**Lab 10: Azure Differential Privacy**

This lab will include setting up the environment, preparing the data, applying differential privacy to the data analysis, and understanding the impact of privacy settings on the results.

**Step 1: Environment Setup**

**Objective:** Install necessary Python packages to work with SmartNoise SQL.

**Instructions:**

1. Open your command prompt or terminal.
2. Run the following command to install the SmartNoise SQL package:

sql

pip install smartnoise-sql

**Step 2: Prepare the Dataset**

**Objective:** Load and prepare mock data for analysis.

**Instructions:**

1. Create a CSV file named **mockdata.csv** with the following columns: **age** and **diabetic**. Populate it with 1,000 records of random data.
2. Create a metadata file **mockdata.yaml** that describes the schema of **mockdata.csv**. Here’s a simple example of what it might look like:

OR JUST DOWNLOAD THE DATA FROM HERE:

<https://github.com/fenago/ai-security/tree/main/lab-guides>

table: mockdata columns: - name: age type: integer - name: diabetic type: boolean

1. Use Python to load and display the actual average age by diabetic status:

import pandas as pd

# Load the dataset data\_path = 'mockdata.csv'

mockdata = pd.read\_csv(data\_path)

# Calculate and display the actual average age

actualdata = mockdata[['age', 'diabetic']].groupby(['diabetic']).mean()

print(actualdata.to\_markdown())

**Step 3: Apply Differential Privacy**

**Objective:** Analyze the data using differential privacy to protect individual privacy.

**Instructions:**

1. Import necessary modules and set privacy parameters with different epsilon values for comparison.
2. Use the SmartNoise SQL API to query the data:

import snsql from snsql

import Privacy

import pandas as pd

# Privacy settings

privacy = Privacy(epsilon=0.05, delta=0.01)

# More privacy

privacy\_less = Privacy(epsilon=0.90, delta=0.01)

# Less privacy

# Load the dataset and metadata

csv\_path = 'mockdata.csv'

meta\_path = 'mockdata.yaml'

mockdata = pd.read\_csv(csv\_path)

# Reader for differential privacy queries

reader = snsql.from\_df(mockdata, privacy=privacy, metadata=meta\_path)

# Execute the privacy-preserving query

result = reader.execute("SELECT diabetic, AVG(age) AS age FROM mockdata GROUP BY diabetic")

print(result)

**Step 4: Analysis and Comparison**

**Objective:** Compare the results of analyses with different levels of privacy.

**Instructions:**

1. Run the query multiple times with different epsilon values (**0.05**, **0.90**).
2. Record the results in a table or graph to visually compare how the added noise affects the average ages reported for diabetics and non-diabetics.
3. Discuss the trade-offs between privacy (higher epsilon) and accuracy (lower epsilon).

**Step 5: Reflect on Privacy and Fairness**

**Objective:** Reflect on the implications of differential privacy on data utility and fairness.

**Instructions:**

1. Discuss how the use of differential privacy can impact the predictions of a model, especially when sensitive data affects the outcomes.
2. Consider scenarios where protecting privacy might lead to less accurate but more private results.
3. Explore how to balance the need for privacy with the requirement for fair and accurate data analysis.

By following these steps, participants in the lab will gain hands-on experience with differential privacy using the SmartNoise SQL library, understanding both its application and implications.