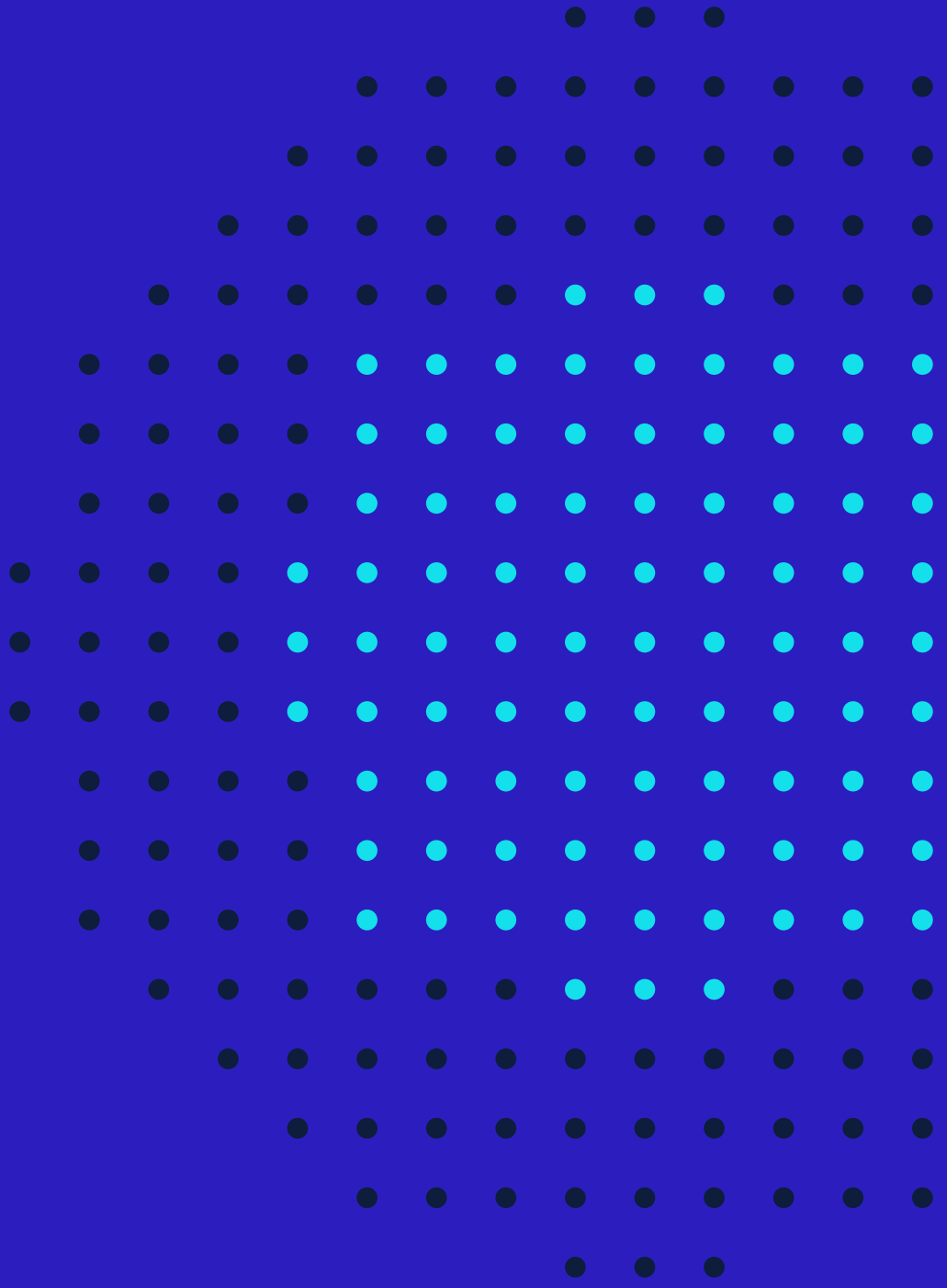
EBOOK

The Top 7 Techniques for De-identifying Data



PRIVITAR

When you want to ensure the safety of the data associated with your employees, patients, partners, or customers, you'll need to be able to anonymize or de-identify that data.

De-identifying or anonymizing their data is one of the best ways to ensure the safety of your customers, patients, partners, and employees. De-identification replaces raw data values with safer alternatives in places where unauthorized people could use sensitive data in your datasets to identify these data subjects.

Why and how you'll choose to de-identify data varies by use case. Using the right de-identification techniques for a particular use case will help you achieve multiple organizational goals:

- Minimize the risk of personal and business-sensitive information being leaked, breached, or used without authorization
- Increase safe use of data in a wider range of use cases, for a broader set of stakeholders
- Create confidence among customers by ensuring the privacy of their data
- Build trust among internal stakeholders by providing them safe and useful data
- Comply with data protection and privacy regulations, thereby avoiding fines and penalties

Consider how the seven techniques that follow would help you protect your sensitive data in data-driven use cases. Assess the pros and cons of each, and keep in mind that for each use case you may need to use multiple techniques in combination to achieve the desired result.

