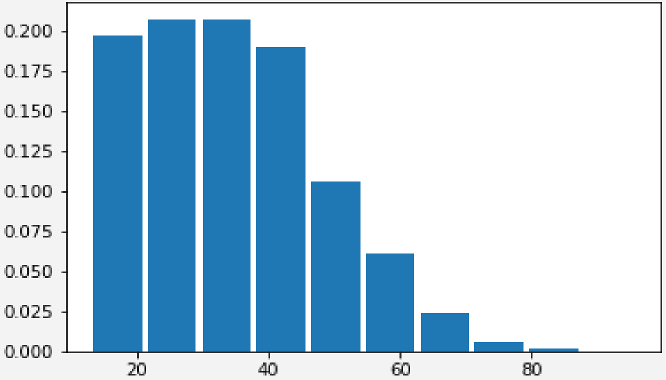
Modules: Mechanisms, Models, Tools, Accountant

```
>>> import diffprivlib.tools as tools
>>> tools.mean(Adult_ages, range=100)
38.57757804280589

>>> tools.std(Adult_ages, range=100)
13.672743942658721

>>> tools.histogram(Adult_ages, range=(0,100))
(array([ 1, 1658, 8054, 8611, 7175, 4418, 2015, 508, 77, 43]),
  array([ 0., 10., 20., 30., 40., 50., 60., 70., 80., 90.,
  100.]))
```

- NumPy functions for simple data analytics
- Histograms are especially useful in differential privacy

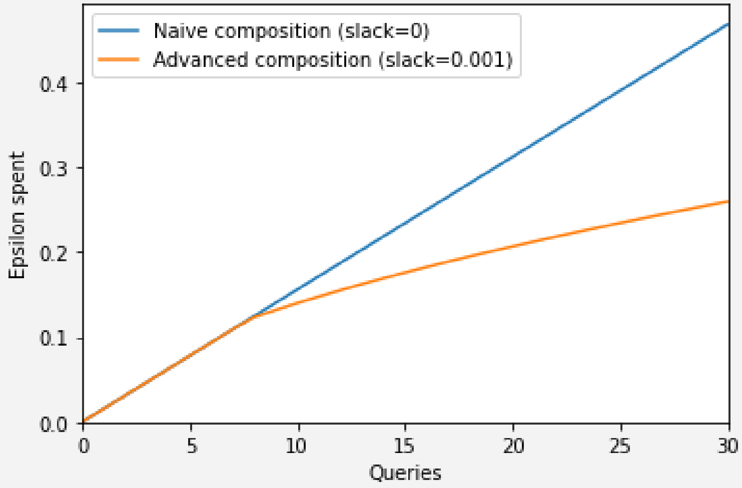


Modules: Mechanisms, Models, Tools, Accountant

```
>>> import diffprivlib as dp
>>> with dp.BudgetAccountant() as acc:
...       mean = dp.tools.mean(Adult_ages, epsilon=0.1)
...       std = dp.tools.std(Adult_ages, epsilon=0.1)
...       hist = dp.histogram(Adult_ages, epsilon=0.1)

>>> acc.total()
(epsilon=0.3, delta=0.0)
```

- Track privacy budget spend across multiple calls to diffprivlib
- Advanced composition techniques ensure better accuracy with the same privacy budget



Demo

Additional Resources

- Github repository:
  github.com/IBM/differential-privacy-library
- Documentation:
  diffprivlib.readthedocs.io
- Installation:
  `pip install diffprivlib`


IBM Research
Dublin Research Lab
naoise@ibm.com

Back-up slides

A simple example

Participant	Actual answer		Noisy answer
A	0	→	1
B	0	→	0
C	1	→	0
D	1	→	1
⋮	⋮		⋮
Z	1	→	0
Total	**17**	**→**	**16**

Published data

- Individual values are not reliable
- No way to reconstruct originals
- Aggregate statistics still representative

Model parameters control privacy/accuracy trade-off

What is Differential Privacy?

