A Review-3 Report on

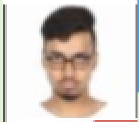
Team ID & Title: 20W1025

Utility-preserving anonymization for health data publishing

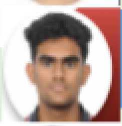
Submitted
as part of **BCI2001-Data Privacy Course Project Component**
by

18BCI0109
Rakshith Sachdev

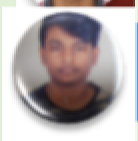
18BCI0110
Rishab Krishnamurthy



18BCI0207
Shashank Gummuluru



18BCI0247
Rohan Allen



18BCI0253
Phanider Edukulla

To

Dr M Rajasekhara Babu

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

February 2020

Index

Chapter	Topic	Page No.
	Abstract	
1	Introduction	1
2	Problem statement & Objectives	2
3	Literature Review	
3.1.	Existing models/methods/algorithms	
3.2.	Gaps identified in existing literature	
4	Software Requirements	
5	Design	
6	Implementation	
7	Results Analysis	
8.	Applicability category	
9.	Conclusions	
7.	References	

Abstract

Electronic health records (EHR) contain data that can very easily identify and expose sensitive information about the users that malicious parties can exploit and use to their advantage. However, analysis must be performed in order to analyze these records and provide useful results and conclusions from the given records. To achieve this, anonymization techniques are used. These techniques help achieve utility while preserving the privacy of the data. In the medical industry, generalization is the most common method to achieve this. In this project we implement a utility-preserving method for the privacy preserving data publishing((PPDP). The method is broken down into three main steps or categories. The first one deals with the utility-preserving model. Then we insert the counterfeit records. Finally, the counterfeit records are cataloged. This applies full domain generalization. Previous methods like suppression and relocation come with the drawback of not being scalable to large datasets. With all the metrics, our proposed method shows a lower information loss than the current existing methods while maintaining the utility.

Keywords: *Data Privacy; Utility-preserving; Data Anonymization; Grouping; k-anonymity; Medical privacy; privacy preserving data publishing (PDPP);*

Chapter 1
INTRODUCTION

Making electronic health records (EHRs) public to the masses may expose sensitive information and thus compromise the privacy and identity of an individual. Usually health records are anonymized before publishing, therefore satisfying privacy models such as *k*-anonymity. Generalization is the most commonly used anonymization algorithm which leads to immense information loss. Therefore we incorporate a utility preserving model called *h*-ceiling which restricts generalization. Thus data utility is preserved and this data can be useful to data analysts.

Protecting the privacy of medical data is extremely vital given the sensitivity of this information. It can lead to severe consequences if it falls into the wrong hands. At the same time preserving the utility of medical records is also necessary so that this information can be used in surveys, analysis, etc to improve the quality of healthcare provided. Our project attempts to delicately balance both these priorities so that neither data privacy nor utility is compromised.

CHAPTER 2

PROBLEM STATEMENT & OBJECTIVES

2.1. Problem Statement:

To implement a utility-preserving anonymization algorithm and to show that the utility of EHRs anonymized by the proposed method is significantly better than those anonymized by previous approaches.

2.2. Objectives:

1	To anonymize and protect EHRs
2	To preserve utility of EHRs
3	To design an algorithm to balance both privacy and utility of EHRs
4	To compare information loss between proposed and existing algorithm

CHAPTER 3
LITERATURE REVIEW

3.1. Existing models/methods/algorithms

Generalization is traditionally used which causes information loss, and thus it is not prefered.

Existing techniques for privacy-preserving data sharing deal largely with structured data.

Current privacy approaches for EHRs focus on detection and removal of patient identifiers from the data, which may be inadequate for protecting privacy or preserving data quality.

3.1. Gaps identified in existing literature

Extreme Data loss.

Data utility not a key priority.

Data quality not measured

Uselessness of EHRs for data analysts and testers.

