The University of Vermont

# Evaluating the Usability of Differential Privacy Tools with Data Practitioners

**Ivoline Ngong, Brad Stenger , Joseph Near , Yuanyuan Feng**

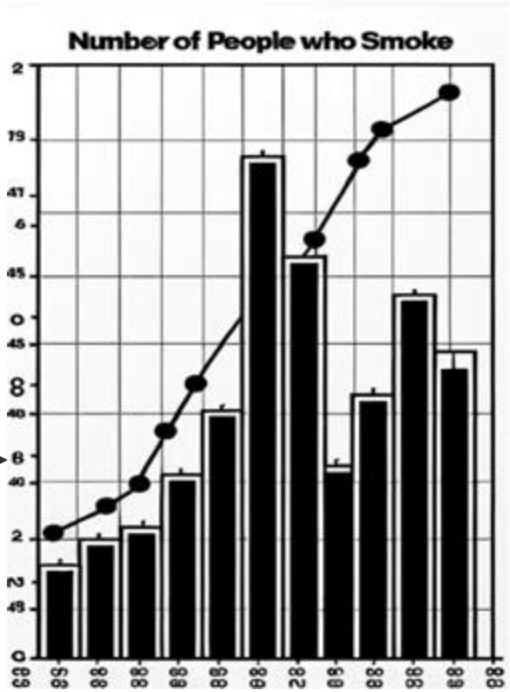

SOUPS – August 2024

## GOALS

- Understand usability issues in differential privacy.
- Make recommendations for improving usability

Let’s see a chart of the number of people who smoke versus the number of people with cancer

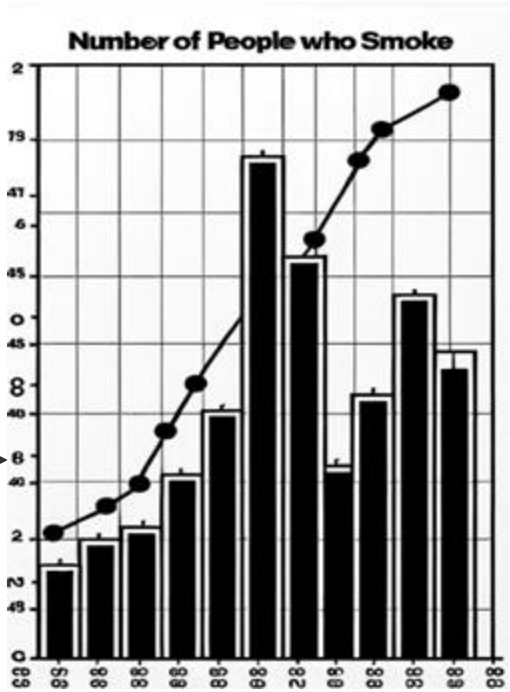
Oh no, the results could accidentally reveal that John has cancer!




Sensitive Query Result

**Smoking Causes Cancer**

Differential Privacy
is the solution!

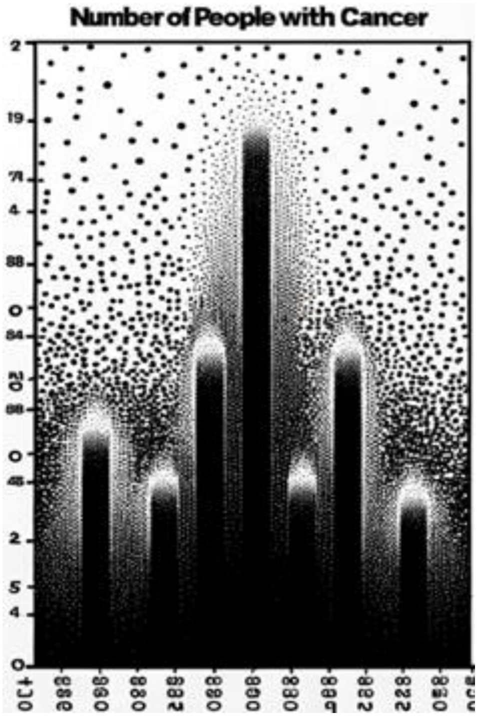



Sensitive Query Result

**Smoking Causes Cancer**

+

Calibrated Random Noise




Differentially Private Result

**Smoking Causes Cancer**

# DP is Challenging to Implement and Error Prone

- It's hard to understand DP terms and concepts especially for non-experts.
- There are many tools out there; it's hard to choose the right one.
- Even after using a tool, it's easy to make mistakes & interpreting results is hard. Are the results correct? what do they mean? how do you use them?
- Existing tool effectiveness and ease of use are unclear, possibly slowing DP adoption.

**Goal**: `Compare usability of open-source DP tools for data scientist"`

# Research Questions

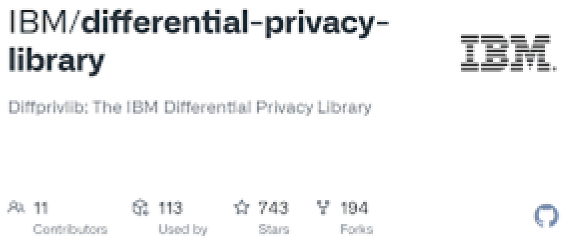
RQ1: How effectively can DP tools help data practitioners understand DP concepts?
(**DP Understanding**)

RQ2: How effectively can DP tools help data practitioners implement DP solutions?
(**DP Implementation**)

RQ3: How satisfied are data practitioners with DP tools for their DP implementation?
(**User Satisfaction)**

# Selected DP Tools

Tumult Analytics




DiffPrivLib

PipelineDP

OpenDP

# Selected DP Tools

IBM/differential-privacy-

Tumult Analytics

### Selection Criteria

1. Open source
2. Python-based
3. Aim for accessibility for non-experts
4. Comprehensive Documentation
5. Support Usability Tasks
6. No server requirements

PipelineDP

OpenDP

## Study Design


- Distributed **eligibility survey** alongside recruitment ads.
- Assessed Python and **DP knowledge**.
- **24 data practitioners**, including 12 DP novices & 12 experts
- **Between-subjects design** - each participant to one of the four DP tools
- **DP Experts -** answered 3/4 DP questions correctly

# Study Design


**DP Questions**

from Eligibility Survey

(6) Have you heard of the term differential privacy (DP) before?
  (a) No
  (b) Yes
(7) Have you ever written code to implement differential privacy (DP) in any capacity?
  (a) No
  (b) Yes
(8) In differential privacy, which value of the privacy parameter $\epsilon$ provides stronger privacy?
  (a) $\epsilon = 0.1$
  (b) $\epsilon = 1.0$
  (c) I don't know
(9) Releasing two differentially private statistics, one with $\epsilon_1 = 0.1$ and the other with $\epsilon_2 = 0.5$, results in a total privacy loss of:
  (a) $\epsilon = 0.1$
  (b) $\epsilon = 0.5$
  (c) $\epsilon = 0.6$
  (d) $\epsilon = 0.05$
  (e) I don't know
(10) If the mechanism M returns a number and satisfies differential privacy with $\epsilon = 0.1$, does abs(M(x)) satisfy differential privacy, where abs is the absolute value function?
  (a) No, not necessarily
  (b) Yes, for $\epsilon = 0.1$
  (c) Yes, for some $\epsilon > 0.1$
  (d) I don't know
(11) Which of the following is an advantage of using Differential Privacy?
  (a) It guarantees complete anonymity of the data subjects
  (b) It ensures that the data is completely accurate
  (c) It provides a tradeoff between privacy and utility of the data
  (d) It is a computationally simple method for preserving privacy in large datasets
  (e) I don't know