Differential privacy is a measurement of privacy

- “SI Unit” for privacy of data release algorithm
- Provides an explicit, objective mathematical way to measure privacy
- Symbol: ϵ
- Quantity: Stochastic privacy

ϵ

$\epsilon = 0$
Perfect privacy
(Zero utility)

$\epsilon = \infty$
Perfect utility
(Zero privacy)

Solutions are use-case driven

- No silver bullet
- Toolbox of solutions needed for every problem
- **Key challenge:** Preserve privacy and maintain accuracy

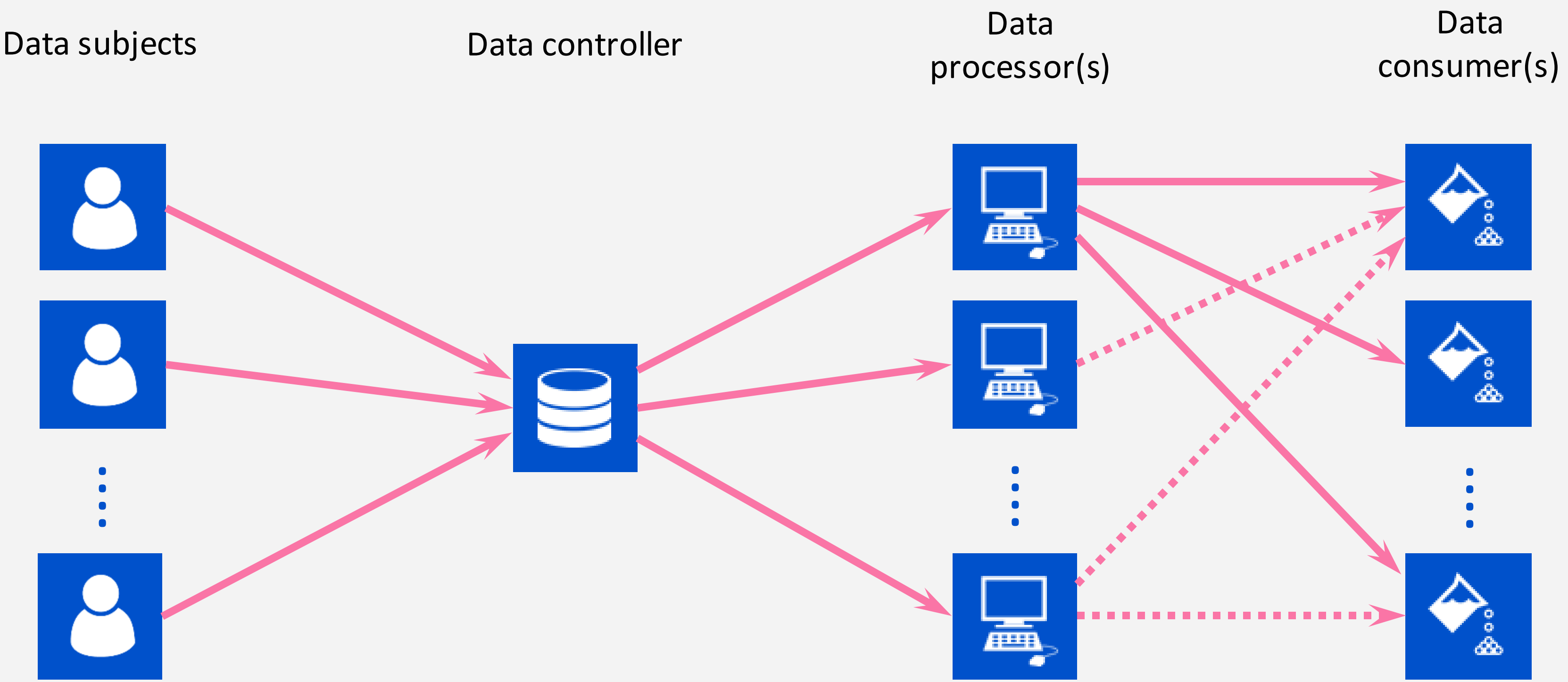
Differential Privacy Checklist:

- Large quantity of data
- Tolerance to error
- Appreciable privacy risk

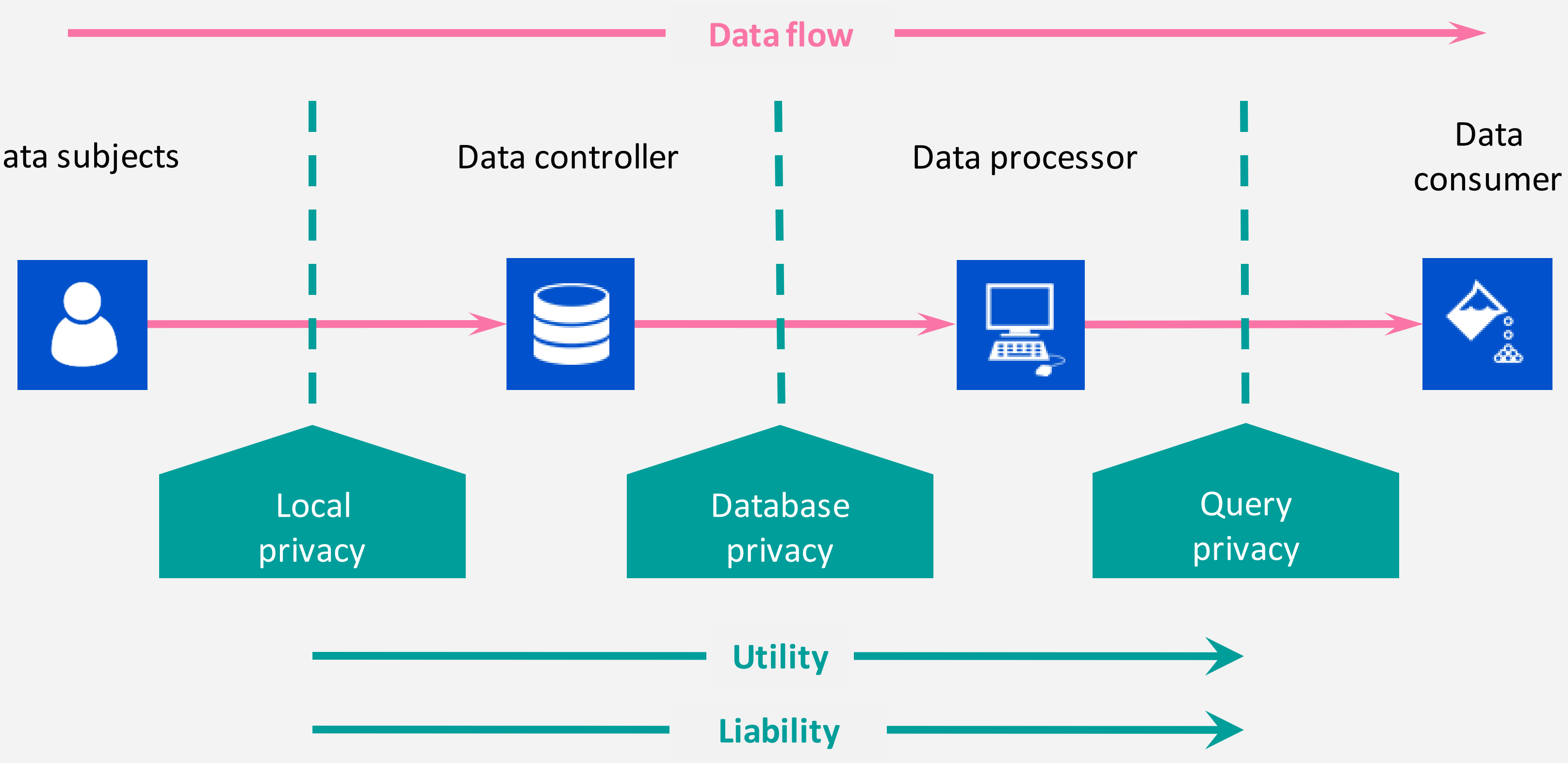
Weak use-case: Doctor's access to a patient's health records (errors not tolerable)

