
# ルール説明

PWS2020実行委員会
Cupワーキンググループ

初版 2020/08/26
第2版 2020/09/04

# PWS2020実行委員会　Cupワーキンググループ　メンバ

- 千田 浩司（NTT）
- 荒井 ひろみ（理研）
- 井口 誠（Kii）
- 小栗 秀暢（富士通研）
- 菊池 浩明（明治大）
- 黒政 敦史（FJCT）
- 中川 裕志（理研）
- 中村 優一（早稲田大）
- 西山 賢志郎（BizReach）
- 野島 良（NICT）
- 長谷川 聡（NTT）
- 波多野 卓磨（日鉄ソリューションズ）
- 濱田 浩気（NTT）
- 古川 諒（NEC）
- 村上 隆夫（産総研）
- 山岡 裕司（富士通研）
- 山田 明（KDDI総研）
- 渡辺 知恵美（筑波技術大）

# スケジュール

| | |
|---|---|
| 08/07(金) - 08/26(水) | エントリー受付 |
| 08/26(水) | ルール公開 |
| 08/27(木) - 09/07(月) | 予備戦(匿名化フェーズ) |
| 09/09(水) - 09/18(金) | 予備戦(攻撃フェーズ) |
| 09/22(火) | 予備戦結果通知 |
| 09/24(木) - 10/05(月) | 本戦(匿名化フェーズ) |
| 10/07(水) - 10/20(火) | 本戦(攻撃フェーズ) |
| 10/27(火) | CSS2020にて、最終結果発表 |
| 10/27(火) | CSS2020にて、各チームの<br>加工・攻撃手法のポスターセッション |

# はじめに

- PWSCUP2020：今年で6回目となる、匿名化とその攻撃の技術を競うコンテスト
- 通称 AMIC（“Anonymity against Membership Inference” Contest)
- 各参加チームは例年通り、与えられたデータを攻撃されないよう匿名化するタスクと、他の参加チームが匿名化したデータを攻撃するタスクを行う

- これまでのPWSCUPでは主に再識別攻撃（レコードリンケージ）を安全性の指標としていたが、今回は**メンバシップ推定**攻撃を安全性の指標として競う
  - メンバシップ推定：匿名化されたデータから、誰のデータが含まれているか推定
  - 機械学習分野等で注目されている**擬似データ生成**を含む、より多くの匿名化手法の安全性指標として利用できる
  - 機械学習分野のトップ会議 NeurIPS2020 のコンテスト hide-and-seek privacy challenge でも擬似データに対するメンバシップ推定をテーマにしている
- メンバシップ推定は、例えば新型コロナウイルス感染者のデータ分析等、学習データに含まれる対象者の存在自体が機微な場合の安全性指標にも有用と考えられる

# はじめに（続き）

- 表彰（各部門、得点が高い順に1位から3位まで表彰。得点の定義はp.10参照）
  - 総合部門：1位～3位
  - 匿名化部門：1位～3位
  - 攻撃部門：1位～3位

  } 重複受賞OK
  - 各部門の結果をHPに公開予定です
- 禁止事項
  - 各参加チームが出題者から個別に受け取る「サンプリングデータ」に関する情報の別参加チームへの提供
  - 各参加チームが独自に考案したアイデアを実装したプログラムの別参加チームへの提供
    - アイデア出しやルール確認のために一緒に議論することはOKとする
  - 意図的に運営を妨害する行為
- その他
  - 予備戦の内容をふまえて、本戦までにルールが変更される可能性があります。ルールを変更する場合は速やかに参加チームにご連絡します
  - PWS2020実行委員会 Cupワーキンググループ(WG) メンバの参加も認めています。ただし、メンバの参加者に対しては問い合わせ先MLから外すなど公平性を期すようにしています
  - WGから、匿名化データを評価してもらう目的で匿名化部門のみに参加するチームがおりますが、当該チームは各部門の順位から除外します
  - 予備戦の総合部門、匿名化部門、攻撃部門の結果をHPに公開予定です（表彰はありません）

# 再識別（レコードリンケージ）とメンバシップ推定

①パーソナルデータセット
（個人とIDが1対１に対応）

| ID | 属性1 | 属性2 | … |
|---|---|---|---|
| 00001 | | | |
| 00002 | | | |
| 00003 | | | |
| 00004 | | | |
| 00005 | | | |
| … | | | |

②行をランダムに入れ替え

| ID | 属性1 | 属性2 | … |
|---|---|---|---|
| 00002 | | | |
| 00005 | | | |
| 00001 | | | |
| 00004 | | | |
| 00003 | | | |
| … | | | |

③匿名化データ

| PID(仮名) | 属性1 | 属性2 | … |
|---|---|---|---|
| 10001 | | | |
| 10002 | | | |
| 10003 | | | |
| 10004 | | | |
| 10005 | | | |
| … | | | |

加工
（行の関係は維持）

**再識別（レコードリンケージ）**
**①と③から②のID列を推定**

| PID | ID |
|---|---|
| 10001 | 00004 |
| 10002 | 00002 |
| 10003 | 00001 |
| 10004 | 00005 |
| 10005 | 00003 |

**正解率 40%**

④行をサンプリング

| ID | 属性1 | 属性2 | … |
|---|---|---|---|
| 00001 | | | |
| 00003 | | | |
| 00004 | | | |
| … | | | |

⑤匿名化データ

| 属性1 | 属性2 | … |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

加工

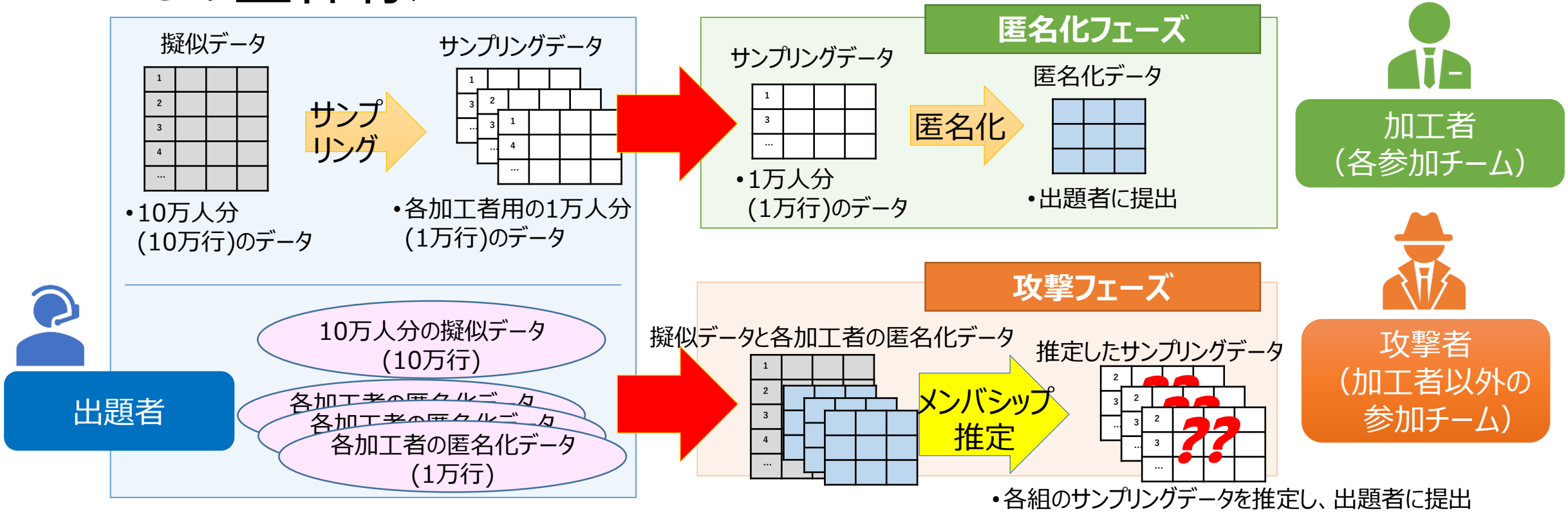
**メンバシップ推定**
**①と⑤から④のID列を推定(順不同)**

| ID |
|---|
| 00002 |
| 00003 |
| 00005 |

**正解率 33%**

- 用語
  - パーソナルデータセット：各行に個人のデータが記載された表形式のデータ
    - 本コンテストでは1人1行とする（複数行に同一IDは無い）
  - 攻撃：メンバシップ推定や再識別（レコードリンケージ）のこと。正解率が高いほど強力な攻撃となる
  - 匿名化：攻撃を防ぐためのパーソナルデータセットの加工（①→③、①→⑤、②→③、④→⑤）