## Study Design

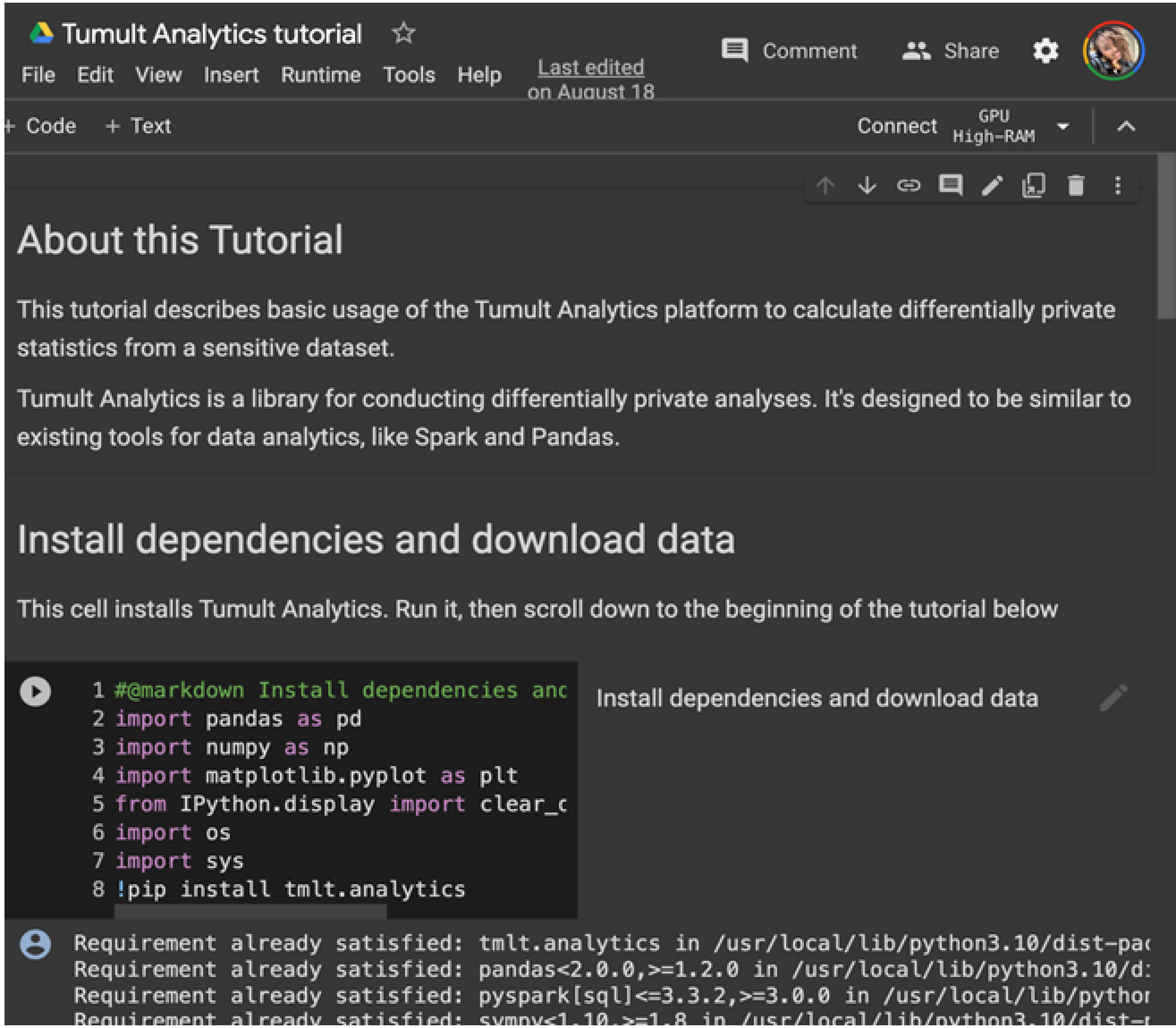


- Invited qualified respondents to usability study on Microsoft Teams.
- Participants **shared screens** and learned about think-aloud methodology.
- Reviewed DP fundamentals and tool requirements via **handout and tutorial.**
- Given access to **tool documentation** and permitted to use Google search (but not StackOverflow).

# Study Design



[DP Tool Tutorial]



# Study Design


- **Three tasks**: perform differentially private data analysis using DP tools.
- Participants spent 60 minutes **coding solutions with the tool.**

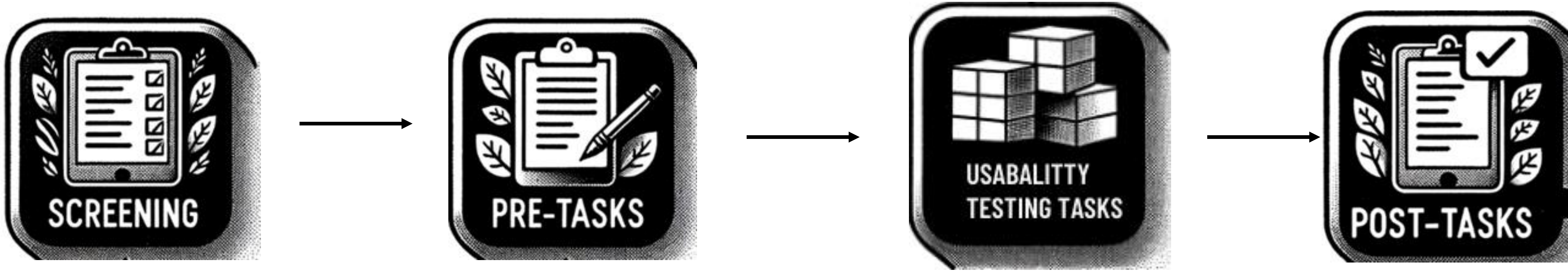
| Task | Description |
|---|---|
| Task 1 | How crowded is the restaurant on weekdays? (total number of visits for each weekday) |
| Task 2 | Total amount of time spent by visitors on each weekday (exclude weekends). |
| Task 3 | Average amount of time spent by visitors on each weekday (exclude weekends) |