Strong use-case: Data scientist's access to a hospital's patient dataset

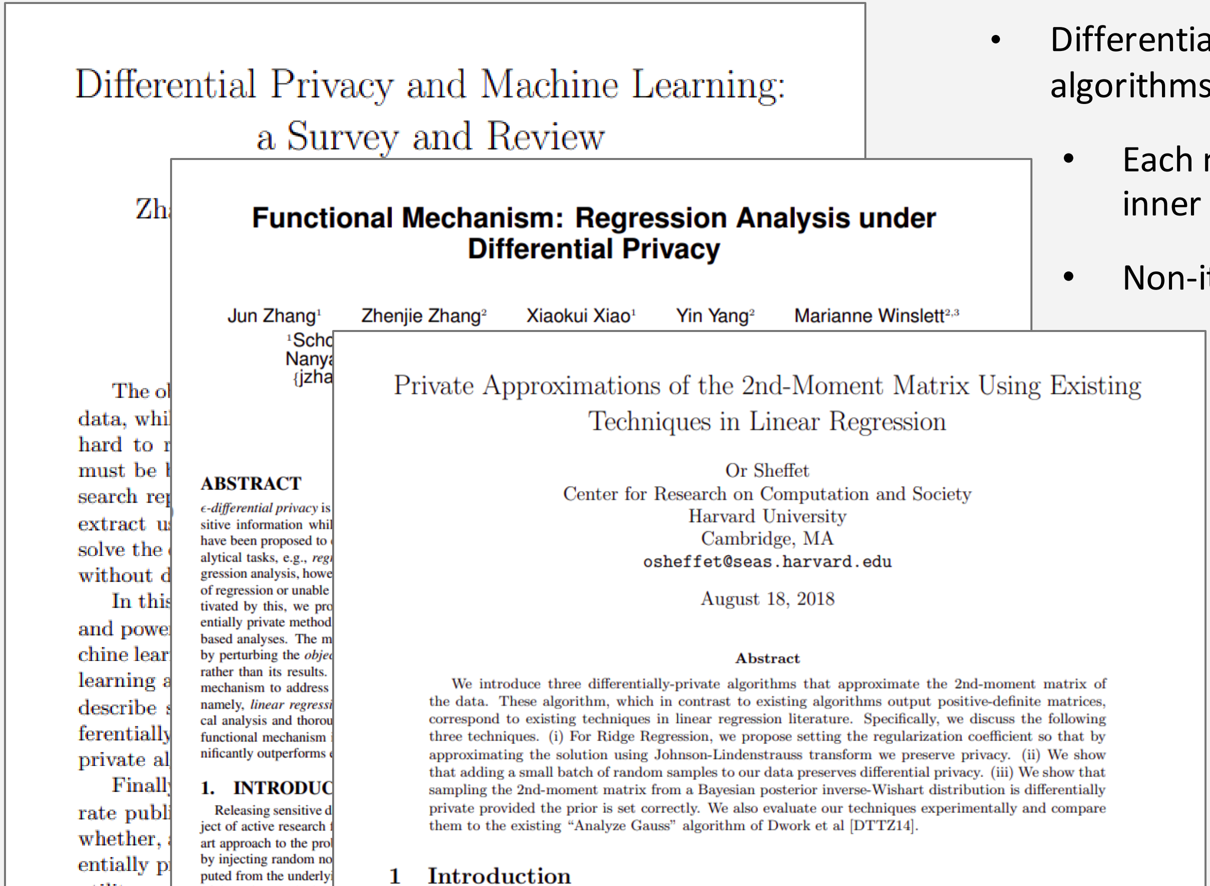
Who do we trust with data?