# AMICの全体像




- Remark
  - **パーソナルデータセットの擬似データを用いる** → 加工者に非サンプリングデータを知られないようにするため
    - 擬似データ (Synthetic Data) : パーソナルデータセットとデータの分布等の特徴が類似した別のデータセット
    - パーソナルデータセットは、公開データセットの Census Income Data Set を使用（後述）
  - **擬似データは加工者に対しても秘匿**（サンプリングデータのみ提供。擬似データは攻撃フェーズで攻撃者に提供）
  - サンプリングは、**加工者毎にランダムサンプリング**
  - メンバシップ推定は、各匿名化データに対して**最も自信のある100レコード（の行番号）を提出**

# AMICで用いるパーソナルデータセット

- **Census Income Data Set** https://archive.ics.uci.edu/ml/datasets/census+income
- 機械学習の試用を想定した15属性の訓練用データ32,561レコード、テスト用データ16,281レコード
- 次頁で説明する擬似データ生成手法を活用し、**重複レコードの無い10万レコードの擬似データ**を生成
  - 重複を含む100万レコードの擬似データを生成してから、重複を削除し、10万レコードをランダムサンプリング
- 属性値の数・分布や重複度等を考慮し、以下の**9属性を用いる** (73*8*16*7*14*6*2*99*2=2,175,731,712通り)

**表1:AMICで用いる属性と属性値**

age: continuous. [17-90]
workclass(8): Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
education(16): Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
marital-status(7): Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
occupation(14): Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
relationship(6): Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
sex(2): Female, Male. hours-per-week: continuous. [1-99] income(2): >50K, <=50K

ヘッダ→
(今回はヘッダ行を除いたファイルを扱う)

| age | workclass | education | marital-status | occupation | relationship | sex | hours-per-week | income |
|---|---|---|---|---|---|---|---|---|
| 39 | State-gov | Bachelors | Never-married | Adm-clerical | Not-in-family | Male | 40 | <=50K |
| 50 | Self-emp-not-inc | Bachelors | Married-civ-spouse | Exec-managerial | Husband | Male | 13 | >50K |
| 38 | Private | HS-grad | Divorced | Handlers-cleaners | Not-in-family | Male | 40 | <=50K |
| 53 | Private | 11th | Married-civ-spouse | Handlers-cleaners | Husband | Male | 40 | <=50K |
| 28 | Private | Bachelors | Married-civ-spouse | Prof-specialty | Wife | Female | 40 | >50K |
| … | … | … | … | … | … | … | … | … |

# AMICで用いる擬似データ生成アルゴリズム

- [OMTH17] 岡田ら: 統計値を用いたプライバシ保護擬似データ生成手法, CSS2017 3F3-4.
- 入力のパーソナルデータセットの**各数値属性の平均、分散共分散行列と等しい擬似データ**
  - ただし実際には、離散化や最大小値補正により多少異なる
- カテゴリ属性はダミー変数化して数値属性として扱う
- **擬似データのレコード数は任意に設定**できる

Input: 各属性の平均のベクトル $\boldsymbol{\mu}$、分散共分散行列 $\Sigma$ 、各属性のヒストグラム、擬似データのレコード数

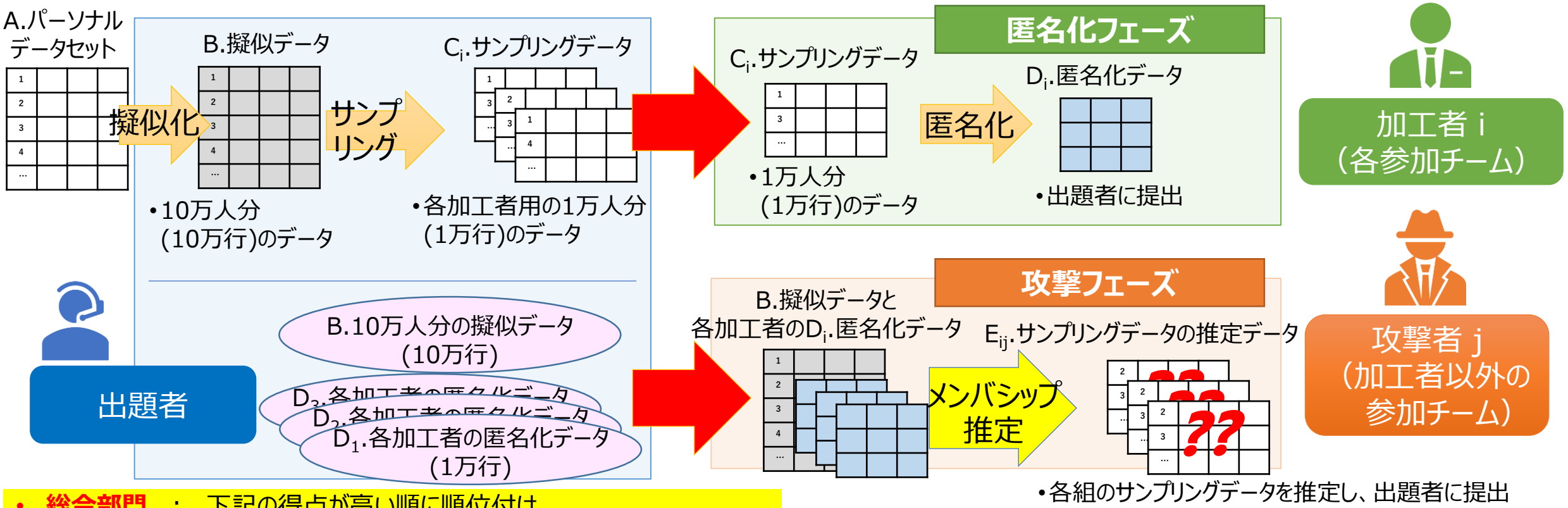
Output: 擬似データ $Z$

1. 各属性のヒストグラムとレコード数が入力と一致するランダムなデータセット $Y$ を生成
2. $Y$ を白色化し、各属性の平均が0、分散共分散行列が単位行列となるデータセット $Y'$ を生成
3. $\Sigma = U\Lambda U^T$ となる回転行列 および拡大縮小行列 $\Lambda^{1/2}$ （ $\Lambda$ の各要素を平方根に変換）
4. $Y'(U\Lambda^{1/2})^T$ を計算し、各行に $\boldsymbol{\mu}$ を足したデータセット $Z^*$ を計算
5. $Z^*$ に離散化や最大小値補正を行ったものを擬似データ $Z$ をとして出力