COUNT

SUM

MEAN

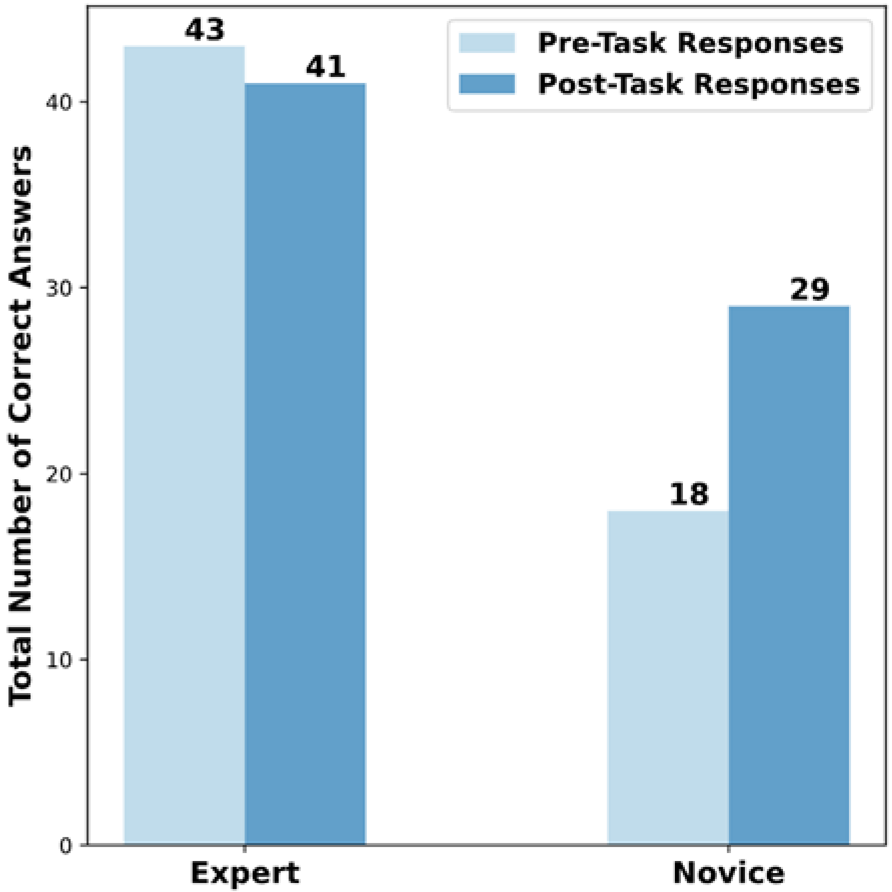
# Study Design



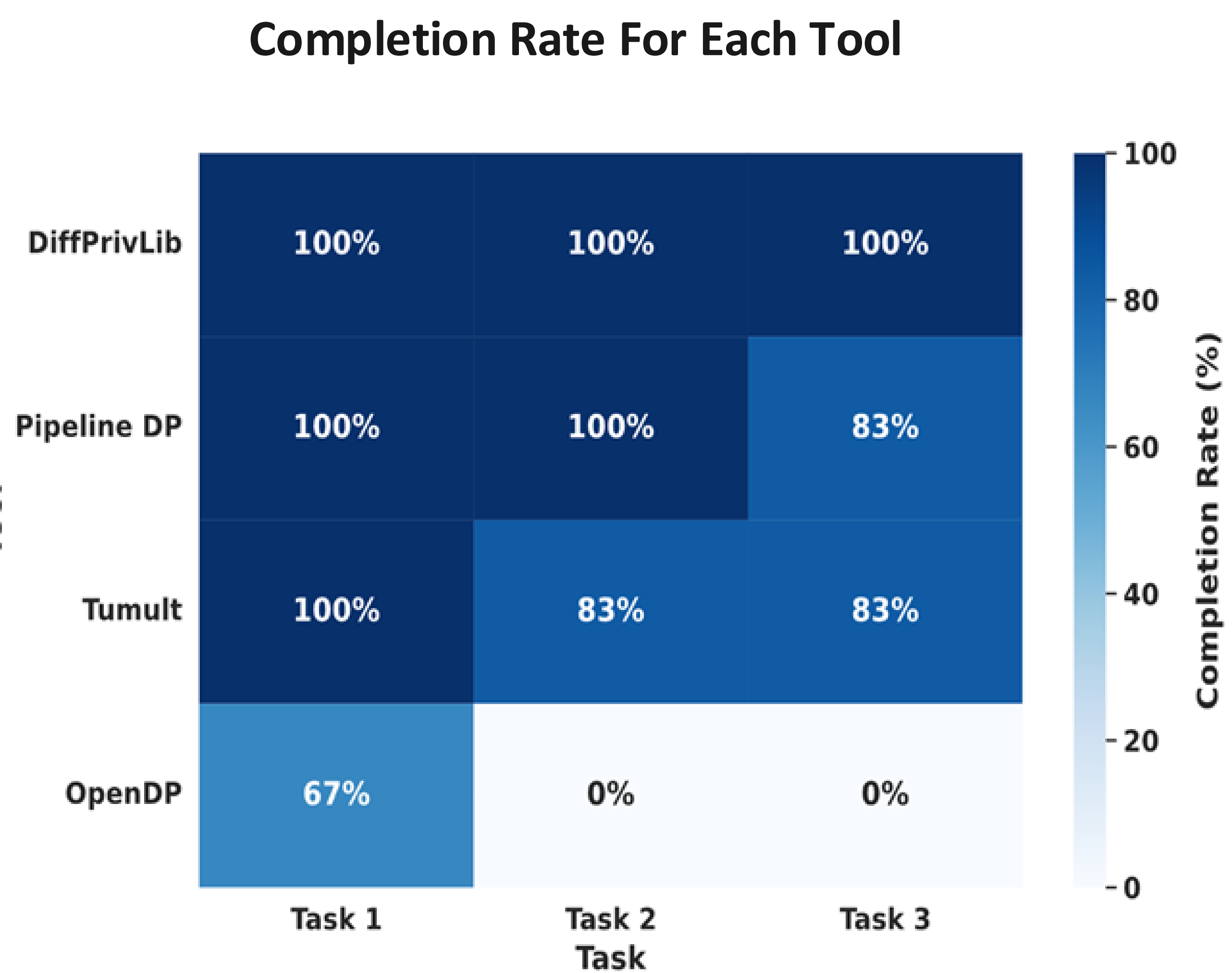

- Participants completed a **post-task survey** and **interview.**
- Survey assessed learning outcomes, experiences, and confidence.
- Interview provided deeper insights into participants' preferences, challenges, and suggestions.