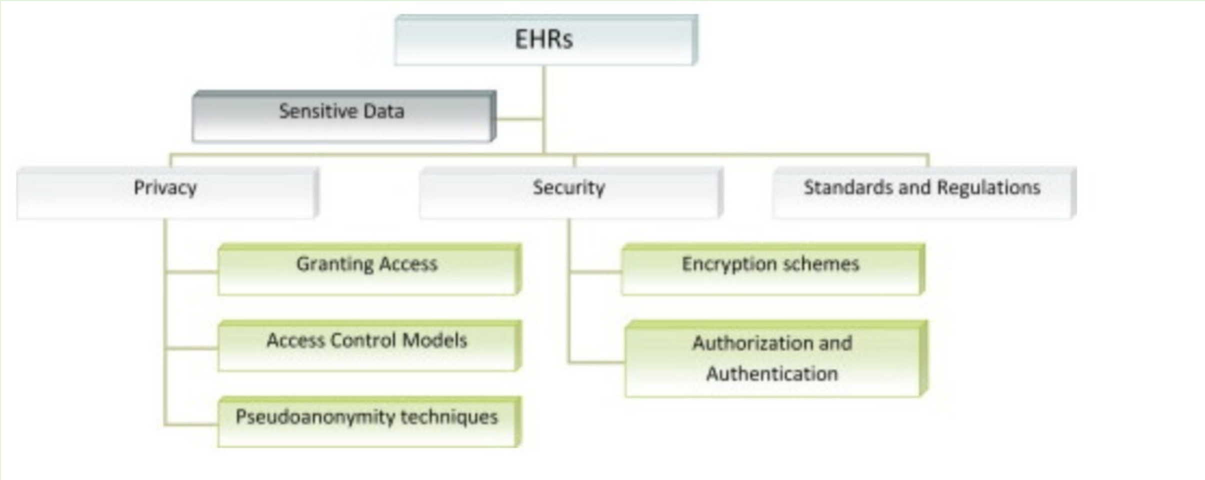
CHAPTER 4

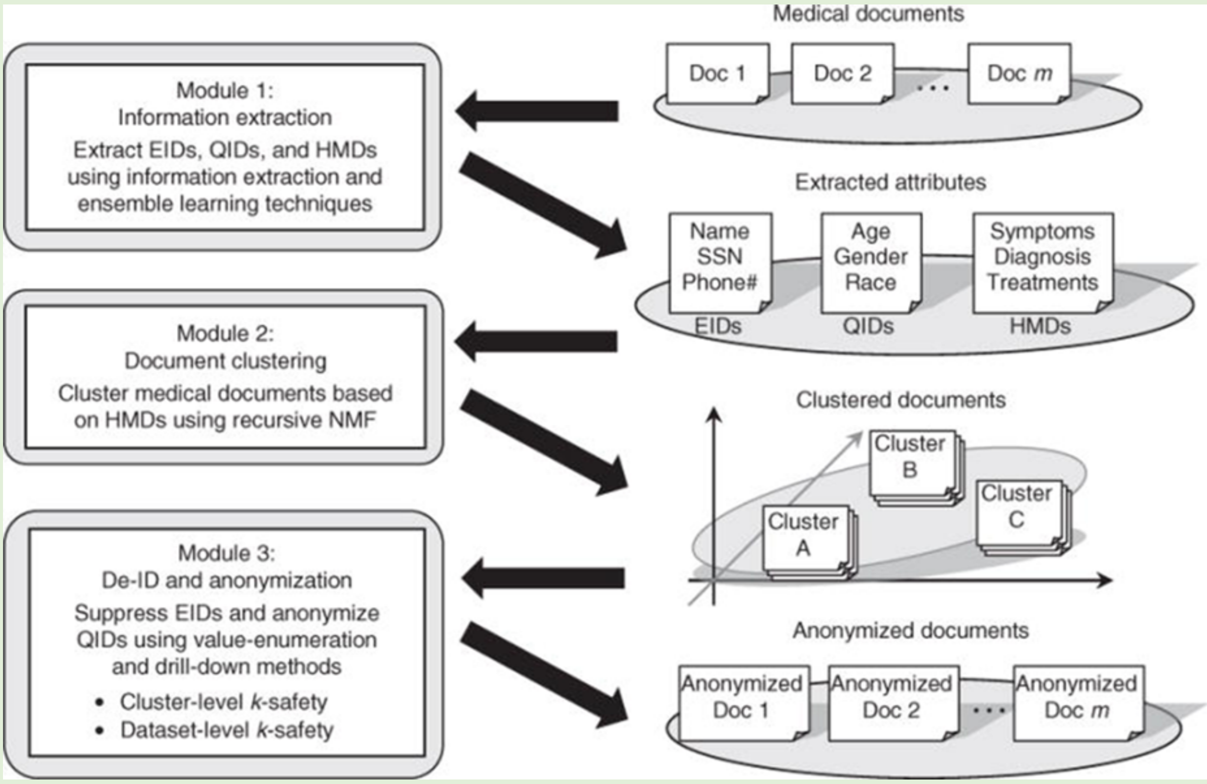
REQUIREMENTS

4.1. Software Requirements

- Jupyter Notebooks
  - Description: to store and run ipynb files.
- Python
  - Description: programming language used to write the code
- Python libraries
  - Various libraries used like pandas, numpy, sklearn etc.

CHAPTER 5
Design





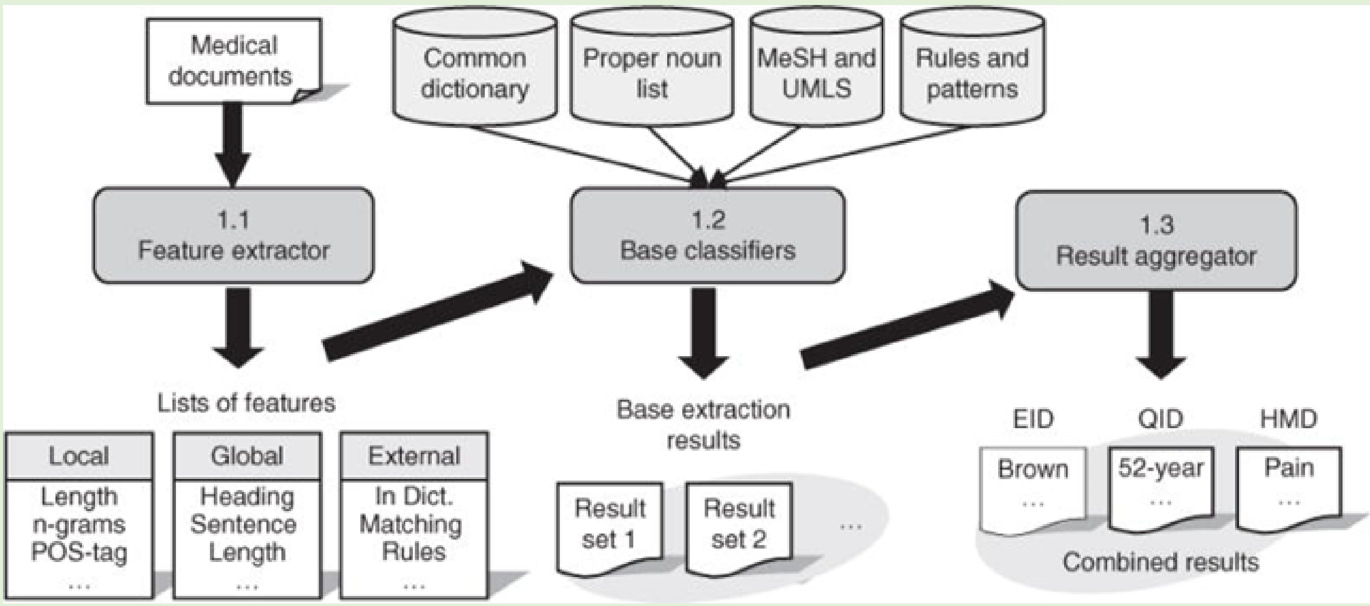
CHAPTER 6
IMPLEMENTATION (Refer to chapter 7 for the snapshots of the code)

Algorithm 1: Anonymization Algorithm

```
Input  : Original data O, Generalization rule G,
         Privacy parameter k, utility parameter h
Output: Anonymized data AT, Catalog for
         counterfeit records C
Create hierarchical lattice hl for all possible
generalization cases, except for the case where the
degree of generalization is more than h.
min = Maximum value of RCE
for each node n_i ∈ hl do
    TempC = ∅, C = ∅
    T̂* = generalization(O, n_i)
    E_m ← list of equivalent class in T̂*
    for j = 1 to |m| do
        if |E_j| < k then
            for j = 1 to |m| do
                addCounterfeitRecords(E_j, TempC);
            end
        end
    end
    C = Grouping(T̂*, TempC)
    result = CalcuateRCE(T̂*, C)
    if min > result && result ≠ null then
        AT = T̂*
        min = result
    end
end
return AT and C
```

