Classifier behavior on k-anonymous datasets

Project report

Tanja Šarčević

TU Wien

November 19, 2018

1 Introduction

Today a large amount of data is collected, especially online, for different recommendation systems, personalizing websites, or some useful statistics. Often, it includes personal information, bank accounts, medical records and so on. For any statistical analysis the data is the key ingredient, however, individuals' privacy can be compromised. Netflix Prize contest from 2007 is a famous example of how customer privacy can be threatened without any identifiers, just by matching two related datasets.

The idea behind k-anonymity is to make sure that it is impossible, by using any background information the attacker might have about the individual and the data from the dataset, to identify that individual with 100% certainty within some dataset. Anonymization techniques such as k-anonymity, as well as it's modifications l-diversity and t-closeness provide individuals' anonymity in datasets, but can significantly disturb the performance of machine learning algorithms. Therefore, the cost of privacy is paid by the risk of significantly lower insights.

In previous work of Malle et al. [1] a series of experiments applying different algorithms to a binary classification problem under anonymization and perturbation is introduced. In the latter paper [2] those experiments were extended by multi-class classification. The work in this project is highly based on work from the latter paper. We tried to recreate the experiments by implementing and applying the same anonymization algorithm and Machine Learning pipeline. Additionally, we introduced some related experiments with other available anonymization tools and compared the results for the better and broader insights.

2 K-Anonymity

K-anonymity is a property of a database to contain at least *k-1* duplicate entries for every occurring combination of attributes. Before going further into details of k-anonymity, we will introduce three different categories of data we have when dealing with tabular data, according to [2].

- **Identifiers** directly reveal the identity of a person without having further analysis of the data. Examples are first and last names, email address or social security number. As personal identifiers are hard to generalize in a meaningful way, those columns are usually removed.
- **Sensitive data** is a crucial information for statisticians or researchers and can therefore not be erased or perturbed; such data usually remains untarnished withing the released dataset. The example is the target attribute in the classification tasks performed within this project.
- **Quasi-identifiers** do not directly identify a person (for example, age), but can be used in combination to restrict possibilities to such a degree that a specific identity follows logically. On the other hand, this information might hold significant information for the purpose of research. Therefore, we generalize this kind of information, which means to lower its level of granularity. As an example, one could generalize specific age, eg. 38, to a range of values, eg. 30-40.

One can imagine k-anonymization as a clustering problem with each cluster's quasi-identifier state being identical for every data point it contains. This can be achieved by suppression and generalization, where by suppression we mean simple deletion, whereas generalization refers to a decrease in a value's granularity. Generalization works through a concept called *generalization hierarchies*, which run from leaf nodes denoting particular values via internal nodes to their most general root. In the generalization process for k-anonymity, one traverses the tree from a leaf node of the original input value upwards until we can construct an equivalence group with all quasi-identifiers being duplicates of one another. As each level of generalization invokes an increasing loss of specificity, we want to minimize a dataset's overall information loss. This makes k-anonymization an NP-hard problem due to an exponential number of possible data-row combinations one can examine.

3 Software and Methodology

3.1 Anonymization Algorithm

SaNGreeA In the paper by Malle et al. [2] experimentation is done using a version of a greedy clustering algorithm called SaNGreeA (Social network greedy clustering), [3] in JavaScript. For this experiment, the same algorithm is

implemented in Python3 [4] using the fragments of code from repository "Machine learning for health informatics" at TU Wien [5]. SaNGreeA consists of two strategies for tabular as well as network anonymization, with two respective metrics for information loss. The *Generalization Information Loss* or *GIL* consists of a categorical and continuous part. Categorical part measures the distance of a generalization level from its original leaf node in the generalization hierarchy, while the continuous part measures the range of a continuous-valued generalization divided by the whole range of the respective attribute.

$$GIL(cl) = |cl| \cdot (\sum_{j=1}^{s} \frac{size(gen(cl)[N_j])}{size(min_{x \in N}(X[N_j]), max_{x \in N}(X[N_j]))}) + \sum_{j=1}^{t} \frac{height(A(gen(cl)[C_j]))}{height(H_{C_j})}$$

where:

- $|cl|$ denotes the cluster cl's cardinality
- $size([i1, i2])$ is the size of the interval $[i1, i2]$, i.e., $(i2 - i1)$
- $A(w), w \in H_{C_j}$ is the sub-hierarchy of H_{C_j} rooted in w
- $height(H_{C_j})$ denotes the height of the tree hierarchy H_{C_j}

The total generalization information loss is then given by:

$$GIL(G, S) = \sum_{j=1}^{v} GIL(cl_j)$$

For the networking part of this algorithm the measure called *structural information loss* or *SIL* is introduced, however since we do not need this measure in this experiment, the details on the mathematical definition will be skipped. The reader is kindly referred to the original paper [3] for those details.

SaNGreeA is a Social Network Greedy Anonymization algorithm based on the concept of greedy clustering. Normally, its input is given in the form of a graph structure, but since for the experimentation we don't use the networking part of the algorithm, the input is given as a list of feature vectors (in a CVS file). In order to compute its clusters, SaNGreeA takes into account GIL function, which measures the degree to which the features of a cluster would have to be generalized in order to incorporate a new node. Implementation of the algorithm also takes as an input the generalization hierarchies for each of the categorical features in form of JSON files. Algorithm 1 on the following page is showing the high-level SaNGreeA algorithm.

```
Data: list of feature vectors, generalization hierarchies files, k
      anonymization factor, weights vector
Result: k-anonymous list of feature vectors
reading input;
for each unassigned feature vector do
    if number of unassigned feature vectors smaller than k then
        break;
    end
    initialize a new cluster;
    mark feature vector as assigned to a new cluster;
    while size of the cluster smaller than k do
        for each unassigned candidate feature vector do
            compute GIL(candidate vector, cluster);
            update best cost;
        end
        add candidate with best cost to the current node;
        mark candidate as assigned to the current node;
    end
end
for each unassigned feature vectors do
    for each cluster do
        compute GIL(vector, cluster);
        update best cost;
    end
    add feature vector to the best cluster;
    mark the vector as assigned to the cluster;
end
```

Algorithm 1: SaNGreeA

The algorithm starts with the first feature vector from the list and initiates the node with it. Subsequently, it individually adds k-1 feature vectors to the cluster based on GIL value which is checked for each of the unassigned features and the current cluster. When cluster reaches k elements, the process continues with initializing a new cluster with the next unassigned feature vector. When there are less than k unassigned feature vectors left, they are dispersed among the existing clusters. The choice of clusters is also decided by calculating GIL. Another interesting element of the algorithm are the weights that are giving the importance to attributes in the generalization process. GIL of each attribute is simply multiplied by attribute's weight, meaning that the larger weight we set for the attribute, more we prefer that this attribute stays ungeneralized over the others. The main drawback of this algorithm is its reduced algorithmic performance of $O(n^2)$ which for this experiments took up to 3 hours for one run.

Figure 1: Lattice diagram of possible generalization solutions, with the optimal one in the yellow square.

ARX ARX [6] is an open-source software for anonymizing personal data. The user can define their own hierarchies for quasi-identifiers, it contains the interface for comparing non- with anonymized data, as well as the solution space in lattice graph (figure 1), statistical information, and much more features. It supports many privacy models including k-anonymity, l-diversity, t-closeness, differential privacy, ... The algorithm used for k-anonymization is called *Flash* [7] and is much faster than our implemented algorithm.

3.2 Data

As input data, the training set of the Adults dataset from the UCI Machine Learning repository [8] is used. Every one of the following modifications of the original Adult dataset is the same as in the work from paper [2], therefore we work on the same input data. The dataset contains 15 columns, and all but one were used for experimentation (column education is excluded since it contains duplicate data from education-num, only as categorical values). There are approximately 32 000 entries in the original dataset, however, for the experiments we use only rows without missing values, 30 162 of them. Another modification of the original adult dataset is made; the distribution of the values in the column *native-country* is dominated by the value *United-States* shown by the fact that *United-States* is in 91.19% of all the rows, and the remaining 8.81% of the rows have some of the other 40 values. Therefore, the values for the column *native-country* are changed to *United-States* and *non-US*. The distribution of values for every column of the modified dataset is shown in figure 2. According to the distributions, our dataset consists mainly of white males under 50 from the US.