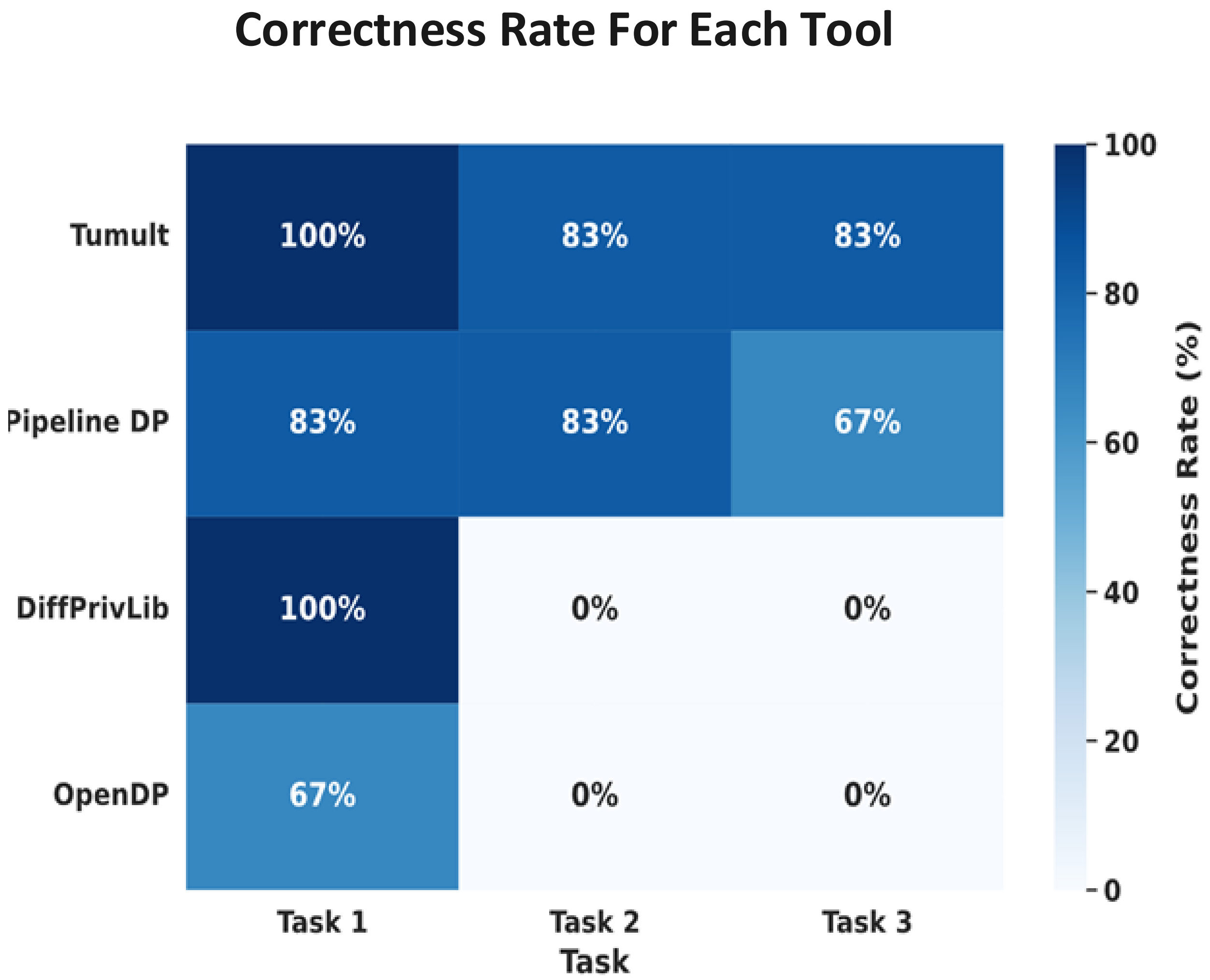
# Results

- Novices **scored higher** on conceptual questions in post-task survey



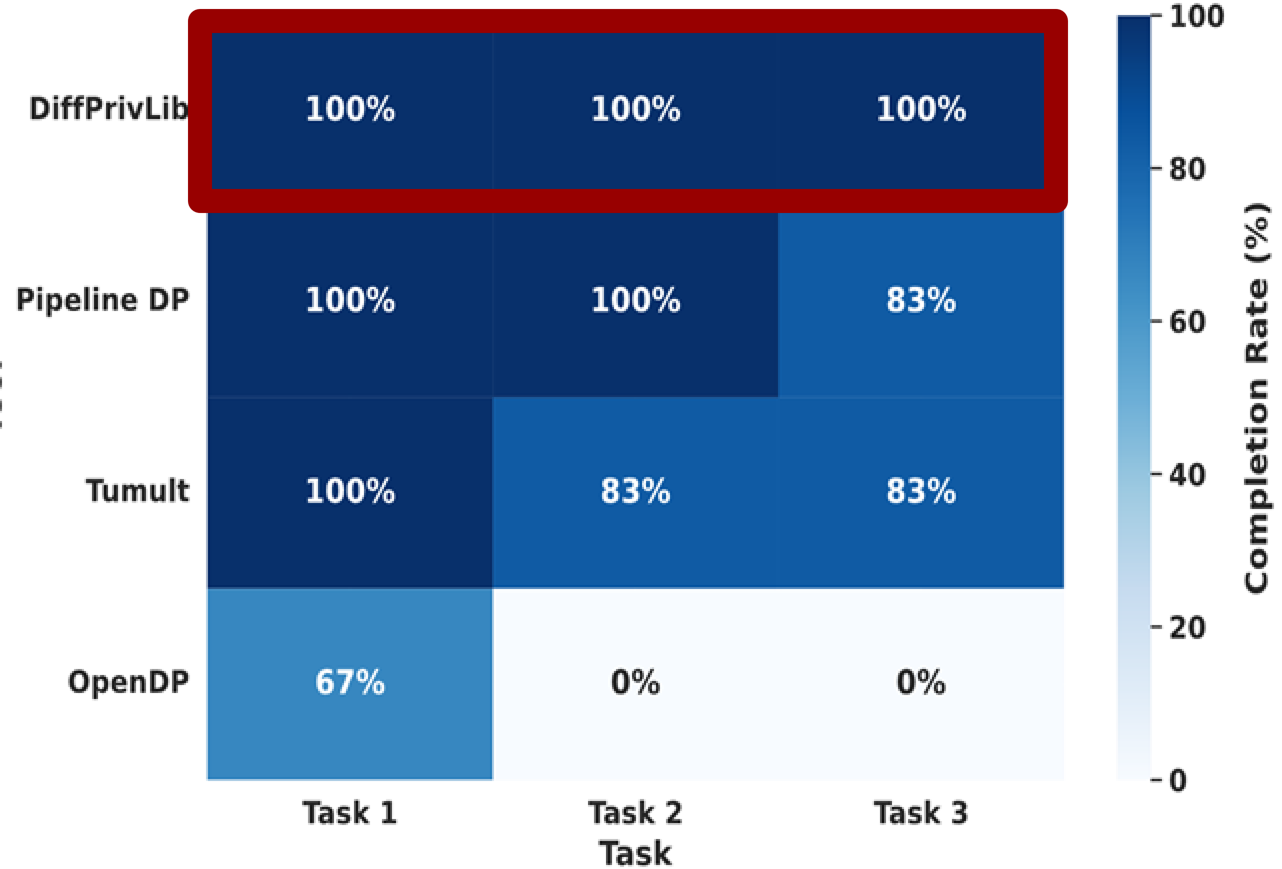
**Completion Rate**: code executed without error and produced correctly formatted responses.
**Correctness Rate:** code output satisfied DP and had comparable utility to our reference solutions.