# 勝敗

A. パーソナルデータセット
B. 擬似データ
$C_i$. 加工者 i のサンプリングデータ
$D_i$. 加工者 i の匿名化データ
$E_{ij}$. 攻撃者 j が加工者 i のサンプリングデータを推定したデータ

**“A,B,C,D,E” と覚えてください**

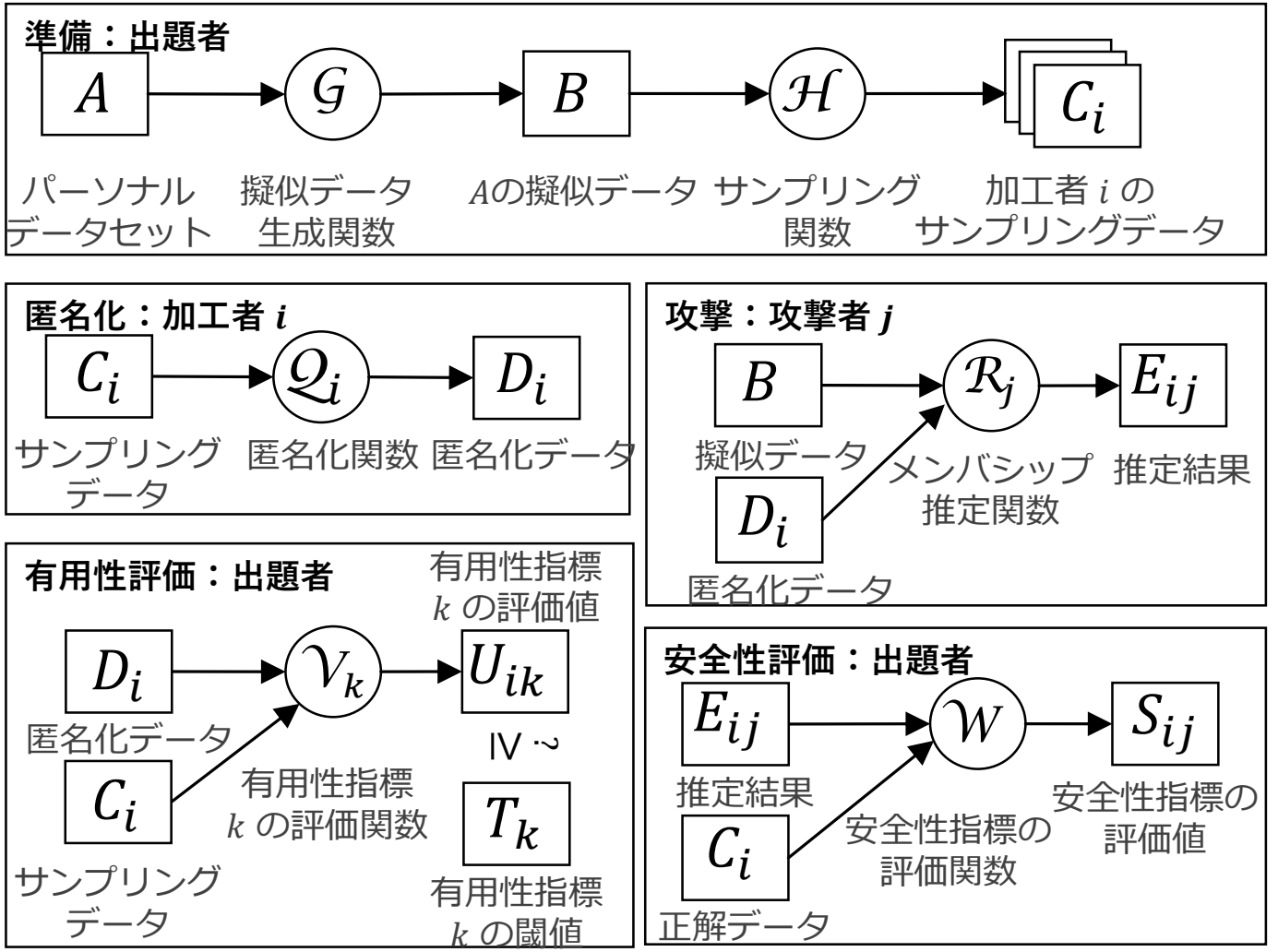



- **総合部門** ： 下記の得点が高い順に順位付け
  - 参加チームの得点：匿名化部門と攻撃部門の順位の和の逆数
    - 例：匿名化部門5位、攻撃部門3位の場合の得点は、$(5+3)^{-1}=0.125$
- **匿名化部門** ： 下記の得点が高い順に順位付け
  - 攻撃者は、各加工者の匿名化データ $D_i$ をメンバシップ推定 → 推定データ $E_{ij}$ を提出
  - 攻撃者 $j$ の加工者 $i$ に対する正解率の定義：$|E_{ij} \cap C_i|/100$
  - 加工者 $i$ の得点：1から全攻撃者の中の最高正解率を引いた値

- **攻撃部門** ： 下記の得点が高い順に順位付け
  - 攻撃者の得点：匿名化部門の1～3位（自身が1～3位の場合は、自身の順位を除く1～4位）の匿名化データに対する正解率の平均
- 予備戦の得点1割、本戦の得点9割として、足した値を匿名化部門、攻撃部門それぞれの得点とする
- その他：得点は細かい値にならないよう1,000倍および小数点以下切り捨て予定

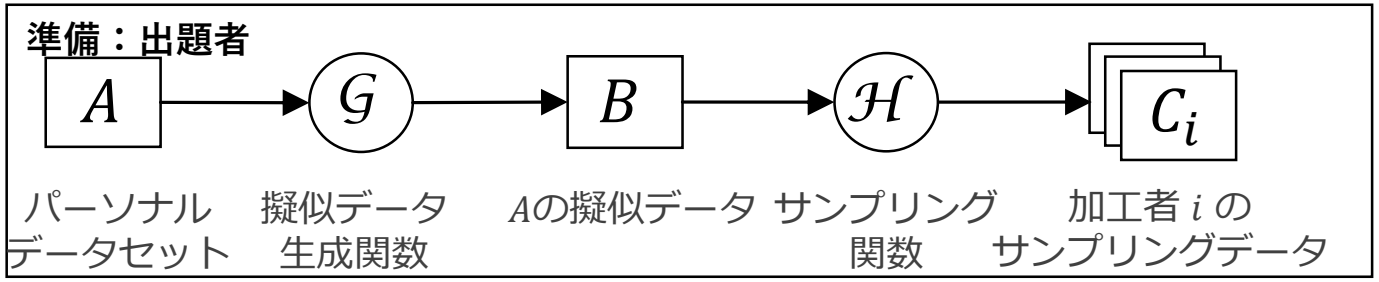
# 記号と処理フロー

| | |
|---|---|
| $A$ | パーソナルデータセット |
| $B$ | $A$ の擬似データ |
| $C_i$ | 加工者 $i$ のサンプリングデータ |
| $D_i$ | 加工者 $i$ の匿名化データ |
| $E_{ij}$ | 攻撃者 $j$ の加工者 $i$ に対するメンバシップ推定データ |
| $\mathcal{G}$ | 擬似データ生成関数 $(B = \mathcal{G}(\mathrm{seed}, A))$ |
| $\mathcal{H}$ | サンプリング関数 $(C_i = \mathcal{H}(i, B))$ |
| $\mathcal{Q}_i$ | 加工者 $i$ の匿名化関数 $(D_i = \mathcal{Q}_i(C_i))$ |
| $\mathcal{R}_j$ | 攻撃者 $j$ のメンバシップ推定関数 $(E_{ij} = \mathcal{R}_j(B, D_i))$ |
| $S_{ij}$ | 加工者 $i$ の攻撃者 $j$ に対する安全性指標の評価値 |
| $T_k$ | 有用性指標 $k$ の閾値 |
| $U_{ik}$ | 加工者 $i$ の有用性指標 $k$ の評価値 |
| $\mathcal{V}_k$ | 有用性指標 $k$ の評価関数 $(U_{ik} = \mathcal{V}_k(C_i, D_i))$ |
| $\mathcal{W}$ | 安全性指標の評価関数 $(S_{ij} = \mathcal{W}(C_i, E_{ij}))$ |