De-identification Techniques:

- Redaction
- Hashing
- Encryption
- Tokenization
- Generalization
- Substitution
- Perturbation

Example: Name

INPUT Thomas Anderson
REDACTION ********

Redaction

Redaction plays an important part in data minimization strategies by deleting any values that aren't necessary for the analysis—in short, if the data isn't needed, redact it.

Redaction removes an original value from a dataset, either completely or partially. Full redaction involves removing values entirely—for example, removing an attribute or column or replacing all values with a constant, such as "XXXXX" or "0." Deleting a name and a social security number is an easy first step in redacting information from any dataset.

You can also partially redact (also known as clipping or truncating) by deleting only part of a value. You will commonly see this applied to credit card numbers, where only the last four digits are retained.

Another form of redaction is full value masking. While replacing values with a constant is a weaker form of redaction than completely removing the field, it is useful in certain situations.

Pros: Redaction eliminates all noncritical data, so you won't inadvertently expose identifying information to risk while satisfying the query. Redaction is simple and has a very high impact in reducing privacy risk.

Cons: Used on the wrong fields, redaction can remove valuable information that is necessary for meaningful analysis. Also, as more datasets are combined, redaction might not be sufficient to protect all identifying information.

Hashing

Hashing converts an original value into a fixed-length output known as a "hash." Keep in mind that the hash cannot be converted back to the original value.

Pros: The ease of hashing records makes this technique an appealing, fast solution for de-identifying direct identifiers. If you don't need the reversibility of tokenization, hashing is an easy and less compute-intensive way to de-identify data.

Cons: Unlike encryption, hashing is a one-way operation, which cannot be reversed. Hashing can also be vulnerable to attack. Use it with caution. The best practice for minimizing the hash vulnerability is to use salted hashing where you add a random string (a "salt") to the value before it is hashed.

Example: Date

INPUT 2022-02-07

HASHING

FB078F7A9FA1D4A9D7C66BC07C5FFE2F3
0E4EA8C7A98C90A6077F02 A1DFC8A04

Example: Name

INPUT Thomas Anderson

HASHING

0ECABC84C19EBC0B06C7B618D4035551
5406A0BE8E416AB7C9CFC55 6948761BD

Encryption

Encryption uses algorithms to replace original values, referred to as "plaintext," with another value. This new value, called a "ciphertext," has no exploitable meaning. Only authorized users with a "key" can decrypt and access the original value. Without authorization and access to the key, you can't access the source data.

Pros: Encrypted data is less vulnerable to exposure in a data breach. You can store and share data with greater freedom once it has been encrypted.

Cons: Encryption helps protect against unauthorized access, but doesn't prevent misuse by people who are authorized. You also need to be aware that encryption can prove costly, as the systems to maintain the original data and manage keys can grow quickly.

Example: Date

INPUT 2022-02-07

ENCRYPTION:

rK1tAE0kFerT9oMhqU7zPyfTDYd8pJ5RwPM
i9YOOSs8=

Example: Name

INPUT Thomas Anderson

ENCRYPTION

gt7qzLTMwn81hd3bAQMrLe7kGzRN651F7XA
pIlJfzcc=

Tokenization

Tokenization is a technique where you replace an original value with a randomly generated equivalent, called a "token." The token has no exploitable meaning or value, but it acts as a reference that maps back to the original value should you need to unmask it in the future.

Tokens can conform to a specific format or length when needed. Generating a token with the correct formatting allows the tokenized data to pass a validation test. For example, Social Security numbers in the United States are always nine digits formatted as 000-00-0000.

Pros: Tokenization reduces privacy risk by replacing identifiable values from datasets. It also gives you the option to reverse the process, if needed. If you perform tokenization consistently, the data still retains some of its value for analytics. In general, tokenization is still suitable for machine learning.

Cons: Even if tokenization conforms to a specific format, it doesn't provide values you can aggregate for meaningful analysis. Tokenization may be more compute-intensive compared to the other forms of de-identification discussed here. You may also require data storage to map tokens back to their original values.