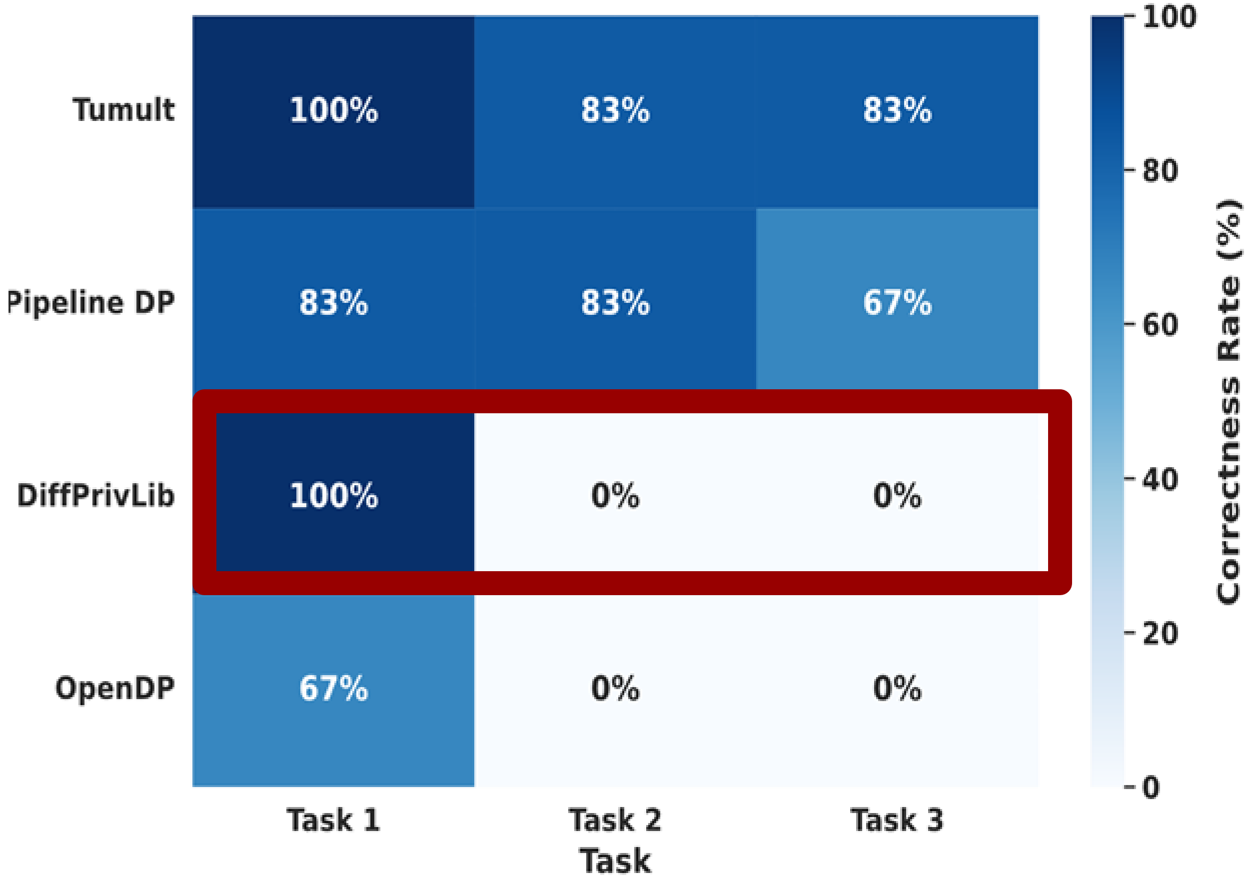


# DiffPrivLib users completed all tasks, but most were incorrect

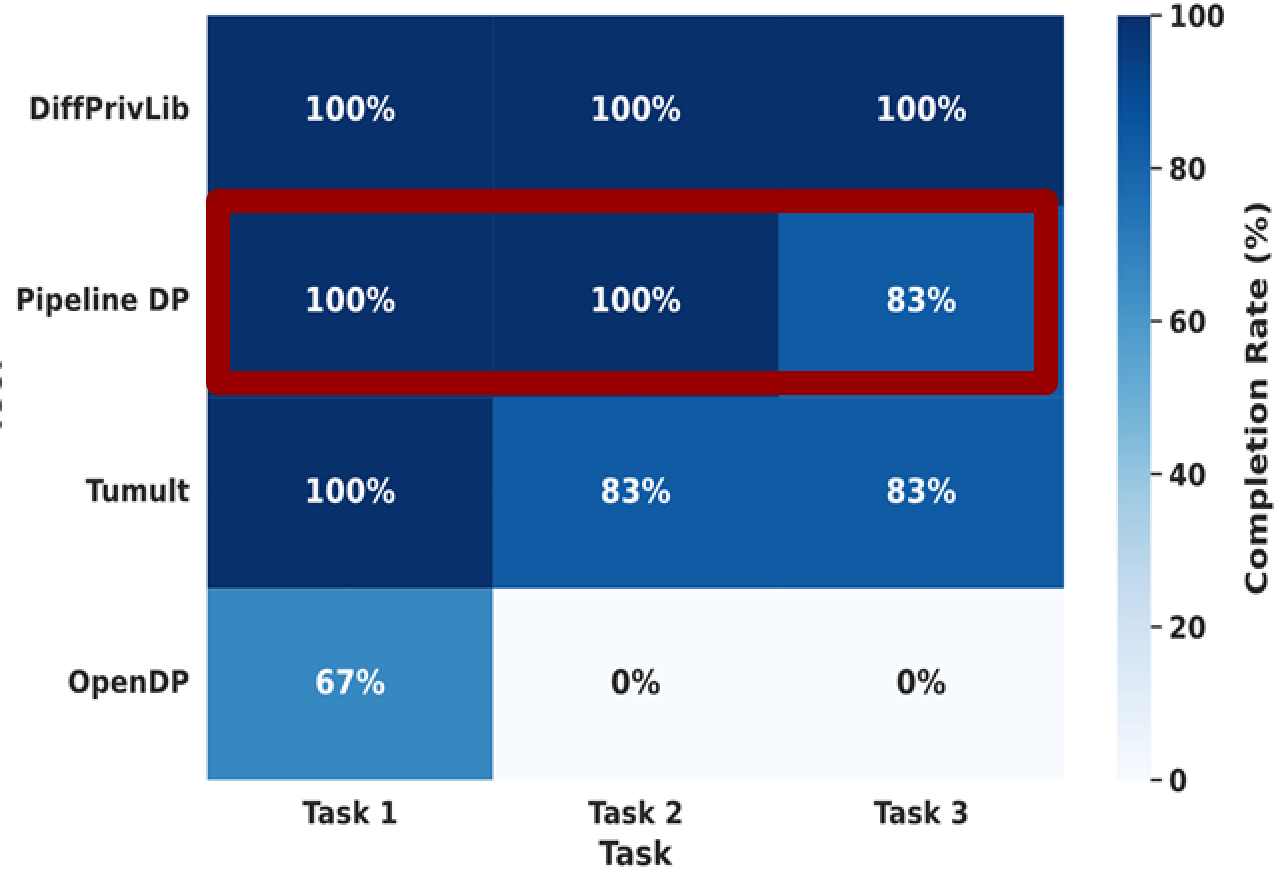
- All 6 participants violated dp
- Incorrect sensitivity settings / No clipping
- No error warning