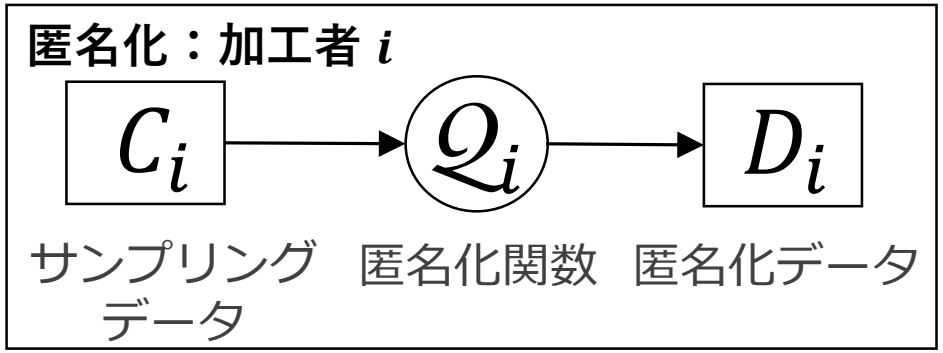

- 有用性指標
  - パーソナルデータセットとその匿名化データからそれぞれ得られる統計量や機械学習の予測・分類結果等の類似性を定量的に評価したもの
  - 類似しているほど有用性が高い
- 安全性指標
  - 攻撃に対する耐性を定量的に評価したもの
  - メンバシップ推定の正解率が低いほど安全性が高い

# 準備：出題者のタスク



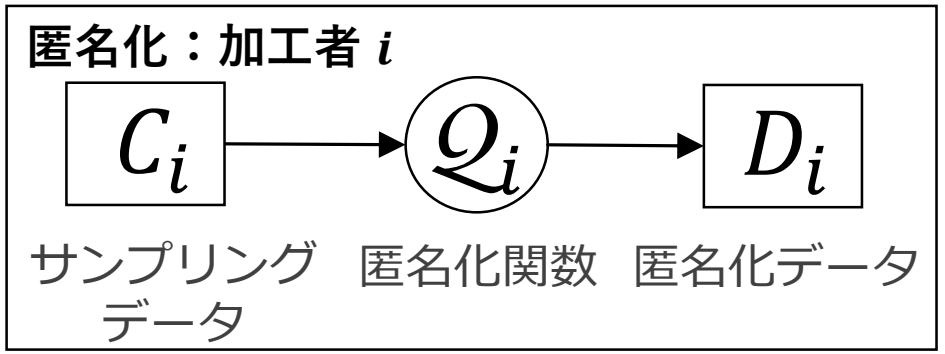

- パーソナルデータセット
  - Census Income Data Set の訓練用データ32,561レコードのうち、欠損値の無い30,162レコード
  - age, workclass, education, marital-status, occupation, relationship, sex, hours-per-week, incomeの9属性を利用
- 擬似データ生成関数
  - [OMTH17]方式を採用
  - 任意レコード数を出力できる（ここでは重複の無い10万レコードの擬似データを出力）
  - 各属性の平均・分散、二属性間の共分散が元のパーソナルデータセットと類似（カテゴリ属性はダミー変数化）
  - **実装コード：gen.py → お試し用に公開**
- 擬似データ
  - パーソナルデータセットと平均・分散・共分散が類似した、重複の無い10万レコードのデータセット
  - 加工者には渡さない（サンプリングデータしか渡さない）
- サンプリング関数
  - データセットを入力し、指定したサンプリング率(0～1)のレコードを出力
  - ここではサンプリング率=0.1とし、10万レコードの擬似データからランダムに1万レコードのサンプリングデータを出力
  - **実装コード：randomsampling.py → お試し用に公開**
- サンプリングデータ
  - 加工者毎に異なる、擬似データからランダムに1万レコードを抽出したデータ
- その他
  - 出題者は加工者毎にサンプリングしたレコードに対応する擬似データの行番号を記録・保持（**行番号は0スタートなので注意！！**）

# 匿名化：加工者のタスク




- サンプリングデータ
  - 出題者が作成した、加工者毎に異なる、擬似データからランダムに1万レコードを抽出したデータ
- 匿名化関数
  - 加工者が作成
  - 安全性指標の評価値が高くなるような（＝攻撃者がメンバシップ推定しにくくなるような）匿名化データを生成して出力
  - **サンプルコードをいくつか公開（後述）**
- 匿名化データ
  - 出題者に提出
  - レコード数はサンプリングデータのレコード数と一致しなくてもよい（ただし1,000～100,000とする）
    - ただしレコード数の差が大きくなるほど、有用性指標の評価値が下がるよう設計されているので注意
  - **所定の有用性指標の評価値を満たす必要がある（後述）※満たさないと失格になるので注意！！**
  - **制限事項：有用性指標の都合上、匿名化データの属性はp.7表1の9属性とし、各属性の属性値もp.7表1の何れかとする**
    - 例：属性ageの属性値は17～99の整数→匿名化データのageの属性値も17～99の整数（100、20代、[20-24]といった属性値は不可）

# 匿名化：公開サンプルコード



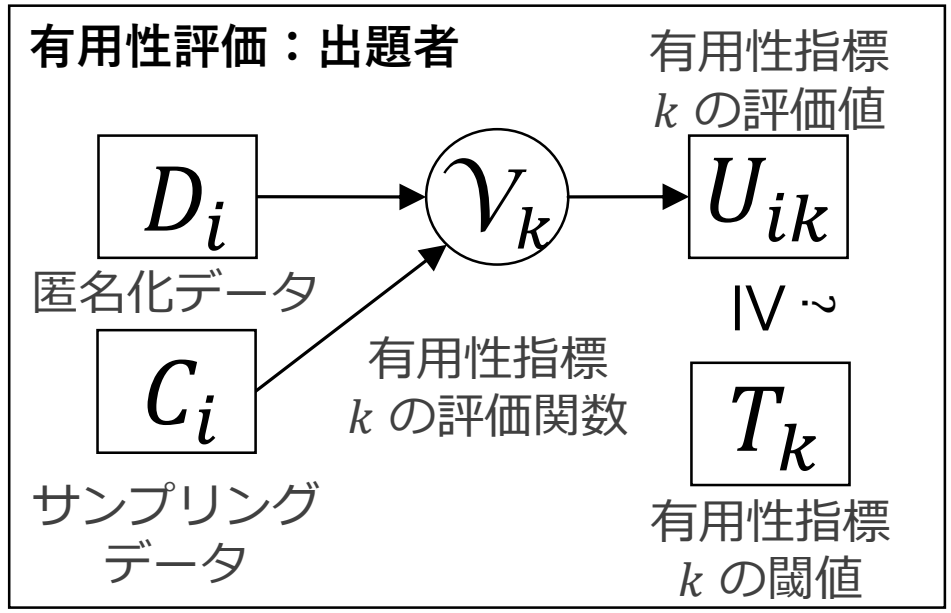

- synthetic.py　（擬似データ生成）
  - 入力データの各属性の平均・分散、二属性間の共分散が類似する擬似データを生成して出力
  - 擬似データのレコード数を指定できる

