Diffprivlib: Privacy-preserving machine learning with Scikit-learn

Naoise Holohan
IBM Research Europe – Ireland
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 21\textsuperscript{st} Century Big Data: Differential Privacy

Key Idea: Blur the data

- Individual privacy preserved
- Population trends still observable
- Privacy is future proof
- Queries have a privacy budget $\epsilon$
Example use-case

Sensitive dataset → Machine Learning / AI

Differential Privacy

Data Analyst

Diffprivlib
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

- 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

```python
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, 2, 1, 0, 1, 1, 2, 1, 2, 1, 2, 1, 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
```
• Primitives for noise addition to achieve differential privacy

• Used under-the-hood in all tools/models
Machine learning models with differential privacy built-in

Each model inherits its Scikit-Learn equivalent as its parent class

```python
>>> 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 additional 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, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 0, 2, 1, 1, 0, 0, 1, 0, 0, 0, 0]])

>>> (clf.predict(X_test) == y_test).sum() / y_test.shape[0]
0.9333333333333333
```
• NumPy functions for simple data analytics
• Histograms are especially useful in differential privacy
Modules: Mechanisms, Models, Tools, Accountant

```python
>>> 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:

• Documentation:
  [diffprivlib.readthedocs.io](https://diffprivlib.readthedocs.io)

• Installation:
  `pip install diffprivlib`
IBM Research
Dublin Research Lab
nai0ise@ibm.com
A simple example

<table>
<thead>
<tr>
<th>Participant</th>
<th>Actual answer</th>
<th>Noisy answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0 → 1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0 → 0</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1 → 0</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>1 → 1</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Z</td>
<td>1 → 0</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>17 → 16</td>
<td></td>
</tr>
</tbody>
</table>

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: $\epsilon$
- Quantity: Stochastic privacy

$\epsilon = 0$
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?

Data subjects -> Data controller -> Data processor(s) -> Data consumer(s)

AI, Privacy and Security Team, Dublin Research Lab / June 2020 / © 2020 IBM Corporation
Trust boundaries

- Data subjects
- Data controller
- Data processor
- Data consumer

- Local privacy
- Database privacy
- Query privacy

Utility

Liability
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