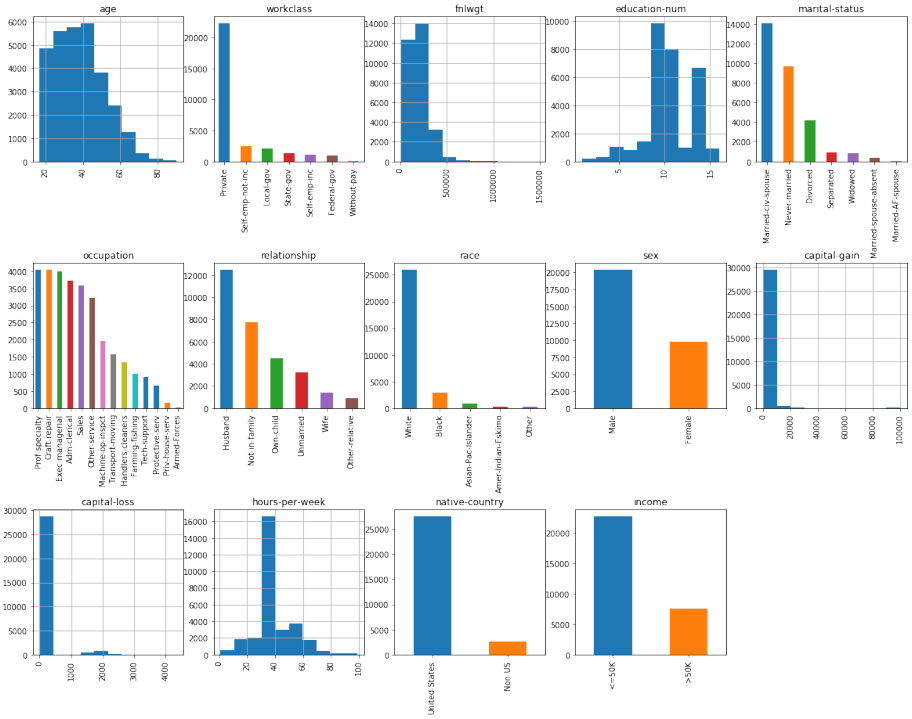

Figure 2: Distribution for each of the attributes in modified Adult dataset.

3.3 Classification

Classification is done using classifiers from Python package, scikit-learn. For the experiments with ARX, we also used classifiers from Weka[1] and RapidMiner[2] in order to obtain better results.

4 Experiments

For the experimentation, there were two main goals: reproducing the results from section 4.2. Anonymized Datasets from the paper [2] and the observation of algorithmic performance for multiple settings both for SaNGreeA and ARX.

For the purpose of reproducing the results, we performed the same experiments as stated in the paper [2]. We anonymize the Adult dataset with a range of values of k, $k \in \{3, 7, 11, 15, 19, 23, 27, 31, 35, 100\}$. We are examining multi-class classification performance with two different targets, 'marital-status' and 'education-num'. For 'marital-status' we left the 7 categorical values in the original dataset unchanged, whereas we grouped the 16 continuous 'education-num' levels into the 4 groups 'elementary-school', 'high-school', 'college-up-to-Bachelors' and 'advanced-studies'. Furthermore, to generate k-anonymized datasets we used each of these settings with three different weight vectors: (1) equal weights for all attributes, (2) age information preferred ($\omega(age) = 0.88, \omega(other\_attributes) = 0.01$) and (3) race information preferred ($\omega(race) = 0.88, \omega(other\_attributes) = 0.01$). We ran four different classifiers on the resulting data and computed their respective F1 score. The 4 classifiers used were *gradient boosting*, *random forest*, *logistic regression* and *linear SVC*. As each of the used classifiers were from scikit-learn, they all required some preprocessing as scikit-learn does not support categorical data for those classifiers. For *gradient boosting* and *random forest* we used scikit-learn's LabelEncoder, and for the other two classifiers, *logistic regression* and *linear SVC* we performed one-hot encoding.