Algorithm 2: Grouping Algorithm

```
Input  : Generalized data T̂*, Temporary catalog for
         counterfeit records TempC
Output: Catalog for counterfeit records C
Let e be the equivalent class.
Create a list of set S_e < SensitiveInformation_d,
Count_d, CounterfeitRecordsCount_d, > with respect
to T̂* and TempC
S_e is sorted by the sum of CounterfeitRecordsCount
groupedIDList = ∅
C= ∅
for i = 1 to |e| do
    max = 0
    remainCounterfeitRecords = the sum of
    CounterfeitRecordsCount in S_i
    while remainCounterfeitRecords > 0 do
        for j = 1 to |e| do
            if i == j then continue
            cnt = matching(S_i, S_j, C)
            if cnt ≥ max then
                tempClassID = j
                max = cnt
            end
        end
        if max == 0 then return null
        addToC(S_i, S_tempClassID)
        addToGroupedIDList(i, tempClassID)
        remainCounterfeitRecords =
        countRemainedRecords(i, C)
    end
end
return C
```



Medical documents
Common dictionary
Proper noun list
MeSH and UMLS
Rules and patterns
1.1
Feature extractor
1.2
Base classifiers
1.3
Result aggregator
Lists of features
Local
Length
n-grams
POS-tag
...
Global
Heading
Sentence
Length
...
External
In Dict.
Matching
Rules
...
Base extraction results
Result set 1
Result set 2
...
EID
QID
HMD
Brown
...
52-year
...
Pain
...
Combined results

CHAPTER 7
RESULTS ANALYSIS

NOTE: The following snapshots and tables are explained in detail in the YouTube presentation. Also, the justification of our limited implementation is present over there. It can be found here.

In [1]:
```
import pandas as pd
import numpy as np
import scipy.stats
%matplotlib inline
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
# get rid of warnings
import warnings
warnings.filterwarnings("ignore")
# get more than one output per Jupyter cell
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
# for functions we implement later
from utils import best_fit_distribution
from utils import plot_result
```

In [2]:
```
df = pd.read_csv("health_data.csv")
```

In [3]:
```
df.shape
df.head()
```

Out[3]: (891, 10)

Out[3]:

	PatientID	Insured	numVisitors	Name	Sex	Age	RoomNum	Bill (in thousand)	docRef	Condition
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	373450	8.0500	NaN	S

In [4]:
```
df.drop(columns=["PatientID", "Name"], inplace=True) # dropped because unique for every row
df.drop(columns=["RoomNum", "docRef"], inplace=True) # dropped because almost unique for every row
df.dropna(inplace=True)
```

In [5]:
```
df.shape
df.head()
```

Out[5]: (713, 6)

Out[5]:

	Insured	numVisitors	Sex	Age	Bill (in thousand)	Condition
0	0	3	male	22.0	7.2500	S
1	1	1	female	38.0	71.2833	C
2	1	3	female	26.0	7.9250	S
3	1	1	female	35.0	53.1000	S
4	0	3	male	35.0	8.0500	S

In [7]:
```
encoders = [(["Sex"], LabelEncoder()), (["Condition"], LabelEncoder())]
mapper = DataFrameMapper(encoders, df_out=True)
new_cols = mapper.fit_transform(df.copy())
df = pd.concat([df.drop(columns=["Sex", "Condition"]), new_cols], axis="columns")
```

In [8]:
```
df.shape
df.head()
```

Out[8]: (713, 6)

Out[8]:

	Insured	numVisitors	Age	Bill (in thousand)	Sex	Condition
0	0	3	22.0	7.2500	1	2
1	1	1	38.0	71.2833	0	0
2	1	3	26.0	7.9250	0	2
3	1	1	35.0	53.1000	0	2
4	0	3	35.0	8.0500	1	2

```
In [9]: df.nunique()

Out[9]: Insured                2
        numVisitors            3
        Age                   88
        Bill (in thousand)   220
        Sex                    2
        Condition              3
        dtype: int64
```

```
In [10]: categorical = []
         continuous = []
```

```
In [11]: for c in list(df):
             col = df[c]
             nunique = col.nunique()
             if nunique < 20:
                 categorical.append(c)
             else:
                 continuous.append(c)
```

```
In [12]: categorical

Out[12]: ['Insured', 'numVisitors', 'Sex', 'Condition']
```

```
In [13]: continuous

Out[13]: ['Age', 'Bill (in thousand)']
```

```
In [14]: for c in categorical:
             counts = df[c].value_counts()
             np.random.choice(list(counts.index), p=(counts/len(df)).values, size=5)

Out[14]: array([0, 0, 0, 0, 0])

Out[14]: array([1, 3, 3, 3, 2])

Out[14]: array([1, 0, 0, 0, 1])

Out[14]: array([2, 1, 2, 1, 2])
```

```
In [15]: # https://stackoverflow.com/a/37616966/1820480
```

```
In [16]: best_distributions = []
```

```
In [17]: # for c in continuous:
         #     data = df[c]
         #     best_fit_name, best_fit_params = best_fit_distribution(data, 50)
         #     best_distributions.append((best_fit_name, best_fit_params))
```

```
In [18]: best_distributions

Out[18]: []
```

```
In [19]: best_distributions = [
             ('fisk', (11.744665309421649, -66.15529969956657, 94.73575225186589)),
             ('halfcauchy', (-5.537941926133496e-09, 17.86796415175786))]
```

In [21]:
```
def generate_like_df(df, categorical_cols, continuous_cols, best_distributions, n, seed=0):
    np.random.seed(seed)
    d = {}

    for c in categorical_cols:
        counts = df[c].value_counts()
        d[c] = np.random.choice(list(counts.index), p=(counts/len(df)).values, size=n)

    for c, bd in zip(continuous_cols, best_distributions):
        dist = getattr(scipy.stats, bd[0])
        d[c] = dist.rvs(size=n, *bd[1])

    return pd.DataFrame(d, columns=categorical_cols+continuous_cols)
```

In [22]:
```
gendf = generate_like_df(df, categorical, continuous, best_distributions, n=100)
```

In [23]:
```
gendf.shape
gendf.head()
```

Out[23]: (100, 6)

Out[23]:

	Insured	numVisitors	Sex	Condition	Age	Bill (in thousand)
0	0	1	1	0	25.406552	9.474289
1	1	3	0	2	51.812626	11.859376
2	1	1	1	2	12.387505	19.327654
3	0	2	1	2	54.595218	43.251377
4	0	3	1	2	45.181993	10.322591

In [24]:
```
gendf.columns = list(range(gendf.shape[1]))
```

In [25]:
```
gendf.to_csv("output.csv", index_label="id")
```

In [26]:
```
gendf.shape
gendf.head()
```

Out[26]: (100, 6)

Out[26]:

	0	1	2	3	4	5
0	0	1	1	0	25.406552	9.474289
1	1	3	0	2	51.812626	11.859376
2	1	1	1	2	12.387505	19.327654
3	0	2	1	2	54.595218	43.251377
4	0	3	1	2	45.181993	10.322591

CHAPTER 8
Applicability category

The healthcare industry stores a large amount of data on their patients. This contains a lot of sensitive data which must be protected from malicious parties. This project helps in anonymizing this data hence protecting the PILs of the record owners. This speaks to the relevance in society as privacy protection is key in today's world. This project's core idea revolves around the healthcare industry. The project also incorporates data structures and algorithms in order to implement the anonymizing algorithm. The project is done as a team with each person getting assigned specific tasks in order to complete the project in an efficient manner.

Chapter 9
REFERENCES

1. Hyukki Lee, Soohyung Kim, Jong Wook Kim, and Yon Dohn Chung, “Utility-preserving anonymization for health data publishing”, BMC Med Inform Decis Mak. 2017; 17: 104. doi: 10.1186/s12911-017-0499-0.

2. Xiao-Bai Li, Jialun Qin, “Anonymizing and Sharing Medical Text Records”, Published in final edited form as: Inf Syst Res. 2017; 28(2): 332–352.

3. https://www.hhs.gov/hipaa/for-professionals/privacy/special-topics/de-identification/index.html

4. Fernandez-Aleman, J., 2015. Security And Privacy In Electronic Health Records: A Systematic Literature Review. [online] science direct. Available at: <https://www.sciencedirect.com/science/article/pii/S1532046412001864> [Accessed 5 June 2020].