- rr.py　（ランダム化/Randomized Response/PRAM）
  - 維持確率 $p$（0～1）を指定し、各セルの値に対して確率 $p$ でそのまま維持し、確率 1-$p$ でランダムな属性値に置き換え

- rrp.py　（ランダム化/Randomized Response/PRAM）
  - 維持確率 $p$（0～1）を指定し、各セルの値に対して確率 $p$ でそのまま維持し、確率 1-$p$ で入力データの分布に従ったランダムな属性値に置き換え

- kanony.py　（$k$-匿名化（レコード削除））
  - 属性および閾値 $k$ の値を指定し（属性は複数指定可）、指定した属性の各レコードについて、全く同じ属性値の組が $k$ 個以上あればそのまま維持し、$k$-1 個以下であればレコード毎削除する

# 有用性評価：出題者、加工者のタスク




- 有用性指標（再掲）
  - パーソナルデータセットとその匿名化データからそれぞれ得られる統計量や機械学習の予測・分類結果等の類似性を定量的に評価したもの
  - 類似しているほど有用性が高い
- 有用性指標の評価関数
  - 加工者は、次頁以降で詳述する、**ヒストグラム、分散共分散行列、決定木分析の有用性指標**が閾値以上となる匿名化データを作成する必要がある
  - **検証用コード utilityfunc.py を公開** → 加工者は自身が作成した匿名化データの有用性が閾値以上かどうか確認できる
- 有用性指標の評価値と閾値
  - 次頁以降で詳述
- その他
  - 出題者は、加工者から提出された匿名化データに対して utilityfunc.py を実行し、有用性指標の評価値が閾値以上かどうか確認する → **閾値未満の場合は失格とする**

# 有用性指標：ヒストグラム

- 出題者に提出する匿名化データは、有用性指標の評価値が閾値未満だと失格になるので注意！！

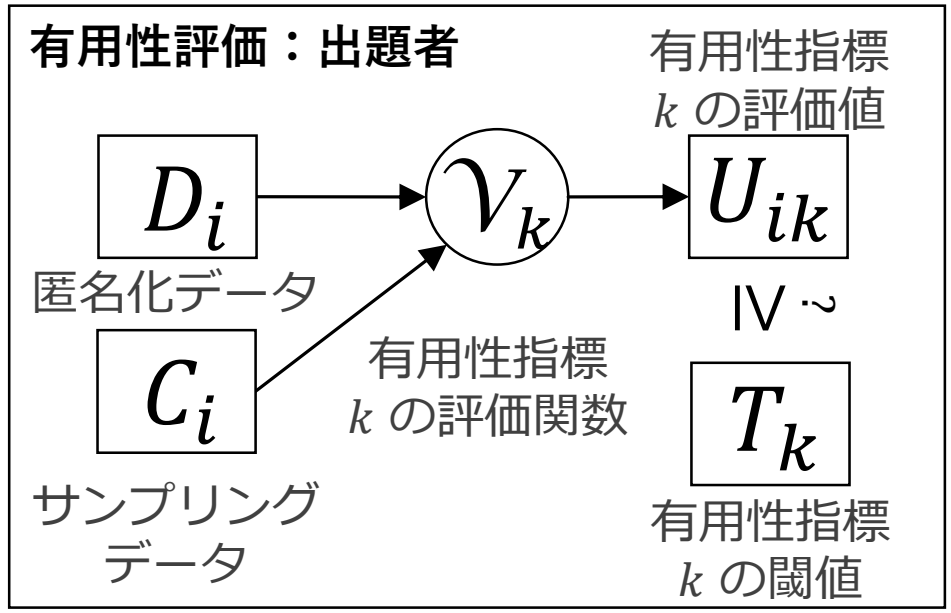
**Histgram:**

The frequencies of each value in each attribute of anonymized data $D_i$ and sub-sampled data $C_i$, respectively. For each value $X_{lm}$ of attribute $X_l$ and the frequencies $X^D_{lm}$ and $X^C_{lm}$ of $D_i$ and $C_i$ resp., the score of utility measure is defined as follows.

$$U_{i\,\text{histogram}} = 1 - \frac{\sum_{l,m} \left| X^D_{lm} - X^C_{lm} \right|}{2 C_i^{\text{Rec}} C_i^{\text{Att}}}, \qquad (1)$$

where $C_i^{\text{Rec}}$ and $C_i^{\text{Att}}$ are the numbers of records and attributes of $C_i$, respectively.

- 検証用コード utilityfunc.py を公開 → 閾値以上かどうか確認できる
- 有用性指標の閾値：$T_{\text{histgram}} = 0.99$ とする




| 属性名 | 種別 | 値域 |
|---|---|---|
| $X_1$: age | 整数値 | 17 – 90 |
| $X_2$: workclass | カテゴリ | Private 等 8 種類 |
| $X_3$: education | カテゴリ | Bachelors 等 16 種類 |
| $X_4$: marital-status | カテゴリ | Married-civ-spouse 等 7 種類 |
| $X_5$: occupation | カテゴリ | Tech-support 等 14 種類 |
| $X_6$: relationship | カテゴリ | Wife 等 6 種類 |
| $X_7$: sex | カテゴリ | Female, Male |
| $X_8$: hours-per-week | 整数値 | 1 – 99 |
| $X_9$: income | カテゴリ | >50K, <=50K |

# 有用性指標：分散共分散行列

- 出題者に提出する匿名化データは、有用性指標の評価値が閾値未満だと失格になるので注意！！

**Variance-covariance matrix:**

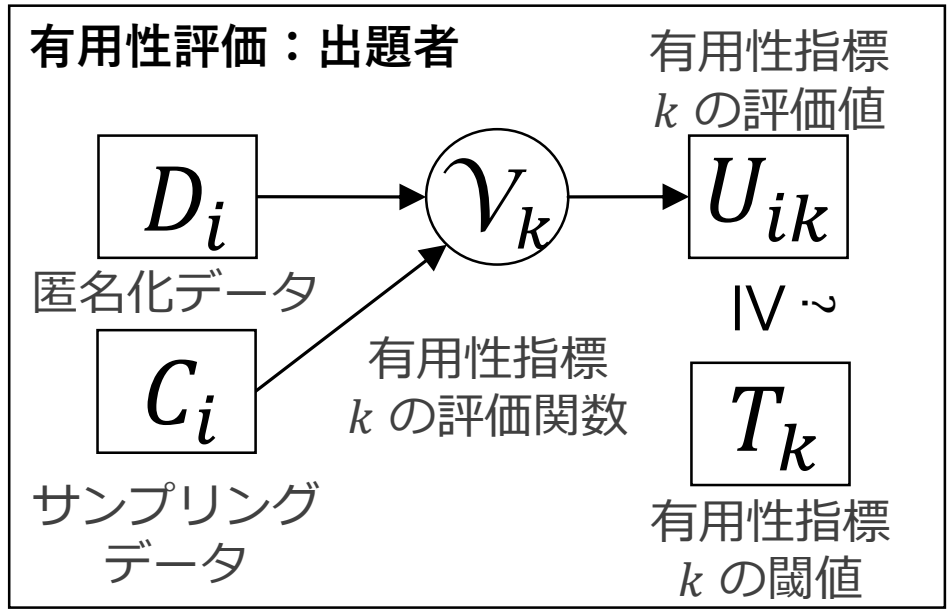
The score is based on the difference of each element of the variance-covariance matrices obtained from anonymized data $D_i$ and sub-sampled data $C_i$. Assume all attributes are numeric thanks to a dummy variable conversion for simplicity. For variance $\sigma_{ll}$ of $X_l$ and covariance $\sigma_{ll'}$ of $X_l$ and $X_{l'}$, the score of utility measure is defined as follows.