In the experiments with anonymous data obtained by ARX we follow the same steps as stated above (the same choice of k-values, target attributes, classification tasks and classifiers). Instead of using only scikit-learn, we experimented with three different classification tools: scikit-learn's implementation of Logistic Regression and Linear SVC, Weka for Random Forest and RapidMiner for Gradient Boosting. Furthermore, experiments are done with different ways of preprocessing data as well (eg. using number-encoded values or binary features for classifiers that accept only numerical data, etc.). With ARX we had two options for the way of anonymizing the dataset: (1) Global Transformation - all values from a single column are anonymized to the same level; this results in many unnecessarily anonymized values (table 2), and (2) Local transformation - data is divided into clusters to obtain the smallest possible level of anonymization (table 3); this results in values of the same column anonymized to different

[1] https://www.cs.waikato.ac.nz/ml/weka/
[2] https://rapidminer.com/

age	workclass	occupation	race	sex	hours-per-week	native-country
38	Private	Craft-repair	White	Female	40	United-States
31	Private	Adm-clerical	White	Female	40	United-States
30	Private	Transport-moving	White	Male	40	United-States
18	Local-gov	Prof-specialty	White	Female	60	United-States
32	Private	Sales	Other	Male	40	Non-US
54	Private	Adm-clerical	White	Male	60	United-States
27	Private	Prof-specialty	White	Male	40	United-States
22	Private	Other-service	White	Male	40	United-States
23	Private	Exec-managerial	White	Male	40	United-States

Table 1: A sample from Adult dataset

age	workclass	occupation	race	sex	hours-per-week	native-country
*	*	*	*	Female	*	*
*	*	*	*	Female	*	*
*	*	*	*	Female	*	*
*	*	*	*	Male	*	*
*	*	*	*	Male	*	*
*	*	*	*	Male	*	*
*	*	*	*	Male	*	*
*	*	*	*	Male	*	*
*	*	*	*	Male	*	*

Table 2: 3-anonimous dataset sample obtained by global transformation

levels. Tables 2 and 3 show the globally and locally 3-anonymized datasets from the original dataset sample in table 1, respectively. As seen from this example, less represented values (for example, 'Non-US' or 'Local-gov') lead to high anonymization levels for the attribute if a global transformation is applied, and this leads to great loss of useful data. Local transformation keeps more information in the anonymized dataset while obtaining the same anonymity level for the dataset; both datasets in tables 2 and 3 are 3-anonymous, however with an obvious difference in the amount of information preserved.

age	workclass	occupation	race	sex	hours-per-week	native-country
30s	Private	*	White	*	40	United-States
30s	Private	*	White	*	40	United-States
30s	Private	*	White	*	40	United-States
*	*	Non-Technical	*	*	*	*
*	*	Non-Technical	*	*	*	*
*	*	Non-Technical	*	*	*	*
20s	Private	Non-Technical	White	Male	40	United-States
20s	Private	Non-Technical	White	Male	40	United-States
20s	Private	Non-Technical	White	Male	40	United-States