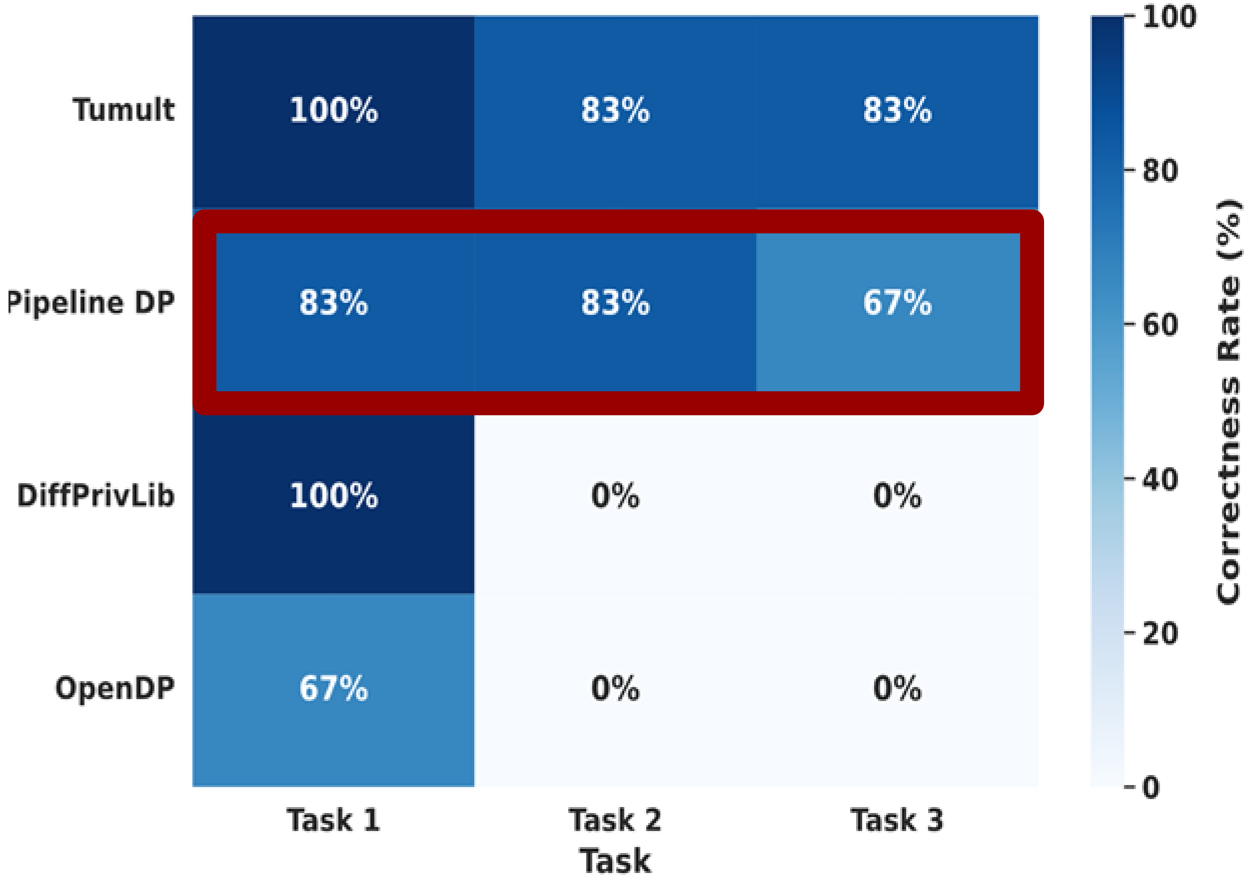


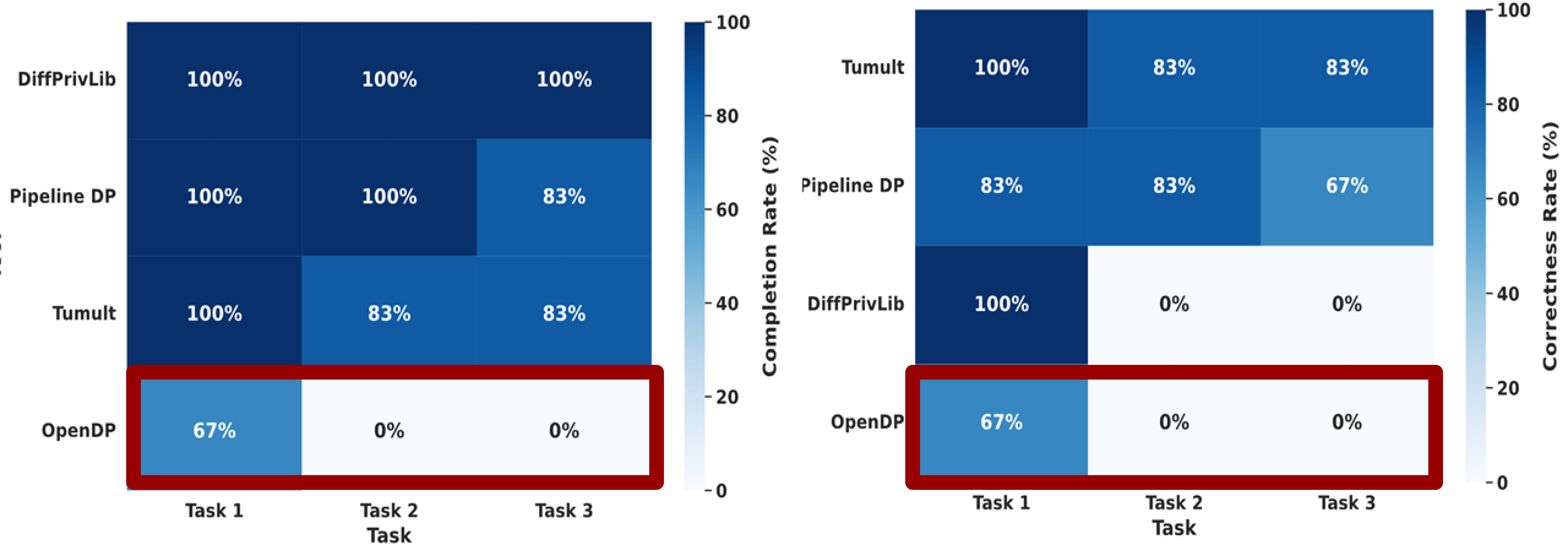
# PipelineDP users completed most tasks, but some were incorrect

- Easy to make mistakes
- No way validate noisy results
- Confusion about privacy budget tracking