Example: SSA Number

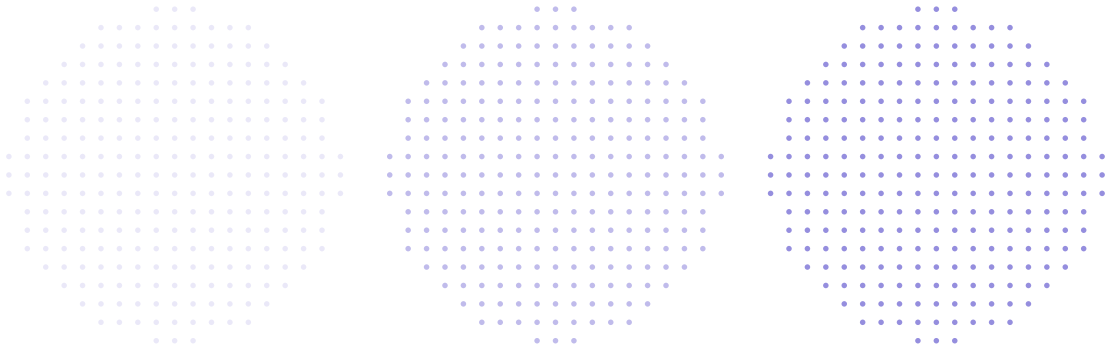
INPUT 01 2-34-5678

TOKENIZATION 234-56-7890

Example: Name

INPUT Thomas Anderson

TOKENIZATION oijpijn ddqwed



Generalization

Generalization transforms an original value into one that's more general, reducing the value's precision to protect privacy, while still retaining valuable information for analytics. It's used where a field or combination of fields might form a unique combination which could be used to single out an individual in a dataset, or linked to other background information to identify an individual.

There are various means to achieve generalization depending on the data in question:

- Binning changes a specific value into a range. For example, you can represent a $12 transaction by the range of $11 - $20.
- Rounding changes a specific value to a nearby round number. For example, you can round a $19 transaction to the nearest 10: $20.
- Clipping reduces the length of a specific value. For example, clipping a postal code to just the first three digits, or clipping a date to give just the month, not the day.

You can use generalization to achieve k-anonymity, a scientifically tested method to prove that individual data subjects can't be re-identified—if enough of the included records share the same identifying information.

Pros: Generalization hides the characteristics of individuals within a group while still providing values that can be aggregated, computed, and analyzed. In complex datasets, advanced generalization algorithms empower you to prioritize values where more precision is needed.

Cons: If you have a limited sample size, conclusions based on aggregated, generalized values might give less accurate results. Also, with a small sample size, it may be easier to identify individuals.

Example: Date

INPUT: 2022-02-07
GENERALIZATION: 2022

k-anonymity

A dataset is k-anonymous when the information for each person represented in the data can't be distinguished from at least k - 1 other individuals whose information is also in the dataset.

Substitution

Example: Name

INPUT: iPhone8

SUBSTITUTION: Phone

Substitution replaces an original value with another value from a predefined list. This can provide a form of generalization, where the substituted value is less precise than the original, and many values map to it. For example, you could substitute "iPhone" for both of the values "iPhoneX" and "iPhone8."

Pros: Substitution provides a convenient way to de-identify, and sometimes generalize the data using a customer-specific inventory of terms.

Cons: As is the case with generalization, replacing specific data with something more generic might leave data that is not precise enough for meaningful analysis. It may also lead to biases that yield invalid or erroneous conclusions. Preparing and managing the substitution lists is time-consuming and labor-intensive—and it may introduce the risk of re-identification if the substitution lists leak out.

Perturbation

Example: Date

INPUT: 2022-02-07

PERTURBATION: 2022-02-12

Perturbation adds random "noise" to an original value. It protects data against privacy attacks that rely on knowledge of specific values, such as numeric values, dates, and timestamps. As an example, you could perturb a transaction value of $182 by $5 on either side to generate an output value in the range of $177–$187.

Pros: Perturbation prevents bad actors from deriving identifying information from a given value, while still producing values you can aggregate and analyze to find insight.

Cons: If the degree of noise you add to a dataset is too high, you might lose valuable insight. Perturbation could introduce bias that gives you results substantially different from what the original values would provide. Perturbation isn't an effective technique for protecting data with uneven distribution, like a dataset with a value (an outlier) that falls outside the range of the majority of the data.

Choosing the Right Solution for You

Each of these de-identification techniques has its uses, and some have distinct advantages, but none is likely to provide the perfect solution by itself. A single technique or policy on its own is rarely adequate to protect against re-identification. In most cases, the best solution combines techniques in ways that maintain the balance between data utility and data privacy.

To find out more about advanced techniques and technologies to address your data privacy needs contact us.

About Privitar

Privitar empowers organizations to use their data safely and ethically. Our modern data provisioning solution builds collaborative workflows and policy-based data protection into data operations. Only Privitar has the right combination of technology, domain expertise, and best practices to support data-driven innovation while navigating regulations and protecting customer trust.

For more information, visit www.privitar.com.

PRIVITAR

Reference: P1006-EB-Top-7-De-identification-Techniques-DP.