Data Science made in Switzerland

Ein Blog der ZHAW Zürcher Hochschule für Angewandte Wissenschaften

Data Anonymization

Posted on **5. March 2014** by **Marc Rennhard**

I'm glad that Thilo mentioned Security & Privacy as part of the data science skill set in his recent blog post. In my opinion, the two most interesting questions with respect to security & privacy in data science are the following:

- *Data science for security*: How can data science be used to make security-relevant statements, e.g. predicting possible large scale cyber attacks based on analysing communication patterns?
- *Privacy for data science*: how can data that contains personal identifiable information (PII) be anonymized before providing them to the data scientists for analysis, such that the analyst cannot link data back to individuals? This is typically identified with data anonymization.

This post deals with the second question. I'll first show why obvious approaches to anonymize data typically don't offer true anonymity and will then introduce two approaches that provide better protection.

Removing or Transforming Data

Assume we have a database that contains data about individuals including their name, gender, birth date, zip code, income, drinking habits and health problems. An obvious anonymization technique is to simply remove (this is also named suppression) all names and release the database to the public for analysis. Besides removing the names, they could also be transformed (e.g. by encryption or keyed hashing) such that without knowledge of the key, the original data cannot be uncovered. The problem with removing or transforming data is that if there exists another publicly available database that contains names, genders, birth dates and zip codes, it is relatively easy to de-anonymize the data in most cases. For instance, studies have shown that it is possible to uniquely identify more than 60% of all North Americans based on birth date, gender, and zip code [1]. This de-anonymization is not just theoretical, as the following two examples illustrate:

- In 1997, the Massachusetts Group Insurance Commission released “anonymized” data that showed health diagnoses and prescriptions of state employees. This motivated a student to buy a voter registration list, which allowed her to de-anonymize large portions of the data, including the data of the Governor of Massachusetts at that time. The student couldn’t resist to send the Governor’s health record to his office [2].
- In 2006, Netflix started an open competition with the goal to find algorithms that allow to predict user ratings for films. As a basis, Netflix provided a large data set of user ratings as training data, where both users and movies were replaced by nummerical IDs. By correlating this data with ratings from the Internet Movie Database, two researchers demonstrated that it is possible to de-anonymize users [3].

These examples illustrate that removing or transforming data has its limitations if an attacker has access to additional data. Of course, the more data is removed or transformed in the original dataset, the “higher” the expected anonymity, but this also reduces the usefulness of the data for the data scientist.

To overcome these limitations, further techniques for de-anonymization have been proposed with the goal to provide scientific (provable) anonymity guarantees. Two of them, k-anonymity and differential privacy are described below.

k-Anonymity

k-anonymity was proposed in 2002 with the goal to modify data in way such that a specific person’s data cannot be distinguished from at least k-1 persons in the same dataset [2]. This can be achieved with suppression (replace attributes or parts of them with *) or generalization (replace attributes with broader categories, e.g. replace age 42 with age 40-49).

As an example, consider the following table from a health study:

NAME	AGE	GENDER	ZIP CODE	DISEASE
Angela	55	Female	44320	Heart-related
Bill	63	Male	44312	Heart-related
Cindy	33	Female	44485	Viral infection
Daniel	35	Male	44340	Cancer
Edwin	68	Male	44346	Heart-related
George	43	Male	44349	No illness
Mandy	59	Female	44387	No illness
Paula	48	Female	44414	Cancer

Our goal is now to achieve 2-anonymity with respect to the attributes name, age, gender, and zip code. Obviously, we could simply suppress all values, but this would also significantly limit the value of the resulting data. Instead, we use a combination of suppression and generalisation with the goal to keep as much information as possible. The following table shows one way how this could be done:

NAME	AGE	GENDER	ZIP CODE	DISEASE
*	50-59	Female	443*	Heart-related
*	60-69	Male	443*	Heart-related
*	<50	Female	444*	Viral infection
*	<50	Male	443*	Cancer
*	60-69	Male	443*	Heart-related
*	<50	Male	443*	No illness
*	50-59	Female	443*	No illness
*	<50	Female	444*	Cancer

Inspecting the table shows that there are at least two datasets for any combination of the first four attributes, meaning 2-anonymity is achieved.

So what does this help? Assume that an attacker somehow knows that Angela is included in the dataset and he knows her other attributes (age, gender, zip code). Looking at the anonymized data, he can identify two records (the 1st and the 7th) that may represent Angela, but he cannot conclude which one. And consequently he also cannot conclude whether Angela suffers from a heart-related disease or is healthy. Naturally, higher values of k also imply higher anonymity and in general, the larger the number of records in a dataset, the higher k can be chosen without rendering the anonymized data useless for further analyses.

