Differential Privacy

Privacy Preserving Information Access: Homework 2

Reza Ghasemi

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO MATEMATICA

A.Y. 2022/2023

Diabetes Dataset

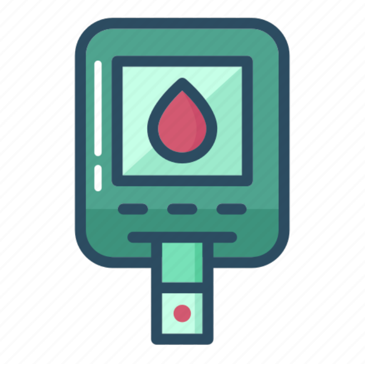
```
In [2]: diabetes_ds = pd.read_csv("diabetes.csv")

        # display dataset
        diabetes_ds
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

Context

- Used to predict whether a patient has diabetes or not.
- The dataset contains the following information regarding patients:
- **Pregnancy: number of pregnancies**
- **BMI: body mass index (weight in kg/(height in m)^2)**
- **Age: years**
- **Glucose : glucose concentration**
- **BloodPressure: Diastolic blood pressure (mm Hg)**
- **output: diagnosis of diabetes (0 or 1)**
- **...**
- Identifiers have been removed (e.g., name, SSN), making this dataset "anonymized."



DIPARTIMENTO MATEMATICA

Fake Data Generation

Università degli Studi di Padova

Fake information was added to make the dataset more interesting: name, SSN, and job.

In [4]:
```
name_list = []
ssn_list = []
job_list = []


# use italian locale
fake = Faker('it_IT')

# generate fake data for all entries
for _ in range(768):
    name_list.append(fake.name_female())
    ssn_list.append(fake.ssn())
    job_list.append(fake.job())

# add new columns to csv file
diabetes_ds['Name'] = name_list
diabetes_ds['SSN'] = ssn_list
diabetes_ds['Job'] = job_list

diabetes_ds = diabetes_ds[['Name', 'SSN', 'Age', 'Job', 'Pregnancies',
                           'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                           'BMI', 'DiabetesPedigreeFunction', 'Outcome']]

# save the dataset
diabetes_ds.to_csv('output.csv')
```

Dipartimento Matematica

Dataset with PII

```
In [5]: # read new dataset
        diabetes_dataset = pd.read_csv("output.csv")
        diabetes_dataset.head()
```

Out[5]:

Name	SSN	Age	Job	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Outcome
Virginia Benigni	PVNLLT00E15I692J	50	Surveyor, planning and development	6	148	72	35	0	33.6	0.627	1
Sig.ra Ludovica Costanzi	SLTGDI16C14G909X	31	Editor, magazine features	1	85	66	29	0	26.6	0.351	0
Germana Schiavone	LZUGBL37H03C463U	32	Hospital doctor	8	183	64	0	0	23.3	0.672	1
Sig.ra Tiziana Alboni	MSTRNL72S25E539D	21	Programme researcher, broadcasting/film/video	1	89	66	23	94	28.1	0.167	0
Sig.ra Licia Malenchini	NTTVCR69A12I928V	33	Geographical information systems officer	0	137	40	35	168	43.1	2.288	1

Università degli Studi di Padova
DIPARTIMENTO MATEMATICA

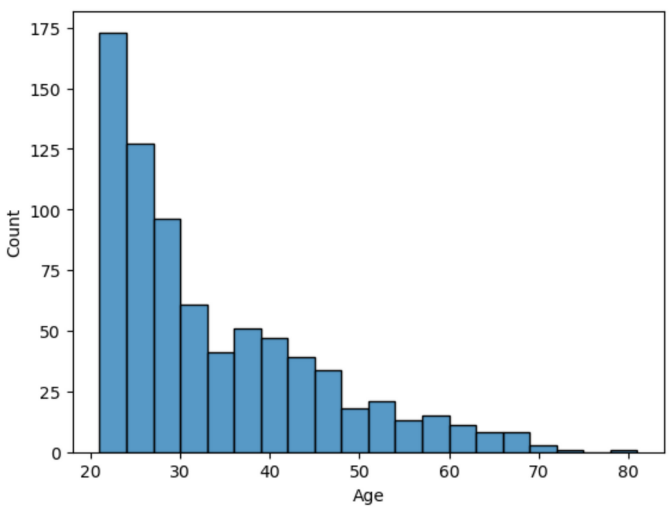
Histograms

Age:

Pregnancies:

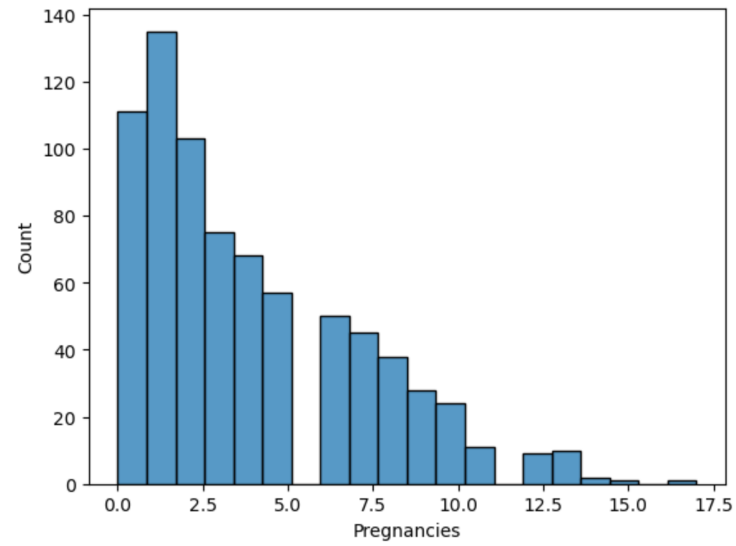
```
In [6]: sns.histplot(data=diabetes_dataset, x='Age', bins=20)
```

Out[6]: <AxesSubplot:xlabel='Age', ylabel='Count'>