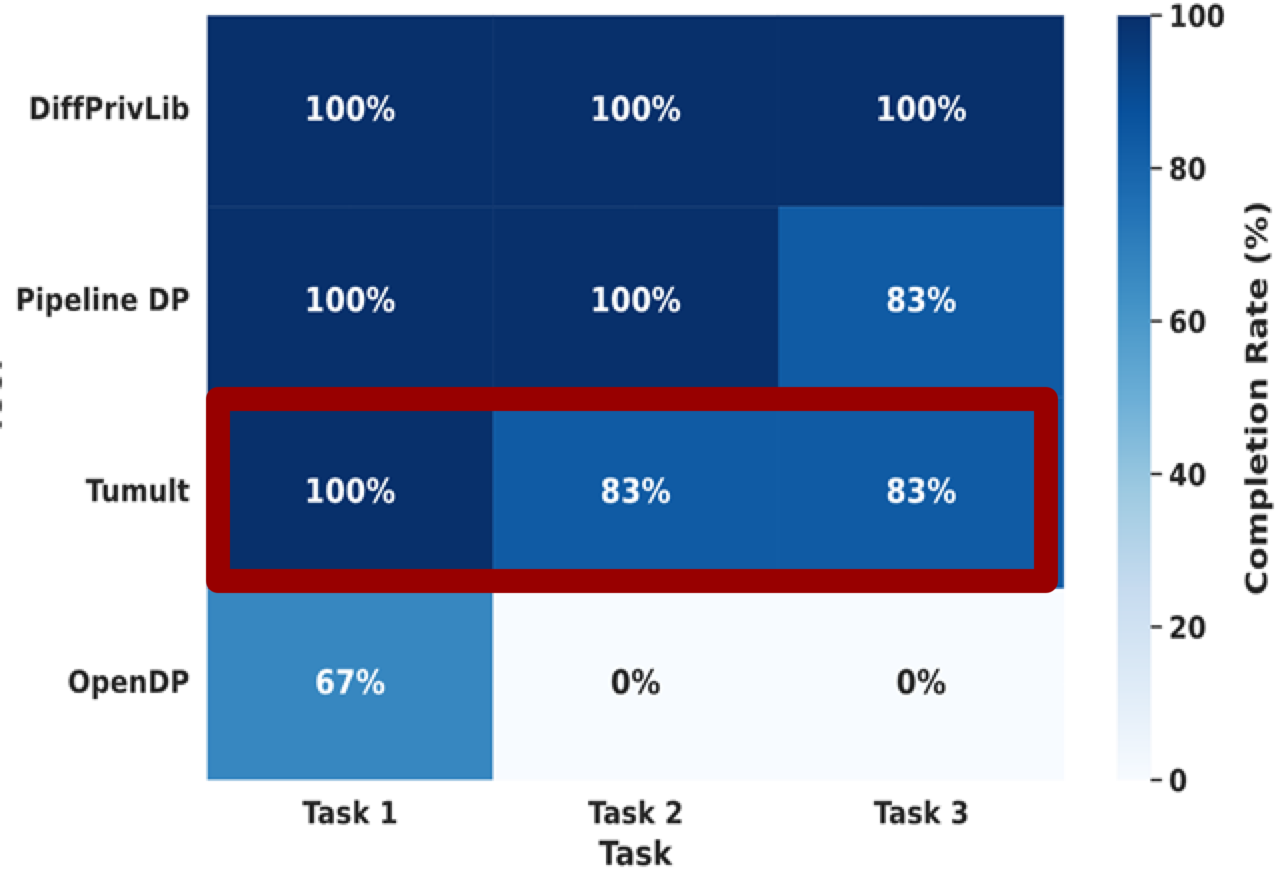


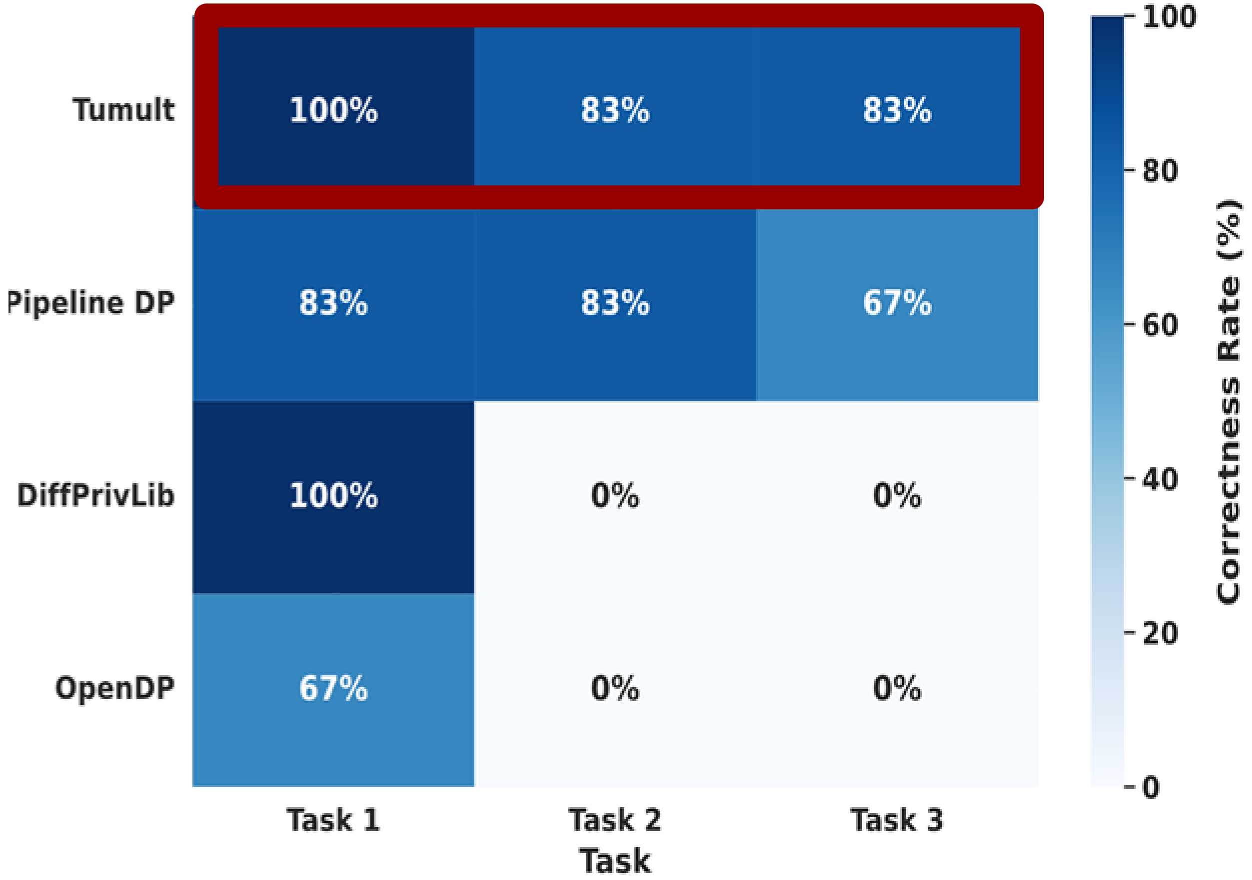



# OpenDP users completed fewest tasks, and all were correct



# Tumult Analytics users completed most tasks, and all were correct



# API designs impacted completion rates.

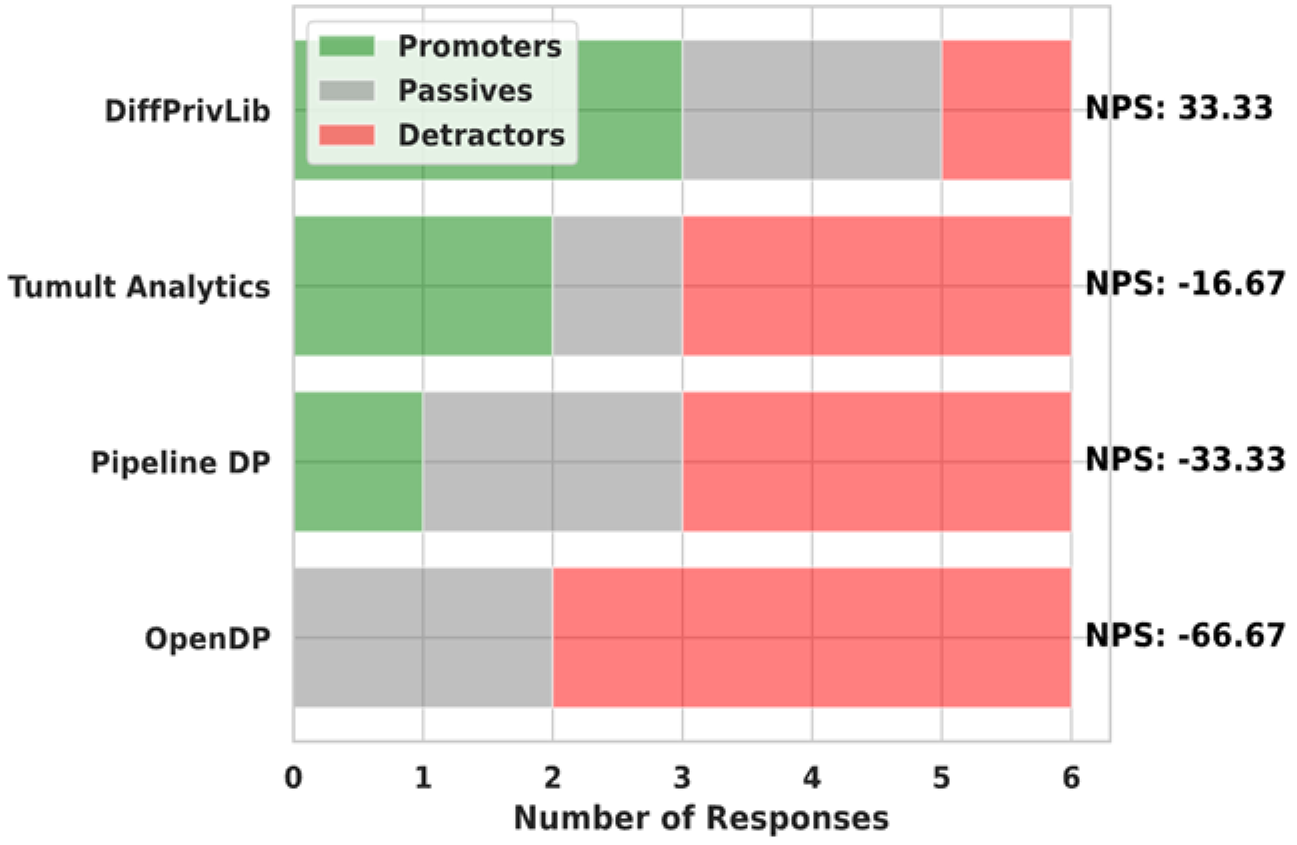
- DiffPrivLib uses a minimal API and works well with popular data analytics libraries like Pandas.
- Tumult Analytics mimics the Spark data analytics API.
- OpenDP requires understanding complex differential privacy details.