$$U_{i\,\mathrm{VCM}} = \left( \max_{l'} \sum_{l} \left| \sigma^{D}_{ll'} - \sigma^{C}_{ll'} \right| \right)^{-1}, \qquad (2)$$

where $\sigma^{D}_{ll'}$ and $\sigma^{C}_{ll'}$ are variance or covariance of $D_i$ and $C_i$ resp. and Eq (2) returns $\infty$ when

$$\max_{l'} \sum_{l} \left| \sigma^{D}_{ll'} - \sigma^{C}_{ll'} \right| = 0.$$

- 検証用コード utilityfunc.py を公開 → 閾値以上かどうか確認できる
- 有用性指標の閾値：$T_{\mathrm{VCM}} = 0.4$ とする




| 属性名 | 種別 | 値域 |
|---|---|---|
| $X_1$: age | 整数値 | 17 – 90 |
| $X_2$: workclass | カテゴリ | Private 等 8 種類 |
| $X_3$: education | カテゴリ | Bachelors 等 16 種類 |
| $X_4$: marital-status | カテゴリ | Married-civ-spouse 等 7 種類 |
| $X_5$: occupation | カテゴリ | Tech-support 等 14 種類 |
| $X_6$: relationship | カテゴリ | Wife 等 6 種類 |
| $X_7$: sex | カテゴリ | Female, Male |
| $X_8$: hours-per-week | 整数値 | 1 – 99 |
| $X_9$: income | カテゴリ | >50K, <=50K |

# 有用性指標：決定木分析

- 出題者に提出する匿名化データは、有用性指標の評価値が閾値未満だと失格になるので注意！！

**Decision-tree analysis:**

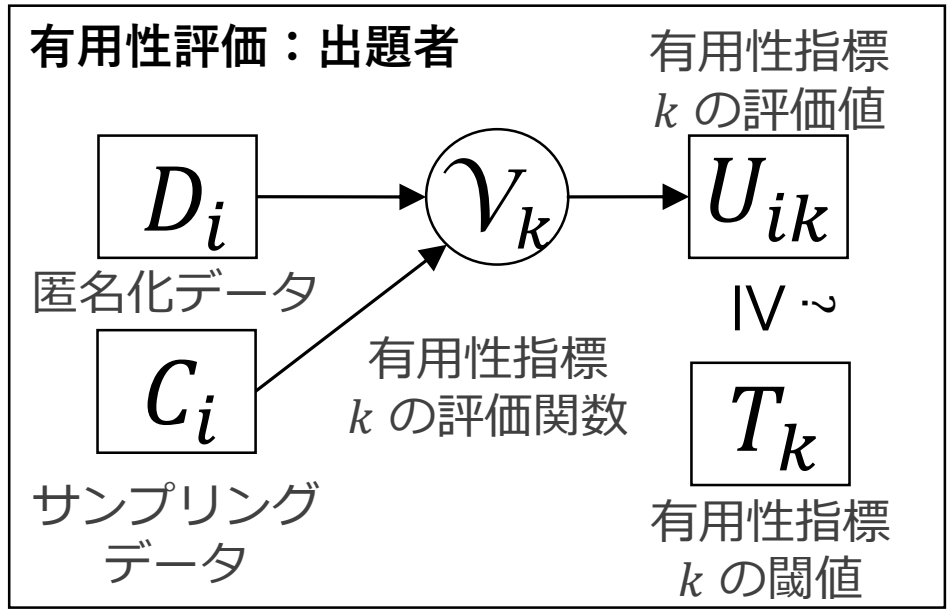
The score is based on the value of F-measure obtained from decision-tree functions $\mathcal{Y}^D_{X_l}$ of anonymized data $D_i$ and $\mathcal{Y}^C_{X_l}$ of sub-sampled data $C_i$, where $X_l$ is an objective variable. The test data are 16,281 training records $Z_0, \ldots, Z_{16280}$ in Census Income Data Set. The objective variables are $X_6$: relationship and $X_9$: income. Define $TP_6$, $FP_6$, $FN_6$, $FN_6$ are respectively the frequencies such that

$$(\mathcal{Y}^D_{X_6}(Z_m), \mathcal{Y}^C_{X_6}(Z_m)) = \begin{cases} \text{“Husband”, “Husband”} \\ \text{“Husband”, “Others”} \\ \text{“Others”, “Husband”} \\ \text{“Others”, “Others”} \end{cases}$$

for $m = 0, \ldots, 16280$. Then the score of utility measure for $X_6$ is defined as follows.

$$U_{i\,\mathrm{DTA}} = \frac{2\mathrm{Recall} \cdot \mathrm{Precision}}{\mathrm{Recall} + \mathrm{Precision}}, \qquad (3)$$

where Precision$= TP_6/(TP_6 + FP_6)$ and Recall$= TP_6/(TP_6 + FN_6)$. The score of utility measure for $X_9$ can be defined similarly.




| 属性名 | 種別 | 値域 |
|---|---|---|
| $X_1$: age | 整数値 | 17 – 90 |
| $X_2$: workclass | カテゴリ | Private 等 8 種類 |
| $X_3$: education | カテゴリ | Bachelors 等 16 種類 |
| $X_4$: marital-status | カテゴリ | Married-civ-spouse 等 7 種類 |
| $X_5$: occupation | カテゴリ | Tech-support 等 14 種類 |
| $X_6$: relationship | カテゴリ | Wife 等 6 種類 |
| $X_7$: sex | カテゴリ | Female, Male |
| $X_8$: hours-per-week | 整数値 | 1 – 99 |
| $X_9$: income | カテゴリ | >50K, <=50K |

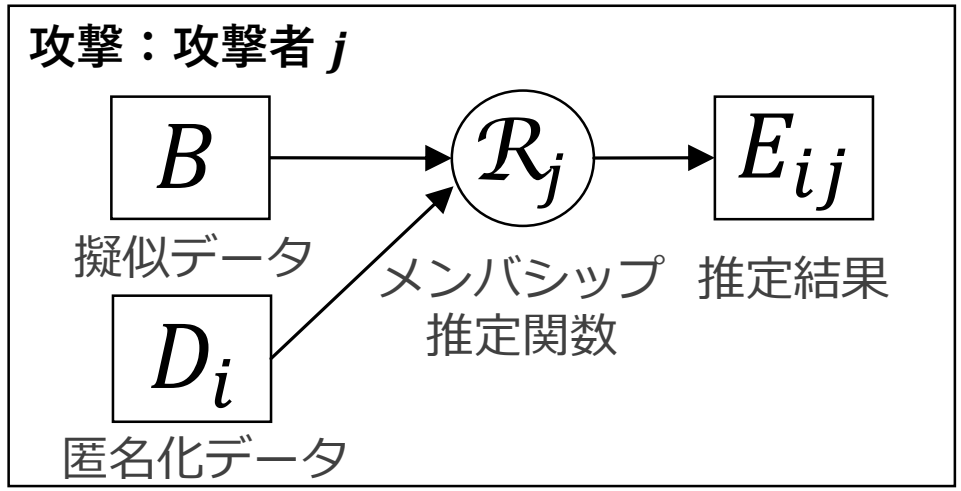
- 検証用コード utilityfunc.py を公開 → 閾値以上かどうか確認できる
- 有用性指標の閾値：$T_{\mathrm{DTA}} = 0.85$ とする（$X_6$,$X_9$ともに）