Table 3: 3-anonimous dataset sample obtained by local transformation

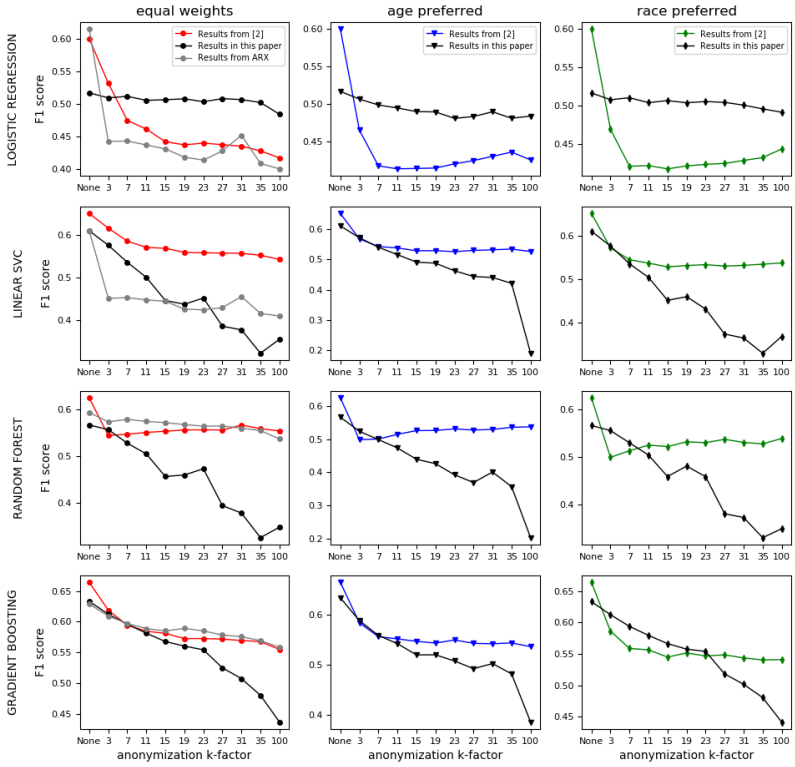

Figure 3: Multi-class classification on target *education-num* on the adult dataset under several degrees of k-anoynmity.

5 Results

As mentioned in section 4, we have done multi-class classification on two different targets: *education-num* and *marital-status*. Figure 3 shows results for target *education-num*. The results from the paper [2] are included in order to compare them to results in our experiments. For the target *education-num* and classifier Logistic Regression, we obtained slightly better results for smaller values of k, even though the main behaviour remained similar: the performance measured by F1 score gradually decreases as k increases. It is interesting to observe that changing the weights of the attributes in our case does not result in such as fast decrease of the performance as it is the case for the results in the paper. In other cases, results seem to be close to those in the original ones, except for larger

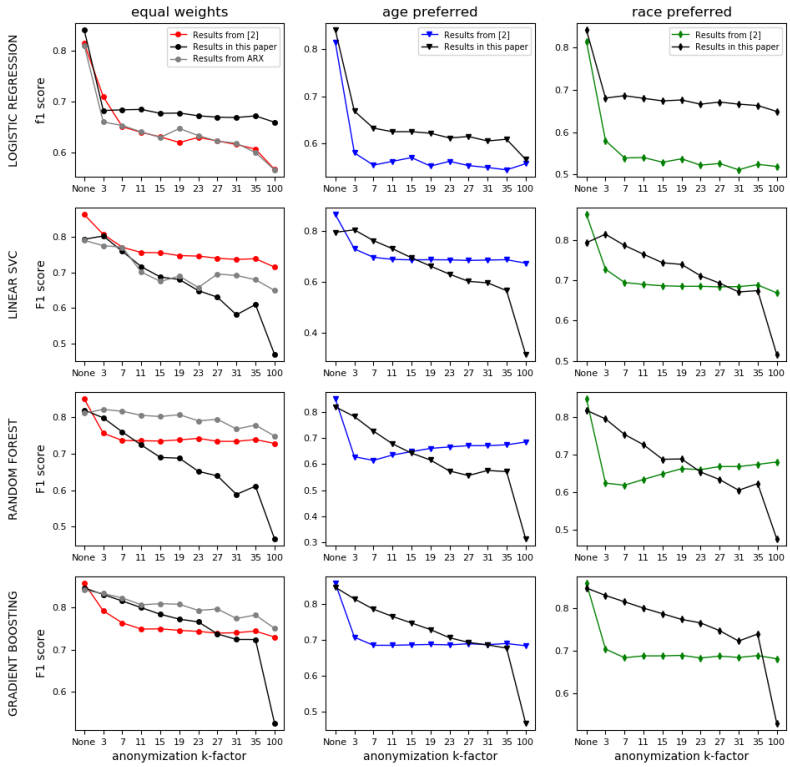

Figure 4: Multi-class classification on target *marital-status* on the adult dataset under several degrees of k-anoynmity.