Trust boundaries



State-of-the-art: Literature



Differential Privacy and Machine Learning: a Survey and Review

Functional Mechanism: Regression Analysis under Differential Privacy

Jun Zhang[1] Zhenjie Zhang[2] Xiaokui Xiao[1] Yin Yang[2] Marianne Winslett[2,3]

Private Approximations of the 2nd-Moment Matrix Using Existing Techniques in Linear Regression

Or Sheffet
Center for Research on Computation and Society
Harvard University
Cambridge, MA
osheffet@seas.harvard.edu

August 18, 2018

Abstract

We introduce three differentially-private algorithms that approximate the 2nd-moment matrix of the data. These algorithm, which in contrast to existing algorithms output positive-definite matrices, correspond to existing techniques in linear regression literature. Specifically, we discuss the following three techniques. (i) For Ridge Regression, we propose setting the regularization coefficient so that by approximating the solution using Johnson-Lindenstrauss transform we preserve privacy. (ii) We show that adding a small batch of random samples to our data preserves differential privacy. (iii) We show that sampling the 2nd-moment matrix from a Bayesian posterior inverse-Wishart distribution is differentially private provided the prior is set correctly. We also evaluate our techniques experimentally and compare them to the existing "Analyze Gauss" algorithm of Dwork et al [DTTZ14].

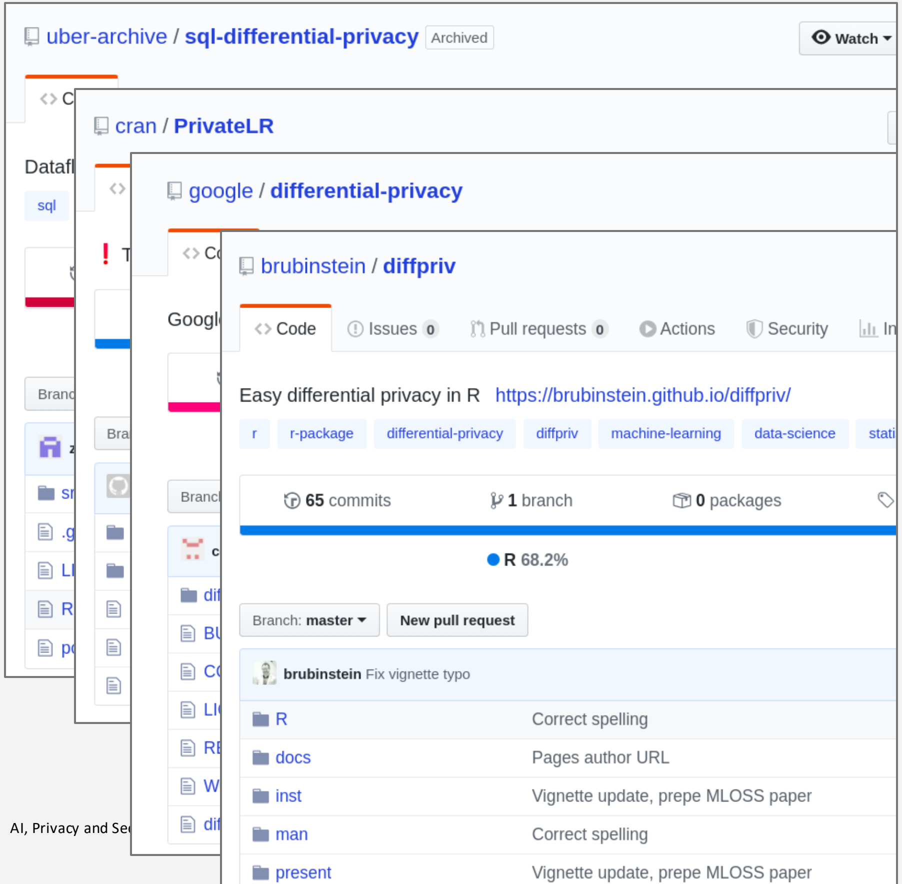
1 Introduction

- Differentially private solutions to machine learning algorithms already exist
  - Each model requires a custom solution to fit the inner workings of that model
  - Non-iterative models suit best

Existing solutions include:

- Linear regression
- Logistic regression
- Decision trees
- Random forest
- Principal Component Analysis
- Support Vector Machines
- K-means clustering
- Naïve Bayes

State-of-the-art: Code



- Many distinct libraries
- No common codebase, no standard syntax
- Many different languages
- ML “libraries” implementing a single algorithm