```
In [7]: sns.histplot(data=diabetes_dataset, x='Pregnancies', bins=20)
```

Out[7]: <AxesSubplot:xlabel='Pregnancies', ylabel='Count'>



K-anonymity

Definition: A property of a dataset indicating its level of anonymity.

We say a dataset is k-anonymous when we have k-1 individuals with the same quasi-identifiers.

What is the best k value? Well, it depends...

```
In [9]: diabetes_ds = pd.read_csv("diabetes.csv")

        is_kanonymous(dataset=diabetes_ds, k_value=1)

Out[9]: True
```

For k-anonymity, two values **must be avoided**: k = 1 and k = n (table dimension).

```
In [14]: is_kanonymous(diabetes_ds, k_value=50)

Out[14]: False
```

We do not have 50-anonymity

Generalization

Suppression and **generalization** can be used to achieve k-anonymity.

In the following example, the generalization function is used to remove the second digit:

In [17]:
```
generalization_function(diabetes_ds, generalization_degree)
```

Out[17]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0	140	70	30	0	30	0	50	0
1	0	80	60	20	0	20	0	30	0
2	0	180	60	0	0	20	0	30	0
3	0	80	60	20	90	20	0	20	0
4	0	130	40	30	160	40	0	30	0
...	...	...	...	...	...	...	...	...	...
763	10	100	70	40	180	30	0	60	0
764	0	120	70	20	0	30	0	20	0
765	0	120	70	20	110	20	0	30	0
766	0	120	60	0	0	30	0	40	0
767	0	90	70	30	0	30	0	20	0

768 rows × 9 columns

Value is completely lost!

Generalization (cont.)

Did we achieve the desired level of privacy?

```
In [19]: is_kanonymous(generalization_function(diabetes_ds, generalization_degree), 2)
Out[19]: False
```

Keep on generalizing until we reach it...

```
generalization_function(diabetes_ds, generalization_degree)
```

Out[20]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
763	0	0	0	0	0	0	0	0	0
764	0	0	0	0	0	0	0	0	0
765	0	0	0	0	0	0	0	0	0
766	0	0	0	0	0	0	0	0	0
767	0	0	0	0	0	0	0	0	0

768 rows × 9 columns

```
In [21]: is_kanonymous(generalization_function(diabetes_ds, generalization_degree), 2)
Out[21]: True
```

We achieved it, but at what cost?

Distributional Queries

If we wanted to understand how many individuals in the dataset have a BMI over 50, we could do the following:

```
In [22]: diabetes_ds[diabetes_ds["Age"] >= 60].shape[0]
Out[22]: 32
```

Raw Results

How many individuals have been pregnant more than five times?

```
In [23]: diabetes_ds[diabetes_ds["Pregnancies"] > 5].shape[0]
Out[23]: 219
```

How can we protect patients?

Possible solution: Differential Privacy

DP: Laplace Mechanism

$$F(x) = \underbrace{f(x)}_{\text{Raw Results}} + \underbrace{\mathrm{Lap}\left(\frac{s}{\epsilon}\right)}_{\text{Added noise to achieve privacy}}$$

Raw Results

Added noise to achieve privacy

ε: Privacy Budget

S: Sensitivity

DIPARTIMENTO MATEMATICA

DP: Laplace Mechanism (cont.)

UNIVERSITÀ DEGLI STUDI DI PADOVA

A query that looks for users with a BMI greater than or equal to 60.