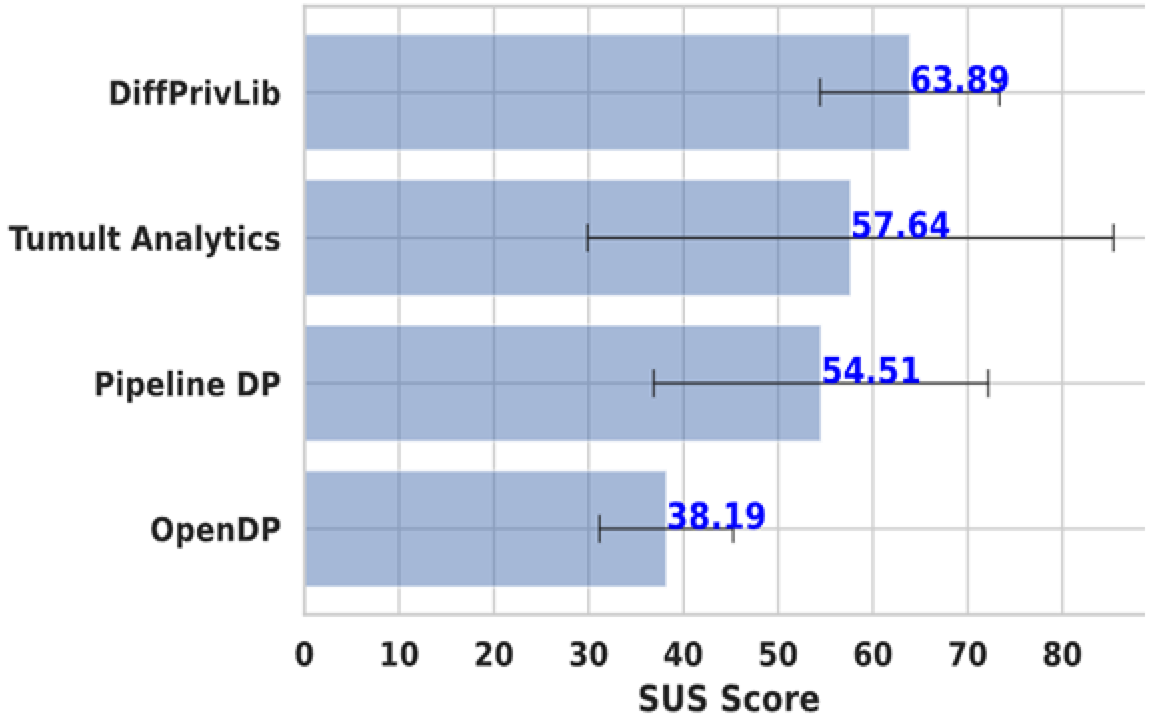
Participants preferred APIs similar to tools they already knew.

Tumult Analytics to Spark. "*I think the fact that it was very similar to Spark was really helpful,*" one expert participant (*E006*)

# Diffprivlib Leads in User Satisfaction but DP Tools Need Improvement

- Participants most satisfied with DiffPrivLib
- Participants least satisfied with OpenDP



# DP Violation vs Usability Tradeoff

**API Design** had an impact on all 3 parts we measured , and **it is possible to do well on all 3.**

| Tool | Prevented Violations? | Completion Rate | Satisfaction |
|---|---|---|---|
| **OpenDP** | Yes | Low | Low |
| **DiffPrivLib** | No | High | High |
| **PipelineDP** | No | Low | Low |
| **Tumult** | Yes | High | High |

# Make API Design Intuitive

- **Leverage familiarity with mainstream APIs** (e.g. Pandas, Spark)
  Participants appreciated tools that mimicked or used familiar APIs.

*"I think the fact that it was very similar to Spark was really helpful...I have a decent amount of experience with Spark and Pandas, so that was very intuitive."*

-Tumult Analytics

# Improve Error Prevention & Recovery

- **Warn users about DP violations**

  DiffPrivLib users did not realize they had violated DP.

- **Provide clear, actionable error messages**

  OpenDP users were confused by Rust-related implementation details.

```
OpenDPException                          Traceback (most recent call last)
<ipython-input-48-58830e6b5f15> in <cell line: 6>()
      4   return time_spent_total >> make_base_discrete_laplace(scale=scale, D=AtomDomain[int])
      5
----> 6 binary_search_chain(status_dp, d_in=1, d_out=budget)(raw_data)

                              2 frames
/usr/local/lib/python3.10/dist-packages/opendp/_lib.py in unwrap(result, type_)
    159
    160     # Rust stack traces follow from here:
--> 161     raise OpenDPException(variant, message, backtrace)
    162
    163

OpenDPException: Continued Rust stack trace:
    opendp_core__transformation_invoke
    opendp::core::Function<TI,TO>::make_chain::{{closure}}
    opendp::core::Function<TI,TO>::make_chain::{{closure}}
    opendp::core::Function<TI,TO>::make_chain::{{closure}}
    opendp::core::Function<TI,TO>::make_chain::{{closure}}
    opendp::core::Function<TI,TO>::make_chain::{{closure}}
    <opendp::core::Function<TI,TO> as opendp::ffi::any::IntoAnyFunctionExt>::into_any::{{closure}}
    opendp::transformations::dataframe::apply::make_apply_transformation_dataframe::{{closure}}
    opendp::data::Column::as_form
  FailedCast("tried to downcast to "Vec<i32>"")
```

*"I don’t really know any Rust. Coming from a Python experience, [it] might be better to have error messages in Python that indicate the error in the line of Python."* -OpenDP

# Provide Usable Documentation and DP Foundations

- **Provide searchable documentation with lots of examples**
  Users of all tools struggled with single-page documentation lacking examples.
  Provide advice on what to do, not just how to do it (e.g which mechanism would be the best choice?)

- **Help users understand how to set privacy parameters**
  (e.g. total privacy budget, $\epsilon$ per query, upper bound on data values, etc)

# Thank you!

API design impacts **correctness**, **completion**, and **satisfaction.**


## Takeaways

- Mimic existing APIs
- Raise errors for DP violations
- Provide usable documentation
- Help users with DP foundations

Presenter: **Ivoline Ngong**

Email: **kngongiv@uvm.edu**

**Paper:** https://arxiv.org/pdf/2309.13506

**Study Materials**: https://osf.io/ag2fj/?view_only=29a9bc2a30574befa9f3d0643951b9c6