16-124-836
Marcel Matti ZAUDER

6.1 *k*-Anonymity and and *l*-Diversity

6.1.1 Distinct *l*-Diversity

For the distinct *l*-Diversity it is trivial to show that *l*-divers sanitized dataset also satisfies *l*-Anonymity. *l*-Diversity says that every subset contains at least *l* different values of the sensitive attributes. Because this is a given each subset must at least contain *l* different entries in order to satisfy this statement. This is of course the definition of *l*-Anonymity which is therefore also satisfied.

6.1.2 Probabilistic *l*-Diversity

The definition of probabilistic *l*-Diversity is that any value has a relative frequency of at most $\frac{1}{l}$. Therefore if a sanitized dataset fulfills this requirement each subset will have at least *l* entries, which is also satisfying the *l*-Anonymity criteria.

6.2 Implementing *k*-Anonymity with the Mondrian Algorithm

6.2.a Normalized Certainty Penalty

Using only PLZ and points as QI (and represented as numerical attributes), and with system as S, compute at least a 3-, 5-, and 10-anonymization of the dataset and report its NCP.
What is the NCP of a 1-anonymization and that of a 74-anonymization?

For the different k-anonymizations we get the NCP values:

(1) 3-Anonymity: 16,40%
(2) 5-Anonymity: 26,49%
(3) 10-Anonymity: 48,08%

The NCP of a 1-anonymization would be 0,00 % and for 74-anonymization the NCP would be 100,00%.

6.2.b Normalized Certainty Penalty of Permutations

Permute the dataset randomly (e.g., calling shuf) and observe the outcome. Extend the algorithm (using randomization) to compute improved 3-, 5-, and 10-anonymizations, that is, achieving better NCP than under a).

When shuffling the data beforehand the NPCs change in a range of $\pm$*a couple of%*.
In order to automate the whole process a new class was created with the following code:

```
import os

if __name__ == '__main__':
    for i in range(10):
        os.system("python anonymizer.py a d 5 | findstr NCP") # for windows
        os.system("python anonymizer.py a d 5 | grep NCP") # for linux
```

With this command we can run the anonymizer algorithm several times and get the lowest NCP. The shuffling of the data is done within the read_dataset script:

```
def read_data():
        data = []
        data_file = open('data/ex06_fake_dataset.csv', 'rU')
        ########################
        # parseFile into data #
        ########################
        random.shuffle(data)
        return data
```

This is only possible if we use "Ort" as a QI instead of the PLZ.