```
In [26]: diabetes_ds[diabetes_ds["BMI"] >= 60].shape[0]
Out[26]: 1
```

Only one person

If the value for epsilon is **large**, we will not be able to protect the privacy of individuals, and **the value will be close to original value**:

```
In [27]: # Laplace mechanism definition:
         # F(x) = f(x) + Lap(s/eps)
         # s: sensitivity
         # eps: privacy budget

         sensitivity = 1
         eps = 0.8

         diabetes_ds[diabetes_ds["BMI"] >= 60].shape[0] + npr.laplace(scale=sensitivity/eps)
Out[27]: 0.325684438035028
```

DIPARTIMENTO MATEMATICA

DP: Laplace Mechanism (cont.)

If the value for epsilon is **small**, we will **focus more on protecting the privacy of the user**, and **the returned value will be far from the original value**:

```
In [28]: sensitivity = 1
         eps = 0.001

         diabetes_ds[diabetes_ds["BMI"] >= 60].shape[0] + npr.laplace(scale=sensitivity/eps)

Out[28]: -220.89136497917102
```

Raw result + Laplace noise

Laplace Mechanism on Histograms

Università degli Studi di Padova

We create a histogram based on blood pressure, with the number of individuals as the count (e.g., 57 individuals with blood pressure equal to 70).

In [33]:
```
# reset dataset

diab_ds = pd.read_csv("output.csv")
```

In [99]:
```
# choose variable glucose and count how many individuals have the same values
hist_diabetes = diab_ds["BloodPressure"].value_counts()

# display
hist_diabetes.to_frame().head(11)
```

Out[99]:

	BloodPressure
70	57
74	52
78	45
68	45
72	44
64	43
80	40
76	39
60	37
0	35
62	34

Dipartimento Matematica

Laplace Mechanism on Histograms (cont.)

Università degli Studi di Padova

Laplace noise can be applied to histograms:

In [55]:
```
sensitivity = 1
eps = 0.03

lap_noise = lambda x: x + npr.laplace(scale=sensitivity/eps)

hist_noise_added = hist_diabetes.apply(lap_noise)

# noisy table
hist_noise_added.to_frame().head()
```

Out[55]:

	BloodPressure
70	30.907527
74	27.433998
78	48.338123
68	40.342932
72	59.949283

Values have been perturbed according to the privacy budget that we defined.

Dipartimento Matematica

Università degli Studi di Padova

Laplace noise can be applied to <u>contingency tables</u> as well. Using **pd.crosstab()**, multiple columns are joined to build the following table:

```
In [87]: contingency_table = pd.crosstab([diab_ds["Job"], diab_ds["Age"]], diab_ds['Outcome'])

         # display contingency table
         contingency_table
```

Out[87]:

Job	Age	Outcome 0	Outcome 1
Academic librarian	22	0	1
	36	0	1
	41	1	0
Accommodation manager	28	1	0
	70	0	1
...	...	...	...
Web designer	50	1	0
	60	0	1
Writer	21	1	0
	25	1	0
Youth worker	24	0	1

748 rows × 2 columns

The outcome determines whether or not a person has diabetes.(1 indicates diabetes; 0 indicates no diabetes).

DIPARTIMENTO MATEMATICA

Laplace Mechanism on Contingency Tables (cont.)

Similar to what we encountered before, **if the privacy budget is large**, the final outcome **will not be drastically different from the original value**:

In [93]:
```
# define privacy budget which is large
eps = 50

noisy_contingency_table = contingency_table.apply(lap_noise)

# display
noisy_contingency_table
```

Out[93]:

	Outcome	0	1
Job	**Age**		
Academic librarian	**22**	0.008824	1.004068
	36	0.008824	1.004068
	41	1.008824	0.004068
Accommodation manager	**28**	1.008824	0.004068
	70	0.008824	1.004068
...	**...**	...	...
Web designer	**50**	1.008824	0.004068
	60	0.008824	1.004068
Writer	**21**	1.008824	0.004068
	25	1.008824	0.004068
Youth worker	**24**	0.008824	1.004068

748 rows × 2 columns

Laplace Mechanism on Contingency Tables (cont.)

If the privacy budget is small, the end result **will be distant from the original value**:

In [94]:
```python
# privacy budget chosen to be small
eps = 0.0005

noisy_contingency_table = contingency_table.apply(lap_noise)

# display
noisy_contingency_table
```

Out[94]:

	Outcome	0	1
Job	**Age**		
Academic librarian	**22**	-624.156473	1646.68496
	36	-624.156473	1646.68496
	41	-623.156473	1645.68496
Accommodation manager	**28**	-623.156473	1645.68496
	70	-624.156473	1646.68496
...	**...**	...	...
Web designer	**50**	-623.156473	1645.68496
	60	-624.156473	1646.68496
Writer	**21**	-623.156473	1645.68496
	25	-623.156473	1645.68496
Youth worker	**24**	-624.156473	1646.68496

748 rows × 2 columns

Final Remarks

- Resistant to record linkage attacks
- Bugs in implementation are possible (due to <u>incorrect sensitivity calculation</u> or <u>adding the incorrect amount of noise</u>).

- Not suitable for every scenario (e.g., medical data is sensitive; a small privacy budget is needed. Simultaneously, too much noise will make diagnosis difficult for doctors.)