Since the original paper by Sweeney [2], several further research on k-anonymity has been conducted, including analyses of the complexity of optimal k-anonymity, algorithms to achieve (nearly optimal) k-anonymity in a given dataset, and possible attacks against k-anonymity. Attacks are for instance possible if the diversity of the non-anonymized attributes is low. In the example above, all males with age between 60 and 69 suffer from heart-related diseases. So if an attacker knows that Bill is included in the dataset, he directly learns about his disease.

Differential Privacy

The approaches described above are intended for non-interactive settings: the data owner anonymizes the data and releases them for further analyses. When performing analyses on the sanitized data, no interaction with the original data takes place. In interactive settings, the analyst directly queries the original data. If we want to avoid PII from leaking to the analyst in such a setting, the returned results must be sanitized. This is typically achieved by distorting the results by adding noise.

One way to achieve this is with differential privacy, originally published in 2006 [4]. The basic idea is that accessing a database should not allow anyone to learn anything about an individual if her data is in the database that he couldn't learn if her data were not in the database. Or more informally: The data of any individual in the database should not (significantly) affect the the outcome of the result.

Let's consider a database that contains several persons including whether they have liver problems (0=no, 1=yes) and their number of doctor's visits in 2013 (0-5, we assume 5 is the maximum).

NAME	HAS LIVER PROBLEMS	# DOCTOR'S VISITS IN 2013
Angela	0	3
Bill	1	4
Cindy	1	1
Daniel	0	0
Edwin	1	2
George	0	0
Mandy	1	5
Paula	0	1

Now assume that it is allowed to query the database for the total number of persons with liver problems. If the analyst somehow knows the individuals in the database and he knows of all persons except Mandy whether they have liver problems, he can easily abuse this to learn about Mandy' liver condition: If the query returns 3, Mandy has no liver problems; if it returns 4, she does.

The goal of differential privacy is to prevent such information leakage. With differential privacy, the query does not simply return 4, but a value "close to 4", e.g. 2.8, 3.5 or 4.3. As a result, the analyst (even if the returned result is >4) cannot conclude that Mandy has a liver condition or not and likewise, Mandy can plausibly deny having a liver condition.

For details about the math refer to the original paper, but the idea how noise is created is as follows:

1. Determine by how much the answer of an individual can influence the result. Using the liver problems example, this value is 1 because Mandy can influence the total at most by one.
2. To determine the noise to add to a specific result, us a random value from a Laplacian distribution (which uses the value (1) above as a parameter).
3. As the mean of the Laplacian distribution is 0, the noise can be negative or positive, resulting in a distorted result that is smaller or larger than the correct result.

Now consider a second query, which delivers the total number of doctor's visits in 2013. In this case, an individual can influence the total at most by a value of 5 and consequently, 5 is used as a parameter in the Laplacian distribution. The larger this parameter, the larger the standard deviation (the mean is always 0) of the Laplacian distribution and consequently, the larger the expected noise.

You may argue that the noise will significantly disturb the result as in the example above, the number of actual doctor's visits (16) could easily be reported as 8 or 25 after adding noise. But with large datasets (e.g. 1000s of entries), the noise is relatively insignificant compared to the actual result. But of course, the result will always be somewhat affected by the noise – that's the price of privacy.

Just like with k-anonymity, lots of research has been conducted since the idea of differential privacy was originally published [4]. This includes proposals to address various problems that arise in practice (e.g. how to deal with an analyst that executes the same query again and again with the goal to determine the correct result as the noise cancels out over time), analyses of the usefulness of the noisy results, and extensions to the original proposals that use less noise while losing some privacy.

Conclusions

The amount of research and published papers on data anonymization demonstrate that the problem is both relevant and challenging. It's hard to tell in which direction privacy will develop in general but the reports about large scale global surveillance programs that became public in 2013 may result in more privacy-awareness of users, companies and (some) governments. Independent of future developments and coming back to Thilo's data science skill set in his recent blog post: data scientists should indeed be aware of privacy and the technologies described above can provide starting points to get more familiar with the topic.

[1] Philippe Golle. Revisiting the Uniqueness of Simple Demographics in the US Population. In Proceedings of the 5th ACM Workshop on Privacy in the Electronic Society, 2006.
[2] L. Sweeney. k-Anonymity: A Model for Protecting Privacy. In International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10 (5), 2002.
[3] Arvind Narayanan and Vitaly Shmatikov. Robust De-anonymization of Large Sparse Datasets. In Proceedings of the 29th IEEE Symposium on Security and Privacy, 2008.
[4] Cynthia Dwork. Differential Privacy. In 33rd International Colloquium on Automata, Languages and Programming, part II, 2006.

This entry was posted in **Blog**, **Research** and tagged **Data Anonymization**, **Security & Privacy** by **Marc Rennhard**. Bookmark the **permalink [https://blog.zhaw.ch/datascience/data-anonymization/]** .