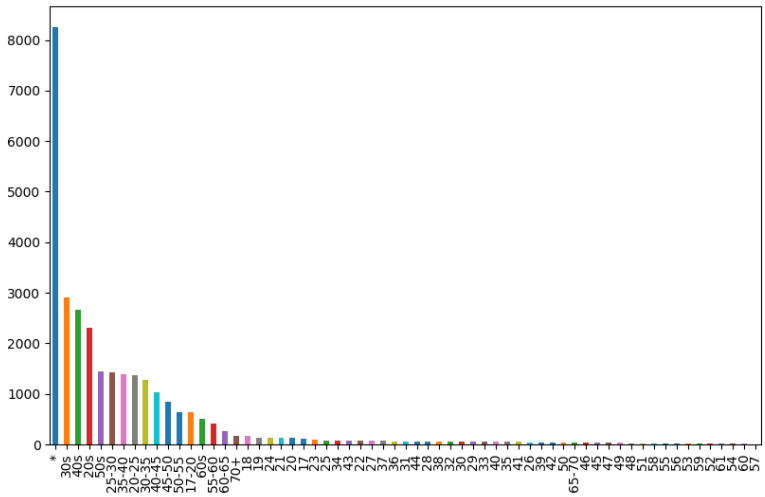

Figure 5: Distribution of the attribute 'age' for k=7 under local transformation.

k-s. The Random Forest classifier shows some increases in performance for a larger k-value (for example, k=23 in figure 3 for equal weights). This might be caused by the difference in generalization patterns between different k-values. In case some 'important' attributes are more generalized, it might lead to worse classifier performance, and if they stay more unchanged it would lead to better performance, which could be the reason in this case.

Performance on target *marital-status* (figure 4) is somewhat more similar to original results than in the previous case. In almost every case we have similar behaviour - gradually worse performance with k getting larger, and a drastic drop in the performance with k reaching 100. For Linear SVC, Random Forest and Gradient Boosting in every of the weight settings, in figure 4 we can see a sharp drop in the performance for k=100. This could be explained by the fact that most of the data in 100-anonymous datasets are generalized to the highest level (value "*") and therefore lost. However, it is hard to explain the large difference in performance for these cases between our results and results in [2] due to unavailability of datasets from the mentioned paper.

In section 4 we mentioned two ways for anonymizing the dataset in ARX, globally and locally. Both techniques had one main drawback that leads to bad results. For global transformation, the drawback is the fact that we unnecessarily lose a lot of information, when we could have obtained the same level of privacy by anonymizing fewer values, as already argued in section 4. The drawback of the local transformation is the inconsistency of data types within one attribute. This is best described with the example of a distribution of val-

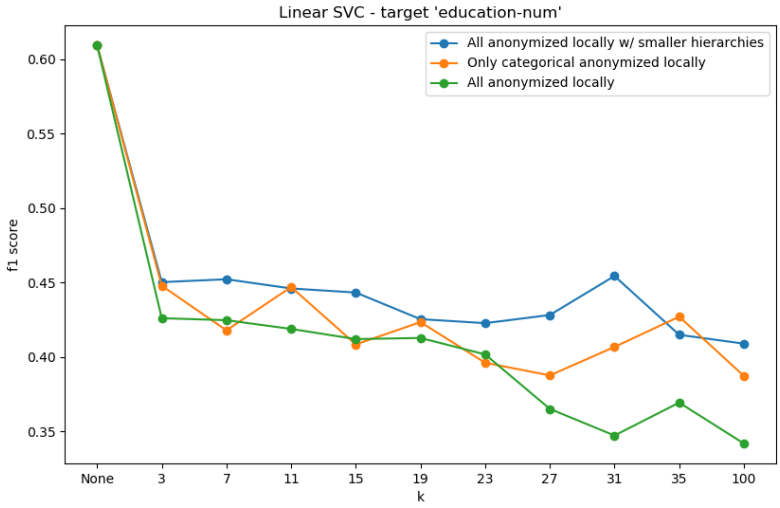

Figure 6: Classifier performance under different types of anonymization.