# 決定木分析のアルゴリズム

- 決定木分析
  - scikit-learn の sklearn.tree.DecisionTreeClassifer を利用
    - https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifer.html

- DecisionTreeClassifer のパラメータは下記とする。
  - random_state = 0
  - max_depth = 5 (income 推定)
  - max_depth = 3 (relationship 推定)
  - その他は指定せず、デフォルトのパラメータを用いる。

# 攻撃：攻撃者のタスク




- 擬似データ、各加工者の匿名化データ
  - 出題者から受け取る
- メンバシップ推定関数
  - 擬似データと匿名化データから、擬似データのどのレコードがサンプリングされて匿名化されたかを推定して出力（出題者に提出）
  - **サンプルコード attack.py を公開**
    - 擬似データ $B$ と匿名化データ $D_i$ の各レコードの距離に基づき、サンプリングデータ $C_i$ を推定したデータ $E_{ij}$ を出力
    - レコード間のユークリッド距離（整数値の属性であれば属性値の差、カテゴリ属性であれば属性値が一致すれば 0、そうでなければ 1 とし、それらの合計）を計算し、匿名化データの各レコードと最も距離が小さい擬似データ $B$ のレコードの行番号を返す

匿名化データ

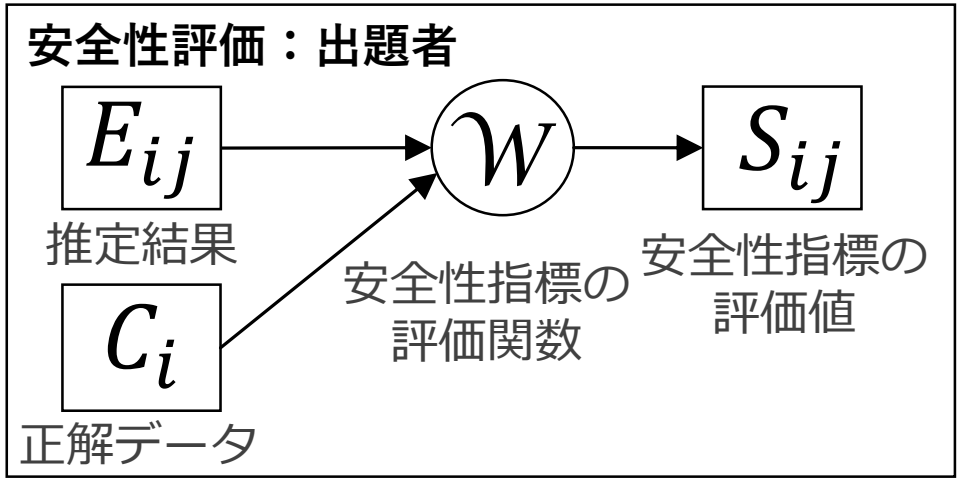
| | | |
|---|---|---|
| 37 | Male | <=50K |
| 44 | Female | >50K |
| 28 | Female | <=50K |

擬似データ

| | | | 距離 |
|---|---|---|---|
| 43 | Male | <=50K | 6 |
| 39 | Male | >50K | 3 |
| 52 | Female | >50K | 17 |
| … | … | … | |

- 推定結果
  - 攻撃者が出題者に提出
  - 擬似データのどのレコードから匿名化データを生成したか推定
  - 擬似データの行番号100個（0スタートなので注意！）
- TIPS(?)
  - 匿名化データとサンプリングデータは、ヒストグラム、分散共分散行列、決定木分析の分類結果が類似（閾値以上を満たす） → 匿名化データとヒストグラム、分散共分散行列、決定木分析の分類結果が類似（閾値以上を満たす）するようなサンプリングデータの候補を疑似データの中から見つける

# 安全性指標：出題者のタスク




- 推定結果
  - 攻撃者が出題者に提出した、疑似データのどのレコードから匿名化データを生成したか推定した擬似データの行番号100個（0スタートなので注意！）
- 正解データ
  - 出題者が加工者毎に渡したサンプリングデータ
  - 実際には擬似データのどの行をサンプリングしたか分かる行番号のみでよい（行番号は0スタートなので注意！）
- 安全性指標の評価関数
  - 推定結果（100行）に対する、正解データ（1万行）との一致数を安全性指標の評価値として返す
    - すなわち安全性指標の評価値は最高100点、最低0点となる
  - **実装コード：privacymeasure.py** → お試し用に公開

# 出題者が準備するデータやコード等のまとめ

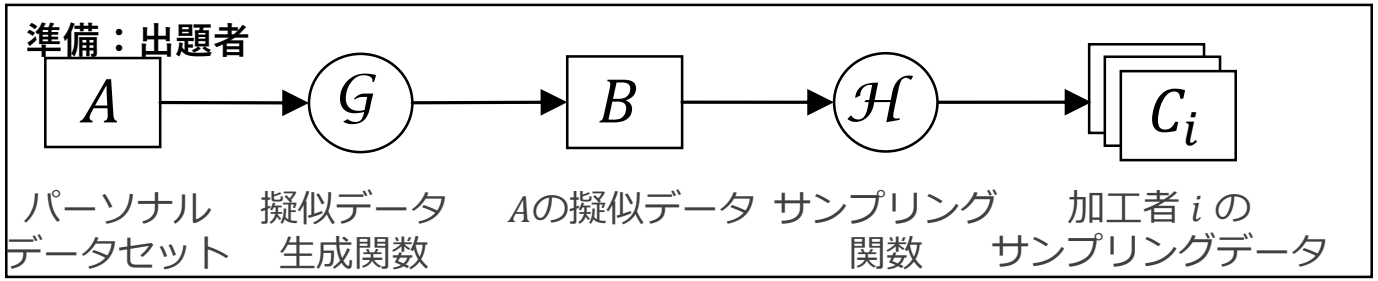
- 加工者に提供するデータ・コード
  - サンプリングデータ（加工者毎に異なる）
  - 匿名化サンプルコード1 synthetic.py
  - 匿名化サンプルコード2 rr.py
  - 匿名化サンプルコード3 kanony.py
  - 有用性評価関数 utilityfunc.py

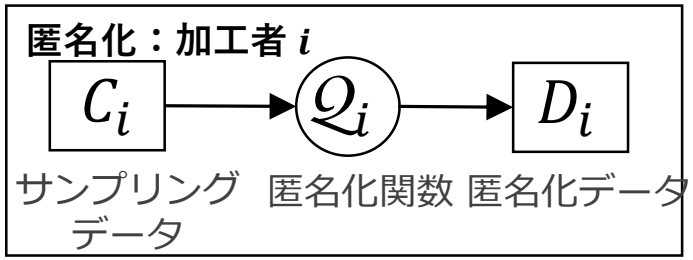
匿名化データを作成して出題者に提出

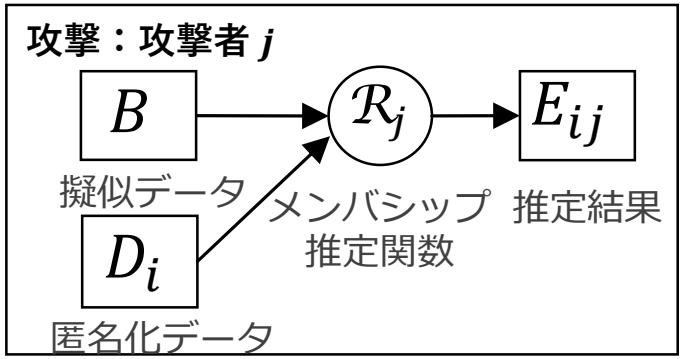
- 攻撃者に提供するデータ・コード
  - 擬似データ
  - 各加工者の匿名化データ
  - 攻撃サンプルコード attack.py

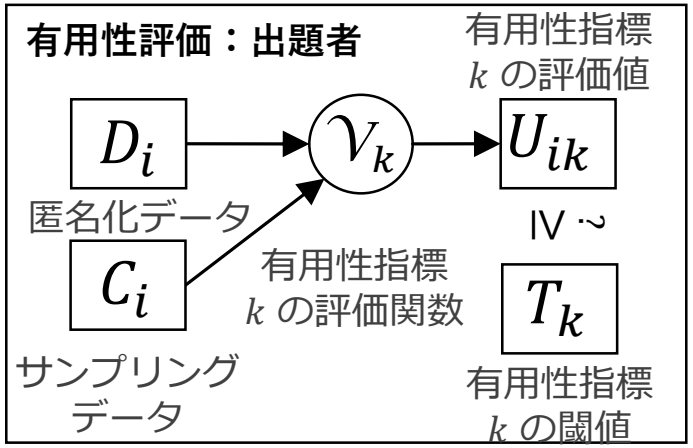
攻撃結果(推定結果)を出題者に提出

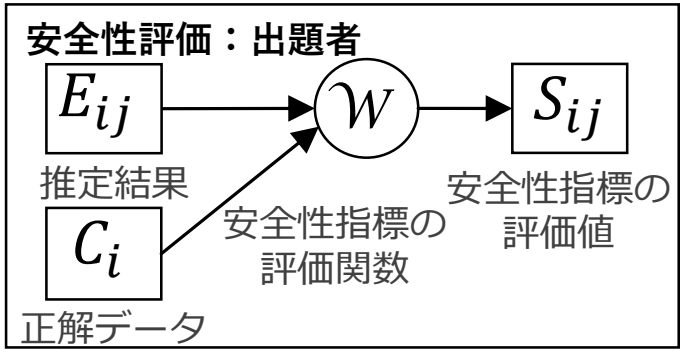
- お試し用データ・コード
  - Census Income Data Set
  - お試し用擬似データ
  - お試し用サンプリングデータ
  - お試し用匿名化データ
  - お試し用正解データ
  - 擬似データ生成関数 gen.py
  - サンプリング関数 randomsampling.py
  - 安全性評価関数 privacymeasure.py
- 有用性指標の閾値
  - ヒストグラム：0.99
  - 分散共分散行列：0.4
  - 決定木分析：0.85



# 出題者が準備/取得するデータやコードの形式

・文字コード: UTF-8
・pre/main: 予備戦=pre, 本戦=main

| 分類 | 説明 | ファイル名 | 形式/引数 | 備考 |
|---|---|---|---|---|
| 出題者が保持するデータ | サンプリングした行番号 | [pre/main]_answer_[加工者番号].index | 1列100行のindexファイル | • 行番号は0スタート |
| 加工者に提供するデータ・コード | サンプリングデータ | [pre/main]_samplingdata_[加工者番号].csv | 9列1万行のcsvファイル | • 加工者番号: 01～99 |
| | 匿名化サンプルコード1 | synthetic.py | readme.txt 参照 | |
| | 匿名化サンプルコード2 | rr.py | readme.txt 参照 | |
| | 匿名化サンプルコード3 | rrp.py | readme.txt 参照 | |
| | 匿名化サンプルコード4 | kanony.py | readme.txt 参照 | |
| | 有用性評価関数 | utilityfunc.py | readme.txt 参照 | |
| | 決定木分析テストデータ | test.csv | 9列15,060行のcsvファイル | |
| 加工者から取得するデータ | 匿名化データ | [pre/main]_anonymizeddata_[加工者番号].csv | 9列1,000～10万行のcsvファイル | |
| 攻撃者に提供するデータ・コード | 擬似データ | [pre/main]_syntheticdata.csv | 9列10万行のcsvファイル | |
| | 各加工者の匿名化データ | [pre/main]_anonymizeddata_[加工者番号].csv | 9列1,000～10万行のcsvファイル | |
| | 攻撃サンプルコード | attack.py | readme.txt 参照 | |
| 攻撃者から取得するデータ | 推定結果データ | inference_[加工者番号]_[攻撃者番号].index | 1列100行のindexファイル | |
| お試し用データ・コード | Census Income Data Set | census_income.data.csv | 9列30,1621行のcsvファイル | |
| | お試し用擬似データ | trial_syntheticdata.csv | 9列10万行のcsvファイル | |
| | お試し用サンプリングデータ | trial_samplingdata.csv | 9列1万行のcsvファイル | |
| | お試し用正解データ | trial_answer.index | 1列1万行のindexファイル | |
| | お試し用匿名化データ | trial_anonymizeddata.csv | 9列1万行のcsvファイル | |
| | 擬似データ生成関数 | gen.py | readme.txt 参照 | |
| | サンプリング関数 | randomsampling.py | readme.txt 参照 | • ランダムサンプリング |
| | 安全性評価関数 | privacymeasure.py | readme.txt 参照 | |

あくまでお試し用であって、これらを使って匿名化や攻撃を行わなければいけないものではありません

# アンケート

- 予備戦/本戦の匿名化フェーズ終了後、加工者は以下のアンケートにご回答ください
  - どのような匿名化手法を用いたか、攻撃者に通知する予定です
  - 匿名化データの利用者の立場からすると、どのような匿名化手法を用いたかは必要な情報とおもわれるため、このようなルールとしました

| 安全性指標 | |
|---|---|
| 差分プライバシー | $\epsilon, \delta$の値： |
| *k*-匿名性 | *k* の値： |
| *Pk*-匿名性 | *k* の値： |
| $\delta$-存在性 | $\delta$ の値： |
| その他： | ?? の値： |

| 加工方法 | |
|---|---|
| レコード削除 | 適用したレコード数(おおよそ)： |
| トップ(ボトム)コーディング | 適用した属性番号(１８)<br>適用したレコード数(おおよそ)： |
| レコード一部抽出 | 適用したレコード数(おおよそ)： |
| ミクロアグリゲーション | 適用した属性番号(１８)<br>適用したレコード数(おおよそ)： |
| データ交換(スワッピング) | 適用した属性番号(１２３４５６７８９)<br>適用したレコード数(おおよそ)： |
| ノイズ(誤差)付加 | 適用した属性番号(１２３４５６７８９)<br>適用したレコード数(おおよそ)： |
| 疑似データ生成 | 適用した属性番号(１２３４５６７８９)<br>適用したレコード数(おおよそ)： |
| ダミーレコード追加 | 適用したレコード数(おおよそ)： |
| ランダム化/Randomized Response/PRAM | 適用した属性番号(１２３４５６７８９)<br>適用したレコード数(おおよそ)： |
| その他： | 適用した属性番号(１２３４５６７８９)<br>適用したレコード数(おおよそ)： |