ues of one numerical attribute in Adult dataset shown in figure 5. By locally generalizing numerical values the columns contain the mixture of original continuous values and generalized ranges of values. The issue here is that those attributes cannot anymore be treated as continuous, but as a categorical with a large number of different possible values and it leads to bad classifier performance. One way to avoid this problem is to (1) keep the local transformation of categorical features, and at the same time apply the global transformation on numerical features. Another solution could be (2) defining generalization hierarchies with fewer levels for categorical data and keep applying the local transformation on each of the features. Figure 6 shows the example of classifier performance (LinearSVC) on datasets obtained by local transformation and the two proposed solutions.

The green line represents the performance when all of the attributes are anonymized locally. Orange and blue line represent performance for the datasets obtained by solutions (1) and (2) respectively. In both of those cases, performance is slightly better than the first (green) line. With the orange line we get drops and raises in the performance. This might be due to the importance of some numerical values on the classifier performance, therefore if we generalize the entire column to a certain generalization level, it highly affects the classifier performance. With the blue line, we get the 'smoother' performance results and overall the best of three presented cases. In further in experiments in ARX we apply exactly the local transformation on all features, with hierarchies defined with fewer levels.

6 Conclusion

In this paper, we showed the results and effects of anonymization on classifier performance. We also dealt with recreating the existing results, having a few main difficulties: unavailability of the algorithm implementation used in the original paper, as well as not knowing the exact definition of hierarchies used to generalize categorical attributes. Furthermore, tools, parameters and preprocessing methods used for classification tasks stayed unclear as well, leaving us assuming and experimenting with different possibilities to obtain similar results. Classifier performance in most of the cases is worse with k-value being larger, as expected. From the experiments with ARX, we have shown that the way we anonymized data and the way we define generalization hierarchies can have a large impact as well, as we obtained better results when applying a local transformation on categorical data and global transformation on numerical data to avoid the explosion of possible values for a single attribute. Also, better results are obtained by simply reducing the number of levels in generalization hierarchies of numerical attributes for the same reason. Furthermore, it is shown that the difference in algorithmic performance between SaNGreeA (a rather simple algorithm) and the one used by ARX is very big, meaning that state-of-art anonymization techniques can provide competitive Machine Learning pipelines for real-world usage, and decrease the effects of a currently necessary tradeoff between privacy and ML performance.

References

[1] B. Malle, P. Kieseberg, E. Weippl, and A. Holzinger, "The right to be forgotten: towards machine learning on perturbed knowledge bases," in *International Conference on Availability, Reliability, and Security*, pp. 251–266, Springer, 2016.

[2] B. Malle, P. Kieseberg, and A. Holzinger, "Do not disturb? classifier behavior on perturbed datasets," in *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pp. 155–173, Springer, 2017.

[3] A. Campan and T. M. Truta, "Data and structural k-anonymity in social networks," in *Privacy, Security, and Trust in KDD*, pp. 33–54, Springer, 2009.

[4] T. Šarčević, "K-anonymity with sangreea." `https://github.com/tanjascats/anonymization.git`, 2018.

[5] B. Malle, "Machine learning for health informatics, assignment 3: Social network anonymization with sangreea." `https://github.com/cassinius/mlhi-ass2-anonymization.git`, 2016.

[6] F. Prasser and F. Kohlmayer, "Putting statistical disclosure control into practice: The arx data anonymization tool," in *Medical Data Privacy Handbook*, pp. 111–148, Springer, 2015.

[7] F. Kohlmayer, F. Prasser, C. Eckert, A. Kemper, and K. A. Kuhn, "Flash: efficient, stable and optimal k-anonymity," in *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pp. 708–717, IEEE, 2012.

[8] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017.